FUNDAMENTALS AND APPLICATIONS OF AI: AN INTERDISCIPLINARY PERSPECTIVE

EDITED BY: Víctor M. Eguíluz, Claudio Mirasso and Raul Vicente
PUBLISHED IN: Frontiers in Physics, Frontiers in Computer Science and Frontiers in Artificial Intelligence

frontiers Research Topics

frontiers

ISSN 1664-8714
ISBN 978-2-88966-531-0
DOI 10.3389/978-2-88966-531-0

About Frontiers

Frontiers is more than just an open-access publisher of scholarly articles: it is a pioneering approach to the world of academia, radically improving the way scholarly research is managed. The grand vision of Frontiers is a world where all people have an equal opportunity to seek, share and generate knowledge. Frontiers provides immediate and permanent online open access to all its publications, but this alone is not enough to realize our grand goals.

Frontiers Journal Series

The Frontiers Journal Series is a multi-tier and interdisciplinary set of open-access, online journals, promising a paradigm shift from the current review, selection and dissemination processes in academic publishing. All Frontiers journals are driven by researchers for researchers; therefore, they constitute a service to the scholarly community. At the same time, the Frontiers Journal Series operates on a revolutionary invention, the tiered publishing system, initially addressing specific communities of scholars, and gradually climbing up to broader public understanding, thus serving the interests of the lay society, too.

Dedication to Quality

Each Frontiers article is a landmark of the highest quality, thanks to genuinely collaborative interactions between authors and review editors, who include some of the world's best academicians. Research must be certified by peers before entering a stream of knowledge that may eventually reach the public - and shape society; therefore, Frontiers only applies the most rigorous and unbiased reviews.
Frontiers revolutionizes research publishing by freely delivering the most outstanding research, evaluated with no bias from both the academic and social point of view. By applying the most advanced information technologies, Frontiers is catapulting scholarly publishing into a new generation.

What are Frontiers Research Topics?

Frontiers Research Topics are very popular trademarks of the Frontiers Journals Series: they are collections of at least ten articles, all centered on a particular subject. With their unique mix of varied contributions from Original Research to Review Articles, Frontiers Research Topics unify the most influential researchers, the latest key findings and historical advances in a hot research area! Find out more on how to host your own Frontiers Research Topic or contribute to one as an author by contacting the Frontiers Editorial Office: frontiersin.org/about/contact

FUNDAMENTALS AND APPLICATIONS OF AI: AN INTERDISCIPLINARY PERSPECTIVE

Topic Editors:
Víctor M. Eguíluz, Institute of Interdisciplinary Physics and Complex Systems (IFISC), Spain
Claudio Mirasso, Institute of Interdisciplinary Physics and Complex Systems (IFISC), Spain
Raul Vicente, Max Planck Institute for Brain Research, Germany

Citation: Eguíluz, V. M., Mirasso, C., Vicente, R., eds. (2021). Fundamentals and Applications of AI: An Interdisciplinary Perspective. Lausanne: Frontiers Media SA. doi: 10.3389/978-2-88966-531-0

Table of Contents

frontiers in Physics

EDITORIAL
published: 19 January 2021
doi: 10.3389/fphy.2020.633494

Editorial: Fundamentals and Applications of AI: An Interdisciplinary Perspective

Víctor M. Eguíluz[1]*, *Claudio R. Mirasso*[1] *and Raúl Vicente*[2]

[1]Instituto de Física Interdisciplinary Sistemas Complejos IFISC (CSIC-UIB), Palma de Mallorca, Spain, [2]Institute of Computer Science, University of Tartu, Tartu, Estonia

Keywords: artificial intelligence, applications, fundamentals, multidisciplinary and interdisciplinary approach, machine learning

Editorial on the Research Topic

Fundamentals and Applications of AI: An Interdisciplinary Perspective

Machines, computers, and algorithms appear recurrently in the future imagined in science fiction pieces. Terminator's Skynet, Space Odyssey's HAL 9000, Psychohistory of Asimov's Foundation, and Westworld's Dolores are just a few examples of our collective imaginary of a (days) topic future. Interestingly, the year 2019 has represented the future in some works. Toronto Star in 1983 asked Asimov to predict the future, the world of 2019. This is also the case for Blade Runner, where the action runs in a dystopic LA, in 2019, with replicants having "almost" human cognitive capabilities. Although we have not reached most of the utopian pictures, the growth of Big Data and Artificial Intelligence algorithms is unquestionable (**Figure 1**). Thus, celebrating the unstoppable advance of AI, we collect in this RT several studies addressing fundaments and applications from a physics perspective.

In 2020, AI has continued its penetration into classical fields and emerging technologies. Fueled by deep learning and the automatic generation of data, the techniques developed in AI are being applied to predict and control physical, biological, engineering, and even commercial systems. Given the two-way interaction between AI and different fields and including how these fields inspire novel methods and theory in AI, we had envisioned a volume illustrating such an interdisciplinary perspective. Contributions include quantum annealers and quantum neural networks, echo state networks, machine learning (reinforcement learning and graph-based methods), and applications to optimization, classification of heartbeats, animal collective movement, and climate forecast, and the use of AI to discover physical laws.

A fast machine learning model for ECG-based heartbeat classification and arrhythmia detection was developed, based on *echo state networks* [1]. The classifier requires a small number of features and a single ECG signal suffices. The possibility of using a combination of ensembles allows them to exploit parallelism to train the classifier with remarkable speed. The sensitivity and predictive values are comparable with those of the state of the art in fully automatic ECG classifiers and even outperform other ECG classifiers that follow more complex feature selection approaches.

Reservoir computers are investigated in two contributions. First, a coherent all-optical fiber-ring reservoir computer with distributed Kerr nonlinearity is investigated numerically and experimentally [2]. The system is based on a passive coherent optical fiber-ring cavity where part of the nonlinearity is due to the Kerr effect. They compare the nonlinear transformations of information in the reservoir's input layer, the reservoir itself, and the readout layer. They find that the Kerr effect enhances the computational capability of the reservoir, in particular, its nonlinear computational capacity. Second, the trade-off

OPEN ACCESS

Edited and reviewed by:
Alex Hansen,
Norwegian University of Science and Technology, Norway

***Correspondence:**
Víctor M. Eguíluz
victor@ifisc.uib-csic.es

Specialty section:
This article was submitted to Interdisciplinary Physics, a section of the journal Frontiers in Physics

Received: *25 November 2020*
Accepted: *30 November 2020*
Published: *19 January 2021*

Citation:
Eguíluz VM, Mirasso CR and Vicente R (2021) Editorial: Fundamentals and Applications of AI: An Interdisciplinary Perspective. Front. Phys. 8:633494. doi: 10.3389/fphy.2020.633494

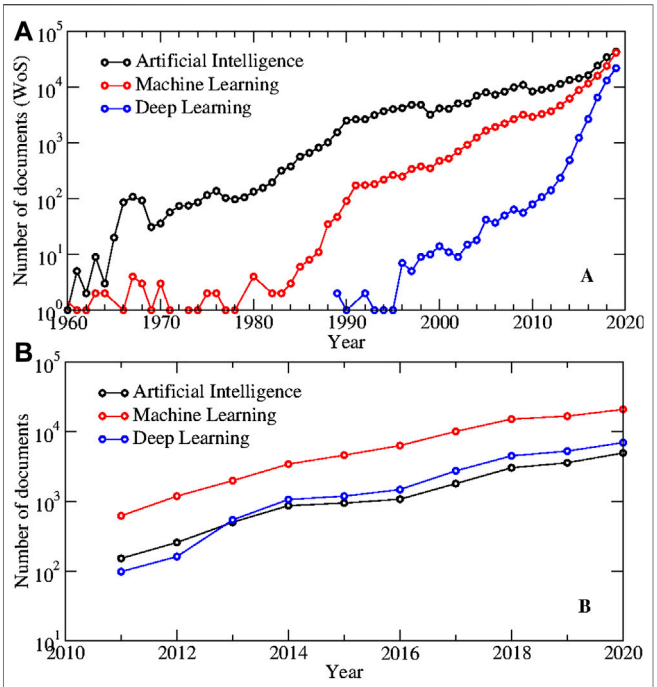


FIGURE 1 | Growth of the number of documents related to artificial intelligence in **(A)** the WoS and **(B)** Dimensions. **(A)** Between 1990 and 2010, the growth seems exponential with a doubling time of 8 years for artificial intelligence and 3.5 years for machine learning. In more recent years, the growth is even faster. Number of documents obtained in the Web of Science with the terms "artificial intelligence," "machine learning," and "deep learning." **(B)** The number of documents identified in any Frontiers journal also shows an exponential growth with a doubling time smaller than 2 years. Number of documents obtained in Dimensions with the terms "artificial intelligence," "machine learning," and "deep learning" and filtering by Frontiers journals.

between information processing capacity and rate is analyzed in Ref. [3] using a delay-based reservoir computer. Delay-based reservoir computers have a trade-off between computational capacity and processing speed due to the nonzero response time of the nonlinear node. They find that the computational capacity degrades for a sampling output rate that is higher than the inverse of the response time of the system. Moreover, the computational capacity also depends on the misalignment between the delay time of the nonlinear node and the data injection time.

In the realm of biology, the collective behavior of animals is a fascinating field aiming to understand how patterns of coordination among a large number of individuals emerge as a result of local interactions. Inferring the rules that give rise to a certain behavior is a challenging problem as any occasional observer of flocks and fish schools can attest. Ref. [4] uses the framework of reinforcement learning, machine learning techniques in which an agent acts in an environment and learns to maximize a reward signal, to model the problem of automatically discovering local rules of interaction that would lead to a desired collective behavior. To that end, Costa et al. apply evolutionary strategies to optimize a single policy (mapping from sensory inputs to actions) followed by the agents so that a desired collective behavior would emerge.

The potential of graph-based methods in combination with machine learning algorithms is addressed in Ref. [5]. The authors explore two methods to detect outliers, with applications to high-dimensional datasets. The first method measures the fragmentation of a graph, where the data samples are the nodes of the graph, while the second method is based on the Isomap algorithm, a dimensionality reduction technique. The performance is compared with alternative methods and assessed on the dependence on the size of an anomalous region within an image, a known problem in anomaly and outlier detection.

An example of large-scale coordination occurs in the climate and ocean circulation systems. Predicting temporal patterns such as El Nino oscillation is a problem of considerable practical and fundamental interest. Ref. [6] reviews different machine learning techniques to predict El Niño events for lead times larger than 12 months and studies which type of attributes is most relevant for an accurate forecast. The review focuses on feed-forward artificial neural networks from early work back in the late 90s to the more recent graph-based methods.

Quantum approaches are explored in three contributions. Refs. [7 and 8] implement quantum annealers to solve quadratic unconstrained binary optimization (QUBO) in two applications: an industrial problem, the control of automated guided vehicles in a factory [7], and the problem of item listing optimization for e-commerce [8]. The contributions probe the capacity of quantum annealers to address industrial and commercial problems stimulating further research of quantum annealers to solve optimization problems in real-world systems. Ref. [9] addresses the no-cloning theorem–the impossibility to duplicate a quantum state—and how to circumvent this limitation for applying quantum computing. In particular, they obtain lower bounds of input redundancy, that is, how many times the data must be reintroduced in parameterized quantum circuits (PQCs) (also referred to as quantum neural networks or variational quantum circuits). This contribution analyzes two different functions for the encoding (linear encoding and arcsin encoding) and proves that lower bounds are logarithmic in terms of a linear algebraic complexity measure of the target function.

A different approach is the implementation of internal processing capacity in models of complex systems, in particular, agent-based models. Ref. [10] introduces an agent-based model with cognitive and social dynamics, the Bayesian word learning model, to study the effects of cognitive and social dynamics on the emergence of linguistic consensus in the naming game. In the game, agents learn new words by generalization, using Bayes statistics, from previous experience. The novelty of the approach, with agents modeled after a Bayesian inference framework, *captures important properties of human behavior and learning* and opens the possibility to study applications to language, semiotics cognitive science, and complex systems.

Finally, a research line that is attracting interest is the development of an automated scientist, to extract interpretable dynamics and laws directly from data. In this direction, Ref. 11 uses machine learning and data-driven approaches, the sparse identification of nonlinear dynamics algorithm, to model the motion of falling objects. One conclusion is that blindly applying machine learning techniques can be inappropriate without using domain-specific knowledge. Additionally, the contribution addresses interesting issues related

to model "discoverability" and "interpretability," important concepts that we are only starting to understand, and that can play an important role in the future. Indeed, AI is already being used to learn models from quantum mechanics to statistical physics, and this trend can only be expected to grow.

AUTHOR CONTRIBUTIONS

All authors contributed to the writing of the editorial.

ACKNOWLEDGMENTS

We acknowledge funding by Agencia Estatal de Investigación (AEI) and Fondo Europeo de Desarrollo Regional (FEDER) through project SPASIMM FIS2016-80067-P (AEI/FEDER, UE), the Spanish State Research Agency through the María de Maeztu Program for Units of Excellence in R&D (MDM-2017-0711 to IFISC). We also acknowledge support from Frontiers Editorial Office, in special to Marta Brucka who obtained the data for **Figure 1B**.

REFERENCES

1. Alfaras M, Soriano MC, Ortín S. A fast machine learning model for ECG-based heartbeat classification and Arrhythmia detection. *Front Phys* (2019) 7:103. doi:10.3389/fphy.2019.00103
2. Pauwels J, Verschaffelt G, Massar S, Van der Sande G. Distributed Kerr non-linearity in a coherent all-optical fiber-ring reservoir computer. *Front Phys* (2019) 7:138. doi:10.3389/fphy.2019.00138
3. Ortín S, Pesquera L. Tackling the trade-off between information processing capacity and rate in delay-based reservoir computers. *Front Phys* (2019) 7:210. doi:10.3389/fphy.2019.00210
4. Costa T, Laan A, Heras FJH, de Polavieja GG. Automated discovery of local rules for desired collective-level behavior through reinforcement learning. *Front Phys* (2020) 8:200. doi:10.3389/fphy.2020.00200
5. Amil P, Almeira N, Masoller C. Outlier mining methods based on graph structure analysis. *Front Phys* (2019) 7:194. doi:10.3389/fphy.2019.00194
6. Dijkstra HA, Petersik P, Hernández-García E, López C. The application of machine learning techniques to improve El Niño prediction skill. *Front Phys* (2019) 7:153. doi:10.3389/fphy.2019.00153
7. Ohzeki M, Miki A, Miyama MJ, Terabe M. Control of automated guided vehicles without collision by quantum annealer and digital devices. *Front Comput Sci* (2019) 1:9. doi:10.3389/fcomp.2019.00009
8. Nishimura N, Tanahashi K, Suganuma K, Miyama MJ, Ohzeki M. Item listing optimization for E-commerce websites based on diversity. *Front Comput Sci* (2019) 1:2. doi:10.3389/fcomp.2019.00002
9. Gil Vidal FJ, Theis DO. Input redundancy for parameterized quantum circuits. *Front Phys* (2020) 8:297. doi:10.3389/fphy.2020.00297
10. Marchetti G, Patriarca M, Heinsalu E. A Bayesian approach to the naming game model. *Front Phys* (2020) 8:10. doi:10.3389/fphy.2020.00010
11. de Silva BM, Higdon DM, Brunton SL, Kutz JN. Discovery of physics from data: universal laws and discrepancies. *Front Artif Intell* (2020) 3:25. doi:10.3389/frai.2020.00025

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Computer Science

ORIGINAL RESEARCH
published: 16 July 2019
doi: 10.3389/fcomp.2019.00002

Item Listing Optimization for E-Commerce Websites Based on Diversity

Naoki Nishimura[1,2]*, *Kotaro Tanahashi*[2], *Koji Suganuma*[1,2], *Masamichi J. Miyama*[3,4] and *Masayuki Ohzeki*[3,4,5]

[1] Internet Business Development Division, Recruit Lifestyle Co., Ltd., Tokyo, Japan, [2] ICT Solution Department, Recruit Communications Co., Ltd., Tokyo, Japan, [3] Graduate School of Information Sciences, Tohoku University, Sendai, Japan, [4] Jij Inc., Tokyo, Japan, [5] Institute of Innovative Research, Tokyo Institute of Technology, Yokohama, Japan

For e-commerce websites, deciding the manner in which items are listed on webpages is an important issue because it can dramatically affect item sales. One of the simplest strategies for listing items to improve the overall sales is to do so in a descending order of popularity representing sales or sales numbers aggregated over a recent period. However, in lists generated using this strategy, items with high similarity are often placed consecutively. In other words, the generated item list might be biased toward a specific preference. Therefore, this study employs penalties for items with high similarity being placed next to each other in the list and transforms the item listing problem to a quadratic assignment problem (QAP). The QAP is well-known as an NP-hard problem that cannot be solved in polynomial time. To solve the QAP, we employ quantum annealing, which exploits the quantum tunneling effect to efficiently solve an optimization problem. In addition, we propose a problem decomposition method based on the structure of the item listing problem because the quantum annealer we use (i.e., D-Wave 2000Q) has a limited number of quantum bits. Our experimental results indicate that we can create an item list that considers both popularity and diversity. In addition, we observe that using the problem decomposition method based on a problem structure can provide to a better solution with the quantum annealer in comparison with the existing problem decomposition method.

Keywords: item listing, e-commerce, quadratic assignment problem, quantum annealing, D-Wave, problem decomposition

OPEN ACCESS

Edited by:
Víctor M. Eguíluz, Institute of Interdisciplinary Physics and Complex Systems (IFISC), Spain

Reviewed by:
Jonas Maziero, Universidade Federal de Santa Maria, Brazil
Hsi-Sheng Goan, National Taiwan University, Taiwan

***Correspondence:**
Naoki Nishimura
nishimura.n.ab@gmail.com

Specialty section:
This article was submitted to Quantum Computing, a section of the journal Frontiers in Computer Science

Received: *27 March 2019*
Accepted: *01 July 2019*
Published: *16 July 2019*

Citation:
Nishimura N, Tanahashi K, Suganuma K, Miyama MJ and Ohzeki M (2019) Item Listing Optimization for E-Commerce Websites Based on Diversity. Front. Comput. Sci. 1:2. doi: 10.3389/fcomp.2019.00002

1. INTRODUCTION

Several companies have recently started operating e-commerce websites to sell their items and services to the public considering the widespread use of the internet. For these companies, deciding on the order in which items are listed on their website's pages is important because this ordering has the potential to dramatically affect the sales of their items or services. **Figure 1** shows a snapshot of a hotel reservation website. This is an example of the items being listed on an e-commerce page. On this website, hotels at different locations are listed in the order of popularity calculated based on various indicators from top to bottom. These sorted items for display on webpages are collectively referred to as an item list.

To improve sales on e-commerce websites, placing items in the descending order of popularity representing sales or sales numbers aggregated over a recent period is a simple strategy for determining the list order of items (Long and Chang, 2014). In addition, the popularity of an item can be estimated by it is placing it at different positions in the item list and determining the position of each product to maximize the total popularity estimate. In particular, if p_{ij} is the estimated popularity of an item $i \in I$ when it is placed in a position $j \in J$, then the total popularity of all items can be maximized by solving the following integer programming problem (Wang et al., 2016):

$$\begin{aligned} &\text{maximize} && \sum_{i\in I}\sum_{j\in J} p_{ij}x_{ij} \\ &\text{subject to} && \sum_{i\in I} x_{ij} = 1, \quad j \in J, \\ & && \sum_{j\in J} x_{ij} = 1, \quad i \in I, \\ & && x_{ij} \in \{0,1\}, \quad i \in I, \ j \in J. \end{aligned} \tag{1}$$

where x_{ij} is a binary variable that indicates whether or not to assign item i to position j. The abovementioned constraints ensure that only one item is allocated to each position, and only one position is allocated to each item. In this study,

$$P(\mathbf{x}) = \sum_{i\in I}\sum_{j\in J} p_{ij}x_{ij}$$

is referred to as the popularity term for $\mathbf{x} = (x_{11}, x_{12}, \cdots)$. This problem can be interpreted as a network flow problem, and an efficient technique to solve such a problem in polynomial time exists. Furthermore, the solution obtained by solving this network flow problem with $x_{ij} \in [0,1]$ coincides with the solution of the abovementioned integer programming problem (Vazirani, 2013).

However, in the case of the list of items generated using such a strategy, the relationship between the different objects is ignored because the popularity of each item p_{ij} is considered independently. For example, let us assume that customers visit an e-commerce website and browse the page of a particular item group. If the relationships among different items are not considered while placing items in an item list, several items with high similarities can possibly be placed close to each other, thereby reducing the value of the item list for customers in terms of item diversity. Considering this, several attempts have been made to include item diversity in item recommendation lists for users to ensure that they find the recommendation lists useful (Adomavicius and Kwon, 2011; Antikacioglu and Ravi, 2017). In these previous studies, the measures of diversity in the item recommendation lists for customers were improved by solving the maximum matching problem of the bipartite graph obtained after Top-N recommendation.

In this study, we introduce diversity into the item list for the entire user base, as well as methods for improving the usefulness of recommendation lists. An item list is generated by solving an optimization problem that imposes a penalty when items with high similarity are placed in adjacent to each other. Considering both popularity and diversity, the item list generation problem can be formulated as a quadratic assignment problem (QAP) as detailed below.

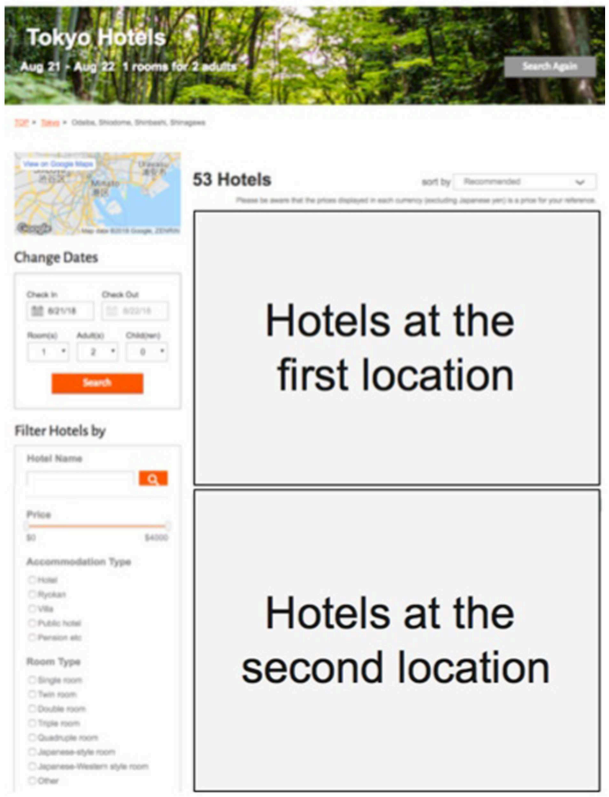


FIGURE 1 | An example of an item list on a hotel reservation website.

We employ quantum annealing (QA) herein to solve the QAP (Kadowaki and Nishimori, 1998). An optimization problem formulated with discrete variables can be efficiently solved using the Ising model or a quadratic unconstrained binary optimization problem (QUBO) because of the introduction of the quantum tunneling effect by QA. Currently, the protocol of QA is artificially realized in an actual quantum device known as a quantum annealer (Berkley et al., 2010; Harris et al., 2010; Johnson et al., 2010; Bunyk et al., 2014). The quantum annealer has been tested for numerous applications, including portfolio optimization (Rosenberg et al., 2016), protein folding simulation (Perdomo-Ortiz et al., 2012), online advertisement allocation optimization (Tanahashi et al., 2019), molecular similarity problem (Hernandez and Aramon, 2017), computational biology (Li et al., 2018), job-shop scheduling (Venturelli et al., 2015), traffic optimization (Neukart et al., 2017), election forecasting (Henderson et al., 2018), machine learning (Crawford et al., 2016; Khoshaman et al., 2018; Neukart et al., 2018), and for automated guided vehicles in plants (Ohzeki et al., 2019). In addition, several other studies have been conducted to efficiently solve various problems using the quantum annealer (Arai et al., 2018; Ohzeki et al., 2018a,b; Takahashi et al., 2018; Okada et al., 2019).

In particular, our problem can be solved using the quantum annealer by formulating our QAP as a QUBO. However, a QUBO for such a large number of items cannot be directly solved in one instance with the current state-of-the-art quantum annealer, namely D-Wave 2000Q because it employs the chimera graph. The physical qubits available on D-Wave 2000Q are less than 2048 because the qubits might have defects. In addition, the connection between the physical qubits is sparse and limited on the chimera graph. Thus, several embedding techniques have been proposed; however, the number of logical qubits available to represent the optimization problems to be solved is drastically reduced (Boothby et al., 2016). To address this issue, a heuristic method has been proposed to solve a large-sized problem using a limited number of hardware bits. D-Wave Systems, which is the manufacturer of D-Wave 2000Q, has developed an open-source software `qbsolv` (Booth et al., 2017) that solves a large-sized problem by dividing it into small subproblems. However, the decomposed QAP might not necessarily lead to feasible solutions because `qbsolv` selects the subset of variables in the order of the energy impact of each variable for division of a problem. Furthermore, in the literature (Okada et al., 2019), the division of an original problem into subproblems based on its structure is a promising method to efficiently solve large-sized problems using D-Wave 2000Q. Considering this, we propose herein a method to obtain better objective values for the QAP problem compared to those provided by the existing method (Booth et al., 2017) in the same calculation time by decomposing the problem based on the set of items and positions that they can be assigned to in an item list. In addition, we assess the performance of the proposed method using the actual access log of a hotel reservation website.

The primary contributions of our study are summarized as follows:

- We propose a method of creating item lists on an e-commerce website as a QAP considering the popularity and diversity of the items.
- We convert the QAP to the QUBO to solve the abovementioned problem with D-wave 2000Q.
- We propose a decomposition technique exploiting the structure of the item list.

2. MODELS

2.1. Formulating the Item Listing Optimization Problem as a QAP

We introduce the diversity term in our proposed model to add diversity in the item list. In particular, we calculate the similarity $f_{ii'}$ for pairs of items $i, i' \in I$ to introduce diversity in the item list. The diversity of the item list (i.e., the diversity term) is defined as the negative value of the summation of the items' similarity degree $f_{ii'}$ for overall adjacent items:

$$D(\mathbf{x}) = -\sum_{i\in I}\sum_{i'\in I}\sum_{j\in J}\sum_{j'\in J} f_{ii'}d_{jj'}x_{ij}x_{i'j'}.$$

where $d_{jj'}$ is the adjacent flag of the position j and j'; $d_{jj'} = 1$ is for the adjacent positions; and $d_{jj'} = 0$ is for the non-adjacent positions. The value of function $D(\mathbf{x})$ decreases because the high-similarity items are adjacent. We solve a multi-objective optimization problem based on two values: the popularity of individual products $P(\mathbf{x})$ and diversity of the item list $D(\mathbf{x})$. Let w be a parameter used to determine the penalty for listing items with high similarity. This problem is formulated as follows:

$$\begin{aligned} &\text{maximize} && \sum_{i\in I}\sum_{j\in J} p_{ij}x_{ij} - w\sum_{i\in I}\sum_{i'\in I}\sum_{j\in J}\sum_{j'\in J} f_{ii'}d_{jj'}x_{ij}x_{i'j'} \\ &\text{subject to} && \sum_{i\in I} x_{ij} = 1, \quad j \in J, \\ &&& \sum_{j\in J} x_{ij} = 1, \quad i \in I, \\ &&& x_{ij} \in \{0,1\}, \quad i \in I, \ j \in J. \end{aligned} \tag{2}$$

Two methods for calculating the similarity $f_{ii'}$ are introduced: explicit and implicit expressions. In the explicit expression, semantic features, such as product category and average price, can be quantified using the distances between the feature vectors that can be calculated. The smaller the feature distance, the higher the similarity $f_{ii'}$ between the items. In the implicit expression, the higher the co-browsing number of an item (i.e., the number of times that the items were viewed in the same session by the same user on the website), the higher the similarity $f_{ii'}$.

The advantage of the first approach is that the interpretation of the result is straightforward. Also, the semantic features of each item are available when the item is added to the database, that is, so-called cold start problems are avoided. Nevertheless, it suffers from a disadvantage in that appropriate semantic features must be created and quantified. In contrast, the advantage of the second approach is that it involves easy calculations and can consider various information reflecting customer behavior; however, its disadvantage is that semantic interpretation might be difficult. Section 3.1 describes the two methods for the calculation of the similarity $f_{ii'}$ in detail, and section 3.2 compares the item lists created using these measures.

As previously specified, the optimization problem (2) is a QAP. The QAP is well-known as an NP-hard problem that cannot be solved in polynomial time (Anstreicher, 2003; Abdel-Basset et al., 2018).

2.2. Formulating the Item Listing Optimization Problem as a QUBO

We utilize QA to solve our optimization problem. The details of QA pertaining to D-Wave 2000Q are outlined in **Appendix A**. The optimization problem must be expressed in the form of a QUBO to use QA as a solver. QUBO is given as followsi (Lucas, 2014):

$$\begin{aligned} &\text{minimize} && \mathbf{x}^{\mathrm{T}}Q\mathbf{x} \\ &\text{subject to} && \mathbf{x} \in \{0,1\}^N, \end{aligned} \tag{3}$$

where $Q \in \mathbb{R}^{N\times N}$. Thus, our optimization problem can be transformed into a QUBO by employing a penalty function for

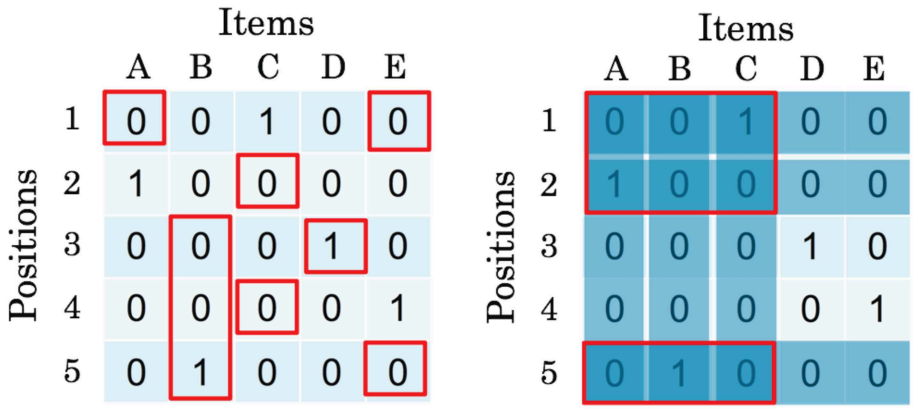


FIGURE 2 | Example of problem decomposition. The red frames represent the variables of the subproblem to be selected. The left figure is a selection of a subproblem without considering the problem structure. A subproblem has no feasible solution. The right figure is a selection of a subproblem based on the logical structure of the problem. A subproblem has feasible solutions.

violating constraints and adding this penalty function to the objective function:

$$\begin{aligned} &\text{minimize } -\sum_{i\in I}\sum_{j\in J} p_{ij}x_{ij} + w\sum_{i\in I}\sum_{i'\in I}\sum_{j\in J}\sum_{j'\in J} f_{ii'}d_{jj'}x_{ij}x_{i'j'} \\ &\qquad +M\left(\sum_{i\in I}\left(\sum_{j\in J}x_{ij}-1\right)^2+\sum_{j\in J}\left(\sum_{i\in I}x_{ij}-1\right)^2\right) \\ &\text{subject to } x_{ij}\in\{0,1\},\ \ i\in I,\ \ j\in J. \end{aligned} \tag{4}$$

where M is a parameter used to prevent the violation of the constraint conditions. This is ensured by setting an appropriate value for M. In theory, M should take an extremely large value. However, we cannot set M to such a large value because of the limitations of the current version of the quantum annealer used (i.e., D-Wave 2000Q). Thus, for simplicity, we present M to the size of the largest element of the absolute value of Q in (3).

2.3. Decomposition Methods for Item Listing Problems

`qbsolv` is a software tool released by D-Wave Systems that enables solving a QUBO larger than one that can be processed using D-Wave 2000Q (D-Wave Systems Inc., 2017). `qbsolv` is essentially a decomposing solver that divides a large problem into smaller parts, which can then be solved by D-Wave 2000Q. Thus, when a large QUBO is inputted, `qbsolv` divides the problem and sends each part of the problem independently to D-Wave 2000Q for calculation to obtain partial solutions. This process is repeated by selecting different parts of the problem using the tabu search until solution improvement stops. See Booth et al. (2017) for the detailed algorithm of `qbsolv`. Furthermore, `qbsolv` selects the subset of variables in the order of the energy impact of each variable for division of a problem. However, in some cases, no feasible solution can be obtained when the target variables are extracted, regardless of the structure of the original problem.

Therefore, we focus herein on the structure of the assignment problem and propose a method to extract problems with feasible solutions. Particularly in the case of an assignment problem, one condition involves each item being necessarily assigned to one position and another condition, in which each position is necessarily assigned to one item. Therefore, while dividing the problem, we have to select variables with candidate combinations of items and positions that are already assigned. **Figure 2** shows an example of the decomposition.

The original problem can be decomposed as follows if the number of items in the original problem is N_{org} and the number of items solved by a partial problem is N_{sub}:

1. Let $\mathcal{N}_s$ be the set of N_{sub} items extracted from N_{org} items.
2. Let $\mathcal{P}_s$ be the set of positions of items of $\mathcal{N}_s$.
3. Let $\mathcal{N}_s \times \mathcal{P}_s$ be the variables of the decomposed problem.

This procedure involves ${}_{N_{\text{org}}^2}C_{N_{\text{sub}}^2}$ candidates for variable combinations in the selection of the subproblems; however, the number of solution candidates can be reduced to ${}_{N_{\text{org}}}C_{N_{\text{sub}}}$ exploiting the structure of the item list.

In practice, it is most important to determine the order in which items are listed in the upper positions of the item list because they are the items that are browsed most often. Therefore, it is effective to solve the entire list as an integer programming problem as in Problem (1) first, then only resolve the particularly important upper positions of the list using the QAP (2).

3. RESULTS AND DISCUSSION

3.1. Experimental Setup

For our experiments, we used the actual access log data of the online hotel reservation site Jalan[1]. On this e-commerce website, a hotel list is created daily based on each area in Japan and the number of guests, including adults and children, that the hotels can accommodate in their rooms. The access

[1]https://www.jalan.net/en/japan_hotels_ryokan/

log includes the date and the time the customer accessed the item list screen, position of each item when the item list screen was accessed, and information on the hotel at which the customer made a reservation. We estimated the popularity p_{ij} and similarity $f_{ii'}$ for the top 10 accessed areas on the hotel reservation website using the access log for the past 6 months. The similarity $f_{ii'}$ was estimated using two methods: the co-browsing similarity and the semantic similarity. The co-browsing similarity was estimated using the log of the co-browsed items in the same customer's session. On the other hand, semantic similarity was created and quantified by area and type of hotel. In our experiments, we used the co-browsing similarity, except for the comparison of the similarity measure in **Figure 7**. Various methods, such as machine learning algorithm (e.g., deep learning or gradient boosting) can be considered for estimating p_{ij} and $f_{ii'}$. Furthermore, p_{ij} and $f_{ii'}$ were normalized such that their average was 0 and the standard deviation was 1 for each item list.

We conducted two experiments in this study:

- evaluating the effect of the diversity term by comparison of solutions when the diversity control parameter is changed for the QAP (2), and
- evaluating the performance of problem decomposition by comparison of the objective values when the structure of the item list is considered for the `qbsolv` problem decomposition.

As previously specified, we used D-Wave 2000Q (DW_2000Q_VFYC_2) for our experiments. Coupler strengths mapping logical to physical couplers with two physical couplers connecting each pair of logical qubits were set as 3.0. **Table 1** lists the values set for the parameters of D-Wave 2000Q and `qbsolv`. `num_read`, `annealing_time`, and `repeats` represent the number of requests for problems, time per annealing, and number of times the main loop of the algorithm is repeated with no change in the optimal value before stopping, respectively (Booth et al., 2017; D-Wave Systems Inc., 2017).

TABLE 1 | Parameters used for solving the problem in our experiments.

Parameter	Value
`num_reads` (D-Wave Systems Inc., 2017)	1,000
`annealing_time` (D-Wave Systems Inc., 2017)	20 [μs]
`auto_scale` (D-Wave Systems Inc., 2017)	True
`postprocess` (D-Wave Systems Inc., 2017)	Optimization
`num_spin_reversal_transforms` (D-Wave Systems Inc., 2017)	4
`timeout` (Booth et al., 2017)	20 [s]
`repeats` (Booth et al., 2017)	5
`subproblemSize` (Booth et al., 2017)	64

3.2. Effect of the Diversity Term

D-Wave 2000Q has less than 2048 qubits because the qubits typically have defects. In addition, as previously specified, the connection between the physical qubits is sparse and limited on the chimera graph, in which D-Wave 2000Q has been based on. Thus, we can consider the problem with eight items per subproblem because complete graph embedding can be applied to arbitrary problem graphs with less than 64 logical variables. Therefore, we first compare the popularity term $P(\mathbf{x})$ and the diversity term $D(\mathbf{x})$ obtained by solving the QAP (2) by changing the diversity control parameter w in the problem when the number of items is 6 and 8. The obtained solution will be the same as that obtained via solving (1) if w is 0.

In **Figures 3**, **4**, the horizontal axes represent the value of w in (2); the right hand side in the figures indicates that the larger the similarity between the similar items listed together, the larger the penalty. In addition, the vertical axes in **Figures 3**, **4** represent the values of $P(\mathbf{x})$ and $D(\mathbf{x})$, respectively. The average value of all solutions is approximately 0 because p_{ij} and $f_{ii'}$ are normalized for each area. **Figures 3B**, **4D** depict plots of $P(\mathbf{x})$ and $D(\mathbf{x})$, respectively, for three typical areas X, Y, and Z from among the 10 areas for the problem with eight items. **Figures 3A**, **4C** are plotted by aggregating the typical values for the 10 areas, wherein

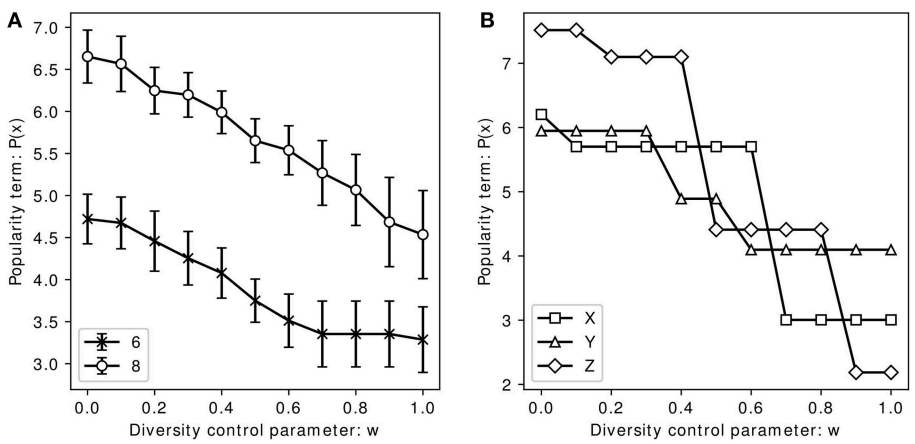


FIGURE 3 | Changes in the popularity term when the diversity control parameter is changed. **(A)** The average popularity term value of 10 areas change when the number of items is changed. **(B)** Changes in the typical popularity term when the number of items is 8.

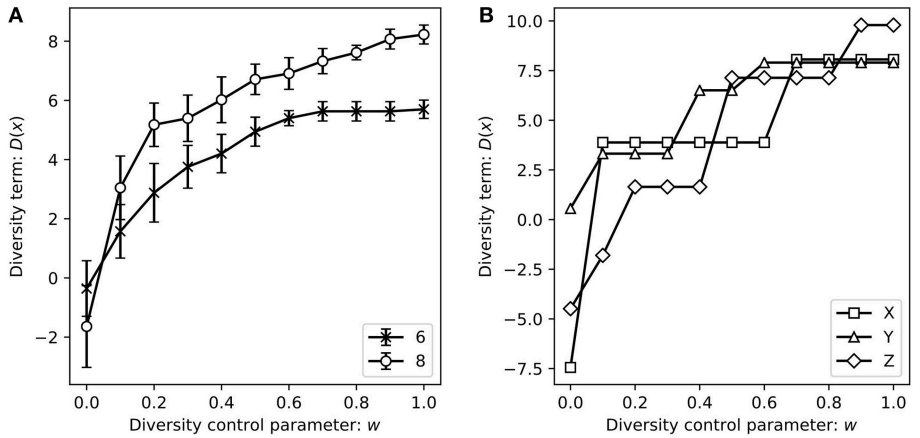


FIGURE 4 | Changes in the diversity term when the diversity control parameter is changed. **(A)** The average diversity term of 10 areas change when the number of items is changed. **(B)** Changes in the typical diversity term when the number of items is 8.

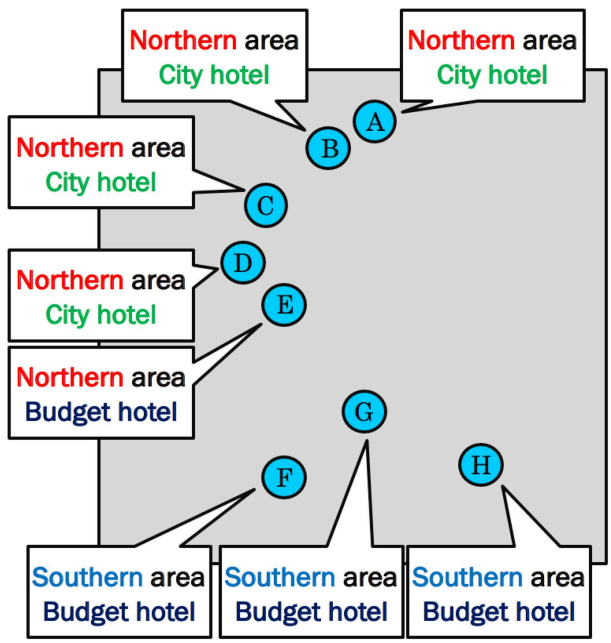


FIGURE 5 | Locations and type of a particular area's hotels.

the marker and the bar represent the average and standard error values, respectively.

Figure 3A shows that the average of the popularity term $P(\mathbf{x})$ is higher than 0 for any w. In addition, $P(\mathbf{x})$ gradually decreases as w increases. In other words, a trade-off relationship exists between increasing the popularity term $P(\mathbf{x})$ and ensuring that similar items are not placed adjacent to each other in the item list. Furthermore, the decrease in the popularity term value slows down as w increases. **Figure 4C** shows that when diversity is not considered for item listing, the diversity term $D(\mathbf{x})$ is lower than the average value of 0 and increases with increasing w. Furthermore, the increase in the diversity term $D(\mathbf{x})$ gradually slows down as w increases.

Note that the behavior of $P(\mathbf{x})$ and $D(\mathbf{x})$ based on w is the same whether or not the number of items is 6 or 8.

Figure 5 shows the positional relationships between the top eight hotels and their type in area X of **Figures 3B**, **4D**. The top eight hotels in area X particularly include four city hotels and four budget hotels, which can also be classified as in the north or south region of area X; hence, it was chosen as an example.

Figure 6 represents the actual hotel lists in area X obtained by solving (2) for different values of w. The top five hotels in the northern area are consecutively listed when w is 0. The city hotels are listed in succession among the top four; thus, the list is biased. In contrast, only the top two hotels are the same in terms of both area and hotel type when w is between 0.1 and 0.6, that is, when diversity is considered. Furthermore, no similarity exists between the consecutively listed hotels both in terms of area and hotel type when w is larger than 0.7. Thus, despite not using semantic information (e.g., area and hotel type) in the actual calculations, a semantically diverse item list was created using the number of co-browsers as the similarity measure.

So far, we have used the co-browsing similarity as $f_{ii'}$. In **Figure 7**, we compared the item lists in area X using the co-browsing similarity and the semantic similarity as similarity measures. Each item list was obtained by solving (2) for $w = 1$. The semantic similarity was calculated using the negative Euclidean distance of each hotel based on the two-dimensional vector of whether the area is in the north or south, and the hotel type (city hotel or budget hotel).

Figure 7 shows that the item list is not consecutive with respect to both the area and the hotel type by solving (2) even if both similarity types are used. The difference between these two item lists is that E, the budget hotel northern area, rose from the fourth to the first place. The item lists created using the explicit semantic similarity have fewer sequences for the hotel type than those created using the co-browsing similarity. This result confirms that the item list, which considered diversity for the purpose of this study, was consistently created regardless of the similarity measure used.

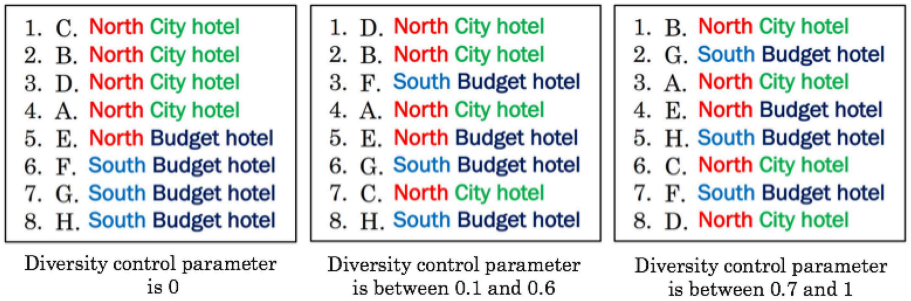


FIGURE 6 | A particular area's item lists when the diversity control parameter w is between 0 and 1.

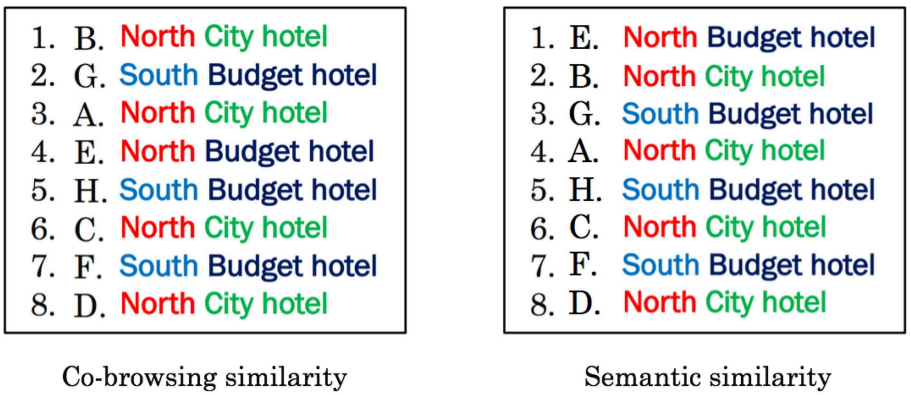


FIGURE 7 | Item lists using co-browsing similarity and semantic similarity as similarity measures in a particular area. The diversity control parameter w in creating each item list is 1.

3.3. Performance Evaluation of Problem Decomposition

We compared the method of extracting partial problems by `qbsolv` and our method of extracting partial problems considering the problem structure of the item list. Our experiments were performed by comparing the objective values of (4) when solving the problem of 12, 16, 20 and 24 items using each method.

Table 2 lists the average of the objective values for the 10 areas obtained by solving the problem for different numbers of items and the gap between the objective values of the proposed and original `qbsolv` methods. The goal of the problem was to minimize the objective value, indicating that the smaller the objective value, the better the performance. The gap between the objective values of the proposed and original `qbsolv` methods increased as the number of items increased. The effectiveness of the proposed method also increased. We conducted a Wilcoxon signed-rank test on the difference between the objective values of the original `qbsolv` and proposed methods (Bonferroni correction was performed on the number of items). Consequently, the null hypothesis in this case was rejected at a significance level of 5% (p-value = 0.020).

In terms of application, our proposed method is not only limited to the item list optimization problem, but can also be widely applied to other problems involving similar constraints, such as the assignment problem (1). For example, our method can be applied to the traveling salesman problem, which typically includes two constants A, B along with the following penalty terms:

$$A\sum_{i\in I}\left(\sum_{j\in J}x_{ij}-1\right)^2+B\sum_{j\in J}\left(\sum_{i\in I}x_{ij}-1\right)^2.$$

4. CONCLUSION

This study proposed a method of creating item lists on an e-commerce website as a QAP considering item popularity and diversity. We converted the QAP to a QUBO such that it can be directly solved by the quantum annealer, D-Wave 2000Q. Direct manipulation to solve the resulting QUBO was not possible in the case with a large number of items because of the limited number of qubits available in the current version of the quantum annealer and the restriction on specifying connections between the qubits. Therefore, we proposed a decomposition technique exploiting the structure of the problem. The original large problem was divided into several subproblems, which can eventually be solved by D-Wave 2000Q individually. Our experiments using actual real-world data demonstrated the efficiency of our proposed approach. A remarkable observation

TABLE 2 | Comparison of the objective values for problem decomposition.

	Methods		
Number of items	**qbsolv**	**Proposed method**	**Gap**
12	−160.038	−160.337	−0.299
16	−268.777	−270.176	−1.399
20	−391.428	−393.051	−1.623
24	−505.634	−509.266	−3.632

made from the experimental results was that the output item list changed based on the diversity control parameter. Our formulation led to the antiferromagnetic Ising model with a random field. The resulting lists were "aligned" along the random field when the diversity control parameter was small. In contrast, increasing the diversity control parameter eliminated the order in the item list and introduced diversity.

However, our research has some limitations. The item list created by our method will not work well when the data needed to calculate the popularity p_{ij} and similarity $f_{ii'}$ are insufficient, because no reliable estimate values will exist in that case. Obtaining good estimates for p_{ij} and $f_{ii'}$ requires access logs pertaining to when items are placed at various positions. The determination of the diversity consideration parameter w was also a limitation. w depends on the scale of diversity a customer of an e-commerce service wants; thus, it should be adjusted by changing w and monitoring performance, which is cumbersome to implement in real-world scenarios.

As the number of qubits available in the quantum annealer increases in the future, our method for division of a large-sized problem into smaller subproblems will become more useful. The experiments in this study clearly showed that our method performed better than `qbsolv` when the size of the problem became large. The structure of our problem is similar to that of the traveling salesman problem and the scheduling problem in that it takes the form of a QUBO with two quadratic functions based on two constraints (i.e., one for distance or time and the other for list locations or tasks).

Thus, our method has a wide range of applications involving optimization problems that can be solved via QUBO formulation, such as those using the QA, D-Wave 2000Q, and other types of QUBO solvers. Our results indicate that not only the evolution of hardware devices, but also the development of better software based on the structure of problems are essential for future QA applications.

DATA AVAILABILITY

The datasets generated for this study can be found in the GitHub repository of the Recruit Communications Co., Ltd[2].

AUTHOR CONTRIBUTIONS

NN contributed to the conception and design of the study, performed all the experiments, and wrote the first draft of the manuscript. KT implemented the program of our problem decomposition method. MO, MM, and KS contributed to the manuscript revision. All authors discussed this study, then reviewed and approved the final version of the manuscript.

ACKNOWLEDGMENTS

The authors would like to thank Recruit Lifestyle Co., Ltd. and Recruit Communications Co., Ltd. for their support in this exploratory research project.

[2]https://github.com/recruit-communications/Item-Listing-Datasets

REFERENCES

Abdel-Basset, M., Manogaran, G., Rashad, H., and Zaied, A. N. H. (2018). A comprehensive review of quadratic assignment problem: variants, hybrids and applications. *J. Amb. Intell. Human. Comput.* doi: 10.1007/s12652-018-0917-x. [Epub ahead of print].

Adomavicius, G., and Kwon, Y. (2011). "Maximizing aggregate recommendation diversity: a graph-theoretic approach," in *Proceedings of the 1st International Workshop on Novelty and Diversity in Recommender Systems (DiveRS 2011)* (Chicago, IL), 3–10.

Anstreicher, K. M. (2003). Recent advances in the solution of quadratic assignment problems. *Math. Progr.* 97, 27–42. doi: 10.1007/s10107-003-0437-z

Antikacioglu, A., and Ravi, R. (2017). "Post processing recommender systems for diversity," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Halifax, NS), 707–716. doi: 10.1145/3097983.3098173

Arai, S., Ohzeki, M., and Tanaka, K. (2018). Deep neural network detects quantum phase transition. *J. Phys. Soc. Jpn.* 87:033001. doi: 10.7566/JPSJ.87.033001

Berkley, A. J., Johnson, M. W., Bunyk, P., Harris, R., Johansson, J., Lanting, T., et al. (2010). A scalable readout system for a superconducting adiabatic quantum optimization system. *Superconduct. Sci. Technol.* 23:105014. doi: 10.1088/0953-2048/23/10/105014

Booth, M., Reinhardt, S. P., and Roy, A. (2017). *Partitioning Optimization Problems for Hybrid Classical/Quantum Execution*. Available online at: https://www.dwavesys.com/resources/publications

Boothby, T., King, A. D., and Roy, A. (2016). Fast clique minor generation in chimera qubit connectivity graphs. *Quant. Inform. Process.* 15, 495–508. doi: 10.1007/s11128-015-1150-6

Bunyk, P., Hoskinson, E. M., Johnson, M. W., Tolkacheva, E., Altomare, F., Berkley, A. J., et al. (2014). Architectural considerations in the design of a superconducting quantum annealing processor. *IEEE Trans. Appl. Superconduct.* 24, 1–10. doi: 10.1109/TASC.2014.2318294

Crawford, D., Levit, A., Ghadermarzy, N., Oberoi, J. S., and Ronagh, P. (2016). Reinforcement learning using quantum boltzmann machines. *arXiv: 1612.05695*.

D-Wave Systems Inc. (2017). *D-Wave Solver Properties and Parameters Reference*. Available online at: https://docs.dwavesys.com/docs/latest

D-Wave Systems Inc. (2018). *Getting Started With the D-Wave System*. Available online at: https://docs.dwavesys.com/docs/latest

Harris, R., Johnson, M. W., Lanting, T., Berkley, A. J., Johansson, J., Bunyk, P., et al. (2010). Experimental investigation of an eight-qubit unit cell in a superconducting optimization processor. *Phys. Rev. B* 82:024511. doi: 10.1103/PhysRevB.82.024511

Henderson, M., Novak, J., and Cook, T. (2018). Leveraging adiabatic quantum computation for election forecasting. *arXiv: 1802.00069*.

Hernandez, M., and Aramon, M. (2017). Enhancing quantum annealing performance for the molecular similarity problem. *Quant. Inform. Process.* 16:133. doi: 10.1007/s11128-017-1586-y

Johnson, M., Bunyk, P., Maibaum, F., Tolkacheva, E., Berkley, A., Chapple, E., et al. (2010). A scalable control system for a superconducting adiabatic quantum optimization processor. *Superconduct. Sci. Technol.* 23:065004. doi: 10.1088/0953-2048/23/6/065004

Kadowaki, T., and Nishimori, H. (1998). Quantum annealing in the transverse ising model. *Phys. Rev. E* 58:5355. doi: 10.1103/PhysRevE.58.5355

Khoshaman, A., Vinci, W., Denis, B., Andriyash, E., and Amin, M. H. (2018). Quantum variational autoencoder. *Quant. Sci. Technol.* 4:014001. doi: 10.1088/2058-9565/aada1f

Li, R. Y., Felice, R. D., Rohs, R., and Lidar, D. A. (2018). Quantum annealing versus classical machine learning applied to a simplified computational biology problem. *NPJ Quant. Inform.* 4:14. doi: 10.1038/s41534-018-0060-8

Long, B., and Chang, Y. (2014). *Relevance Ranking for Vertical Search Engines*. Waltham, MA: Elsevier. Available online at: https://www.elsevier.com/books/relevance-ranking-for-vertical-search-engines/long/978-0-12-407171-1

Lucas, A. (2014). Ising formulations of many np problems. *Front. Phys.* 2:5. doi: 10.3389/fphy.2014.00005

Neukart, F., Compostella, G., Seidel, C., von Dollen, D., Yarkoni, S., and Parney, B. (2017). Traffic flow optimization using a quantum annealer. *Front. ICT* 4:29. doi: 10.3389/fict.2017.00029

Neukart, F., Von Dollen, D., Seidel, C., and Compostella, G. (2018). Quantum-enhanced reinforcement learning for finite-episode games with discrete state spaces. *Front. Phys.* 5:71. doi: 10.3389/fphy.2017.00071

Ohzeki, M., Miki, A., Miyama, M. J., and Terabe, M. (2019). Control of automated guided vehicles without collision by quantum annealer and digital devices. *arXiv: 1812.01532*.

Ohzeki, M., Okada, S., Terabe, M., and Taguchi, S. (2018a). Optimization of neural networks via finite-value quantum fluctuations. *Sci. Rep.* 8:9950. doi: 10.1038/s41598-018-28212-4

Ohzeki, M., Takahashi, C., Okada, S., Terabe, M., Taguchi, S., and Tanaka, K. (2018b). Quantum annealing: next-generation computation and how to implement it when information is missing. *Nonlin. Theory Its Appl.* 9, 392–405. doi: 10.1587/nolta.9.392

Okada, S., Ohzeki, M., Terabe, M., and Taguchi, S. (2019). Improving solutions by embedding larger subproblems in a d-wave quantum annealer. *Sci. Rep.* 9:2098. doi: 10.1038/s41598-018-38388-4

Perdomo-Ortiz, A., Dickson, N., Drew-Brook, M., Rose, G., and Aspuru-Guzik, A. (2012). Finding low-energy conformations of lattice protein models by quantum annealing. *Sci. Rep.* 2:571. doi: 10.1038/srep00571

Rosenberg, G., Haghnegahdar, P., Goddard, P., Carr, P., Wu, K., and de Prado, M. L. (2016). Solving the optimal trading trajectory problem using a quantum annealer. *IEEE J. Select. Top. Sig. Process.* 10, 1053–1060. doi: 10.1109/JSTSP.2016.2574703

Takahashi, C., Ohzeki, M., Okada, S., Terabe, M., Taguchi, S., and Tanaka, K. (2018). Statistical-mechanical analysis of compressed sensing for hamiltonian estimation of ising spin glass. *J. Phys. Soc. Jpn.* 87:074001. doi: 10.7566/JPSJ.87.074001

Tanahashi, K., Takayanagi, S., Motohashi, T., and Tanaka, S. (2019). Application of ising machines and a software development for ising machines. *J. Phys. Soc. Jpn.* 88:061010. doi: 10.7566/JPSJ.88.061010

Vazirani, V. V. (2013). *Approximation Algorithms*. Springer Science & Business Media.

Venturelli, D., Dominic, J. M., and Rojo., G. (2015). Quantum annealing implementation of job-shop scheduling. *arXiv: 1506.08479*.

Wang, Y., Yin, D., Jie, L., Wang, P., Yamada, M., Chang, Y., et al. (2016). "Beyond ranking: optimizing whole-page presentation," in *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining* (San Francisco, CA), 103–112.

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

APPENDIX A

Quantum Annealing (QA) on D-Wave 2000Q

QA belongs to a class of meta-heuristic algorithms, which exploit the quantum tunneling effect to efficiently solve an optimization problem (Kadowaki and Nishimori, 1998). The quantum processing unit (QPU) is designed to find the lowest energy state of a spin glass system. This energy state is described by an Ising Hamiltonian:

$$H_{\mathrm{P}} = \sum_{i \in V} h_i s_i + \sum_{(i,j) \in E} J_{ij} s_i s_j,$$

where h_i is the on-site energy of qubit i, and J_{ij} denotes the interaction energies of two qubits i and j. The binary variables $s_i \in \{-1, +1\}$ are called spins, and are fixed in a lattice graph G with vertices and edges (V, E). Finding the ground state of such a spin glass system, that is, the state with the lowest energy is an NP problem. Therefore, QA can find the solutions of NP problems by mapping them onto spin glass systems. The basic process of QA is to interpolate physically between an initial Hamiltonian H_0, with an easy-to-implement ground state, and a problem Hamiltonian H_{P}, whose minimal configuration needs to be explored. Then, we change the Hamiltonian slowly such that it is the spin glass Hamiltonian at time T:

$$H(t) = \left(1 - \frac{t}{T}\right) H_0 + \left(\frac{t}{T}\right) H_{\mathrm{P}}.$$

If T is long enough, according to the adiabatic theorem, the system will be in the ground state of the spin glass Hamiltonian H_P.

For computation on D-Wave 2000Q, the problem is first mapped to the Ising binary and quadratic structures. Then, it is embedded in the available qubit lattice. The qubits are arranged according to a chimera graph on D-Wave 2000Q. Each qubit couples to five or six others, except when the qubit has defects. If the problem does not embed directly, auxiliary qubits can be introduced to augment the available couplings. However, introducing auxiliary qubits is a significant cost in qubits. Both mapping and embedding imply restrictions on the types of problems that can effectively solved with the D-Wave 2000Q. For more details on QA in the D-Wave 2000Q, see D-Wave Systems Inc. (2018).

frontiers in Physics

ORIGINAL RESEARCH
published: 18 July 2019
doi: 10.3389/fphy.2019.00103

A Fast Machine Learning Model for ECG-Based Heartbeat Classification and Arrhythmia Detection

Miquel Alfaras, Miguel C. Soriano and Silvia Ortín*

Instituto de Física Interdisciplinar y Sistemas Complejos, IFISC (UIB-CSIC), Palma de Mallorca, Spain

We present a fully automatic and fast ECG arrhythmia classifier based on a simple brain-inspired machine learning approach known as Echo State Networks. Our classifier has a low-demanding feature processing that only requires a single ECG lead. Its training and validation follows an inter-patient procedure. Our approach is compatible with an online classification that aligns well with recent advances in health-monitoring wireless devices and wearables. The use of a combination of ensembles allows us to exploit parallelism to train the classifier with remarkable speeds. The heartbeat classifier is evaluated over two ECG databases, the MIT-BIH AR and the AHA. In the MIT-BIH AR database, our classification approach provides a sensitivity of 92.7% and positive predictive value of 86.1% for the ventricular ectopic beats, using the single lead II, and a sensitivity of 95.7% and positive predictive value of 75.1% when using the lead V1'. These results are comparable with the state of the art in fully automatic ECG classifiers and even outperform other ECG classifiers that follow more complex feature-selection approaches.

Keywords: Echo State Networks, reservoir computing, arrhythmia classification, GPU, ECG

OPEN ACCESS

Edited by:
Raul Vicente,
Max-Planck-Institut für Hirnforschung, Germany

Reviewed by:
Haroldo Valentin Ribeiro,
State University of Maringá, Brazil
Reinaldo Roberto Rosa,
National Institute of Space Research (INPE), Brazil

****Correspondence:***
Miquel Alfaras
m.phy@live.com

Specialty section:
This article was submitted to Interdisciplinary Physics, a section of the journal Frontiers in Physics

Received: *14 May 2019*
Accepted: *03 July 2019*
Published: *18 July 2019*

Citation:
Alfaras M, Soriano MC and Ortín S (2019) A Fast Machine Learning Model for ECG-Based Heartbeat Classification and Arrhythmia Detection. Front. Phys. 7:103. doi: 10.3389/fphy.2019.00103

1. INTRODUCTION

Electrocardiogram (ECG) analysis has been established at the core of cardiovascular pathology diagnosis since its development in the twentieth century. The ECG signals reflect the electrical activity of the heart. Thus, heart rhythm disorders or alterations in the ECG waveform are evidences of underlying cardiovascular problems, such as arrhythmias. Non-invasive arrhythmia diagnosis is based on the standard 12-lead electrocardiogram, which measures electric potentials from 10 electrodes placed at different parts of the body surface, six in the chest and four in the limbs. In order to provide an effective treatment for arrhythmias, an early diagnosis is important. Early detection of certain types of transient, short-term or infrequent arrhythmias requires long-term monitoring (more than 24 h) of the electrical activity of the heart. The fast development of the digital industry has allowed for improvements in devices, data acquisition and computer-aided diagnosis methods.

The open access to ECG databases [1] has led to the development of many methods and approaches for computer-aided ECG arrhythmia classification over the last decades, fostering the productive cross-disciplinary efforts that engineers, physicists or non-linear dynamics researchers are no strangers to. Almost every computer-aided ECG classification approach involves four main steps, namely, the preprocessing of the ECG signal, the heartbeat detection, the feature extraction and selection and finally the classifier construction. The preprocessing of the ECG signal and the heartbeat detection are out of the scope of this work, both widely studied, and the heartbeat detection is close to optimal results [2].

A large number of classifiers have been proposed for arrhythmia discrimination. The proposed techniques range from simple classifiers, such as linear discriminants (LD) [3–5] or decision trees [5–7], to more sophisticated ones, such as traditional neural networks [8–13], Support Vector Machines (SVM) [9, 14–18], conditional random fields [19], and more recently deep learning techniques [13, 20–22]. In addition, many works have been devoted to finding the best combination of features, sometimes even developing complex signal processing methods, and to choosing the best subset (dimensionality reduction) for the arrhythmia classification [23]. On the one hand, popular choices for the input features are morphological features extracted from the time domain (such as inter-beat intervals, amplitudes, areas) [3, 14, 15, 24], frequency-domain features [6, 7, 16, 17, 25], wavelet transforms [4, 8–11, 18, 26], complex heartbeat representations [16] or higher order statistics (HOS) [4, 6, 7, 9]. On the other hand, feature selection methods, such as the independent component analysis (ICA) [18, 26], principal component analysis (PCA) [18], particle swarm optimization (PSO) [16], or the genetic algorithm—back propagation neural networks (GA-BPNN) [23], have been used.

Despite the good performance in classifying arrhythmias achieved by these methods, many of them require long computation times to optimize the classifiers. The use of complex classification or preprocessing methods is not suitable for online calculations or demand a lot of computational power. In this work, we present a fully automatic and fast classifier of arrhythmias that can be implemented online and analyze long sequences of ECG records efficiently. By loosening the requirements for feature extraction, we propose an implementation fundamentally based on raw signals, single lead information and heart rates that aims at reducing computation time while achieving low error classification results.

Cardiologists use mostly the raw ECG to diagnose. The simplest and fastest method of feature extraction is then to extract sampled points from an ECG signal curve. However, one should be aware of the fact that the amount of the extracted features used to characterize the heartbeat can be a burden for the classification algorithm. For this reason, most of the works that use the raw signal perform a down sampling of the waveform or some feature selection in order to reduce the computation time [3, 4, 15]. In order to circumvent this issue, a simple machine learning method is chosen to classify the arrhythmias. One of the advantages of the proposed method is that the number of features barely affects the speed of the classification since the classifier parameters related to the input are not optimized and remain random, as it will be described in more detail later in the text. As a result, the raw waveform of the heartbeat can be used for the classification without compromising speed. This simple machine learning method also allows a fast retraining of the classifier if new ECG data become available.

In this work, we propose an ensemble of Echo State Networks (ESNs) [27] as the classifier method, using the raw ECG waveforms and time intervals between the heartbeats as the input features. A particular advantage of the ESNs is that they have recurrent connections, being able to take into account time dependencies between neighboring heartbeats. This property is beneficial since, in the case of a normal or an abnormal heartbeat, there are more chances that the subsequent heartbeat will also be a healthy or a pathological one. Moreover, the ESN method can take advantage of the power of a parallel computing architecture, such as a graphics processing unit (GPU). Hence, we compare the computation times between a GPU and a central processing unit (CPU), showing that the implementation in a GPU outperforms its CPU counterpart in the classification of the heartbeats. The computation times of the GPU outperform those of the CPU even in the training part of the classifier, i.e., the entire system can be trained extremely fast with a GPU.

Finally, it is worth noting that our classifier is based on a single lead ECG. Long-term monitoring generally involves devices with fewer electrodes than the standard 12 leads ECG in order to allow the patient to have a normal activity, requiring computer-aided techniques to analyze the huge amounts of data generated. We show that our heartbeat classification method outperforms other classifiers that rely on much more complicated feature selection techniques and complex calculations. We evaluate the proposed classifier in two different ECG databases and leads to test the robustness of the proposed algorithm.

2. MATERIALS AND METHODS

2.1. Databases

The performance of the proposed heartbeat classification method has been evaluated in two internationally recognized ECG databases: the MIT-BIH arrhythmia (MIT-BIH AR) [28] and the AHA [29]. The MIT-BIH AR database is a golden standard to evaluate arrhythmia classifiers. This benchmark database consists of 48 half-hour ECG records sampled at 360 Hz. Each ECG record contains two leads: lead II (modified limb lead II, obtained from electrodes on the chest) and lead V1' (modified lead V1, and in some records V2, V4, or V5). The AHA database contains 154 ECG recordings of 3 h long but only the last 30 min have information about the beat class. The AHA ECG recordings have two leads (A,B) sampled at 250 Hz. The documentation of the AHA database does not provide the name of the leads.

Both databases have annotations indicating the class of the heartbeat and its position verified by independent experts. Following the standards and recommendations of the American National Standards Institute developed by the Association for the Advancement of Medical Instrumentation (AAMI) for the evaluation of ECG classifiers [30], all the heartbeat annotation labels are converted to five heartbeat types: N (normal beats), S (supraventricular ectopic beats), V (ventricular ectopic beats), F (fusion beats), and Q (unclassifiable beats). The Q beats were excluded in this research because they are not representative [31]. Also in accordance to the AAMI standard, ECG recordings with paced beats are removed (i.e., four ECG records in the MIT-BIH AR database and three ECG records in the AHA database are excluded from the analysis). It is worth mentioning that the original annotations of the AHA database do not differentiate between N and S beats.

2.1.1. Training and Test Datasets

Each database is split into two sets: one for training (DS1) and one for testing (DS2). This division of the data is chosen to balance the presence of the different types of heartbeats and number of subjects in each dataset. It takes into account the inter-patient division, i.e., the subjects used to construct or optimize the classifier (DS1) are different from the subjects used to evaluate it (DS2). It has been demonstrated [3] that models which use heartbeats of the same patient in both the training and test are biased and their results can not be replicated in real environments.

For the MIT-BIH AR database we adopted the same set division as in de Chazal et al. [3] for comparative purposes of the results. 22 of the 44 ECG records of the MIT-BIH AR database are part of the set DS1 and the other 22 are part of the set DS2. For the AHA database, we use the recordings recommended for the training and testing procedure in the original AHA database description. In the AHA database, the set DS1 contains 79 ECG recordings with the label *series* = 0 and the DS2, 75 recordings labeled with *series* = 1. The division scheme for the MIT-BIH AR and AHA databases is summarized in **Tables 1**, **2**, respectively. The beat class distributions of the different databases are given in **Table 3**.

TABLE 1 | Distribution of the MIT-BIH AR database ECG recordings into the training (DS1) and testing (DS2) sets.

Dataset	MIT-BIH AR recordings
DS1	101, 106, 108, 109, 112, 114, 115, 116, 118, 119, 122, 124, 201, 203, 205, 207, 208, 209, 215, 220, 223, 230
DS2	100, 103, 105, 111, 113, 117, 121, 123, 200, 202, 210, 212, 213, 214, 219, 221, 222, 228, 231, 232, 233, 234

TABLE 2 | Distribution of the AHA database ECG recordings into the training (DS1) and testing (DS2) sets.

Dataset	AHA recordings
DS1	1,001–1,004, 1,006–1,010, 2,001, 2,003–2,010, 3,001–3,010, 4,001–4,010, 5,001–5,010, 6,001–6,010, 7,001–7,010, 8,001–8,004, 8,006–8,010
DS2	1,101–1,110, 2,101–2,110, 3,101–3,110, 4,101–4,110, 5,101–5,105, 6,101–6,110, 7,101–7,110, 8,101–8,105, 8,107–8,110

The ECG recording names in the AHA database are of the form CSNN, where C is the arrhythmia category, S is the series and NN is the file number in the category.

TABLE 3 | Heartbeat class distribution of the training (DS1) and testing (DS2) sets.

	SVEB+ class		VEB+ class	
Database	N	S	V	F
MIT-BIH AR (DS1)	45,783	943	3,785	414
MIT-BIH AR (DS2)	44,179	1,834	3,216	388
AHA (DS1)	158,587		15,075	292
AHA (DS2)	156,992		15,855	437

The beats at the beginning and at the end of the recordings are discarded as they do not provide information about the temporal distance to the neighboring beats.

2.2. Performance Metrics

The performance of the proposed algorithm is evaluated using the MIT-BIH AR and AHA databases on a single lead basis. The performance of each classification algorithm is assessed using four standard statistical measures: sensitivity (Se), positive predictive value (PPV), specificity (Sp), and accuracy (Acc). They are calculated as follows:

$$Se = TP/(TP + FN), \tag{1}$$

$$PPV = TP/(TP + FP), \tag{2}$$

$$Sp = TN/(TN + FP), \tag{3}$$

$$Acc = (TP + TN)/(TP + TN + FP + FN) \tag{4}$$

True positives (TP) indicate correctly predicted positive class and true negatives (TN) indicate correctly predicted negative class heartbeats. A good classifier is the one that minimizes false negatives (FN) and false positives (FP).

The F1 score is the harmonic mean of Se and PPV, $F1 = 2(Se \cdot PPV)/(Se + PPV)$. The F1 score is used to choose the optimum parameters of our classifier during the training phase.

2.3. The Heartbeat Classifier

The proposed heartbeat classifier is based on an Echo State Network (ESN). It classifies the heartbeats of the processed ECG recordings in two classes based on morphology: SVEB+ and VEB+. SVEB+ class includes normal (N) and supraventricular ectopic (S or SVEB) heartbeats. These heartbeats have a normal morphology and a supraventricular origin as opposed to VEB+ heartbeats that present ventricular origin or abnormal morphology. The VEB+ class comprises the ventricular ectopic beats (V or VEB) and the fusion beats (F).

The overall process is schematically represented in **Figure 1**. The two stages are clearly differentiated:

- Stage 1–Processing of the ECG recordings: this procedure involves the filtering, heartbeat detection, heartbeat segmentation, and feature extraction. We include morphological and time intervals between heartbeats in our model.
- Stage 2–Classification between SVEB+ and VEB+ classes: we use an ensemble of ESNs with ring topology to perform this classification task.

We discuss the classification procedure in stage 2 in more detail later in the text.

2.4. Processing of the ECG and Feature Extraction

In order to accomplish arrhythmia classification, minor preprocessing needs to be applied to the source ECG records. In

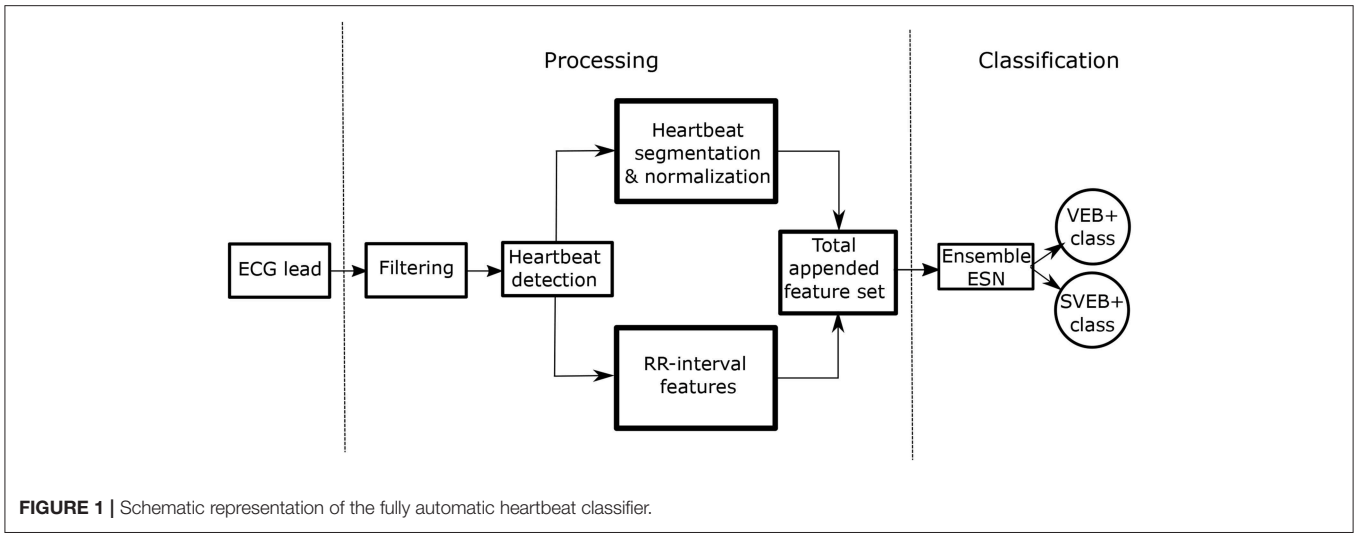


FIGURE 1 | Schematic representation of the fully automatic heartbeat classifier.

our system, the processing of the ECG recordings includes the following steps:

1. ECG re-sampling: ECG signals are processed with a common sampling rate of 250 Hz. The AHA database (250 Hz) keeps its original sampling rate and the MIT-BIH AR database (360 Hz) is resampled to 250 Hz using the PhysioToolkit software package [1].
2. ECG filtering: All ECG recordings are filtered in a bandwidth ν (Hz) $\in [0.5, 35]$, to correct the baseline and remove unwanted high frequency noise. A Butterworth high-pass filter (with a cutoff frequency $\nu_c = 0.5$ Hz) and a finite impulse response filter of 12th order (35 Hz, at 3-dB point) are used, following standard procedure.
3. Heartbeat detection: To determine the position of the heartbeats, the annotated positions provided by the databases are used. In the MIT-BIH AR database the annotation position occurs at the largest of the local extrema of the QRS complex. Beat detection is beyond the scope of this study. Highly accurate automated beat detection methods have already been reported [32].
4. RR calculation: The RR interval is defined as the time interval between successive heartbeats. The RR interval associated to a heartbeat i, $RR(i)$, corresponds to the time difference between the heartbeat i and the previous heartbeat $(i-1)$.
5. Heartbeat segmentation: The ECG signal is segmented around the annotated position given by each database. The size of the segmented heartbeat is 240 ms (60 samples at 250 Hz) and it is centered around the annotation position.
6. Heartbeat normalization: Each segmented heartbeat is normalized between $[-1, 1]$. This scaling operation results in a signal that is independent of the original ECG recording amplitude.

After processing the ECG recordings, each heartbeat is represented by a set of features. One of the main goals related to the feature selection in our model is to avoid complicated features with a high computational cost, since we aim to design a fast and real-time heartbeat classifier. Therefore, we focus on simple ways to extract features. In our case, we use the raw waveform of each heartbeat around the heartbeat position to represent it. The raw data of each beat was represented by an equal number of samples from each side from the point of the beat annotation. In order to learn from the temporal characteristics of each beat, information about the RR intervals is also added to the heartbeat features. The RR intervals are features used in almost all the methods to classify arrhythmic heartbeats. For instance, it is well-known that VEB heartbeats are characterized by shorter RR intervals than the N heartbeats. We found that using the logarithm of the RR intervals, as in Llamedo and Martinez [33], leads to a slightly better performance of the classifier. All the features that characterize the ith heartbeat are listed below:

- 60 raw samples of the segmented heartbeat waveform centered around the position annotated for the heartbeat.
- $\ln(RR(i))$: logarithm of the current RR interval.
- $\ln(RR(i+1))$: logarithm of the next RR interval.
- $\ln(RR_{mean})$ logarithm of an average of the previous 250 RR intervals (averaging over the n available RR intervals when $n < 250$).

At the end of the processing and feature extraction stage, each heartbeat is represented as a d-dimensional vector containing three features related to the RR intervals and 60 morphological features, which are simply the samples of the ECG waveform around the position annotated for each heartbeat. This d-dimensional vector ($d = 63$) is the input for the classification algorithm.

2.5. Classification Algorithm: Echo State Network

Our classifier is built upon an ESN with a ring topology. ESNs are a popular implementation of Reservoir Computing (RC). RC is an established paradigm in machine learning that has been successfully applied in a variety of different tasks [27, 34]. This computing paradigm is made of three layers: input, reservoir and output (see general ESN scheme in **Figure 2A**. In the case of the ESN, the reservoir is a

recurrent neural network with random input and random connection weights between the neurons. Thanks to the recurrence of the network, current reservoir responses depend on the previous state of the reservoir, yielding an ESN capable of performing context-dependent computations. The reservoir benefits from a high-dimensional non-linear mapping of the input, so that the reservoir response is easier to classify than the original input by means of a simple linear regression technique.

At the input stage, the ECG data must be fed into the reservoir network. In this process, dimensions must change from $d \times Hb$ to $N \times Hb$, where d, Hb, and N are the number of input features, heartbeats, and network neurons, respectively. The mapping from the input into the reservoir is done through a random input matrix $\mathbf{W}^{in}_{N\times d}$ generated from a uniform distribution $\in [-1, 1]$. Hence, the ECG data original features vector $\mathbf{u}_{d\times Hb}$ is modified according to:

$$\mathbf{X}_{N\times Hb} = (\mathbf{W}^{in}_{N\times d} \times \mathbf{u}_{d\times Hb}). \tag{5}$$

Once the first data is fed into the reservoir, the input proceeds sequentially and further reservoir responses are computed

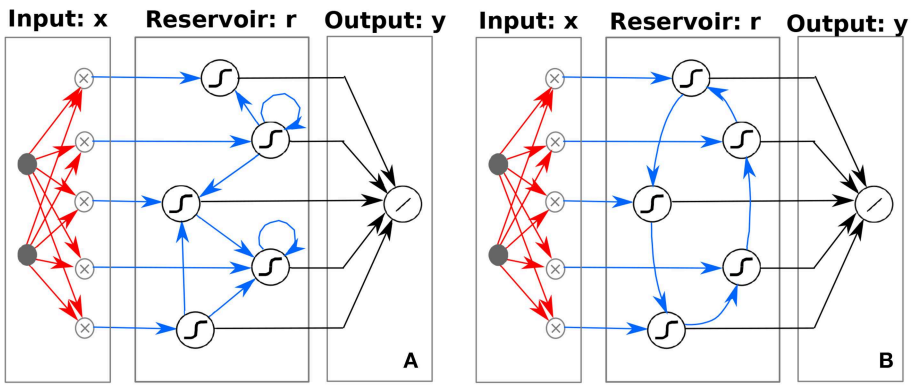


FIGURE 2 | Schematic illustration of **(A)** traditional ESN, depicting the high-dimensional non-linear mapping of the input to a reservoir with random and sparse internal node connectivity and **(B)** ring ESN, depicting the high-dimensional non-linear mapping of the input to a reservoir with a specific ring topology internal node connectivity. Weights optimized during the learning process are indicated by black arrows ($\mathbf{W}^{out}$), whereas random weights are depicted with red arrows ($\mathbf{W}^{in}$). Random **(A)** or predefined **(B)** weights are depicted with blue arrows (**W**). Although it is not explicitly depicted in the figure, the d-dimensional input **x** is augmented with an additional constant node accounting for the bias term.

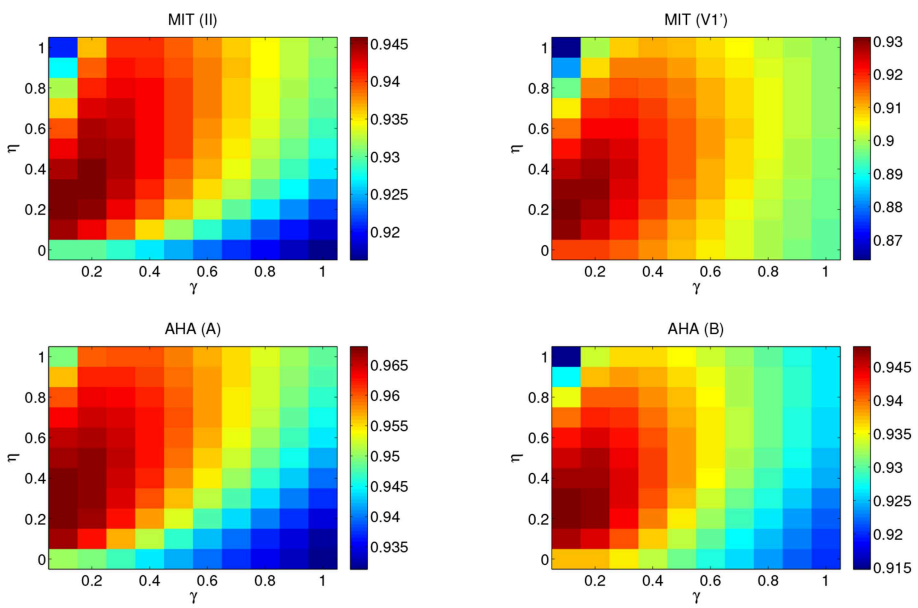


FIGURE 3 | Performance map of the F1 score obtained for the MIT-BIH AR and AHA databases from a 5-fold cross-validation on the set DS1. The number of neurons is $N = 500$ and the results have been averaged over 100 different input random matrices. η ranges from 0 to 1 and γ from 0.1 to 1. Top panels correspond to the MIT-BIH AR database lead II (left) and V1' (right). Bottom panels correspond to the AHA database lead A (left) and B (right). Each performance map adapts the color range so that optimal values can be easily identified by visual inspection.

iteratively. The reservoir matrix response $\mathbf{r}$ for the nth heartbeat for the standard ESN is obtained as follows:

$$\mathbf{r}(n) = F(\gamma \mathbf{X}(n) + \eta \mathbf{W}\mathbf{r}(n-1)), \tag{6}$$

where $\mathbf{W}$ is the random connection square matrix, with dimensions $N \times N$, F is the ESN activation function and γ and η are the input and connection scaling parameters, respectively. For the standard ESN, $\mathbf{W}$ is also generated from a uniform distribution $\in [-1, 1]$ and defines the connection weights between the internal neurons. For the non-linear function, we choose the classical sigmoid function with exponent -4 and a bias of 0.5, i.e., $F(x) = \frac{1}{1+e^{-4x}} - 0.5$. Reservoir computers with these sigmoid functions have shown optimal results solving different tasks [35]. Other activation functions, such as rectifiers can also be used.

In this method, only the connections between the reservoir responses and the output are optimized using, usually, some simple linear regression. The response of the ESN to the input, $\mathbf{r}(n)$, is used to calculate the expected output, $\hat{\mathbf{y}}(n)$, according to:

$$\hat{\mathbf{y}}(n) = \mathbf{W}^{out}\mathbf{r}(n), \tag{7}$$

where $\mathbf{W}^{out}_{l\times N}$ are the output weights of the ESN and l the number of output nodes. The output weights are computed by minimizing the squared error between the train outputs and their corresponding target class values, usually employing a linear regression method [36]. In addition, the normal equation formulation is adopted. For the heartbeat classifier we have found that due to the experimental noise present in the original data, simple linear regression results are similar to ridge regression results. For this reason, we prefer the use of linear regression. In this work we deal with a classification task that requires a binary output, e.g., 0 and 1, for the SVEB+ and VEB+ classes, respectively. Thus, the continuous output given by Equation (7) is converted into a binary one by means of a decision threshold of 0.5.

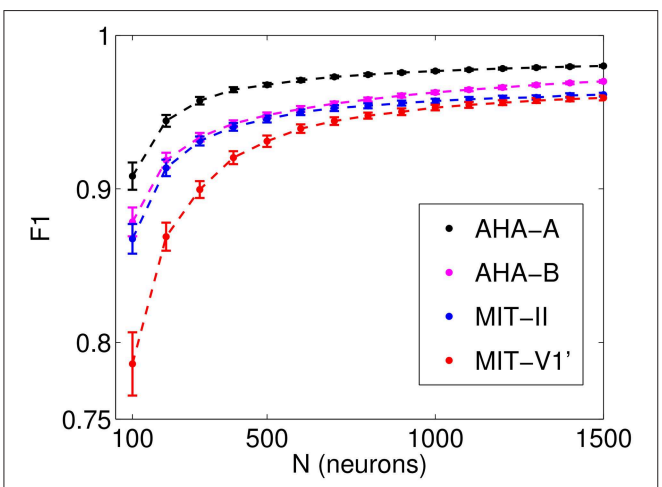


FIGURE 4 | Performance (F1 score) obtained from a 5-fold cross-validation on the set DS1 as a function of the number of neurons (N). Results for $\eta = 0.2$, $\gamma = 0.1$, which have been averaged over 100 different input random matrices.

In most of the ESN approaches, the connection matrix $\mathbf{W}$ is a sparse random matrix. This general form is schematically represented in **Figure 2A**. However, it has recently been shown that simpler ESN with ring topologies perform as well as those with a standard random connection matrix [37]. The ring ESN presents fixed random connections at the input layer $\mathbf{W}^{in}$ and fixed deterministic weights between internal reservoir neurons, with a connection matrix $\mathbf{W}$ of only non-zero elements in the lower sub-diagonal $\mathbf{W}_{i+1,i} = 1$ and at the upper-right corner $\mathbf{W}_{1,N} = 1$. The ring ESN is schematically illustrated in **Figure 2B**.

In this work, we use a ESN with ring topology for convenience. The simplicity of the ring ESN allows for an easy exploration of the system parameters in contrast to the computationally demanding trial and error process in ESNs with random topologies [37]. Moreover, this simplicity also allows an easy hardware implementation of the ring ESN using delay-coupled systems [38–41].

2.6. Parameter Optimization of the ESN for the SVEB+ and VEB+ Classification

The ring ESN topology allows for a simple optimization procedure, in contrast to the complex trial and error ESN construction with random topologies. The typical model construction decisions in a ring ESN include: setting the network size (N), the scaling parameters γ and η and the random input connections ($\mathbf{W}^{in}$). In this heartbeat arrhythmia classification task, the data are very imbalanced [the number of VEB+ cases is much smaller than the SVEB+ ones (see **Table 3**)], and the system is prone to have a high accuracy but a poor classification performance. Thus, the criterion to choose the optimum ring ESN parameters to discriminate between the SVEB+ and VEB+ classes is the one that maximizes the F1 score over the training set DS1.

The optimal η and γ values for each lead and database are determined via a 5-fold cross-validation over the corresponding training set. **Figure 3** shows the performance of the combinations of the pair (η, γ) with a fixed number of neurons $N = 500$ for the MIT-BIH AR and the AHA databases. To avoid an undesired dependence on the sparsity and randomness of the input connections, we average over 100 different input random matrices ($\mathbf{W}^{in}$). The parameter pair that yields the best overall classification is $\eta = 0.2$ and $\gamma = 0.1$. It is worth mentioning that the memory of past heartbeats helps the classification of heartbeats because the case of $\eta = 0$ (where ESN has no recurrent connections and it is just a feed-forward neural network with one hidden layer) is out of the optimum performance area. This suggests that the memory of past heartbeats helps the classification of present heartbeats. Once the pair (η, γ) is set, their optimal values are used to explore the dependence on the number of neurons (N) via a 5-fold cross-validation over the corresponding training set. The F1 score as a function of the number of neurons for the value pair ($\eta = 0.2$, $\gamma = 0.1$) is represented in **Figure 4**. As expected, the performance improves with the number of neurons but it starts to saturate for network

sizes over 700 neurons. One of the advantages of the ESN is that they are not prone to overfitting. Performance vs. N in the test set follows a similar trend than in the training set. We choose a value of $N = 1,000$ that suits a compromise between good performance for all the studied databases and leads and the computational time. The performance for $N = 1,000$ is only slightly lower than the one obtained for a larger number of neurons but requires a moderate computational time. The outcome of the optimization must be a fast algorithm suitable for real-time monitoring that, in addition, can be easily retrained when new data are available.

Subsequently, we search for the optimum input connectivity matrix $\mathbf{W}^{in}$. A usual approach would be to randomly generate several input matrices and choose the one that performs better in the training set. However, we note that optimizing the input matrix for the training set does not necessarily yield the optimum performance in the test set. Instead, we use a parallel ensemble method in our case since it yields an improvement in the performance. Ensemble methods have already been successfully used for arrhythmia classification [12, 15, 42]. Parallel ensemble methods are learning models that combine the outputs of multiple base classifiers generated in parallel. They exploit the independence between the base classifiers to obtain more accurate predictions than the average error of the individual classifiers. Ensembles are an effective technique if the base classifiers are reasonably accurate and there is diversity between their responses. In an ESN, the mapping of the input data to a high-dimensional non-linear reservoir varies depending on the randomly generated input matrix and this yields variability in the ESN outputs. The output of the ensemble is just the majority voting over the individual outputs of the ESNs. In **Figure 5**, we show the F1 score over the training set DS1 for an ensemble of ESNs with different input matrices as the number of members of the ensembles increases. After combining the outputs of 30 ring ESNs, the classifier performance does not improve when adding new members to the ensemble. Therefore, in the evaluation phase, we use ensembles of 30 ESNs.

In addition, we assess whether a faster alternative to the (η, γ) parameter optimization is feasible. To that end, we carry out an ensemble test on a classification that uses random values for the (η, γ) reservoir parameters. In this case, each member of the ensemble takes random values for the (η, γ) drawn from a uniform distribution between $[0, 0.8]$ and $[0.01, 0.5]$ for the η and γ parameters, respectively. Thus, the optimization of (η, γ) on the training set would not be necessary. However, we have found that the choice of random (η, γ) parameter values is valid for the classification of leads II (MIT-BIH AR) and A (AHA) but it yields a significant decrease in the PPV of leads V1' (MIT-BIH AR) and B (AHA). Therefore, $\eta = 0.2$ and $\gamma = 0.1$ are the optimum values used in the Results section.

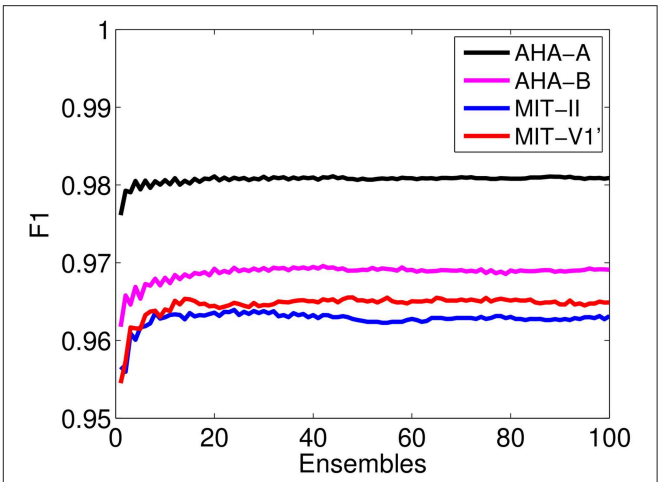


FIGURE 5 | Performance of the F1 score obtained on the set DS1 as a function of the number of ensembles of ESNs with different input random matrices. The parameters of the ring ESN that form the ensemble are set at their optimal values, $\eta = 0.2$, $\gamma = 0.1$, and $N = 1,000$.

TABLE 4 | VEB+ performance over the test set DS2 using an ensemble of 30 ring ESNs.

Database	Lead	Se (%)	PPV (%)	Sp (%)	Acc (%)
MIT-BIH AR	II	84.4 (82.9)	95.8 (85.5)	99.7(98.8)	98.6 (97.7)
	V1'	81.5 (78.9)	76.2 (66.0)	98.0 (96.6)	96.8 (95.3)
AHA	A	90.4 (87.2)	94.9 (92.4)	99.5 (99.2)	98.6 (98.5)
	B	87.9 (85.8)	89.6 (83.4)	98.9 (98.2)	97.8 (97.0)

The values into parenthesis show average of the individual performances of each ring ESN that is part of the ensemble.

3. RESULTS

3.1. Classifier Evaluation

After optimizing the parameters of the classifier over the training set (DS1) as described in the Methods section, we evaluate the classifier using the optimal parameters. The final performance is evaluated in the test phase with heartbeats that have not been used in the training set and come from different subjects (DS2 set).

Table 4 shows the classification performance obtained by an ensemble of 30 ring ESNs over the test set DS2. The parameters of the individual ESN are the ones optimized in the training phase. We highlight the fact that the optimal regime for the ESN coincides regardless of database and lead. Since the original heartbeat waveform is normalized between $[-1, 1]$ and the RR intervals are similar between both databases, the optimum ESN parameters ($\eta = 0.2$, $\gamma = 0.1$, and $N = 1000$) coincide for the MIT-BIH AR and the AHA databases. Thus, we expect that these optimum parameters can also be valid for other databases.

The best performance is obtained for the lead A of the AHA database. In the MIT-BIH AR, the lead II gives the best results. Comparing the ensemble results with those obtained with the average of ensemble base classifiers, it is clear that the ensembles reduce the overall error given by a single ESN. The ensembles remarkably reduce the incidence of the false negatives, leading to higher PPV. An ensemble of classifiers has already been used to classify heartbeats and significant improvements have been reported [12, 15]. The improvement in the classification accuracy thanks to the ensembles comes at the cost of higher computation

TABLE 5 | Cross database VEB+ performance over the test set DS2 using an ensemble of 30 ring ESNs.

Train (DS1)	Test (DS2)	Se (%)	PPV (%)	Sp (%)	Acc (%)
AHA A	AHA A	90.4	94.9	99.5	98.6
	AHA B	87.2	92.4	99.2	98.1
	MIT-BIH AR II	78.2	98.5	99.9	98.3
	MIT-BIH AR V1'	71.5	80.6	98.7	96.7
AHA B	AHA A	82.2	97.1	99.7	98.1
	AHA B	87.9	89.6	98.9	97.8
	MIT-BIH AR II	84.9	97.2	99.8	98.7
	MIT-BIH AR V1'	79.1	43.4	91.9	91.0
MIT-BIH AR II	AHA A	69.4	20.5	71.4	71.2
	AHA B	58.8	23.9	80.0	78.0
	MIT-BIH AR II	84.4	95.8	99.7	98.6
	MIT-BIH AR V1'	39.9	17.5	85.2	81.9
MIT-BIH AR V1'	AHA A	77.0	49.6	91.7	90.3
	AHA B	74.7	49.1	91.8	90.1
	MIT-BIH AR II	72.6	97.6	99.9	97.9
	MIT-BIH AR V1'	81.5	76.2	98.0	96.8

times. However, ensembles are inherently parallel, which can make them much more efficient at training and test time if one has access to a computer with multiple processors.

As part of our study, we assess the generalization capability of our SVEB+ and VEB+ classifier by evaluating the performance of the classifier on a lead and/or database different from the one used to train it. The results are shown in **Table 5**. The best generalization capability is obtained when the classifier is trained either with the AHA lead A or lead B, performing relatively well for all the analyzed leads in the test. The bigger size and the richer variety of the AHA database is likely the reason of the better generalization capability of the classifiers trained with the AHA leads than those trained with the MIT leads. The classification into SVEB+ and VEB+ is based mainly on the morphological shape of the lead. In spite of this lead dependency, the classifier can to some extent generalize to other leads. It is worth mentioning that the MIT-BIH AR cross database performance is relatively poor, specially for the lead II. Some ECG recordings of MIT-BIH AR lead V1' are V2 or V5, which could lead to a better generalization capability of the lead V1' but also to a worse performance in the intra-lead classification when compared with the other intra-lead performances (see **Table 4**).

3.2. Computational Times

Besides providing a detailed characterization of the arrhythmia heartbeat classifier based on ESNs, our study also aims at achieving computational times that allow for real-time processing of ECG data. In particular, we have implemented the ESN classifiers described here independently in an unparallelized C++ version for the CPU and a C++/CUDA version for the GPU. C++ refers to the object oriented programming language and CUDA is a parallel computing platform developed by the company Nvidia to interface with their GPUs. The specific technical details for the CPU and GPU are summarized in **Table 6**.

TABLE 6 | Technical specifications of the CPU and GPU used in this work.

	CPU	GPU
Processor	Intel(R) Core(TM) i7-4790K	NVIDIA TITAN X Pascal (3584 CUDA cores)
Clock frequency	4,400 MHz	1,417 MHz
Memory	32 GB	12 GB
Max. Mem. Bandwidth	25.6 GB/s	480 GB/s

Although ensembles are inherently independent, making them good candidates for parallel multi-processor implementations, the presence of large matrix products and non-linear mapping functions in the reservoir paradigm also makes serial implementations suitable for the exploration of computationally fast approaches. These approaches, such as GPU implementations, are capable of reducing the latency and increasing the throughput.

In order to explore the computational time and reservoir size (N) dependence, a series of training and classification procedures for the MIT-BIH AR database are analyzed. Linear regressions are carried out by means of lower-upper decomposition. C++ implementations benefit from the Eigen library [1], while C++/CUDA use cuSolver, cuBLAS products and a CUDA kernel implemented for the non-linear mapping.

Figure 6 shows the computational times of a training and a testing realization for the DS1 and DS2 sets of the MIT-BIH AR databases, respectively, vs. the number of neurons. The GPU and CPU comparison highlights the advantage of using a GPU implementation, with significantly lower training times. The depicted computational times include, on the one hand, the random non-linear mapping of the input onto the reservoir and, on the other hand, the calculation of the output weights $\mathbf{W}^{out}$ over the entire train dataset. The insets in **Figure 6** show the computational time for the final classification product steps that calculate the output in the test dataset. As expected, the processing time increases with the number of neurons, especially in the training procedure. The influence of small sized products on cuBLAS scaling, intrinsic to the library, can be seen in the piece-wise linear trend present in the GPU Classification product. The reported computational times account for 11 h of ECG recordings, allowing the exploration of different parameter regimes and providing fast classifications clearly suitable for real-time scenarios that may include statistical ensembles.

3.3. Comparison With Other Heartbeat Arrhythmia Classifiers

The MIT-BIH AR database is by far the most used to evaluate methods on the ventricular arrhythmia classification. However, making a fair comparison between heartbeat classifiers is a difficult task. For instance, classifiers sharing heartbeats for the

[1] Eigen v3.3—Gaël Guennebaud, Benoît Jacob et al. http://eigen.tuxfamily.org/

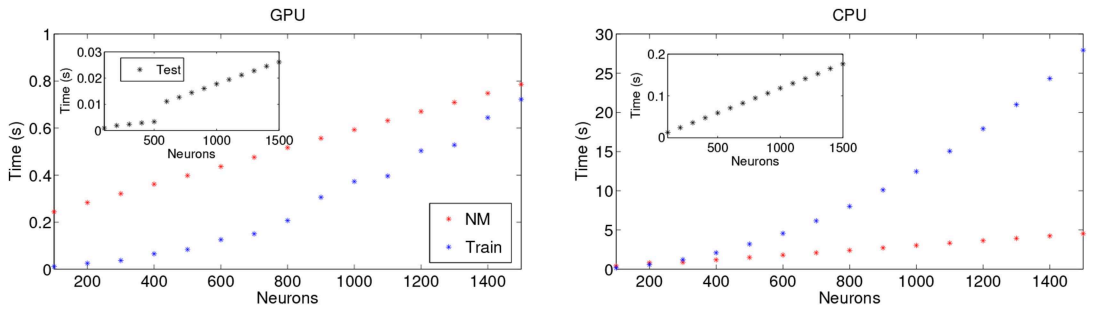


FIGURE 6 | Dependence of the computational times as a function of the number of neurons for **(Left)** GPU and **(Right)** CPU implementations. NM stands for Non-linear Mapping and consists in the input-random matrix multiplication and the application of the non-linear function in accordance with the ESN. The single test and training times shown are over the whole DS1 and DS2 sets of one lead of the MIT-BIH AR database, respectively.

TABLE 7 | VEB performance of the heartbeat classifiers on the MIT-BIH AR database.

				VEB	
Work	**Feature set**	**Classifier**	**Leads**	**Se**	**PPV**
de Chazal et al. [3]	Morphological, RR-intervals	Weighted LD	II + V1'	77.7	81.9
Ye et al. [18]	Morphological, RR-intervals, wavelet, ICA, PCA	SVM	II + V1'	81.5	63.1
Zhang et al. [14]	Morphological, RR-intervals	Feature selection + SVM	II + V1'	85.5	92.8
Mar et al. [4]	Morphological, HOS, temporal features	Feature selection + MLP	II + V1'	86.8	75.9
Garcia et al. [16]	Morphological, wavelets, TVCG	PSO + SVM	II + V1'	87.3	59.4
Llamedo and Martinez [31]	Morphological, RR-interval, VCG, wavelet	LD+ EMC	II + V1'	83.0	88.0
Llamedo and Martinez [33]	RR-interval, wavelet	LD+ EMC	II + V1'	89.0	87.0
Ye et al. [26]	Morphological, RR-intervals, wavelet, ICA	General + specific classification model	II + V1'	91.8	98.0
Tejeiro et al. [43]	Morphological, rhythm features, RR-intervals	Abductive interpretation	II + V1'	94.6	96.8
Ghorbani et al. [7]	Morphological, RR-intervals, statistical features, GMM + EM	Decision trees	II + V1'	96	77.6
Krasteva et al. [5]*	Morphological, RR-intervals, correlations	Decision trees	II + V1'	96.7	99.2
Wu et al. [20]	DBN, RR-intervals	Softmax regression	II	80.5	81.4
Lannoy et al. [19]	Morphological, RR-intervals, HOS, HBF coeff	Weighted conditional random fields	II	85.1	–
Rahhal et al. [22] *	Raw ECG data	Deep neural networks	II	91.0	79.5
Raj et al. [17]	DOST	PSO + SVM	II	87.5	65.4
Sultan Qurraie and Ghorbani Afkhami [6]	RR-interval, HOS, time–frequency	Decision trees	II	95.4	94.1
Herry et al. [44]	RR-interval, SST	SVM	II	77.5	79.1
			V1'	79.6	62.7
Huang et al. [15]	Random projections	SVM ensembles	II	93.9	90.9
			V1'	78.1	43.8
This work	Raw ECG data, RR-intervals	ESN ensembles	II	92.7	95.7
			V1'	86.1	75.1

Only the best fully automatic work result is reported. All the classifiers have been trained over the set DS1 and tested over DS2, except the ones marked with. Rahhal et al. [22] and Krasteva et al. [5] test against all the MIT-BIH AR database. Rahhal et al. [22] trains over the DS1 and Krasteva et al. [5] uses three databases (AHA, MIT-BIH-SV, and EDB) to train the model. See the text for a description of the different methods and features.*

same subjects in the training and test set have unrealistically better evaluation results than classifiers that follow the inter-patient procedure [7]. Semi-automatic heartbeat classifiers (that require some assistance for expert cardiologist) also have a better performance than the fully automatic approaches [33]. Thus, to be as fair as possible, we only compared our method with other fully automatic heartbeat classifiers that make the test over the DS2 set of the MIT-BIH AR database and whose train set does not share subjects with the testing set.

Focusing on the detection of ventricular arrhythmia, we compare the VEB (V) performance instead of the VEB+ (V+F), as the VEB+ performance is usually not reported in the literature. The VEB performance has then been calculated in our algorithm without taking into account the F heartbeats, which are rather rare. **Table 7** compares the VEB detection performance of state-of-the-art algorithms with the method proposed in this manuscript. **Table 7** also provides information about the features and classifiers used by the different approaches. In most cases,

the computational cost of these methods, either during the training or the test phases, is not mentioned. **Table 7** presents a wide variety of methods, such as Multilayer Perceptron (MLP), temporal vectorcardiogram (TVCG), Expectation-maximization clustering algorithm (EMC), Gaussian mixture modeling (GMM), Enhanced expectation maximization (EM), Orthogonal Stockwell Transform (DOST), Deep Belief Networks (DBN), and synchrosqueezing transform (SST).

Our method outperforms or shows state-of-the-art results with methods that used much more complicated procedures to extract and select the heartbeat features for the VEB class. Some of the methods with better performance than the method proposed here are not well-suited for real-time applications, as the feature extraction stage can not be implemented online, such as in [43] or imply a high computational cost [6]. Moreover, our approach outperforms the other single lead classifiers reported for the VEB classification based on the MIT-BIH AR lead V1', showing a better generalization capability than the other methods based on a single lead. Finally, the excessive false alarm rate (low PPV) is a major problem for clinical use since it diminishes the confidence in the algorithm. The approach discussed in this manuscript has the best PPV for the VEB class among the single lead classifiers.

4. DISCUSSION

The proposed method shows excellent classification results for the VEB class on the MIT-BIH AR and the AHA databases, outperforming existing single lead classification algorithms in the detection of ventricular arrhythmia. The presented ESN approach is suitable for processing long-term recordings and large databases as the feature extraction and the algorithm itself both have minimal computational requirements.

Overall, the ESN presents two main advantages over other classical methods that have been used to classify heartbeats, such as the SVM, NN, and decision trees (see **Table 7**). First, the aforementioned methods involve relatively time consuming complex computations in the training phase that in ESN are easily computed. We have checked that the computation times of the classification algorithm for the evaluation of 11 h of ECG recordings amounts to <0.2 s for a lab CPU, while the use of a GPU (see **Table 6**) offers at least a speedup of an order of magnitude. Second, past heartbeats play a role in the classification task in the case of the ESN thanks to its intrinsic memory, having a positive impact on the performance.

In this work, heartbeats are classified as SVEB+ and VEB+. Future work will focus on the extension of these results to the five heartbeat classes recommended by the AAMI. Another important aspect not covered in our study is the fixed heartbeat window length that can be inappropriate in the case of fast and slowly varying heart rhythms when changing physical activity. Thus, there is a need to study adaptive beat size segmentation. The understanding of the exact relation between underlying physiology and features is a potential question to address. However, there are no conclusive guidelines about which features should be used to diagnose arrhythmias from the ECG using computer aided systems.

DATA AVAILABILITY

The MIT-BIH AR publicly available dataset was part of the analysis presented in this work. This database can be found here: https://physionet.org/physiobank/database/mitdb/.

AUTHOR CONTRIBUTIONS

MA implemented the classifier and performed the computational realizations. SO and MS designed and supervised the project. All authors contributed to the discussion of the results and to the writing of the manuscript.

FUNDING

Authors gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X Pascal GPU used for this research. This work was partially funded by the Spanish Ministerio de Economía y Competitividad (MINECO) and Fondo Europeo de Desarrollo Regional (FEDER) and the European Social Fund through project TEC2016-80063-C3-3-R (MINECO/AEI/FEDER/UE). MA was supported by the Beca de colaboración 012/2016 UIB fellowship on Information processing in neural and photonic systems. MS was supported by the Spanish Ministerio de Economía, Industria y Competitividad through a Ramón y Cajal Fellowship (RYC-2015-18140). SO was supported by the Conselleria d'Innovació, Recerca i Turisme del Govern de les Illes Balears and the European Social Fund.

ACKNOWLEDGMENTS

The authors would like to thank Claudio Mirasso, Ingo Fischer, Xavier Ibáñez Català, and Agustín Maciá for valuable scientific discussions.

REFERENCES

1. Goldberger AL, Amaral LAN, Glass L, Hausdorff JM, Ivanov PC, Mark RG, et al. PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. *Circulation.* (2000) **101**:e215–20. doi: 10.1161/01.cir.101.23.e215
2. da S Luz EJ, Schwartz WR, Cámara-Chávez G, Menotti D. ECG-based heartbeat classification for arrhythmia detection: a survey. *Comput Methods Programs Biomed.* (2016) **127**:144–64. doi: 10.1016/j.cmpb.2015.12.008
3. de Chazal P, O'Dwyer M, Reilly RB. Automatic classification of heartbeats using ECG morphology and heartbeat interval features. *IEEE Trans Biomed Eng.* (2004) **51**:1196–206. doi: 10.1109/TBME.2004.827359
4. Mar T, Zaunseder S, Martínez JP, Llamedo M, Poll R. Optimization of ECG classification by means of feature selection. *IEEE Trans Biomed Eng.* (2011) **58**:2168–77. doi: 10.1109/TBME.2011.2113395
5. Krasteva V, Jekova I, Leber R, Schmid R, Abacherli R. Superiority of classification tree versus cluster, fuzzy and discriminant models

in a heartbeat classification system. *PLoS ONE.* (2015) **13**:e0140123. doi: 10.1371/journal.pone.0140123
6. Sultan Qurraie S, Ghorbani Afkhami R. ECG arrhythmia classification using time frequency distribution techniques. *Biomed Eng Lett.* (2017) **7**:325–32. doi: 10.1007/s13534-017-0043-2
7. Ghorbani Afkhami R, Azarnia G, Tinati MA. Cardiac arrhythmia classification using statistical and mixture modeling features of ECG signals. *Pattern Recognit Lett.* (2016) **70**:45–51. doi: 10.1016/j.patrec.2015.11.018
8. Dokur Z, Ölmez T. ECG beat classification by a novel hybrid neural network. *Comput Methods Programs Biomed.* (2001) **66**:167–81. doi: 10.1016/S0169-2607(00)00133-4
9. Elhaj FA, Salim N, Harris AR, Swee TT, Ahmed T. Arrhythmia recognition and classification using combined linear and nonlinear features of ECG signals. *Comput Methods Programs Biomed.* (2016) **127**:52–63. doi: 10.1016/j.cmpb.2015.12.024
10. Martis RJ, Acharya UR, Min LC. ECG beat classification using PCA, LDA, ICA and discrete wavelet transform. *Biomed Signal Process Control.* (2013) **8**:437–48. doi: 10.1016/j.bspc.2013.01.005
11. Inan OT, Giovangrandi L, Kovacs GTA. Robust neural-network-based classification of premature ventricular contractions using wavelet transform and timing interval features. *IEEE Trans Biomed Eng.* (2006) **53**:2507–15. doi: 10.1109/TBME.2006.880879
12. Javadi M, Ebrahimpour R, Sajedin A, Faridi S, Zakernejad S. Improving ECG classification accuracy using an ensemble of neural network modules. *PLoS ONE.* (2011) **6**:e24386. doi: 10.1371/journal.pone.0024386
13. Kiranyaz S, Ince T, Gabbouj M. Real-time patient-specific ECG classification by 1-D convolutional neural networks. *IEEE Trans Biomed Eng.* (2016) **63**:664–75. doi: 10.1109/TBME.2015.2468589
14. Zhang Z, Dong J, Luo X, Choi KS, Wu X. Heartbeat classification using disease-specific feature selection. *Comput Biol Med.* (2014) **46**:79–89. doi: 10.1016/j.compbiomed.2013.11.019
15. Huang H, Liu J, Zhu Q, Wang R, Hu G. A new hierarchical method for inter-patient heartbeat classification using random projections and RR intervals. *Biomed Eng Online.* (2014) **13**:90. doi: 10.1186/1475-925X-13-90
16. Garcia G, Moreira G, Menotti D, Luz E. Inter-patient ECG heartbeat classification with temporal VCG optimized by PSO. *Sci Rep.* (2017) **7**:10543. doi: 10.1038/s41598-017-09837-3
17. Raj S, Ray KC, Shankar O. Cardiac arrhythmia beat classification using DOST and PSO tuned SVM. *Comput Methods Programs Biomed.* (2016) **136**:163–77. doi: 10.1016/j.cmpb.2016.08.016
18. Ye C, Kumar BVKV, Coimbra MT. Heartbeat classification using morphological and dynamic features of ECG signals. *IEEE Trans Biomed Eng.* (2012) **59**:2930–41. doi: 10.1109/TBME.2012.2213253
19. De Lannoy G, François D, Delbeke J, Verleysen M. Weighted conditional random fields for supervised interpatient heartbeat classification. *IEEE Trans Biomed Eng.* (2012) **59**:241–7. doi: 10.1109/TBME.2011.2171037
20. Wu Z, Ding X, Zhang G. A novel method for classification of ECG arrhythmias using deep belief networks. *Int J Comput Intell Appl.* (2016) **15**:1650021. doi: 10.1142/S1469026816500218
21. Acharya UR, Oh SL, Hagiwara Y, Tan JH, Adam M, Gertych A, et al. A deep convolutional neural network model to classify heartbeats. *Comput Biol Med.* (2017) **89**:389–96. doi: 10.1016/j.compbiomed.2017.08.022
22. Rahhal MMA, Bazi Y, Alhichri H, Alajlan N, Melgani F, Yager RR. Deep learning approach for active classification of electrocardiogram signals. *Inf Sci.* (2016) **345**:340–54. doi: 10.1016/j.ins.2016.01.082
23. Li H, Yuan D, Ma X, Cui D, Cao L. Genetic algorithm for the optimization of features and neural networks in ECG signals classification. *Sci Rep.* (2017) **7**:41011. doi: 10.1038/srep41011
24. Ortín S, Soriano MC, Alfaras M, Mirasso CR. Automated real-time method for ventricular heartbeat classification. *Comput Methods Programs Biomed.* (2019) **169**:1–8. doi: 10.1016/j.cmpb.2018.11.005
25. Zidelmal Z, Amirou A, Ould-Abdeslam D, Merckle J. ECG beat classification using a cost sensitive classifier. *Comput Methods Programs Biomed.* (2013) **111**:570–7. doi: 10.1016/j.cmpb.2013.05.011
26. Ye C, Kumar BVK, Coimbra MT. An automatic subject-adaptable heartbeat classifier based on multiview learning. *IEEE J Biomed Health Inform.* (2016) **20**:1485–92. doi: 10.1109/JBHI.2015.2468224
27. Jaeger H, Haas H. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. *Science.* (2004) **304**:78–80. doi: 10.1126/science.1091277
28. Moody GB, Mark RG. The impact of the MIT-BIH arrhythmia database. *IEEE Eng Med Biol Mag.* (2001) **20**:45–50. doi: 10.1109/51.932724
29. Emergency Care Research Institute. *American Heart Association (AHA) Ventricular Arrhythmia ECG Database*. Plymouth Meeting, PA: Emergency Care Research Institute (2003).
30. ANSI/AAMI. *Testing and Reporting Performance Results of Cardiac Rhythm and ST Segment Measurement Algorithms*. Arlington, VA: American National Standards Institute, Inc (ANSI), Association for the Advancement of Medical Instrumentation (AAMI), ANSI/AAMI/ISO EC57, 1998-(R)2008 (2008).
31. Llamedo M, Martinez JP. Heartbeat classification using feature selection driven by database generalization criteria. *IEEE Trans Biomed Eng.* (2011) **58**:616–25. doi: 10.1109/TBME.2010.2068048
32. Martinez JP, Almeida R, Olmos S, Rocha AP, Laguna P. A wavelet-based ECG delineator: evaluation on standard databases. *IEEE Trans Biomed Eng.* (2004) **51**:570–81. doi: 10.1109/TBME.2003.821031
33. Llamedo M, Martinez JP. An automatic patient-adapted ECG heartbeat classifier allowing expert assistance. *IEEE Trans Biomed Eng.* (2012) **59**:2312–20. doi: 10.1109/TBME.2012.2202662
34. Lukoševičius M, Jaeger H. Reservoir computing approaches to recurrent neural network training. *Comput Sci Rev.* (2009) **3**:127–49. doi: 10.1016/j.cosrev.2009.03.005
35. Ortín S, Pesquera L. Reservoir computing with an ensemble of time-delay reservoirs. *Cognit Comput.* (2017) **9**:327–36. doi: 10.1007/s12559-017-9463-7
36. Lukoševičius M. A practical guide to applying echo state networks. In: Montavon G, Orr GB, Müller KR, editors. *Neural Networks: Tricks of the Trade*. 2nd ed. Berlin; Heidelberg: Springer Berlin Heidelberg (2012). p. 659–86. doi: 10.1007/978-3-642-35289-8_36
37. Rodan A, Tino P. Minimum complexity echo state network. *IEEE Trans Neural Netw.* (2011) **22**:131–44. doi: 10.1109/TNN.2010.2089641
38. Appeltant L, Soriano MC, Van der Sande G, Danckaert J, Dambre J, Schrauwen B, et al. Information processing using a single dynamical node as complex system. *Nat Commun.* (2011) **2**:468. doi: 10.1038/ncomms1476
39. Paquot Y, Duport F, Smerieri A, Dambre J, Schrauwen B, Haelterman M, et al. Optoelectronic reservoir computing. *Sci Rep.* (2012) **2**:287. doi: 10.1038/srep00287
40. Brunner D, Soriano MC, Mirasso C, Fischer I. Parallel photonic information processing at gigabyte per second data rates using transient states. *Nat Commun.* (2013) **4**:1364. doi: 10.1038/ncomms2368
41. Ortín S, Soriano MC, Pesquera L, Brunner D, San-Martín D, Fischer I, et al. A unified framework for reservoir computing and extreme learning machines based on a single time-delayed neuron. *Sci Rep.* (2015) **5**:14945. doi: 10.1038/srep14945
42. Osowski S, Linh TH. ECG beat recognition using fuzzy hybrid neural network. *IEEE Trans Biomed Eng.* (2001) **48**:1265–71. doi: 10.1109/10.959322
43. Teijeiro T, Felix P, Presedo J, Castro D. Heartbeat classification using abstract features from the abductive interpretation of the ECG. *IEEE Journal of Biomedical and Health Inform.* (2018) **22**:409–20. doi: 10.1109/JBHI.2016.2631247
44. Herry CL, Frasch M, Seely AJE, Wu HT. Heart beat classification from single-lead ECG using the synchrosqueezing transform. *Physiol Meas.* (2017) **38**:171–87. doi: 10.1088/1361-6579/aa5070

Conflict of Interest Statement: MA is currently employed by company PLUX S.A. This research was entirely conducted while he was an IFISC researcher.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Physics

ORIGINAL RESEARCH
published: 03 October 2019
doi: 10.3389/fphy.2019.00138

Distributed Kerr Non-linearity in a Coherent All-Optical Fiber-Ring Reservoir Computer

Jaël Pauwels[1,2]*, *Guy Verschaffelt*[1], *Serge Massar*[2] *and Guy Van der Sande*[1]

[1] *Applied Physics Research Group, Vrije Universiteit Brussel, Brussels, Belgium,* [2] *Laboratoire d'Information Quantique, Université Libre de Bruxelles, Brussels, Belgium*

We investigate, both numerically and experimentally, the usefulness of a distributed non-linearity in a passive coherent photonic reservoir computer. This computing system is based on a passive coherent optical fiber-ring cavity in which part of the non-linearities are realized by the Kerr non-linearity. Linear coherent reservoirs can solve difficult tasks but are aided by non-linear components in their input and/or output layer. Here, we compare the impact of non-linear transformations of information in the reservoirs input layer, its bulk—the fiber-ring cavity—and its readout layer. For the injection of data into the reservoir, we compare a linear input mapping to the non-linear transfer function of a Mach Zehnder modulator. For the reservoir bulk, we quantify the impact of the optical Kerr effect. For the readout layer we compare a linear output to a quadratic output implemented by a photodiode. We find that optical non-linearities in the reservoir itself, such as the optical Kerr non-linearity studied in the present work, enhance the task solving capability of the reservoir. This suggests that such non-linearities will play a key role in future coherent all-optical reservoir computers.

Keywords: photonic, reservoir computing, passive, coherent, distributed non-linearity, Kerr, fiber-ring

OPEN ACCESS

Edited by:
Claudio Mirasso,
Institute of Interdisciplinary Physics and Complex Systems (IFISC), Spain

Reviewed by:
Apostolos Argyris,
Institute of Interdisciplinary Physics and Complex Systems (IFISC), Spain
Vasileios Basios,
Free University of Brussels, Belgium
Luis Pesquera,
University of Cantabria, Spain

***Correspondence:**
Jaël Pauwels
jael.pauwels@vub.be

Specialty section:
This article was submitted to Interdisciplinary Physics, a section of the journal Frontiers in Physics

Received: *23 May 2019*
Accepted: *06 September 2019*
Published: *03 October 2019*

Citation:
Pauwels J, Verschaffelt G, Massar S and Van der Sande G (2019) Distributed Kerr Non-linearity in a Coherent All-Optical Fiber-Ring Reservoir Computer. Front. Phys. 7:138. doi: 10.3389/fphy.2019.00138

1. INTRODUCTION

In this work, we discuss an efficient, i.e., high speed and low power, analog photonic computing system based on the concept of reservoir computing (RC) [1, 2]. This framework allows to exploit the transient dynamics of a non-linear dynamical system for performing useful computations. In this neuromorphic computing scheme, a network of interconnected computational nodes (called neurons) is excited with input data. The ensemble of neurons is called the reservoir, and the interneural connections are fixed and can be chosen at random. For the coupling of the input data to the reservoir an input mask is used: a set of input weights which determines how strongly each of the inputs couples to each of the neurons. The randomness in both the input mask and internal reservoir connections ensures diversity in the neural responses. The reservoir output is constructed through a linear combination of neural responses (possibly first processed by a readout function) with a set of readout weights. The strength of the reservoir computing scheme lies in the simplicity of its training method, where only the readout weights are tuned to force the reservoir output to match a desired target. In general, a reservoir exhibits internal feedback through loops in the neural interconnections. As a result any reservoir has memory, which means it can retain input data for a finite amount of time, and it can compute linear and non-linear functions of the retained information.

Within the field of reservoir computing two main approaches exist: in the network-based approach networks of neurons are implemented by connecting multiple discrete nodes [3], and in the delay-based approach networks of virtual neurons are created by subjecting a single node (often a non-linear dynamical device) to delayed feedback [4]. In the latter, the neurons are called virtual because they correspond with the traveling signals found in consequent timeslots in the continuous delay-line system. On account of this time-multiplexing of neurons, the input weights are translated into a temporal input mask, which is mixed with the input data before it is injected into the reservoir. Besides ensuring diversity in the neural responses, this input mask also keeps the virtual neurons in a transient dynamic regime, which is a necessary condition for good reservoir computing performance.

Multiple opto-electronic reservoirs have been implemented, both delay-based [5–8] and network-based [9]. Several all-optical reservoirs have been realized, both network-based systems [9–13] and delay-based systems [14–16]. An overview of recent advances is given in reference [17]. We observe that in the field of optical reservoir computing, some implementations operated in an incoherent regime, while others operated in a coherent regime. Coherent reservoirs have the advantage that they can exploit the complex character of the optical field, exploit interferences, and can use the natural quadratic non-linearity of photodiodes. As a drawback, coherent bulk optical reservoirs typically need to be stabilized, but this is not a problem for on chip implementations. Here we investigate the potential advantage of having a coherent reservoir with non-linearity inside the reservoir. We show that it can increase the performance of the reservoir on certain tasks and we expect that future coherent optical reservoir computers will make use of such non-linearities.

State of the art photonic implementations target simple reservoir architectures [13], which can easily be upscaled to increase the number of computational nodes or neurons, thereby enhancing the reservoirs computational capacity. Even a linear photonic cavity can be a potent reservoir [16], provided that some non-linearity is present either in the mapping of input data to the reservoir, or in the readout of the reservoirs response. Despite advances toward all-optical RC [18], many state of the art photonic reservoir computers inherently contain some non-linearity as they are usually set up to process and produce electronic signals. This means that even if the reservoir is all-optical, the reservoir computer in its entirety is of an opto-electronic nature. Commonly used components like a Mach-Zehnder modulators (MZM) and photodetectors (PD) provide means for transitioning back and forth between the electronic and optical domains, and they also—almost inevitably—introduce non-linearities which boost the opto-electronic reservoir computers performance beyond the merits of the optical reservoir itself. When transitioning toward all-optical reservoir computers, such non-linearities can no longer be relied on, and thus the required non-linear transformation of information must originate elsewhere. One option is then to use multiple strategically placed non-linear components in the reservoir, but this can be a costly strategy when upscaling the reservoir [10].

In this paper, we study a delay-based reservoir computer, based on a passive coherent optical fiber ring cavity following reference [16] and exploit the inherent non-linear response of the waveguiding material to build a state-of-the-art photonic reservoir. This means that the non-linearity of our photonic reservoir is not found in localized parts, but rather it is distributed over the reservoirs entire extent. To correctly characterize the effects of such distributed non-linearity, we also consider in this study all other non-linearities that may surround the reservoir. In terms of the reservoirs input mapping, we examined the system responses when receiving optical inputs (linear mapping), and when receiving electronic inputs coupled to the optical reservoir through a Mach-Zehnder modulator with a non-linear mapping. For the reservoirs readout layer, we examined both linear readouts (coherent detection) and non-linear readouts through the quadratic non-linearity of a photodiode measuring the power of the optical field. Taking these different options into account, we then constructed different scenarios in terms of the presence of non-linearities in the input and/or output layer of these reservoir computers. In all these scenarios we numerically benchmarked the RC performance, thus quantifying the difference in performance between systems which do or do not have such distributed non-linearity inside the reservoir. In the next sections, we show our numerical results, which show a broad range of optical input power levels at which these RCs benefit from the self-phase modulation experienced by the signals due to the non-linear Kerr effect induced by the waveguide material. We also show the results of our experimental measurements that indicate how much this distributed non-linearity boosts the reservoir's capacity to perform non-linear computation. In the discussion section, we analyze the impact of these findings on the future of photonic reservoir computing.

2. MATERIALS AND METHODS

2.1. Setup

Our reservoir computing simulations and experiments are based on the set of dynamical systems which are discussed in this section. The reservoir itself is implemented in the all-optical fiber-ring cavity shown in **Figure 1**, using standard single-mode fiber. A polarization controller is used to ensure that the input field E_{in} (originating from the green arrow) excites a polarization eigenmode of the fiber-ring cavity. A fiber coupler, characterized by its power transmission coefficient $T = 50\%$, couples light in and out of the cavity. The fiber-ring is characterized by the roundtrip length $L = 10\,\mathrm{m}$ (or roundtrip time t_R), the propagation loss α (taken here $0.18\,\mathrm{dB\,km^{-1}}$), the fiber non-linear coefficient γ (which is set to 0 to simulate a linear reservoir, and set to $\gamma_{Kerr} = 2.6\,\mathrm{mrad\,m^{-1}\,W^{-1}}$ to simulate a non-linear reservoir), and the cavity detuning δ_0, i.e., the difference between the roundtrip phase and the nearest resonance (multiple of 2π). This low-finesse cavity is operated off-resonance, with a maximal input power of 50 mW (17 dBm). A network of time-multiplexed virtual neurons is encoded in the cavity field envelope. The output field E_{out} is sent to the readout

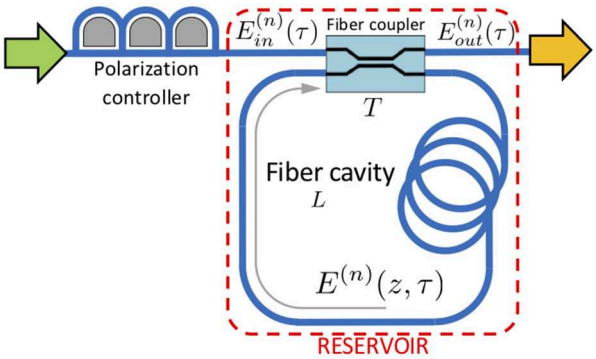


FIGURE 1 | Schematic of the fiber-ring cavity of length L used to implement an optical reservoir. The green (orange) arrow indicates a connection with an input (output) layer. A polarization controller maps the input polarization onto a polarization eigenmode of the cavity. A coupler with power transmission coefficient T couples the input field $E_{in}^{(n)}(\tau)$ to the cavity field $E^{(n)}(z,\tau)$ and couples to the output field $E_{out}^{(n)}(\tau)$, where n is the roundtrip index, τ is time (with $0 < \tau < t_R$) and z is the longitudinal position in the ring cavity.

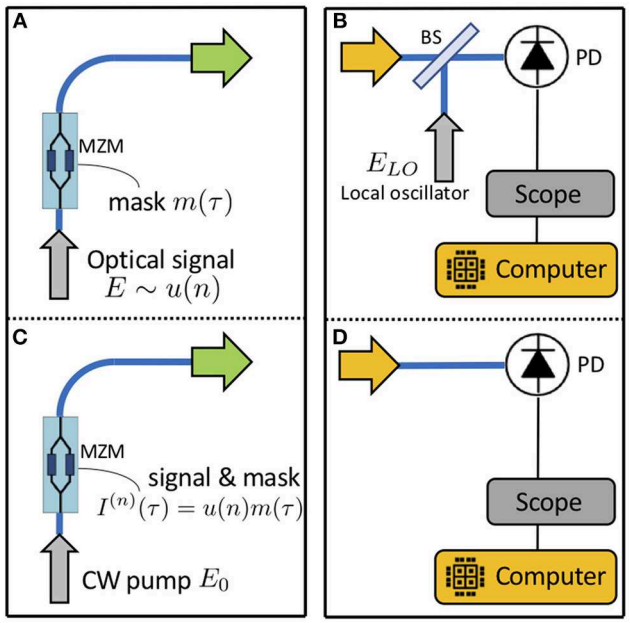


FIGURE 2 | Schematics of input and output layers connecting to the reservoir shown in **Figure 1**. In the linear input scheme **(A)** the Mach-Zehnder modulator (MZM) superimposes the reservoir's input mask $m(\tau)$ on the optical signal $E \sim u(n)$ carrying the input data. In the (possibly) non-linear input scheme **(B)** the input data is mixed with the input mask and then drives the MZM to modulate a CW optical pump. In the linear output scheme **(C)** a reference field E_{LO} is used to implement coherent detection, allowing a quadrature of the complex optical field to be measured. Note that coherent detection requires two such readout arms with phase-shifted reference fields in order to measure the complex output field E_{out}. In the non-linear output scheme **(D)** only a photodetector (PD) is used, thus only allowing the optical output power $|E_{out}|^2$ to be recorded.

layer (through the orange arrow) where the neural responses are demultiplexed.

The input field E_{in} can originate from one of two different optoelectronic input schemes. Firstly we consider a scenario where the input signal $u(n)$ (with discrete time n) is amplitude-encoded in an optical signal $E \sim u(n)$, as shown in **Figure 2A**. The reservoir's input mask $m(\tau)$ is mixed with the input signal by periodic modulation of the optical input signal using an MZM. This scheme was implemented in reference [7], but the non-linearity of the MZM was avoided through pre-compensation of the electronic input signal. Note that the discrete time n corresponds with the roundtrip index. And as delay-based reservoirs are typically set up to process 1 sample each roundtrip, n also corresponds with the sample index. However, we have chosen to hold each input sample over multiple roundtrips, for reasons which are explained in the Results section [that is, $u(n)$ is constant over multiple values of n]. Secondly we consider a scenario where we use the MZM to modulate a CW optical pump following reference [14], as shown in **Figure 2B**. Here the input signal is first mixed with the input mask and then used to drive the MZM. It is known that the MZM's non-linear transfer function can affect the RC system's performance [16], but the implications for a coherent non-linear reservoir have not yet been investigated.

Similarly, the output field E_{out} can be processed by two different optoelectronic readout schemes. Firstly we consider a coherent detection scheme as shown in **Figure 2C**. Mixing the reservoir's output field with a reference field E_{LO} allows to record the complex neural responses, time-multiplexed in the output field E_{out}. Secondly, we consider a readout scheme where a photodetector (PD) measures the optical power of the neural responses $|E_{out}|^2$, as shown in **Figure 2D**.

With high optical power levels and small neuron spacing (meaning fast modulation of the input signal), dynamical and non-linear effects other than the Kerr non-linearity may appear, such as photon-phonon interactions causing Brillouin and Raman scattering, and bandwidth limitations caused by the driving and readout equipment. We want to focus in the present work on the effects of the Kerr non-linearity. Combined with the memory limitations of the oscilloscope, we therefore limit our reservoir to 20 neurons, with a maximal input power of 100 mW.

The current setup is not actively stabilized. We have found that the cavity detuning δ_0 does not vary more than a few mrad over the course of any single reservoir computing experiment, where a few thousand input samples are processed. A short header, added to the injected signal, allows us to recover the detuning δ_0 post-experiment. We effectively measure the interference between a pulse which reflects off the cavity and a pulse which completes one roundtrip through the cavity. However, we find that the precise value of δ_0 has no significant influence on the experimental reservoir computing results.

2.2. Physical Model

Here we discuss the mean-field model used to describe the temporal evolution of the electric field envelope $E^{(n)}(z,\tau)$ inside the cavity, where n is the roundtrip index, $0 < \tau < t_R$ is time (bound by the cavity roundtrip time t_R and $0 < z < L$ is the longitudinal coordinate of the fiber ring cavity with length L. The position $z = 0$ corresponds to the position of the fiber

coupler. The position $z = L$ corresponds to the same position, but after propagation through the entire fiber-ring. We will describe the evolution on a per-roundtrip basis (i.e., with varying roundtrip index n). With this notation $E^{(n)}(z,\tau)$ represents the cavity field envelope measured at position z at time τ during the n-th roundtrip. For each roundtrip we model propagation through the non-linear cavity to obtain $E^{(n)}(z = L,\tau)$ from $E^{(n)}(z = 0,\tau)$. We then express the cavity boundary conditions to obtain $E^{(n+1)}(0,\tau)$ from $E^{(n)}(L,\tau)$ and to obtain the field $E^{(n)}_{out}(\tau)$ at the output of the fiber-ring reservoir. For now we will omit τ.

Firstly, to model propagation in the fiber-ring cavity we take into account propagation loss and the non-linear Kerr-effect. Since the non-linear propagation model is independent from the roundtrip index n, this subscript is omitted in the following description. The non-linear propagation equation is given by

$$\partial_z E = i\gamma|E|^2E - \alpha E. \tag{1}$$

Here, α is the propagation loss and γ is the non-linear coefficient which is set to $\gamma = 0$ to simulate a linear reservoir, and set to $\gamma = \gamma_{Kerr}$ to include the non-linear Kerr effect caused by the fiber waveguide. We do not include dispersion effects at the current operating point of the system, since the neuron separation is much larger than the diffusion length, hence also τ can be omitted in the non-linear propagation model. The evolution of the power $|E(z)|^2$ is readily obtained by solving the corresponding propagation equation

$$\partial_z|E|^2 = E^*\partial_z E + E\partial_z E^* = -2\alpha|E|^2, \tag{2}$$

$$|E(z)|^2 = |E(0)|^2e^{-2\alpha z}. \tag{3}$$

With ϕ_z the non-linear phase acquired during propagation over a distance z, we know that the solution of $E(z)$ will be of the form

$$E(z) = E(0)e^{i\phi_z - \alpha z}. \tag{4}$$

Since this non-linear phase depends on the power evolution given by Equation (2), an expression for ϕ_z is found to be

$$\phi_z = \gamma\int_0^z |E(v)|^2\delta v = \gamma|E(0)|^2\int_0^z e^{-2\alpha v}\delta v = \gamma|E(0)|^2\frac{1-e^{-2\alpha z}}{2\alpha}. \tag{5}$$

At this point, we can introduce the effective propagation distance z_{eff} as

$$z_{eff} = \frac{1-e^{-2\alpha z}}{2\alpha}. \tag{6}$$

In general (since $\alpha \geq 0$) we have $z_{eff} \leq z$. Substituting these result in Equation (4) yields the complete solution for propagation of the cavity field envelope

$$E(z) = E(0)\exp\left(i\gamma|E(0)|^2z_{eff} - \alpha z\right). \tag{7}$$

Finally, we reinstitute the roundtrip index n and the time parameter τ which allows us to combine this non-linear propagation model with the cavity boundary conditions.

$$\begin{cases} E^{(n)}(L,\tau) = E^{(n)}(0,\tau)\exp\left(i\gamma|E^{(n)}(0,\tau)|^2L_{eff} - \alpha L\right) \\ E^{(n+1)}(0,\tau) = \sqrt{T}E^{(n+1)}_{in}(\tau) + \sqrt{1-T}e^{i\delta_0}E^{(n)}(L,\tau) \\ E^{(n+1)}_{out}(\tau) = \sqrt{1-T}E^{(n+1)}_{in}(\tau) + \sqrt{T}e^{i\delta_0}E^{(n)}(L,\tau) \end{cases} \tag{8}$$

In these equations, T represents the power transmission coefficient of the cavity coupler, and δ_0 represents the cavity detuning (i.e., difference between the roundtrip phase and the closest cavity resonance). Further, the input field $E_{in} = E^{(n)}_{in}(\tau)$ changes with the roundtrip index n as new data samples can be injected into the system, and is modulated in time using the input mask to create a network of virtual neurons. The output field $E_{out} = E^{(n)}_{out}(\tau)$ containing the neural responses is sent to a measurement stage.

2.3. Reservoir Computing

The framework of reservoir computing allows to exploit the transient non-linear dynamics of a dynamical system to perform useful computation [1, 2]. For the purpose of reservoir computing, virtual neurons (dynamical variables, computational nodes) are time-multiplexed in τ-space of the physical system described by Equation (8), following the delay-based reservoir computing scheme originally outlined in reference [4]. As such, the input field $E^{(n)}_{in}(\tau)$ varies with n as new input samples arrive, and varies with τ to implement the input mask, which excites the neurons into a transient dynamic regime. Subsequently, the neural responses are encoded in the output field $E^{(n)}_{out}(\tau)$ and need to be demultiplexed from τ-space. As in references [5, 16] the length t_M of the input mask $m(\tau)$ is deliberately mismatched from the cavity roundtrip time t_R. Instead, we set $t_M = t_RN/(N+1)$ which provides interconnectivity between the N virtual neurons in a ring topology. The input mask $m(\tau)$ is a piecewise constant function, with intervals of duration $\theta = t_M/N$. The signal $I^{(n)}(\tau)$ injected into the RC is constructed by multiplying the input series $u(n)$ with the input mask, $I^{(n)}(\tau) = u(n)m(\tau)$. When the input is coupled linearly to the reservoir then $E^{(n)}_{in}(\tau) \sim I^{(n)}(\tau)$. This would be the case when $u(n)$ is an optical signal periodically modulated with the input mask signal $m(\tau)$. When a MZM modulator with transfer function f is used to convert the electronic signal $I^{(n)}(\tau)$ to the optical domain then $E^{(n)}_{in}(\tau) \sim f(I^{(n)}(\tau))$, where f can be non-linear.

Note that in reference [16] the sample duration t_S is matched to the length of the input mask t_M, allowing the reservoir to process 1 input sample approximately every roundtrip, as $t_S = t_M \lesssim t_R$. However, for reasons explained in the Results section, we will study different sample durations by holding input samples over multiple durations of the input mask, $t_S = k\,t_M$ with integer k as illustrated in **Figure 3**. This inevitably slows the reservoir down, as it only processes 1 input sample approximately every k roundtrips. But it also provides practically straightforward means to accumulate more non-linear processing of the data inside the reservoir, which can then be measured and quantified.

Since the virtual neurons are time-multiplexed in this delay-based reservoir computer, they need to be de-multiplexed from $E^{(n)}_{out}(\tau)$ in the readout layer by sampling this output field at a set of times $\{\tau_i\}$ (with i the neuron index and $1 < i < N$ when N neurons are used) as shown in **Figure 3**. The dynamical neural responses $x_i(n) = E^{(n)}_{out}(\tau_i)$ are recorded and used to train the reservoir to perform a specific task. That is, we optimize a

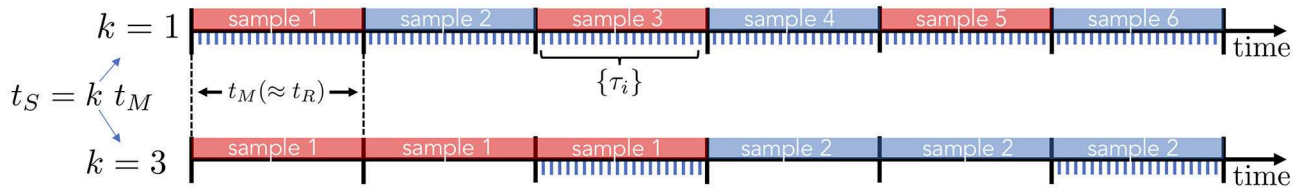


FIGURE 3 | Schematic of input and output timing, with t_S the sample duration, t_M the input mask duration and t_R the roundtrip time. Input samples are injected during (integer) k roundtrips (bars in alternating colors) and the neural responses are recorded at times $\{\tau_i\}$ (blue tick marks) during the last of those k roundtrips.

set of readout weights w_i which are used to combine the neural readouts into a single scalar reservoir output $y(n)$. In general the reservoir output is constructed as

$$y(n) = \sum_{i=1}^{N} w_i g(x_i(n)) \tag{9}$$

where the neural responses $x_i(n)$ are first parsed by an output function $g(x)$ taking into account the operation of the readout layer and readout noise ν. In all simulations the fixed level of readout noise is matched to the experimental conditions. When the complex-valued reservoir states are directly recorded, then $g(x) = x + \nu$ and the readout weights w_i are complex too, such that y is real. If however, a PD measures the power of the neural responses, then $g(x) = |x|^2 + \nu$ which is real-valued, and the readout weights will be real-valued too. Tasks are defined by the real-valued target output $\hat{y}$. Optimization of the readout weights occurs over a training set of T_{train} input and target samples, and is achieved through least squares regression. This procedure minimizes the mean squared error between the reservoir output y and target output $\hat{y}$, averaged over all samples.

$$\{w_i\} = \arg\min_{\{w_i\}} \langle \left(\hat{y} - \sum_{i=1}^{N} w_i g(x_i) \right)^2 \rangle_{T_{train}}. \tag{10}$$

These optimized readout weights are then validated on a test set of T_{test} new input and target samples. A common figure of merit to quantify the reservoir's performance is the normalized mean square error (NMSE) defined as

$$NMSE(y, \hat{y}) = \frac{\langle (y - \hat{y})^2 \rangle_{T_{test}}}{\langle \hat{y}^2 \rangle_{T_{test}}}. \tag{11}$$

2.4. Balanced Mach-Zehnder Modulator Operation

Here we briefly investigate the relevant non-linearities which occur when mapping an electronic signal to an optical signal using an MZM. The operation of our balanced MZM can be described as

$$\frac{E_{in}}{E_0} = \cos\left(\frac{V}{V_\pi} \frac{\pi}{2} \right) \tag{12}$$

where E_0 represents the incident CW pump field, E_{in} is the transmitted field which will be the input field to the optical reservoir, V_π determines at which voltage the zero intensity point occurs (point of no transmission), and V is the voltage of the applied electronical signal consisting of a bias contribution V_b and a zero-mean signal V_s, i.e., $V = V_b + V_s$. For our numerical investigation, we will set the amplitude of the signal voltage to $|V_s| = V_\pi/2$. First, we investigate the zero intensity bias point, $V_b = V_\pi$. In this case, we can approximate Equation (12) with the following Taylor expansion

$$\frac{E_{in}}{E_0} = f(V_s) + O\left(V_s^5\right) \tag{13}$$

$$f(V_s) = -\frac{\pi}{2V_\pi} V_s + \frac{1}{6} \left(\frac{\pi}{2V_\pi} \right)^3 V_s^3 \tag{14}$$

With $(E_{in}/E_0)_{max}$ representing the maximal value of $\frac{E_{in}}{E_0}$ with the given bias voltage V_b and signal amplitude $|V_s|$, the relative error *r.e.* of the Taylor expansion (14)

$$r.e. = \frac{|\frac{E_{in}}{E_0} - f(V_s)|}{\left(\frac{E_{in}}{E_0}\right)_{max}} \tag{15}$$

is smaller than 1%. When the cubic term ($\sim V_s^3$) of the approximation $f(V_s)$ is omitted, this error increases to 11%. This means that at this operating point of the MZM, there is a significant non-linearity which scales with the input signal cubed.

Next, we investigate the linear intensity operating point, $V_b = V_\pi/2$. Although the MZM's transfer function at this operating point is the most linear in terms of the transmitted optical power, it is highly non-linear in terms of the transmitted optical field. In this case, we replace Equation (14) with

$$f(V_s) = \frac{1}{\sqrt{2}} \left(1 - \frac{\pi}{2V_\pi} V_s + \frac{1}{2} \left(\frac{\pi}{2V_\pi} \right)^2 V_s^2 + \frac{1}{6} \left(\frac{\pi}{2V_\pi} \right)^3 V_s^3 + \frac{1}{24} \left(\frac{\pi}{2V_\pi} \right)^4 V_s^4 \right), \tag{16}$$

as we need all polynomial terms up to order 4 to keep the relative error defined by Equation (15) below 1%. In this case, omitting terms of orders above 1 in the approximation $f(V_s)$ increases the relative error of the Taylor expansion to 26%. This means that at this operating point of the MZM there are multiple polynomial non-linearities and that the total non-linear signal distortion is stronger compared with the zero intensity bias point.

Furthermore, during our experiments we have decided to operate the MZM in a linear regime. This allows for the non-linear effects inside the reservoir to be more readily measured. To this end, we tuned the MZM close to the zero intensity operating point, $V_b = V_\pi - \delta_V$ with $\delta_V \ll V_\pi$ and reduced the signal amplitude $|V_s|$. The small deviation δ_V is used to generate a bias in the optical field injected into the reservoir.

2.5. Memory Capacities

To benchmark the performance of an RC, one can train it to perform one or several benchmark tasks. Alternatively, there exists a framework to quantify the system's total information processing capacity. This capacity is typically split into two main parts: the capacity of the system to retain past input samples is captured by the linear memory capacity [19], and the capacity of the system to perform non-linear computation is captured by the non-linear memory capacity [20]. It is known that the total memory capacity has an upper bound given by the number of dynamical variables in the system, which in our system is the number of neurons in the reservoir. It is also known that readout noise reduces this total memory capacity, and that there is a trade-off between linear and non-linear memory capacity, depending on the operating regime of the dynamical system. In order to measure these capacities for our reservoir computer a series of independent and identically distributed input samples $u(n)$ drawn uniformly from the interval $[-1, 1]$ is injected into the reservoir, with discrete time n. The RC is subsequently trained to reconstruct a series of linear and non-linear polynomial functions depending on past inputs $u(n-i)$, looking back i steps in the past. Following reference [20] these functions are chosen to be Legendre polynomials $P_d(u)$ (of degree d), because they are orthogonal over the distribution of the input samples. As an example, we can train the reservoir to reproduce the target signal $\hat{y}(n)$, given by

$$\hat{y}(n) = P_2(u(n-1))P_1(u(n-3)). \tag{17}$$

The ability of the RC to reconstruct each of these functions is evaluated by comparing the reservoir's trained output y with the target $\hat{y}$ for previously unseen input samples. This yields a memory capacity C which lies between 0 and 1 [20],

$$C = 1 - \frac{\langle (\hat{y} - y)^2 \rangle}{\langle \hat{y}^2 \rangle}, \tag{18}$$

where $\langle . \rangle$ denotes the average over all samples used for the evaluation of C. Due to the orthogonality of the polynomial functions over the distribution of the input samples, the capacities corresponding to different functions yield independent information and can thus be summed to quantify the total memory capacity, i.e., the total information processing capacity of the RC. The memory functions are typically grouped by their total degree, which is the sum of degrees over all constituent polynomial functions, e.g., Equation (17) has total degree 3. Summing all memory capacities corresponding with functions of identical total degree yields the total memory capacity per degree. This allows to quantify the contributions of individual degrees to the total memory capacity of the RC, which is the sum over all degrees. As the memory capacities will become small for large degrees, the total memory capacity is still bound.

Since the reservoirs are trained and their performance is evaluated on finite data sets, we run the risk of overestimating the memory capacities C, whose estimator Equation (18) is plagued by a positive bias [20]. Therefore, a cutoff capacity C_{co} is used ($C_{co} \approx 0.1$ for 1,000 test samples) and capacities below this cutoff are neglected (i.e., they are assumed to be 0).

Note that the trade-off between linear and non-linear memory capacity is typically evaluated by comparing the total memory capacity of degree 1 (linear) with the total memory capacity of all higher degrees (non-linear). However, special attention is due when a PD is present in the readout layer of our RC. If a reservoir can (only) linearly retain past inputs $u(n-i)$ (i steps in the past) then any neural response $x(n)$ consists of a linear combination (with a bias term b and fading coefficients a_i) of those past inputs

$$x(n) = b + \sum_i a_i u(n-i) \tag{19}$$

and subsequently the optical power P_x measured by the PD is given by

$$P_x(n) = x(n)\bar{x}(n) = |b|^2 + \sum_i 2\mathrm{Re}(b\bar{a}_i)u(n-i) + \sum_{i,j} 2\mathrm{Re}(a_i\bar{a}_j)u(n-i)u(n-j) \tag{20}$$

which consists of polynomial functions of past inputs of degrees 1 and 2. Thus, in this case the total linear memory capacity of the RC is represented by the total memory capacity of degrees 1 and 2 combined. In case the bias term b is lacking, only memory capacities of degree 2 will be present. On the other hand, if a PD is used in the output and memory capacities of degree higher than 2 are present, then this indicates that the reservoir itself is not linear, i.e., cannot be represented by a function of the form Equation (19).

3. RESULTS

3.1. Numerical RC Performance: Sante Fe Time Series Prediction

For the injection of input samples to the optical reservoir, we consider two strategies as discussed in section 2.1 and in **Figures 2A,B**, referred to here as the linear and non-linear input regimes, respectively. The exact shape of the non-linearity in the non-linear regime depends, among other things, on the operating point (or bias voltage) of the MZM, as discussed in section 2.4. We will demonstrate this by showing results around both the linear intensity operating point and the zero intensity operating point of the MZM. For the readout of the reservoir response, we also consider two cases as discussed in section 2.1 and in **Figures 2C,D**, referred to here as the linear and non-linear output regimes, respectively.

We have thus identified four different scenarios based on the absence or presence of non-linearities in the input and output layer of the reservoir computer. As we will show, we have for

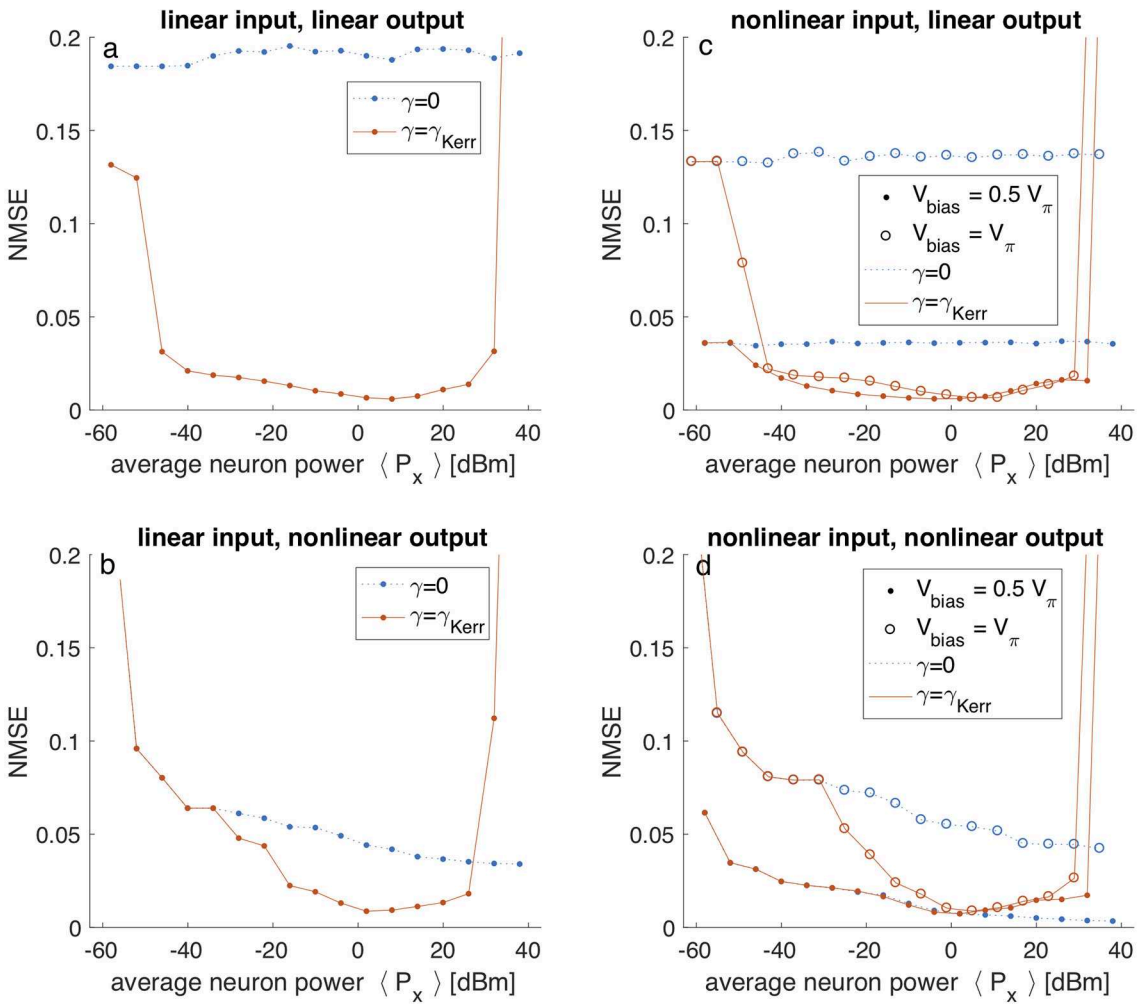


FIGURE 4 | Numerical results of fiber-ring reservoir computer on Santa Fe time series prediction tasks. In all panels the prediction error (NMSE) is plotted vs. the average neuron power $\langle P_x \rangle$. **(a,b)** Correspond with a linear input layer, where **(c,d)** correspond with a non-linear input layer using the MZM's non-linear transfer function. The non-linear input regime shows results for two different operating points of the MZM with different strengths of non-linear transformation. **(a,c)** Correspond with a linear output layer, where **(b,d)** correspond with a non-linear output layer using the PD.

each of these cases numerically investigated the effect of the distributed non-linear Kerr effect, present in the fiber waveguide, on RC performance. For this evaluation, we have used 100 neurons to solve the Santa Fe time series prediction task [21] and each input sample is injected during six roundtrips ($t_S = kt_M$ with $k = 6$) for reasons which will become clear in section 3.2. Here, a pre-existing signal generated by a laser operating in a chaotic regime is injected into the reservoir. The target at each point in time is for the reservoir computer to predict the next sample. Performance is evaluated using the NMSE, where lower is better. **Figure 4** has four panels corresponding to these four scenario's. Each panel shows the NMSE as function of the average optical power per neuron inside the cavity. Dashed blue lines correspond with simulation results of linear reservoirs (i.e., with the non-linear coefficient γ set to 0), and full red lines correspond with simulation results of reservoirs with Kerr non-linear waveguides (i.e., γ set to γ_{Kerr}).

In **Figure 4a** both the input and output layers of the reservoir are strictly linear (i.e., optical input and coherent detection). It is clear that the linear reservoir ($\gamma = 0$) scores poorly, with the NMSE approaching 20%. For a wide range of optical power levels, the presence of the Kerr non-linear effect ($\gamma = \gamma_{Kerr}$) induced by the fiber waveguide boosts the RC performance, with an optimal NMSE just below 1%. This can be readily understood as it is well-known that for this task, some non-linearity is required in order to obtain good RC performance. Note that the average neuron power $\langle P_x \rangle$ can be used to estimate the average non-linear phase ϕ_{Kerr} the signals will acquire during the sample duration t_S, as $\phi_{Kerr} = \gamma_{Kerr} \langle P_x \rangle L t_S / t_M$. We observe that without the presence of phase noise in the cavity, the boost to the RC performance due to the Kerr effect starts at very small values of the estimated non-linear phase, and breaks down when $\phi_{Kerr} \gtrsim 1$. Switching to **Figure 4b** we have now introduced the square non-linearity by using a PD in the readout layer. Focusing on the results obtained

with a linear reservoir, we see that the PD's non-linearity alone decreases the NMSE down from 20 to ∼5% ($\gamma = 0$). Although the PD's non-linearity clearly boosts the RC performance on this task, its effect is rather restricted. The PD only generates squared terms, and linear terms if a bias is present, see section 2.5, depending on the MZM's operating point. Furthermore, this non-linearity does not affect the neural responses nor the operation of the reservoir itself, as it only applies to the readout layer. It can thus be understood that the introduction of the Kerr non-linearity inside the reservoir warrants an additional significant drop in NMSE, to below 1% ($\gamma = \gamma_{Kerr}$). In **Figure 4c**, the output layer is linear again, but now we have introduced the MZM in the input layer. The closed markers correspond with simulations where the MZM operates around the zero intensity operating point or the point of minimal transmission ($V_{bias} = V_\pi$). In terms of the optical field modulation, this is the most linear regime. It is thus no surprise that the performance of both linear and non-linear reservoirs mimics that **Figure 4a** where no non-linearity was present in the input layer. The only difference is that the error of the linear reservoir drops from 20% to about 13%($\gamma = 0$, $V_{bias} = V_\pi$) because of the small residual non-linearity at this operating point of the MZM. The round markers correspond with simulations where the MZM operates around the linear intensity operating point ($V_{bias} = V_\pi/2$). In terms of the optical field modulation, the non-linearity in the mapping of input samples to the optical field injected into the reservoir is more non-linear at this operating point. This is why even the linear reservoir manages to achieve errors below 4% ($\gamma = 0$, $V_{bias} = V_\pi/2$). Again we see that the introduction of the non-linear Kerr effect allows the NMSE to drop even further, to below 1% ($\gamma = \gamma_{Kerr}$). In fact, this scenario is similar to the scenario with linear input mapping and non-linear output mapping, **Figure 4b**. Finally, in **Figure 4d**, non-linearities are present in both the input mapping and readout layer. With the MZM operating around the zero intensity operating point, there is only a weak non-linearity in the input mapping and thus, as expected, both linear and non-linear reservoirs show trends which are very similar to the scenario where the input mapping is linear, **Figure 4c**. With the MZM operating around the linear intensity operating point ($V_{bias} = V_\pi/2$) however, we observe a scenario in which the RC does not seem to benefit from the presence of the Kerr non-linear effect. It seems that with significant non-linearities present in both input and output layers of the RC the distributed non-linear effect inside the reservoir cannot further decrease the NMSE below values attained by the linear reservoir, which is below 1% ($V_{bias} = V_\pi/2$). In all other cases, **Figures 4a–c**, we find that the distributed non-linearity inside the reservoir significantly boosts RC performance, and we find that its presence is critical when no other non-linearities are available.

3.2. Experimental Verification: Linear and Non-linear Memory Capacity

In this section we compare experimental results with detailed numerical simulations. For the experimental verification of our work, we are currently limited to operate with 20 neurons, as explained in section 2.1. Therefore, we have chosen not to perform the reservoir computing experiment on the Santa Fe task. With this few neurons, tasks like the Santa Fe task become hard for the reservoir. Instead we turn to a more academic task which allows us to quantify the reservoir's memory and non-linear computational capacity in a more complete and task-independent way. We experimentally measure the linear and non-linear memory capacities considered in section 2.5. Even with this few neurons the evaluation of the memory capacities can yield meaningful results while taking up comparatively little processing time.

For these experiments, the input layer to our fiber-ring reservoir contains a balanced MZM tuned to operate in a linear regime as outlined in section 2.4. The output layer employs a PD to measure the neural responses. That is, we use the setups of **Figures 2B,D** but with the MZM operated as in Equation (2.4). Following reference [20], we have driven the reservoir with a series of independent and identically distributed random samples and trained the RC to reproduce different linear and non-linear polynomial functions of past input samples. The capacity of the reservoir to reconstruct these functions was then evaluated and results were grouped according to the function's polynomial degree. To retain oversight on the results, we will only show the total capacity per degree, by summing all capacities corresponding with functions of the same total polynomial degree. In **Figure 5** we show the total memory capacity per degree, encoded in the height of vertically stacked and color-coded bars. The stacking allows to visualize the contributions of individual degrees to the total overall memory capacity (summed over all degrees). Capacities of degree higher than 4 are not considered, as they were found not to contribute significantly to the total memory capacity of the system. For results labeled *bias off* the MZM operates at the zero-intensity point ($V_{bias} = V_\pi$), and moving toward the *bias on* label, we tuned the MZM's bias voltage ($V_{bias} = V_\pi - \delta_V$, with $\delta_V \ll V_\pi$). This introduces a small bias component to the optical field injected into the reservoir, without compromising the linear operation of the MZM. The experiment was also repeated for different values of the sample duration t_S with respect to the input mask periodicity t_M (approximately equal to the cavity roundtrip t_R). We expect the sample duration to play a very important role, since it determines how much time a piece of information spends inside the cavity, and thus how much non-linear phase can be acquired. The ratio t_S/t_M is gradually increased from $t_S = 2t_M$ in (first row) **Figures 5A–C**, to $t_S = 6t_M$ in (middle row) **Figures 5D–F**, and finally to $t_S = 10t_M$ in (bottom row) **Figures 5G–I**. The experimental results in (left column) **Figures 5A–G** are compared with numerical results on a linear reservoir ($\gamma = 0$) in (middle column) **Figures 5B–H**, and a non-linear reservoir ($\gamma = \gamma_{Kerr}$) in (right column) **Figures 5C–I**.

Firstly, in **Figure 5A** we observe that without bias to the optical input field ($V_{bias} = V_\pi$) the total memory capacity originates almost completely from the polynomial functions of degree 2 which means (given the presence of the PD in the readout layer) that the optical system is almost completely linear. Then, as an optical field bias is introduced we find

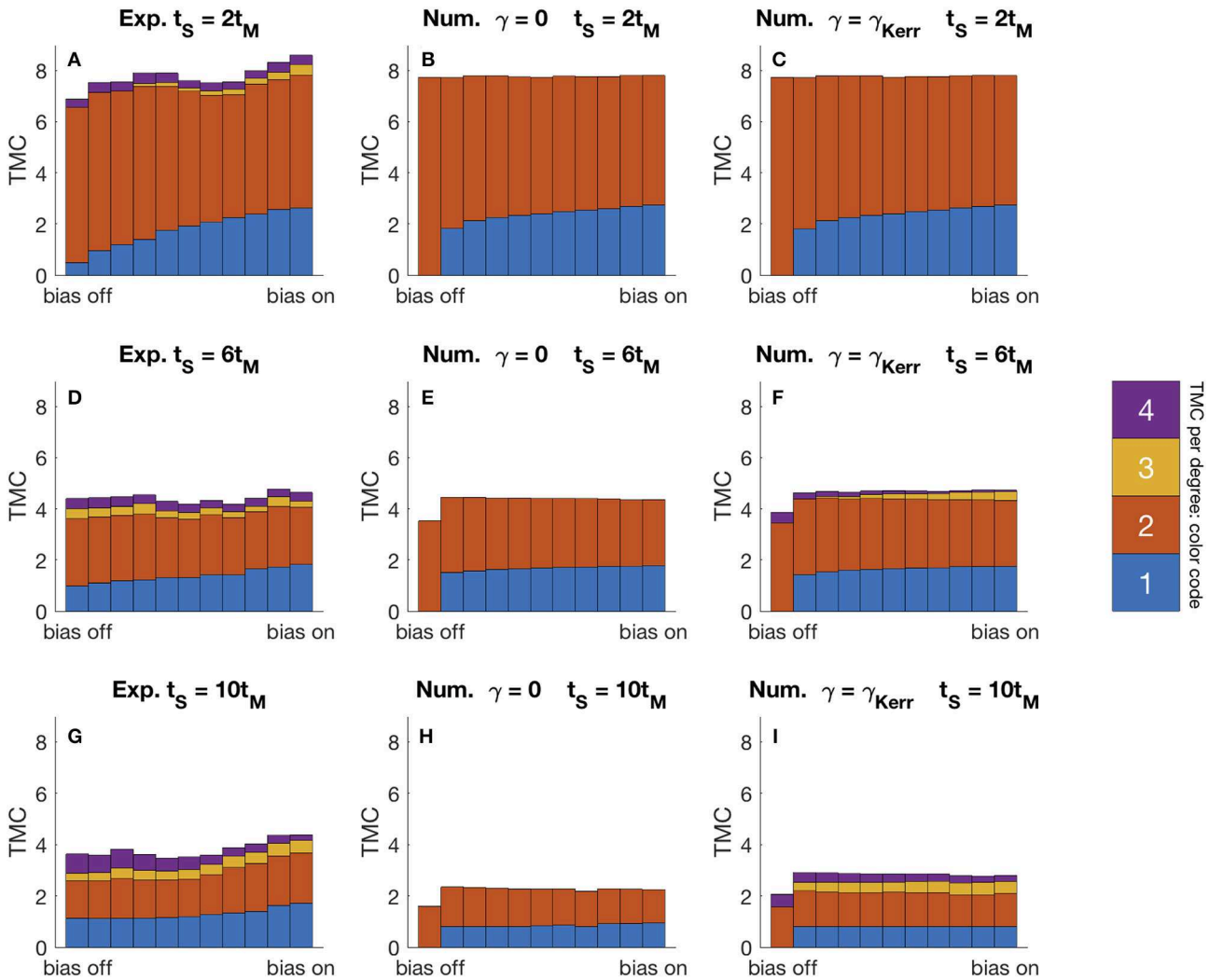


FIGURE 5 | Comparison between experimental results **(A,D,G)** and numerical models with linear ($\gamma = 0$) **(B,E,H)** and non-linear ($\gamma = \gamma_{Kerr}$) reservoirs **(C,F,I)**. The stacked vertical bars are color-coded to represent the total memory capacities (TMC) of degree 1 (blue), 2 (red), 3 (orange), and 4 (purple). As such, the total height represents the total overall memory capacity. A control variable to the MZM δ_V, is varied to include a small bias component to the injected optical field, where *bias off* corresponds with $\delta_V = 0$ and *bias on* corresponds with a small non-zero value $0 < \delta_V \ll V_\pi$. The sample duration t_S is varied from 2 times **(A–C)**, to 6 times **(D–F)** and finally to 10 times **(G–I)** the input mask period t_M ($\approx$ cavity roundtrip time t_R).

that the total linear memory capacity of the system is now shared between degrees 1 and 2. As expected on account of quadratic non-linearity due to the PD, Equation (20), the contribution of (odd) degree 1 grows with the increasing bias. Beyond these capacities of degrees 1 and 2, we also observe a small contribution of capacities of degrees 3 and 4. We ascribe these contributions to the imperfect tuning of the MZM and thus a small residual non-linearity in the input mapping. Note that the simulations take into account the quasi-linear input mapping of the MZM, but seemingly underestimate the residual non-linearities to be insignificant. The imperfection of the MZM tuning also leads to a small residual bias component to the optical injected field, resulting in a small non-zero capacity of degree 1. Numerical simulations of linear ($\gamma = 0$) and non-linear ($\gamma = \gamma_{Kerr}$) reservoirs in **Figures 5B,C**, respectively, show the same growth in the memory capacity of degree 1 at the expense of the memory capacity of degree 2 when the bias is changed. Note that both simulations seem to overestimate the minimal bias required to obtain a significant memory capacity of degree 1. At this sample duration ($t_S = 2t_M$) neither simulations indicate any significant contributions of capacities with degrees beyond 2.

When increasing the sample duration ($t_S = 6t_M$ and $t_S = 10t_M$), the experimental results in **Figures 5D,G** show a steady increase in the contributions of capacities with degrees 3 and 4. This increase is attributed to the non-linear Kerr effect, due to the larger accumulation of non-linear phase during the time each sample is presented to the reservoir. At the same time we see a decrease in the capacities of degrees 1 and 2. As

explained before, due to the PD these capacities capture the reservoir's capacity to linearly retain past samples. This trade-off between linear memory capacity (here degrees 1 and 2) and non-linear computational capacity (here degrees 3 and 4) is well-documented [20]. Because we use the sample duration ($t_S = kt_M \approx kt_R$) to control the cumulative non-linear effect inside the reservoir, we inevitably increase the mismatch between the inherent timescale of the input data (i.e., the sample duration t_S) and the inherent timescale of the reservoir (i.e., the cavity roundtrip t_R). and alter the reservoirs internal topology. When each sample is presented longer, past samples have spent more time inside the lossy cavity by the time they are accessed through the reservoirs noisy readout. Thus, on the longer timescales (t_S) at which information is now processed, it is harder for the reservoir (operating at timescale t_R) to retain past information. These aspects explain why the overall total memory capacity (summed over all degrees) decreases with increased sample duration t_S. The numerical results on both the linear reservoir ($\gamma = 0$) in **Figures 5E,H** and the non-linear reservoir ($\gamma = \gamma_{Kerr}$) in **Figures 5F,I** correctly predict a drop in the total linear memory capacities (degrees 1 and 2). Due to the memory capacity cutoff explained in section 2.5, small capacities are harder to quantify accurately and systematic underestimation can occur. This explains why the small total memory capacities obtained experimentally are larger than the small total memory capacity obtained numerically. The correspondence for large total memory capacities is better as they are largely unaffected by the cutoff. But besides the drop in linear memory capacities, only the non-linear reservoir model can explain the steady increase in non-linear memory capacities (degrees 3 and 4) with longer sample durations. With increasing sample duration t_S the simulated non-linear reservoir shows the contribution of the total non-linear memory capacity (degrees 3 and 4) to the total memory capacity (all degrees) growing from 0 to 25.4%, and in the experiment this contribution starts at 6.4% and grows up to 23.6%. This sizable increase in non-linear computation capacity can be of considerable significance to the reservoir's performance on other tasks, as shown earlier. When comparing the experimental results with the non-linear reservoir model for all given sample durations t_S, the main difference is that the capacities of degree 3 seem to appear sooner (i.e., for smaller sample duration) in the experiment. This can be explained by the residual bias component to the optical injected field. Such a bias makes it easier to produce polynomial functions of odd degrees, thus explaining their earlier onset. This can be explained by the quadratic nature of the Kerr non-linearity, as the reasoning previously applied to the quadratic non-linearity of the PD in Equation (20) can be generalized to memory capacities of higher degree.

4. DISCUSSION

We have identified and investigated the role of non-linear transformation of information inside a photonic computing system based on a passive coherent fiber-ring reservoir. Non-linearities can occur at different places inside a reservoir computer: the input layer, the bulk and the readout layer. State-of-the-art opto-electronic RC systems often include one or several components which inevitably introduce non-linearities to the computing system. On the reservoir's input side, we have compared a linear input regime with the usage of a MZM, which has a non-linear transfer function, to convert electronic data to an optical signal. On the reservoir's output side, we have compared a linear output regime with the usage of a PD which measures optical power levels, that scale quadratically with the optical field strength of the neural responses. We numerically evaluated such systems using a benchmark test and found that non-linear input and/or output components are needed to obtain good RC performance when the optical reservoir itself (i.e., the core of the RC system) is a strictly linear system.

Internal to the reservoir, we investigated the effect of the optical Kerr non-linear effect on RC performance. Our numerical benchmark test showed a large band of optical powers where the presence of this distributed non-linear effect, caused by the waveguiding material of the reservoir, significantly decreased the RC's error figure. Our numerical and experimental measurements of the linear and non-linear memory capacity of this RC system showed that the accumulation of non-linear phase due to the distributed non-linear Kerr effect strongly improves the system's non-linear computational capacity. We can thus conclude that for photonic reservoir computers with non-linear input and/or output components, the presence of a distributed non-linear effect inside the optical reservoir improves the RC performance. Furthermore, the distributed non-linearity is essential for good performance in the regime where non-linearities are absent from both the input and output layer. This may be the case in an all-optical reservoir computer (i.e., with optical input and output layers). We have shown that the effect of the distributed non-linearity is strong enough to compensate for the lack of non-linear transformation of information elsewhere in the system, and that it allows to build a computationally strong photonic computing system.

Finally, we expect a design approach including distributed non-linear effects to improve the scalability of these types of computational devices. In general, when harder tasks are considered, larger reservoirs are required. One way to increase the size of a delay-based reservoir is to implement a longer delay-line. This increase in length of the signal propagation path naturally increases the effect of distributed non-linearities as considered in this work. Similarly, increasing the size of a network-based reservoir will also lead to more and/or longer signal paths, resulting in the increased accumulation of non-linear effects, although waveguides with stronger non-linear effects may have to be considered to compensate for the shorter connection lengths in on-chip implementations. We believe that the natural increase in the strength of non-linear effects, following the increase in size of the reservoir, may diminish the need to place discrete non-linear components inside large networks used for strongly non-linear tasks. As such, both the complexity and cost of such systems would be reduced. Since the waveguiding material itself is used to induce non-linear effects, the waveguide properties (such as material and geometry) determines the optical field confinement and thus regulate the strength of non-linear interactions. Consequently it

may be possible to create reservoirs where deliberate variations in the waveguide properties are used to tune the strength of the distributed non-linear effect in different regions of the system. This would allow for a trade off between the system's linear memory capacity and its non-linear computational capacity, such that a large number of past input samples can be retained (in some parts of the system) and then non-linearly processed to solve difficult tasks (in other parts of the system). These considerations indicate why distributed non-linear effects may play a major role in future implementations of powerful photonic reservoir computers.

DATA AVAILABILITY STATEMENT

The data used in this study for the Sante Fe prediction task [21] is one of the data sets from the "Time Series Prediction Competition" sponsored by the Santa Fe Institute, initiated by Neil Gershenfeld and Andreas Weigend in the early 90s, no licenses/restrictions apply. No further datasets were used or generated.

AUTHOR CONTRIBUTIONS

The idea was first conceived by GVa and finalized together with GVe and SM. JP was responsible for the physical modeling, the numerical calculations, the experimental verification, and wrote most of the manuscript. All coauthors contributed to the discussion of the results and writing of the manuscript.

FUNDING

We acknowledge financial support from the Research Foundation Flanders (FWO) under grants 11C9818N, G028618N, and G029519N, the Fonds de la Recherche Scientifique (FRS-FNRS), the Hercules Foundation and the Research Council of the VUB.

REFERENCES

1. Maass W, Natschläger T, Markram H. Real-time computing without stable states: a new framework for neural computation based on perturbations. *Neural Comput.* (2002) **14**:2531–60. doi: 10.1162/089976602760407955
2. Jaeger H, Haas H. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. *Science.* (2004) **304**:78–80. doi: 10.1126/science.1091277
3. Verstraeten D, Schrauwen B, dHaene M, Stroobandt D. An experimental unification of reservoir computing methods. *Neural Netw.* (2007) **20**:391–403. doi: 10.1016/j.neunet.2007.04.003
4. Appeltant L, Soriano MC, Van der Sande G, Danckaert J, Massar S, Dambre J, et al. Information processing using a single dynamical node as complex system. *Nat Commun.* (2011) **2**:468. doi: 10.1038/ncomms1476
5. Paquot Y, Duport F, Smerieri A, Dambre J, Schrauwen B, Haelterman M, et al. Optoelectronic reservoir computing. *Sci Rep.* (2012) **2**:287. doi: 10.1038/srep00287
6. Larger L, Soriano MC, Brunner D, Appeltant L, Gutiérrez JM, Pesquera L, et al. Photonic information processing beyond turing: an optoelectronic implementation of reservoir computing. *Opt Express.* (2012) **20**:3241–9. doi: 10.1364/OE.20.003241
7. Duport F, Smerieri A, Akrout A, Haelterman M, Massar S. Fully analogue photonic reservoir computer. *Sci Rep.* (2016) **6**:22381. doi: 10.1038/srep22381
8. Larger L, Baylón-Fuentes A, Martinenghi R, Udaltsov VS, Chembo YK, Jacquot M. High-speed photonic reservoir computing using a time-delay-based architecture: million words per second classification. *Phys Rev X.* (2017) **7**:011015. doi: 10.1103/PhysRevX.7.011015
9. Bueno J, Maktoobi S, Froehly L, Fischer I, Jacquot M, Larger L, et al. Reinforcement learning in a large-scale photonic recurrent neural network. *Optica.* (2018) **5**:756–60. doi: 10.1364/OPTICA.5.000756
10. Vandoorne K, Dambre J, Verstraeten D, Schrauwen B, Bienstman P. Parallel reservoir computing using optical amplifiers. *IEEE Trans Neural Netw.* (2011) **22**:1469–81. doi: 10.1109/TNN.2011.2161771
11. Vandoorne K, Mechet P, Van Vaerenbergh T, Fiers M, Morthier G, Verstraeten D, et al. Experimental demonstration of reservoir computing on a silicon photonics chip. *Nat Commun.* (2014) **5**:3541. doi: 10.1038/ncomms4541
12. Katumba A, Heyvaert J, Schneider B, Uvin S, Dambre J, Bienstman P. Low-loss photonic reservoir computing with multimode photonic integrated circuits. *Sci Rep.* (2018) **8**:2653. doi: 10.1038/s41598-018-21011-x
13. Harkhoe K, Van der Sande G. Dual-mode semiconductor lasers in reservoir computing. In: *Neuro-Inspired Photonic Computing.* Vol. 10689. Straatsburg: International Society for Optics and Photonics (2018). p. 106890B.
14. Duport F, Schneider B, Smerieri A, Haelterman M, Massar S. All-optical reservoir computing. *Opt Express.* (2012) **20**:22783–95. doi: 10.1364/OE.20.022783
15. Brunner D, Soriano MC, Mirasso CR, Fischer I. Parallel photonic information processing at gigabyte per second data rates using transient states. *Nat Commun.* (2013) **4**:1364. doi: 10.1038/ncomms2368
16. Vinckier Q, Duport F, Smerieri A, Vandoorne K, Bienstman P, Haelterman M, et al. High-performance photonic reservoir computer based on a coherently driven passive cavity. *Optica.* (2015) **2**:438–46. doi: 10.1364/OPTICA.2.000438
17. Van der Sande G, Brunner D, Soriano MC. Advances in photonic reservoir computing. *Nanophotonics.* (2017) **6**:561–76. doi: 10.1515/nanoph-2016-0132
18. Bienstman P, Dambre J, Katumba A, Freiberger M, Laporte F, Lugnan A. Photonic reservoir computing: a brain-inspired approach for information processing. In: *Optical Fiber Communication Conference*. San Diego, CA: Optical Society of America (2018). p. M4F–4.
19. Jaeger H. Short term memory in echo state networks. GMD-Report 152. In: *GMD-German National Research Institute for Computer Science*. Citeseer (2002). Available online at: http://www.faculty.jacobs-university.de/hjaeger/pubs/STMEchoStatesTechRep.pdf
20. Dambre J, Verstraeten D, Schrauwen B, Massar S. Information processing capacity of dynamical systems. *Sci Rep.* (2012) **2**:514. doi: 10.1038/srep00514
21. Weigend AS, Gershenfeld NA. Results of the time series prediction competition at the Santa Fe Institute. In: *IEEE International Conference on Neural Networks*. San Fransisco, CA: IEEE (1993). p. 1786–93.

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

The reviewer VB declared a shared affiliation, with no collaboration, with the authors JP and SM to the handling editor at time of review.

frontiers in Physics

REVIEW
published: 10 October 2019
doi: 10.3389/fphy.2019.00153

The Application of Machine Learning Techniques to Improve El Niño Prediction Skill

Henk A. Dijkstra[1,2]*, *Paul Petersik*[1], *Emilio Hernández-García*[3] and *Cristóbal López*[3]

[1] Department of Physics, Institute for Marine and Atmospheric Research Utrecht, Utrecht University, Utrecht, Netherlands, [2] Department of Physics, Center for Complex Systems Studies, Utrecht University, Utrecht, Netherlands, [3] IFISC (Spanish National Research Council - University of the Balearic Islands), Instituto de Física Interdisciplinar y Sistemas Complejos, Palma de Mallorca, Spain

We review prediction efforts of El Niño events in the tropical Pacific with particular focus on using modern machine learning (ML) methods based on artificial neural networks. With current classical prediction methods using both statistical and dynamical models, the skill decreases substantially for lead times larger than about 6 months. Initial ML results have shown enhanced skill for lead times larger than 12 months. The search for optimal attributes in these methods is described, in particular those derived from complex network approaches, and a critical outlook on further developments is given.

Keywords: El Niño, prediction, machine learning, neural networks, attributes, climate networks

OPEN ACCESS

Edited by:
Raul Vicente,
Max-Planck-Institut für Hirnforschung, Germany

Reviewed by:
William W. Hsieh,
University of British Columbia, Canada
Diego R. Amancio,
University of São Paulo, Brazil

***Correspondence:**
Henk A. Dijkstra
h.a.dijkstra@uu.nl

Specialty section:
This article was submitted to Interdisciplinary Physics, a section of the journal Frontiers in Physics

Received: *15 June 2019*
Accepted: *23 September 2019*
Published: *10 October 2019*

Citation:
Dijkstra HA, Petersik P, Hernández-García E and López C (2019) The Application of Machine Learning Techniques to Improve El Niño Prediction Skill. Front. Phys. 7:153. doi: 10.3389/fphy.2019.00153

1. INTRODUCTION

Techniques of Artificial Intelligence (AI) and Machine Learning (ML) are very well developed [1], and massively applied in many scientific fields, like in medicine [2], finance [3], and geophysics [4]. Although the application to climate research has been around for a while [5–7], there is much renewed interest recently [8–10]. A main issue in which breakthroughs are expected is the representation of unresolved processes (e.g., clouds, ocean mixing) in numerical weather prediction models and in global climate models. For example, recently a ML-inspired (random-forest) parameterization of convection gave accurate simulations of climate and precipitation extremes in an atmospheric circulation model [11]. ML has also been used to train statistical models which mimic the behavior of climate models [12, 13]. Another area of potential breakthrough is the skill enhancement of forecasts for weather and particular climate phenomena, such as the El Niño-Southern Oscillation (ENSO) in the tropical Pacific.

During an El Niño, the positive phase of ENSO, sea surface temperatures in the eastern Pacific increase with respect to average values and upwelling of colder, deep waters diminishes. The oscillation phase opposite to El Niño is La Niña, with a colder eastern Pacific and increased upwelling. A measure of the state of ENSO is the NINO3.4 index (**Figure 1A**), which is the area-averaged Sea Surface Temperature (SST) anomaly (i.e., deviation with respect to the seasonal cycle) over the region 170–120°W × 5°S–5°N. Averaging over other areas defines other indices such as NINO3. For ENSO predictions, often the Oceanic Niño Index (ONI) is used which refers to the 3-months running mean of the NINO3.4 index.

El Niño events typically peak in boreal winter, with an irregular period between two and seven years, and strength varying irregularly on decadal time scales. The most recent strong El Niño had its maximum in December 2015 (**Figure 1A**). The spatial pattern of ENSO variability is often represented by methods from principal component analysis [14], detecting patterns of maximal variance. The first Empirical Orthogonal Function (EOF) of SST anomalies, obtained from the

Hadley Centre Sea Ice and Sea Surface Temperature (HadISST) dataset [15] over the period 1950–2010, shows a pattern strongly confined to the equatorial region with largest amplitudes in the eastern Pacific (**Figure 1B**).

El Niño events typically cause droughts on the western part of the Pacific and flooding events on the eastern part and hence affect climate worldwide. Estimated damages for the 1997–1998 event were in the order of billions of US$ [16]. The development of skillful forecasts of these events, preferably with a one year lead time, is hence important. These forecasts will enable policy makers to mitigate the negative impacts of the associated weather anomalies. For example, farmers can be advised to use particular types of corn in El Niño years and others during La Niña years (see e.g., http://globalagrisk.com).

Although more detailed regional measures are sometimes desired in a forecast, most focus is on spatially averaged indices such as the NINO3.4 (cf. **Figure 1A**). Forecasting this time series is an initial value problem requiring the specification of initial conditions (of relevant observables) and a model, which can be either statistical or dynamical. With this model, one can predict future values of these observables or of other ones from which meaningful diagnostics, such as the NINO3.4 index, can be obtained. Due to many efforts in the past, detailed observations of relevant oceanic and atmospheric variables are available (since the mid-1980s) through the TAO-TRITON observation array in the tropical Pacific, and satellite data of sea surface height, surface wind stress and sea surface temperature [17]. In addition, reanalysis data (i.e., model simulations which assimilate existing observations) such as ERA-Interim [18] provide a rather detailed characterization of present and past state of the Pacific, essential for successful prediction of the future.

This paper provides an overview of efforts to use ML, mainly Artificial Neural Network (ANN) approaches, to predict El Niño events, and putting them in the context of classical prediction methodologies. In section 2, we describe the state-of-the-art in current prediction practices, the efforts to understand the results, and in particular what determines the skill of these forecasts. Then results of ML-based approaches are described in section 3 and challenges and outlook are described in section 4.

2. EL NIÑO PREDICTION: STATE OF THE ART

There have been many reviews on El Niño predictability (e.g., [19–22]) and a recent one [23], reviewing also most of the Chinese-community studies on this topic. Over the last decade, a multitude of models is used for El Niño prediction and results are available at several websites. Multi-model ensemble results are given at the International Research Institute for Climate and Society (IRI)[1] providing results from both dynamical models (i.e., models based on underlying physical conservation laws) and statistical models (those capturing behavior of past statistics). The NCEP Climate Forecast System CFSv2 [24][2], provides a dynamical single-model ensemble forecast. The forecast systems developed in China, such as the SEMAP2 and the NMEFC/SOA are discussed in detail in Tang et al. [23] so they are further discussed here. It is illustrative to show the results of both the IRI and CFSv2 model systems for the last strong El Niño event, that of 2015–2016, which was discussed in detail by L'Heureux et al. [25]. Forecasts starting in June 2015 are shown in **Figure 2** indicating that these models are able to provide a skillful forecast of NINO3.4. Nevertheless, the dispersion in the predictions of the different models is huge, and even between ensemble members of the same model, highlighting the difficulty of reliable prediction.

The US National Oceanic and Atmospheric Administration (NOAA) will release an El Niño advisory when (i) the 1-month NINO-3.4 index value is at or in excess of 0.5°C, (ii) the atmospheric conditions are consistent with El Niño (i.e., weaker low-level trade winds, enhanced convection over the central or eastern Pacific Ocean), and (iii) at least five overlapping seasonal (3-months average) NINO3.4 SST index values are at or in excess of 0.5°C, supporting the expectation that El Niño will persist. The purpose of the forecasting efforts such as those in **Figure 2** is to predict in advance when those conditions will occur. Both the IRI and CFSv2 predicted already in June 2014 an El Niño event for next winter which turned out to be wrong as there was a dip in NINO3.4 at the end of 2014 (due to easterly winds). However, most models did very well in predicting (from June 2015) the winter 2015-2016 strong event (see **Figure 2**).

The skill of El Niño forecasts is usually measured by the anomaly correlation coefficient (AC) given by:

$$AC = \frac{\overline{m'o'}}{\sigma_m \sigma_o}, \tag{1}$$

where m' indicates NINO3.4 index of the model, o' that of observations and σ_x indicates the standard deviations of the time series x. The overbar indicates averaging of all time series elements. The AC is the Pearson correlation coefficient between prediction and observation. In Barnston et al. [21], the skill of the models over the period 2002–2011 was summarized with help of **Figure 3A** which indicates that skill beyond a 6-months lead time becomes overall lower than 0.5. Some general conclusions from these and many other prediction exercises are that (i) dynamical models do better than statistical models and (ii) models initialized before the Northern-hemispheric spring perform much worse than models initialized after spring. The latter notion is known as the "spring predictability barrier" problem. The concept of *persistence* is another way to look at the predictability barrier. It can be defined in terms of autocorrelation coefficients and in particular their decay with increasing lead times. SST anomalies originating from the spring seasons have the least persistence while those originating from summer seasons (**Figure 3B**) tend to have the greatest persistence [27].

El Niño events are difficult to predict as they have an irregular occurrence, and each time have a different development [17, 22]. The ENSO phenomenon is thought to be an internal mode of the coupled equatorial ocean-atmosphere system which can be self-sustained or excited by random noise [28]. The interactions

[1]https://iri.columbia.edu/our-expertise/climate/forecasts/enso/current/?enso_tab=enso-sst_table

[2]https://www.cpc.ncep.noaa.gov/products/CFSv2/CFSv2seasonal.shtml

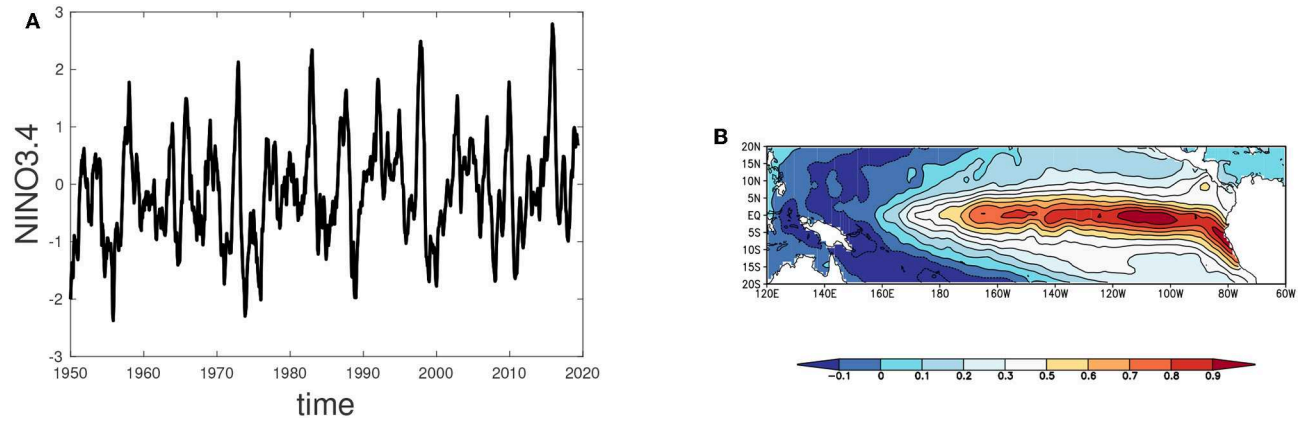


FIGURE 1 | (A) NINO3.4 index (°C) over the period 1950–2019. **(B)** Pattern of the first EOF (arbitrary units) of Sea Surface Temperature determined from the HadISST data set over the period 1950–2010. Data source: Climate Explorer (http://climexp.knmi.nl).

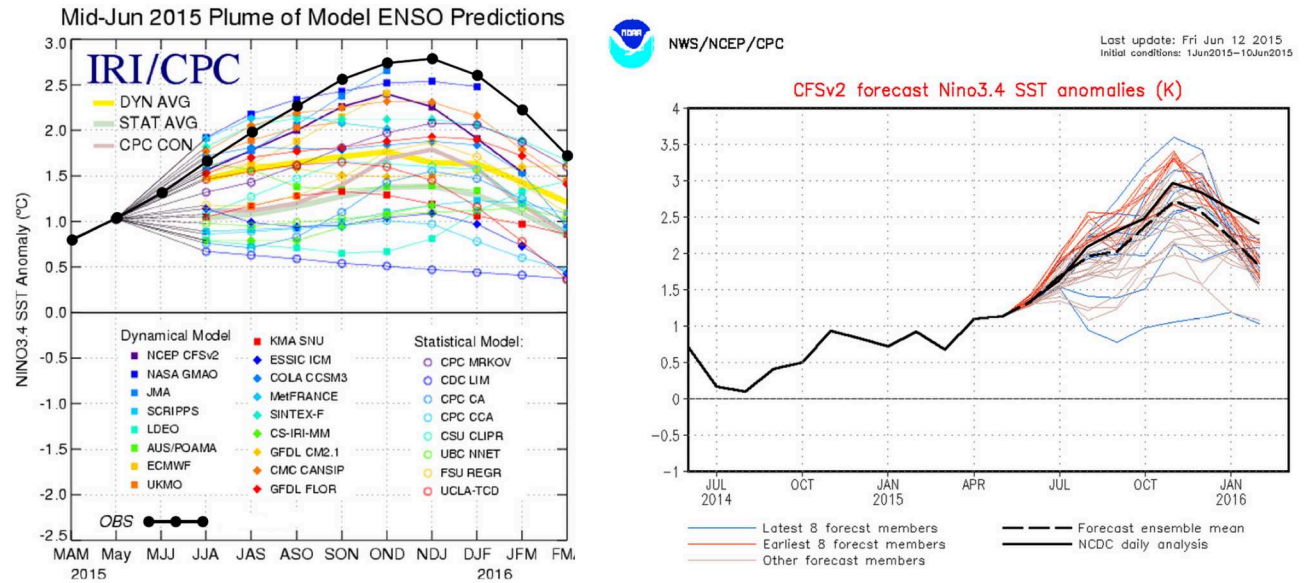


FIGURE 2 | (A) Predictions of NINO3.4 from the IRI multi-model ensemble initialized with data up to mid-June 2015. Full and empty symbols are predictions from dynamical and statistical models, respectively. Continuous lines without symbols display the average of the predictions of all the dynamical models, of all the statistical ones, and of the four models run at the NOAA Climate Prediction Center (CPC). The black line with black symbols is the observed seasonal NINO3.4 index (from ERSST.v5 [26]). **(B)** Predictions from the CFSv2 single-model ensemble (each member of the ensemble is initialized slightly differently at mid-July). The continuous black line is the (later added) observed monthly NINO3.4 index (from ERSST.v5) and the dashed line is the forecast ensemble mean. Both panels are slightly modified (by adding the later observed values) from the ones on the websites listed in the footnotes.

of the internal mode and the external seasonal forcing can lead to chaotic behavior through nonlinear resonances [29, 30]. On the other hand, the dynamical behavior can be strongly influenced by noise, in particular westerly wind bursts [31] which can either be viewed as additive [32] or multiplicative noise [33]. Coupled processes between the atmosphere and ocean are seasonally dependent. During boreal spring the system is most susceptible to perturbations [34] leading to a spring predictability barrier [35]. The growth of perturbations from a certain initial state has been investigated in detail from one of the available intermediate-complexity models, the Zebiak-Cane model (ZC, [36]), using the methodology of optimal modes [37–39]. It was indeed shown that spring is the most sensitive season as perturbations are amplified over a 6-months lead time.

In summary, the low skill after 6 months as seen in **Figure 3A** is believed to be due to both effects of smaller scale processes (noise) and nonlinear effects. Moreover, it is supposed that the period 2002-2011 was particularly difficult to predict because of the frequent occurrence of central Pacific (CP) El Niño types [40] for which the zonal advection feedback plays a key role for the development. In contrast, for an eastern Pacific (EP) El Niño, the thermocline feedback is most important [41]. As can be seen from

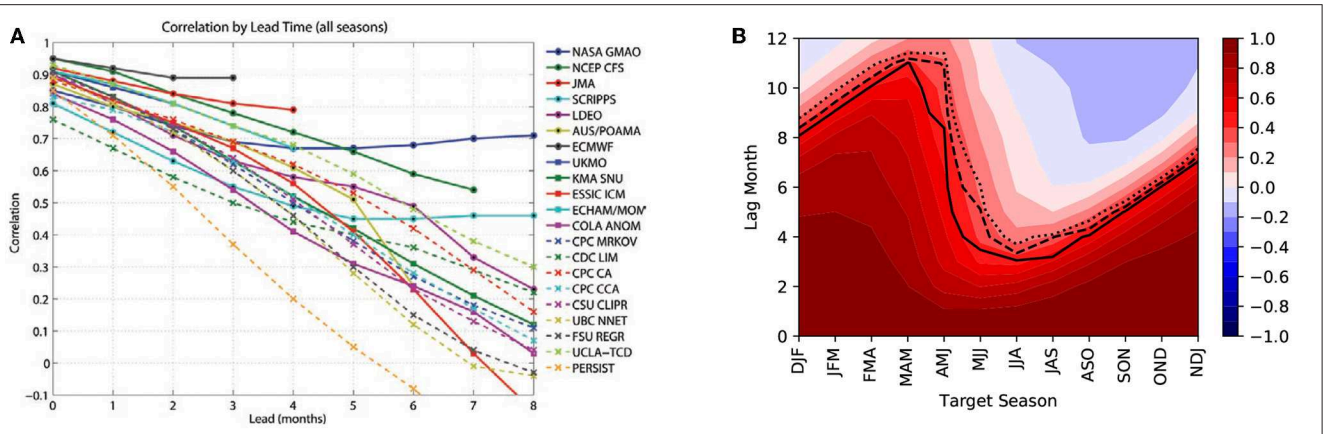


FIGURE 3 | (A) Correlation skill (AC as in 1) of NINO3.4 prediction for different models from the IRI ensemble over the period 2002–2011 [figure from Barnston et al. [21], with permission from the AMS]. Continuous lines are for dynamical models, whereas dashed lines are for statistical ones. The yellow "PERSIST" line assumes simply the persistence of the initial conditions. **(B)** Pearson correlation between the ONI index (source https://www.cpc.ncep.noaa.gov/data/indices/oni.ascii.txt, which is computed as the 3-months running mean NINO3.4 value) at the target season (x-axis) and itself at the specified lag time previous to the target season (y-axis) for the period between 1980 and 2018. The contour lines indicate the 90% (dotted), 95% (dashed), and 99% (solid) two-sided significance levels for a positive autocorrelation.

the NINO3.4 time series (**Figure 1A**) strong events appear about every 15 years (1982, 1996, 2015). There are likely other factors involved in the prediction skill of these strong events [42, 43], which we do not further discuss here.

3. MACHINE LEARNING APPROACHES

ML is being used in a variety of tasks that include regression and classification. ML algorithms can be divided into three main categories [44]: supervised, unsupervised, and reinforcement learning. In supervised learning, a model is build from labeled instances. In a unsupervised model, there are no labeled instances and the goal is to find hidden patterns (e.g., clustering) in the available data. In reinforcement learning, a particular target is pursued and feedbacks from the environment drive the learning process [1]. The usual procedure in supervised learning is as follows: the predictor model [e.g., an artificial neural network (ANN) or genetic programming (GP)] is trained with data from a training set in order to determine a set of optimal parameter values. Then the generalization capabilities of the model are tested on a validation data set. Once the predictor model is validated, a third so-called test data set that was hold out during training and validation can be used to evaluate the prediction skill.

Many types of ML methodologies have been developed. In ANNs, the basic element is the neuron, or perceptron (i.e., logistic or other function units which locally discriminate different inputs). An ANN has a multilayer structure—an input layer, an output layer and a few (or zero) hidden layers, in which each neuron is connected to all neurons in the previous and following layers. The system thus maps some input applied to the input layer to some output or prediction. The weights of the neuron connections are tuned to provide the optimal predictor model. Another ML technique, GP, is a symbolic regression method used to find, by optimization procedures inspired by biological evolutionary processes, the functional form that fits the available data [45]. *Reservoir computing* [46] is another type of ML methodology in which input is injected into a high-dimensional dynamical system called "reservoir." The response of the reservoir is recorded at particular output nodes with associated "output weights," and linear regression is used to optimize these weights so that the recorded response performs the desired prediction.

Although there are a few ML attempts to forecast El Niño events by evolutionary or genetic algorithms [47, 48], and by other methods [49], we will focus here on ML prediction schemes based on the most popular approach, which is the use of feed-forward ANNs. Such ANNs with at least one hidden layer, also called multilayer perceptrons, have the powerful capability to approximate any nonlinear function to an arbitrary accuracy given enough input data and hidden-layer neurons (e.g., [1, 50]).

3.1. Early ML Approaches

There is a large freedom on the implementation of ANN methods and choices have to be made regarding which variables to use as inputs (called in this context the *attributes*, *features*, or *predictors*), the architecture of the ANN and training method. ANNs, as any other supervised learning technique, require to split the available data in at least two parts: a training set, on which parameters of the ANN are optimized, and a test set, on which the skill of the optimized ANN is evaluated. Furthermore, it is good practice to use a third data set, often called validation data set, to tune hyperparameters and to check for overfitting. For ENSO prediction, it is of particular importance to split the data into connected time series. If instead the data set would be split by random sampling, training and test data points would be temporally close to each other. Due to the strong autocorrelations within the ENSO system, the test data set would be strongly

correlated with the training data set and hence could not serve as an independent data set. Because of the shortness of the available time series, the fact that El Niño repeats only about every four years on average, and that only a subset of them are strong, training and validation sets do not contain many significant events and statistical estimation of ANN skills is not very precise.

In using ANN's one can basically focus on two different supervised learning tasks: classification (will there be an El Niño event or not) and regression (predicting an index, e.g., the NINO3.4, with a certain lead time). Early ANN-based El Niño predictions [51] for the regression task used as predictors wind-stress fields and the NINO3.4 time series itself. More explicitly, the time series of the seven leading principal components of the wind-stress field (i.e., the amplitudes of their seven leading EOFs) in a large region of the tropical Pacific were averaged seasonally in each of the four seasons previous to the start of prediction. These numbers, together with the last value of the NINO3.4 time-series make a total of $4 \times 7 + 1 = 29$ inputs to be fed into the ANN. Tangang et al. [52] noticed that using sea level pressure (SLP) fields gave better results at long lead times than using wind-stress fields. Also, averaging forecasts from an ensemble of ANNs with different random weights assigned to the neurons at the start of the learning phase improved results with respect to using the results of a single ANN. Maas et al. [53] further analyzed this fact and suggested using it to estimate prediction reliability. Tangang et al. [54] simplified the ANN architecture by using extended EOFs (EEOFs), which project the observed fields (wind stress or SLP) onto spatio-temporal patterns, instead of on spatial ones (using EOFs). In this way, input from the year previous to the forecast start was compressed to $7 + 1 = 8$ variables, instead of the previous 29. In these earlier studies, all of which used a single hidden layer in the ANN, high forecast skills (values of the correlation *AC* above 0.6 even at lead times above one year) were reported. However, there were large differences in performance depending, for example, on the season of the year or the particular year or decade being predicted.

Later implementations of these early ANN methods indicated that the skill was relatively low. For example, the curve labeled UBC-NNET in **Figure 3A** (coming from the ANN model by the University of British Columbia group, based in Tangang et al. [54]) has the second lowest skill at 6 months lead time, improving only the forecast made by the simple "persistence" assumption. This can partially be attributed to the fact that the model architecture of the UBC-NNET changed in May 2004 from predicting the ONI to predicting the amplitudes of the leading EOFs. There are also differences in the climatology used by UBC-NNET and the one used for the tests in Barnston et al. [21]. Moreover, only during December 2004 and November 2005 the UBC-NNET included subsurface temperature data, while the thermal state of the subsurface can contain important information about the future state of the ENSO [55]. For the remaining period, the model lacked this subsurface information. Ideas to improve ANN performance included optimization regularization [56] and linear corrections that help to quantify prediction errors [57]. Another one to focus on forecasting individual principal components of the SST field and combining them to obtain climatic indices such as NINO3.4 [58], instead of trying to predict directly the climatic index. In general, for a one year lead time, no *AC* values higher than 0.5–0.6 were obtained with these methods, although larger *AC* values have been reported for specific seasons or years.

An exception is the work of Baawain et al. [59] in which very high correlations (above 0.8 for lead times between 1 and 12 months) were reported for prediction of the NINO3 index using as inputs the two surface-wind components and the SAT at four selected locations in the Pacific (thus, 12 inputs). The high forecast skill may arise from the careful and systematic determination of the ANN architecture (again a single hidden layer but with up to 16 neurons, and different activation functions), or perhaps from the choices of training and validation data sets. Some of the practices in Baawain et al. [59], however, are rather questionable and can lead to substantial overfitting for the ENSO prediction. First, they perform the hyperparameter optimization on their test data set. A better practice is to tune hyperparameters on an additional validation data set and hold out the test data set completely during the training and hyperparameter optimization. Second, they do not precisely report how the data is split into the training and test data set. If they split the data by random splitting, the model is likely overfitted due to the problem mentioned earlier. The very small difference between the prediction skill on the training ($r = 0.91$) and the test ($r = 0.90$) data set indicates that they might split their data set by random splitting. A better practice would be to split the data into two connected time series. In addition to pure ANN prediction, also hybrid approaches that use a dynamical ocean model driven by wind stresses provided by an ANN fed by the ocean state have been applied [60, 61]. Skill in predicting El Niño is similar to purely dynamical models, but at a smaller computational cost.

The key for a successful application of ANNs to ENSO prediction is to determine the correct attributes to include in the training of the model. The attributes used in Tangang et al. [54], based on EEOFs of SLP and SST, may be not optimal considering where the memory of the coupled ocean-atmosphere system originates from, i.e., from the subsurface ocean.

3.2. Attributes: Role of Network Science

Although network science had been applied to many other branches of science, it was not applied to climate science until one realized that easy mappings between continuous observables (e.g., temperature) and graphs could be made [62–64]. One can consider these observables to be on a grid (observation locations or of model grid points), which then are the "nodes" of the graph. A measure of correlation between the time series of an observable at two locations, such as the Pearson correlation or mutual information, can then be used to define a "connection" or "link," and eventually to assign a "strength" or "weight" to that connection [65].

Ludescher et al. [66] used the link strength concept for El Niño prediction. They determined the average link strength S of the climate network constructed from a Surface Air Temperature (SAT) data set. They suggested that when S crosses a threshold Θ while monotonically increasing, an El Niño will develop about one year later. The rationale behind this is that, during El Niño,

correlations of climatic variables at many locations with variables in the tropical Pacific are very high, so that an increase of these correlations, conveniently revealed by the connectivity (or "cooperativity") of the climate network, is an indicator of an approach to an El Niño state. A training set over the period 1950–1980 was used to determine the threshold Θ. The result for the test period 1980–2011 showed a remarkable skill of this predictor [66]. By using this method, also a successful prediction of the onset of the weak 2014 El Niño was made [67].

The time-varying characteristics of climate networks (where correlations defining link strength between nodes are calculated on successive time windows) has also been used in a different way. The increase of connectivity of the climate network, occurring when approaching an El Niño event, may lead to a *percolation transition* [68] in which initially disconnected parts of the network become connected into a single component.

The study by Rodríguez-Méndez et al. [69] introduced percolation-based early warnings in climate networks for an upcoming El Niño/La Niña event. Here, the climate networks are generated with a relatively high threshold for the cross-correlation between two nodes to be considered as connected. Hence, one finds a lot of isolated nodes in these networks.

However, even long before an El Niño event is approached these isolated nodes become connected to other ones building clusters of size two since correlations between nodes increase. If the correlation building continues, more small clusters of size two emerge and the proportion of nodes in clusters of size two, indicated by c_2, increases. Approaching further the transition, small clusters can connect to more nodes and form even bigger clusters counter-balancing the increased probabilities for smaller clusters. Hence, in a typical percolation transition, the first sign of the transition is indicated by a peak of c_2. This peak is followed by peaks in the proportion of nodes in clusters of increasing size, the closer the system is to the percolation point. At the percolation point spatial correlations in the system become so strong that a giant component in the network emerges and incorporates nearly all nodes of the system [69, 70]. If the system moves again away from the transition point, peaks in the proportion of nodes in clusters of different sizes appear in reversed order, starting with peaks coming from larger clusters followed by peaks from smaller clusters.

3.3. Recent ML-Based Predictions

Networks are often associated with machine learning techniques, either to generate attributes or to create the learning method itself [71]. The advantage of using the network approach in climate research is that, during network construction, the temporal information is often included to determine the properties of climate networks. In this way, the machine learning techniques will, by default, take the temporal information into account in making predictions of the future states of the system.

A first effort to combine complex network metrics with ANN's for the prediction of the NINO3.4 index was made in Feng et al. [72]. They considered the classification problem (determining if El Niño will occur) with an ANN (two hidden layers with three neurons each) in which attributes were only the climate-network-based quantities from Gozolchiani et al. [64]. The period May 1949 to June 2001 was used as a training set, and the period June 2001 to March 2014 as the test set. The prediction lead time was set to 12 months. Classification results on the test set are shown in **Figure 4A**. Here a 1 indicates the occurrence of an El Niño event (in a 10-days window) and 0 indicates no event. When a filter is applied which eliminates the isolated and transient events and joins the adjacent events, the result is shown in **Figure 4B**. This forecasting scheme can hence give skillful predictions 12 months ahead for El Niño events.

The regression problem, i.e., forecasting the values of time series such as NINO3.4, was addressed by Nooteboom et al. [73] who combined the use of network quantities with a thorough search for attributes based on the physical mechanism behind ENSO. A two-step methodology was used which resulted in a hybrid model for ENSO prediction. In a first step, a classical Autoregressive Integrated Moving Average (ARIMA) linear statistical method [74] is optimized to perform a linear forecast using past NINO3.4 values. Specifically, ARIMA(12,1,0) and ARIMA(12,1,1) were implemented, which means that the NINO3.4 values in the 12 months previous to the start of the prediction were used. The linear prediction was far from perfect, and then an ANN was trained from single-time attributes to forecast the residuals between the linear prediction and the true NINO3.4 values. The sum of the linear forecast and the nonlinear ANN prediction completes the final hybrid model forecast. In Hibon and Evgeniou [75], it is shown that, compared to a single prediction method, this hybrid methodology is more stable and reduces the risk of a bad prediction. This is probably due to the fact that long memory is taken into account, but not in the ANN part, which remains then relatively simple with respect to inputs and can then be more efficiently trained.

To motivate the choice of the attributes in the ANN, Nooteboom et al. [73] used the ZC model [36]. In this model, the physical mechanisms of ENSO are clearly represented and it can be used for extensive testing of different attributes, specially network-based ones which contain correlations and spatial information. Several interesting network variables, such as the cross clustering and an eigenvalue quantifying the coupling between wind and SST networks, were determined from an analysis of the ZC model. More importantly, it revealed the importance of the dynamics of the thermocline, which can be quantified in properties of the thermocline-depth network or the related sea surface height (SSH) network. Also the zonal skewness in the degree field of the thermocline network and two variables related to a percolation-like transition [69, 70], namely the temporal increment in the size of the largest connected cluster and the fraction of nodes in clusters of size two c_2 (see previous subsection) in the SSH network, had good prediction properties. These variables taken from the SSH network are related to the warm water volume (WWV, the integrated volume above the 20°C isotherm between 5°N–5°S and 120-280°E), which was also tested as input in the ANN forecast, and contain information on the physics of the recharge/discharge mechanism of ENSO [76]. It turns out that c_2 performs better than WWV when used in long-lead-time predictions.

Furthermore, apart from these "recharge/discharge" related quantities, a sinusoidal seasonal cycle (SC), introducing

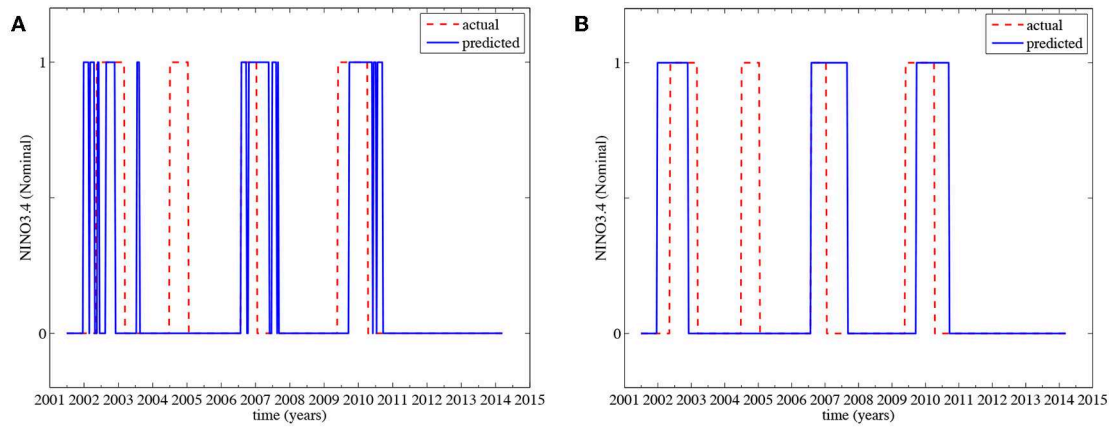


FIGURE 4 | Prediction results for occurrence (from June 2001 to March 2014) of El Niño in time windows of 10 days. Dashed line gives actual observations and the solid one is the prediction. **(A)** Raw prediction. **(B)** With filtering of isolated and transient events. Reproduced from Feng et al. [72] under open access license.

information needed for the phase locking of ENSO, and the second principal component (PC_2) of a modified zonal component of the wind stress, which carries information on westerly wind bursts, were included as predictors. A single sinusoid is not a complete representation of the annual solar forcing, but it gives to the algorithm the phase information necessary to lock El Niño events to the annual cycle. The hybrid model improves on the CFSv2 ensemble at short lead times (up to 6 months) and it had also a better prediction result than all members of the CFSv2 ensemble in the case study of January 2010 [73]. From now on, the Normalized Root Mean Squared Error (NRMSE) is used to indicate the skill of prediction within the test set:

$$NRMSE(y^A, y^B) = \frac{1}{\max\left(y^A, y^B\right) - \min\left(y^A, y^B\right)} \sqrt{\frac{\sum_{t_1^{test} \le t_k \le t_n^{test}} \left(y_k^A - y_k^B\right)^2}{n}}. \quad (2)$$

Here y_k^A, y_k^B are respectively the NINO3.4 index and its prediction at time t_k in the test set. n is the number of points in the test set. A low NRMSE indicates the prediction skill is better.

For short lead times, the hybrid model was used with the WWV, PC_2, SC and NINO3.4 itself as attributes. A temporal shift can be seen in the CFSv2 ensemble NINO3.4 results, both for the 3- and 6-months lead-time prediction (**Figure 5**). The hybrid model predictions used ARIMA(12,1,0) for the linear part, and the eighty-four possible ANN structures with three hidden layers with up to four neurons each were tested. **Figure 5** shows the results from the structures giving the lowest NRMSE.

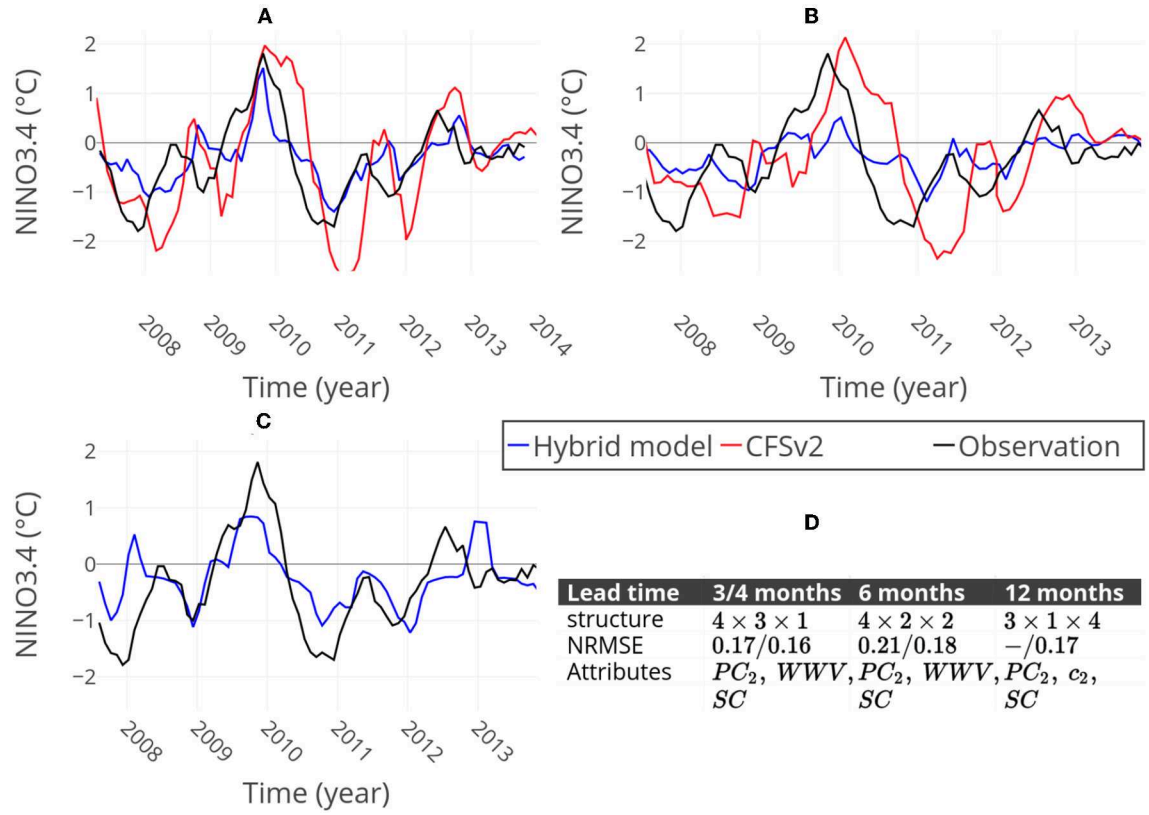
The prediction skill of the hybrid model decreased at a 6-months lead, while the shift and amplification of the CFSv2 prediction increased. Although the hybrid model did not suffer as much from the shift, at this lead time it underestimated (or missed) the El Niño event of 2010. In terms of NRMSE the hybrid model still obtained a better prediction skill than the CFSv2 (**Figures 5A,B**). The attributes from the shorter lead time predictions were found to be insufficient for the 12-months-lead prediction. However, c_2 of the SSH network was predictive at this lead time and hence the WWV was replaced by c_2. The 12-months lead time prediction of the hybrid model even improved the 6-months lead time prediction. On average the prediction did not contain a shift for this lead time (**Figure 5C**).

A prediction was made in Nooteboom et al. [73] for the year 2018, starting in May 2017 (**Figure 6A**). Different hybrid models were used at different lead times, always with ARIMA(12,1,0). The training set was from 1980 until May 2017 and the ANN structures used are the optimal ones at different lead times. For the predictions up to 5 months, the attributes WWV, PC_2, and SC were used whereas for the 12 months lead time prediction, the WWV was replaced by c_2. Here c_2 was computed from the SSALTO/DUACS altimetry dataset[3], which starts from 1993, and thus leads to a reduced training set. The hybrid model typically predicted much lower Pacific temperatures than the CFSv2 ensemble and was much closer to the eventual observations (black curve in **Figure 6A**). The uncertainty of the CFSv2 ensemble was large, since the spread of predictions is between a strong El Niño (NINO3.4 index between 1.5 and 2) and a moderate La Niña (NINO3.4 index between –1 and –1.5°C) for the following 9 months, being the ensemble average prediction close to a neutral state. The hybrid model of Nooteboom et al. [73] predicted development of a strong La Niña (NINO3.4 index lower than –1.5°C) the coming year. There was indeed a La Niña event in 2017/2018, although the NINO3.4 index remained above –1°C. A new prediction starting from December 2018 with the hybrid model is presented in **Figure 6B**, indicating the weak El Niño 2018–2019 to end by June 2019.

3.4. Prediction Uncertainty

In contrast to ensemble predictions of dynamical models, the proposed simple ANN-models lack the ability to estimate the predictive uncertainty. For instance, the ensemble spread of 84

[3]http://marine.copernicus.eu



D

Lead time	3/4 months	6 months	12 months
structure	$4 \times 3 \times 1$	$4 \times 2 \times 2$	$3 \times 1 \times 4$
NRMSE	$0.17/0.16$	$0.21/0.18$	$-/0.17$
Attributes	PC_2, WWV, SC	PC_2, WWV, SC	PC_2, c_2, SC

FIGURE 5 | NINO3.4 predictions of the CFSv2 ensemble mean (red) and the hybrid model of Nooteboom et al. [73] (blue), compared to the observed index (black). For the hybrid model predictions, ARIMA(12,1,0) was used and the eighty-four possible ANN structures with three hidden layers with up to four neurons each were tested. Results from the structures giving the lowest NRMSE are presented. **(A)** The 3-months lead time prediction of CFSv2 and 4-months lead time prediction of the hybrid model, **(B)** the 6-months lead time predictions and **(C)** 12-months lead prediction. The CFSv2 ensemble does not predict 12 months ahead. **(D)** Table containing information about all predictions: ANN optimal hidden-layers structures of the hybrid model, NRMSEs of the CFSv2 ensemble mean/NRMSE of the hybrid model, and attributes used in the hybrid model predictions. Reproduced from Nooteboom et al. [73] under open access license.

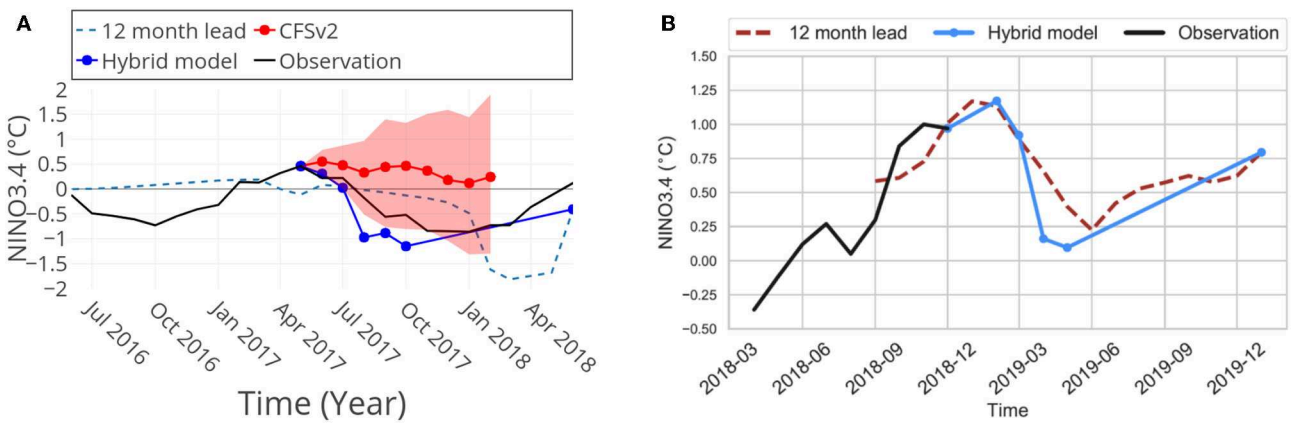


FIGURE 6 | (A) Result of the NINO3.4 prediction from May 2017 as in Nooteboom et al. [73]. The dashed blue line is the running 12-months lead-time prediction and in black the (later added) observed index. Red is the CFSv2 ensemble prediction mean and the shaded area is the spread of the ensemble. The hybrid model prediction in blue is given by predictions from hybrid models found to be most optimal at the different lead times, always with ARIMA(12,1,0) and starting on May 2017. **(B)** The most recent prediction of the hybrid model starting from December 2018. The black line is the observed index, blue line the prediction starting from December 2018, and dashed red line is the running 12-months lead-time prediction. **(A)** is slightly adapted from Nooteboom et al. [73] under open access license.

ANNs in Figure 8 of Nooteboom et al. [73] does not encompass most of the observed NINO3.4 index values. Hence, although the 84 ANN models had different architectures and initial weights (but were trained on the same training data set), the trained models predicted nearly the same NINO3.4 index values. Bootstrap-aggregating (bagging) methods [77] can be used to obtain a larger and more realistic ensemble spread. Another approach to better estimate the predictive uncertainty for neural network models is the so-called Bayesian neural networks (BNN). Here, all weights of the network have a distribution that can be learned by Bayesian inference. A first application to ENSO prediction in combination with a recurrent neural network (RNN) architecture is shown in McDermott and Wikle [78]. Unfortunately, the authors just present results for a short time period between 2015 and 2016. A comprehensive analysis of the application of BNNs for ENSO prediction is still lacking.

The endeavor of training a BNN is a far from trivial task. A simpler approach to estimate uncertainties in the prediction of ENSO is the application of the so-called Deep Ensembles (DEs) as presented in Lakshminarayanan et al. [79]. These DEs consist of multiple feed-forward neural network models that have two output neurons to predict the mean and the standard deviation of a Gaussian distribution. Instead of choosing the weights that minimize the mean-squared error, the models are trained by minimizing the negative log-likelihood of a Gaussian distribution with the predicted mean $\hat{\mu}$ and variance $\hat{\sigma}^2$, given the observation y, i.e.,

$$-\log P(y|\hat{\mu},\hat{\sigma}^2) = \frac{1}{2}\log\hat{\sigma}^2 + \frac{(y-\hat{\mu})^2}{2\hat{\sigma}^2} + constant\,, \qquad (3)$$

The final prediction for the variable and its uncertainty is obtained by combining the Gaussian distributions from all members of the ensemble. In plain words, if the model does not find strong relations between predictor variables and the predicted variable in the training data, it is still able to optimize the negative log-likelihood to some extent by increasing $\hat{\sigma}$. Therefore, it is less prone to be overconfident about any weak relationship in the data.

Here, we give an example [80] of the application of this method for the prediction of the NINO3.4 index. For this, a DE was trained to predict the future values of the 3-month running mean NINO3.4 index. To keep the example simple, the NINO3.4 index, WWV and SC were used as input variables, where for each variable the past 12 months were included in the feature set. Hence, each ANN had 36 inputs. Each ensemble member had one hidden layer with 16 neurons with a Rectified Linear Unit as activation function. The output neuron for the mean was equipped with a linear activation function and the output neuron for the standard deviation with a softplus function ($f(x) = \log(1 + e^x)$). To avoid overfitting, various regularization techniques were applied (early stopping, Gaussian noise to the inputs, dropout and L_1 as well as L_2 penalty terms). The training/validation period was set to be 1981-2002 and the test period 2002–2018. The training/validation data was further divided into 5 segments. One ensemble member was trained on 4 segments and validated on the remaining one to check for overfitting. This was repeated until each segment was one time the validation data set. Therefore, the DE had in total 5 ensemble members. Here, lead time was defined as in Barnston et al. [21] being the time that passed between the last date of the initial period and the first date of the target period.

Exemplary results for a 3-month lead-time prediction are shown in **Figure 7A**. In contrast to Nooteboom et al. [73], the confidence intervals of the prediction using the test data (blue line and shadings) could incorporate actual observation (black line) to a good extend. In fact, 55% were incorporated in the 1-standard deviation and 91% were inside the 2-standard deviation interval for the predictions on the test data. This indicated that such a prediction model could estimate the predictive uncertainty to a good extend. Interestingly, the predicted uncertainty had a seasonal cycle with lower uncertainties during boreal summer and higher uncertainties during boreal winter. This fitted the observations of the NINO3.4 values that follow the (same) seasonal cycle. The correlation skill on the test data set between the predicted mean and the observed NINO3.4 index is shown in **Figure 7B**. The relatively low skill values during the seasons AMJ to JAS indicates the spring predictability barrier. The overall correlation skill of the predicted mean on the test data set was 0.65 and the overall root-mean-square error 0.68.

4. DISCUSSION AND OUTLOOK

Machine Learning techniques are potentially useful to improve the skill of El Niño predictions. The choice of attributes is crucial for the degree of improvement. We have highlighted here the use of network science based attributes and the benefits of using physical knowledge to select them. Network variables provide global information on the building of correlations which occur when approaching an El Niño event, and knowledge of the physical mechanisms behind ENSO helps in determining which variables store relevant memory of the dynamics, and help to overcome the spring predictability barrier. Several network variables resulted in a clear success when applied to the ZC model [73], but not necessarily when predicting the real climatic phenomena. Work on the systematic identification of good attributes needs to be continued.

Most of the ANN studies to predict El Niño used simple architectures with a single hidden layer. Recently deeper architectures have been successfully tested [72, 73]. Nevertheless, a very complex ANN architecture will face the problem of overfitting, since the available time series are not very long and the number of parameters to optimize grows rapidly with ANN complexity. Probably, what makes El Niño prediction so challenging is that every event looks somehow different [17], and we still lack enough data to systematize these differences. Most of the methods aimed for a prediction model being most optimal in terms of least squares minimization. However, it could be interesting to put larger weight at predicting the extreme events in the optimization scheme. For example, the 6-months lead predictions of Nooteboom et al. [73] hybrid model missed the 2010 El Niño event (cf. **Figure 5**). Apart from this, it is important to investigate the exact reason why the

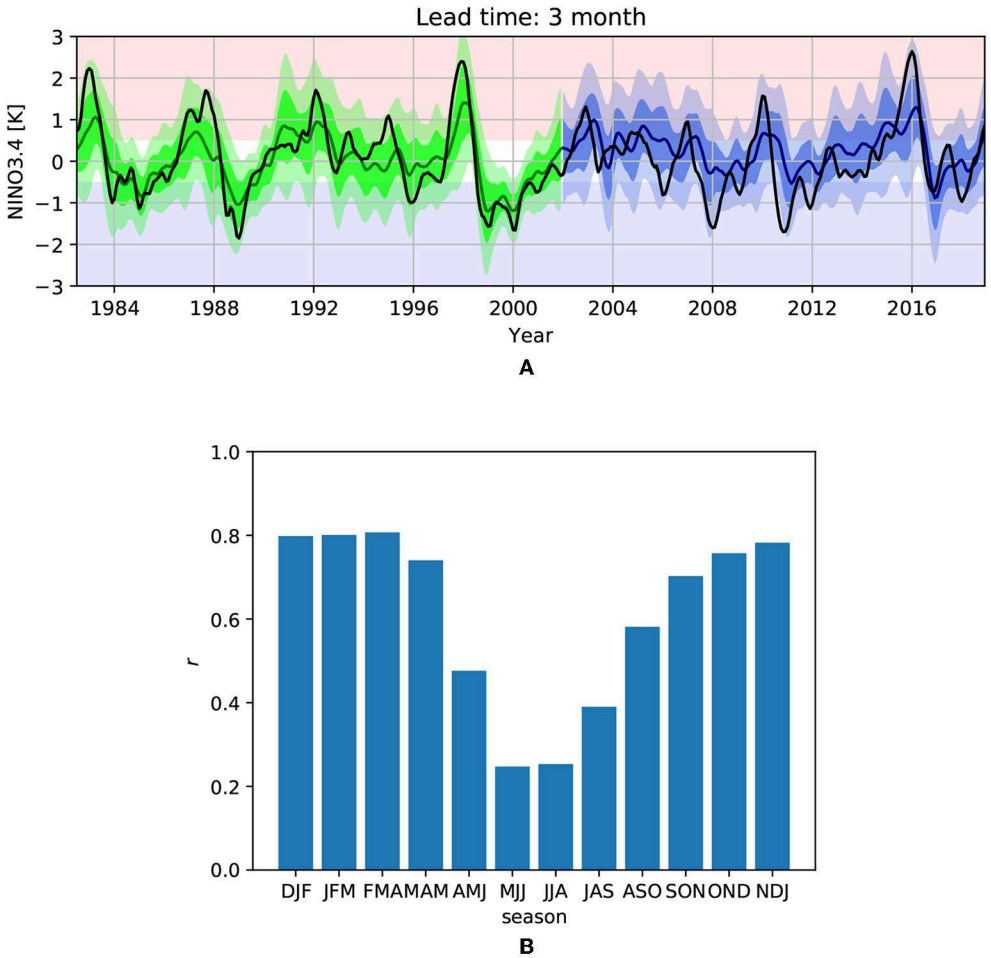


FIGURE 7 | (A) Results from the DE prediction approach [80]. Predictions for the 3-month lead time for the training data set (green) and the test data set (blue). The solid line indicates the mean of the predictions. The dark shading shows the 1-standard deviation confidence interval and the brighter shading the 2-standard deviation confidence interval. **(B)** Correlation skill of the predicted mean on the test data set of the DE for various seasons for the 3-month lead time.

hybrid model [73] provides such a good skill for a one-year lead time.

Despite the positive findings in applying ANNs for the ENSO prediction in work of the British Columbia group [58] or of Nooteboom et al. [73], the application of neural networks to ENSO prediction is still surrounded by inconsistent, non-transparent and unfavorable practices. Whereas, Tangang et al. [54] defined lead time as in Barnston et al. [21], i.e., as the time between the latest observed date and the first date of the target period, Wu et al. [58] defined lead time as the time from the center of the period of the latest predictors to the center of the target period. We suggest to use the definition of lead time as given in Barnston et al. [21], as also applied in Feng et al. [72] or Nooteboom et al. [73], in future research.

Furthermore, the problem of ENSO prediction is limited by a very low amount of data. Since 1980 there have been just 3–4 major El Niño (and a similar number of major La Niña) events. This little amount of data makes neural networks extremely susceptible to overfitting. To avoid this, it is necessary to regularize neural networks using methods such as Gaussian Noise layers, Dropout, Early Stopping, L_1 or L_2 penalty terms. Another problem that can arise due to the low amount of data is, that accidentally a signal in a variable exists in the training and the test data set, making the researcher confident that the model is a good generalization of the system. However, as the failure of the UBC-NNET model in the [21] study indicates, one has to be careful and not to put too much trust into the neural network predictions on ENSO considering the low amount of data. We advice not to use any variables as input to the neural network that do not have a justified reason to be a predictor for ENSO (e.g., the 9th leading EOF of the SSTA used in Wu et al. [58]).

The low amount of data, e.g., just three major El Niño events occurred since 1980, can make an educated choice of predictors very beneficial for the forecast model. This is because the ML-model cannot distinguish between relevant (deterministic concurrence) and non-relevant (random concurrence) information in a relatively large predictor

data set when the amount of training data is low. In general, if rather vague variables are used, there should be a method such as the L_1-penalty term, also called Lasso (least absolute shrinkage and selection operator), that is able to perform a feature selection and regularization [81].

Finally, past studies often did not provide the codes that they used for their results. This makes it increasingly difficult for the reader to build upon previous work and check the work for mistakes. Nowadays online platforms exist that make it easily possible to share code in a public repository and we advice that this should be the standard for any future research in the ML-ENSO prediction. To develop the idea of public available codes mentioned above one step further, we want to motivate that it would be very beneficial for this community to work together on a public repository that provides a framework for new investigations. All definitions, i.e., the lead time, as well as the used data sources with the applied preprocessing should be incorporated in this framework. Such a framework would lead to more transparency, prevent inconsistency between different research efforts as well as foster collaboration. A starting point for this endeavor could be the repository ClimateLearn published for the study of Feng et al. [72] on GitHub (https://github.com/Ambrosys/climatelearn).

AUTHOR CONTRIBUTIONS

The review was lead by HD. All authors contributed to writing of the paper.

FUNDING

The paper originated from a visit of HD to IFISC in January 2019 and was funded by the University of the Balearic Islands. EH-G and CL were supported by the Spanish Research Agency, through grant MDM-2017-0711 from the Maria de Maeztu Program for Units of Excellence in R&D. HD also acknowledges support from the Netherlands Earth System Science Centre (NESSC), financially supported by the Ministry of Education, Culture and Science (OCW), grant no. 024.002.001.

ACKNOWLEDGMENTS

We thank Peter Nooteboom and Qingyi Feng (IMAU, Utrecht University, The Netherlands) for their excellent work on ML-based ENSO prediction and for helping with several figures for this paper.

REFERENCES

1. Bishop CM. *Pattern Recognition and Machine Learning*. New York, NY: Springer (2006).
2. Kourou K, Exarchos TP, Exarchos KP, Karamouzis MV, Fotiadis DI. Machine learning applications in cancer prognosis and prediction. *Comput Struct Biotechnol J*. (2015) **13**:8–17. doi: 10.1016/j.csbj.2014.11.005
3. Heaton JB, Polson NG, Witte JH. Deep learning for finance: deep portfolios. *Appl Stochast Models Business Indust*. (2017) **33**:3–12. doi: 10.1002/asmb.2209
4. McCoy JT, Auret L. Machine learning applications in minerals processing: a review. *Miner Eng*. (2019) **132**:95–109. doi: 10.1016/j.mineng.2018.12.004
5. Blackwell WJ, Chen FW. *Neural Networks in Atmospheric Remote Sensing*. Boston, MA: Artech House (2009).
6. Haupt SE, Pasini A, Marzban C. *Artificial Intelligence Methods in the Environmental Sciences*. Berlin: Springer Verlag (2009).
7. Hsieh WW. *Machine Learning Methods in the Environmental Sciences: Neural Networks and Kernels*. Cambridge: Cambridge University Press (2009).
8. Schneider T, Lan S, Stuart A, Teixeira J. Earth system modeling 2.0: a blueprint for models that learn from observations and targeted high-resolution simulations. *Geophys Res Lett*. (2017) **44**:12,396–417. doi: 10.1002/2017GL076101
9. Dueben PD, Bauer P. Challenges and design choices for global weather and climate models based on machine learning. *Geosci Model Dev*. (2018) **11**:3999–4009. doi: 10.5194/gmd-11-3999-2018
10. Reichstein M, Camps-Valls G, Stevens B, Jung M, Denzler J, Carvalhais N, et al. Deep learning and process understanding for data-driven Earth system science. *Nature*. (2019) **566**:195–204. doi: 10.1038/s41586-019-0912-1
11. O'Gorman PA, Dwyer JG. Using machine learning to parameterize moist convection: potential for modeling of climate, climate change, and extreme events. *J Adv Model Earth Syst*. (2018) **10**:2548–63. doi: 10.1029/2018MS001351
12. Anderson GJ, Lucas DD. Machine learning predictions of a multiresolution climate model ensemble. *Geophys Res Lett*. (2018) **45**:4273–80. doi: 10.1029/2018GL077049
13. Scher S. Toward data-driven weather and climate forecasting: approximating a simple general circulation model with deep learning. *Geophys Res Lett*. (2018) **45**:12,616–22. doi: 10.1029/2018GL080704
14. Preisendorfer RW. *Principal Component Analysis in Meteorology and Oceanography*. Amsterdam: Elsevier (1988).
15. Rayner N, Parker D, Horton E, Folland C, Alexander L, Rowell D, et al. Global analyses of sea surface temperature, sea ice, and night marine air temperature since the late nineteenth century. *J Geophys Res*. (2003) **108**:4407. doi: 10.1029/2002JD002670
16. Adams RM, Chen CC, McCarl BA, Weiher RF. The economic consequences of ENSO events for agriculture. *Clim Res*. (1999) **13**:165–72.
17. McPhaden MJ, Timmermann A, Widlansky MJ, Balmaseda MA, Stockdale TN. The curious case of the EL Niño that never happened: a perspective from 40 years of progress in climate research and forecasting. *Bull Amer Meteor Soc*. (2015) **96**:1647–65. doi: 10.1175/BAMS-D-14-00089.1
18. Dee DP, Uppala SM, Simmons AJ, Berrisford P, Poli P, Kobayashi S, et al. The ERA-Interim reanalysis: configuration and performance of the data assimilation system. *Quart J Roy Meteorol Soc*. (2011) **137**:553–97. doi: 10.1002/qj.828
19. Latif M. Dynamics of interdecadal variability in coupled ocean-atmosphere models. *J Climate*. (1998) 11:602–24.
20. Chen D, Cane MA. El Niño prediction and predictability. *J Comput Phys*. (2008) **227**:3625–40. doi: 10.1016/j.jcp.2007.05.014
21. Barnston AG, Tippett MK, L'Heureux ML, Li S, DeWitt DG. Skill of real-time seasonal ENSO model predictions during 2002–11: is our capability increasing? *Bull Amer Meteor Soc*. (2012) **93**:631–51. doi: 10.1175/BAMS-D-11-00111.1
22. Timmermann A, An SI, Kug JS, Jin FF, Cai W, Capotondi A, et al. El Niño–southern oscillation complexity. *Nature*. (2018) **559**:535–45. doi: 10.1038/s41586-018-0252-6
23. Tang Y, Zhang RH, Liu T, Duan W, Yang D, Zheng F, et al. Progress in ENSO prediction and predictability study. *Natl Sci Rev*. (2018) **5**:826–39. doi: 10.1093/nsr/nwy105
24. Saha S, Moorthi S, Wu X, Wang J, Nadiga S, Tripp P, et al. The NCEP climate forecast system version 2. *J Clim*. (2014) **27**:2185–208. doi: 10.1175/JCLI-D-12-00823.1
25. L'Heureux ML, Takahashi K, Watkins AB, Barnston AG, Becker EJ, Di Liberto TE, et al. Observing and predicting the 2015/16 El Niño. *Bull Amer Meteor Soc*. (2017) **98**:1363–82. doi: 10.1175/BAMS-D-16-0009.1

26. Huang B, Thorne PW, Banzon VF, Boyer T, Chepurin G, Lawrimore JH, et al. Extended Reconstructed Sea Surface Temperature, Version 5 (ERSSTv5): upgrades, validations, and intercomparisons. *J Clim*. (2017) **30**:8179–205. doi: 10.1175/JCLI-D-16-0836.1
27. McPhaden MJ. Tropical pacific ocean heat content variations and ENSO persistence barriers. *Geophys Res Lett*. (2003) **30**:2705–9. doi: 10.1029/2003GL016872
28. Federov A, Harper S, Philander S, Winter B, Wittenberg A. How predictable is El Niño? *Bull Amer Meteor Soc*. (2003) **84**:911–9. doi: 10.1175/BAMS-84-7-911
29. Tziperman E, Stone L, Cane MA, Jarosh H. El Niño chaos: overlapping of resonances between the seasonal cycle and the Pacific ocean-atmosphere oscillator. *Science*. (1994) **264**:72–74.
30. Jin FF, Neelin JD, Ghil M. El Niño on the devil's staircase: annual subharmonic steps to chaos. *Science*. (1994) **264**:70–2.
31. Lian T, Chen D, Tang Y, Wu Q. Effects of westerly wind bursts on El Niño: a new perspective. *Geophys Res Lett*. (2014) **41**:3522–7. doi: 10.1002/2014GL059989
32. Roulston M, Neelin JD. The response of an ENSO model to climate noise, weather noise and intraseasonal forcing. *Geophys Res Lett*. (2000) **27**:3723–6. doi: 10.1029/2000GL011941
33. Eisenman I, Yu L, Tziperman E. Westerly wind bursts: ENSO's tail rather than the dog? *J Climate*. (2005) **18**:5224–38. doi: 10.1175/JCLI3588.1
34. Webster PJ. The annual cycle and the predictibility of the tropical coupled ocean-atmosphere system. *Meteor Atmos Phys*. (1995) 56:33–55.
35. Latif M, Barnett TP, Cane MA, Flügel M, Graham NE, von Storch H, et al. A review of ENSO prediction studies. *Clim Dynam*. (1994) **9**:167–79.
36. Zebiak SE, Cane MA. A model El Niño-Southern oscillation. *Mon Wea Rev*. (1987) **115**:2262–78.
37. Mu M, Duan W, Wang B. Season-dependent dynamics of nonlinear optimal error growth and ENSO predictability in a theoretical model. *J Geophys Res*. (2007) **112**:D10113. doi: 10.1029/2005JD006981
38. Duan W, Liu X, Zhu K, Mu M. Exploring the initial errors that cause a significant "spring predictability barrier" for El Niño events. *J Geophys Res*. (2009) **114**:C04022. doi: 10.1029/2008JC004925
39. Yu Y, Mu M, Duan W. Does model parameter error cause a significant "spring predictability barrier" for El Niño events in the Zebiak-Cane model? *J Climate*. (2012) **25**:1263–77. doi: 10.1175/2011JCLI4022.1
40. Horii T, Ueki I, Hanawa K. Breakdown of ENSO predictors in the 2000s: decadal changes of recharge/discharge-SST phase relation and atmospheric intraseasonal forcing. *Geophys Res Lett*. (2012) **39**:L10707. doi: 10.1029/2012GL051740
41. Chen D, Lian T, Fu C, Cane MA, Tang Y, Murtugudde R, et al. Strong influence of westerly wind bursts on El Niño diversity. *Nat Geosci*. (2015) **8**:339. doi: 10.1038/ngeo2399
42. Timmermann A, Jin FF, Abshagen J. A nonlinear theory of El Niño bursting. *J Atmospheric Sci*. (2003) **60**:165–76. doi: 10.1175/1520-0469(2003)060<0152:ANTFEN>2.0.CO;2
43. Guckenheimer J, Timmermann A, Dijkstra H, Roberts A. (Un)predictability of strong El Niño events. *Dynam Statist Climate Syst*. (2017) **2**:2399–412. doi: 10.1093/climsys/dzx004
44. Russell SJ, Norvig P. *Artificial Intelligence: A Modern Approach, 2nd Edn*. Upper Saddle River, NJ: Pearson Education (2003).
45. Affenzeller M, Wagner S, Winkler S, Beham A. *Genetic Algorithms and Genetic Programming: Modern Concepts and Practical Applications*. Boca Raton, FL: Chapman and Hall/CRC (2009).
46. Lukoševičius M, Jaeger H. Reservoir computing approaches to recurrent neural network training. *Comput Sci Rev*. (2009) **3**:127–49. doi: 10.1016/j.cosrev.2009.03.005
47. Álvarez A, Vélez P, Orfila A, Vizoso G, Tintoré J. Evolutionary computation for climate and ocean forecasting: "El Niño forecasting." In: Fiemming NC, Vallerga S, Pinardi N, Behrens HWA, Manzella G, Prandle D, et al., editors. *Operational Oceanography: Implementation at the European and Regional Scales, Vol. 66 of Elsevier Oceanography Series*. Amsterdam: Elsevier (2002). p. 489–94. Available online at: http://www.sciencedirect.com/science/article/pii/S0422989402800551
48. De Falco I, Della Cioppa A, Tarantino E. A genetic programming system for time series prediction and its application to El Niño forecast. In: Hoffmann F, Köppen M, Klawonn F, Roy R, editors. *Soft Computing: Methodologies and Applications*. Berlin; Heidelberg: Springer (2005). p. 151–62.
49. Lima AR, Cannon AJ, Hsieh WW. Nonlinear regression in environmental sciences using extreme learning machines: a comparative evaluation. *Environ Model Softw*. (2015) **73**:175–88. doi: 10.1016/j.envsoft.2015.08.002
50. Cybenko G. Approximation by superpositions of a sigmoidal function. *Math Control Signals Syst*. (1989) **2**:303–14.
51. Tangang FT, Hsieh WW, Tang B. Forecasting the equatorial Pacific sea surface temperatures by neural network models. *Climate Dynam*. (1997) **13**:135–47.
52. Tangang FT, Hsieh WW, Tang B. Forecasting regional sea surface temperatures in the tropical Pacific by neural network models, with wind stress and sea level pressure as predictors. *J Geophys Res Oceans*. (1998) **103**:7511–22.
53. Maas O, Boulanger JP, Thiria S. Use of neural networks for predictions using time series: Illustration with the El Niño Southern oscillation phenomenon. *Neurocomputing*. (2000) **30**:53–8. doi: 10.1016/s0925-2312(99)00142-3
54. Tangang FT, Tang B, Monahan AH, Hsieh WW. Forecasting ENSO events: a Neural Network-Extended EOF approach. *J Climate*. (1998) **11**:29–41.
55. Meinen CS, McPhaden MJ. Observations of warm water volume changes in the equatorial pacific and their relationship to El Niño and La Niña. *J Climate*. (2000) **13**:3551–9. doi: 10.1175/1520-0442(2000)013<3551:OOWWVC>2.0.CO;2
56. Yuval. Neural network training for prediction of climatological time series, regularized by minimization of the generalized cross-validation function. *Month Weather Rev*. (2000) **128**:1456–73. doi: 10.1175/1520-0493(2000)128<1456:NNTFPO>2.0.CO;2
57. Yuval. Enhancement and error estimation of neural network prediction of Niño-3.4 SST anomalies. *J Climate*. (2001) **14**:2150–63. doi: 10.1175/1520-0442(2001)014<2150:EAEEON>2.0.CO;2
58. Wu A, Hsieh WW, Tang B. Neural network forecasts of the tropical Pacific sea surface temperatures. *Neural Netw*. (2006) **19**:145–54. doi: 10.1016/j.neunet.2006.01.004
59. Baawain MS, Nour MH, El-Din AG, El-Din MG. El Niño southern-oscillation prediction using southern oscillation index and Niño3 as onset indicators: application of artificial neural networks. *J Environ Eng Sci*. (2005) **4**:113–21. doi: 10.1139/s04-047
60. Tang Y. Hybrid coupled models of the tropical Pacific. I: interannual variability. *Clim Dyn*. (2002) **19**:331–42. doi: 10.1007/s00382-002-0230-3
61. Tang Y, Hsieh WW. Hybrid coupled models of the tropical Pacific – II ENSO prediction. *Clim Dynam*. (2002) **19**:343–53. doi: 10.1007/s00382-002-0231-2
62. Tsonis AA, Swanson KL, Roebber PJ. What do networks have to do with climate? *Bull Am Meteorol Soc*. (2006) **87**:585–95. doi: 10.1175/BAMS-87-5-585
63. Donges JF, Zou Y, Marwan N, Kurths J. Complex networks in climate dynamics. *Eur Phys J Spec Top*. (2009) **174**:157–79.
64. Gozolchiani A, Havlin S, Yamasaki K. Emergence of El Niño as an autonomous component in the climate network. *Phys Rev Lett*. (2011) **107**:148501. doi: 10.1103/PhysRevLett.107.148501
65. Dijkstra HA, Hernández-García E, Masoller C, Barreiro M. *Networks in Climate*. Cambridge: Cambridge University Press (2019).
66. Ludescher J, Gozolchiani A, Bogachev MI, Bunde A, Havlin S, Schellnhuber HJ. Improved El Niño forecasting by cooperativity detection. *Proc Natl Acad Sci USA*. (2013) **110**:11742–5. doi: 10.1073/pnas.1309353110
67. Ludescher J, Gozolchiani A, Bogachev MI, Bunde A, Havlin S, Schellnhuber HJ. Very early warning of next El Niño. *Proc Natl Acad Sci USA*. (2014) **111**:2064–6. doi: 10.1073/pnas.1323058111
68. Stauffer D, Aharony A. *Introduction to Percolation Theory, 2nd Edn*. Philadelphia: Taylor and Francis Inc. (1994).
69. Rodríguez-Méndez V, Eguíluz M VM, Hernández-García E, Ramasco JJ. Percolation-based precursors of transitions in extended systems. *Sci Rep*. (2016) **6**:29552. doi: 10.1038/srep29552
70. Meng J, Fan J, Ashkenazy Y, Havlin S. Percolation framework to describe El Niño conditions. *Chaos*. (2016) **27**:1–15. doi: 10.1038/srep30993

71. Muscoloni A, Thomas JM, Ciucci S, Bianconi G, Cannistraci CV. Machine learning meets complex networks via coalescent embedding in the hyperbolic space. *Nat Commun*. (2017) **8**:1615. doi: 10.1038/s41467-017-01825-5
72. Feng QY, Vasile R, Segond M, Gozolchiani A, Wang Y, Abel M, et al. ClimateLearn: a machine-learning approach for climate prediction using network measures. *Geosci Model Dev Discuss*. (2016). doi: 10.5194/gmd-2015-273
73. Nooteboom PD, Feng QY, López C, Hernández-García E, Dijkstra HA. Using network theory and machine learning to predict El Niño. *Earth Syst Dynam*. (2018) **9**:969–83. doi: 10.5194/esd-9-969-2018
74. Shumway RH, Stoffer DS. *Time Series Analysis and Its Applications. 4th Edn*. New York, NY: Springer (2017).
75. Hibon M, Evgeniou T. To combine or not to combine: selecting among forecasts and their combinations. *Int J Forecast*. (2005) **21**:15–24.
76. Jin FF. An equatorial ocean recharge paradigm for ENSO. Part II: a stripped-down coupled model. *J Atmos Sci*. (1997) **54**:830–47.
77. Kotsiantis SB. Bagging and boosting variants for handling classifications problems: a survey. *Knowledge Eng Rev*. (2014) **29**:78–100. doi: 10.1017/S0269888913000313
78. McDermott PL, Wikle CK. Bayesian recurrent neural network models for forecasting and quantifying uncertainty in spatial-temporal data. *Entropy*. (2019) **21**:184. doi: 10.3390/e21020184
79. Lakshminarayanan B, Pritzel A, Blundell C. Simple and scalable predictive uncertainty estimation using deep ensembles. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, et al., editors. *Advances in Neural Information Processing Systems 30. Curran Associates, Inc.* Long Beach, CA (2017). p. 6402–13. Available online at: http://papers.nips.cc/paper/7219-simple-and-scalable-predictive-uncertainty-estimation-using-deep-ensembles.pdf.
80. Petersik P. *Machine Learning in El Niño Prediction*. MSc thesis. Utrecht University (2019).
81. Tibshirani R. Regression shrinkage and selection via the lasso. *J Roy Stat Soc Ser B*. (1996) **58**:267–88.

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Computer Science

ORIGINAL RESEARCH
published: 19 November 2019
doi: 10.3389/fcomp.2019.00009

Control of Automated Guided Vehicles Without Collision by Quantum Annealer and Digital Devices

Masayuki Ohzeki[1,2,3,4]*, *Akira Miki*[5], *Masamichi J. Miyama*[1,3,4] *and Masayoshi Terabe*[5]

[1] Graduate School of Information Sciences, Tohoku University, Sendai, Japan, [2] Institute of Innovative Research, Tokyo Institute of Technology, Kanagawa, Japan, [3] Sigma-i Inc., Tokyo, Japan, [4] J_{ij} Inc., Tokyo, Japan, [5] Electronics R & I Division, DENSO Corporation, Tokyo, Japan

Recent advance on quantum devices realizes an artificial quantum spin system known as the D-Wave 2000Q, which implements the Ising model with tunable transverse field. In this system, we perform a specific protocol of quantum annealing to attain the ground state, the minimizer of the energy. Therefore the device is often called the quantum annealer. However the resulting spin configurations are not always in the ground state. It can rather quickly generate many spin configurations following the Gibbs-Boltzmann distribution. In the present study, we formulate an Ising model to control a large number of automated guided vehicles in a factory without collision. We deal with an actual factory in Japan, in which vehicles run, and assess efficiency of our formulation. Compared to the conventional powerful techniques performed in digital computer, still the quantum annealer does not show outstanding advantage in the practical problem. Our study demonstrates a possibility of the quantum annealer to contribute solving industrial problems.

Keywords: quantum annealing, automated guided vehicle (AGV), optimization problem, Ising model, digital annealer

OPEN ACCESS

Edited by:
Raul Vicente, Max-Planck-Institut für Himforschung, Germany

Reviewed by:
Marcos César de Oliveira, Campinas State University, Brazil
Abdullah Makkeh, University of Göttingen, Germany

***Correspondence:**
Masayuki Ohzeki
mohzeki@tohoku.ac.jp

Specialty section:
This article was submitted to Quantum Computing, a section of the journal Frontiers in Computer Science

Received: *19 January 2019*
Accepted: *30 October 2019*
Published: *19 November 2019*

Citation:
Ohzeki M, Miki A, Miyama MJ and Terabe M (2019) Control of Automated Guided Vehicles Without Collision by Quantum Annealer and Digital Devices. Front. Comput. Sci. 1:9. doi: 10.3389/fcomp.2019.00009

1. INTRODUCTION

Quantum annealing is a technology recently attracting attentions from both of academic and business sides. It solves the unconstrained binary quadratic programming problem (recently also termed as the quadratic unconstrained binary optimization (QUBO) problem) written as the following cost function

$$E(\mathbf{q}) = \mathbf{q}^{\mathrm{T}} Q \mathbf{q}, \tag{1}$$

where $\mathbf{q}$ is a vector of binary variables and Q is a matrix characterizing the problem to be solved. Surprisingly, QA is realized in an actual quantum device using present-day technology (Berkley et al., 2010; Harris et al., 2010; Johnson et al., 2010; Bunyk et al., 2014). We call the device performing the protocol of QA as the quantum anneler. However the optimization problem, which includes the unconstrained binary quadratic programming problem, is solved following adequate algorithm on the digital computer. In this sense, QA is not necessarily an alternative way to solve the optimization problem but it rather provides Because QA is one of the natural computing, utilizing quantum tunneling effect, which escapes from local minima into a global minimum

(Kadowaki and Nishimori, 1998), compared to the conventional approach solving the optimization problem, it does without program a priori. In addition, the well-known quantum annealer, the D-Wave 2000Q, does not demand a huge amount of electric power for the computational part of the quantum devices compared to the high-performance computing. In this sense, QA is an optional way of computing, and main target of researches on QA can be searching its applicable situation in practical problems.

Unfortunately the range of applications is restricted to the case with the specific form as in Equation (1). The well-known optimization problem can be recasted by the form as in Equation (1) (Lucas, 2014), but the performance of QA is not necessarily revealed. The formulations of the specific form and QA for them have been tested such as portfolio optimization (Rosenberg et al., 2016), protein folding (Perdomo-Ortiz et al., 2012), the molecular similarity problem (Hernandez and Aramon, 2017), computational biology (Li et al., 2018), job-shop scheduling (Venturelli et al., 2015), election forecasting (Henderson et al., 2018), and machine learning (Crawford et al., 2016; Arai et al., 2018a; Khoshaman et al., 2018; Neukart et al., 2018; Ohzeki et al., 2018b; Takahashi et al., 2018). In addition, studies on implementing the quantum annealer to solve various problems have been performed (Arai et al., 2018a; Ohzeki et al., 2018a,b,b; Takahashi et al., 2018). The potential of QA might be boosted by the nontrivial quantum fluctuation, referred to as the nonstoquastic Hamiltonian, for which efficient classical simulation is intractable (Seki and Nishimori, 2012, 2015; Ohzeki, 2017; Arai et al., 2018b; Okada et al., 2019b). Most of them have not sufficed demand from practical situations as the size of the problems and time to solutions. Even one of the attractive formulations, the traffic optimization (Neukart et al., 2017), has not reached a level at the practical demand.

In a point of theoretical view, the potential performance of QA is well known. When the protocol of QA follows the quantum adiabatic condition, the ground state can be efficiently attained (Suzuki and Okada, 2005; Morita and Nishimori, 2008; Ohzeki and Nishimori, 2011b). This is not a realistic situation in performing QA in quantum devices such as D-Wave 2000Q. Thus, in the current version of quantum annealer, the attained solution is not always optimal owing to the limitations of devices and environmental effects (Amin, 2015). Although several protocols based on QA do not follow adiabatic quantum computation are proposed (Ohzeki, 2010; Ohzeki and Nishimori, 2011a; Ohzeki et al., 2011; Somma et al., 2012, the application of QA should be considered by taking account into an uncertain behavior of outputs from the quantum annealer. Recently, characteristic behavior on outputs of the quantum annealer is partially clarified. The outputs fall into a wide-flat valley of the cost function to be solved by QA rather than a sharp one (Kadowaki and Ohzeki, 2019). This fascinating property of QA is found in its application to the machine learning (Ohzeki et al., 2018a). The solutions in a wide-flat valley have robustness against the errors in the cost function. In the context of the machine learning, the errors in the cost function exist between formulations for the training and test data. However the solutions attained by QA shows good performance for the test data even although optimization is performed for the training data. In the case of formulating the optimization problem, we can not avoid the error in the cost function because we do not necessarily find the way to accomplish the desired task or we do not directly optimize the desired quantity by controlling the tunable parameters.

In the present study, we deal with the controlling problem of automated guided vehicles (AGVs), which are portable robots for moving materials in manufacturing facilities and warehouses (Ullrich, 2014; Fazlollahtabar et al., 2015; Fazlollahtabar and Saidi-Mehrabad, 2016), by use of the quantum annealer. The automated guided vehicles move along markers or wire on floors or uses vision, magnets, or lasers for navigation in a few cases. Currently, in most of factories, transportations of materials relies on AGVs and their smooth control. However, in limited-size factories, AGVs are frequently involved in traffic congestion around intersections because a large number of AGVs cross them simultaneously. Then we need a simple but smart system for controlling the AGVs without any collision. In the control of AGVs, rapid response is necessary for dealing with instantaneous changes in a system. Thus, it is expected that D-Wave 2000Q can provide a method for establishing the future infrastructure for controlling AGVs because it can output approximate solutions in a few tens of microseconds. The practical problem on facilities in actual factory has not been considered yet in the context of practical application of QA.

The remaining part of the paper is organized as follows: In the next section, we formulate the control of AGVs as the QUBO problem, which can be solved using D-Wave 2000Q. The solution does not always satisfy certain constraints for controlling AGVs, and output solutions must be postprocessed. We explain how to attain reasonable solutions via the postprocessing. In the third section, we solve the QUBO problem via D-Wave 2000Q and the corresponding integer programming via the Gurobi Optimizer (Gurobi Optimization, 2018) to check the validity of the solutions from the quantum annealer. In the following section, we report the results attained by D-Wave 2000Q and other solvers as references. In the last section, we summarize our study and discuss the direction of future work of the quantum annealer.

2. METHODS

We give the Ising model or QUBO problem for controlling AGVs in this section. Below we demonstrate their movements in the Japanese actual factory following our formulation, but it is generic and not specific to individual situations. We do not formulate the entire plan as QUBO problem to control all AGVs simultaneously. This is one of the essential bottleneck of the current version of quantum annealaer. We must reduce the number of binary variables to describe the problems within the maximum number of qubits in the quantum annealer, and simplify the formulation as far as possible. We consider iterative scheme to provide an adequate route for each AGV during time period T. At time t_0, we gather information on the location, x_i, and the task, s_i, distributed to each AGV. We solve our QUBO problem and employ its solution to control the AGVs

during time period T. After moving the AGVs at $t_0 + T$, we again gather information on the current situation and iterate the above procedure.

We focus on a controlling plan in time period T. We define the binary variable for each AGV as $q_{\mu,i} = 0, 1$, where μ is the index for a route and i is that for an AGV. The index of the route is selected from a set of routes, $M(x_i, s_i)$, where s_i is the given task for the i-th AGV. The index of i runs from 1 to N, which is the number of AGVs. The set of routes is constructed a priori following the tasks and the structure of the factory in which the AGVs run. One of the indicators for representing efficiency of the controlling AGVs is their waiting rate. The waiting rate is calculated by the ratio of the number of stopping AGVs and the total number of AGVs. However it is not straightforward to formulate the cost function to minimize the waiting rate. Instead we simply maximize the movements of AGVs while avoiding the collisions between them as

$$\begin{aligned} E(\mathbf{q}) = & -\sum_{i=1}^{N} \sum_{\mu \in M(x_i,s_i)} d_\mu q_{\mu,i} \\ & + \lambda_1 \sum_{i=1}^{N} \left(\sum_{\mu \in M(x_i,s_i)} q_{\mu,i} - 1 \right)^2 \\ & + \lambda_2 \sum_{e \in E} \sum_{t=1}^{T} \left(\sum_{i=1}^{N} \sum_{\mu \in M(x_i,s_i)} F_{\mu,t,e} q_{\mu,i} - 1 \right)^2, \end{aligned} \quad (2)$$

where E denotes all edges of the network along which the AGVs move in the factory, λ_1 and λ_2 are predetermined coefficients, and d_μ is the length of the route μ. The first term in Equation (2) is to achieve an efficient control of the AGVs, we define the simple cost function for increasing the total length in traveling of the AGVs. We count the total length of the routes employed by each AGV $d_\mu q_{\mu,i}$. The second and third terms represent the penalties for avoiding unfeasible solutions. The second term ensures that each AGV $q_{\mu,i}$ select a single route. The third term avoids collision between different AGVs for each t, which ranges from $t = 1$ to $t = T$ and each e, which denotes an edge in the routes for $F_{\mu,t,e} \neq 0$ in the factory. Here we define a binary quantity for characterizing the μ-th route as $F_{\mu,t,e}$ with 0 and 1. For each route, $F_{\mu,t,e} = 1$ on the edge occupied by the selected route, μ, at time t. On the contrary, $F_{\mu,t,e} = 0$ on the edge unoccupied by the selected route, μ, at time t.

We here add a comment on the relationship of our problem with the previous study for reducing the traffic flow of taxis in the literature (Neukart et al., 2017). The similar formulation was proposed for the traffic-flow optimization of moving taxis. However, the previous study did not consider the time dependence of $F_{\mu,t,e}$. In the present study, we assume that the speed of the AGVs is almost constant. In addition, the AGVs can move as expected and can be predicted precisely. They did not also include the length of tours for each taxi and time dependence on movement along the tour of each taxi. In order to more clarify the connection with the previous study, let us expand the third term in Equation (2). We then obtain a quadratic term as $\lambda_2 \sum_{e \in E} \sum_{t=1}^{T} \left(\sum_{i=1}^{N} \sum_{\mu \in M(x_i,s_i)} F_{\mu,t,e} q_{\mu,i} \right)^2$ and a linear term as $-2\lambda_2 \sum_{i=1}^{N} \sum_{\mu \in M(x_i,s_i)} d_\mu q_{\mu,i}$, because $\sum_{e \in E} \sum_{t=1}^{T} F_{\mu,t,e} = d_\mu$. When $\lambda_2 = 1$, the first term in Equation (2) vanishes with the resultant linear term and then the cost function (2) coincides with that in the previous study. In this sense, the present study is an extension of the previous one. We apply our formulation straightforwardly to the optimization problem on the traffic flow of taxis.

Once we formulate the QUBO problem, we immediately generate the binary configurations as the outputs of the D-Wave 2000Q. We attain numerous outputs from D-Wave 2000Q for the same QUBO problem in a short time. In our case, we set the annealing time to attain a single output as 20 [μs] due to limitation of the quantum coherence time. It is thus difficult to certainly attain the ground state of the QUBO problem. In this sense, the quantum annealer does not work well for solving the optimization problem. The short annealing time is a bottleneck of the D-Wave 2000Q in a sense. However the outputs can be quickly attained. Let us here take the bottleneck as advantage of the D-Wave 2000Q. We generate many of outputs from the D-Wave 2000Q as sampling of binary configurations. The samples follow the Gibbs-Boltzmann distribution of the QUBO problem but with a finite strength of the quantum fluctuation as discussed in the literature (Amin, 2015). However, the solutions employed to control the AGVs must satisfy all constraints. We then filter out the outputs that do not satisfy the constraints from those of D-Wave 2000Q. As a result, we obtain feasible solutions without collisions and the multiple selection of routes. We check the efficiency of our postprocessed solutions in the next section to verify the capability of the D-Wave 2000Q in a limited practical application such as controlling the AGVs in factories.

We formulate the QUBO problem for the quantum annealer to contribute to the practical application appearing in various factories but we may utilize other solvers rather than the D-Wave 2000Q. In the present study, we also test the Fujitsu digital annealer (DA), which can solve the QUBO problem quickly as the D-Wave 2000Q (Tsukamoto et al., 2017; Aramon et al., 2018).

In order to check the validity of our QUBO problem, which is not a direct formulation of the efficiency controlling the AGVs, we solve it in an adequate way. Our formulation can be reformulated as the integer programming as

$$\max_{\mathbf{q}} \left\{ \sum_{i=1}^{N} \sum_{\mu \in M(x_i,s_i)} d_\mu q_{\mu,i} \right\}, \quad (3)$$

$$\text{s.t.} \sum_{\mu \in M(x_i,s_i)} q_{\mu,i} = 1 \quad \forall i \quad \text{and} \quad \sum_{i=1}^{N} \sum_{\mu \in M(x_i,s_i)} F_{\mu,t,e} q_{\mu,i} = 1 \quad \forall t, \forall e.$$

We solve this integer programming by the branch and bound method via the Gurobi Optimizer (Gurobi Optimization, 2018) to confirm validity of our formulation.

We describe our whole system for controlling the AGVs in **Figure 1**. In order to shorten the time of the whole procedure to control the AGVs, we prepare a database that stores the set of routes when we create the QUBO during the time period T. In advance, we generate the shortest paths from an origin to a

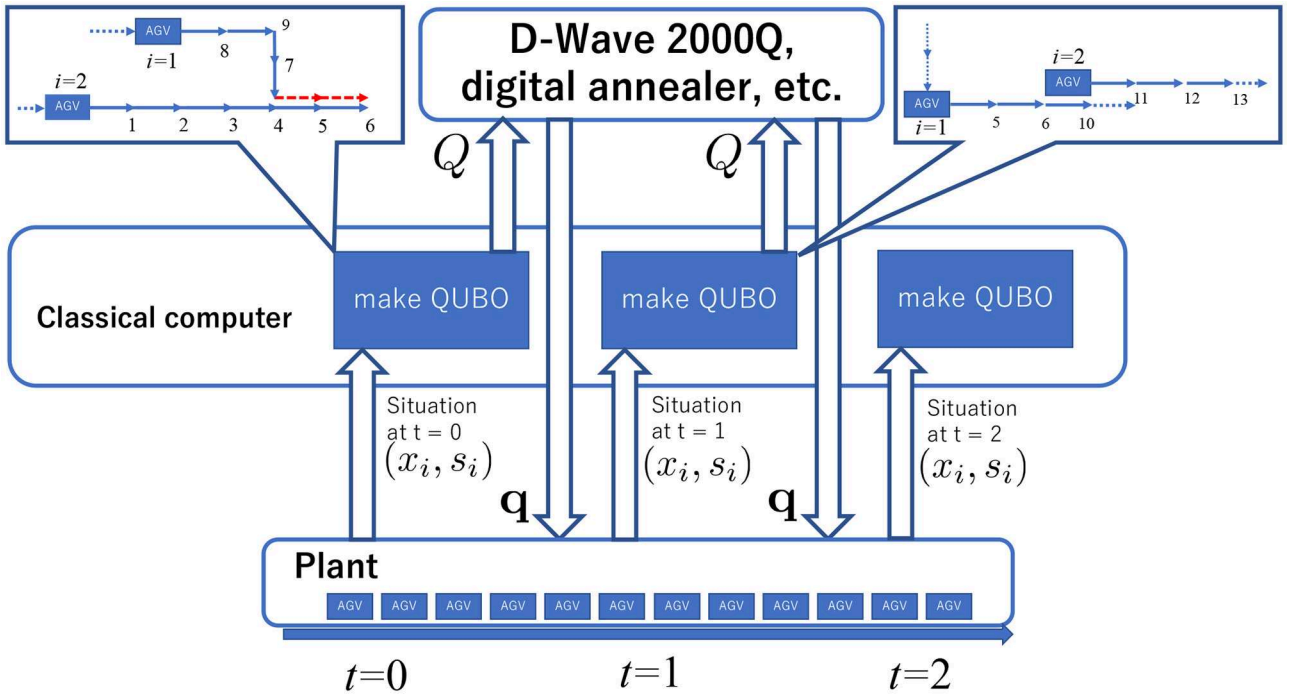


FIGURE 1 | Our system for controlling AGVs. In the sector of classical computer, we prepare the QUBO at each time according to the current situation of the factory (x_i, s_i). For this task, we have database storing the set of routes.

destination for each task. We divide the shortest paths into sets of several vertices at the longest vT, where v is the maximum speed of the AGVs, and store them. When we build the QUBO matrix, we only elucidate a vertex set included in a part of the shortest paths for achieving the given task beginning at x_i up to the reachable position at the end of period T. For instance, let us consider the case as in **Figure 1**. We take the first AGV at $x_1 = 8$ at $t = 0$, which has the shortest path of the route for achieving its task consisting of the node set $\{8, 9, 7, 4, 5, 6\}$. Then, we prepare the route set as $\{8\}$, $\{8, 9\}$, $\{8, 9, 7\}$, $\{8, 9, 7, 4\}$, $\{8, 9, 7, 4, 5\}$, and $\{8, 9, 7, 4, 5, 6\}$, which indicate "stop," "1 step ahead," and "2 steps ahead," etc.. The second AGV at $x_2 = 1$ at $t = 0$ has the route set at $\{1\}$, $\{1, 2\}$, $\{1, 2, 3\}$, $\{1, 2, 3, 4\}$, $\{1, 2, 3, 4, 5\}$, and $\{1, 2, 3, 4, 5, 6\}$. In order to increase the total length of the routes, two AGVs prefer to select $\{8, 9, 7, 4, 5, 6\}$ and $\{1, 2, 3, 4, 5, 6\}$, respectively. However the third term in Equation (2) does not allow this solution. The overlap between two routes increases the value of $f(\mathbf{q})$. The minimization of $f(\mathbf{q})$ avoids collision between two AGVs and select the solution with $\{8, 9, 7, 4\}$ and $\{1, 2, 3, 4, 5, 6\}$, or $\{8, 9, 7, 4, 5, 6\}$ and $\{1, 2, 3, 4\}$. The solution of $\mathbf{q}$ comes from the D-Wave 2000Q, the digital annealer etc.. in short time. As detailed below, we need to filter out the infeasible solutions satisfying the constraints for safely controlling the AGVs in practice. The whole time for the above procedure should be short for efficient control of the AGVs. We ere utilize the current version of the quantum annealer, which does not necessarily find the optimal solutions of our QUBO problem but quickly generates feasible solutions. Below we confirm availability of the D-Wave 2000Q in our proposed system by simulating the whole system utilizing the feasible solutions attained from our scheme.

3. RESULTS

In this section, we report the results attained by iteratively solving the QUBO problem by using the D-Wave 2000Q at each time period for controlling the AGVs. For proving the efficiency of our method, we prepare a simulation environment for an actual factory as shown in **Figure 2**. The map is one of the actual factories in Japan. Although we below take a single map as a test of our formulation, we prepare different situations by increasing the number of AGVs and changing initial conditions. These are the essentially different situations in terms of that we attain a completely different matrix Q.

First, we test our formulation in the real setting of the actual factory. The factory usually utilizes 10 AGVs for product delivery, and the AGVs move simultaneously along four fixed routes according to predetermined tasks. We generate six candidates movement of each AGV. Thus the maximum size of the QUBO matrix is 60, which is embeddable in the D-Wave 2000Q. Notice that the QUBO matrix becomes very sparse in our formulation. Thus we further enlarge the size of the problem without an efficient embedding program (Okada et al., 2019a,c). The speed of each AGV is 0.5 m/s. The distance between nodes is 10 m.

We simulate the controlled AGV movement following the results by the following different methods. One is the conventional method, and the other is our method attained by

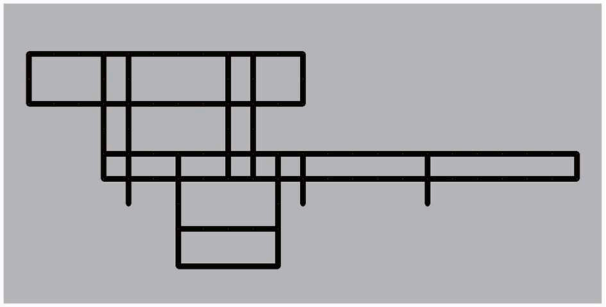


FIGURE 2 | Factory used in the present study. In this factory, 10 AGVs move along a road and complete their tasks.

the outputs from D-Wave 2000Q. Notice that the conventional method for controlling the AGVs is a rule-based method at every intersection in the actual factory. The rule is that when the AGVs require the same intersection route, only one AGV can move in and out at the intersection. For example, when two AGVs require the same intersection, one AGV waits until the other AGV leaves the intersection. The AGVs that move along the circumference of the factory have higher priority for entering an intersection for increasing the working rate. On the other hand, we solve the QUBO problem via D-Wave 2000Q at each time period. The time period is set to be 3 s, namely $T = 3$ [s]. We set the parameters as $\lambda_1/(1+\lambda_2) = 1.0$ and $\lambda_2/(1+\lambda_2) = 2.0$. Because D-Wave 2000Q does not deal with large elements of QUBO matrix Q_{ij}, the elements of the QUBO matrix is rescaled within the range of the available magnitude. D-Wave 2000Q solves the QUBO problem 1000 times for finding reasonable solutions. We filter the solutions that do not satisfy the constraints and select one of the reasonable solutions for moving the AGVs further. The AGVs move following the selected solution during the time period of 3 s. The solution indicates the movement in the next 5 s. Thus, the movement of the AGVs is updated before they reach the end of the given route.

First the results attained by the conventional method and D-Wave 2000Q are shown in **Figure 3**. We simulate the AGVs in the actual factory for 1,000 s and indicate the accumulated waiting time by circles. The waiting rate is calculated by the ratio of the number of stopping AGVs and the total number of AGVs. Several circles represent the locations that frequent traffic jams of the AGVs happened. The size of a circle is proportional to the accumulated waiting time of the AGVs at that point. In the case of the conventional method, the time average of the waiting rate converges to 20%. On the other hand, in the case with the D-Wave 2000Q, It can be seen from **Figure 3** that the number of circles, which represent the accumulated waiting time, is considerably reduced compared to the conventional method. The time average of the waiting rate converges to 5%. The actual movement of the AGVs from an initial condition is shown in the **Supplemental Video Files**. Compared to the result of the conventional method, the AGVs move smoothly following the solution attained by our method with the D-Wave 2000Q. The readers can find the smooth movements of the AGVs in the **Supplemental Video Files**.

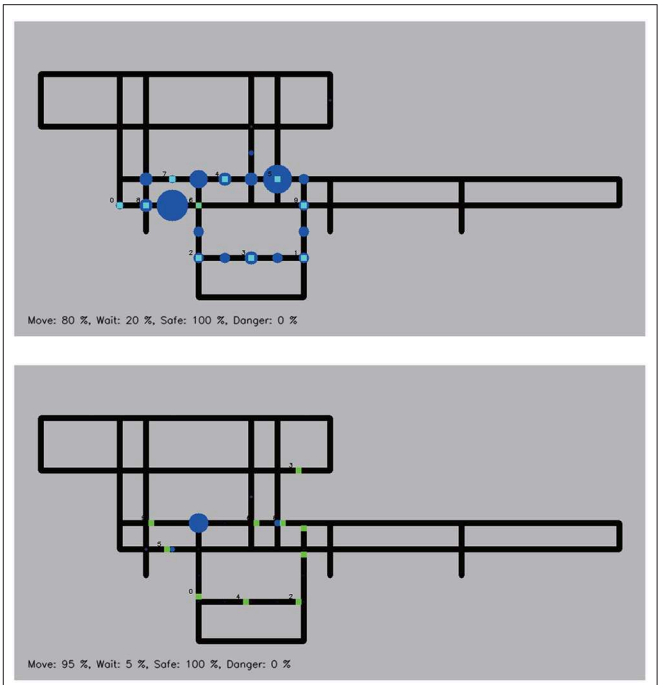


FIGURE 3 | Comparison among the solvers: (upper) Conventional method and (bottom) D-Wave 2000Q. The green dots denote the locations of the AGVs at the end. The blue circles represent the accumulated waiting time for the AGVs.

4. OTHER SOLVERS AND VALIDITY OF FORMULATION

It is not necessary to solve our QUBO problem using D-Wave 2000Q; one can utilize other solvers. One method is the DA, which solves the QUBO problem using an improved version of SA. Notice that the DA can solve the QUBO problem with a large number of binary variables compared to D-Wave 2000Q. The number of binary variables in our QUBO problem is 60, which is the product of the actual number of the AGVs (10) in the Japanese factory and the number of candidates of routes (6), which is set a priori so as to be embedded on the D-Wave 2000Q. Thus, the number of the binary variables is quite small. Even though the DA does not exhibit its potential efficiency in this case, we find that the time average of the waiting rate converges around 6% as shown in **Figure 4**. Similarly to the case with the D-Wave 2000Q, DA also leads to nice performance to control the AGVs by use of our formulation.

In addition, in order to verify our formulation of the QUBO problem, we solve the corresponding integer programming through the relaxation of the binary variables to continuous variables by utilizing the branch and bound method via Gurobi Optimizer version 8.01 on a 4-core Intel i7 4770K processor with 32 GB RAM. In this case, we attain the optimal solution of the corresponding integer programming in a very short time and utilize the optimal solution to control the AGVs. Similarly to the previous results attained by the D-Wave 2000Q and DA, the optimal solutions controlls the AGVs without collisions.

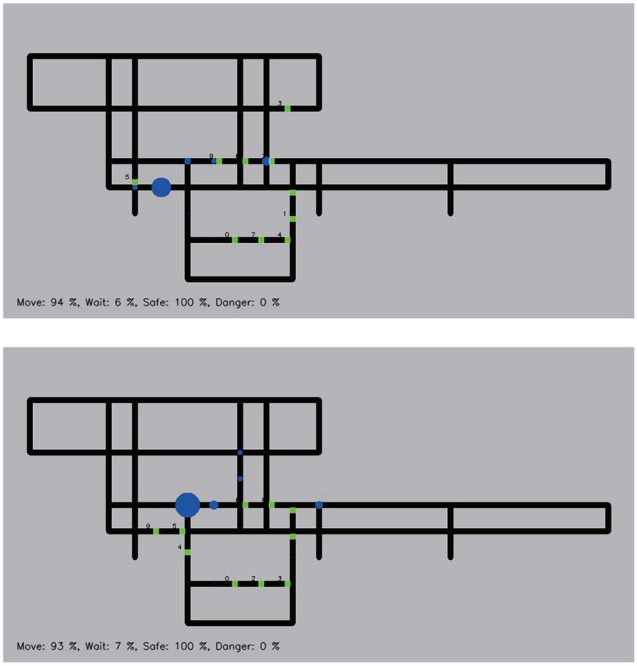


FIGURE 4 | Comparison among the solvers: (upper) Fujitsu digital annealer and (bottom) Gurobi Optimizer as a reference. The same symbols are used in **Figure 3**.

TABLE 1 | Working rates of the AGVs obtained by the conventional method, D-Wave 2000Q, Fujitsu digital annealer, Gurobi Optimizer, and modified optimization problem for Gurobi Optimizer.

2pt	Conventional	D-Wave 2000Q	Fujitsu digital annealer	Gurobi	Gurobi +
Average	80	94.2 ± 1.2	93.4 ± 1.2	93	96
Max	80	96	94	93	96

The time average of the waiting rate converges to 7%, which is slightly higher than the results of D-Wave 2000Q and DA. This is due to stochasticity of D-Wave 2000Q and the DA. The cost function itself is not necessarily a direct indicator of performance. Thus the optimal solution for the cost function is not always optimal for the actual performance in terms of the waiting rate. Similar phenomena appear in machine learning. Generalization performance, which is the measure of potential power in machine learning but not directly related to the cost function to be optimized, can be enhanced via stochastic methods to optimize cost functions. In particular, QA actually leads to better generalization performance, as shown in the literature (Ohzeki et al., 2018a). This is indirect evidence of the robustness of the solutions in the wide-flat minimum attained in the quantum annealer as reported in the literature (Kadowaki and Ohzeki, 2019).

In order to assess the typical performance of our QUBO problem, we repeat the iterative optimization for controlling the AGVs at each time period starting from the same initial condition 10 times. Because the D-Wave 2000Q and DA have stochasticity, we compute the average and maximum performance, as shown in **Table 1**. As shown in **Table 1**, the variances among the different runs are small for each solver. The Gurobi Optimizer always leads to the optimal solutions, but the waiting rates are not less than the results obtained by D-Wave 2000Q and DA. This is because the optimal solutions do not always lead to the best control of the AGVs in terms of the waiting rate. Our QUBO problem is not directly related to the waiting rate. In order to reduce the waiting rate, we add another constraint for the AGVs such that if several AGVs reach the same intersection, the AGV with more following AGVs is preferentially allowed to enter the intersection. We solve the improved integer programming with the additional constraint by employing the Gurobi Optimizer and also show its efficiency in **Table 1**. As shown in **Table 1**, the waiting rate is reduced by the improved integer programming and the result is comparable with the D-Wave 2000Q and DA with stochasticity. As well known, the integer programming can be easily improved by considering deeply the structure of the target problem. In addition, the digital computer can accept any formulation of the integer programming. This is the most advantage of the digital computer. The quantum annealer is not acceptable for an intricate QUBO problem due to the limitation of the quantum device. However our QUBO problem is simple but valuable for the quantum annealer to control the AGV in the factory, which is one of the important problems in industry. This is the first evidence showing possibility for the quantum annealer to contribute on the practical application although it has many bottlenecks to be solved.

Below, we discuss the efficiency of the solvers from another point of view, the computational time. We investigate the "actual" computational time, which is obtained in a standard-user environment, and the quality of the attained solutions against the increase in the number of the AGVs and candidate routes. We prepare a hundred of different initial locations of the AGVs such as each pair of the AGVs encounter at an intersection and solve the optimization problem. We report the comparison results in average and variance below.

The D-Wave 2000Q takes 20 μs, which is predetermined by users, to once solve the optimization problem in the quantum chip with superconducting qubits. However preprocessing and postprocessing for preparation to solve the optimization problem, the latency of the network when we utilize the D-Wave 2000Q via cloud service, and the queueing time can not be avoided. Thus the actual computational time takes a little bit longer. The D-Wave 2000Q outputs many samples of the solutions once. We set the number of samples as 1,000 and measure the actual computational time. We then estimate the actual computational time per output sample as 1.39(33) ms for 9 spins, 1.33(11) ms for 21 spins, 1.51(5) ms for 30 spins, 1.45(12) ms for 39 spins, 1.90(16) ms for 51 spins, and 2.22(22) ms for 60 spins. These computational times per output sample are only to solve the QUBO problems without any assurance of precision of the attained solutions. The probability for attaining the ground state P_0 gradually decreases as the number of spins increases. In fact, $P_0 = 1.00$ for 9 spins, $P_0 = 0.99(6)$ for 21 spins, $P_0 = 0.97(2)$

for 30 spins, $P_0 = 0.91(1)$ for 39 spins, $P_0 = 0.87(2)$ for 51 spins and $P_0 = 0.74(2)$ for 60 spins.

The number of binary variables consists of the multiplication of that of the AGVs and the routes. The computational time drastically increases for the case of D-Wave 2000Q beyond 60 spins. This is due to the limitation of the number of binary variables to be solved simultaneously. We solve the case with a larger number of binary variables by utilizing qbsolv, which divides the original problem into a number of small problems. To iteratively use D-Wave 2000Q, we must wait for several seconds owing to the job queue via the cloud service provided by the D-Wave systems Inc. at each iteration to solve the small problems. The actual computational time per output sample and iteration is 1.80(44) ms for 90 spins, 1.77(59) ms for 399 spins, and 1.37(53) ms for 900 spins. The iteration numbers become 2×10 for 90 spins, 8×10 for 399 spins, and 33×10 for 900 spins. The former number in the product is the number of division of the original large problem into small subproblems, and the latter one is that of repetition to solve the optimization problem. Thus, the actual computational time can be extremely long. In addition, the probabilities for attaining the ground state get worse as $P_0 = 0.27(12)$ for 90 spins $P_0 = 0.03(5)$ for 399 spins and $P_0 = 0.001(9)$ for 900 spins. This is a weak point to employ the D-Wave 2000Q to solve the QUBO problem. Although it seems that the computational time does not depend on the number of binary variables, the probability for attaining the ground state gradually decreases as the number of binary variables increases. On the other hand, the Gurobi Optimizer leads to the optimal solutions for each case. Its computational time to attain the optimal solution depends on the number of binary variables. 2.79(6) ms for 30 spins, 3.46(5) for 60 spins 4.25(6) for 90 spins, and 8.70(6) ms for 400 spins.

On the other hand, for the DA, the machine time is set to be enough to solve the optimization problem about 8 ms. The actual computational time per output sample takes a little bit longer than the machine time as 0.216(2) s for 9 spins, 0.219(4) s for 21 spins, 0.222(6) s for 30 spins, 0.220(7) s for 39 spins, and 0.232(9) for 51 spins, 0.240(11) for 60 spins, 0.230(6) ms for 90 spins, 0.336(18) s for 399, and 0.519(32) ms for 900 spins. Up to 1,024 spins, the current version of the DA can solve once the optimization problem without dividing it into small subproblems. This is an advantage point of the DA in comparison with the D-Wave 2000Q. In addition, the probability for attaining the ground state P_0 is relatively higher compared to that of the D-Wave 2000Q as $P_0 = 1.0$ for 9, 21, 30, 39, and 51 spins, $P_0 = 1.000(5)$ fpr 60 spins, $P_0 = 0.97(3)$ for 90 spins, $P_0 = 0.71(3)$ for 399 spins, and 0.37(12) for 900 spins. Notice that the higher value of the probability for attaining the ground state is obtained by tuning the annealing schedule. Instead, the actual computational time takes longer.

We compute the time to solutions (TTS) defined as

$$\mathrm{TTS}(p) = t_c \frac{\log(1-p)}{\log(1-P_0)}, \tag{4}$$

where t_c is the actual computational time per output sample and p is a predetermined precision to attain the ground state. The time to solution is an indicator of the performance of the solver in the stochastic way. We show the comparison data of TTS (0.99) of the D-Wave 2000Q and the DA and the actual computational time of the Gurobi Optimizer in **Figure 5**. In the successful cases with $P_0 = 1.0$, we plot the actual computational time instead of the TTS. The actual computational time per output sample can be upper bound for the TTS.

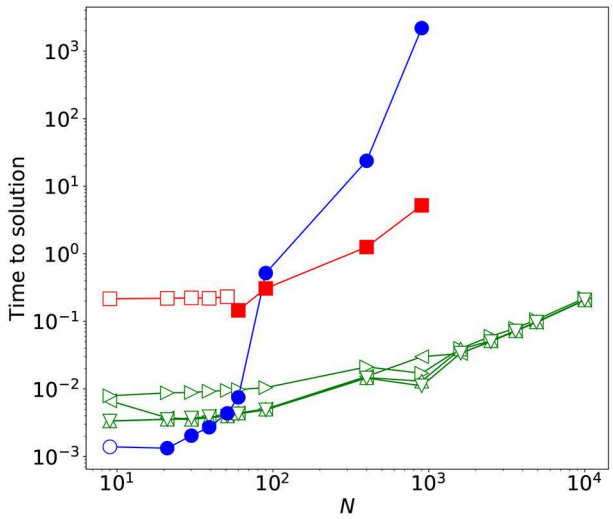


FIGURE 5 | Comparison of the time to solution (TTS). The horizontal axis denotes the number of AGV N and the vertical one represents time to solution in seconds. The filled circles and squares denote the TTS obtained by D-Wave 2000Q and the DA, respectively. The outlined circles and squares represent the actual computational time (upper bound of the TTS) by the D-Wave 2000Q and the DA. In addition, we plot the actual computational time by the Gurobi Optimizer by the triangles. The directions of the triangles distinguish the results by different levels of the "presolve" option for the Gurobi Optimizer as "default," "none," "conservative," and "aggressive".

5. CONCLUSIONS

We formulate the QUBO problem for controlling the AGVs in the actual factory in Japan. This is the first step of the practical application of the quantum annealer to the actual situation in industry. In order to reduce the number of binary variables, which is embeddable on the D-Wave 2000Q, we do not deal with the whole control of the AGVs but iterate the procedure in the predetermined time period, $T = 3$ s. The numbers of the binary variables that can be solved within $T = 3$ s, which is determined by the product of the numbers of AGVs and routes, are up to $\sim$ 400 for D-Wave 2000Q with a technique of division of the large problem, known as qbsolv, $\sim$ 90 for the DA, and over 10,000 for the Gurobi Optimizer in terms of the TTS and the actual computational time to attain the optimal solution. Notice that, in order to control the AGVs, it is not necessarily to find the optimal solutions. In this sense, the present study discovers possibility of the current version of the quantum annealer for contributing on the practical applications in the actual situation in industry. We emphasize that our formulation

is very simple to control the AGVs, which can be mapped into the integer programming. The quantum annealer is not acceptable for an intricate QUBO problem, as in our formulation, due to the limitation of the quantum device. The digital computer can accept any formulation of the integer programming. Thus further improvements of our formulation will be achievable by considering a better problem setting. This is the most advantage of the digital computer. In this sense, this is the first evidence showing possibility for the quantum annealer to contribute on the practical application although it has many bottlenecks to be solved.

Notice that we employ the actual computational time basically to estimate the performance of the D-Wave 2000Q and the DA, not the machine time. In future, if we can avoid the latency of the communication and queuing time for dealing with the jobs to solve the optimization problem in both of the devices via cloud services, better efficiency can be achieved. In this sense, the computational time of the D-Wave 2000Q and the DA can be reduced significantly. For instance, the machine time for solving the QUBO problem by the D-Wave 2000Q can be set to be 20μs and that of the DA is 8 ms. The D-Wave 2000Q can be a candidate for controlling the AGVs in real factories. The time period was set in the present study following the current situation of the real factory, in which several workers walks, In the cases without any workers, the AGVs can move faster than the setting of the present study. Then shorter response time for controlling the AGVs is necessary. The next-generation quantum annealer beyond the D-Wave 2000Q is expected as a candidate for controlling the AGVs in such future factories. The D-Wave quantum processing units continues to steadily grow in number of qubits. The precision to find the ground state getting better, the TTS becomes shorter. In this sense, the shorter response time can be achieved and such future factories can be created by the next-generation quantum annealer, although the current version, the D-Wave 2000, is just a proof of concept. In the intermediate stage, the hybrid computation of the digital computer and the quantum annealer, or several simulations on the digital hardware are valuable as discussed in the literatures (Ohzeki, 2019; Waidyasooriya et al., 2019). Although the digital computer works quite well at the level of our formulation only with a few ingredients to control the AGVs, the present study is the first step toward the efficient control of AGVs in future factories as one of the candidates in the real-world application of the QA.

DATA AVAILABILITY STATEMENT

The datasets generated for this study are available on request to the corresponding author.

AUTHOR CONTRIBUTIONS

All authors listed have made a substantial, direct and intellectual contribution to the work, and approved it for publication.

FUNDING

This present work was financially supported by MEXT KAKENHI Grant Nos. 15H03699 and 16H04382, and by JST START.

ACKNOWLEDGMENTS

The authors would like to thank Shu Tanaka for fruitful discussions.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fcomp.2019.00009/full#supplementary-material

REFERENCES

Amin, M. H. (2015). Searching for quantum speedup in quasistatic quantum annealers. *Phys. Rev. A* 92:052323. doi: 10.1103/PhysRevA.92.052323

Arai, S., Ohzeki, M., and Tanaka, K. (2018a). Deep neural network detects quantum phase transition. *J. Phys. Soc. Jpn.* 87:033001. doi: 10.7566/JPSJ.87.033001

Arai, S., Ohzeki, M., and Tanaka, K. (2018b). Dynamics of order parameters of non-stoquastic hamiltonians in the adaptive quantum Monte Carlo method. *Phys. Rev. E* 99:032120. doi: 10.1103/PhysRevE.99.032120

Aramon, M., Rosenberg, G., Valiante, E., Miyazawa, T., Tamura, H., and Katzgraber, H. G. (2018). Physics-inspired optimization for quadratic unconstrained problems using a digital annealer. *Front. Phys.* 7:48. doi: 10.3389/fphy.2019.00048

Berkley, A. J., Johnson, M. W., Bunyk, P., Harris, R., Johansson, J., Lanting, T., et al. (2010). A scalable readout system for a superconducting adiabatic quantum optimization system. *Superconduct. Sci. Technol.* 23:105014. doi: 10.1088/0953-2048/23/10/105014

Bunyk, P. I., Hoskinson, E. M., Johnson, M. W., Tolkacheva, E., Altomare, F., Berkley, A. J., et al. (2014). Architectural considerations in the design of a superconducting quantum annealing processor. *IEEE Trans. Appl. Superconduct.* 24, 1–10. doi: 10.1109/TASC.2014.2318294

Crawford, D., Levit, A., Ghadermarzy, N., Oberoi, J. S., and Ronagh, P. (2016). Reinforcement learning using quantum Boltzmann machines. *ArXiv e-prints*.

Fazlollahtabar, H., and Saidi-Mehrabad, M. (2016). *Autonomous Guided Vehicles: Methods and Models for Optimal Path Planning*, 1st Edn. Springer International Publishing

Fazlollahtabar, H., Saidi-Mehrabad, M., and Balakrishnan, J. (2015). Mathematical optimization for earliness/tardiness minimization in a multiple automated guided vehicle manufacturing system via integrated heuristic algorithms. *Robot. Auton. Syst.* 72, 131–138. doi: 10.1016/j.robot.2015.05.002

Gurobi Optimization, L. (2018). Gurobi Optimizer Reference Manual.

Harris, R., Johnson, M. W., Lanting, T., Berkley, A. J., Johansson, J., Bunyk, P., et al. (2010). Experimental investigation of an eight-qubit unit cell in a superconducting optimization processor. *Phys. Rev. B* 82:024511. doi: 10.1103/PhysRevB.82.024511

Henderson, M., Novak, J., and Cook, T. (2018). Leveraging adiabatic quantum computation for election forecasting. *J. Phys. Soc. Jpn.* 88:061009. doi: 10.7566/JPSJ.88.061009

Hernandez, M., and Aramon, M. (2017). Enhancing quantum annealing performance for the molecular similarity problem. *Quant. Informat. Process.* 16:133. doi: 10.1007/s11128-017-1586-y

Johnson, M. W., Bunyk, P., Maibaum, F., Tolkacheva, E., Berkley, A. J., Chapple, E. M., et al. (2010). A scalable control system for a superconducting adiabatic quantum optimization processor. *Superconduct. Sci. Technol.* 23:065004. doi: 10.1088/0953-2048/23/6/065004

Kadowaki, T., and Nishimori, H. (1998). Quantum annealing in the transverse Ising model. *Phys. Rev. E* 58, 5355–5363. doi: 10.1103/PhysRevE.58.5355

Kadowaki, T., and Ohzeki, M. (2019). Experimental and theoretical study of thermodynamic effects in a quantum annealer. *J. Phys. Soc. Jpn.* 88:061008. doi: 10.7566/JPSJ.88.061008

Khoshaman, A., Vinci, W., Denis, B., Andriyash, E., and Amin, M. H. (2018). Quantum variational autoencoder. *Quant. Sci. Technol.* 4:014001. doi: 10.1088/2058-9565/aada1f

Li, R. Y., Di Felice, R., Rohs, R., and Lidar, D. A. (2018). Quantum annealing versus classical machine learning applied to a simplified computational biology problem. *npj Quant. Informat.* 4:14. doi: 10.1038/s41534-018-0060-8

Lucas, A. (2014). Ising formulations of many np problems. *Front. Phys.* 2:5. doi: 10.3389/fphy.2014.00005

Morita, S., and Nishimori, H. (2008). Mathematical foundation of quantum annealing. *J. Math. Phys.* 49:125210. doi: 10.1063/1.2995837

Neukart, F., Compostella, G., Seidel, C., von Dollen, D., Yarkoni, S., and Parney, B. (2017). Traffic flow optimization using a quantum annealer. *Front. ICT* 4:29. doi: 10.3389/fict.2017.00029

Neukart, F., Von Dollen, D., Seidel, C., and Compostella, G. (2018). Quantum-enhanced reinforcement learning for finite-episode games with discrete state spaces. *Front. Phys.* 5:71. doi: 10.3389/fphy.2017.00071

Ohzeki, M. (2010). Quantum annealing with the jarzynski equality. *Phys. Rev. Lett.* 105:050401. doi: 10.1103/PhysRevLett.105.050401

Ohzeki, M. (2017). Quantum monte carlo simulation of a particular class of non-stoquastic hamiltonians in quantum annealing. *Sci. Rep.* 7:41186. doi: 10.1038/srep41186

Ohzeki, M. (2019). Message-passing algorithm of quantum annealing with nonstoquastic hamiltonian. *J. Phys. Soc. Jpn.* 88:061005. doi: 10.7566/JPSJ.88.061005

Ohzeki, M., and Nishimori, H. (2011a). Nonequilibrium work performed in quantum annealing. *J. Phys.* 302:012047. doi: 10.1088/1742-6596/302/1/012047

Ohzeki, M., and Nishimori, H. (2011b). Quantum annealing: an introduction and new developments. *J. Comput. Theor. Nanosci.* 8, 963–971. doi: 10.1166/jctn.2011.1776963

Ohzeki, M., Nishimori, H., and Katsuda, H. (2011). Nonequilibrium work on spin glasses in longitudinal and transverse fields. *J. Phys. Soc. Jpn.* 80:084002. doi: 10.1143/JPSJ.80.084002

Ohzeki, M., Okada, S., Terabe, M., and Taguchi, S. (2018a). Optimization of neural networks via finite-value quantum fluctuations. *Sci. Rep.* 8:9950. doi: 10.1038/s41598-018-28212-4

Ohzeki, M., Takahashi, C., Okada, S., Terabe, M., Taguchi, S., and Tanaka, K. (2018b). Quantum annealing: next-generation computation and how to implement it when information is missing. *Nonlin. Theory Its Appl.* 9, 392–405. doi: 10.1587/nolta.9.392

Okada, S., Ohzeki, M., and Tanaka, K. (2019a). The efficient quantum and simulated annealing of Potts models using a half-hot constraint. *arXiv:1904.01522*.

Okada, S., Ohzeki, M., and Tanaka, K. (2019b). Phase diagrams of one-dimensional ising and xy models with fully connected ferromagnetic and anti-ferromagnetic quantum fluctuations. *J. Phys. Soc. Jpn.* 88:024802. doi: 10.7566/JPSJ.88.024802

Okada, S., Ohzeki, M., Terabe, M., and Taguchi, S. (2019c). Improving solutions by embedding larger subproblems in a d-wave quantum annealer. *arXiv:1901.00924*. doi: 10.1038/s41598-018-38388-4

Perdomo-Ortiz, A., Dickson, N., Drew-Brook, M., Rose, G., and Aspuru-Guzik, A. (2012). Finding low-energy conformations of lattice protein models by quantum annealing. *Sci. Rep.* 2:571. doi: 10.1038/srep00571

Rosenberg, G., Haghnegahdar, P., Goddard, P., Carr, P., Wu, K., and de Prado, M. L. (2016). Solving the optimal trading trajectory problem using a quantum annealer. *IEEE J. Select. Top. Sig. Process.* 10, 1053–1060. doi: 10.1109/JSTSP.2016.2574703

Seki, Y., and Nishimori, H. (2012). Quantum annealing with antiferromagnetic fluctuations. *Phys. Rev. E* 85:051112. doi: 10.1103/PhysRevE.85.051112

Seki, Y., and Nishimori, H. (2015). Quantum annealing with antiferromagnetic transverse interactions for the hopfield model. *J. Phys. A Math. Theor.* 48:335301. doi: 10.1088/1751-8113/48/33/335301

Somma, R. D., Nagaj, D., and Kieferová, M. (2012). Quantum speedup by quantum annealing. *Phys. Rev. Lett.* 109:050501. doi: 10.1103/PhysRevLett.109.050501

Suzuki, S., and Okada, M. (2005). Residual energies after slow quantum annealing. *J. Phys. Soc. Jpn.* 74, 1649–1652. doi: 10.1143/JPSJ.74.1649

Takahashi, C., Ohzeki, M., Okada, S., Terabe, M., Taguchi, S., and Tanaka, K. (2018). Statistical-mechanical analysis of compressed sensing for hamiltonian estimation of ising spin glass. *J. Phys. Soc. Jpn.* 87:074001. doi: 10.7566/JPSJ.87.074001

Tsukamoto, S., Takatsu, M., Matsubara, S., and Tamura, H. (2017). An accelerator architecture for combinatorial optimization problems. *FUJITSU Sci. Tech. J.* 53:8.

Ullrich, G. (2014). *Automated Guided Vehicle Systems: A Primer with Practical Applications*. Berlin; Heidelberg: Springer-Verlag.

Venturelli, D., Marchand, D. J. J., and Rojo, G. (2015). Quantum annealing implementation of job-shop scheduling. *ArXiv e-prints*.

Waidyasooriya, H. M., Hariyama, M., Miyama, M. J., and Ohzeki, M. (2019). OpenCL-based design of an FPGA accelerator for quantum annealing simulation. *J. Supercomput.* 75, 5019–5039. doi: 10.1007/s11227-019-02778-w

Conflict of Interest: The authors are representing a collaboration between Tohoku University and DENSO corporations.

frontiers in Physics

ORIGINAL RESEARCH
published: 26 November 2019
doi: 10.3389/fphy.2019.00194

Outlier Mining Methods Based on Graph Structure Analysis

Pablo Amil[1]*, *Nahuel Almeira*[2,3] and *Cristina Masoller*[1]

[1] Department of Physics, Universitat Politécnica de Catalunya, Barcelona, Spain, [2] Facultad de Matemática, Astronomía, Física y Computación, Universidad Nacional de Córdoba, Cordoba, Argentina, [3] Instituto de Física Enrique Gaviola (CONICET), Cordoba, Argentina

Outlier detection in high-dimensional datasets is a fundamental and challenging problem across disciplines that has also practical implications, as removing outliers from the training set improves the performance of machine learning algorithms. While many outlier mining algorithms have been proposed in the literature, they tend to be valid or efficient for specific types of datasets (time series, images, videos, etc.). Here we propose two methods that can be applied to generic datasets, as long as there is a meaningful measure of distance between pairs of elements of the dataset. Both methods start by defining a graph, where the nodes are the elements of the dataset, and the links have associated weights that are the distances between the nodes. Then, the first method assigns an outlier score based on the percolation (i.e., the fragmentation) of the graph. The second method uses the popular IsoMap non-linear dimensionality reduction algorithm, and assigns an outlier score by comparing the geodesic distances with the distances in the reduced space. We test these algorithms on real and synthetic datasets and show that they either outperform, or perform on par with other popular outlier detection methods. A main advantage of the percolation method is that is parameter free and therefore, it does not require any training; on the other hand, the IsoMap method has two integer number parameters, and when they are appropriately selected, the method performs similar to or better than all the other methods tested.

Keywords: outlier mining, anomaly detection, complex networks, machine learning, unsupervised learning, supervised learning, percolation

OPEN ACCESS

Edited by:
Víctor M. Eguíluz,
Institute of Interdisciplinary Physics and Complex Systems (IFISC), Spain

Reviewed by:
Thomas Schlegl,
Medical University of Vienna, Austria
Antonio Scialdone,
Helmholtz Center Munich, Germany
Paul Honeine,
EA4108 Laboratoire d'Informatique, de Traitement de l'Information et des Systèmes (LITIS), France

***Correspondence:**
Pablo Amil
pamil@fisica.edu.uy

Specialty section:
This article was submitted to Biophysics, a section of the journal Frontiers in Physics

Received: *15 July 2019*
Accepted: *06 November 2019*
Published: *26 November 2019*

Citation:
Amil P, Almeira N and Masoller C (2019) Outlier Mining Methods Based on Graph Structure Analysis. Front. Phys. 7:194. doi: 10.3389/fphy.2019.00194

1. INTRODUCTION

When working with large databases, it is common to have entries that may not belong to the database. Sometimes this is because they were mislabeled, or some automatic process failed and introduced artifacts. On the other hand, anomalous items that appear not to belong, may actually be legitimate, just extreme cases of the variability of a large sample. All these elements are usually referred to as outliers [1, 2]. In general, outliers are observations that appear to have been generated by a different process than that of the other (normal) observations.

There are many definitions of what an outlier is, which vary with the system under consideration. For example, rogue waves (or freak waves), which are extremely high waves that might have different generating mechanisms than normal waves [3], have been studied in many fields [4–8], including hydrodynamics and optics. They are usually defined as the extremes in the tail of the distribution of wave heights, however, their precise definition varies, as in hydrodynamics a wave whose height is larger than three times the average can be considered extreme, while in optics, much higher waves compared to the average can be observed [9].

In the field of computer science, a practical definition of outlier elements is that they are those elements that, when they are removed from the training data set, the performance of a machine learning algorithm improves [10]. Outlier mining allows to identify and eliminate mislabeled data [11, 12]. In other situations, the outliers are the interesting points, for example to perform fraud detection [13, 14] or novelty detection [15]. The terms novelty detection, outlier detection and anomaly detection are sometimes used as synonyms in the literature [15, 16].

In spatial objects, the identification of anomalous regions that have distinct features from those of their surrounding regions can reveal valuable information [17–19]. This is the case of biomedical images where particular anomalies characterize the presence of a disease [20, 21]. For example, [22] recently proposed a generative adversarial network for detecting anomalies in OCT retinal images. Another relevant problem consists in anomaly detection in sequences of ordered events, a comprehensive review was provided in Chandola et al. [23], where three main types of formulations of the problem were identified: (i) to determine if a given sequence is anomalous with respect to a database of sequences; (ii) to determine if a particular segment is anomalous within a sequence; and (iii) to determine if the frequency of given event of sequence of events is anomalous with respect to the expected frequency.

With increasing computer power, neural networks are also an attractive option for detecting outliers [24, 25] and anomalies [26]. Hodge and Austin [2] have classified outlier detection methods in three groups: unsupervised (methods that use no prior knowledge of the data), supervised (methods which model both normal and outlier points), and semi-supervised (methods that model only normal points, or only outliers), although the latter can also include a broader spectrum of algorithms (for example a combination of fully unsupervised method and a supervised one). A recent review of outlier definitions and detection methods is presented in Zimek and Filzmoser [27].

We are interested in outlier detection in data that belong to a metric space [28–31]. In this type of dataset, a distance can be defined between items. A relevant example is a wireless sensor network, where localization is based on the distances between nodes and the presence of outliers in data results in localization inaccuracy [32, 33]. Abukhalaf et al. [34] presents a comprehensive survey of outlier detection techniques for localization in wireless sensor networks.

Here we propose two methods that use, as input, only the distances between items in the dataset. Both methods define a graph, or a network, where the nodes are the items of the dataset, and the links have associated weights which are the distances. Then, each method identifies outliers by analyzing the structure of the graph. The first method assigns to each item an outlier score based on the percolation (i.e., the fragmentation) of the graph. The second method uses the IsoMap algorithm [35] (a non-linear dimensionality reduction algorithm that learns the manifold in which the data is embedded in a reduced space), and assigns to each element an outlier score by comparing the geodesic distances with the distances in the reduced space.

Numerous algorithms have been proposed in the literature that use manifold embedding, or more in general, graph embedding, either explicitly or implicitly, to detect anomalies in data [36–41]. A comprehensive review of the literature is out of the scope of the present work, but here we discuss a few relevant examples. Agovic et al. [42, 43] and Wang et al. [44] used the IsoMap algorithm as a preprocessing step, before applying the actual outlier finding algorithm. Our approach differs fundamentally because we take into account how well or how poorly items fit in the manifold, which is disregarded by the cited methods, as they only perform outlier detection in the reduced space.

In Brito et al. [45] the authors use the distance matrix to build a graph where two nodes are connected if each of them is between the k's closest neighbors. For a sufficiently large value of k, the graph will be connected, while, for small values of k, disjoint clusters will appear. If the clusters that appear are large enough, they are considered as classes, while if they are small, they can be interpreted as outliers. In contrast to traditional k-NN algorithms, where the number of neighbors has to be determined a priori, the method proposed by Brito et al. [45] finds the value of k automatically. Nevertheless, the method is not truly parameter-free, as there are two parameters that have to be adjusted which depend on both the dimension and size of the dataset. We speculate that this graph fragmentation method identifies similar outliers as our percolation method, which has the advantage of being parameter free.

We demonstrate the validity of the percolation and IsoMap methods using several datasets, among them, a database of optical coherence tomography (OCT) images of the anterior chamber of the eye. OCT anterior chamber images are routinely used for the early diagnosis of glaucoma. We show that, when images with artifacts (outliers) are removed from the training dataset, the performance of the unsupervised ordering algorithm [46] improves significantly. We also compare the performance of these methods with the performance of other popular methods used in the literature. We show that our results are at worst comparable to those methods.

The paper is organized as follows, in section 2 we describe the proposed methods and also, other popular methods that we use for comparison. In section 3, we describe the datasets analyzed. In section 4 we present the results and in section 5, we summarize our conclusions.

2. METHODS

In this section we describe the two proposed methods, which we refer to as percolation-based method and IsoMap-based method. Both methods require the definition of a distance measure between pairs of elements of the dataset. We also describe three other outlier mining methods, which we used for comparison.

We consider a dataset with N elements and let i and j be two elements, which have associated vectors with m features, $V_i = \{v_1^i \dots v_m^i\}$ and $V_j = \{v_1^j \dots v_m^j\}$. The distance between these elements can be defined as

$$D_{ij} = \left(\sum_k \left| v_k^i - v_k^j \right|^p \right)^{1/p} \quad (1)$$

with p an integer number, taken equal to 2 (Euclidian distance) unless otherwise stated. The selection of an appropriate distance measure is of the utmost importance, since it must capture the similarities and differences of the data. Adding a preprocessing step before calculating the distance matrix may also be necessary to obtain significant distances.

2.1. Percolation-Based Method

The method is described in **Figure 1** (a video is also included in the **Supplementary Information**). We begin by considering a fully connected graph, where the nodes are the elements of the set and where the links are weighted by the distance matrix D_{ij}. Now, we proceed in the following way: we remove the links one by one, from higher to lower weights (i.e., the link representing the highest distance between a pair of elements is removed first). If only a few links are removed, the graph will remain connected, but if one continues, the graph will start to break into different components. As it is well-known from percolation theory [47, 48], it is expected for most of the nodes to remain connected inside a single *giant connected component* (GCC), and for the rest of them to distribute into many small components. If we remove enough links, even the giant component disappears. This transition between the existence and non-existence of a giant component is known as a percolation transition, and is one of the most studied problems of statistical physics [49, 50]. Here, we are interested in the percolated state, i.e., when such a giant component exists. In particular, the nodes that do not belong to the GCC are candidates for being considered as outliers, as they are relatively distant to the rest of the graph.

Following this idea, we can label each node with an outlier score (OS), defined as the weight of the link that, after being removed, separates the node from the GCC. Thus, the first elements to leave the GCC are the ones with the highest OS, while the last ones have the lowest OS.

For this method to correctly identify the outliers, we assume that normal points occupy more densely populated zones than outliers, thus having (normal points) local neighborhoods connected with small distances while outliers are connected to normal points via longer distances. Such outliers will become disconnected from the giant connected component sooner than the normal ones in the described procedure.

It is worth noting that the computation of the GCC can be performed efficiently using a variation of the union-find algorithm [51], thus making this method suitable for large datasets.

2.2. IsoMap-Based Method

The basic idea of this method is to use the well-known algorithm IsoMap [35] to perform dimensionality reduction on the raw data, and to analyze the manifold structure in the reduced space, assigning to each point an outlier score that measures how well it fits in the manifold.

The method consists of the following steps

- We apply IsoMap to the distance matrix D_{ij} (computed from the raw features) and obtain two matrices: 1) a new set of features for each element of the database, $V^i = \{v'^i_1 \dots v'^i_r\}$ with $i = 1 \dots N$ and 2) a matrix of graph distances, D^G_{ij} in the geodesic space as described in Tenenbaum et al. [35].
- Using the new set of features, we calculate a new distance matrix $\tilde{D}_{ij}$, using the Euclidean distance (Equation 1 with $p = 2$).
- The third step is to compare $\tilde{D}_{ij}$ with D^G_{ij}: for each element i we compute the similarity, ρ_i, between vectors $(D^G_{i1}, \dots D^G_{iN})$ and $(\tilde{D}_{i1}, \dots \tilde{D}_{iN})$, using the Pearson correlation coefficient.
- The final step is to define the outlier score as $OS_i = 1 - {\rho_i}^2$. For "normal" elements, we expect high similarity, while for abnormal ones, we expect low similarity.

With this method, the assumption is that normal points lie in a low dimensional manifold embedded in the full-dimensional space, and outliers lie outside such manifold. If the parameters of the IsoMap are such that the low dimensional manifold structure is recovered successfully, the distances between points in the new set of features ($\tilde{D}_{ij}$), the geodesic distances in the manifold, and the graph distances (D^G_{ij} an approximation of the geodesic distance) should all be similar for normal points lying on the manifold. However, for outliers the geodesic distance is not defined and thus, the graph distances and the distances in the new set of features will disagree. When we compute the similarity, ρ_i, assessing this disagreement, normal points will have a high value ρ_i (near 1) and outliers a low value of ρ_i, therefore the outlier score should be high for outliers and low for normal points.

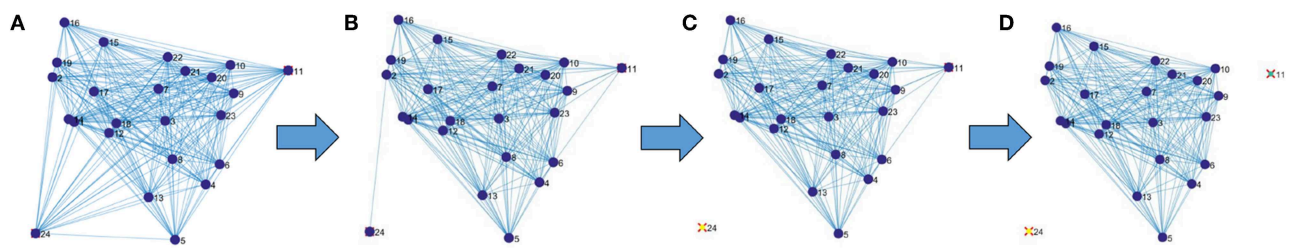


FIGURE 1 | Example of application of the percolation method. Starting with a fully connected graph **(A)** links are removed according to their distances (longer distances are removed first). In **(B)** some of the longest connections have been eliminated, but the graph remains fully connected, in **(C)** the first outlier is identified as the first element that becomes disconnected from the giant connected component, and in **(D)** two elements are disconnected. A video illustrating the procedure is presented in the **Supplementary Information**.

The parameters of this method, are the parameters of the IsoMap algorithm, namely, the dimensionality of the objective space (d) and neighborhood size (number of neighbors, k) to construct the graph. In this work, the parameters of the IsoMap were optimized (when a training set was available) by maximizing the average precision doing an extensive search in the parameter space.

2.3. Other Methods

We compared the performance of both methods with:

- The simplest way to define an outlier score: the distance to the center-of-mass (d2CM) in the original feature space, $V_i = \{v_1^i \dots v_m^i\}$. For "normal" elements, we expect short distance, while for abnormal ones, we expect high distance.
- A popular distance-based method, which will be referred to as Ramaswamy et al. [29]. This method is based on the distance of a point from its kth nearest neighbor, in the raw (original) high-dimensional feature space. The method assigns an outlier score to each point equal to its distance to its kth nearest neighbor.
- And a very popular method, One Class Support Vector Machine (OCSVM) which uses the inner product between the elements in the database to estimate a function that is positive in a subset of the input space where elements are likely to be found, and negative otherwise [52].

2.4. Implementation

All the methods were implemented and run in MatLab. The IsoMap method was build modifying the IsoMap algorithm implementation by Van Der Maaten et al. [53], the percolation method was implemented using graph objects in MatLab. With a simple database of 1,000 elements with 30 dimensions, the percolation method takes around 6 s to run and the IsoMap method takes around 18 s, while One Class Support Vector Machine takes around 0.2 s to run, Ramaswamy about 0.04 s to run, and distance to center of mass 0.01 s to run on an Intel i7-7700HQ laptop. Both methods could significantly improve their runtime by optimizing the code and translating it into a compiled language.

3. DATA

We tested the above described methods in several databases. In the main text we present three examples: a database of anterior chamber Optical Coherent Tomography (OCT) images, a database of face images with added artifacts, and a database of credit card transactions. Additional synthetic examples are presented in the **Supplementary Information**.

3.1. Anterior Chamber OCT Images

This database consists of 1213 OCT images of the anterior chamber of the eye of healthy and non-healthy patients of the *Instituto de Microcirugia Ocular* in Barcelona. The database was analyzed in Amil et al. [46] where an unsupervised algorithm for ordering the images was proposed. The images had been classified in four categories (closed, narrow, open, and wide open) by two expert ophthalmologists. By using manually extracted features, and the features returned by the unsupervised algorithm, a similar separation in the four classes was found. Here we will demonstrate that the similarity is further improved when images containing artifacts (outliers) are removed from the dataset given to the unsupervised algorithm.

Examples taken from the database are shown in **Figure 2**.

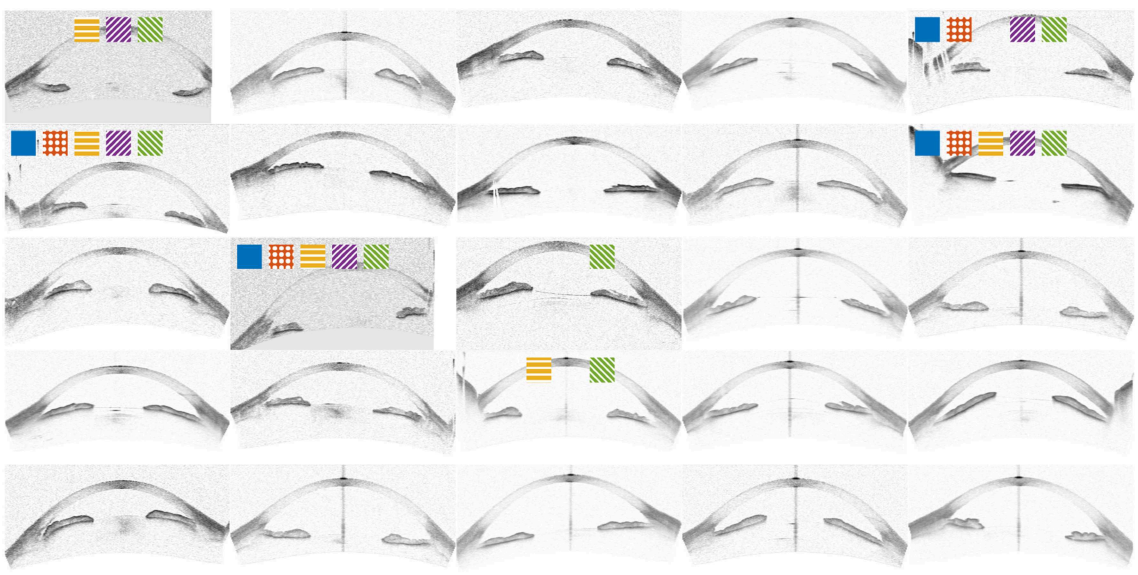


FIGURE 2 | Example images from the OCT database, all except the first one were randomly sampled. Marked images correspond to top 15% outlier score for OCSVM (Blue), distance to center of mass (Orange), IsoMap (Yellow), Percolation (Purple), and Ramaswamy (Green). The first image corresponds to the marked improvement in **Figure 4**.

The distance matrix D_{ij} was calculated as described in detail in Amil et al. [46]: by comparing pixel-by-pixel, after pre-processing the images to adjust the alignment and to enhance the contrast. For the algorithms that don't use the distance matrix (OCSVM and distance to center of mass), the same pre-processing was used.

3.2. Face Database

This publicly available database [54], kindly provided by AT&T Laboratories Cambridge, is constituted by face images (photographs of 40 subjects with 10 different images per subject) with outliers that were added similarly to Ju et al. [55]: first we rescaled the images to 64 by 64 pixels, and then, we added a square of noise to one randomly selected image per subject. Examples are shown in **Figure 3**. When using the parameters proposed in Ju et al. [55] to generate the artifacts, all the methods have a perfect performance (average precision = 1), so we generated the artifacts in the following manner: We used only square artifacts whose size we varied from 0 (no artifact added) to 64 (the whole image), the square was placed randomly in the image and its content was gray-scale pixels whose gray-scale value was randomly sampled such that the distribution was the same as the gray-scale value distribution of the combination of all the images in the database. We also generated a database with outliers whose brightness was modify by simply multiplying all the image by a constant factor.

For this database (and also for the databases analyzed in the **Supplementary Information**, which also have added outliers), we generated two independent sets for each square size: one was used to find, in the case of the IsoMap and Ramaswamy methods, the optimal parameters, and the second one was used for testing.

For this database, the distance matrix was calculated as the Euclidean pixel-by-pixel distance.

3.3. Credit Card Transactions

This publicly available database [56–61] contains credit card transactions made in September 2013 by European cardholders. It contains 284807 transactions made in 2 days, of which 492 correspond to frauds. In order to preserve confidentiality, for each transaction the data set only includes the amount of money in the transaction, a relative time, and 28 features that are the output of a principal component analysis (PCA) of all the other metadata related to the transaction. In our analysis we divided the total dataset into 8 sets of about 4,000 entries (due to computational constrains) according to the amount of the transaction and computed the distance as the euclidean distance using these 28 features.

4. RESULTS

4.1. Anterior Chamber OCT Images

For the OCT database, there is no a priori definition of outliers (i.e., no ground truth), all the images were drawn from the same database. However, as a proxy for determining the performance of the outlier finding methods, we used the performance of the unsupervised methods proposed in Amil et al. [46] when ignoring the images identified as outliers.

As removing outliers should improve the performance of machine learning algorithms, we performed two tests: first, we recalculated the correlation metrics presented in Amil et al. ([46], Table 1), removing the first n outliers that were identified by each method. Second, to test the significance of the improved performance, we repeated the calculation, now removing random images. The results presented in **Figure 4** confirm that removing the detected outliers improves the performance, while removing random images has no significant effect. We also see that IsoMap is the method that produces the highest improvement, while d2CM and OCSVM have low-significance performance improvement. For the IsoMap method we set the parameters to $d = 10$ and $k = 15$, while for the Ramaswamy method we used $k = 6$.

4.2. Face Database

For this database, as explained in section 3.2, we generated artifacts artificially and tried to find the images presenting artifacts as outliers. We varied the size of the artifact generated to evaluate the robustness of the methods. For each size, we generated two different databases with artifacts (with the same parameters but different random seeds), we used the first one to optimize the parameters of IsoMap and Ramaswamy algorithms, and the second one to test the algorithms. We show the results of evaluating the performance on the second database for each square size in **Figure 5A**, we used the average precision based on the precision-recall curve as performance measure, this measure

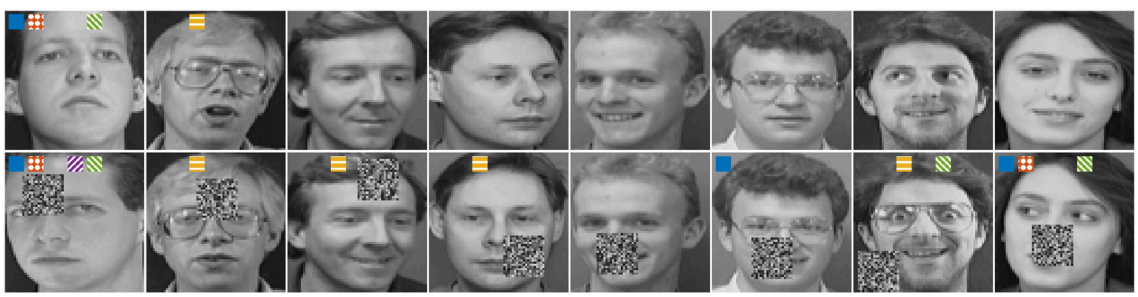


FIGURE 3 | Example images from the face database. Eight original images at the top, and eight images with added artifacts at the bottom. Marked images correspond to top 10% outlier score for OCSVM (Blue), distance to center of mass (Orange), IsoMap (Yellow), Percolation (Purple), and Ramaswamy (Green).

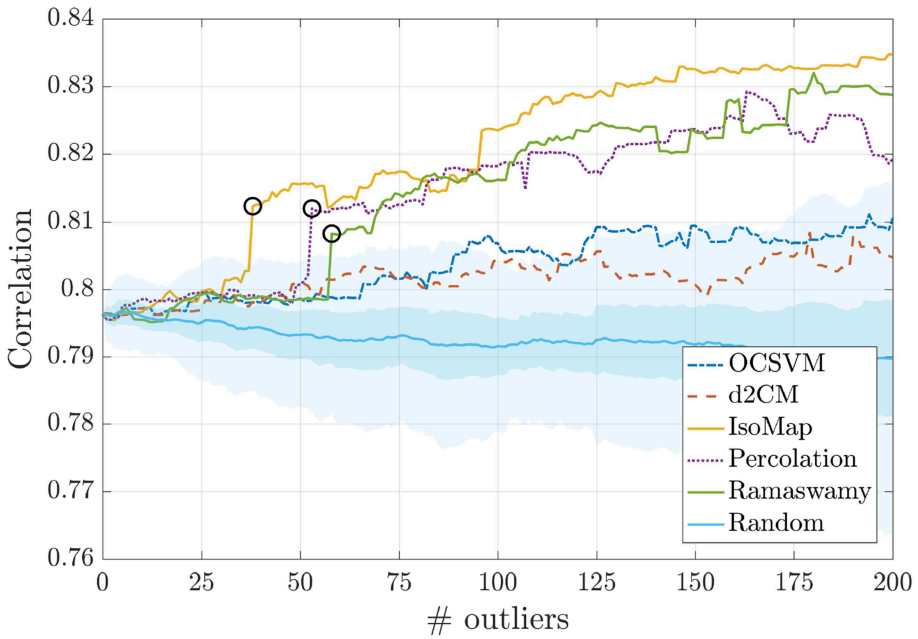


FIGURE 4 | Performance of the OCT image ordering algorithm as a function of the number of outliers that are removed from the database. As expected, we see that the performance, which is measured by the correlation coefficient between the feature returned by the ordering (unsupervised) algorithm and the feature provided by manual expert annotation (mean angle), improves as the outliers detected are removed. The different lines indicate the method of outlier identification and the colored region indicates results when the images removed are randomly selected, one standard deviation is shown in dark coloring, while three standard deviations is shown in light coloring. In this case, as expected, no significant change in the performance is seen. For some methods a sharp improvement is observed when eliminating one specific image (marked with a black circle), this image corresponds to the first one shown in **Figure 2**.

computed as the area under the precision-recall curve [62] is more appropriate than other more commonly used metrics for class imbalance scenarios. In **Figure 5A** we see that Ramaswamy tends to slightly outperform all other methods, in particular, the percolation-based method shifts from being the worst method (when the squares are small) to the second best (when the squares are large). In **Figure 5D** we show the performance of the IsoMap method as a function of its parameters, we depict two zones with better performance, one with fairly low dimensionality and a low number of neighbors (more neighbors translate to a more linear mapping), and another zone with greater dimensionality and almost the maximum possible number of neighbors. In general, performance is very sensitive to parameter variations. In **Figure 5C** we show how altering the brightness of some images can also be perceived as outliers due to the distance measure used (Euclidean pixel-by-pixel).

Also, to evaluate how robust the methods are when changing the distance measure, we varied p in the Minkowski distance family (Equation 1), and evaluated the methods for the parameters optimized for $p = 2$ (Euclidean), $p = 1$ and $p = 10$, the average precision as a function of p for the distance-based methods is shown in **Figure 5B**. As we can see, for $p > 4$ Ramaswamy and Percolation-based perform similarly well, also, the parameters of Ramaswamy are very robust when changing p in the training set (the Ramaswamy method was also train with $p = 1$ and $p = 10$ obtaining the same parameters as for $p = 2$), while IsoMap is very sensitive to such changes.

We generated a different dataset whose outliers were images that, instead of having added noise, were multiplied by a constant (brightness) factor. We varied the brightness from 0 (the image being all black) to 3. The results of this study is shown in **Figure 5C**.

4.3. Credit Card Transactions

In this database the ground truth (the fraud credit card transactions) is known and thus, the performance of the different methods is, as in the prior example, quantified with the average precision based on the precision-recall curve.

The database was divided into several subsets according to the amount of money of each transaction (see **Figure 6**), each set (of around 4,000 transactions) was further randomly divided into two sets in order to use one for training and the other one for testing. The results are summarized in **Figure 6** that displays the average precision for all testing sets. We can see that the performance of the methods is very heterogeneous.

To try to understand the origin of the large variability, we conducted an additional experiment in which we considered groups of 3,900 normal transactions chosen at random (without considering the amount of the transaction) and 100 frauds also chosen at random, which were divided equally in training and test subsets. We repeat this experiment 8 times with different random seeds, and the results are presented in **Figure 7** in this experiment the average precision of the methods was increased due to a larger fraction of frauds in the test sets.

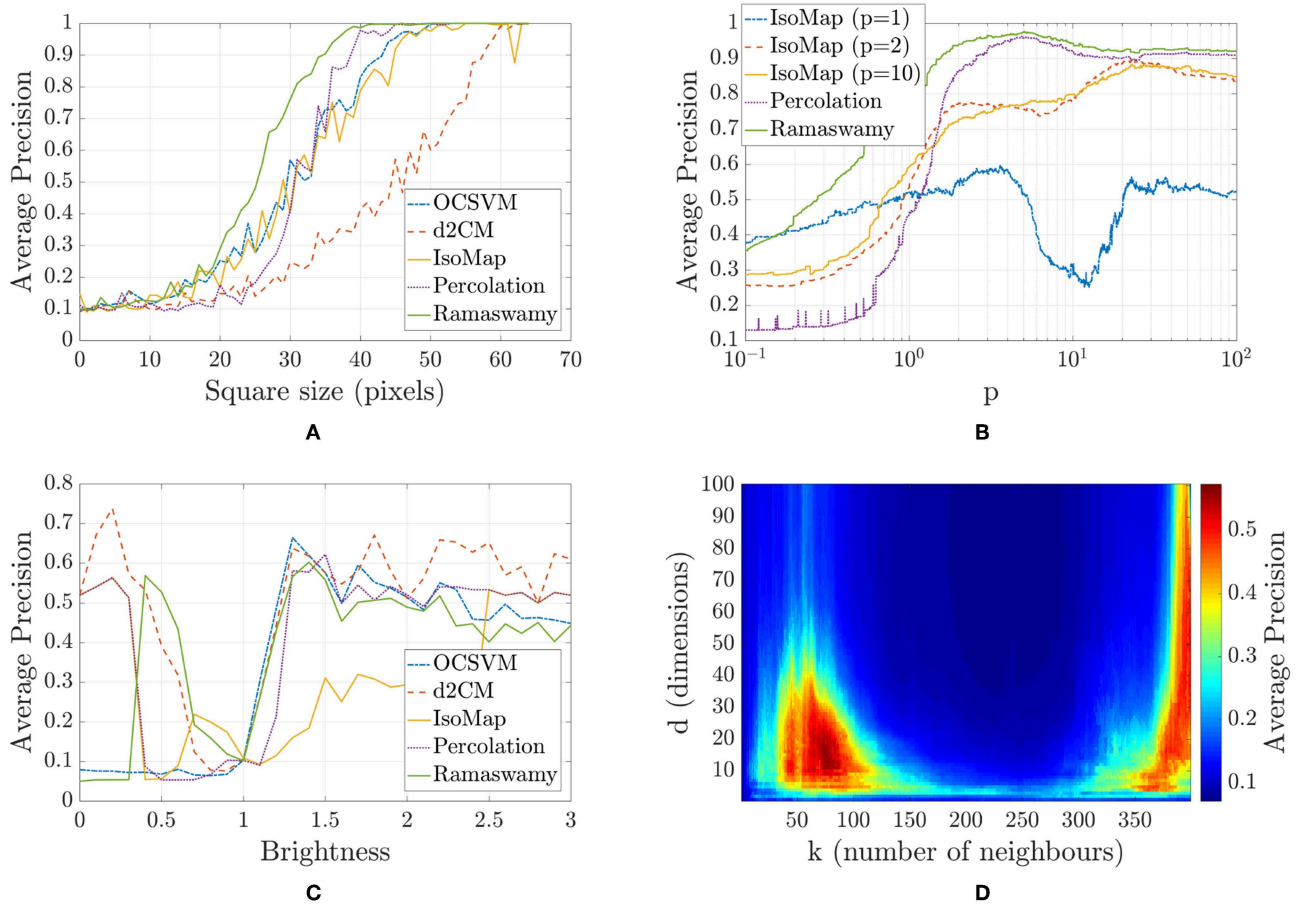


FIGURE 5 | Analysis of the face database. **(A)** Average precision as a function of the square size for the different outlier finding methods, One Class Support Vector Machine (OCSVM) in blue, distance to center of mass (d2CM) in orange, IsoMap in yellow, Percolation-based method in purple, and Ramaswamy in green. The Average precision was calculated on databases independently generated from those used to set the parameters of the methods. **(B)** Average precision as a function of the distance measure for different outlier finding methods with a square size of 36 pixels, IsoMap (trained with $p = 1$) in blue, IsoMap (trained with $p = 2$) in orange, IsoMap (trained with $p = 10$) in yellow, Percolation-based method in purple, and Ramaswamy in green. The Average precision was calculated on databases independently generated from those used to set the parameters of the methods. **(C)** Average precision as a function of the brightness multiplier of the outliers. **(D)** Average precision in the training set as a function of the IsoMap parameters with a square size of 30 pixels.

4.4. Discussion

Figure 8 presents the comparison of the results obtained with the five methods used, for the three databases analyzed. **Figure 8A** summarizes the results for the OCT database, with the boxplot we can see the minimum, first quartile, median, third quartile, and maximum of the correlation coefficient when varying the amount of outliers considered (corresponds to **Figure 4**). **Figure 8B** summarizes, in a similar manner, the results for the face database showing the boxplot of the average precision values when varying the square size (corresponds to **Figure 5A**). **Figure 8C** summarizes the results for the credit card transactions showing the boxplot of the average precision values when changing the amount range (corresponds to **Figure 6**). As we can see in **Figure 8**, the IsoMap and Percolation methods perform well in the three databases; their performance being either better than or comparable to the performance of the other three methods. Additional examples presented in the **Supplementary Information** confirm the good performance of IsoMap and Percolation methods.

Figure 5B shows how the performance of distance-based methods is affected by the definition of the distance. We can see that the performance of all the methods depends on the definition of the distance. The methods are also sensitive to changes in the preprocessing of the data, therefore, well-prepared data with a meaningful distance definition is needed for optimizing the performance of all methods.

It is important to consider how the two methods proposed here scale with the dimension of the data, d (i.e., the number of features of each sample), and the number of samples, N, in the database. Since both methods begin by calculating the distance matrix, the processing time is at least of the order dN^2 because the calculation of the distance between pairs of elements linearly

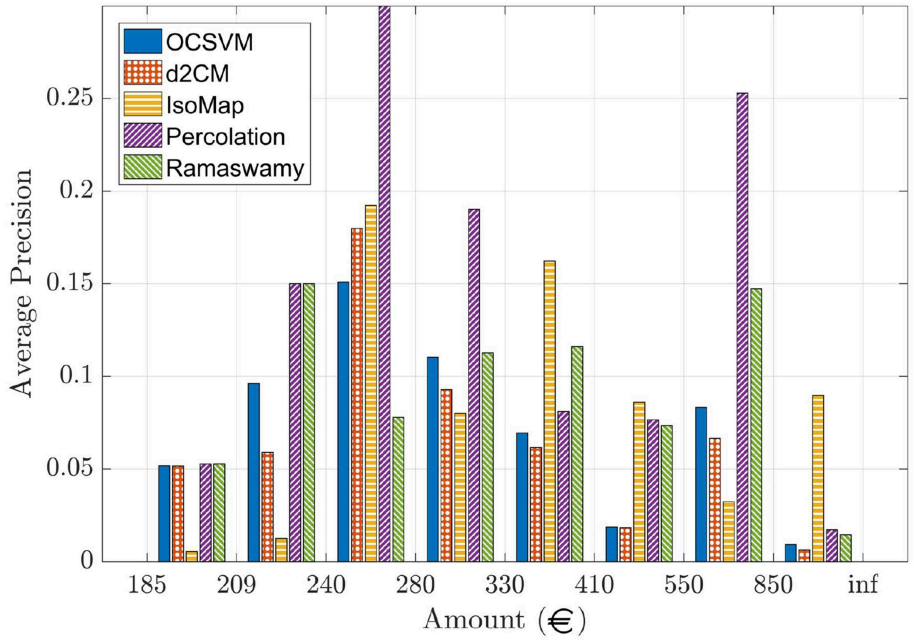


FIGURE 6 | Performance of all the outlier finding methods for the credit card transactions on the test subsets for each amount range.

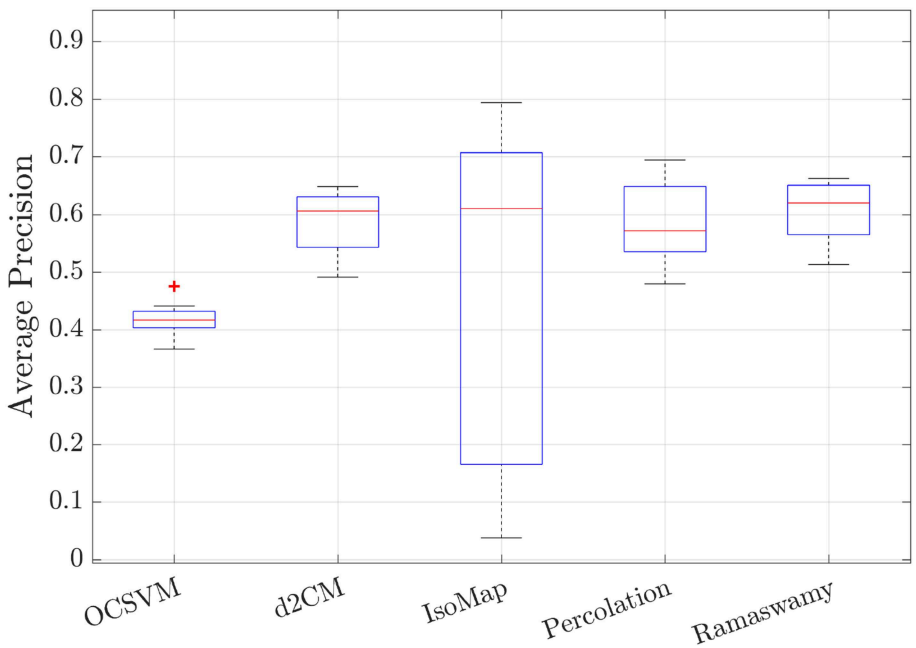


FIGURE 7 | Performance of all the outlier finding methods for the credit card transactions on the test subsets of the random groups. The random groups were generated by randomly choosing 3,900 normal transactions and 100 frauds, and it was further randomly divided into two subsets, a training and a testing subsets.

increases with d and quadratically with N. Both methods need to store in memory the distance matrix and analize it, this imposes memory requirements that can limit their applicability for large datasets. In the case of IsoMap, this analysis is of order N^2. In the case of the percolation method, a threshold needs to be gradually varied in order to precisely identify the order in which the elements became disconnected from the giant component. This results in a runtime of the order of N^2 using the algorithm

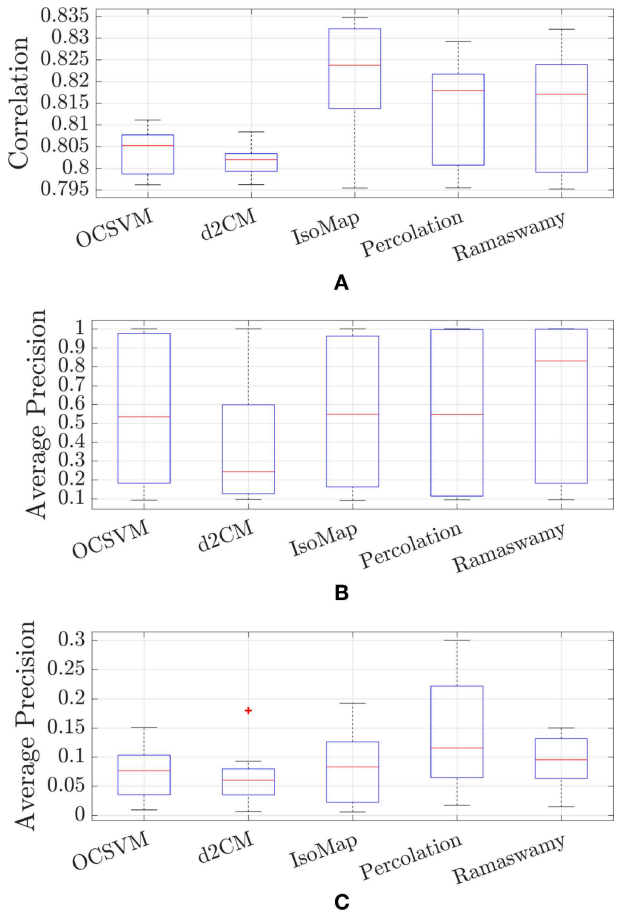


FIGURE 8 | Box-plot summarizing the results of all the outlier finding methods in all the databases. **(A)** Anterior Chamber OCT images, **(B)** Face database (only testing groups), **(C)** Credit Card transactions (only testing groups).

proposed in Newman and Ziff [51]. Regarding the dimensionality of the data, because both methods only need to hold in memory the N^2 distance matrix (and not the dN features of the N samples) they are suitable for very high dimensional data (where $d \gg N$) because once the distances of an element to all other elements have been computed, the d features of that element will not be needed again.

5. CONCLUSIONS

We have proposed two methods for outlier mining that rely on the definition of a meaningful measure of distance between pairs of elements in the dataset, one being fully unsupervised without the need of setting any parameters, and other which has 2 integer number parameters that can be set using a labeled training set. Both methods define a graph (whose nodes are the elements of the dataset, connected by links whose weights are the distances between the nodes) and analyze the structure of the graph. The first method is based on the percolation of the graph, while the second method uses the IsoMap non-linear dimensionality reduction algorithm. We have tested the methods on several real and synthetic datasets (additional examples are presented in the **Supplementary Information**), and compared the performance of the proposed algorithms with the performance of a "naive" method (that calculates the distance to the center of mass) and two popular outlier finding methods, Ramaswamy and One Class Support Vector Machine (OCSVM).

Although the percolation algorithm performs comparably to (or slightly lower than) other methods, it has the great advantage of being parameter-free. In contrast, the IsoMap method has two parameters (natural numbers) that have to be selected appropriately. The performance of the methods varies with the dataset analyzed because the underlying assumption of what an outlier is, is different for the different methods. The percolation method assumes that the normal elements will be in one large cluster, with outliers being far from that cluster; IsoMap assumes that the normal elements lie on a manifold, and that outliers lie outside such manifold; the Ramaswamy and OCSVM methods assume that the outliers lie in a less densely populated sector of the space, while the "naive" method simply assumes that outliers are the furthest elements from the center of mass. These assumptions do not always hold, which results in the identification of normal elements as outliers. For example, in the OCT database there were some duplicated entries which were assigned by the Ramaswamy method the least outlier score, in spite of having a minor artifact.

The percolation algorithm is immune to duplicate entries, as it assigns the same outlier score as if there was only one element. On the other hand, the effect of duplicate entries on the IsoMap and "naive" methods is more difficult to asses, but is to be expected that if the duplicated elements are only few, they won't have a large effect in the manifold learned, or in the center of mass calculated.

The execution time of both methods scales at least as dN^2 where d is the number of features of each item and N is the number of items in the database (as dN^2 is the time needed to compute the distance matrix). Therefore, the methods are suitable for the analysis of small to medium-size databases composed of high-dimensional items.

DATA AVAILABILITY STATEMENT

Some of the datasets analyzed in this manuscript are not publicly available. Requests to access such datasets should be directed to pamil@fisica.edu.uy.

AUTHOR CONTRIBUTIONS

PA, NA, and CM designed the algorithms and wrote the manuscript. PA and NA implemented them. PA ran the test on the proposed databases.

ACKNOWLEDGMENTS

PA and CM acknowledge support by the BE-OPTICAL project (EU H2020-675512). CM also acknowledges support from the

Spanish Ministerio de Ciencia, Innovación y Universidades (PGC2018-099443-B-I00) and ICREA ACADEMIA (Generalitat de Catalunya). NA and CM acknowledge the hospitality of the International Centre for Theoretical Physics-South American Institute for Fundamental Research (ICTP-SAIFR) where a collaboration was established and this work started.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fphy.2019.00194/full#supplementary-material

Video S1 | Video illustrating the whole percolation process. It starts with a fully connected graph where links are eliminated one by one according to the distance of the nodes in the original space.

REFERENCES

1. Grubbs FE. Procedures for detecting outlying observations in samples. *Technometrics*. (1969) **11**:1–21. doi: 10.1080/00401706.1969.10490657
2. Hodge V, Austin J. A survey of outlier detection methodologies. *Artif Intell Rev*. (2004) **22**:85–126. doi: 10.1023/B:AIRE.0000045502.10941.a9
3. Onorato M, Residori S, Bortolozzo U, Montina A, Arecchi F. Rogue waves and their generating mechanisms in different physical contexts. *Phys Rep*. (2013) **528**:47–89. doi: 10.1016/j.physrep.2013.03.001
4. Solli D, Ropers C, Koonath P, Jalali B. Optical rogue waves. *Nature*. (2007) **450**:1054–7. doi: 10.1038/nature06402
5. Zhen-Ya Y. Financial rogue waves. *Commun Theor Phys*. (2010) **54**:947. doi: 10.1088/0253-6102/54/5/31
6. Shats M, Punzmann H, Xia H. Capillary rogue waves. *Phys Rev Lett*. (2010) **104**:104503. doi: 10.1103/PhysRevLett.104.104503
7. Katz RW, Parlange MB, Naveau P. Statistics of extremes in hydrology. *Adv Water Resour*. (2002) **25**:1287–304. doi: 10.1016/S0309-1708(02)00056-8
8. Chabchoub A, Hoffmann N, Akhmediev N. Rogue wave observation in a water wave tank. *Phys Rev Lett*. (2011) **106**:204502. doi: 10.1103/PhysRevLett.106.204502
9. Akhmediev N, Kibler B, Baronio F, Belić M, Zhong WP, Zhang Y, et al. Roadmap on optical rogue waves and extreme events. *J Opt*. (2016) **18**:063001. doi: 10.1088/2040-8978/18/6/063001
10. Liu H, Shah S, Jiang W. On-line outlier detection and data cleaning. *Comput Chem Eng*. (2004) **28**:1635–47. doi: 10.1016/j.compchemeng.2004.01.009
11. Brodley CE, Friedl MA. Identifying and eliminating mislabeled training instances. In: *Proceedings of the 13th National Conference on Artificial Intelligence*. Portland, OR: AAAI Press (1996). p. 799–805.
12. Brodley CE, Friedl MA. Identifying mislabeled training data. *J Artif Intell Res*. (1999) **11**:131–67. doi: 10.1613/jair.606
13. Aleskerov E, Freisleben B, Rao B. Cardwatch: a neural network based database mining system for credit card fraud detection. In: *Proceedings of the IEEE/IAFE 1997 Computational Intelligence for Financial Engineering (CIFEr)*. IEEE (1997). p. 220–6.
14. Cheng Q, Varshney PK, Michels JH, Belcastro CM. Fault detection in dynamic systems via decision fusion. *IEEE Trans Aerospace Electron Syst*. (2008) **44**:227–42. doi: 10.1109/TAES.2008.4517001
15. Pimentel MA, Clifton DA, Clifton L, Tarassenko L. A review of novelty detection. *Signal Process*. (2014) **99**:215–49. doi: 10.1016/j.sigpro.2013.12.026
16. Agrawal S, Agrawal J. Survey on anomaly detection using data mining techniques. *Proc Comput Sci*. (2015) **60**:708–13. doi: 10.1016/j.procs.2015.08.220
17. Kou Y, Lu CT, Chen D. Spatial weighted outlier detection. In: *Proceedings of the 2006 SIAM International Conference on Data Mining*. SIAM (2006). p. 614–8.
18. Lu CT, Chen D, Kou Y. Detecting spatial outliers with multiple attributes. In: *Proceedings 15th IEEE International Conference on Tools with Artificial Intelligence*. Sacramento, CA: IEEE (2003). p. 122–8.
19. Sun P, Chawla S. On local spatial outliers. In: *Fourth IEEE International Conference on Data Mining (ICDM' 04)*. IEEE (2004). p. 209–16.
20. Spence C, Parra L, Sajda P. Detection, synthesis and compression in mammographic image analysis with a hierarchical image probability model. In: *Proceedings IEEE Workshop on Mathematical Methods in Biomedical Image Analysis (MMBIA 2001)*. IEEE (2001). p. 3–10.
21. Taoum A, Mourad-Chehade F, Amoud H. Early-warning of ARDS using novelty detection and data fusion. *Comput Biol Med*. (2018) **102**:191–9. doi: 10.1016/j.compbiomed.2018.09.030
22. Schlegl T, Seeböck P, Waldstein SM, Langs G, Schmidt-Erfurth U. f-AnoGAN: fast unsupervised anomaly detection with generative adversarial networks. *Med Image Anal*. (2019) **54**:30–44. doi: 10.1016/j.media.2019.01.010
23. Chandola V, Banerjee A, Kumar V. Anomaly detection for discrete sequences: a survey. *IEEE Transn Knowl Data Eng*. (2010) **24**:823–39. doi: 10.1109/TKDE.2010.235
24. Hawkins S, He H, Williams G, Baxter R. Outlier detection using replicator neural networks. In: *International Conference on Data Warehousing and Knowledge Discovery*. Aix-en-Provence: Springer (2002). p. 170–80.
25. Chen J, Sathe S, Aggarwal C, Turaga D. Outlier detection with autoencoder ensembles. In: *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM (2017). p. 90–8.
26. Sabokrou M, Fayyaz M, Fathy M, Moayed Z, Klette R. Deep-anomaly: fully convolutional neural network for fast anomaly detection in crowded scenes. *Comput Vision Image Understand*. (2018) **172**:88–97. doi: 10.1016/j.cviu.2018.02.006
27. Zimek A, Filzmoser P. There and back again: outlier detection between statistical reasoning and data mining algorithms. *Wiley Interdiscipl Rev Data Min Knowl Discov*. (2018) **8**:e1280. doi: 10.1002/widm.1280
28. Knox EM, Ng RT. Algorithms for mining distancebased outliers in large datasets. In: *Proceedings of the International Conference on Very Large Data Bases*. Citeseer (1998). p. 392–403.
29. Ramaswamy S, Rastogi R, Shim K. Efficient algorithms for mining outliers from large data sets. In: *ACM Sigmod Record*. Vol. 29. ACM (2000). p. 427–38.
30. Angiulli F, Pizzuti C. Outlier mining in large high-dimensional data sets. *IEEE Trans Knowl Data Eng*. (2005) **17**:203–15. doi: 10.1109/TKDE.2005.31
31. Angiulli F, Fassetti F. Dolphin: an efficient algorithm for mining distance-based outliers in very large datasets. *ACM Trans Knowl Discov. Data*. (2009) **3**:4. doi: 10.1145/1497577.1497581
32. Yang Z, Wu C, Chen T, Zhao Y, Gong W, Liu Y. Detecting outlier measurements based on graph rigidity for wireless sensor network localization. *IEEE Trans Vehicul Technol*. (2012) **62**:374–83. doi: 10.1109/tvt.2012.2220790
33. Abukhalaf H, Wang J, Zhang S. Mobile-assisted anchor outlier detection for localization in wireless sensor networks. *Int J Future Gen Commun Netw*. (2016) **9**: 63–76. doi: 10.14257/ijfgcn.2016.9.7.07
34. Abukhalaf H, Wang J, Zhang S. Outlier detection techniques for localization in wireless sensor networks: a survey. *Int J Future Gen Commun Netw*. (2015) **8**:99–114. doi: 10.14257/ijfgcn.2015.8.6.10
35. Tenenbaum JB, De Silva V, Langford JC. A global geometric framework for nonlinear dimensionality reduction. *Science*. (2000) **290**:2319–23. doi: 10.1126/science.290.5500.2319
36. Pang Y, Yuan Y. Outlier-resisting graph embedding. *Neurocomputing*. (2010) **73**:968–74. doi: 10.1016/j.neucom.2009.08.020
37. Schubert E, Gertz M. Intrinsic t-stochastic neighbor embedding for visualization and outlier detection. In: *International Conference on Similarity Search and Applications*. Springer (2017). p. 188–203.
38. Madabhushi A, Shi J, Rosen M, Tomaszeweski JE, Feldman MD. Graph embedding to improve supervised classification and novel class detection: application to prostate cancer. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Berlin, Heidelberg: Springer (2005). p. 729–37.
39. Cook DJ, Holder LB. Graph-based data mining. *IEEE Intell Syst Appl*. (2000) **15**:32–41. doi: 10.1109/5254.850825
40. Eberle W, Holder L. Anomaly detection in data represented as graphs. *Intell Data Anal*. (2007) **11**:663–89. doi: 10.3233/IDA-2007-11606

41. Rahmani A, Afra S, Zarour O, Addam O, Koochakzadeh N, Kianmehr K, et al. Graph-based approach for outlier detection in sequential data and its application on stock market and weather data. *Knowl Based Syst.* (2014) **61**:89–97. doi: 10.1016/j.knosys.2014.02.008
42. Agovic A, Banerjee A, Ganguly AR, Protopopescu V. Anomaly detection in transportation corridors using manifold embedding. In: *Knowledge Discovery from Sensor Data*. (2008). p. 81–105. Available online at: https://www.researchgate.net/profile/Auroop_Ganguly/publication/220571551_Anomaly_detection_using_manifold_embedding_and_its_applications_in_transportation_corridors/links/5400c3590cf2c48563aee68e/Anomaly-detection-using-manifold-embedding-and-its-applications-in-transportation-corridors.pdf
43. Agovic A, Banerjee A, Ganguly A, Protopopescu V. Anomaly detection using manifold embedding and its applications in transportation corridors. *Intell Data Anal.* (2009) **13**:435–55. doi: 10.3233/IDA-2009-0375
44. Wang L, Li Z, Sun J. Improved ISOMAP algorithm for anomaly detection in hyperspectral images. In: *Fourth International Conference on Machine Vision (ICMV 2011): Machine Vision, Image Processing, and Pattern Analysis*. Vol. 8349. International Society for Optics and Photonics (2012). p. 834902.
45. Brito M, Chavez E, Quiroz A, Yukich J. Connectivity of the mutual k-nearest-neighbor graph in clustering and outlier detection. *Stat Probab Lett.* (1997) **35**:33–42.
46. Amil P, González L, Arrondo E, Salinas C, Guell JL, Masoller C, et al. Unsupervised feature extraction of anterior chamber OCT images for ordering and classification. *Sci Rep*. (2019) 9:1157. doi: 10.1038/s41598-018-38136-8
47. Barrat A, Barthélemy M, Vespignani A. *Dynamical Processes on Complex Networks*. Cambridge University Press (2008).
48. Cohen R, Havlin S. *Complex Networks: Structure, Robustness and Function*. Cambridge University Press (2010).
49. Stauffer D. *Introduction to Percolation Theory: Revised Second Edition*. Taylor & Francis (1994).
50. Callaway DS, Newman MEJ, Strogatz SH, Watts DJ. Network robustness and fragility: percolation on random graphs. *Phys Rev Lett*. (2000) **85**:5468–71. doi: 10.1103/physrevlett.85.5468
51. Newman MEJ, Ziff RM. Fast Monte Carlo algorithm for site or bond percolation. *Phys Rev E*. (2001) **64**:016706. doi: 10.1103/physreve.64.016706
52. Schölkopf B, Platt JC, Shawe-Taylor J, Smola AJ, Williamson RC. Estimating the support of a high-dimensional distribution. *Neural Comput*. (2001) **13**:1443–71. doi: 10.1162/089976601750264965
53. Van Der Maaten L, Postma E, Van den Herik J. Dimensionality reduction: a comparative. *J Mach Learn Res*. (2009) **10**:13. Available online at: http://www.math.chalmers.se/Stat/Grundutb/GU/MSA220/S18/DimRed2.pdf
54. Samaria FS, Harter AC. Parameterisation of a stochastic model for human face identification. In: *Proceedings of 1994 IEEE Workshop on Applications of Computer Vision*. IEEE (1994). p. 138–42.
55. Ju F, Sun Y, Gao J, Hu Y, Yin B. Image outlier detection and feature extraction via L1-Norm-Based 2D probabilistic PCA. *IEEE Trans Image Process*. (2015) **24**:4834–46. doi: 10.1109/TIP.2015.2469136
56. Dal Pozzolo A, Caelen O, Johnson RA, Bontempi G. Calibrating probability with undersampling for unbalanced classification. In: *2015 IEEE Symposium Series on Computational Intelligence*. IEEE (2015). p. 159–66.
57. Dal Pozzolo A, Caelen O, Le Borgne YA, Waterschoot S, Bontempi G. Learned lessons in credit card fraud detection from a practitioner perspective. *Expert Syst Appl*. (2014) **41**:4915–28. doi: 10.1016/j.eswa.2014.02.026
58. Dal Pozzolo A, Boracchi G, Caelen O, Alippi C, Bontempi G. Credit card fraud detection: a realistic modeling and a novel learning strategy. *IEEE Trans Neural Netw Learn Syst*. (2018) **29**:3784–97. doi: 10.1109/TNNLS.2017.2736643
59. Dal Pozzolo A. *Adaptive Machine Learning for Credit Card Fraud Detection*. (2015). Available online at: https://pdfs.semanticscholar.org/bcfb/f068dff507b9ef11240e69f96d24f5d89fc1.pdf.
60. Carcillo F, Dal Pozzolo A, Le Borgne YA, Caelen O, Mazzer Y, Bontempi G. Scarff: a scalable framework for streaming credit card fraud detection with spark. *Inform Fusion*. (2018) **41**:182–94. doi: 10.1016/j.inffus.2017.09.005
61. Carcillo F, Le Borgne YA, Caelen O, Bontempi G. Streaming active learning strategies for real-life credit card fraud detection: assessment and visualization. *Int J Data Sci Anal*. (2018) **5**:285–300. doi: 10.1007/s41060-018-0116-z
62. Saito T, Rehmsmeier M. The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PLoS ONE*. (2015) **10**:e0118432. doi: 10.1371/journal.pone.0118432

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Physics

ORIGINAL RESEARCH
published: 12 December 2019
doi: 10.3389/fphy.2019.00210

Tackling the Trade-Off Between Information Processing Capacity and Rate in Delay-Based Reservoir Computers

Silvia Ortín[1]* *and Luis Pesquera*[2]

[1] Unidad Cientifica de Innovación empresarial, Instituto de Neurociencias, CSIC-UMH, Sant Joan d'Alacant, Spain,
[2] Departamento de Estructura de la Materia, Instituto de Física de Cantabria, CSIC-UC, Santander, Spain

We study the role of the system response time in the computational capacity of delay-based reservoir computers. Photonic hardware implementation of these systems offers high processing speed. However, delay-based reservoir computers have a trade-off between computational capacity and processing speed due to the non-zero response time of the non-linear node. The reservoir state is obtained from the sampled output of the non-linear node. We show that the computational capacity is degraded when the sampling output rate is higher than the inverse of the system response time. We find that the computational capacity depends not only on the sampling output rate but also on the misalignment between the delay time of the non-linear node and the data injection time. We show that the capacity degradation due to the high sampling output rate can be reduced when the delay time is greater than the data injection time. We find that this mismatch gives an improvement of the performance of delay-based reservoir computers for several benchmarking tasks. Our results show that the processing speed of delay-based reservoir computers can be increased while keeping a good computational capacity by using a mismatch between delay and data injection times. It is also shown that computational capacity for high sampling output rates can be further increased by using an extra feedback line and delay times greater than the data injection time.

Keywords: reservoir computing, delayed-feedback systems, memory capacity, system response time, information processing rate

OPEN ACCESS

Edited by:
Víctor M. Eguíluz,
Institute of Interdisciplinary Physics and Complex Systems (IFISC), Spain

Reviewed by:
Alexander Vladimirovich Bogdanov,
Saint Petersburg State University, Russia
Ignazio Licata,
Institute for Scientific Methodology (ISEM), Italy
Guy Verschaffelt,
Vrije University Brussel, Belgium
Apostolos Argyris,
Institute of Interdisciplinary Physics and Complex Systems (IFISC), Spain

***Correspondence:**
Silvia Ortín
silortin@gmail.com

Specialty section:
This article was submitted to Interdisciplinary Physics, a section of the journal Frontiers in Physics

Received: *20 August 2019*
Accepted: *21 November 2019*
Published: *12 December 2019*

Citation:
Ortín S and Pesquera L (2019) Tackling the Trade-Off Between Information Processing Capacity and Rate in Delay-Based Reservoir Computers. Front. Phys. 7:210. doi: 10.3389/fphy.2019.00210

1. INTRODUCTION

Reservoir computing (RC) is a successful brain-inspired concept to process information with temporal dependencies [1, 2]. RC conceptually belongs to the field of recurrent neural networks (RNN) [3]. In these systems, the input signal is non-linearly projected onto a high-dimensional state space where the task can be solved much more easily than in the original input space. The high-dimensional space is typically a network of interconnected non-linear nodes (called neurons). The ensemble of neurons is called the reservoir. RC implementations are generally composed of three layers: input, reservoir, and output (see **Figure 1**). The input layer feeds the input signal to the reservoir via fixed weighted connections. The input weights are often chosen randomly. These weights determine how strongly each of the inputs couples to each of the neurons. In traditional

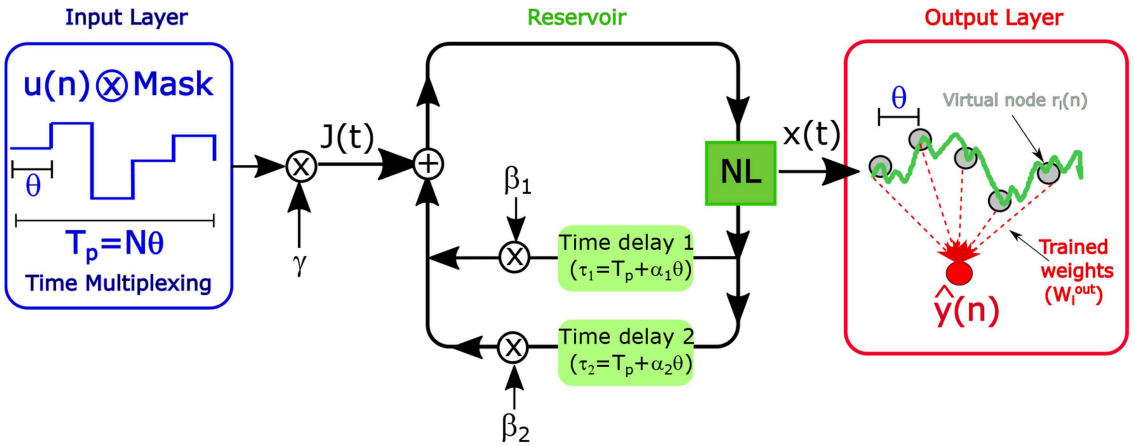


FIGURE 1 | Schematic illustration of delay-based RC. NL stands for Non-linear Node. The NL can have one ($\beta_2 = 0$) or two delay lines. The points $r_i(n)$ represent the virtual nodes separated by time intervals θ. The masked input $u(n+1) \otimes$ Mask is injected directly following $u(n) \otimes$Mask.

RNN the connections among the neurons are optimized to solve the task. Nevertheless, in RC, the coupling weights in the reservoir are not trained and can be chosen at random. The reservoir state is given by the combined states of all the individual nodes. Under the influence of input signals, the nodes of the reservoir remain in a transient state such that each input is injected in the presence of the response to the previous input. As a result the reservoir can retain input data for a finite amount of time (short-term memory [4]), and it can compute linear and non-linear functions of the retained information. The reservoir output is constructed through a linear combination of neural responses, with readout weights that are trained for the specific task. These weights are typically obtained by a simple linear regression. The strength of the reservoir computing scheme lies in the simplicity of its training method, where only the connections with the output are optimized.

Hardware implementations of RC are sought because they offer high processing speed [5], parallelism, and low power consumption [6] compared to digital implementations. However, traditional RC involves a large number of interconnected non-linear neurons, so the hardware implementation is very challenging. Recently, it has been shown that RC can be efficiently implemented using a single non-linear dynamical system (neuron) subject to delayed feedback (delay-based RC) [7]. This architecture emulates the dynamic complexity traditionally achieved by a network of neurons. In delay-based RC, the spatial multiplexing of the input in standard RC systems with N neurons is replaced by time-multiplexing (see **Figure 1**). The reservoir is composed of N sampled outputs of the non-linear node distributed along the delay line, called virtual nodes. Connections between these N virtual nodes are established through the delayed feedback when a mismatch between the delay and data injection times is introduced [8]. Delay-based RC has facilitated hardware implementation in photonic systems that have the potential to develop high-speed information processing. An overview of recent advances is given in Van der Sande et al. [9]. However, the information processing rate is limited by the non-zero response time of the system. The reservoir state is obtained from the sampled output of the non-linear node. The information processing (or data injection) time is given by $T_p = N\theta$, where θ is the inverse of the output sampling rate, i.e., the time interval between two virtual nodes (see **Figure 1**). The information processing rate ${T_p}^{-1}$ can be increased by decreasing the node distance (higher sampling output rate). However, when θ is less than the response time of the system T, virtual nodes are connected through the non-linear node dynamics. Network connections due to inertia lead to virtual node-states with similar dependence on inputs. Then the number of independent virtual nodes decreases and the diversity of the reservoir states is reduced. As a consequence computational capacity is degraded. Then there is a trade-off between information processing capacity and rate in delay-based reservoir computers.

In this work we show, using numerical simulations, that the computational capacity is degraded when the sampling output rate is higher than the inverse of the system response time. We obtain the memory capacities for different values of θ/T and the mismatch between the delay and data injection times. Until now only two different delay-based reservoir architectures have been considered: $\theta < T$ without mismatch [7] and $\theta \gg T$ with mismatch time θ [8]. We find that the computational capacity depends not only on the sampling output rate but also on the misalignment between the delay time of the non-linear node and the data injection time. We show that the capacity degradation due to high sampling output rate can be reduced when the delay time is greater than the data injection time. We also find that this mismatch gives an improvement of the performance of delay-based reservoir computers for several benchmarking tasks. Then, delay-based reservoir computers can achieve a high processing speed and good computational capacity using a mismatch between delay and data injection times.

We first consider a simple architecture of a single non-linear node with one feedback delay line. The linear and non-linear

information processing capacities are obtained for different values of θ/T. It is found that information processing capacity is boosted for small values of θ/T if the delay of the non-linear node τ is greater than T_p. A similar performance is obtained for small and large values of θ/T for channel equalization and also for NARMA-10 task if values of the delay time greater than T_p are used. Then the information processing rate is increased without causing system performance degradation. This is due to the increase in reservoir diversity. Another strategy to increase reservoir diversity is to use an extra feedback line. We show that memory capacity can be further increased with this architecture for small values of θ/T when the delay time is greater than the information processing time.

2. MATERIALS AND METHODS

2.1. Delay-Based Reservoir Computers

Traditional RC implementations consist of a large number N of randomly interconnected non-linear nodes [3]. The state of the reservoir at time step n, $\mathbf{r}(n)$, is determined by:

$$\mathbf{r}(n) = f(\gamma \mathbf{W}^{in}\mathbf{u}(n) + \beta \mathbf{W}\mathbf{r}(n-1)), \tag{1}$$

where $\mathbf{u}(n)$ is sequentially injected input data and f is the reservoir activation function. The matrices $\mathbf{W}$ and $\mathbf{W}^{in}$ contain the (generally random) reservoir and input connection weights, respectively. The matrix $\mathbf{W}$ ($\mathbf{W}^{in}$) is rescaled with a connection (input) scaling factor β (γ). The exact internal connectivity is not crucial. In fact, it has been shown that simple non-random connection topologies (e.g., a simple chain or ring) gives a good performance [10].

Delay-based RC is a minimal approach to information processing based on the emulation of a recurrent network via a single non-linear dynamical node subject to delayed feedback. The reservoir nodes (called virtual nodes) are the sampled outputs of the non-linear node distributed along the delay line (see **Figure 1**). In the time delay-based approach there is only one real non-linear node. Thus, the spatial multiplexing of the input in standard RC is replaced here by time multiplexing. The advantage of delay-based RC lies in the minimal hardware requirements. There is a price to pay for this hardware simplification: compared to an N-node standard spatially-distributed reservoir, the dynamical behaviour in the system has to run at an N-times higher speed in order to have equal input-throughput.

The dynamics of a delay-based reservoir has been described as [7, 11–16]:

$$T\dot{x}(t) = -x(t) + f\left(\beta x(t-\tau) + \gamma J(t)\right), \tag{2}$$

where T is the response time of the system, τ the delay time, $\beta > 0$ the feedback strength and γ the input scaling. The masked input $J(t)$ is the continuous version of the discrete random mapping of the original input $\mathbf{W}^{in}\mathbf{u}(n)$. In our approach, every time interval of the data injection/processing time T_p represents another discrete time step. This time is given by $T_p = N\theta$, where θ is the temporal separation between virtual nodes. Individual virtual nodes are addressed by time-multiplexing the input signal. An input mask is used to emulate the input weights of traditional RC. This mask function is a piecewise constant function, constant over an interval of θ, and periodic with period T_p. The N mask values m_i are drawn from a random uniform distribution in the interval [–1,1] The procedure to construct the continuous data $J(t)$ is the following. First, the input stream $u(n)$ undergoes a sample and hold operation to define a stream which is constant during one T_p, before it is updated. Every segment of length T_p is multiplied by the mask (see **Figure 1**). The masked input $u(n+1) \otimes$ Mask is injected directly following $u(n) \otimes$Mask. After a time T_p, each virtual node is updated.

The reservoir state that corresponds to the input $u(n)$, $\mathbf{r}(n) = [r_1(n) \dots r_N(n)]$, is the collection of N outputs of the dynamical system, $r_i(n) = x(nT_p - (N-i)\theta)$, where $i = 1, \dots, N$ (see **Figure 1**). These N points are called virtual nodes because they correspond to taps in the delay line and play the same role as the neurons in standard RC. The node responses $r_i(n)$ are used to train the reservoir to perform a specific task. As in the standard RC [1, 17], only the output weights $\mathbf{W}^{out}$ are computed to obtain the output $\hat{y} = \mathbf{W}^{out}\mathbf{r}$. A linear regression method is used to minimize the error between the output $\hat{y}$ and the desired target y in the training phase. The testing is then performed using previously unseen input data of the same kind as those used for training.

2.1.1. Interconnection Structure of Delay-Based Reservoir Computers

In delay-based reservoir computers virtual nodes are connected through the feedback loop with nodes affected by previous inputs. Virtual node states also depend on close (in time) nodes through the inherent dynamics of the non-linear node. We can identify four time scales in the delayed feedback system with external input described by Equation (2): the response time T of the non-linear node, the delay time τ, the separation of the virtual nodes θ, and the data injection/processing time T_p. Setting the values of the different time scales creates a fixed interconnection structure. The virtual nodes can set up a network structure via the feedback loop by introducing a mismatch between T_p and τ. Interconnection between virtual nodes due to the inherent dynamics of the non-linear node is obtained if the node separation θ is smaller than the response time of the system T. Due to inertia the response of the system is not instantaneous. Therefore, the state of a virtual node depends on the states of nodes that correspond to previous taps in the delay line. However, if θ is too short, the non-linear node will not be able to follow the changes in the input signal and the response signal will be too small to measure. Typically, a number of $\theta = 0.2T$ is quoted [7, 11–16, 18].

When $\theta \gg T$ the state of a given virtual node is independent of the states of the neighboring virtual nodes. Then virtual nodes are not coupled through the non-linear node dynamics. The reservoir state is only determined by the instantaneous value of the input $J(t)$ and the delayed reservoir state. The system given by Equation (2) can then be described with a map:

$$x(t) = f\left(\beta x(t-\tau) + \gamma J(t)\right). \tag{3}$$

A network structure can be obtained via the feedback loop by introducing a mismatch between T_p and τ. This mismatch can be quantified in terms of the number of virtual nodes by $\alpha = (\tau - N\theta)/\theta$. In the case of $0 \leq \alpha < N$ and $\theta \gg T$, the virtual node states are given by:

$$r_i(n) = \begin{cases} f(\beta r_{i-\alpha}(n-1) + \gamma m_i u(n)) & \text{if } \alpha < i \leq N \\ f(\beta r_{N+i-\alpha}(n-2) + \gamma m_i u(n)) & \text{if } i \leq \alpha \end{cases}$$

The network topology depends on the value of α. When $\alpha = 1$ (i.e., $\tau = T_p + \theta$) the topology is equivalent to the ring topology in standard RC systems [10]. When $\alpha < 0$, a number $|\alpha|$ of virtual nodes are not connected through the feedback line with nodes at a previous time. When α and N have no common divisors, all virtual nodes are connected through feedback in a single ring. However, when N and α are not coprimes, subnetworks are formed with a similar dependence on inputs and the reservoir diversity is reduced.

Although the two types of virtual node connections are not exclusive, only two cases have been considered until now: delay-based reservoirs connected through system dynamics ($\alpha = 0$ and $\theta < T$) [7, 12–18], or by the feedback line ($\theta \gg T$) [8, 15, 19].

It is clear that the information processing rate of delay-based reservoir computers ${T_p}^{-1}$ depends on the node separation. Then reservoir computers with nodes connected only through the feedback line ($\theta \gg T$) are slower than a counterpart exploiting the virtual connections through the system dynamics ($\theta < T$). However, as we will show in 3.1, information processing capacity is degraded when $\theta < T$. In this case, the computational capacity increases with the mismatch between the delay and data injection times (see section 3.1).

2.2. Computational Capacity

Delay-based reservoir computers can reconstruct functions of h previous inputs $\mathbf{y_k}(n) = y(u(n-k_1), \ldots, u(n-k_h))$ from the state of a dynamical system using a linear estimator $\hat{\mathbf{y}}_\mathbf{k}$. Here $\mathbf{k}$ denotes the vector $(k_1, \ldots, k_h)$. The estimator $\hat{\mathbf{y}}_\mathbf{k}$ is obtained from N internal variables (node states) of the system. The suitability of a reservoir to reconstruct $\mathbf{y_k}$ can be quantified by using the capacity [20]:

$$C\left[\mathbf{y_k}\right] = \left(1 - \frac{\sum_n (\hat{\mathbf{y}}_\mathbf{k}(n) - \mathbf{y_k}(n))^2}{\sum_n (\mathbf{y_k}(n))^2}\right). \quad (4)$$

The capacity is $C\left[\mathbf{y_k}\right] = 1$ when the reconstruction error for $\mathbf{y_k}$ is zero. The capacity for reconstructing a function of the inputs $\mathbf{y}$, $C\left[\mathbf{y}\right]$, is given by the sum of $C\left[\mathbf{y_k}\right]$ over all sequences of past inputs [20]:

$$C\left[\mathbf{y}\right] = \sum_\mathbf{k} C\left[\mathbf{y_k}\right]. \quad (5)$$

The total computational capacity C_T is the sum of $C\left[\mathbf{y_k}\right]$ over all sequences of past inputs and a complete orthonormal set of functions. When $\mathbf{y_k}$ is a linear function of one of the past inputs, $\mathbf{y_k}(n) = u(n-k)$, the capacity $C\left[\mathbf{y}\right]$ corresponds to the linear memory capacity introduced in Jaeger [4]. The capacity of the system to compute non-linear functions of the retained information is given by the non-linear memory capacity [20]. The computational capacity is given by the sum of the linear and non-linear memory capacities. The total capacity is limited by the dimension of the reservoir. As a consequence, there is a trade-off between linear and non-linear memory capacities [20].

The total computational capacity of delay-based reservoirs is given by the number of linearly independent virtual nodes. The computational power of delay-based reservoir computers is therefore hidden in the diversity of the reservoir states. In the presence of inertia ($\theta < T$) non-linear node dynamics couples close (in time) virtual nodes. This coupling reduces reservoir diversity, and then computational capacity is degraded. The computational capacity of delay-based reservoir depends not only on the separation between the virtual nodes but also on the misalignment between T_p and τ, given by α. When $\alpha < 0$, the state of a virtual node of index $i > (N - |\alpha|)$, $r_i(n)$, is a function of the virtual node state $r_{i-N+|\alpha|}(n)$ at the same time. Then the reservoir diversity and computational capacity are reduced. Computational capacity is also reduced if $|\alpha|$ and N are not coprimes. In this case, the N virtual nodes form $gcd(|\alpha|, N)$ ring subnetworks, where gcd is the greatest common divisor. Each subnetwork has $p = N/gcd(|\alpha|, N)$ virtual nodes. Virtual node-states belonging to different subnetworks have a similar dependence on inputs and reservoir diversity is reduced.

2.3. Reservoir Computers With Two Delay Lines

An architecture with several delay lines has been proposed [21, 22] to increase the memory capacity of delay-based reservoir computers with virtual nodes connected only through non-linear system dynamics ($\theta < T$ and $\alpha = 0$). Several delay lines are added to preserve older information. The longer the delay, the older the response that is being fed back. Even without explicitly reading the older states from the delay line, the information is re-injected into the system and its memory can be extended. We apply this approach to delay-based reservoir computers with virtual nodes that are connected through non-linear node dynamics and by the feedback line.

The dynamics of reservoir computers with two delay lines is described by:

$$T\dot{x}(t) = -x(t) + f\left(\beta_1 x(t-\tau_1) + \beta_2 x(t-\tau_2) + \gamma J(t)\right), \quad (6)$$

where $\beta_i \geq 0$ is the feedback strength of the delay line i. The total feedback strength is $\beta = \beta_1 + \beta_2$. The corresponding delays are given by $\tau_1 = N\theta + \alpha_1$ and $\tau_2 = 2N\theta + \alpha_2$, where $0 \leq \alpha_i < N\theta$. The reservoir state is the same as in one delay-based RC, i.e., the virtual nodes correspond to taps only in the shorter (τ_1) delay line. In the case of $\alpha_1 = 0$, it has been shown [23] that the best performance for NARMA-10 task is obtained when τ_1 and τ_2 are coprimes. In this case, the number of virtual nodes that are mixed together within the history of each virtual node is maximized.

If the mismatches α_i ($i = 1, 2$) are zero, the virtual node states at time n depend on the reservoir state at time $(n-1)$ and $(n-2)$ via the delay line 1 and 2, respectively. In one-delay reservoirs ($\beta_2 = 0$), the number of virtual nodes whose state at time n depends on the reservoir state at time $(n-2)$ increases with

the mismatch (see Equation 2.1.1 for the case without inertia). When a second delay is added with a mismatch $\alpha_2 > 0$, some virtual nodes at time n are connected with nodes at time $(n-3)$. The number of virtual nodes with states at time n that depend on the reservoir state at time $(n-3)$ increases with α_2. These connections with older states can extend the memory of the two-delay reservoir computer.

3. RESULTS

In this section, we show the numerical results obtained for the memory capacities and performance of a non-linear delay-based RC system. We study a delay-based reservoir computer with a single non-linear node for the one and two delay lines architectures. The one-delay system is governed by Equation (2) and the two-delay reservoir by Equation (6). In both cases the reservoir activation function f is given by:

$$f(z) = f_s \frac{1 - \exp(-\lambda z)}{a + \exp(-\lambda z)}, \tag{7}$$

where $a = 2$ and $\lambda = 1$. The value of $f_s = 2.5$ is chosen to have, when $\beta < 1$, a stable fixed point for the system defined by Equation (2) in absence of input ($\gamma = 0$). This non-linear function is asymmetric to allow that the reservoir computer reconstructs even functions of the input. Similar results are obtained for different reservoir activation functions, in particular for a sin^2 function, that corresponds to an optoelectronic implementation [8, 11, 13–15].

The number of virtual nodes used in the numerical simulations is a prime number, $N = 97$, to avoid the capacity degradation due to the formation of subnetworks. The rest of fixed parameters are: $T = 1$ and $\beta = \beta_1 = 0.8$ for the one-delay reservoir computer and $\beta_1 + \beta_2 = \beta = 0.8$ for the two-delay reservoir computer. The effective non-linearity of the delay-based reservoir computer can be changed with the scaling input parameter γ. In this work, we consider $\gamma = 0.1$ and $\gamma = 1$ that correspond to low-to-moderate and strong non-linearity, respectively. The total capacity of a linear reservoir computer with $f(z) = z$ will also be analyzed.

All the results presented in this paper are the average over 5 simulation runs with different training/test sets and different masks. A total of 8,000 inputs (6,000 for training and 2,000 for testing) are used for computational capacities and the NARMA-10 task. The dataset for the channel equalization task has 10,000 points for training and 6,000 for testing.

3.1. Computational Capacity

To analyze the computational capacity of the non-linear delay-based reservoir computer, we calculate by using (Equations 4 and 5) four capacities as in Duport et al. [19], namely linear (LMC), quadratic (QMC), cubic (CMC) and cross (XMC) memory capacities, which correspond to functions $\mathbf{y}$ given by the first, second and third order Legendre polynomials, respectively. In order to obtain these capacities a series of i.i.d. input samples drawn uniformly from the interval [–1, 1] is injected into the reservoir. The LMC is obtained by summing over k the capacity $C\left[\mathbf{y_k}\right]$ for reconstructing $\mathbf{y}_k(n) = u(n-k)$. It corresponds to the linear memory capacity introduced in Jaeger [4]. The QMC and CMC are obtained by summing over k the capacity for $\mathbf{y}_k(n) = (3u^2(n-k) - 1)/2$ and $\mathbf{y}_k(n) = (5u^3(n-k) - 3u(n-k))/2$, respectively. The XMC is obtained by summing over k, k' for $k < k'$ the capacities for the product of two inputs, $\mathbf{y}_{k,k'} = u(n-k) \cdot u(n-k')$. In non-linear systems, the sum $C_s = LMC + QMC + CMC + XMC$ does not include all possible contributions to C_T, so $C_s \leq C_T$, whereas for linear systems $C_s = LMC = C_T$. Finally, note that in some cases the main contribution to the LMC is due to the sum of $C\left[\mathbf{y}_k\right]$ over a large range of values of k greater than a certain value k_c with large normalized-root-mean-square reconstruction errors $\mathrm{NRMSRE}(k) = \sqrt{1 - C\left[\mathbf{y}_k\right]}$. This corresponds to a memory function $m(k) = C\left[\mathbf{y}_k\right]$ with a long tail. In these cases a high LMC can be obtained but the reconstruction error for $\mathbf{y}_k$ when $k > k_c$ is large. This low quality memory capacity leads to poor performance for tasks requiring long memory, such as NARMA-10 task [10]. A memory capacity with good quality (quality memory capacity) can be calculated by summing only the capacities for $\mathbf{y}_k$ over k until they drop below a certain value q. If we consider that the error is small when $\mathrm{NRMSRE}(k) < 0.3$, this corresponds to $C\left[\mathbf{y}_k\right] > 0.91$. Then we consider a value $q = 0.9$ to obtain the quality memory capacity $C\left[\mathbf{y}\right]^{q=0.9}$.

3.1.1. Memory Capacities of One-Delay Reservoir Computers

First, we simulate a delay-based reservoir computer with a single delay line. We focus on the influence of the system response time on the computational capacity for different values of the mismatch α between the data injection and delay times. Until now two values of the mismatch have been used: $\alpha = 0$ with $\theta = 0.2T$ [7, 12–18], and $\alpha = 1$ with $\theta \gg T$ [8, 15, 19].

We first consider a linear system with $f(z) = z$ in Equation (2). As stated before, the total computational capacity of this system can be obtained from the linear memory capacity, e.g., $C_T^l = LMC$. **Figure 2** shows the total computational capacity of the linear reservoir computer as a function of the node separation for two different values of the detuning between T_p and τ: $\alpha = 0$ and $\alpha = 1$. For $\alpha = 1$ (**Figure 2B**), C_T^l increases with θ/T and the upper bound $C_T = N = 97$ is almost reached for $\theta/T = 10$. Similar behaviour is obtained for detuning values $1 < \alpha < N$. Then almost all the nodes are linearly independent for $\theta/T = 10$ and non-zero α. The quality memory $C_T^{l(q=0.9)} = LMC^{q=0.9}$ of the linear delay-based reservoir computer also increases with θ/T following the same behavior than C_T^l for $\alpha = 1$. However, when $\theta < T$ a total capacity $C_T^l < 50$ is obtained. Then a clear degradation of the capacity is observed with respect to its upper bound, given by $N = 97$, when the node separation is smaller than the response time of the non-linear node dynamics. In this case virtual nodes with an index difference smaller than T have similar states. Then reservoir diversity is reduced and the information processing capacity is degraded. When θ/T increases the coupling between close (in time) virtual nodes decreases, and the capacity increases.

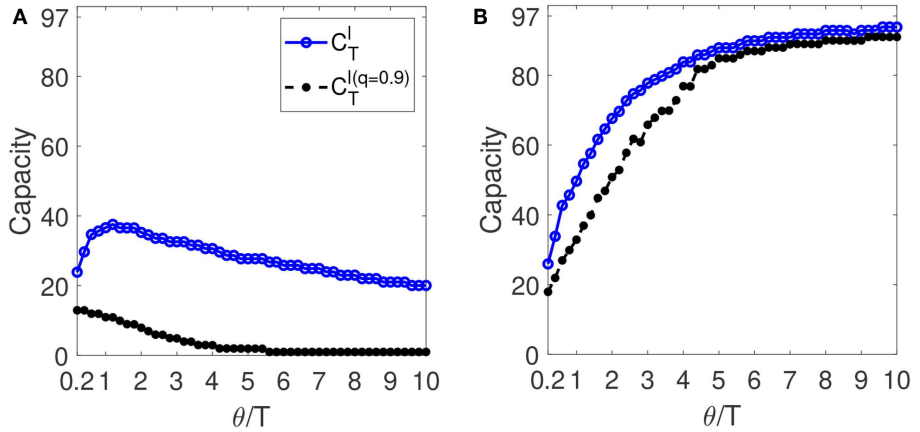


FIGURE 2 | Computational capacity of the linear delay-based RC with one delay line as a function of θ/T for **(A)** $\alpha = 0$ and **(B)** $\alpha = 1$. The solid line with blue circles is the total computational capacity (C_T^I) and the dashed line with black points is the total quality computational capacity calculated for $q = 0.9$.

In the special case of zero detuning ($\alpha = 0$), the only coupling between the virtual nodes is through the system dynamics with non-zero response time. For $\alpha = 0$, the total capacity of the linear delay-based reservoir computer has a maximum value $C_T^I = 38$ at $\theta/T \sim 1.2$ (see **Figure 2A**). In this case a clear degradation of the capacity is observed for any value of θ/T. The maximum is due to the trade-off between the fading of the coupling through the system dynamics for low sampling output rates and the very similar responses to different inputs for small θ. Furthermore, for $\alpha = 0$, the quality memory capacity decreases with θ/T and the maximum $C_T^{I(q=0.9)}$ is obtained at $\theta/T = 0.2$. For low inertia, $\theta/T = 4$, we obtain a normalized-root-mean-square reconstruction error NRMSRE(k) > 0.6 when $k > 2$. For $\theta/T = 1$ a NRMSRE(k) > 0.3 when $k > 12$ is obtained.

We consider now a non-linear delay-based reservoir computer with an activation function given by Equation (7) and a low-to-moderate non-linearity ($\gamma = 0.1$). In this case, the capacity C_s has a behaviour as a function of θ similar to that of the total capacity of the linear case C_T^I (see **Figure 3**). For $\alpha = 1$, C_s increases with θ/T, and a value of $C_s = 93$ is obtained at $\theta/T = 4$. If all the capacities would be considered for $\alpha = 1$, $C_T \sim N$. The increase in C_s with θ/T is mainly due to the *XMC* and to the *LMC*. When $\theta/T < 1$ a capacity $C_s < 75$ is obtained. However, this degradation in C_s is smaller than in the linear case. It is worth mentioning that for $\alpha = 1$, C_s is greater than the total capacity of the linear case C_T^I. Then we have $C_T^I < C_s \leq C_T^{nl}$, where C_T^{nl} is the total capacity of the non-linear system. This is due to the fact that non-linearity increases the number of linearly independent virtual node states, since correlations between virtual nodes are smaller for non-linear delay-based reservoir computer. In the case without mismatch ($\alpha = 0$) the capacity C_s of the non-linear reservoir computer (see **Figure 3A**) has a maximum as in the linear case at $\theta/T \sim 1.2$. The degradation of C_s is smaller than that of C_T^I in the linear case.

We have shown that the computational capacity is degraded when the sampling output rate is higher than the inverse of the system response time. However, the information processing capacity of delay-based reservoir computers depends not only on output sampling rate (i.e., the separation between the virtual nodes) but also on the detuning between T_p and τ, i.e., α. To study this dependency, we calculate the memory capacities as a function of α for a non-linear delay-based reservoir computer with two different response times: an instantaneous response to the input $T = 0$ (**Figures 4C,D**) and $T = \theta/0.2$ (**Figures 4A,B**). This node separation $\theta = 0.2T$ is the one used in most of the reservoirs with connections through system dynamics [7, 12–18]. The capacities for $T = 0$ correspond to a node separation much larger than T. When $\theta/T \gg 1$ the nodes response to an input reach the steady state after a time θ. Then the reservoir state is given by Equation (2) for $T = 0$. As a consequence, when $\theta/T \gg 1$ the computational capacity tends to the value obtained for $T = 0$. For a mismatch $\alpha = 1$ this limit is reached for $\theta/T > 4$ (see **Figure 3B**). Two values of $\gamma = 0.1$ and $\gamma = 1$ that correspond to low-to-moderate and strong non-linearity, respectively are considered. We also calculate the total capacity as a function of α for a linear reservoir computer with $\theta = 0.2T$ (**Figure 4B**).

The virtual states of delay-based systems with an instantaneous response to the input are given by the map of Equation (3). When N and α are coprimes, we have for $0 < \alpha < N$ a total capacity $C_T \approx N$. Thus, increasing α in the case of $T = 0$ does not increase the total capacity; it only changes the relative contribution of the different capacities to C_T^{nl}. This is clearly shown in **Figure 4D** where a low-to-moderate non-linearity ($\gamma = 0.1$) is considered. Here, the non-linear memory capacities of degree greater than two are zero (i.e., CMC), and $C_s \sim 95$ for $0 < \alpha < 90$. This value is very close to the upper bound for the capacity $C_T = N = 97$. Since C_T is limited by N, there is a trade-off between the linear and non-linear capacities. Then the increase in the *LMC* with α is compensated by a decrease of the *XMC* in **Figure 4D**. In the case of strong non-linearity ($\gamma = 1$), **Figure 4C** shows that C_s is not close to the upper bound for the capacity $C_T = N = 97$. Then there is a significant contribution to C_T^{nl} of capacities with

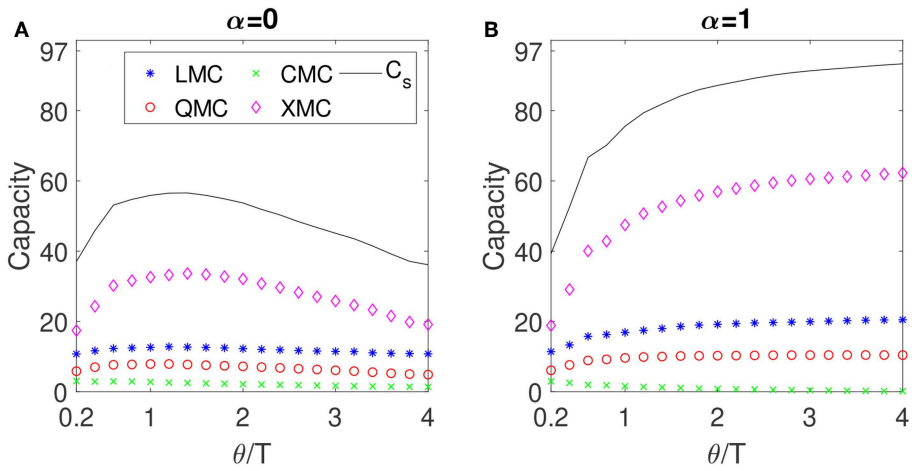


FIGURE 3 | Memory capacities of the non-linear delay-based RC with one delay line as a function of θ/T for **(A)** $\alpha = 0$ and **(B)** $\alpha = 1$ when $\gamma = 0.1$. The blue stars, red circles, green crosses, pink diamonds correspond to the LMC, QMC, CMC, and XMC. The black solid line is the C_s.

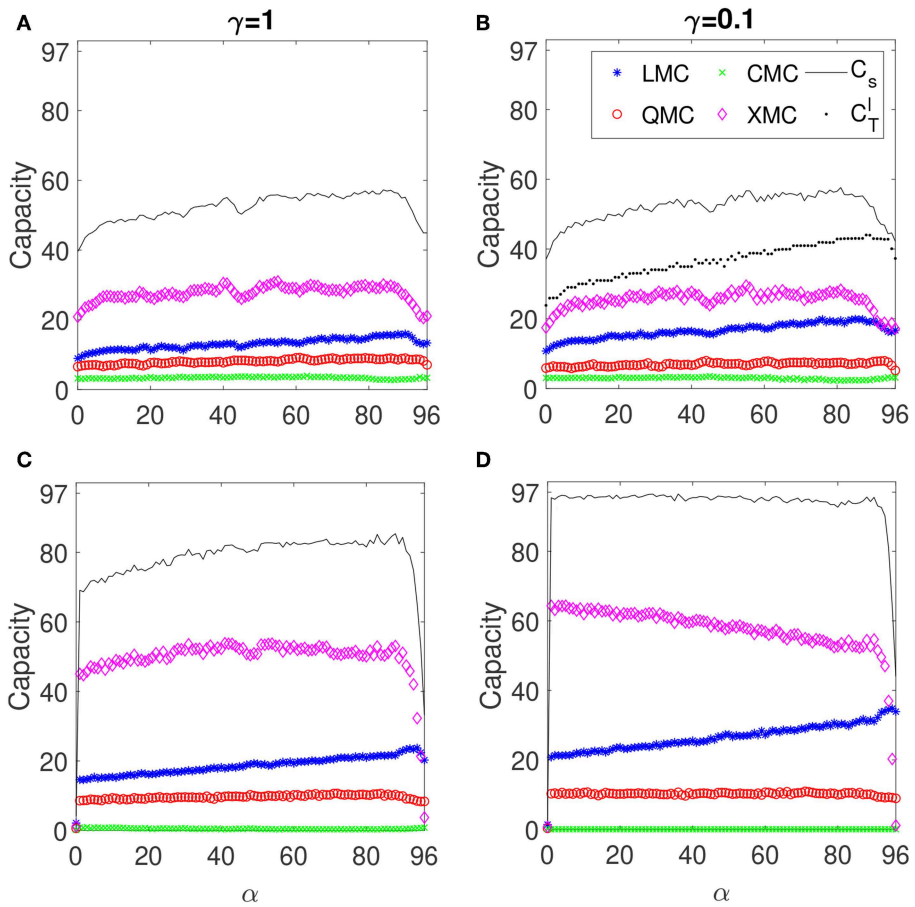


FIGURE 4 | Memory capacities of the one delay-based RC as a function of α. Left panels **(A,C)**: $\gamma = 1$. Right panels **(B,D)**: $\gamma = 0.1$. Top panels **(A,B)**: $T = \theta/0.2$. Bottom panels **(C,D)** $T = 0$. The blue stars, red circles, green crosses, pink diamonds correspond to the LMC, QMC, CMC, XMC, respectively. The solid black line is the C_s. The dotted black line in **(B)** is the C_T^I.

a non-linear degree greater than the ones considered in C_s. An increase in C_s with α is obtained. This increase is mainly due to *LMC* and *XMC*. It only indicates that the contribution to C_T^{nl} of the capacities with a lower non-linear degree considered in C_s increases.

Now we analyze the capacity dependence on α when $\theta/T = 0.2$. We consider integer values of α. Similar results are obtained when α is not an exact integer. We first consider the linear system. In this case the total capacity C_T^l is given by the *LMC*. As seen in **Figure 2A** the capacity is degraded when $\theta < T$ due to the similar evolution in time of close (in time) virtual nodes connected through non-linear node dynamics. **Figure 4B** shows that C_T^l increases with α. A significant increase of nearly 50% is obtained for the capacity when the mismatch is large. This is due to an increase in reservoir diversity. When the mismatch α is increased, virtual nodes are connected through feedback to nodes that are not connected through system dynamics. This improves reservoir diversity, and a larger capacity can be achieved.

In the non-linear case with $\theta/T = 0.2$, **Figures 4A,B** show that regardless of the non-linearity, C_s increases with α. This increase can not be attributed only to a change in the contribution of linear and non-linear capacities to the total capacity C_T^{nl}. As seen for the linear case, when $\theta/T = 0.2$ the total capacity C_T^l increases with α due to an increase in reservoir diversity. This should also lead in the non-linear case to an increase in the total capacity C_T^{nl} with α. It is worth mentioning that in the case of $T = \theta/0.2$ we obtain a similar C_s for low-to-moderate (see **Figure 4B**) and strong (**Figure 4A**) non-linearity. However, the relative contribution of the linear memory capacity is higher for low non-linearity. Finally, note that regardless of the non-linearity and T, higher order capacities such as QMC and CMC remain almost constant with α and the change of C_s is due to LMC and XMC.

3.1.2. Memory Capacities of Two-Delay Reservoir Computers

We have shown that the computational capacity is boosted for small values of θ/T when the delay time of the non-linear node is greater than the data injection time. This mismatch between τ and T_p allows higher processing speeds of delay-based reservoir computers without performance degradation. This is due to the increase in reservoir diversity. To further increase reservoir diversity in the case of $T = \theta/0.2$, we explore the effect of adding a extra feedback line to the non-linear node. **Figure 5** shows the C_s of the two-delay reservoir computer vs. the misalignment of the second delay when $\gamma = 0.1$. The mismatch of the first delay is fixed at $\alpha_1 = 73$ (**Figure 5**, left) and $\alpha = 1$ (**Figure 5**, right). In both cases the maximum of C_s reached for the two-delay system is $C_s \sim 61$. This value is obtained in the two cases, $\alpha_1 = 1$ and $\alpha_1 = 73$, for $\alpha_2 \sim 70$ when $\beta_2 = 0.75$ and just in the case of $\alpha_1 = 73$ also for $\alpha_2 \sim 82$ and $\beta_2 = \beta_1 = 0.4$. The maximum C_s obtained for the two-delay system is slightly higher than the one reached for its one-delay counterpart. In the one-delay system the maximum capacity is $C_s \sim 57$ that is obtained for $\alpha \sim 80$ (see **Figure 4B**). Therefore, the calculated information processing capacity for high sampling output rates can be further increased by using an extra feedback line and delay times greater than the information processing time. However, the second delay does not significantly improve the computational capacity of the one-delay system. Moreover, when the first delay mismatch is fixed near its optimal value for the one-delay system ($\alpha \sim 80$), the effect of the second delay feedback strength or misalignment is small [see **Figure 5** (right)]. However, when the first delay mismatch is not close to its optimal value for the one-delay system, the maximum C_s reached for the one-delay system is outperformed by adding a second delay with a high strength ($\beta_2 = 0.75$) and a mismatch $10 < \alpha_2 < 90$ [see **Figure 5** (left)].

The contributions of the individual memory capacities to C_s for the two-delay system are depicted in **Figures 6**, **7** for $\alpha_1 = 1$ and $\alpha_1 = 73$, respectively. **Figure 6** shows that the increase in C_s obtained for $\alpha_1 = 1$ is mainly due to the increase in LMC and QMC. It is interesting that in the case of $\alpha_2 = 73$, the same $C_s \sim 61$ can be obtained with different relative contributions of the memory capacities to C_s. The case of $\alpha_2 \sim 70$ and $\beta_2 = 0.75$ yields to a higher LMC and a lower XMC than in the one-delay system. The case of $\alpha_2 \sim 82$ and $\beta_2 = 0.4$ gives the $C_s \sim 61$ thanks mainly to the increase in the XMC.

3.2. Delay-Based Reservoir Computer Performance

Finally we study the effect of increasing the mismatch α on the performance of a delay-based reservoir computer for two different response times of the non-linear node dynamics: $T = 0$ and $T = 0.2\theta$. Two tasks are considered: the NARMA-10 task and the equalization of a wireless communication channel. These two tasks are benchmarking tasks used to assess the performance of RC [1, 10].

The NARMA-10 task consists in predicting the output of an auto-regressive moving average from the input $u(t)$. The output $y(t+1)$ is given by:

$$y(t+1) = 0.3y(t) + 0.05y(t)\sum_{i=0}^{9} y(k-i) + 1.5u(t-9)u(t)) + 0.1 \tag{8}$$

The input $u(t)$ is independently and identically drawn from the uniform distribution in [0, 0.5]. Solving the NARMA-10 task requires both memory and non-linearity. **Figure 8** (left) shows the normalized-root-mean-square error (NRMSE) of the NARMA-10 task as a function of α for $\gamma = 0.1$. We consider a small value of $\gamma = 0.1$ because a long memory is required to obtain a good performance for NARMA-10 task. Regardless the response time ($T = 0$ or $T = \theta/0.2$), the NRMSE decreases when the processing and delay times are mismatched ($\alpha > 0$). However, for $T = 0$ the NRMSE is almost the same for a wide variety of values of α, and a mismatch $\alpha = 1$ is enough to obtain a NRMSE = 0.31 close to the absolute minimum (NRMSE = 0.28 for $\alpha = 78$). When the response time of the non-linear node is larger than node separation ($T = \theta/0.2$), the NRMSE decreases from a NRMSE $\approx$ 0.46 at $\alpha = (0, 1)$ to a NRMSE = 0.34 at $\alpha \sim 72$. This is due to the long memory required to obtain a good performance for NARMA-10 task. In the case of $T = \theta/0.2$, the required LMC is not reached until $\alpha \sim 72$ (see **Figure 4B**). Our results show that a similar

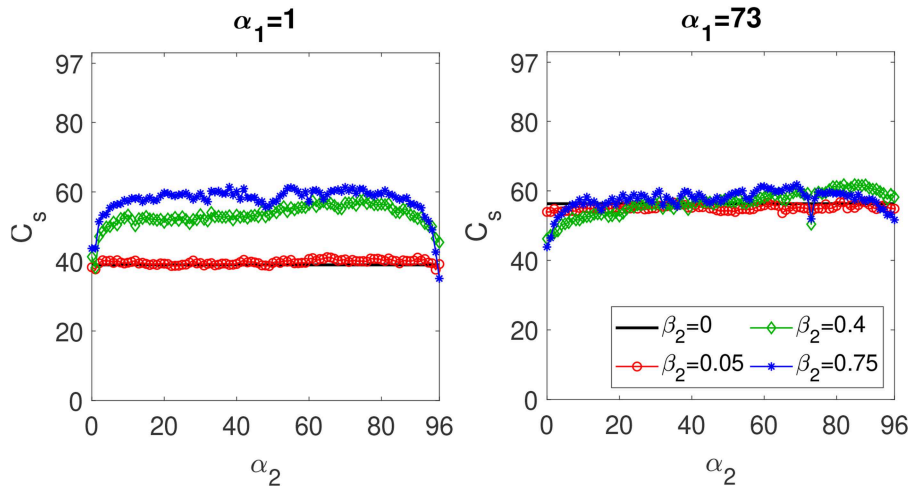


FIGURE 5 | C_s of the two-delay-based RC as function of α_2. **Left**: $\alpha_1 = 1$. **Right**: $\alpha_1 = 73$. The solid black line is the value of C_s for the one-delay case with $\alpha = \alpha_1$. Red circles, green diamonds and blue starts correspond to the C_s with two delays and a β_2 of 0.05, 0.4, and 0.75, respectively. These results are obtained for $T = \theta/0.2$ and $\gamma = 0.1$.

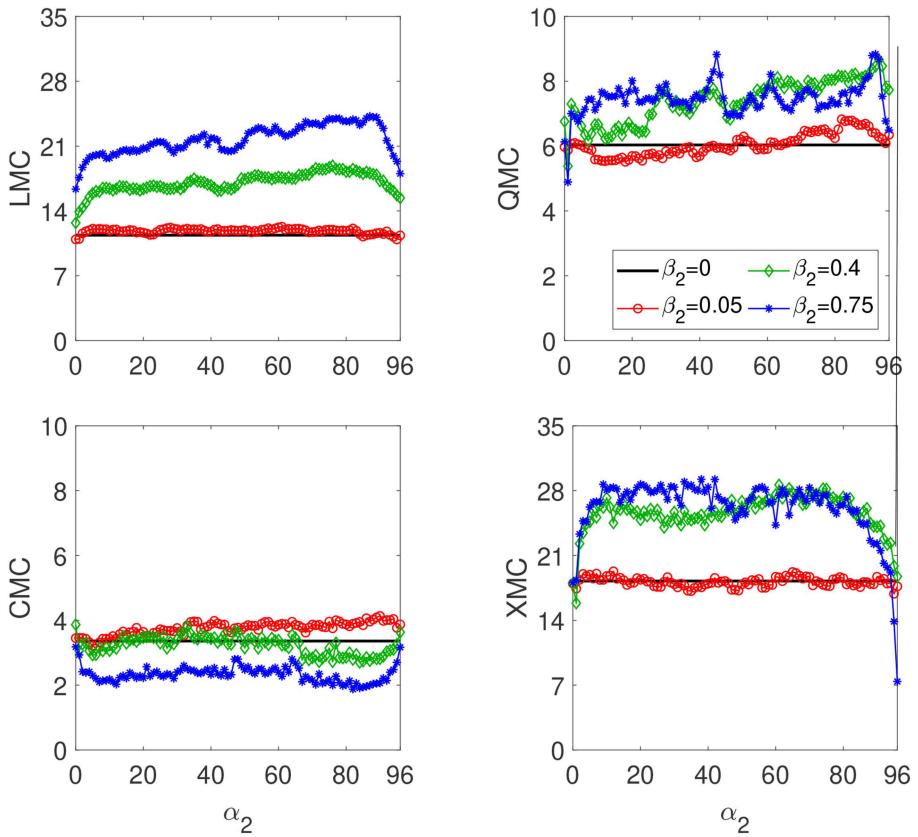


FIGURE 6 | Memory capacities for the two-delay RC as function of α_2 for a fixed $\alpha_1 = 1$, $T = \theta/0.2$ and $\gamma = 0.1$. The red circles, green diamonds and blue stars correspond to β_2 equal to 0.05, 0.4, and 0.75, respectively. The solid black line is for $\beta_2 = 0$ and corresponds to the one-delay system with $\alpha = 1$ and $\beta = 0.8$.

performance can be obtained for small and large values of T/θ thanks to the mismatch α. Therefore, increasing α allows a faster processing information (higher sampling output rate) without causing system performance degradation.

The equalization of a wireless communication channel consists in reconstructing the input signal $s(i)$ from the output sequence of the channel $u(i)$ [1]. The input to the channel is a random sequence of values $s(i)$ taken in

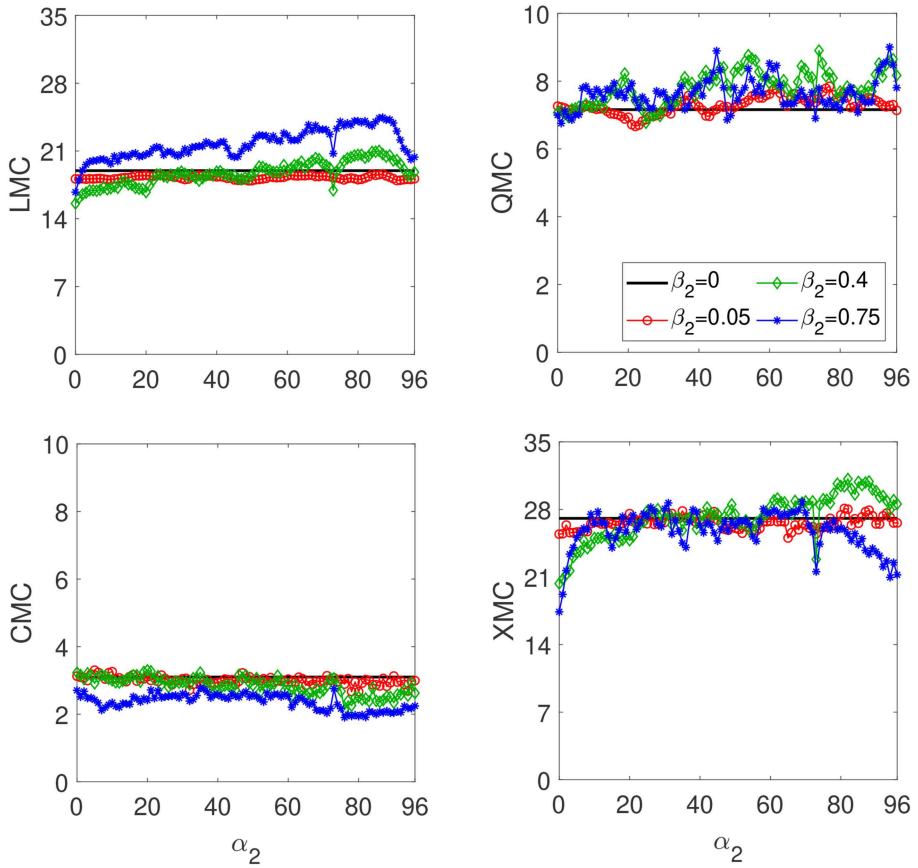


FIGURE 7 | Memory capacities for the two-delay-based RC as function of α_2 for a fixed $\alpha_1 = 73$, $T = \theta/0.2$ and $\gamma = 0.1$. The red circles, green diamonds and blue stars correspond to β_2 equal to 0.05, 0.4, and 0.75, respectively. The solid black line is for $\beta_2 = 0$ and corresponds to the one-delay case with $\alpha = 1$ and $\beta = 0.8$.

$\{-3, -1, 1, 3\}$. The input $s(i)$ first goes through a linear channel yielding:

$$q(i) = 0.08s(i+2) - 0.12s(i+1) + s(i) + 0.18s(i-1) - 0.1s(i-2) + 0.091s(i-3) - 0.05s(i-4) + 0.04s(i-5) + 0.03s(i-6) + 0.01s(i-7)$$

It then goes through a noisy non-linear channel:

$$u(i) = q(i) + 0.036q(i)^2 - 0.011q(i)^3 + v(i), \quad (9)$$

where $v(i)$ is a Gaussian noise with zero mean adjusted in power to give a signal-to-noise ratio (SNR) of 20 dB. The performance is measured using the Symbol Error Rate (SER), that is the fraction of inputs s that are misclassified. The SER for the equalization with a SNR of 20dB is depicted as a function of α for $\gamma = 1$ in **Figure 8** (right). In the case of $T = 0$, there is a clear improvement of the performance from $\alpha = 0$ to $\alpha = 1$ but the errors are almost constant when α is further increased. When $T = \theta/0.2$ performance improves with α until a minimum SER $= 0.012$ is reached when $\alpha \sim 4$. This SER is similar to that obtained when $T = 0$. Then, regardless the value of T/θ, a similar performance is obtained by using the mismatch α. A SER of 0.01 for the channel equalization task has been obtained using an optoelectronic reservoir computer [15].

It is not straightforward how the processing capacity will translate into the performance for specific tasks. Different tasks require to compute functions with different degrees of non-linearity and memory. Information processing capacity should be complemented with those requirements to identify optimized operating conditions for the reservoir. For the channel equalization task, when $T = 0$ the capacities LMC and XMC increase with α showing a very large increase from $\alpha = 0$ to $\alpha = 1$ (see **Figure 4C**). The SER shows also a clear decrease from $\alpha = 0$ to $\alpha = 1$ but it is almost constant when $\alpha > 1$ [see **Figure 8** (right)]. The capacities LMC and XMC achieved for $\alpha = 1$ when $T = 0$ are enough to solve the channel equalization task. However, the quadratic capacity QMC is almost constant when $\alpha > 1$. As a consequence the SER is almost constant for $\alpha > 1$. When taking a small node separation ($\theta = 0.2T$) the capacities LMC and XMC increase with α (see **Figure 4A**). This increase in processing capacity leads to a better performance with α and the SER decreases from 0.017 for $\alpha = 0$ to a minimum error of 0.012 for $\alpha = 4$. This is an improvement in performance of around 30%. However, the increase in the total capacity for $\alpha > 4$ (mainly due to the LMC) does not translate into the

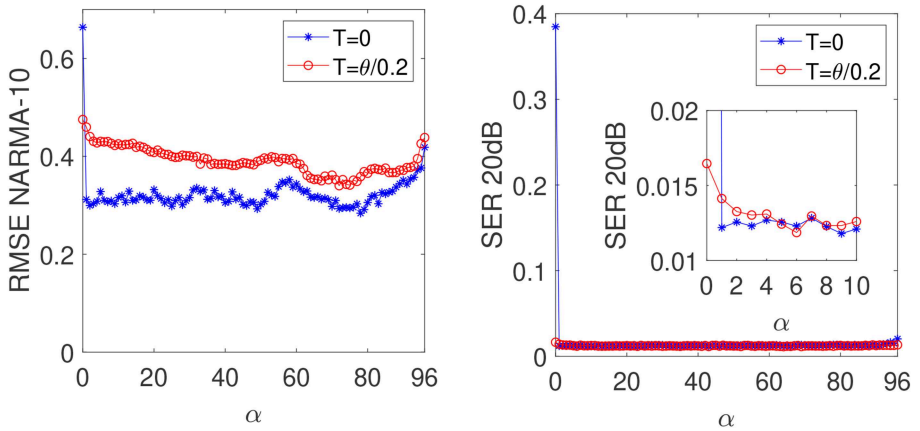


FIGURE 8 | Performance of the non-linear one delay-based RC for two tasks as function of α. **Left**: NARMA-10 for $\gamma = 0.1$. **Right**: Equalization with $SNR = 20$ dB and $\gamma = 1$. The blue stars correspond to the case of $T = 0$ and the red circles to the case of $T = \theta/0.2$.

performance. The reason is the same as for the case of $T = 0$. The capacities LMC and XMC achieved for $\alpha = 4$ are enough to solve the channel equalization task while the capacities QMC and CMC do not increase with α.

The addition of the second delay line to the non-linear node does not improve the performance for the equalization task. In the case of $T = 0$, the extra delay line slightly improves the performance for the NARMA-10 task. The minimum error is NRMSE $\sim$ 0.25 when $\alpha_1 = 77$, $\alpha_2 = 20$ and $\beta_1 = \beta_2 = 0.4$. When $T = \theta/0.2$ a NRMSE = 0.27 is obtained for $\alpha_1 = 77$, $\alpha_2 = 86$, $\beta_1 = 0.05$, and $\beta_2 = 0.75$, while a minimum NRMSE=0.34 was obtained with one delay line for $\alpha \sim 72$. This performance improvement for the NARMA-10 task when $T = \theta/0.2$ is at the cost of adding second delay line and optimizing more parameters to minimize the error. A NRMSE of 0.22 for the NARMA-10 task has been obtained using a photonic reservoir computer based on a coherently driven passive cavity with a greater number of virtual nodes $N = 300$ [24] than the one we used, $N = 97$.

4. DISCUSSION

We have investigated the role of the system response time in the computational capacity of delay-based reservoir computers with a single non-linear neuron. These reservoir computers can be easily implemented in hardware, potentially allowing for high-speed information processing. The information processing rate, given by $1/T_p = (N\theta)^{-1}$, can be increased by using a high sampling output rate (small node separation θ). However, we have shown that the computational capacity is reduced when node separation is smaller than system response time. We can thus conclude that there is a trade-off between information capacity and rate in delay-based reservoir computers. In this context, parallel-based architectures with k non-linear nodes reduce the information processing time by a factor of k for the same total number of virtual nodes. It has been shown [16, 25] that for $(\theta/T) < 1$ and without mismatch between T_p and τ, performance is improved when different activation functions are used for the non-linear nodes. However, the hardware implementation becomes more involved than the one of a delay-based reservoir computer with a single non-linear node.

We have considered a different strategy still based on the simple architecture of a single non-linear node to tackle the trade-off between information capacity and rate. In this strategy, the mismatch between delay and data injection times α is used to increase reservoir diversity when $\theta < T$. For small values of (θ/T) and α, the states of virtual nodes that are separated by less than T (i.e., with an index difference smaller than T/θ) are similar. When the mismatch is increased, virtual nodes are connected through feedback to nodes that are not connected through non-linear node dynamics. Reservoir diversity is then increased. Our results show that the linear memory capacity increases the mismatch α. In this way the capacity degradation due to high sampling output rate is reduced by increasing α.

Another strategy to increase reservoir diversity when $\theta < T$ is to use an extra feedback line. We show that the linear memory capacity can be further increased with this architecture by using long delay times (large mismatch α). However, only a slight increase in the calculated capacity is obtained.

We have also obtained the performance of delay-based reservoir computers for two benchmarking tasks: channel equalization and NARMA-10. Our results show that for fast reservoirs with $\theta < T$ performance improves when the mismatch α increases. A similar performance is obtained for small and large values of (θ/T) for channel equalization and also for NARMA-10 tasks if delay and injection times are mismatched.

We can thus conclude that the processing speed of delay-based reservoir computers can be increased while keeping a good computational capacity by using a mismatch between delay and data injection times.

DATA AVAILABILITY STATEMENT

The datasets generated for this study are available on request to the corresponding author.

AUTHOR CONTRIBUTIONS

SO implemented the program and performed the numerical calculations. All authors contributed to the conception, design of the study, contributed to the discussion of the results, and to the writing of the manuscript.

FUNDING

This work has been funded by the Spanish Ministerio de Ciencia, Innovación y Universidades and Fondo Europeo de Desarrollo Regional (FEDER) through project RTI2018-094118-B-C22.

REFERENCES

1. Jaeger H, Haas H. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. *Science*. (2004) **304**:78–80. doi: 10.1126/science.1091277
2. Maass W, Natschläger T, Markram H. Real-time computing without stable states: a new framework for neural computation based on perturbations. *Neural Comput.* (2002) **14**:2531–60. doi: 10.1162/089976602760407955
3. Lukoševicius M, Jaeger H. Reservoir computing approaches to recurrent neural network training. *Comput Sci Rev*. (2009) **3**:127–49. doi: 10.1016/j.cosrev.2009.03.005
4. Jaeger H. *Short Term Memory in Echo State Networks*. GMD Forschungszentrum Informationstechnik GmbH. GMD Report 152, Sankt Augustin (2002).
5. Larger L, Baylón-Fuentes A, Martinenghi R, Udaltsov VS, Chembo YK, Jacquot M. High-speed photonic reservoir computing using a time-delay-based architecture: million words per second classification. *Phys Rev X*. (2017) **7**:11015. doi: 10.1103/PhysRevX.7.011015
6. Moon J, Ma W, Shin JH, Cai F, Du C, Lee SH, et al. Temporal data classification and forecasting using a memristor-based reservoir computing system. *Nat Electr*. (2019) **2**:480–7. doi: 10.1038/s41928-019-0313-3
7. Appeltant L, Soriano MC, Van Der Sande G, Danckaert J, Massar S, Dambre J, et al. Information processing using a single dynamical node as complex system. *Nat Communicat.* (2011) **2**:468. doi: 10.1038/ncomms1476
8. Paquot Y, Duport F, Smerieri A, Dambre J, Schrauwen B, Haelterman M, et al. Optoelectronic reservoir computing. *Sci Rep*. (2012) **2**:287. doi: 10.1038/srep00287
9. Van der Sande G, Brunner D, Soriano MC. Advances in photonic reservoir computing. *Nanophotonics*. (2017) **6**:561–76. doi: 10.1515/nanoph-2016-0132
10. Rodan A, Tino P. Minimum complexity echo state network. *IEEE T Neural Netw*. (2011) **22**:131–44. doi: 10.1109/TNN.2010.2089641
11. Martinenghi R, Rybalko S, Jacquot M, Chembo YK, Larger L. Photonic nonlinear transient computing with multiple-delay wavelength dynamics. *Phys Rev Let.* (2012) **108**:244101. doi: 10.1103/PhysRevLett.108.244101
12. Soriano MC, Ortín S, Keuninckx L, Appeltant L, Danckaert J, Pesquera L, et al. Delay-based reservoir computing: noise effects in a combined analog and digital implementation. *IEEE Trans Neural Netw Learn Syst*. (2015) **26**:388–93. doi: 10.1109/TNNLS.2014.2311855
13. Larger L, Soriano MC, Brunner D, Appeltant L, Gutierrez JM, Pesquera L, et al. Photonic information processing beyond Turing: an optoelectronic implementation of reservoir computing. *Opt Express*. (2012) **20**:3241–49. doi: 10.1364/OE.20.003241
14. Soriano MC, Ortín S, Brunner D, Larger L, Mirasso CR, Fischer I, et al. Optoelectronic reservoir computing: tackling noise-induced performance degradation. *Opt Express*. (2013) **21**:12–20. doi: 10.1364/OE.21.000012
15. Ortín S, Soriano MC, Pesquera L, Brunner D, San-Martín D, Fischer I, et al. A unified framework for reservoir computing and extreme learning machines based on a single time-delayed neuron. *Sci Rep*. (2015) **5**:14945. doi: 10.1038/srep14945
16. Ortín S, Pesquera L. Reservoir computing with an ensemble of time-delay reservoirs. *Cogn Comput.* (2017) **9**:327–36. doi: 10.1007/s12559-017-9463-7
17. Jaeger H. *Tutorial on training recurrent neural networks, covering BPTT, RTRL, EKF and the 'echo state network' approach*. Technical Report GMD Report 159, German National Research Center for Information Technology, Sankt Augustin (2002).
18. Brunner D, Soriano MC, Mirasso CR, Fischer I. Parallel photonic information processing at gigabyte per second data rates using transient states. *Nat Commun*. (2013) **4**:1364. doi: 10.1038/ncomms2368
19. Duport F, Schneider B, Smerieri A, Haelterman M, Massar S. All-optical reservoir computing. *Opt Express*. (2012) **20**:22783–95. doi: 10.1364/OE.20.022783
20. Dambre J, Verstraeten D, Schrauwen B, Massar S. Information processing capacity of dynamical systems. *Sci Rep*. (2012) **2**:514. doi: 10.1038/srep00514
21. Appeltant L. *Reservoir Computing Based on Delay-Dynamical Systems*. Vrije Universiteit Brussel/Universitat de les Illes Balears, Brussels (2012).
22. Ortín S, Appeltant L, Pesquera L, der Sande G, Danckaert J, Gutierrez JM. Information processing using an electro-optic oscillator subject to multiple delay lines. In: *International Quantum Electronics Conference*. Piscataway, NJ: Optical Society of America (2013).
23. Nieters P, Leugering J, Pipa G. Neuromorphic computation in multi-delay coupled models. *IBM J Res Dev*. (2017) **61**:8:1–8:9. doi: 10.1147/JRD.2017.2664698
24. Vinckier Q, Duport F, Smerieri A, Vandoorne K, Bienstman P, Haelterman M, et al. High-performance photonic reservoir computer based on a coherently driven passive cavity. *Optica*. (2015) **2**:438–46. doi: 10.1364/OPTICA.2.000438
25. Ortín S, Pesquera L, Gutiérrez JM. Memory and nonlinear mapping in reservoir computing with two uncoupled nonlinear delay nodes. In: *Proceedings of the European Conference on Complex Systems 2012*. Berlin: Springer (2013). p. 895–9. doi: 10.1007/978-3-319-00395-5_107

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Physics

ORIGINAL RESEARCH
published: 13 February 2020
doi: 10.3389/fphy.2020.00010

A Bayesian Approach to the Naming Game Model

Gionni Marchetti, Marco Patriarca and Els Heinsalu*

NICPB–National Institute of Chemical Physics and Biophysics, Tallinn, Estonia

We present a novel Bayesian approach to semiotic dynamics, which is a cognitive analog of the naming game model restricted to two conventions. The model introduced in this paper provides a general framework for studying the combined effects of cognitive and social dynamics. The one-shot learning that characterizes the agent dynamics in the basic naming game is replaced by a word-learning process in which agents learn a new word by generalizing from the evidence garnered through pairwise-interactions with other agents. The principle underlying the model is that agents—like humans—can learn from a few positive examples and that such a process is modeled in a Bayesian probabilistic framework. We show that the model presents some analogies with the basic two-convention naming game model but also some relevant differences in the dynamics, which we explain through a geometric analysis of the mean-field equations.

Keywords: complex systems, language dynamics, bayesian statistics, cognitive models, consensus dynamics, semiotic dynamics, naming game, individual-based models

OPEN ACCESS

Edited by:
Raul Vicente,
Max-Planck-Institut für Himforschung, Germany

Reviewed by:
Chengyi Xia,
Tianjin University of Technology, China
Adam Lipowski,
Adam Mickiewicz University, Poland

****Correspondence:***
Marco Patriarca
marco.patriarca@kbfi.ee

Specialty section:
This article was submitted to Interdisciplinary Physics, a section of the journal Frontiers in Physics

Received: *14 June 2019*
Accepted: *09 January 2020*
Published: *13 February 2020*

Citation:
Marchetti G, Patriarca M and Heinsalu E (2020) A Bayesian Approach to the Naming Game Model. Front. Phys. 8:10. doi: 10.3389/fphy.2020.00010

1. INTRODUCTION

A basic question in complexity theory is how the interactions between the units of the system lead to the emergence of ordered states from initially disordered configurations [1, 2]. This general question can concern different phenomena ranging from phase transitions in condensed matter systems and self-organization in living matter to the appearance of norm conventions and cultural paradigms in social systems. In order to study social interactions and cooperation, different models have been used: from those based on analogies with condensed matter systems (such as spin systems) or statistical mechanical models (e.g., using a master equation approach) to those formally equivalent to ecological competition models [1] or many-agents models in a game-theoretical framework [3–5]. Among the various models, opinion dynamics and cultural spreading models represent an example of a valuable theoretical framework for a quantitative description of the emergence of social consensus [2].

Within the spectrum of phenomena associated with consensus dynamics, the emergence of human language remains a challenging question because of its multi-fold nature, characterized by biological, ecological, social, logical, and cognitive aspects [6–10]. Language dynamics [11, 12] has provided a set of models describing various phenomena of language competition and language change in a quantitative way, focusing on the mutual interactions of linguistic traits (such as sounds, phonemes, grammatical rules, or the use of languages understood as fixed entities), possibly under the influence of ecological and social factors, modeling such interactions through analogy with biological competition and evolution.

However, even the basic learning process of a single word has a complex dynamics due to the associated cognitive dimension: to learn a word means to learn both a *concept*, understood as a pointer to a subset of objects [10, 13, 14], and a corresponding linguistic label, for example the

name used for communicating the concept. The double concept↔name nature of words has been studied in semiotic dynamics models, which study the consensus dynamics of language, i.e., if and how consensus about the use of certain names to refer to a certain object-concept emerges in a group of N interacting agents.

Examples of semiotic models are those of Hurford [15] and Nowak et al. [16] (see also [17, 18]). In the basic version of the model of Nowak et al. [16], the language spoken by each agent i ($i = 1, \ldots, N$) is defined by two personal matrices, $\mathcal{U}^{(i)}$ and $H^{(i)}$, representing the links of a bipartite network joining Q names and R concepts: (1) the active matrix $\mathcal{U}^{(i)}$ represents the concept → name links, where the element $\mathcal{U}^{(i)}_{q,r}$ ($q \in (1, Q)$, $r \in (1, R)$) gives the probability that agent i will utter the qth name to communicate the rth concept; (2) the passive matrix $H^{(i)}$ represents the name → concept links, in which the element $\mathcal{H}^{(i)}_{q,r}$ represents the probability that an agent interprets the qth name as referring to the rth concept. In Hurford's and Nowak's models, the languages (i.e., the active and passive matrices) of each individual evolve over time according to a game-theoretical dynamics in which agents gain a reproductive advantage if their matrices are associated with a higher communication efficiency. These studies have achieved interesting results, showing, e.g., that the system self-organizes in an optimal way with only non-ambiguous one-to-one links between objects and sounds, when possible, and explaining why homonyms are more frequent than synonyms [15–18].

Another example of a semiotic model is the naming game (NG) model [19, 41], detailed below, where only one concept is considered ($R = 1$) together with its links to a set of $Q > 1$ different names. It is possible to reformulate the model through the lists of the name↔concept connections $\mathcal{L}_i$ known to each agent i rather than in terms of the matrices $\mathcal{U}^{(i)}$ and $H^{(i)}$. In the case of the NG with two names A and B, the list of the generic ith agent can be $\mathcal{L}_i = \emptyset$ if no such connection is known, $\mathcal{L}_i = (A)$ or $\mathcal{L}_i = (B)$, if only one name is known to refer to concept C, or $\mathcal{L}_i = (A, B)$ if both name↔concept connections are known. At variance with Hurford's and Nowak's models, in the basic NG model, there is no population dynamics, and consensus is achieved through horizontal interactions between pairs of agents, who carry out a negotiation dynamics in which they may agree on the use of a word, possibly erasing the other word from their lists.

In the signaling game of Lenaerts et al. [20], the basic add/remove agreement dynamics of the NG model is replaced by a reinforcement scheme describing an underlying cognitive dynamics. Such a scheme is defined within a learning automata framework in which the single probabilities, linking the qth word and the rth object, are updated in time depending on the outcome of pair-wise communications—the system is characterized by the same complex landscape of R concepts and Q names as in Hurford's and Nowak's models. The model works with a basic horizontal dynamics, as in the NG model, but it has a general framework of language change, which can include oblique (teacher↔student) and vertical (parent↔offspring) communications. An NG-like language dynamics, with a similar cognitive reinforcement mechanism, was also studied by Lipowska & Lipowski, both in the single- and the many-object version [21]. They also studied how the underlying topology, e.g., a random network or a regular lattice, can have a crucial role in determining the type of final state, characterized by a global consensus or by different types of local consensus fragmented into patches.

In the models mentioned, words and concepts are fixed, though their links are dynamically determined through the interactions between agents. To make further extensions of such semiotic dynamics models toward a cognitive direction is not a trivial task, both because of the complexity of the problem—for example, a two-opinion variant of the NG model that takes into account committed groups produces a remarkable phase diagram [22]—and because, in order to describe mathematically actual cognitive effects, entirely new features need to be taken into account [23]. A natural framework is represented by Bayesian inference, both for its general analogy with actual learning processes and especially because supported by various experiments. For example, Bayes inference underlies the agent-based model of binary decision-making introduced by [24], which is shown to interpolate well some real datasets on binary option choices. See Pérez et al. [25] for another example of Bayes-based modeling and reproduction of a real decision-making experiment.

The goal of the present paper is to construct a minimal model to study the interplay of cognitive and social dynamical dimensions. The new model (see section 2.3) is similar to the two-conventions NG but contains relevant differences that describe the cognitive dimension of word-learning. Using semiotic dynamics models as a starting point is a natural choice, and the NG is a convenient framework due to its simple yet general underlying idea, which allows applications to the emergence of different conventions. Furthermore, the NG can be coupled to various underlying processes, such as mutations, population growth, and ecological constraints, and can be easily embedded in the topology of a complex network [19, 26]. The cognitive extension of the NG is done within the experimentally validated Bayesian framework of Tenenbaum [10] (see also [13, 14, 27–30]). In the resulting cognitive framework, an individual can learn a concept from a small number of examples, a very remarkable feature of human learning [10, 31, 32], in contrast with machine learning algorithms, which require a large number of examples to generalize successfully [33–35]. In section 3, we present and discuss the features of the semiotic dynamics emerging from the numerical simulations and quantitatively compare them with those of the two-conventions NG model. It is shown that while the Bayesian NG model always reaches consensus, like the basic NG, the corresponding dynamics presents relevant differences related to the probabilistic learning process. We study in detail the stability and the other novel features of the dynamics in section 4. A summary of the work and a discussion of other possible outcomes to be expected from the interplay of the cognitive and the social dynamics, not considered in this paper but representing natural extensions of the present study, are outlined in section 5.

2. A BAYESIAN LEARNING APPROACH TO THE NAMING GAME

2.1. The Two-Conventions Naming Game Model

Before introducing the new model, we outline the basic two-conventions NG model [36], in which there is a single concept C, corresponding to an external object, and two possible names (synonyms) A and B for referring to C. Thus, the possibility of homonymy is excluded [26]. Each agent i is equipped with the list $\mathcal{L}_i$ of the names known to the agent. We assume that at $t = 0$, each agent i knows either A or B and therefore has a list $\mathcal{L}_i = (A)$ or $\mathcal{L}_i = (B)$, respectively.

During a pair-wise interaction, an agent can act as a speaker, when conveying a word to another agent, or as a hearer, when receiving a word from a speaker. One can think of an agent conveying a word as uttering a name, e.g., A, while pointing at an external object, corresponding to concept C: thus, the hearer records not only the name A but also the name↔concept association between A and C. At a later time $t > 0$, the list $\mathcal{L}_i$ of the ith agent can contain one or both names, i.e., $\mathcal{L}_i = (A)$, (B), or (A, B).

The system evolves according to the following update rules [26]:

1. Two agents i and j, the speaker and the hearer, respectively, are randomly selected.
2. The speaker i randomly extracts a name (here, either A or B) from the list $\mathcal{L}_i$ and conveys it to the hearer j. Depending on the state of agent j, the communication is usually described as:
   a. *Success*: the conveyed name is also present in the hearer's list $\mathcal{L}_j$, i.e., agent j also knows its meaning; then, the two agents erase the other name from their lists, if present.
   b. *Failure*: the conveyed name is *not* present in the hearer's list $\mathcal{L}_j$; then, agent j records and adds it to list $\mathcal{L}_j$.
3. Time is increased by one step, $t \to t + 1$, and the simulation is reiterated from the first point above.

Examples of unsuccessful and successful communications are each schematized in the left panel (A) of **Figure 1**; see [41] for more examples. Despite its simple structure, the basic NG model describes the emergence of consensus about which name to use, which is reached for any (disordered) initial configuration [37].

2.2. Toward a Bayesian Naming Game Model

From a cognitive perspective, a "communication failure" of the NG model can be understood as a learning process in which the hearer learns a new word. It is a "one-shot learning process" because it takes place instantaneously (in a single time step) and independently of the agent's history (i.e., of the previous knowledge of the agent). However, modeling an actual learning process should take into account the agents' experience, based on previous observations (the data already acquired) as well as the uncertain/incomplete character naturally accompanying any learning process.

Here, the one-shot learning is replaced by a process that can describe basic but realistic situations, such as the prototypical "linguistic games" [38]. For example, consider a "lecture game," in which a lecturer (speaker) utters the name A of an object and shows a real example "+" of the object to a student (hearer), repeating this process a few times. Then, the teacher can e.g., (a) show another example and ask the student to name the object, (b) utter the same name and ask the student to show an example of that object, or (c) do both things (uttering the name and showing the object) and asking the student whether the name↔object correspondence is correct. The student will not be able to answer correctly if they have not received some examples enabling them to generalize the concept C corresponding to the object in association with name A. To model these and similar learning processes, we need a criterion enabling the hearer to assess the degree of equivalence between the new example and the examples recorded previously.

The starting point for the replacement of the one-shot learning is Bayes' theorem. According to Bayes' theorem, the posterior probability $p\left(h|X\right)$ that the generic hypothesis h is the true hypothesis, after observing new evidence X, reads [39, 40],

$$p\left(h|X\right) = \frac{p\left(X|h\right)p\left(h\right)}{p\left(X\right)}\,. \tag{1}$$

Here, the prior probability $p\left(h\right)$ gives the probability of occurrence of the hypothesis h before observing the data, and $p\left(X|h\right)$ gives the probability of observing X if h is given. Finally, $p\left(X\right)$ gives the normalization constraint; in applications, it can be evaluated as $p\left(X\right) = \sum_{h'} p\left(X|h'\right)p\left(h'\right)$, where $\{h'\} \in H$ represents the set of hypotheses, within the hypothesis space H.

The next step is to find a way to compute explicitly the posterior probability $p\left(h|X\right)$ through a representation of the concepts and their relative examples in a suitable hypothesis space H of the possible extensions of a given concept C, constituted by the mutually exclusive and exhaustive hypotheses h. Following the experimentally verified Bayesian statistical framework of Tenenbaum [10, 31], we adopt the paradigmatic representation of a concept as a geometrical shape. For example, the concept of the "healthy level" of an individual in terms of the levels of cholesterol x and insulin y, defined by the ranges $x_a \leq x \leq x_b$ and $y_a \leq y \leq y_b$, where x_i and y_i $(i = a, b)$ are suitable values, represents a rectangle in the Euclidean x-y plane $\mathbb{R}^2$. Examples of healthy levels of specific individuals $1, 2, \ldots$ correspond to points $(x_1, y_1), (x_2, y_2), \cdots \in \mathbb{R}^2$. In the following, we assume that a hypothesis h is represented by an axis-parallel rectangular region in $\mathbb{R}^2$. **Figure 2** shows four positive examples, denoted by the symbol "+," associated to four different points of the plane, consistent with (i.e., contained in) three different hypotheses, shown as rectangles.

The problem of learning a word is now recast into an equivalent problem, consisting of acquiring the ability to infer whether a new example z recorded, corresponding to a new point "+" in $\mathbb{R}^2$, corresponds to the concept C after having seen

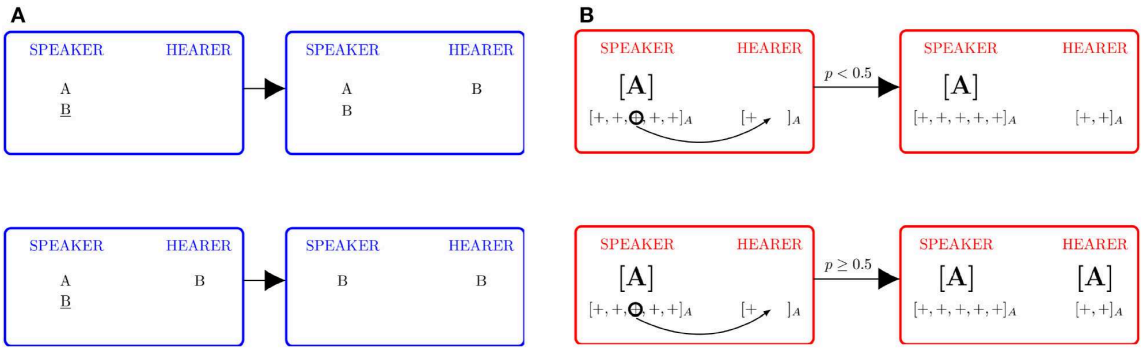


FIGURE 1 | Comparison of the basic and Bayesian NG models. **(A)** Basic two-conventions NG model. In a communication failure (upper figure), the name conveyed, *B* in the example, is not present in the list of the hearer, who adds it to the list. In a communication success (lower figure), the word *B* is already present in the hearer's list, and both agents erase *A* from their lists. **(B)** Bayesian NG model. In order to convey an example "+" to the hearer in association with name *A*, the speaker must have already generalized concept *C* in association with *A*, represented here by the label [A]. In a communication failure (upper figure), the hearer computes the Bayes probability *p*, and the result is a $p < 1/2$; then, the only outcome is that the hearer records the example (*reinforcement*). In the Bayesian NG, there are two ways in which the communication can be successful. The first way (lower figure) is when $p \geq 1/2$: the hearer generalizes *C* in association with *A* and attaches the label [*A*] to the inventory. The second way (not shown) is the *agreement* process, analogous to that of the basic NG, when both agents had already generalized concept *C* in association with name *A* and remove label [B] from their lists, if present. See text for further details.

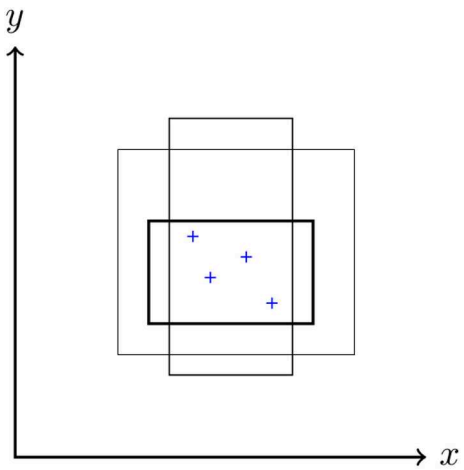


FIGURE 2 | Three different hypotheses represented as axis-parallel rectangles in $\mathbb{R}^2$, and four positive examples "+" that are all consistent with the three hypotheses. The set of all the rectangles that can be drawn in the plane constitutes the hypothesis space *H*.

a small set of positive examples "+" of *C*. More precisely, let $X = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ be a sequence of n examples of the true concept *C*, already observed by the hearer, and $z = (z_1, z_2)$ the new example. The learner does not know the true concept *C*, i.e., the exact shape of the rectangle associated to *C*, but can compute the generalization function $p(z \in C|X)$ by integrating the predictions of all hypotheses h, weighted by their posterior probabilities $p(h|X)$:

$$p(z \in C|X) = \int_{h \in H} p(z \in C|h)\, p(h|X)\, dh\,. \tag{2}$$

Clearly, $p(z \in C|h) = 1$ if $z \in h$ and 0 otherwise. By means of Bayes' theorem (1), one can obtain the right Bayesian probability for the problem at hand. A successful generalization is then defined quantitatively by introducing a threshold p^*, representing an acceptance probability: *an agent will generalize if the Bayesian probability* $p(z \in C|X) \geq p^*$. The value $p^* = 1/2$ is assumed, as in Tenenbaum [31].

We assume that an Erlang prior characterizes the agents' background knowledge. For a rectangle in $\mathbb{R}^2$ defined by the tuple (l_1, l_2, s_1, s_2), where l_1, l_2 are the Cartesian coordinates of its lower-left corner and s_i its sides along dimension $i = 1, 2$, the Erlang prior density is Tenenbaum [10, 31]

$$p_E = s_1 s_2 \exp\left\{-\left(\frac{s_1}{\sigma_1} + \frac{s_2}{\sigma_2}\right)\right\}, \tag{3}$$

where the parameters σ_i represent the actual sizes of the concept, i.e., they are the sides of the concept rectangle *C* along dimension *i*. The choice of a specific informative prior, such as the Erlang prior, is well motivated by the fact that, in the real world, individuals always have some prior knowledge or expectation. In fact, a Bayesian learning framework with an Erlang prior of the form (3) well describes experimental observations of the learning processes of human beings [31]. The final expression used below for computing the Bayesian probability p that, given the set of previous examples X, the new example z falls in the same category of concept *C*, reads [31].

$$p(z \in C|X) \approx \frac{\exp\left\{-\left(\frac{\tilde{d}_1}{\sigma_1} + \frac{\tilde{d}_2}{\sigma_2}\right)\right\}}{\left[\left(1 + \frac{\tilde{d}_1}{r_1}\right)\left(1 + \frac{\tilde{d}_2}{r_2}\right)\right]^{n-2}}. \tag{4}$$

Here, r_i $(i = 1, 2)$ is an estimate of the extension of the set of examples along direction i, given by the maximum mutual distance along dimension i between the examples of X; $\tilde{d}_i$

measures an effective distance between the new example z and the previously recorded examples, i.e., $\tilde{d}_i = 0$ if z_i falls inside the value range of the examples of X along dimension i, otherwise $\tilde{d}_i$ is the distance between z and the nearest example in X along the dimension i. Equation (4) is actually a "quick-and-dirty" approximation that is reasonably good, except for $n \leq 3$ and $r_i \leq \sigma/10$, estimating the actual generalization function within a 10% error; see Tenenbaum [10, 31] for details. Despite these approximations, Equation (4) will ensure that our computational model, described in the next section, retains the main features of the Bayesian learning framework. It is to be noticed that, for the validity of the Bayesian framework, it is crucial that the examples are drawn randomly from the concept (strong sampling assumption), i.e., they are extracted from a probability density that is uniform in the rectangle corresponding to the true concept [31]. This definition of generalization is now applied below to word-learning.

2.3. The Bayesian Word-Learning Model

Based on the Bayesian learning framework discussed above, in this section we introduce a minimal Bayesian individual-based model of word-learning. For the sake of clarity, in analogy with the basic NG model, we study the emergence of consensus in the simple situation, in which two names A and B can be used for referring to the same concept C in pair-wise interactions among N agents.

At variance with the NG model, here, in each basic pair-wise interaction, an agent i, acting as a speaker, conveys an example "+" of concept C, in association with either name A or name B, to another agent j, who acts as hearer ($i, j = 1, \ldots, N$). In order to be able to communicate concept C by uttering a name, e.g., name A, the speaker i must have already generalized concept C in association with name A. This is signaled by the presence of name A in list $\mathcal{L}_i$. On the other hand, the hearer j always records the example received in the respective inventory, in the example, the inventory $[+ + + \ldots]_A$.

The state of a generic agent i at time t is defined by:

- List $\mathcal{L}_i$, to which a name is added whenever agent i generalizes concept C in association with that name; agent i can use any name in $\mathcal{L}_i$ to communicate C;
- Two inventories $[+ + + \ldots]_A$ and $[+ + + \ldots]_B$ containing the examples "+" of concept C received from the other agents in association with name A and B, respectively.

It is assumed that, initially, each agent knows one word: a fraction $n_A(0)$ of the agents know concept C in association with name A, and the remaining fraction $n_B(0) = 1 - n_A(0)$ in association with name B—no agent knows both words, $n_{AB}(0) = 0$. We will examine three different initial conditions:

$$\text{Symmetric initial conditions (\textbf{SIC}):}\ n_A(0) = n_B(0) = 0.5$$
$$\text{Asymmetric initial conditions (\textbf{AIC}):}\ n_A(0) = 0.3,\ n_B(0) = 0.7$$
$$\text{Reversed case of AIC (\textbf{AICr}):}\ n_A(0) = 0.7,\ n_B(0) = 0.3$$

Initially, each agent i, within the fraction $n_A(0)$ of agents that know name A is assigned $n_{ex,A} = 4$ examples "+" of concept C in association with name A but no examples in association with the other name B, so that agent i has an A-inventory $[+ + ++]_A$ and an empty B-inventory $[\cdot]_B$. The complementary situation holds for the other agents that know only name B, who initially receive $n_{ex,B} = 4$ examples of concept C in association with name B but none in association with A. This choice, somehow arbitrary, is dictated by the condition that (Equation 4) becomes a good approximation for $n > 3$ [10].

Examples are points uniformly generated inside the fixed rectangle corresponding to the true concept C, here assumed to be a rectangle with lower-left corner coordinates $(0, 0)$ and sizes $\sigma_1 = 3$ and $\sigma_2 = 1$ along the x- and y-axis, respectively. Results are independent of the assumed numerical values; in particular, no appreciable variation in the convergence times t_{conv} is observed as the rectangle area is varied, which is consistent with the strong sampling assumption on which the Bayesian learning framework rests; see Tenenbaum [10] and section 3.

Furthermore, we introduce an element of asymmetry between the names A and B, related to the word-learning process: different *minimum numbers of examples* $n^*_{ex,A} = 5$ and $n^*_{ex,B} = 6$ will be used, which are needed by agents to generalize concept C in association with A and B, respectively. This is equivalent to assuming that concept C is slightly easier to learn in association with name A than B. Such an asymmetry plays a relevant role in the model dynamics in differentiating the Bayesian generalization functions p_A and p_B from each other; see section 4.

The dynamics of the model can be summarized by the following dynamical rules:

1. A pair of agents i and j, acting as speaker and hearer, respectively, are randomly chosen among the agents.
2. The speaker selects randomly (a) a name from the list $\mathcal{L}_i$ (or selects the name present if $\mathcal{L}_i$ contains a single name), for example, A (analogous steps follow if the word B is selected); (b) an example z among those contained in the corresponding inventory $[+ + + \ldots]_A$;
   then the speaker i conveys the example extracted z in association with (e.g., uttering) the name selected A to the hearer j.
3. The hearer adds the new example z (in association with A) to the inventory $[+ + + \ldots]_A$. This *reinforcement* process of the hearer's knowledge always takes place.
4. Instead, the next step depends on the state of the hearer:
   (a) *Generalization*. If the selected name, A in the example, is *not* present in the hearer's list $\mathcal{L}_j$, then the hearer j computes the relative Bayesian probability $p_A = p(z \in C|X_A)$ that the new example z falls in the same category of concept C, using the examples previously recorded in association with A, i.e., from the set of examples $X_A \in [+ + + \ldots]_A$. If $p_A \geq 1/2$, the hearer has managed to generalize concept C and connects the inventory $[+ + + \ldots]_A$ to name A; this is done by adding name A to list $\mathcal{L}_j$. Starting from this moment, agent j can communicate concept C to other agents by conveying an example taken from the inventory $[+ + + \ldots]_A$ while uttering the name A. If $p_A < 1/2$, the hearer has not managed to generalize the concept and nothing more happens (the

reinforcement of the previous point is the only event taking place).

(b) *Agreement*. The name uttered by the speaker, A in the example, is present in the hearer's list $\mathcal{L}_j$, meaning that agent j has already generalized concept C in association with name A and has connected the corresponding inventory $[+\ +\ +\ldots]_A$ to A. In this case, the hearer and the speaker proceed to make an agreement, analogous to that of the NG model, leaving A in their lists $\mathcal{L}_i$ and $\mathcal{L}_j$ and removing B, if present. No examples contained in any inventory are removed.

5. Time is updated, $t \to t+1$, and the simulation is reiterated from the first point above.

Two examples of the Bayesian word-learning process, a successful and an unsuccessful one, are illustrated in the cartoon in the **Figure 1B**. **Table 1** lists the possible encounter situations, together with the corresponding relevant probabilities.

Notice that an agent i can enter a pair-wise interaction with a non-empty inventory of examples, e.g., $[+\ +\ +\ldots]_A$, associated to name A, without being able to use name A to convey examples to other agents, i.e., without having the name A in list $\mathcal{L}_i$ due to not having generalized concept C in association with A. Those examples can have different origins: (1) in the initial conditions, when $n_{ex,A}$ randomly extracted examples associated to A and $n_{ex,B}$ to B are assigned to each agent; (2) in previous interactions, in which the examples were conveyed by other agents; (3) in an agreement about convention B, which removed label A from list $\mathcal{L}_i$ while leaving all the corresponding examples in the inventory associated to name A. In the latter case, the inventory $[+\ +\ +\ldots]_A$ may be "ready" for a generalization process, since it contains a sufficient number of examples, i.e., agent i will probably be able to generalize as soon as another example is conveyed by an agent. This situation is not as peculiar as it may look at first sight. In fact, there is a linguistic analog in the case where a speaker that loses the habit of using a certain word (or a language) A can regain it promptly if exposed to A again.

Notice also that without the agreement dynamics scheme introduced in the model, borrowed from the basic NG model, the population fraction n_{AB} of individuals who know both A and B ($n_A+n_B+n_{AB}=1$) would be growing, until eventually $n_{AB}=1$.

3. RESULTS

In this section, we numerically study the Bayesian NG model introduced above and discuss its main features. We limit ourselves to studying the model dynamics of a fully-connected network.

In the new learning scheme, which replaces the one-shot learning of the two-conventions NG model, an individual generalizes concept C on a suitable time scale $\Delta t > 1$ rather than during a single interaction. However, a few examples are sufficient for an agent to generalize concept C, as in a realistic concept-learning process. This is visible from the Bayesian probabilities p_A and p_B computed by agents in the role of hearer, according to Equation (4), once at least $n^*_{ex,A}=5$ and $n^*_{ex,B}=6$ examples "+", respectively, have been stored in the

TABLE 1 | Pair-wise interactions in the Bayesian NG model.

S-List (before)	Name conveyed	H-List (before)	Branching probability	Process	Condition	S-List (after)	H-List (after)
(A)	$\xrightarrow{A}$	(A)	$(q=1)$	Reinforcement	always	(A)	(A)
(A)	$\xrightarrow{A}$	(B)	$(q=1)$	Reinforcement	$p_A<1/2$	(A)	(B)
			$(q=1)$	Learning	$p_A\geq 1/2$	(A)	(A,B)
(A)	$\xrightarrow{A}$	(A,B)	$(q=1)$	Agreement	always	(A)	(A)
(B)	$\xrightarrow{B}$	(A)	$(q=1)$	Reinforcement	$p_B<1/2$	(B)	(A)
			$(q=1)$	Learning	$p_B\geq 1/2$	(B)	(A,B)
(B)	$\xrightarrow{B}$	(B)	$(q=1)$	Reinforcement	always	(B)	(B)
(B)	$\xrightarrow{B}$	(A,B)	$(q=1)$	Agreement	always	(B)	(B)
(A,B)	$\xrightarrow{A}$	(A)	$q=1/2$	Agreement	always	(A)	(A)
(A,B)	$\xrightarrow{B}$	(A)	$q=1/2$	Reinforcement	$p_B<1/2$	(A,B)	(A)
				Learning	$p_B\geq 1/2$	(A,B)	(A,B)
(A,B)	$\xrightarrow{A}$	(B)	$q=1/2$	Reinforcement	$p_A<1/2$	(A,B)	(B)
				Learning	$p_A\geq 1/2$	(A,B)	(A,B)
(A,B)	$\xrightarrow{B}$	(B)	$q=1/2$	Agreement	always	(B)	(B)
(A,B)	$\xrightarrow{A}$	(A,B)	$q=1/2$	Agreement	always	(A)	(A)
(A,B)	$\xrightarrow{B}$	(A,B)	$q=1/2$	Agreement	always	(B)	(B)

The speaker (S) conveys a name $\xrightarrow{A}$ or $\xrightarrow{B}$ to the hearer (H) together with an example taken from the speaker's inventory, $[+\ +\ +\ldots]_A$ or $[+\ +\ +\ldots]_B$, respectively—this happens with a branching probability $q=0.5$ if the speaker has the list (A,B) and knows the meaning of both names. The outcome can be: (1) reinforcement (only); (2) generalization of concept C if the Bayes probability is $p\geq 1/2$; (3) an agreement between hearer and speaker if both agents know the meaning of the conveyed name. Even if not indicated, reinforcement takes place also in cases (2) and (3).

inventories associated with the names A and B: **Figure 3** shows the histograms of the p_As and p_Bs computed from the initial time until consensus for a single run with $N = 2000$ agents and starting with SIC. The low frequencies at small values of p_A and p_B and the highest frequencies at values close to unity are due to the fact that the Bayesian probabilities reach values $p_A \approx p_B \approx 1$ very fast, after a few learning attempts, consistently with the size principle, on which the Bayesian learning paradigm, and in turn Equation (4), are based [10].

In order to visualize how the system approaches consensus, it is useful to consider some global observables, such as the fractions $n_A(t)$, $n_B(t)$, and $n_{AB}(t)$ of agents that have generalized concept C in association with name A only, name B only, or both names A and B, respectively, or the success rate $S(t)$. The dynamics of a population of $N = 1,000$ agents (**Figures 4A,B**) using different initial conditions, SIC, AIC, and AICr, and that of a population of $N = 100$ agents starting with SIC (**Figures 4C,D**) are shown in **Figure 4**.

Figure 4A shows only the population fractions corresponding to the name found at consensus, for the sake of clarity (the remaining population fractions eventually go to zero). For an asymmetrical initial condition (AIC or AICr), it is the initial majority that determines the convention found at consensus (that is, B for AIC and A for AICr). If the system starts from SIC, convention A, for which agents can generalize earlier ($n^*_{ex,A} = 5 < n^*_{ex,B} = 6$), is always found at consensus—in this case, it is the asymmetry in the thresholds $n^*_{ex,A}$ and $n^*_{ex,B}$, characterizing the Bayesian learning process, that determine consensus.

Figure 4B shows the success rate $S(t = t_k)$, representing the average over different runs of the instantaneous success rate S_k of the kth interaction at time t_k, defined as follows: $S_k = 1$ in case of agreement between the two agents or when successful learning by the hearer takes places, following a Bayes probability $p \geq 1/2$; or $S_k = 0$ in case of unsuccessful generalization, when $p < 1/2$ and only reinforcement takes place. The success rate $S(t)$ varies between $S(0) \approx (n_A(0))^2 + (n_B(0))^2$, due to the respective fractions of agents that initially know the two conventions A and B, to $S \approx 1$ at consensus, following a typical S-shaped curve of learning processes [41]. In the case of SIC, the initial value is $S(0) \approx 0.5^2 + 0.5^2 = 0.5$, while for AIC or AICr the initial value is $S(0) \approx (0.3)^2 + (0.7)^2 \approx 0.58$.

We now investigate how the modified Bayesian dynamics affects the convergence times to consensus. The study of the size-dependence of the convergence to consensus shows that there is a critical value $N^* \approx 500$ in the case of SIC, such that for $N \leq N^*$, there is a non-negligible probability that the final absorbing state is B. **Figures 4C,D**, representing the results for a system starting with SIC and a smaller size $N = 100$, show the existence of two possible final absorbing states, and that there are different timescales associated with the convergence to consensus: name A is found at consensus in about 90% of cases and name B in the remaining cases. The branching probability into A or B consensus is further investigated in **Figure 5A**, where we plot the branching probabilities $p_{e,A}, p_{e,B}$ vs. the system sizes N. The nonlinear behavior (symmetrical sigmoid) signals the presence of finite-size effects, which are particularly clear for relatively small N-values. In fact, when the fluctuations in the system are larger,

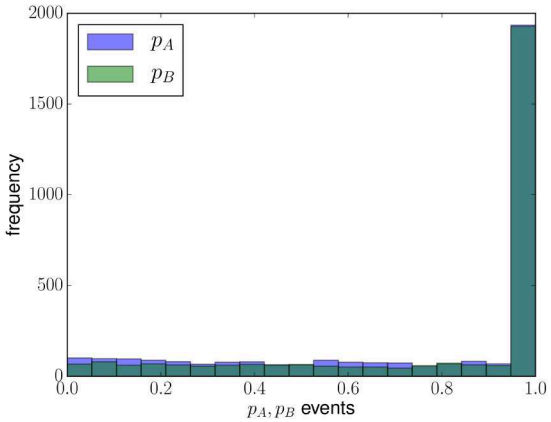


FIGURE 3 | Histograms of the Bayesian probabilities p_A, p_B computed by agents during their learning attempts during a single run (for $N = 2000$ agents, starting with SIC; $n^*_{ex,A} = 5$, $n^*_{ex,B} = 6$).

the system size can play an important role in the dynamics of social systems, as an actual thermodynamic limit is only allowed for simulations of macroscopic physical systems [42].

The convergence time t_{conv} follows a simple scaling rule with the system size N, related to the average number of examples $\bar{n}_{ex,A}, \bar{n}_{ex,B}$ relative to A, B respectively, stored in the agents' inventories at consensus. These values depend on the number of learning and reinforcement processes, and hence are related to the system size N. The average number of interactions undergone by the agents until the system reaches the consensus is given by the sum $\bar{n}_{int} = \bar{n}_{ex,A} + \bar{n}_{ex,B}$[1]. One expects that:

$$t_{conv} \approx \bar{n}_{int} N \,, \tag{5}$$

which suggests a linear scaling law ($t_{conv} \sim N$) for convergence time with the system size N for all the possible initial conditions. Linear behavior is indeed confirmed by the numerical simulations with population sizes $N = 50$, 100, 500, 1,000, 1,500, and 2,000 starting from SIC, AIC, and AICr. The relative numerical results are reported in **Table 2**. Moreover, in Equation (5) the size-dependence of $\bar{n}_{int}$ is ignored as it shows a weak dependence upon N; see **Figure 5B**.

From the above-mentioned scaling law, it is clear that the average number of examples stored by the agents at consensus plays an important role in the semiotic dynamics. In particular, it is found that if the final absorbing state is A (or B), then $\bar{n}_{ex,A} > \bar{n}_{ex,B}$ ($\bar{n}_{ex,B} > \bar{n}_{ex,A}$). Moreover, the average number of examples, relative to the absorbing state, always increases monotonically with the system size while a size-independent behavior is observed in the opposite case; see **Figure 5B**.

Finally, we compare the convergence time of the Bayesian word-learning model, t_{conv}, with that of the two-conventions NG model, $\bar{t}_{conv}$ [36] by studying the corresponding ratio

[1]The $n_{ex,A} = n_{ex,B} = 4$ examples given initially to each agent are not accounted for by $\bar{n}_{ex,A}$ and $\bar{n}_{ex,B}$.

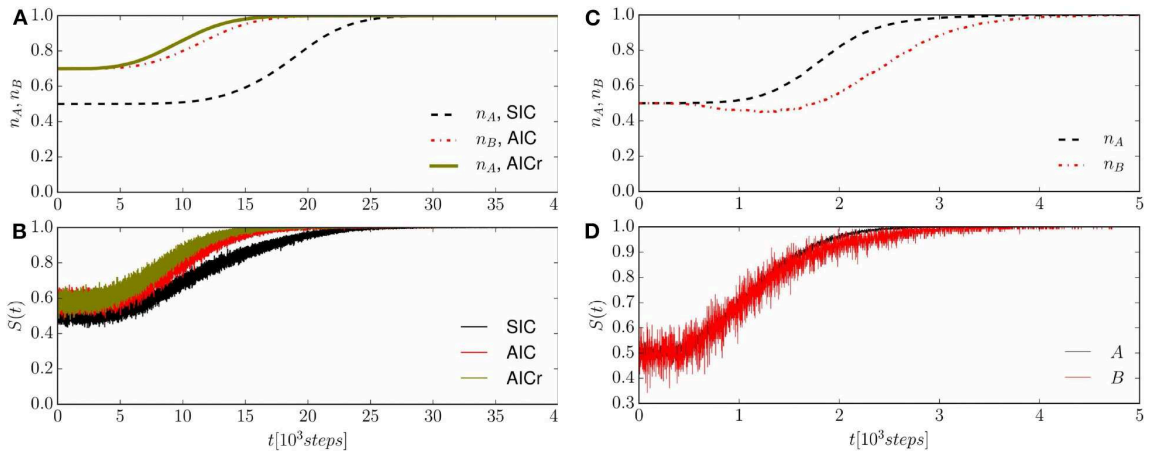


FIGURE 4 | Average population fraction associated to the name shared in the final consensus state **(A,C)** and success rate $S(t)$ **(B,D)** vs. time. **(A,B)** System with $N = 1,000$ agents starting from different initial conditions, SIC, AIC, and AICr; averages obtained over 600 runs. **(C,D)** System with $N = 100$ agents starting from SIC; averages obtained over 1,000 runs—notice that due to the smaller size $N = 100$, the system can converge to consensus both with name A (in a fraction of cases $p_{e,A} \approx 0.9$) and with name B ($p_{e,B} \approx 0.1$).

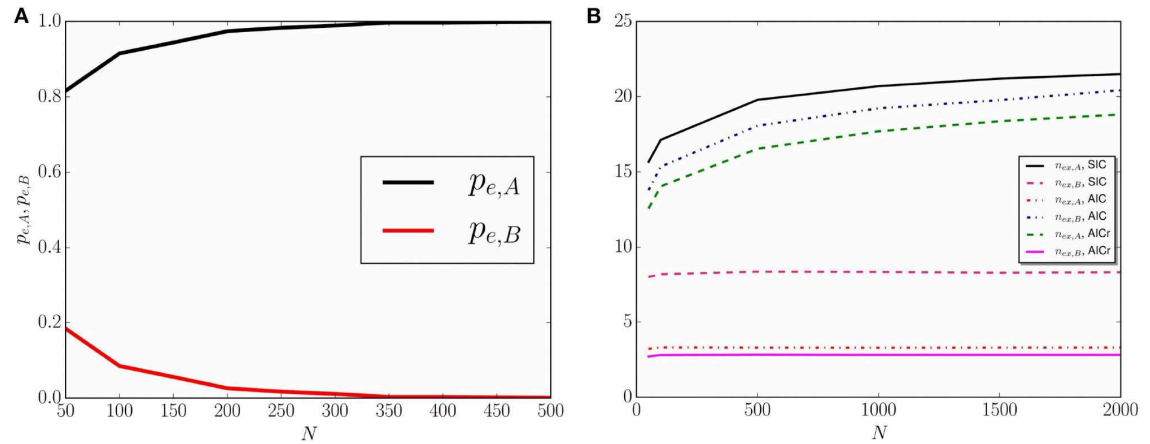


FIGURE 5 | (A) Probabilities $p_{e,A}$ and $p_{e,B}$ that the system reaches consensus at A and B respectively, vs. the system size N, obtained by averaging over 1,000 runs of a system starting with SIC. **(B)** Average number of examples $\bar{n}_{ex,A}$ and $\bar{n}_{ex,B}$ recorded by an agent at consensus for a system of $N = 50$, 100, 500, 1,000, 1,500, or 2,000 agents, starting with SIC, AIC, and AICr. Averages are obtained over 600 runs.

$R = t_{conv}/\bar{t}_{conv}$ for common initial conditions and population sizes. When starting with SIC, the values of the convergence times obtained from the two models become of the same order by increasing N: R decreases with N, reaching unity for $N = 10,000$; see **Figure 6**. In other words, the time scales of the two models become equivalent for relatively large system sizes, i.e., the learning processes of the two models perform equivalently and the Bayesian approach roughly gives rise to the one-shot learning that characterizes the two-conventions NG model. In the next section, we discuss how the Bayesian model becomes asymptotically equivalent to the minimal NG model. The inset of **Figure 6** represents R vs. N for $N < 2000$, given different starting configurations, with SIC, AIC, and AICr, and different population sizes.

4. STABILITY ANALYSIS

In this section, we investigate the stability and convergence properties of the mean-field dynamics of the Bayesian NG model, in which statistical fluctuations and correlations are neglected.

4.1. Mean-Field Equations

In the Bayesian NG model, as in the basic NG, agents can use two non-excluding options A and B to refer to the same concept C. The main difference between the Bayesian model and the basic NG model is in the learning process: a one-shot learning process in the basic NG and a Bayesian process in the Bayesian NG model. In the latter case, the presence of a name in the word list indicates that the agent has generalized the corresponding concept from a set of positive recorded examples.

TABLE 2 | Scaling laws $t_{conv} \sim N^{\alpha}$ with the system size N.

	α	$\bar{n}_{ex,A}$	$\bar{n}_{ex,B}$	Outcome
SIC	1.06	20	8	A, B
AIC	1.08	3	19	B
AICr	1.09	18	3	A

Here the parameters are $n^*_{ex,A} = 5$, $n^*_{ex,B} = 6$ with initial conditions SIC, AIC, and AICr. The average number of examples, $\bar{n}_{ex,A}, \bar{n}_{ex,B}$, stored at t_{conv}, are obtained by averaging over 600 runs of a system with $N = 1{,}000$ agents.

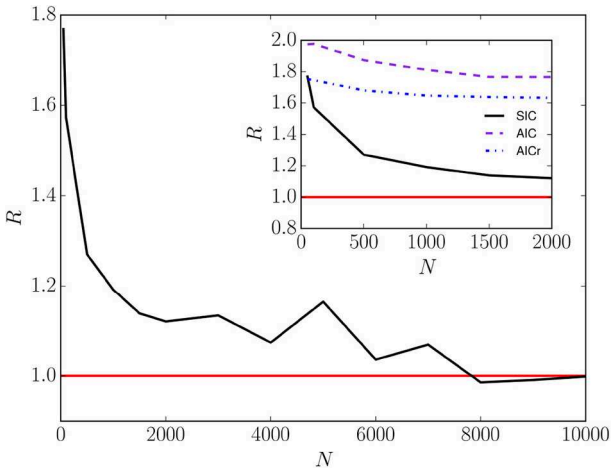


FIGURE 6 | The ratio of the convergence times of the Bayesian word-learning model and the two-conventions NG model, $R = t_{conv}/\bar{t}_{conv}$ vs. the system size N for a system starting with SIC. The inset illustrates the dependence of R on different initial conditions. The curves are obtained by averaging over 900 runs.

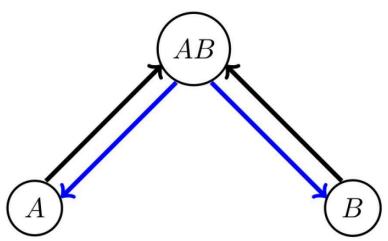


FIGURE 7 | Model scheme with two non-excluding options. Arrows indicate allowed transitions between the "bilingual" state (A,B) and the "monolingual" states A and B. Direct $A \leftrightarrow B$ transitions are not allowed.

The NG model belongs to the wide class of models with two non-excluding options A and B, such as many models of bilingualism [43], in which transitions between state (A) and state (B) are allowed only through an intermediate ("bilingual") state (A, B), as schematized in **Figure 7**. The mean-field equations for the fractions $n_A(t)$ and $n_B(t)$ can be obtained by considering the gain and loss contributions of the transitions depicted in **Figure 7**,

$$\dot{n}_A = p_{AB\to A}\, n_{AB} - p_{A\to AB}\, n_A\,,$$

$$\dot{n}_B = p_{AB\to B}\, n_{AB} - p_{B\to AB}\, n_B\,. \tag{6}$$

Here, $\dot{n}_a(t) = dn_a(t)/dt$ and the quantities $p_{a\to b}$ represent the respective transition rates per individual, corresponding to the arrows in **Figure 7** ($a, b = A, B, AB$). The equation for $n_{AB}(t)$ was omitted, since it is determined by the condition that the total number of agents is constant, $n_A(t) + n_B(t) + n_{AB}(t) = 1$.

The details of the possible pair-wise interactions in the Bayesian naming game are listed in **Table 1**. By adding the various contributions, one obtains the equation for the average population fractions,

$$\begin{aligned}\dot{n}_A &= -p_B n_A n_B + n_{AB}^2 + \frac{3-p_B}{2} n_A n_{AB}\,,\\ \dot{n}_B &= -p_A n_A n_B + n_{AB}^2 + \frac{3-p_A}{2} n_B n_{AB}\,,\end{aligned} \tag{7}$$

which can be rewritten in the form (6) with transition rates per individual given by:

$$p_{A\to AB} = p_B n_B + \frac{1}{2} p_B n_{AB}\,, \qquad p_{B\to AB} = p_A n_A + \frac{1}{2} p_A n_{AB}\,, \tag{8}$$

$$p_{AB\to A} = \frac{3}{2} n_A + n_{AB}\,, \qquad p_{AB\to B} = \frac{3}{2} n_B + n_{AB}\,. \tag{9}$$

Equations (8) provide the transition rates of *learning* processes, while Equations (9) give the transition rates of *agreement* processes.

In the rest of the paper, we set $x \equiv n_A$, $y \equiv n_B$, and $z = n_{AB} \equiv 1 - x - y$, so that the autonomous system (7) becomes:

$$\dot{x} = f_x(x,y) \equiv -p_B xy + (1-x-y)^2 + \frac{1}{2}(3-p_B)x(1-x-y)\,, \tag{10}$$

$$\dot{y} = f_y(x,y) \equiv -p_A xy + (1-x-y)^2 + \frac{1}{2}(3-p_A)y(1-x-y)\,, \tag{11}$$

in which we have defined the velocity field $\mathbf{v} = \big(f_x(x,y), f_y(x,y)\big)$ in the phase plane. The Bayesian probabilities p_A and p_B appear in these equations as time-dependent parameters of the model, but they are actually highly non-linear functions of the variables. In fact, they can be thought as averages of the microscopic Bayesian probability in Equation (4) over the possible dynamical realizations. For this reason, they have also a complex non-local time-dependence on the previous history of the interactions between agents. For the moment, we assume $p_A(t) = p_B(t) = p(t)$, returning later to the general case.

From the conditions defining the critical points, $f_x(x,y) = f_y(x,y) = 0$, one obtains $(x-y)z = 0$. Setting $z = 0$, one obtains two solutions that correspond to consensus in A or B, given by $(x_1, y_1, z_1) = (1, 0, 0)$ and $(x_2, y_2, z_2) = (0, 1, 0)$. Instead, setting $(x-y) = 0$ leads to the equation:

$$2x^2 - (p+5)x + 2 = 0\,, \tag{12}$$

which has the solutions,

$$x_{\pm} = \frac{p + 5 \pm \sqrt{(p+5)^2 - 16}}{4}. \tag{13}$$

One can check that for every value of $p \in (0, 1]$, the corresponding solutions $(x_{\pm}, x_{\pm}, 1 - 2x_{\pm})$ are not suitable solutions, because $z_{\pm} = 1 - 2x_{\pm} < 0$.

This analysis is valid for $p > 0$. In fact, $p = p(t)$ is a function of time and, for a finite interval of time after the initial time, one has that $p = 0$, which defines a different dynamical system: the transition from $p = 0$ to $p > 0$ is accompanied by a bifurcation, as becomes clear by analyzing the equilibrium points. In the initial conditions used, $z(0) = 0$, which implies $z(t) = 0$, $x(t) = x(0)$, and $y(t) = y(0)$ at any later time t as long as $p(t) = 0$, since $\dot{x}(t) = \dot{y}(t) = \dot{z}(t) = 0$ (see Equation 7); in fact, the whole line $x + y = 1$ (for $0 < x, y < 1$) represents a continuous set of equilibrium points. The reason why, in this model, $p(0) = 0$ at $t = 0$ and also during a subsequent finite interval of time is twofold. First, agents do not have any examples associated to the name not known, and they have to receive at least $n^*_{ex,A}$ or $n^*_{ex,B}$ examples before being able to compute the corresponding Bayesian probability $p_A(t)$ or $p_B(t)$—thus, it is to be expected that $p(t) = 0$ meanwhile. Furthermore, even when agents can compute the Bayesian probabilities, the effective probability to generalize is actually zero, due to the threshold $p^* = 0.5$ for a generalization to take place. The existence of the (temporary) equilibrium points on the line $x + y = 1$ ends as soon as the parameter $p(t) > p^*$ and, according to Equation (7), a bifurcation takes place: the two A- and B-consensus states become the only stable equilibrium points, and the representative point in the x-y-plane is deemed to leave the initial conditions on the $z = 1 - x - y = 0$ line due to the stochastic nature of the dynamics, which is not invariant under time reversal [44].

To determine the nature of the critical points $(x_1, y_1) = (1, 0)$ and $(x_2, y_2) = (0, 1)$, one needs to evaluate at the equilibrium points the 2×2 Jacobian matrix $A(x, y) = \{\partial_i f_j\}$, where $i, j = x, y$. Equations (10, 11) give:

$$A(1,0) = \begin{pmatrix} \frac{1}{2}(p-3) & -\frac{1}{2}(p+3) \\ 0 & -p \end{pmatrix}, A(0,1) = \begin{pmatrix} -p & 0 \\ -\frac{1}{2}(p+3) & \frac{1}{2}(p-3) \end{pmatrix}, \tag{14}$$

whose eigenvalues at a given time t are $\lambda_1 = [p(t) - 3]/2$ and $\lambda_2 = -p(t)$. As they are both negative and distinct for $0 < p \leq 1$, $\lambda_1 < \lambda_2 < 0$, the critical points $(0, 1)$ and $(1, 0)$ are asymptotically stable [45]. It can be easily checked that these conclusions are unchanged if the generalization probabilities are different, $p_A(t) \neq p_B(t)$. For instance, one would have:

$$A(1,0) = \begin{pmatrix} \frac{1}{2}(p_B - 3) & -\frac{1}{2}(p_B + 3) \\ 0 & -p_A \end{pmatrix}, \tag{15}$$

associated to eigenvalues with different numerical values but the same sign, not changing the nature of the critical point. Thus, the asymptotically stable nodes $(x_1, y_1) = (1, 0)$ and $(x_2, y_2) = (0, 1)$ are the only absorbing states of the Bayesian naming game.

4.2. A Geometric Analysis of Consensus

We consider a system starting from SIC, defined by the point $(x_0, y_0) = (0.5, 0.5)$ located on the line $x + y = 1$ of the phase plane, representing a system that initially has 50% of agents with list (A), 50% with list (B), and no agent with list (A, B).

From Equation (7) and the fact that, initially, $p_A(0) = p_B(0) = 0$ (and $z(0) = 0$), one can see that the corresponding velocity is $\mathbf{v}(0) = (f_x(x(0), y(0)), f_y(x(0), y(0))) = (0, 0)$, meaning that the initial SIC state $(x(0), y(0)) = (0.5, 0.5)$ is a temporary equilibrium point. As soon as $p_A(t), p_B(t) > 0$, at a time $t = t^* > 0$, the velocity becomes different from zero, and the representative point $(x(t), y(t))$ moves away in the phase plane with velocity $\mathbf{v}(t) = (f_x(x(t), y(t)), f_y(x(t), y(t)))$. The system is observed to eventually reach either A-consensus at $(x_1, y_1) = (1, 0)$ or B-consensus at $(x_2, y_2) = (0, 1)$. These two types of evolution are illustrated in **Figure 8** through the population fractions $n_A(t)$ and $n_B(t)$ vs. time t taken from two single runs of a population with size $N = 100$ agents.

In order to determine the conditions for this to happen, we evaluate the scalar product between the velocity $\mathbf{v}(t^*)$ at time t^* and the versor $\mathbf{u} = (1/\sqrt{2}, -1/\sqrt{2})$, parallel to the line $x + y = 1$ and directed toward the A-consensus state $(1, 0)$; see **Figure 9**. If the velocity vector $\mathbf{v}$ has a positive component along $\mathbf{u}$, the representative point will move from the initial state $(0.5, 0.5)$ toward the A-fixed point $(1, 0)$; instead, in the case of a velocity vector $\mathbf{v}'$ with a negative component along $\mathbf{u}$, the representative point will move toward the B-fixed point $(1, 0)$; see **Figure 9**. From the simulations, we know that during the initial transient, the population dynamics is characterized by a $n_A(t) \approx n_B(t)$, until the critical time $t = t^*$ is reached. This allows one to set $x \approx y$ in this interval of time—this phenomenon is a sort of stiffness of the system before starting to explore the phase plane. The scalar product $\mathbf{u} \cdot \mathbf{v}$, where the velocity vector's components are given explicitly in Equation (7), is positive at $t = t^*$ when:

$$\mathbf{u} \cdot \mathbf{v} = \frac{f_x - f_y}{\sqrt{2}} = \frac{1}{\sqrt{2}} \left[p_A(t^*) - p_B(t^*) \right] x^2 + \frac{1}{2\sqrt{2}} \left[p_A(t^*) - p_B(t^*) \right] x(1 - 2x) > 0, \tag{16}$$

which gives the condition:

$$p_A(t^*) > p_B(t^*). \tag{17}$$

Equation (17) clearly shows that the values of the Bayesian probabilities p_A, p_B become different at the critical time t^*, thus allowing the solutions to split into different orbits, going toward the state of consensus in A and B. The same reasoning, applied to the case of an orbit bending toward the consensus state in B at $(0, 1)$, would give $p_A(t^*) < p_B(t^*)$. The difference between the probabilities p_A and p_B can be traced back to the way the generalization function $p(t)$ is used by agents to compute either $p_A(t)$ or $p_B(t)$. In fact, the value of $p(t)$ depends on the examples recorded, which constitute the inputs of the function. We argue that this behavior is due to the fluctuations of the numbers of examples $\bar{n}_{ex,A}(t)$ and $\bar{n}_{ex,B}(t)$, recorded by the agents until time t, together with the initial asymmetry of the thresholds

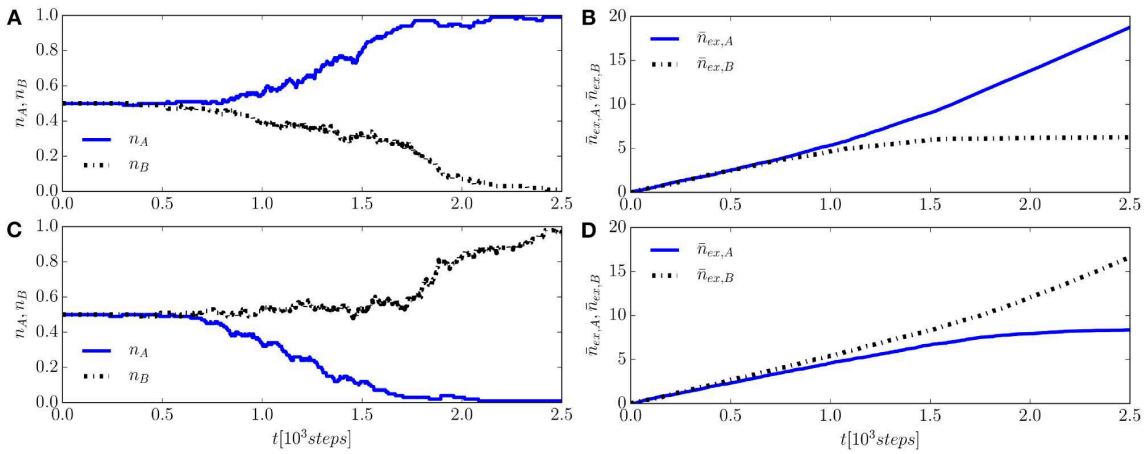


FIGURE 8 | Results from two single simulations of a system with $N = 100$ agents, starting from SIC at $(x_0, y_0) = (0.5, 0.5)$ and reaching two different consensus states about name A or B. **(A,C)** Show the population fractions $x(t) = n_A(t)$ and $y(t) = n_B(t)$. **(B,D)** Show the corresponding average number of examples recorded by an agent, $\bar{n}_{ex,A}(t)$ and $\bar{n}_{ex,B}(t)$.

for generalizing, $n^*_{ex,A} \neq n^*_{ex,B}$. In fact, the stochastic nature of the pairwise-interactions leads to different examples (that can be better or worse for the aim of generalizing) and to different inventory sizes, i.e., the numbers of examples stored by agents at time t; clearly all this strongly affects the path to consensus. Furthermore, asymmetrical thresholds ($n^*_{ex,A} = 5 < n^*_{ex,B} = 6$ were used) produce a bias favoring consensus in A and play a crucial role in the subsequent Bayesian semiotic dynamics, letting concept C be learned more often in association with A than B and contributing to making consensus in A more frequent: swapping the threshold values (setting $n^*_{ex,A} = 6 > n^*_{ex,B} = 5$), the approach to consensus occurs with the outcomes A, B swapped.

For $N \gtrsim N^* \approx 500$, the chances that the system converges to (B) become negligible. This can be seen in of **Figures 8B,D**, showing $\bar{n}_{ex,A}(t)$ and $\bar{n}_{ex,B}(t)$ vs. time (averaged over the agents of the system) for a single run of a system with a population of $N = 100$ agents, starting with SIC. Panels (B) and (D) compare the results obtained from selected runs ending at consensus A and B, respectively. It is evident that, after an initial transient, in which $\bar{n}_{ex,A}(t) \approx \bar{n}_{ex,B}(t)$, they start to differ more and more significantly from each other at times $t > t^*$. In turn, starting from this point, also p_A and p_B begin to differ significantly from each other, thus affecting the rate of depletion of the populations during the subsequent dynamics. For instance, if $p_A > p_B$, then $p_{B\to AB} > p_{A\to AB}$ (see Equations (8)), which means that the depletion of n_B occurs faster then that of n_A. In turn, this favors the decay of the mixed states (A, B) into the state (A) (see Equations (9)), being $n_A > n_B$.

The asymmetry discussed above, about the rate of convergence toward the final consensus states, also affects the values of convergence times t^A_{conv} and t^B_{conv} needed for a system to reach consensus at A and B, respectively: we find $t^B_{conv} > t^A_{conv}$ in all the numerical simulations. The difference in the convergence times is already appreciable, despite the noise, in the output of a single run, such as the population fractions shown in **Figures 8A,C**. Mean fractions $n_A(t)$, $n_B(t)$, and $n_{AB}(t)$ vs. time, obtained by averaging over many runs, result in less noisy outputs and provide a more clear picture of the difference, which is visible in **Figure 10**, obtained using 600 runs starting with SIC and for $N = 100$ agents (**Figures 10A,C**) and $N = 200$ agents (**Figures 10B,D**). In addition, one can notice that the convergence times strongly depend on the system size: increasing the number of agents N slows down the relaxation, and both the times t^A_{conv} and t^B_{conv} increase, as is evident by comparing the (**Figures 10A,C**) ($N = 100$ agents) with the (**Figures 10B,D**) ($N = 200$ agents).

The possibility that a system starting with the same initial conditions and with the same parameters can reach both consensus states is a consequence of the stochastic nature of the pairwise-interactions, together with the asymmetry in the threshold values $n^*_{ex,A} = 5$ and $n^*_{ex,B} = 6$. It stops occurring for $N \gtrsim N^*$, when both $\bar{n}_{ex,A}$ and $\bar{n}_{ex,B}$ reach some threshold values close to those observed at t_{conv}, which is clearly a value sufficient for the agents to generalize concept C. In fact, the scaling law of t_{conv} with N shows that the sum of $\bar{n}_{ex,A}$ with $\bar{n}_{ex,B}$ becomes nearly constant for $N \gtrsim N^*$, implying that the dynamics is uniquely determined, that is, the consensus always occurs at A from SIC, once the agents have stored a threshold number of $\bar{n}_{ex,A}$, $\bar{n}_{ex,B}$. It is found that these threshold values correspond to $\bar{n}_{ex,A} = 21$, $\bar{n}_{ex,B} = 12$. Note that, to the latter values $\bar{n}_{ex,A}$, $\bar{n}_{ex,B}$, we also add the four initial given examples stored in the agents' inventories at the beginning. This is because the generalization function $p(t)$ outputs will effectively depend on them all. Therefore, at these threshold values, it would be very unlikely that $p_B > p_A$, and so it would be the same for the consensus at B.

Now, we consider some variables that characterize the Bayesian process underlying pair-wise interactions and how they vary with time, in particular the Bayesian probabilities $p_A(t)$ and $p_B(t)$, computed by agents, and the corresponding number of learning attempts $no_A(t)$ and $no_B(t)$ made by agents at time t to learn concept C in association with word A or B, respectively, i.e., the number of times that the agents compute p_A or p_B. Only the

case of a system starting with SIC is considered, but the other cases present similar behaviors. We consider a single run of a system with $N = 5,000$ agents and study the average values $\bar{p}_A(t), \bar{p}_B(t)$ obtained by averaging $p_A(t)$ and $p_B(t)$ over the agents of the system. Furthermore, employing a coarse-grained view, an additional average (of both the probabilities $\bar{p}_A(t), \bar{p}_B(t)$ and the numbers of attempts no_A, no_B) over a suitable time-interval (a temporal bin $\Delta t = 16 \times 10^3$) reduces random fluctuations. **Figure 11** shows the time evolution of the average probabilities $\bar{p}_A(t)$ and $\bar{p}_B(t)$ in the time-range where data allow good statistics. The probabilities grow monotonically and eventually reach the value one. While this points at an equivalence between the mean-field regime of the Bayesian naming game and that of the two-conventions NG model, in which agents learn at the first attempt (one-shot learning), such an equivalence is suggested but not fully reproduced by the coarse-grained analysis. The time evolution of the number of learning attempts $no_A(t)$ and $no_B(t)$ shows that they are negligible both at the beginning and at the end of the dynamics—see inset in **Figure 11**. This is due to the fact that at the beginning it is most likely that either interactions between agents with the same conventions take place (starting with SIC, each agent has a probability of 50% to interact with an agent having the same convention) or that interactions between agents with different conventions but with still too small inventories to be able to generalize concept C take place, leading to reinforcement processes only. When approaching

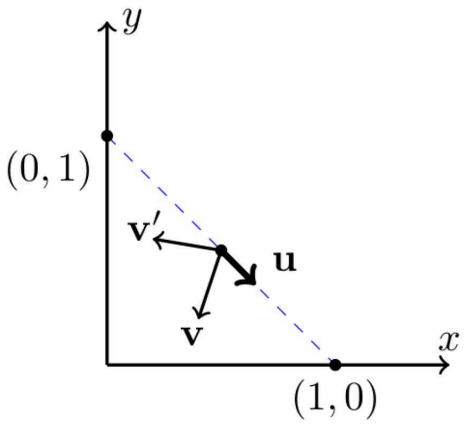


FIGURE 9 | The vectors **v** and **v′** represent two possible velocities emerging from the point $(x_0, y_0) = (0.5, 0.5)$ at $t = t^*$. The unit vector **u** is directed along the line $x + y = 1$ joining the asymptotically stable nodes $(x_1, y_1) = (1, 0)$ and $(x_2, y_2) = (1, 0)$.

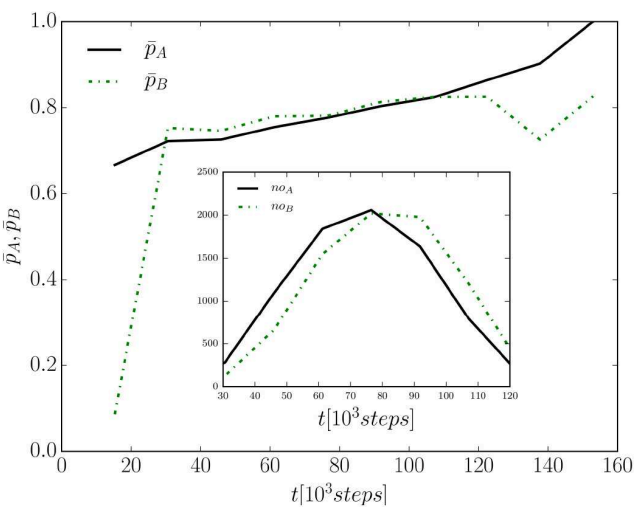


FIGURE 11 | Average values $\bar{p}_A(t)$ and $\bar{p}_B(t)$ computed using a (temporal) bin $\Delta t = 16 \times 10^3$ vs. time from for a single run of a system reaching consensus at A. The convergence time is $t_{conv} \approx 160 \times 10^3$, and the population size is $N = 5,000$. The inset shows the average number of learning attempts no_A, no_B vs. time for the same single run.

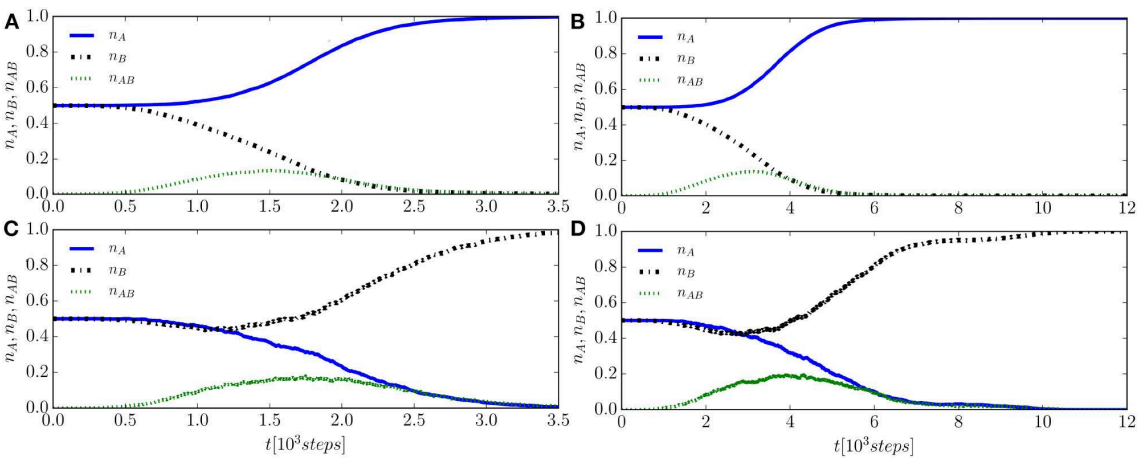


FIGURE 10 | Population fractions $n_A(t)$, $n_B(t)$, and $n_{AB}(t)$ vs. time, starting from SIC; results are obtained by averaging over 600 runs. Left column **(A,C)**: a system with $N = 100$ agents can reach consensus with name A **(A**, about 91% of runs) or name B **(C**, about 9% of runs). Right column **(B,D)**: system with $N = 200$ agents, reaching consensus with name A in about 96% of runs **(B)** and with name B in about in the remaining 4% of runs **(D)**.

consensus, agents with one of the conventions constitute the large majority of the population, and thus they are again most likely to interact through reinforcements only. Thus, the largest numbers of attempts to learn concept C in association with A and B are expected to occur at the intermediate stage of the dynamics. In fact, $no_A(t)$ and $no_B(t)$ are observed to reach a maximum at $t \approx t_{conv}/2$ for any given system size N, as is visible in the inset of **Figure 11**. Notice that also the fraction of agents n_{AB} who know both conventions and can communicate using both name A and name B, possibly allowing other agents to generalize in association with name A or B, reaches its maximum roughly at the same time.

5. CONCLUSION

We constructed a new agent-based model that describes the appearance of linguistic consensus through a word-learning process, representing an original example of an opinion dynamics or culture competition model translated at a cognitive level, something that is not apparent. The model represents a Bayesian extension of the semiotic dynamics of the NG model, with an underlying cognitive process that mimics the human learning processes; it can describe in a natural way the uncertainty accompanying the first phase of a learning process, the gradual reduction of the uncertainty as more and more examples are provided, and the ability to learn from a few examples.

The work presented is exploratory in nature, concerning the minimal problem of a concept, C that can be associated to two different possible names A and B. The resulting semiotic dynamics of the synonyms is different from the basic NG, in that it depends on parameters that are strictly cognitive in nature, such as a minimum level of experience (quantified by the number of examples n^*_{ex} necessary for generalizing) and the threshold for generalizing a concept (represented by a critical value of the Bayesian acceptance probability p^*). The interplay between the asymmetry of the conventions A and B, the system size, and the stochastic character of the time evolution have dramatic consequences on the consensus dynamics, leading to a critical time $t^* > 0$ before the system begins to move in the phase-plane, to converge eventually toward a consensus state; a critical system size N^*, below which there is an appreciable probability that the system can end up in any of the two possible consensus states and, in general, a dependence of the convergence times on N; an asymmetry in the convergence times and the corresponding branching probabilities that the system converges toward one of the two possible conventions; different scaling of the convergence times vs. N with respect to those observed in the basic NG model, due to the dependence on the learning experience of the agents.

The cognitive dimension of the novel model offers the possibility to study the effects that are out of the reach of other opinion dynamics or cultural exchange models, such as the basic NG model. The corresponding dynamical equations, Equations (10, 11), provide a general mean-field description of a group of individuals communicating with each other while undergoing cognitive processes. The cognitive dynamics are fully contained in the functions $p_A(t)$ and $p_B(t)$. Similar models but with different or more general underlying cognitive dynamics are expected to leave the form of Equations (10, 11) unchanged, only changing the functional forms of $p_A(t)$ and $p_B(t)$. In this sense, the model introduced in this work represents a step toward a generalized Bayesian approach to the problem of how social interactions can lead to cultural conventions.

Future work can address specific problems of current interest from the point of view of cognitive processes or features relevant from the general standpoint of complexity theory. In the first case, it is possible to study the semiotic dynamics of homonyms and synonyms, e.g., the problem of a name A_1, associated to a concept C_1, that at some points splits into two related but distinct concepts C_1 and C_2, analyzing the cognitive conditions for the corresponding splitting of name A_1 into two names A_1 and A_2, as the two concepts eventually become distinguishable to the agents—this type of problem cannot be tackled within models of cultural competition. In the second case, one can mention the classical problem of the interplay between a central information source (bias) and the local influences of individuals—this time, in a cognitive framework—and the role of heterogeneity. In fact, heterogeneity concerns most of the known complex systems at various levels, from the diversity in the dynamical parameters of, e.g., the different competing names and concepts to that of the agents. The heterogeneity of individuals is known to lead to counter-intuitive effects, such as resonant behaviors [46, 47]. Furthermore, the complex, heterogeneous nature of a local underlying social network can drastically change the co-evolution of the conventions in competition with each other and therefore the relaxation process [48].

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation, to any qualified researcher.

AUTHOR CONTRIBUTIONS

GM wrote the codes and performed the numerical simulations. GM, MP, and EH contributed to the design of the numerical experiments, the analysis of the results, and the writing of the manuscript.

FUNDING

The authors acknowledge support from the Estonian Ministry of Education and Research through Institutional Research Funding IUT (IUT39-1), the Estonian Research Council through Grant PUT (PUT1356), and the ERDF (European Development Research Fund) CoE (Center of Excellence) program through Grant TK133.

ACKNOWLEDGMENTS

We thank Andrea Baronchelli for providing useful remarks about the naming game model and the manuscript.

REFERENCES

1. Castellano C, Fortunato S, Loreto V. Statistical physics of social dynamics. *Rev Mod Phys.* (2009) **81**:591. doi: 10.1103/RevModPhys.81.591
2. Baronchelli A. The emergence of consensus: a primer. *R Soc Open Sci.* (2018) **5**:172189. doi: 10.1098/rsos.172189
3. Xia C, Ding S, Wang C, Wang J, Chen Z. Risk analysis and enhancement of cooperation yielded by the individual reputation in the spatial public goods game. *IEEE Syst J.* (2017) **11**:1516–25. doi: 10.1109/JSYST.2016.2539364
4. Xia C, Li X, Wang Z, Perc M. Doubly effects of information sharing on interdependent network reciprocity. *N J Phys.* (2018) **20**:075005. doi: 10.1088/1367-2630/aad140
5. Zhang Y, Wang J, Ding C, Xia C. Impact of individual difference and investment heterogeneity on the collective cooperation in the spatial public goods game. *Knowledge Based Syst.* (2017) **136**:150–8. doi: 10.1016/j.knosys.2017.09.011
6. Mufwene S. *The Ecology of Language Evolution.* Cambridge: Cambridge University Press (2001).
7. Lass R. *Historical Linguistics and Language Change.* Cambridge: Cambridge University Press (1997).
8. Berruto C. *Prima lezione di sociolinguistica*. Roma: Laterza (2004).
9. Edelman S, Waterfall H. Behavioral and computational aspects of language and its acquisition. *Phys Life Rev.* (2007) **4**:253–77. doi: 10.1016/j.plrev.2007.10.001
10. Tenenbaum JB. *A Bayesian framework for concept learning.* Ph.D. thesis, Boston, MA: MIT (1999).
11. Wichmann S. The emerging field of language dynamics. *Lang Linguist Compass.* (2008) **3**:442. doi: 10.1111/j.1749-818X.2008.00062.x
12. Wichmann S. Teaching & learning guide for: The emerging field of language dynamics. *Lang Linguist Compass.* (2008) **2**:1294–7. doi: 10.1111/j.1749-818X.2008.00109.x
13. Tenenbaum JB, Xu F. Word learning as Bayesian inference. In: *Proceedings of the 22nd Annual Conference of the Cognitive Science Society* Mahwah, NJ (2000).
14. Xu F, Tenenbaum JB. Word learning as Bayesian inference. *Psychol Rev.* (2007) **114**:245–72. doi: 10.1037/0033-295X.114.2.245
15. Hurford J. Biological evolution of the saussurean sign as a component of the language-acquisition device. *Lingua.* (1989) **77**:187–222.
16. Nowak MA, Plotkin JB, Krakauer DC. The evolutionary language game. *J Theoret Biol.* (1999) **200**:147–62. doi: 10.1006/jtbi.1999.0981
17. Nowak M. Evolutionary biology of language. *Philos Trans R Soc London B Biol Sci.* (2000) **355**:1615–22. doi: 10.1098/rstb.2000.0723
18. Trapa PE, Nowak MA. Nash equilibria for an evolutionary language game. *J Math Biol.* (2000) **41**:172–88. doi: 10.1007/s002850070004
19. Chen G, Lou Y. *Naming Game.* Switzerland: Springer International Publishing (2019).
20. Lenaerts T, Jansen B, Tuyls K, De Vylder, B. The evolutionary language game: an orthogonal approach. *J Theor Biol.* (2005) **235**:566–82. doi: 10.1016/j.jtbi.2005.02.009
21. Lipowska D, Lipowski A. Emergence of linguistic conventions in multi-agent reinforcement learning. *PLoS ONE.* (2018). **13**:e0208095. doi: 10.1371/journal.pone.0208095
22. Xie J, Emenheiser J, Kirby M, Sreenivasan S, Szymanski BK, Korniss G. Evolution of opinions on social networks in the presence of competing committed groups. *PLoS ONE.* (2012) **7**:e33215. doi: 10.1371/journal.pone.0033215
23. Fan ZY, Lai YC, Tang WKS. Knowledge consensus in complex networks: the role of learning. *tt arXiv:1809.00297* (2018).
24. Eguluz VM, Masuda N, Fernández-Gracia J. Bayesian decision making in human collectives with binary choices. *PLoS ONE.* (2015) **10**:e0121332. doi: 10.1371/journal.pone.0121332
25. Pérez T, Zamora J, Eguíluz VM. Collective intelligence: aggregation of information from neighbors in a guessing game. *PLoS ONE.* (2016) **11**:e0153586. doi: 10.1371/journal.pone.0153586
26. Baronchelli A. A gentle introduction to the minimal naming game. *Belg J Linguist.* (2016) **30**:171–92. doi: 10.1075/bjl.30.08bar
27. Tenenbaum JB, Griffiths TL. Generalization, similarity, and Bayesian inference. *Behav Brain Sci.* (2001) **24**:629–40. doi: 10.1017/S0140525X01000061
28. Griffiths TL, Tenenbaum JB. Optimal predictions in everyday cognition. *Psychol Sci.* (2006) **17**:767–73. doi: 10.1111/j.1467-9280.2006.01780.x
29. Perfors A, Tenenbaum JB, Griffiths TL, Xu F. A tutorial introduction to Bayesian models of cognitive development. *Cognition.* (2011) **120**:302–21. doi: 10.1016/j.cognition.2010.11.015
30. Lake BM, Salakhutdinov R, Tenenbaum JB. Human-level concept learning through probabilistic program induction. *Science.* (2015) **350**:1332–8. doi: 10.1126/science.aab3050
31. Tenenbaum JB. Bayesian modeling of human concept learning. In: *Proceedings of the 1998 Conference on Advances in Neural Information Processing Systems II.* Cambridge, MA: MIT Press (1999). p. 59–65
32. Tenenbaum JB, Kemp C, Griffiths TL, Goodman ND. How to grow a mind: statistics, structure, and abstraction. *Science.* (2011) **331**:1279–85. doi: 10.1126/science.1192788
33. Barber D. *Bayesian Reasoning and Machine Learning.* Cambridge: Cambridge University Press (2012).
34. Murphy K. *Machine Learning: a Probabilistic Perspective*. Cambridge, MA: MIT Press (2012).
35. Evgeniou T, Pontil M, Poggio T. Statistical learning theory: a primer. *Int J Comput Vision.* (2000) **38**:9–13. doi: 10.1023/A:1008110632619.
36. Castelló X, Baronchelli A, Loreto V. Consensus and ordering in language dynamics. *Eur Phys J B.* (2009) **71**:557–64. doi: 10.1140/epjb/e2009-00284-2
37. Baronchelli A, Dall'Asta L, Barrat A, Loreto V. Nonequilibrium phase transition in negotiation dynamics. *Phys Rev E.* (2007) **76**:051102. doi: 10.1103/PhysRevE.76.051102
38. Wittgenstein L. *Philosophical Investigations*. New York, NY: Macmillan (1953).
39. Harney HL. *Bayesian Inference. Parameter Estimation and Decisions*. Berlin: Springer (2016).
40. Jeffreys H. *Theory of Probability*. Oxford: Clarendon Press (1939).
41. Baronchelli A, Felici M, Loreto V, Caglioti E, Steels L. Sharp transition towards shared vocabularies in multi-agent systems. *J Statist Mech Theory Exp.* (2006) P06014. doi: 10.1088/1742-5468/2006/06/P06014
42. Toral R, Tessone C. Finite size effects in the dynamics of opinion formation. *Comm Comp Phys.* (2007) **2**:177.
43. Patriarca M, Castelló X, Uriarte J, Eguíluz V, San Miguel M. Modeling two-language competition dynamics. *Adv Comp Syst.* (2012) **15**:1250048. doi: 10.1142/S0219525912500488
44. Hinrichsen H. Non-equilibrium phase transitions. *Phys A Statist Mechan Appl.* (2006) **369**:1–28. doi: 10.1016/j.physa.2006.04.007
45. Strogatz SH. *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry and Engineering.* Northwestern, NW: CRC Press (2015).
46. Tessone C, Toral R. Diversity-induced resonance in a model for opinion formation. *Eur Phys J B.* (2009) **71**:549. doi: 10.1140/epjb/e2009-00343-8
47. Vaz Martins T, Toral R, Santos M. Divide and conquer: resonance induced by competitive interactions. *Eur Phys J B.* (2009) **67**:329–36. doi: 10.1140/epjb/e2008-00437-9
48. Toivonen R, Castelló X, Eguíluz V, Saramäki J, Kaski K, San Miguel M. Broad lifetime distributions for ordering dynamics in complex networks. *Phys Rev E.* (2009) **79**:016109. doi: 10.1103/PhysRevE.79.016109

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Artificial Intelligence

ORIGINAL RESEARCH
published: 28 April 2020
doi: 10.3389/frai.2020.00025

Discovery of Physics From Data: Universal Laws and Discrepancies

Brian M. de Silva[1]*, *David M. Higdon*[2], *Steven L. Brunton*[3] and *J. Nathan Kutz*[1]

[1] *Applied Mathematics, University of Washington, Seattle, WA, United States,* [2] *Department of Statistics, Virginia Polytechnic Institute and State University, Blacksburg, VA, United States,* [3] *Mechanical Engineering, University of Washington, Seattle, WA, United States*

Machine learning (ML) and artificial intelligence (AI) algorithms are now being used to automate the discovery of physics principles and governing equations from measurement data alone. However, positing a universal physical law from data is challenging without simultaneously proposing an accompanying discrepancy model to account for the inevitable mismatch between theory and measurements. By revisiting the classic problem of modeling falling objects of different size and mass, we highlight a number of nuanced issues that must be addressed by modern data-driven methods for automated physics discovery. Specifically, we show that measurement noise and complex secondary physical mechanisms, like unsteady fluid drag forces, can obscure the underlying law of gravitation, leading to an erroneous model. We use the sparse identification of non-linear dynamics (SINDy) method to identify governing equations for real-world measurement data and simulated trajectories. Incorporating into SINDy the assumption that each falling object is governed by a similar physical law is shown to improve the robustness of the learned models, but discrepancies between the predictions and observations persist due to subtleties in drag dynamics. This work highlights the fact that the naive application of ML/AI will generally be insufficient to infer universal physical laws without further modification.

Keywords: dynamical systems, system identification, machine learning, artificial intelligence, sparse regression, discrepancy modeling

OPEN ACCESS

Edited by:
Víctor M. Eguíluz, Institute of Interdisciplinary Physics and Complex Systems (IFISC), Spain

Reviewed by:
Benson K. Muite, University of Tartu, Estonia
Roger Guimera, University of Rovira i Virgili, Spain

***Correspondence:**
Brian M. de Silva
bdesilva@uw.edu

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Artificial Intelligence

Received: *18 June 2019*
Accepted: *30 March 2020*
Published: *28 April 2020*

Citation:
de Silva BM, Higdon DM, Brunton SL and Kutz JN (2020) Discovery of Physics From Data: Universal Laws and Discrepancies. Front. Artif. Intell. 3:25. doi: 10.3389/frai.2020.00025

1. INTRODUCTION

The ability to derive governing equations and physical principles has been a hallmark feature of scientific discovery and technological progress throughout human history. Even before the scientific revolution, the Ptolemaic doctrine of the *perfect circle* (Peters and Knobel, 1915; Ptolemy, 2014) provided a principled decomposition of planetary motion into a hierarchy of circles, i.e., a bona fide theory for planetary motion. The scientific revolution and the resulting development of calculus provided the mathematical framework and language to precisely describe scientific principles, including gravitation, fluid dynamics, electromagnetism, quantum mechanics, etc. With advances in data science over the past few decades, principled methods are emerging for such scientific discovery from time-series measurements alone. Indeed, across the engineering, physical and biological sciences, significant advances in sensor and measurement technologies have afforded unprecedented new opportunities for scientific exploration. Despite its rapid advancements and wide-spread deployment, *machine learning* (ML) and *artificial intelligence* (AI) algorithms for scientific discovery face significant challenges and limitations, including noisy and corrupt data,

latent variables, multiscale physics, and the tendency for overfitting. In this manuscript, we revisit one of the classic problems of physics considered by Galileo and Newton, that of falling objects and gravitation. We demonstrate that a sparse regression framework is well-suited for physics discovery, while highlighting both the need for principled methods to extract parsimonious physics models and the challenges associated with the naive application of ML/AI techniques. Even this simplest of physical examples demonstrates critical principles that must be considered in order to make data-driven discovery viable across the sciences.

Measurements have long provided the basis for the discovery of governing equations. Through empirical observations of planetary motion, the Ptolemaic theory of motion was developed (Peters and Knobel, 1915; Ptolemy, 2014). This was followed by Kepler's laws of planetary motion and the elliptical courses of planets in a heliocentric coordinate system (Kepler, 2015). By hand calculation, he was able to regress Brahe's state-of-the-art data on planetary motion to the minimally parameterized elliptical orbits which described planetary orbits with a terseness the Ptolemaic system had never managed to achieve. Such models led to the development of Newton's $\mathbf{F} = m\mathbf{a}$ (Newton, 1999), which provided a universal, generalizable, interpretable, and succinct description of physical dynamics. Parsimonious models are critical in the philosophy of Occam's razor: the simplest set of explanatory variables is often the best (Blumer et al., 1987; Domingos, 1999; Bongard and Lipson, 2007; Schmidt and Lipson, 2009). It is through such models that many technological and scientific advancements have been made or envisioned.

What is largely unacknowledged in the scientific discovery process is the intuitive leap required to formulate physics principles and governing equations. Consider the example of falling objects. According to physics folklore, Galileo discovered, through experimentation, that objects fall with the same constant acceleration, thus disproving Aristotle's theory of gravity, which stated that objects fall at different speeds depending on their mass. The leaning tower of Pisa is often the setting for this famous stunt, although there is little evidence such an experiment actually took place (Cooper, 1936; Adler and Coulter, 1978; Segre, 1980). Indeed, many historians consider it to have been a thought experiment rather than an actual physical test. Many of us have been to the top of the leaning tower and have longed to drop a bowling ball from the top, perhaps along with a golf ball and soccer ball, in order to replicate this experiment. If we were to perform such a test, here is what we would likely find: Aristotle was correct. Balls of different masses and sizes *do* reach the ground at different times. As we will show from our own data on falling objects, (noisy) experimental measurements may be insufficient for discovering a constant gravitational acceleration, especially when the objects experience Reynolds numbers varying by orders of magnitudes over the course of their trajectories. But what is beyond dispute is that Galileo did indeed *posit* the idea of a fixed acceleration, a conclusion that would have been exceptionally difficult to come to from such measurement data alone. Gravitation is only one example of the intuitive leap required for a paradigm shifting physics discovery. Maxwell's equations (Maxwell, 1873) have a similar story arc revolving around Coulomb's inverse square law. Maxwell cited Coulomb's torsion balance experiment as establishing the inverse square law while dismissing it only a few pages later as an approximation (Bartlett et al., 1970; Falconer, 2017). Maxwell concluded that Faraday's observation that an electrified body, touched to the inside of a conducting vessel, transfers all its electricity to the outside surface as much more direct proof of the square law. In the end, both would have been approximations, with Maxwell taking the intuitive leap that exactly a power of negative two was needed when formulating Maxwell's equations. Such examples abound across the sciences, where intuitive leaps are made and seminal theories result.

One challenge facing ML and AI methods is their inability to take such leaps. At their core, many ML and AI algorithms involve regressions based on data, and are statistical in nature (Breiman, 2001; Bishop, 2006; Wu et al., 2008; Murphy, 2012). Thus by construction, a model based on measurement data would not produce an exact inverse *square* law, but rather a slightly different estimate of the exponent. In the case of falling objects, ML and AI would yield an Aristotelian theory of gravitation, whereby the data would suggest that objects fall at a speed related to their mass. Of course, even Galileo intuitively understood that air resistance plays a significant role in the physics of falling objects, which is likely the reason he conducted controlled experiments on inclined ramps. Although we understand that air resistance, which is governed by latent fluid dynamic variables, explains the discrepancy between the data and a constant gravity model, our algorithms do not. Without modeling these small disparities (e.g., due to friction, heat dissipation, air resistance, etc.), it is almost impossible to uncover universal laws, such as gravitation. Differences between theory and data have played a foundational role in physics, with general relativity arising from inconsistencies between gravitational theory and observations, and quantum mechanics arising from our inability to explain the photoelectric effect with Maxwell's equations.

Our goal in this manuscript is to highlight the many subtle and nuanced concerns related to data-driven discovery using modern ML and AI methods. Specifically, we highlight these issues on the most elementary of problems: modeling the motion of falling objects. Given our ground-truth knowledge of the physics, this example provides a convenient testbed for different physics discovery techniques. It is important that one clearly understands the potential pitfalls in such methods before applying them to more sophisticated problems which may arise in fields like biology, neuroscience, and climate modeling. Our physics discovery method is rooted in the *sparse identification for non-linear dynamics* (SINDy) algorithm, which has been shown to extract parsimonious governing equations in a broad range of physical sciences (Brunton et al., 2016). SINDy has been widely applied to identify models for fluid flows (Loiseau and Brunton, 2018; Loiseau et al., 2018), optical systems (Sorokina et al., 2016), chemical reaction dynamics (Hoffmann et al., 2019), convection in a plasma (Dam et al., 2017), structural modeling (Lai and Nagarajaiah, 2019), and for model predictive control (Kaiser et al., 2018). There are also a number of theoretical extensions to the SINDy framework, including for identifying partial

differential equations (Rudy et al., 2017; Schaeffer, 2017), and models with rational function non-linearities (Mangan et al., 2016). It can also incorporate partially known physics and constraints (Loiseau and Brunton, 2018). The algorithm can be reformulated to include integral terms for noisy data (Schaeffer and McCalla, 2017) or handle incomplete or limited data (Tran and Ward, 2016; Schaeffer et al., 2018). In this manuscript we show that *group sparsity* (Rudy et al., 2019) may be used to enforce that the same model terms explain *all* of the observed trajectories, which is essential in identifying the correct model terms without overfitting.

SINDy is by no means the only attempt that has been made at using machine learning to infer physical models from data. Gaussian processes have been employed to learn conservation laws described by parametric linear equations (Raissi et al., 2017a). Symbolic regression has been successfully applied to the problem of inferring dynamics from data (Bongard and Lipson, 2007; Schmidt and Lipson, 2009). Another closely related set of approaches are process-based models (Bridewell et al., 2008; Tanevski et al., 2016, 2017) which, similarly to SINDy, allow one to specify a library of relationships or functions between variables based on domain knowledge and produce an interpretable set of governing equations. The principal difference between process-based models and SINDy is that SINDy employs sparse regression techniques to perform function selection which allows a larger class of library functions to be considered than is tractable for process-based models. Deep learning methods have been proposed for accomplishing a variety of related tasks, such as predicting physical dynamics directly (Mrowca et al., 2018), building neural networks that respect given physical laws (Raissi et al., 2017b), discovering parameters in non-linear partial differential equations with limited measurement data (Raissi et al., 2017c), and simultaneously approximating the solution and non-linear dynamics of non-linear partial differential equations (Raissi, 2018). Graph neural networks (Battaglia et al., 2018), a specialized class of neural networks that operate on graphs, have been shown to be effective at learning basic physics simulators from measurement data (Battaglia et al., 2016; Chang et al., 2016) and directly from videos (Watters et al., 2017). It should be noted that the aforementioned neural network approaches either require detailed prior knowledge of the form of the underlying differential equations or fail to yield simple sets of interpretable governing equations.

2. MATERIALS AND METHODS

2.1. Fluid Forces on a Sphere: A Brief History

It must have been immediately clear to Galileo and Newton that committing to a gravitational constant created an inconsistency with experimental data. Specifically, one had to explain why objects of different sizes and shapes fall at different speeds (e.g., a feather vs. a cannon ball). Wind resistance was an immediate candidate to explain the *discrepancy* between a universal gravitational constant and measurement data. The fact that Galileo performed experiments where he rolled balls down inclines seems to suggest that he was keenly aware of the need to isolate and disambiguate the effects of gravitational forces from fluid drag forces. Discrepancies between the Newtonian theory of gravitation and observational data of Mercury's orbit led to Einstein's development of general relativity. Similarly, the photoelectric effect was a discrepancy in Maxwell's equations which led to the development of quantum mechanics.

Discrepancy modeling is therefore a critical aspect of building and discovering physical models. Consider the motion of falling spheres as a prototypical example. In addition to the force of gravity, a falling sphere encounters a fluid drag force as it passes through the air. A simple model of the drag force F_D is given by:

$$F_D = \frac{1}{2}\rho v^2 A C_D, \tag{1}$$

where ρ is the fluid density, v is the velocity of the sphere with respect to the fluid, $A = \pi D^2/4$ is the cross-sectional area of the sphere, D is the diameter of the sphere, and C_D is the dimensionless drag coefficient. As the sphere accelerates through the fluid, its velocity increases, exciting various unsteady aerodynamic effects, such as laminar boundary layer separation, vortex shedding, and eventually a turbulent boundary layer and wake (Moller, 1938; Magarvey and MacLatchy, 1965; Achenbach, 1972, 1974; Calvert, 1972; Smits and Ogg, 2004). Thus, the drag coefficient is a function of the sphere's velocity, and this coefficient generally decreases for increasing velocity. **Figure 1** shows the drag coefficient C_D for a sphere as a function of the Reynolds number $Re = \rho v D/\mu$, where μ is the dynamic viscosity of the fluid; for a constant diameter and viscosity, the Reynolds number is directly proportional to the velocity. Note that the drag coefficient of a smooth sphere will differ from that of a rough sphere. The flow over a rough sphere will become turbulent at lower velocities, causing less flow separation and a more streamlined, lower-drag wake; this explains why golf balls are dimpled, so that they will travel farther (Smits and Ogg, 2004). Thus, (1) states that drag is related to the square of the velocity, although C_D has a weak dependence on velocity. When Re is small, C_D is proportional to $1/v$, resulting in a drag force that is linear in v. For larger values of Re, C_D is approximately constant (away from the steep drop), leading to a quadratic drag force. Eventually, the drag force will balance the force of gravity, resulting in the sphere reaching its *terminal velocity*. In addition, as the fluid wake becomes unsteady, the drag force will also vary in time, although these variations are typically fast and may be time-averaged. Finally, objects accelerating in a fluid will also accelerate the fluid out of the way, resulting in an effective mass that includes the mass of the body and an *added mass* of accelerated fluid (Newman, 1977); however, this added mass force will typically be quite small in air.

In addition to the theoretical study of fluid forces on an idealized sphere, there is a rich history of scientific inquiry into the aerodynamics of sports balls (Mehta, 1985, 2008; Smits and Ogg, 2004; Goff, 2013). Apart from gravity and drag, a ball's trajectory can be influenced by the spin of the ball via the Magnus force or lift force which acts in a direction orthogonal to the drag. Other factors that can affect the forces experienced by a

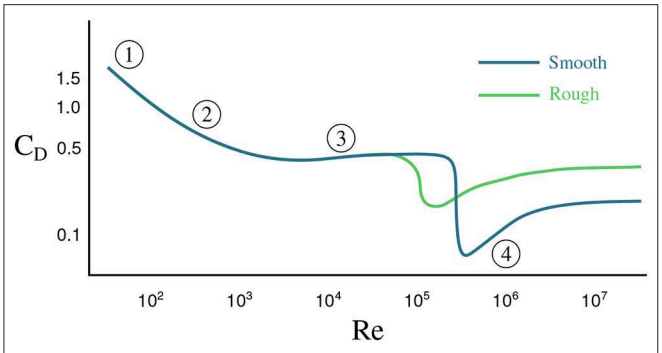


FIGURE 1 | The drag coefficient for a sphere as a function of Reynolds number, *Re*. The dark curve shows the coefficient for a sphere with a smooth surface and the light curve a sphere with a rough surface. The numbers highlight different flow regimes. (1) attached flow and steady separated flow; (2) separated unsteady flow, with laminar flow boundary layer upstream of separation, producing a Kármán vortex street; (3) separated unsteady flow with a chaotic turbulent wake downstream and a laminar boundary layer upstream; (4) post-critical separated flow with turbulent boundary layer.

falling ball include air temperature, wind, elevation, and ball surface shape.

2.2. Data Set

The data considered in this manuscript are height measurements of balls falling through air. These measurements originate from two sources: physical experiments and simulations. Such experiments are popular in undergraduate physics classes where they are used to explore linear vs. quadratic drag (Owen and Ryu, 2005; Kaewsutthi and Wattanakasiwich, 2011; Christensen et al., 2014; Cross and Lindsey, 2014) and scaling laws Sznitman et al. (2017). In June 2013 a collection of balls, pictured in **Figure 2**, were dropped, *twice each*, from the Alex Fraser Bridge in Vancouver, BC from a height of about 35 meters above the landing site. In total 11 balls were dropped: a golf ball, a baseball, two whiffle balls with elongated holes, two whiffle balls with circular holes, two basketballs, a bowling ball, and a volleyball (not pictured). More information about the balls is given in **Table 1**. The air temperature at the time of the drops was 65°F (18°C). A hand held iPad was used to record video of the drops at a rate of 15 frames per second. The height of the falling objects was then estimated by tracking the balls in the resulting videos. **Figure 3** visualizes the second set of ball drops. As one might expect, the whiffle balls all reach the ground later than the other balls. This is to be expected since the openings in their faces increase the drag they experience. Even so, all the balls reach the ground within a second of each other. We also plot the simulated trajectories of two spheres falling with constant linear (in v) drag and the trajectory predicted by constant acceleration. Note that, based on the log-log plot of displacement, none of the balls appears to have reached terminal velocity by the time they hit the ground. This may increase the difficulty of accurately inferring the balls' governing equations. Given only measurements from one regime of falling ball dynamics, it may prove difficult to infer models that generalize to other regimes.

Drawing inspiration from Aristotle, one might form the hypothesis that the amount of time taken by spheres to reach the ground should be a function of the *density* of the spheres. Density takes into account both information about the mass of an object and its volume, which might be thought to affect the air resistance it encounters. We plot the landing time of each ball as a function of its density for both drops in **Figure 4**. To be more precise, because some balls were dropped from slightly different heights, we measure the amount of time it takes each ball to travel a fixed distance after being dropped, not the amount of time it takes the ball to reach the ground. There is a general trend across the tests for the denser balls to travel faster. However, the basketballs defy this trend and complete their journeys about as quickly as the densest ball. This shows there must be more factors at play than just density. There is also variability in the land time of the balls across drops. While most of the balls have very consistent fall times across drops, the blue basketball, golf ball, and orange whiffle ball reach the finish line faster in the first trial than the second one. These differences could be due to a variety of factors, including the balls being released with different initial velocities, or errors in measuring the balls' heights.

There are multiple known sources of error in the measurement data. The relatively low resolution of the videos means that the inferred ball heights are only approximate. In the **Supplementary Material** we attempt to infer the level of noise introduced by our use of heights derived from imperfect video data. Furthermore, the camera was held by a person, not mounted on a tripod, leading to shaky footage. The true bridge height is uncertain because it was measured with a laser range finder claiming to be accurate to within 0.5 m. Because the experiments were executed outside, it is possible for any given drop to have been affected by wind. Detecting exactly when each ball was dropped, at what velocity it was dropped, and when it hit the ground using only videos is certain to introduce further error. Finally, treating these balls as perfect spheres is an approximation whose accuracy depends on the nature of the balls. This idealization seems least appropriate for the whiffle balls, which are sure to exhibit much more complicated aerodynamic effects than, say, the baseball. The bowling ball was excluded from consideration because of corrupted measurements from its first drop.

The situation we strive to mimic with this experiment is one in which the researcher is in a position of ignorance about the system being studied. In order to design an experiment which eliminates the effects of confounding factors, such as air resistance one must already have an appreciation for which factors are worth controlling; one leverages prior knowledge as Galileo did when he employed ramps in his study of falling objects to mitigate the effect of air resistance. In the early stages of investigation of a physical phenomenon, one must often perform poorly-controlled experiments to help identify these factors. We view the ball drop trials as this type of experiment.

In addition to the measurement data just described, we construct a synthetic data set by simulating falling objects with masses of 1 kg and different (linear) drag coefficients. In particular, for each digital ball, we simulate two drops of the same length as the real data and collect height measurements at

FIGURE 2 | The balls that were dropped from the bridge, with the volleyball omitted. From left to right: Golf Ball, Tennis Ball, Whiffle Ball 1, Whiffle Ball 2, Baseball, Yellow Whiffle Ball, Orange Whiffle Ball, Green Basketball, and Blue Basketball. The two colored whiffle balls have circular openings and are structurally identical. The two white whiffle balls have elongated slits and are also identical.

TABLE 1 | Physical measurements, maximum velocities across the two drops, and maximum Reynolds numbers for the dropped balls.

Ball	Radius (m)	Mass (kg)	Density (kg/m)	Max vel. (m/s)	Max Re
Golf ball	0.021963	0.045359	1022.066427	26.63	1.75×10^5
Baseball	0.035412	0.141747	762.037525	26.61	2.83×10^5
Tennis ball	0.033025	0.056699	375.813253	21.95	2.18×10^5
Volleyball	0.105*	NA	NA	22.09	6.96×10^5
Blue basketball	0.119366	0.510291	71.628378	24.80	8.88×10^5
Green basketball	0.116581	0.453592	68.342914	25.06	8.77×10^5
Whiffle ball 1	0.036287	0.028349	141.641937	16.91	1.84×10^5
Whiffle ball 2	0.036287	0.028349	141.641937	16.35	1.78×10^5
Yellow whiffle ball	0.046155	0.042524	103.250857	15.30	2.12×10^5
Orange whiffle ball	0.046155	0.042524	103.250857	15.77	2.18×10^5

**We do not have measurement data for the volleyball, but obtained an estimate for its radius based on other volleyballs in order to approximate its maximum Reynolds number.*

a rate of 15 measurements per second. The balls fall according to the equation $\ddot{x}(t) = -9.8 + D\dot{x}(t)$, with each ball having its own *constant* drag coefficient, $D < 0$. We simulate five balls in total, with respective drag coefficients -0.1, -0.3, -0.3, -0.5, and -0.7. These coefficients are all within the plausible range suggested by the simulated trajectories shown in **Figure 3**. Each object is "dropped" with an initial velocity of 0. Varying amounts of Gaussian noise are added to the height data so that we may better explore the noise tolerance of the proposed model discovery approaches:

$$\tilde{x}_i = x_i + \eta\epsilon_i.$$

where $\eta \geq 0$ and $\epsilon_i \sim N(0,1)$; that is to say ϵ_i is normally distributed with unit variance.

2.3. Methods

In this section we describe the model discovery methods we employ to infer governing equations from noisy data. We first give the mathematical background necessary for learning dynamics via sparse regression and provide a brief overview of the SINDy method in section 2.3.1. In section 2.3.2 we propose a group sparsity regularization strategy for improving the robustness and generalizability of SINDy. We briefly discuss the setup of the model discovery problem we are attempting to solve in section 2.3.3. Finally, we discuss numerical differentiation, a subroutine critical to effective model discovery, in section 2.3.4.

2.3.1. Sparse Identification of Non-linear Dynamical Systems

Consider the non-linear dynamical system for the state vector $\mathbf{x}(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^\top \in \mathbb{R}^n$ defined by

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t)).$$

Given a set of noisy measurements of $\mathbf{x}(t)$, the sparse identification of non-linear dynamics (SINDy) method, introduced in Brunton et al. (2016), seeks to identify $\mathbf{f}: \mathbb{R}^n \to \mathbb{R}^n$. In this section we give an overview of the steps involved in the SINDy method and the assumptions upon which it relies. Throughout this manuscript we refer to this algorithm as the *unregularized SINDy* method, not because it involves no regularization, but because its regularization is not as closely tailored to the problem at hand as the method proposed in section 2.3.2.

For many dynamical systems of interest, the function specifying the dynamics, $\mathbf{f}$, consists of only a few terms. That is to say, when represented in the appropriate basis, there is a sense in which it is sparse. The key idea behind the SINDy method is that if one supplies a rich enough set of candidate functions for representing $\mathbf{f}$, then the correct terms can be identified using sparse regression techniques. The explicit steps are as follows. First we collect a set of (possibly noisy) measurements of the state $\mathbf{x}(t)$ and its derivative $\dot{\mathbf{x}}(t)$ at a sequence of points in time, $t_1, t_2, \ldots, t_m$. These measurements are concatenated into two matrices, the columns of which correspond to different

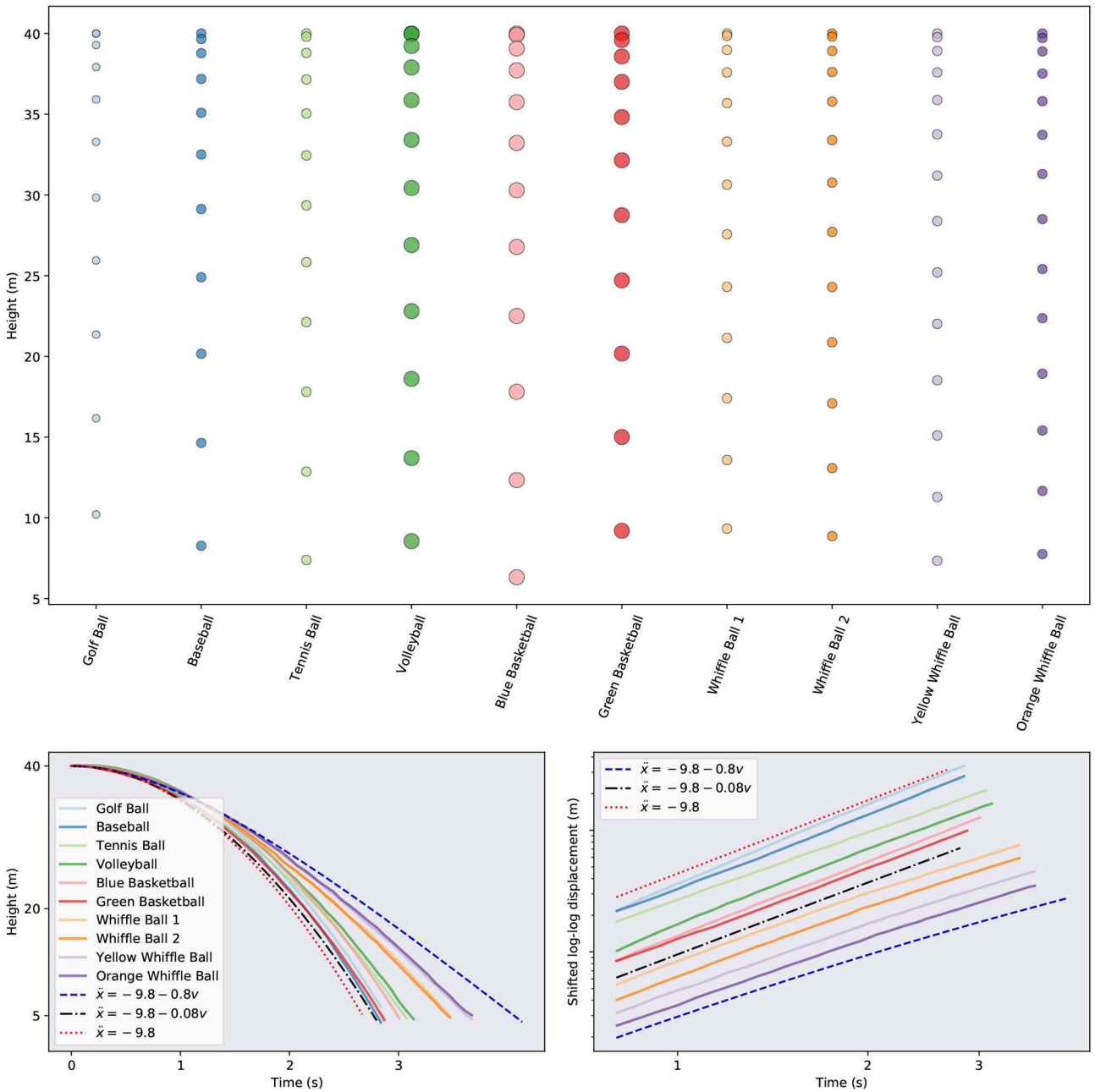


FIGURE 3 | Visualizations of the ball trajectories for the second drop. (Top) Subsampled raw drop data for each ball. (Bottom left) Height for each ball as a function of time. We also include the simulated trajectories of idealized balls with differing levels of drag (black and blue) and a ball with constant acceleration (red). (Bottom right) A log-log plot of the displacement of each ball from its original position atop the bridge. Note that we have shifted the curves vertically and zoomed in on the later segments of the time series to enable easier comparison. In this plot a ball falling at a constant rate (zero acceleration) will have a trajectory represented by a line with slope one. A ball falling with constant acceleration will have a trajectory represented by a line with slope two. A ball with drag will have a trajectory which begins with slope two and asymptotically approaches a line with slope one.

state variables and the rows of which correspond to points in time.

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}(t_1)^\top \\ \mathbf{x}(t_2)^\top \\ \vdots \\ \mathbf{x}(t_m)^\top \end{bmatrix} = \begin{bmatrix} x_1(t_1) & x_2(t_1) & \dots & x_n(t_1) \\ x_1(t_2) & x_2(t_2) & \dots & x_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ x_1(t_m) & x_2(t_m) & \dots & x_n(t_m) \end{bmatrix},$$

$$\dot{\mathbf{X}} = \begin{bmatrix} \dot{\mathbf{x}}(t_1)^\top \\ \dot{\mathbf{x}}(t_2)^\top \\ \vdots \\ \dot{\mathbf{x}}(t_m)^\top \end{bmatrix} = \begin{bmatrix} \dot{x}_1(t_1) & \dot{x}_2(t_1) & \dots & \dot{x}_n(t_1) \\ \dot{x}_1(t_2) & \dot{x}_2(t_2) & \dots & \dot{x}_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ \dot{x}_1(t_m) & \dot{x}_2(t_m) & \dots & \dot{x}_n(t_m) \end{bmatrix}.$$

Next we specify a set of candidate functions, $\{\phi_i(\mathbf{x}) : i = 1, 2, \dots, p\}$, with which to represent $\mathbf{f}$. Examples of candidate

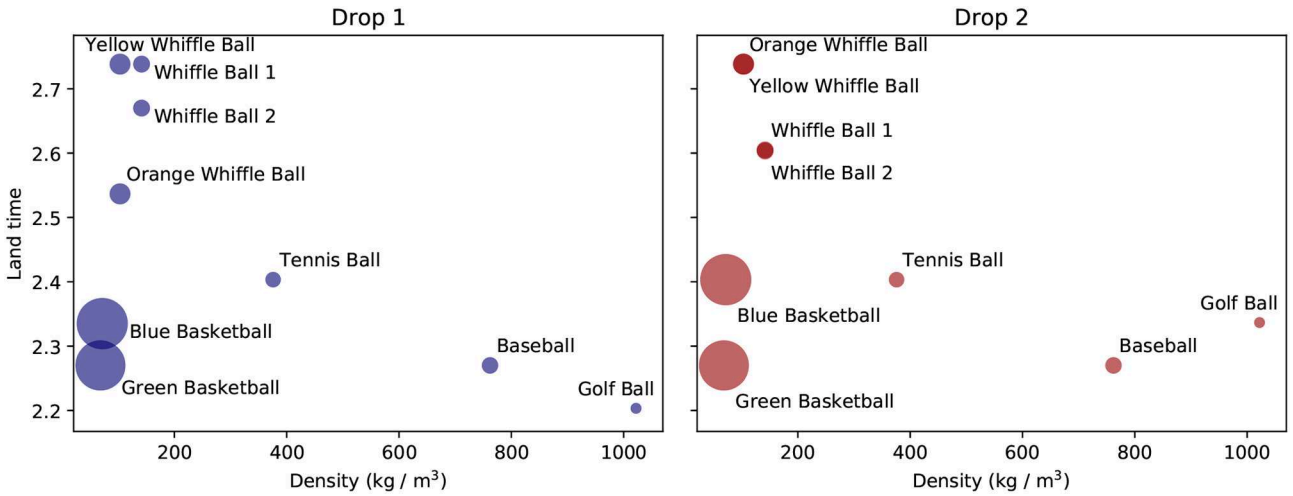


FIGURE 4 | The amount of time taken by each ball to travel a fixed distance as a function of ball density.

functions include monomials up to some finite degree, trigonometric functions, and rational functions. In practice the selection of these functions can be informed by the practitioner's prior knowledge about the system being measured. The candidate functions are evaluated on $\mathbf{X}$ to construct a library matrix

$$\Phi(\mathbf{X}) = \begin{bmatrix} | & | & & | \\ \phi_1(\mathbf{X}) & \phi_2(\mathbf{X}) & \dots & \phi_p(\mathbf{X}) \\ | & | & & | \end{bmatrix}.$$

Note that each column of $\Phi(\mathbf{X})$ corresponds to a single candidate function. Here we have overloaded notation and interpret $\phi(\mathbf{X})$ as the column vector obtained by applying ϕ_i to each row of $\mathbf{X}$. It is assumed that each component of $\mathbf{f}$ can be represented as a *sparse* linear combination of such functions. This allows us to pose a regression problem to be solved for the coefficients used in these linear combinations:

$$\dot{\mathbf{X}} = \Phi(\mathbf{X})\Xi. \tag{2}$$

We adopt MATLAB-style notation and use $\Xi_{(:,j)}$ to denote the j-th column of Ξ. The coefficients specifying the dynamical system obeyed by $\mathbf{x}_j$ are stored in $\Xi_{(:,j)}$:

$$\dot{\mathbf{x}}_j = \mathbf{f}_j(\mathbf{x}) = \Phi\left(\mathbf{x}^\top\right)\Xi_{(:,j)},$$

where $\Phi\left(\mathbf{x}^\top\right)$ is to be interpreted as a (row) vector of symbolic functions of components of $\mathbf{x}$. The full system of differential equations is then given by

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) = \Xi^\top\left(\Phi\left(\mathbf{x}^\top\right)\right)^\top.$$

For concreteness we supply the following example. With the candidate functions $\left\{1, x_1, x_2, x_1x_2, x_1^2, x_2^2\right\}$ the Lotka-Volterra equations

$$\begin{cases} \dot{x}_1 = & \alpha x_1 - \beta x_1 x_2, \\ \dot{x}_2 = & \delta x_1 x_2 - \gamma x_2 \end{cases}$$

can be expressed as

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \Xi^\top\left(\Phi\left(\mathbf{x}^\top\right)\right)^\top = \begin{bmatrix} 0 & \alpha & 0 & -\beta & 0 & 0 \\ 0 & 0 & -\gamma & \delta & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_1x_2 \\ x_1^2 \\ x_2^2 \end{bmatrix}$$

Were we to obtain pristine samples of $\mathbf{x}(t)$ and $\dot{\mathbf{x}}(t)$ we could solve (2) exactly for Ξ. Furthermore, assuming we chose linearly independent candidate functions and avoided collecting redundant measurements, Ξ would be unique and would exhibit the correct sparsity pattern. In practice, however, measurements are contaminated by noise and we actually observe a perturbed version of $\mathbf{x}(t)$.

In many cases $\dot{\mathbf{x}}(t)$ is not observed directly and must instead be approximated from $\mathbf{x}(t)$, establishing another source of error. The previously exact Equation (2), to be solved for Ξ is supplanted by the approximation problem

$$\dot{\mathbf{X}} \approx \Phi(\mathbf{X})\Xi.$$

To find Ξ we solve the more concrete optimization problem

$$\min_{\Xi} \frac{1}{2}\left\|\dot{\mathbf{X}} - \Phi(\mathbf{X})\Xi\right\|_F^2 + \Omega(\Xi), \tag{3}$$

where $\Omega(\cdot)$ is a regularization term chosen to promote sparse solutions and $\|\cdot\|_F$ is the Frobenius norm. Note that because any given column of Ξ encodes a differential equation for a single component of $\mathbf{x}$, each column generates a problem that is decoupled from the problems associated with the other columns. Thus, solving (3) consists of solving n separate regularized least squares problems. Row i of Ξ contains the coefficients of library function ϕ_i for each governing equation.

The most direct way to enforce sparsity is to choose Ω to be the ℓ_0 penalty, defined as $\|\mathbf{M}\|_0 = \sum_{i,j} |\text{sign}(M_{ij})|$. This penalty simply counts the number of non-zero entries in a matrix or vector. However, using the ℓ_0 penalty makes (3) difficult to optimize because $\|\cdot\|_0$ is non-smooth and non-convex. Another common choice is the ℓ_1 penalty defined by $\|\mathbf{M}\|_1 = \sum_{i,j} |M_{ij}|$. This function is the convex relaxation of the ℓ_0 penalty. The LASSO, proposed in Tibshirani (1996), with coordinate descent is typically employed to solve (3) with $\Omega(\cdot) = \|\cdot\|_1$, but this method can become computationally expensive for large data sets and often leads to incorrect sparsity patterns (Su et al., 2017). Hence we solve (3) using the sequential thresholded least-squares algorithm proposed in Brunton et al. (2016), and studied in further detail in Zheng et al. (2018). In essence, the algorithm alternates between (a) successively solving the *unregularized* least-squares problem for each column of Ξ and (b) removing candidate functions from consideration whose corresponding components in Ξ are below some threshold. This threshold or sparsity parameter, is straightforward to interpret: no governing equations are allowed to have any terms with coefficients of magnitude smaller than the threshold. Crucially, it should be noted that just because a candidate function is discarded for one column of Ξ (i.e., for one component's governing equation) does not mean it is removed from contention for the other columns. A simple Python implementation of sequentially thresholded least-squares is provided in the **Supplementary Material**.

We note that if we simulate falling objects with constant acceleration, $\ddot{x}(t) = -9.8$, or linear drag, $\ddot{x}(t) = -9.8 + D\dot{x}(t)$, and add *no noise*, then there is almost perfect agreement between the true governing equations and the models learned by SINDy. The **Supplementary Material** contains a more thorough discussion of such numerical experiments and another example application of SINDy.

SINDy has a number of well-known limitations. The biggest of these is the effect of noise on the learned equations. If one does not have direct measurements of derivatives of state variables, then these derivatives must be computed numerically. Any noise that is present in the measurement data is amplified when it is numerically differentiated, leading to noise in both $\dot{\mathbf{X}}$ and $\Phi(\mathbf{X})$ in (3). In its original formulation, SINDy often exhibits erratic performance in the face of such noise, but extensions have been developed which handle noise more gracefully (Tran and Ward, 2016; Schaeffer and McCalla, 2017). We discuss numerical differentiation further in section 2.3.4. As with other methods, each degree of freedom supplied to the practitioner presents a potential source of difficulty. To use SINDy one must select a set of candidate functions, a sparse regularization function, and a parameter weighing the relative importance of the sparseness of the solution against accuracy. An improper choice of any one of these can lead to poor performance. The set of possible candidate functions is infinite, but SINDy requires one to specify a finite number of them. If one has any prior knowledge of the dynamics of the system being modeled, it can be leveraged here. If not, it is typically recommended to choose a class of functions general enough to encapsulate a wide variety of behaviors (e.g., polynomials or trigonometric functions). In theory, sparse regression techniques should allow one to specify a sizable library of functions, selecting only the relevant ones. However, in practice, the underlying regression problem becomes increasingly ill-conditioned as more functions are added. If one wishes to explore an especially large space of possible library functions it may be better to use other approaches, such as symbolic regression with genetic algorithms (Bongard and Lipson, 2007; Schmidt and Lipson, 2009). A full discussion of how to pick a sparsity-promoting regularizer is beyond the scope of this work. We do note that there have been recent efforts to explore different methods for obtaining sparse solutions when using SINDy (Champion et al., 2019). An appropriate value for the sparsity hyperparameter can be obtained using cross-validation. We note that the need to perform hyperparameter tuning is by no means unique to SINDy. Virtually all machine learning methods require some amount of hyperparameter tuning. There are two natural options for target metrics during cross-validation. The derivatives directly predicted by the linear model can be compared against the measured (or numerically computed) derivatives. Alternatively, the model can be fed into a numerical integrator along with initial conditions to obtain predicted future values for the state variables. These forecasts can then be judged against the measured values. To achieve a balance between model sparsity and accuracy, information theoretic criteria, such as the Akaike information criteria (AIC) or Bayes information criteria (BIC) can be applied (Mangan et al., 2017).

2.3.2. Group Sparsity Regularization

The standard, unregularized SINDy approach attempts to learn the dynamics governing each state variable independently. It does not take into account prior information one may possess regarding relationships between state variables. Intuitively speaking, the balls in our data set (whiffle balls, perhaps, excluded) are similar enough objects that the equations governing their trajectories should include similar terms. In this subsection we propose a group sparsity method which can be interpreted as enforcing this hypothesis when seeking predictive models for the balls.

We draw inspiration for our approach from the group LASSO of Yuan and Lin (2006), which extends the LASSO. The classic LASSO method solves the ℓ_1 regularization problem

$$\beta = \arg\min_{\beta} \frac{1}{2} \|\mathbf{X}\beta - \mathbf{Y}\|_2^2 + \lambda \|\beta\|_1 . \tag{4}$$

which penalizes the magnitude of each component of β *individually*. The group LASSO approach modifies (4) by bundling sets of related entries of β together when computing the penalty term. Let the entries of β be partitioned into G disjoint blocks $\{\beta_1, \beta_2, \ldots, \beta_G\}$, which can be treated as vectors. The group LASSO then solves the following optimization problem

$$\beta = \arg\min_{\beta} \frac{1}{2} \|\mathbf{X}\beta - \mathbf{Y}\|_2^2 + \lambda \sum_{i=1}^{G} \|\beta_i\|_2 . \tag{5}$$

In the case that the groups each consist of exactly one entry of β, (5) reduces to (4). When blocks contain multiple

Algorithm 1: A group sparsity algorithm for the sequential thresholded least squares method.

Data: $\dot{\mathbf{X}} \in \mathbb{R}^{m\times d}$, $\Phi(\mathbf{X}) \in \mathbb{R}^{m\times p}$, and $\delta > 0$
Result: coefficient matrix $\Xi \in \mathbb{R}^{p\times d}$
while *not converged* **do**
  // Solve a least squares problem for each state variable
  for $j \leftarrow 1$ **to** d **do**
    $\Xi_{(:,j)} \leftarrow \arg\min_{\xi} \frac{1}{2} \left\| \dot{\mathbf{X}} - \Phi(\mathbf{X})\xi \right\|_2^2$;
  end
  // Remove library functions with low salience
  for $i \leftarrow 1$ **to** p **do**
    if $R(\Xi_{(i,:)}) < \delta$ **then**
      Delete $\Xi_{(i,:)}$ and $\Phi(\mathbf{X})_{(:,i)}$;
    end
  end
end
Replace deleted rows of Ξ and deleted columns of $\Phi(\mathbf{X})$ with 0's;

entries, the group LASSO penalty encourages them to be retained or eliminated as a group. Furthermore, it drives sets of unimportant variables to truly vanish, unlike the ℓ_2 regularization function which merely assigns small but non-zero values to insignificant variables.

We apply similar ideas in our *group sparsity* method for the SINDy framework and force the models learned for each ball to select the same library functions. Recall that the model variables are contained in Ξ. To enforce the condition that each governing equation should involve the same terms, we identify *rows* of Ξ as sets of variables to be grouped together. Borrowing MATLAB notation again, we let $\Xi_{(i,:)}$ denote row i of Ξ. To perform sequential thresholded least squares with the group sparsity constraint we repeatedly apply the following steps until convergence: (a) solve the least-squares problem (3) *without* a regularization term for each column of Ξ (i.e., for each ball), (b) prune the library, $\Phi(\mathbf{X})$, of functions which have low relevance across most or all of the balls. This procedure is summarized in Algorithm 1.

Here R is a function measuring the importance of a row of coefficients. Possible choices for R include the ℓ_1 or ℓ_2 norm of the input, the mean or median of the absolute values of the entries of the input, or another statistical property of the input entries, such as the lower 25% quantile. In this work we use the ℓ_1 norm. Convergence is attained when no rows of Ξ are removed. Note that while all the models are constrained to be generated by the same library functions, the *coefficients* in front of each can differ from one model to the next. The hyperparameter δ controls the sparsity of Ξ, though not as directly as the sparsity parameter for SINDy. Increasing it will result in models with fewer terms and decreasing it will have the opposite effect. Since we use the ℓ_1 norm and there are 10 balls in our primary data set, rows of Ξ whose average magnitude is $< \frac{\delta}{10}$ are removed.

Because the time series are all noisy, it is likely that some the differential equations returned by the unregularized SINDy algorithm will acquire spurious terms. Insisting that only terms which *most* of the models find useful are kept, as with our group sparsity method, should help to mitigate this issue. In this way we are able to leverage the fact that we have multiple trials involving similar objects to improve the robustness of the learned models to noise. Even if some of the unregularized models from a given drop involve erroneous library functions, we might still hope that, on average, the models will pick the correct terms. Our approach can also be viewed as a type of *ensemble* method wherein a set of models is formed from the time series of a given drop, they are allowed to vote on which terms are important, then the models are retrained using the constrained set of library functions agreed upon in the previous step.

2.3.3. Equations of Motion

Even the simplest model for the height, $x(t)$, of a falling object involves an acceleration term. Consequently, we impose the restriction that our model be a second order (autonomous) differential equation:

$$\ddot{x} = f(x, \dot{x}). \tag{6}$$

The SINDy framework is designed to work with first order systems of differential equations, so we convert (6) into such a system:

$$\begin{cases} \dot{x} = & v \\ \dot{v} = & g(x, v). \end{cases}$$

We then apply SINDy, with $\mathbf{x} = \begin{bmatrix} x & v \end{bmatrix}^\top$ and $f(\mathbf{x}) = \begin{bmatrix} v & g(\mathbf{x}) \end{bmatrix}^\top$, and attempt to learn the function g. In fact, because we already know the correct right-hand side function for $\dot{x}$, we need only concern ourselves with finding an expression for $\dot{v}$.

Our non-linear library consists of polynomials in x and v up to degree three, visualized in **Figure 5**:

$$\Phi(\mathbf{X}) = \begin{bmatrix} | & | & | & | & | & | & & | \\ \mathbf{1} & \mathbf{x}(t) & \mathbf{v}(t) & \mathbf{x}(t)\mathbf{v}(t) & \mathbf{x}(t)^2 & \mathbf{v}(t)^2 & \dots & \mathbf{v}(t)^3 \\ | & | & | & | & | & | & & | \end{bmatrix}. \tag{7}$$

Assuming that the motion of the balls is completely determined by Newton's second law, $F = ma = m\ddot{x}$, we may interpret the SINDy algorithm as trying to discover the force (after dividing by mass) that explains the observed acceleration.

Though we know now that the acceleration of a ball should not depend on its height, we seek to place ourselves in a position of ignorance analogous to the position scientists would have found themselves in centuries ago. We leave it to our algorithm to sort out which terms are appropriate. In practice one might selectively choose which functions to include in the library based on domain knowledge, or known properties of the system being modeled.

2.3.4. Numerical Differentiation

In order to form the non-linear library (7) and the derivative matrix, $\dot{\mathbf{X}}$, we must approximate the first two derivatives of the height data from each drop. Applying standard numerical

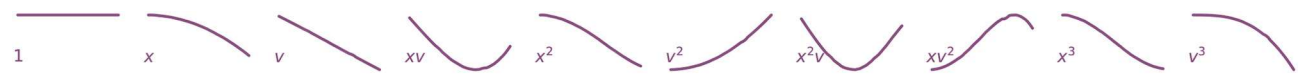


FIGURE 5 | Visualizations of non-linear library functions corresponding to the second green basketball drop. If the motion of the balls is described by Newton's second law, $F = m\ddot{x}$, then these functions can be interpreted as possible forcing terms constituting F.

differentiation techniques to a signal amplifies any noise that is present. This poses a serious problem since we aim to fit a model to the *second* derivative of the height measurements. Because the amount of noise in our data set is non-trivial, two iterations of numerical differentiation will create an intolerable noise level. To mitigate this issue we apply a Savitzky-Golay filter from Savitzky and Golay (1964) to smooth the data before differentiating via second order centered finite differences. Points in a noisy data set are replaced by points lying on low-degree polynomials which are fit to localized patches of the original data with a least-squares method. Other available approaches include using a total variation regularized derivative as in Brunton et al. (2016) or working with an integral formulation of the governing equations as described in Schaeffer and McCalla (2017). We perform a detailed analysis of the error introduced by smoothing and numerical differentiation in the **Supplementary Material**.

3. RESULTS

3.1. Learned Terms

In this section we compare the terms present in the governing equations identified using the unregularized SINDy approach with those present when the group sparsity constraint is imposed. We train separate models on the two drops. The two algorithms are given one sparsity hyperparameter each to be applied for all balls in both drops. The group sparsity method used a value of 1.5 and the other method used a value of 0.04. These parameters were chosen by hand to balance allowing the algorithms enough expressiveness to model the data, while being restrictive enough to prevent widespread overfitting; increasing them produces models with one or no terms and decreasing them results in models with large numbers of terms. See the **Supplementary Material** for a more detailed discussion of our choice of sparsity parameter values.

Figure 6 summarizes the results of this experiment. Learning a separate model for each ball independent of the others allows many models to fall prey to overfitting. Note how most of the governing equations incorporate an extraneous height term. On the other hand, two of the learned models involve only constant acceleration and fail to identify any effect resembling air resistance.

The method leveraging group sparsity is more effective at eliminating extraneous terms and selecting only those which are useful across most balls. Moreover, only the constant and velocity terms are active, matching our intuition that the dominant forces at work are gravity and drag due to air resistance. Interestingly, the method prefers a linear drag term, one proportional to v, to model the discrepancy between measured trajectories and constant acceleration. Even the balls which don't include a velocity term in the unregularized model have this term when group sparsity regularization is employed. This shows that group penalty can simultaneously help to dismiss distracting candidate functions and promote correct terms that may have been overlooked. Is is also reassuring to see that, compared to the other balls, the whiffle ball models have larger coefficients on the v terms. Their accelerations slow at a faster rate as a function of their velocities than do the other balls.

The actual governing equations learned with the group sparsity method are provided in **Table 2**. Every equation has a constant acceleration term within a few meters per second squared of -9.8, but few are quite as close as one might expect. Thus even with a stable method of inferring governing equations, based on this data one would not necessarily conclude that all balls experience the same (mass-divided) force due to gravity. Note also that some of the balls mistakenly adopt *positive* coefficients multiplying v. The balls for which this occurs tend to be those whose motion is well-approximated by constant acceleration. Because the size of the discrepancy between a constant acceleration model and these balls' measured trajectories is not much larger than the amount of error suspected to be present in the data, SINDy has a difficult time choosing an appropriate value for the v terms. One would likely need higher resolution, higher accuracy measurement data in order to obtain reasonable approximations of the drag coefficients or v^2 terms.

At 65°F, the density of air ρ at sea level is 1.211kg/m^3 White and Chul (2011) and its dynamic viscosity μ is $1.82 \times 10^{-5}\text{kg/(m s)}$. The Reynolds number for a ball with diameter D and velocity v will then be

$$Re = 0.667Dv \times 10^5.$$

Table 1 gives the maximum velocities of each ball over the two drops and the corresponding Reynolds numbers. Note that these are the *maximum* Reynolds numbers, not the Reynolds numbers over the entire trajectories. With velocities under 30m/s and diameters from 0.04 to 0.22 m we should expect Reynolds numbers with magnitudes ranging from 10^4 to 10^5 over the course of the balls' trajectories (apart from the very beginnings of each drop). The *average* trajectory consists of about 49 measurements, just over one of which corresponds to a Reynolds number that is $\mathcal{O}\left(10^3\right)$. About 13 of these measurements are associated with Reynolds numbers on the order of 10^4 and roughly 33 with Reynolds numbers of magnitude 10^5. Note that this means the majority of data points were collected when the balls were in the quadratic drag regime. Based on **Figure 1** we should expect balls with Reynolds numbers $< 10^5$ to have drag coefficients of magnitude about 0.5. **Figure 1** suggests that balls experiencing higher Reynolds numbers, such as the volleyball and basketballs should have smaller drag coefficients varying between

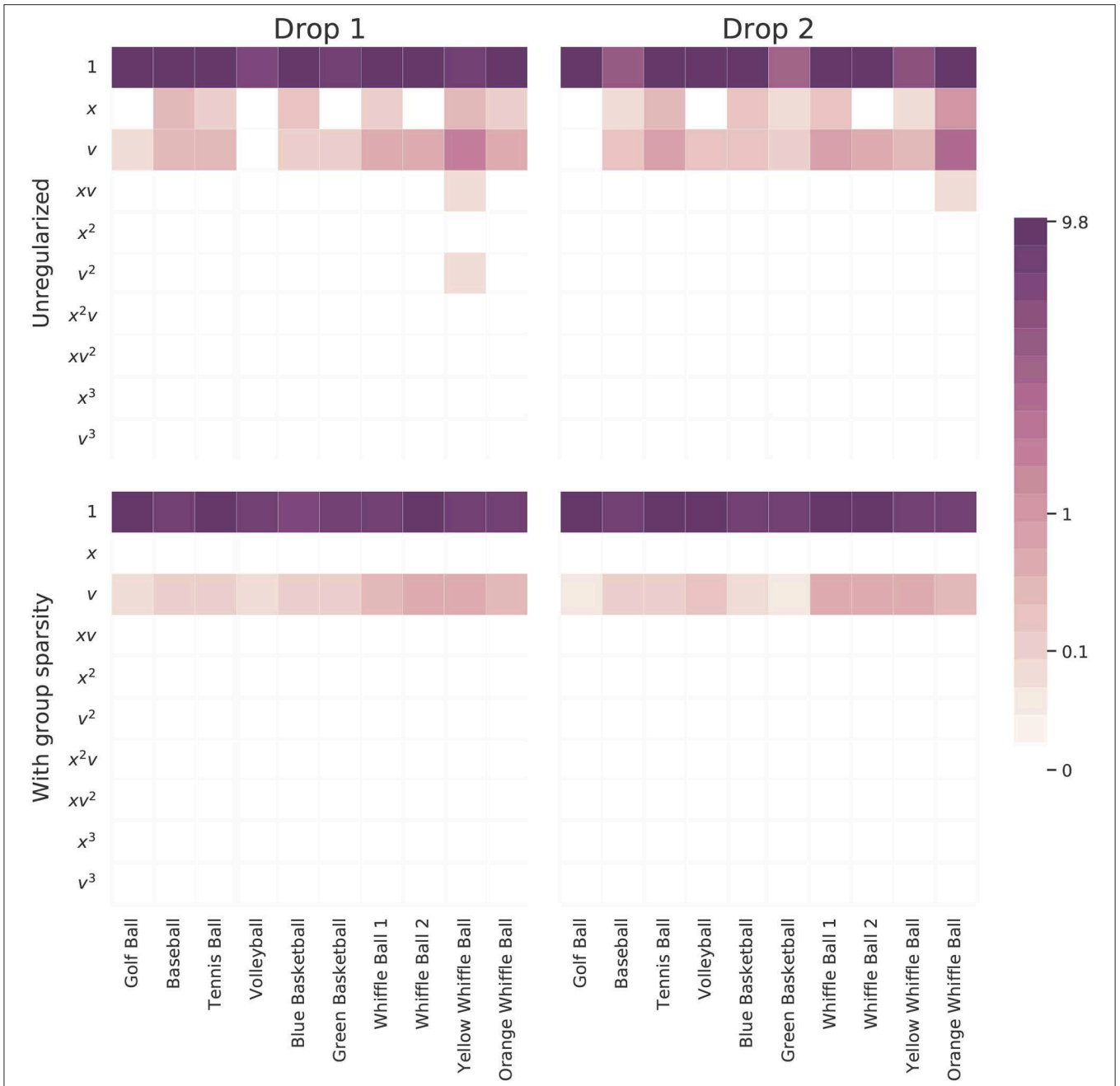


FIGURE 6 | Magnitudes of the coefficients learned for each ball by models trained on one drop either with or without the proposed group sparsity approach. The unregularized approach used a sparsity parameter of 0.04 and the group sparsity method used a value of 1.5. Increasing this parameter slightly in the unregularized case serves to push many models to use only a constant function.

0.05 and 0.3 depending on their smoothness. The predicted (linear) drag coefficients for the volleyball lie in this range while the basketballs' learned drag coefficients are erroneously positive. If the basketballs are treated as being smooth, their drag coefficients predicted by **Figure 1** may be too small for SINDy to identify given the noisy measurement data. A similar effect seems to occur for the golf ball. Though it experiences a lower Reynolds number, its dimples induce a turbulent flow over its surface, granting it a small drag coefficient at a lower Reynolds number. Overall, the linear drag coefficients predicted by the model are at least within a physically reasonable range, with some outliers having incorrect signs.

Next we turn to the simulated data set. We perform the same experiment as with the real world data: we apply both versions of SINDy to a series of simulated ball drops and then note the models that are inferred. Our findings are shown in

TABLE 2 | Models learned by applying SINDy with group sparsity regularization (sparsity parameter $\delta = 1.5$) to each of the two ball drops.

Ball	First drop	Second drop
Golf ball	$\ddot{x} = -9.34 + 0.05v$	$\ddot{x} = -9.44 - 0.03v$
Baseball	$\ddot{x} = -8.51 + 0.14v$	$\ddot{x} = -7.56 + 0.14v$
Tennis ball	$\ddot{x} = -9.08 - 0.13v$	$\ddot{x} = -8.64 - 0.12v$
Volleyball	$\ddot{x} = -8.11 - 0.08v$	$\ddot{x} = -9.64 - 0.23v$
Blue basketball	$\ddot{x} = -6.71 + 0.15v$	$\ddot{x} = -7.50 + 0.07v$
Green basketball	$\ddot{x} = -7.36 + 0.10v$	$\ddot{x} = -8.05 + 0.02v$
Whiffle ball 1	$\ddot{x} = -8.24 - 0.34v$	$\ddot{x} = -9.44 - 0.43v$
Whiffle ball 2	$\ddot{x} = -9.81 - 0.56v$	$\ddot{x} = -9.79 - 0.48v$
Yellow whiffle ball	$\ddot{x} = -8.50 - 0.47v$	$\ddot{x} = -8.45 - 0.46v$
Orange whiffle ball	$\ddot{x} = -7.83 - 0.35v$	$\ddot{x} = -8.03 - 0.42v$

Figure 7. We need not say much about the standard approach: it does a poor job of identifying coherent models for all levels of noise. The group sparsity regularization is much more robust to noise, identifying the correct terms and their magnitudes for noise levels up to half a meter (in standard deviation). For more significant amounts of noise, even this method is unable to decide between adopting x or v into its models. Perhaps surprisingly, if a v^2 term with coefficient $\sim$ 0.1 is added to the simulated model[1], the learned coefficients look nearly identical. Although this additional term visibly alters the trajectory (before it is corrupted by noise), none of the learned equations capture it, even in the absence of noise. One reason for this is because the coefficient multiplying v^2 is too small to be retained during the sequential thresholding least squares procedure. If we decrease the sparsity parameter enough to accommodate it, the models also acquire spurious higher order terms. To infer the v^2 term using the approach outlined here, one would need to design and carry out additional experiments which better isolate this effect, perhaps by using a denser fluid or by dropping a ball with a larger diameter of relatively small mass, thereby increasing the constant multiplying $v^2 C_D$ in (1). A much more realistic drag force based on (1) can be used to simulate falling balls. Such a drag force will shift from being linear to quadratic in v over the course of a ball's trajectory. In this scenario neither version of SINDy identifies a v^2 term, regardless of how much many measurements are collected, but both detect linear drag, exhibiting similar performance as is shown here. A more detailed discussion can be found in the **Supplementary Material**.

3.2. Model Error

We now turn to the problem of testing the predictive performance of models learned from the data. We benchmark four models of increasing complexity on the drop data. The model templates are as follows:

1. Constant acceleration: $\ddot{x} = \alpha$
2. Constant acceleration with linear drag: $\ddot{x} = \alpha + \beta v$
3. Constant acceleration with linear and quadratic drag: $\ddot{x} = \alpha + \beta v + \gamma v^2$
4. Overfit model: Set a low sparsity threshold and allow SINDy to fit a more complicated model to the data

The model parameters α, β, and γ are learned using the SINDy algorithm using libraries consisting of just the terms required by the templates. The testing procedure consists of constructing a total of 80 models (4 templates $\times$ 10 balls $\times$ 2 drops) and then using them to predict a quantity of interest. First a template model is selected then it is trained using one ball's trajectory from one drop. Once trained, the model is given the initial conditions (initial height and velocity) from the same ball's other drop and tasked with predicting the ball's height after 2.8 s have passed[2]. Recall from **Figure 4** that the same ball dropped twice from the same height by the same person on the same day can hit the ground at substantially different times. In the absence of any confounding factors, the time it takes a sphere to reach the ground after being released will vary significantly based on its initial velocity. Since there is sure to be some error in estimating the initial height and velocity of the balls, we should expect only modest accuracy in predicting their landing times. We summarize the outcome of this experiment in **Figure 8**. The error tends to decrease significantly between model one and model two, marking a large step in explaining the discrepancy between a constant acceleration model and observation. There does not appear to be a large difference between the predictive powers of models two and three as both seem to provide similar levels of accuracy. Occam's razor might be invoked here to motivate a preference for model two over model three since it is simpler and has the same accuracy. This provides further evidence that the level of noise and error in the data set is too large to allow one to accurately infer the dynamics due to v^2. Adding additional terms to the equations seems to weaken their generalizability somewhat, as indicated by the slight increase in errors for model four.

Figure 9 visualizes the forecasts of the learned equations for two of the balls along with their deviation from the true measurements. The models are first trained on data from drop 2, then they are given initial conditions from the same drop and made to predict the full trajectories. There are a few observations to be made. The constant acceleration models (model one) are clearly inadequate, especially for the whiffle ball. Their error is much higher than that of the other models indicating that they are underfitting the data, though constant acceleration appears to be a reasonable approximation for a falling golf ball. Models two through four all seem to be imitating the trajectories to about the level of the measurement noise, which is about the most we could hope of them. It is difficult to say which model is best by looking at these plots alone. To break the tie we can observe what happens if we

[1]It should be noted that, based on the balls' approximated velocities, the largest coefficient multiplying v^2 (i.e., $\frac{1}{2m}\rho A C_D$ from (1), where m is the mass of a ball), is < 0.08 in magnitude, across all the trials.

[2]This number corresponds to the shortest set of measurement data across all the trials. All models are evaluated at 2.8 s to allow for meaningful comparison of error rates between models.

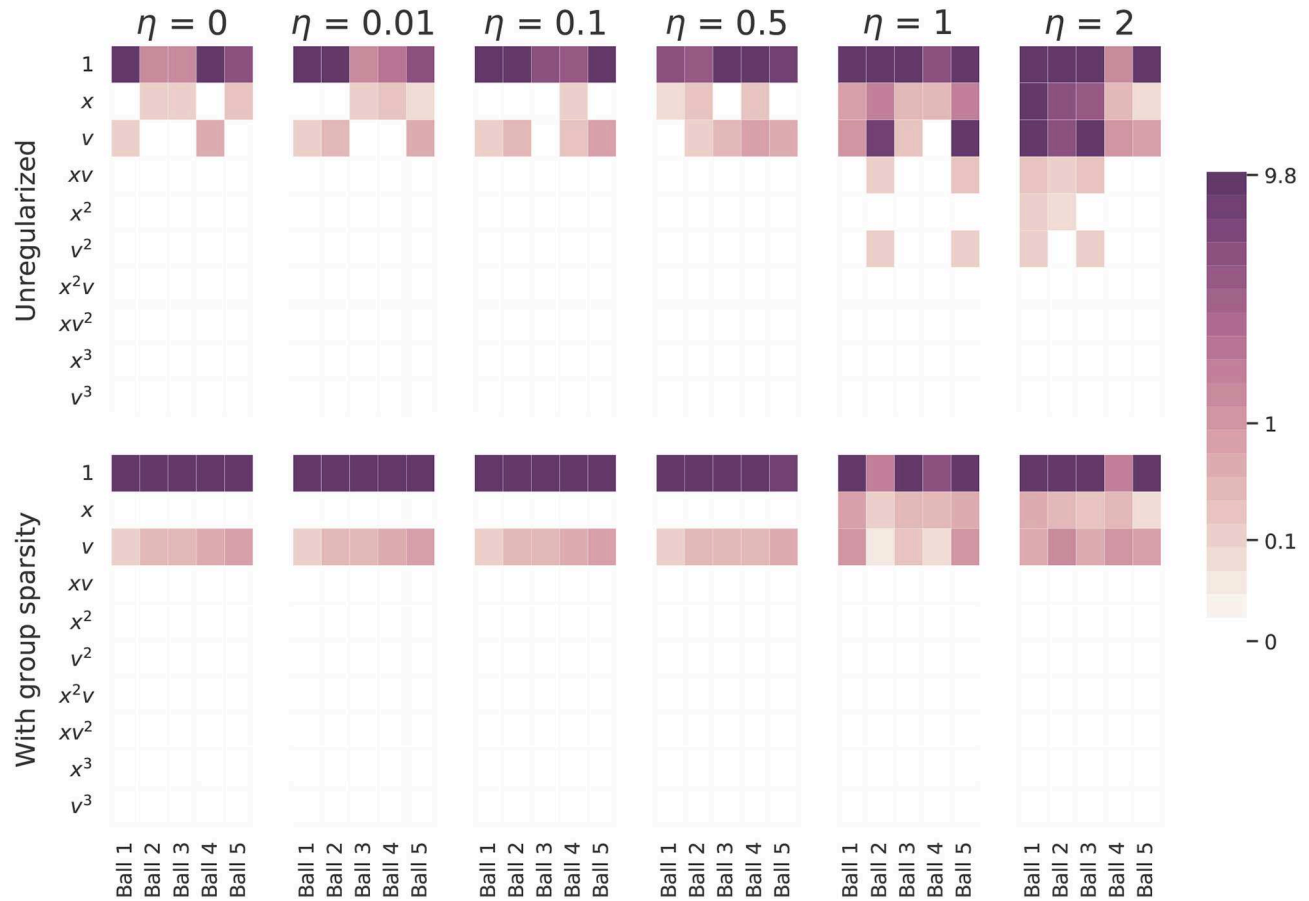


FIGURE 7 | A comparison of coefficients of the models inferred from the simulated falling balls. The top row shows the coefficients learned with the standard SINDy algorithm and the bottom row the coefficients learned with the group sparsity method. η indicates the amount of noise added to the simulated ball drops. The standard approach used a sparsity parameter of 0.05 and the group sparsity method used a value of 1.5. The balls were simulated using constant acceleration and the following respective coefficients multiplying v: −0.1, −0.3, −0.3, −0.5, −0.7.

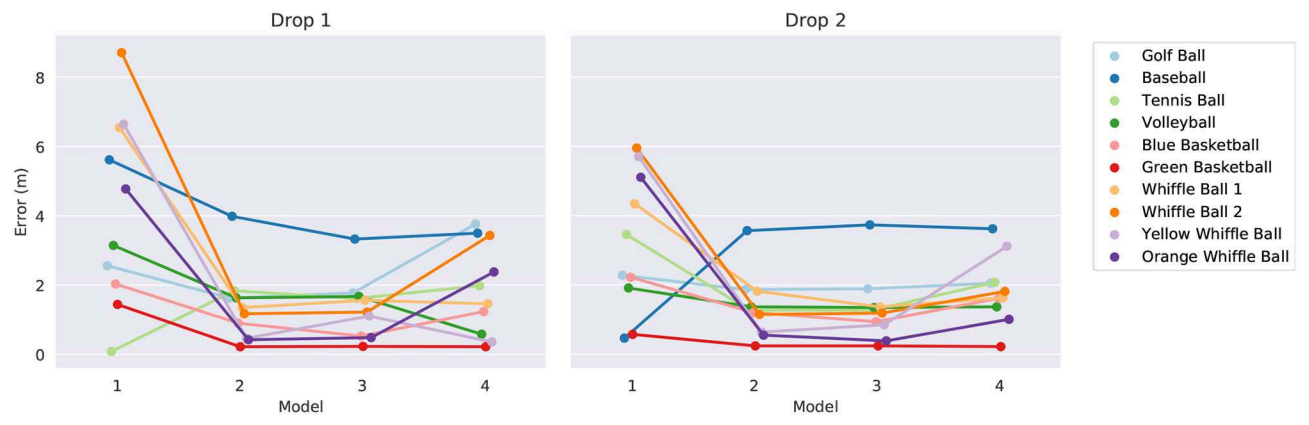


FIGURE 8 | The error in landing time predictions for the four models. The results for the models trained on drops one and two are shown on the left and right, respectively. We have intentionally jittered the horizontal positions of the data points to facilitate easier comparison.

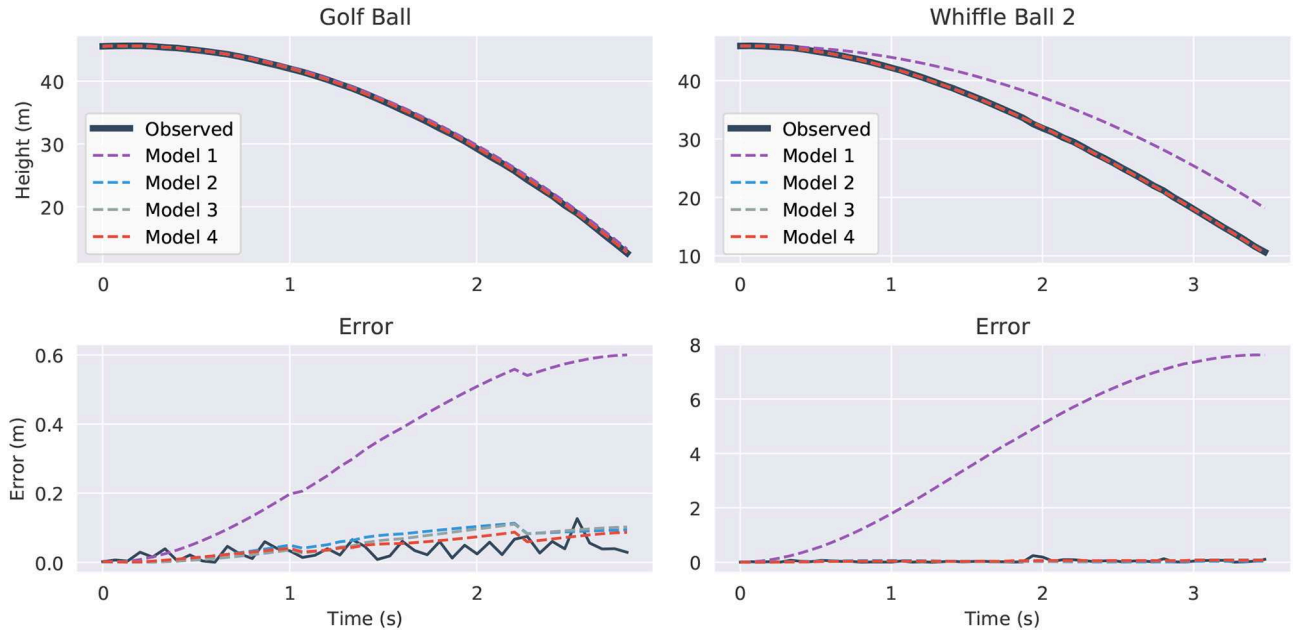


FIGURE 9 | Predicted trajectories and error for the Golf Ball (top) and Whiffle Ball 2 (bottom). On the left we compare the predicted trajectories against the true path and on the right we show the absolute error for the predictions. The "Observed" lines in the error plots show the difference between the original height measurements and the smoothed versions used for differentiation. They give an idea of the amount of intrinsic measurement noise. All models plotted were trained and evaluated on drop 2.

evaluate the models in "unfamiliar" circumstances and force them to extrapolate.

Supplying the same initial conditions as before, with initial height shifted up to avoid negative heights, we task the models with predicting the trajectories out to 15 s. The results are shown in **Figure 10**. All four models fit the observed data itself fairly well. However, 6 or 7 s after the balls are released, a significant degree of separation has started to emerge between the trajectories. The divergence of the model four instances is the most abrupt and the most pronounced. The golf ball's model grows without bound after 7 s. It is here that the danger of overfit, high-order models becomes obvious. In contrast, the other models are better behaved. For the golf ball models one through three agree relatively well, perhaps showing that it is easier to predict the path of a falling golf ball than a falling whiffle ball. That model two is so similar to the constant acceleration of model one also suggests that the golf ball experiences very little drag. The v^2 term for model three has a coefficient which is erroneously positive and essentially cancels out the speed dampening effects of the drag term, leading to an overly rapid predicted descent. Models two and three agree extremely well for the whiffle ball as the learned v^2 coefficient is very small in magnitude.

4. DISCUSSION AND CONCLUSIONS

In this work, we have revisited the classic problem of modeling the motion of falling objects in the context of modern machine learning, sparse optimization, and model selection. In particular, we develop data-driven models from experimental position measurements for several falling spheres of different size, mass, roughness, and porosity. Based on this data, a hierarchy of models are selected via sparse regression in a library of candidate functions that may explain the observed acceleration behavior. We find that models developed for individual ball-drop trajectories tend to overfit the data, with all models including a spurious height-dependent force and lower-density balls resulting in additional spurious terms. Next, we impose the assumption that all balls must be governed by the same basic model terms, perhaps with different coefficients, by considering all trajectories simultaneously and selecting models via group sparsity. These models are all parsimonious, with only two dominant terms, and they tend to generalize without overfitting.

Although we often view the motion of falling spheres as a solved problem, the observed data is quite rich, exhibiting a range of behaviors. In fact, a constant gravitational acceleration is not immediately obvious, as the falling motion is strongly affected by complex unsteady fluid drag forces; the data alone would suggest that each ball has its own slightly different gravity constant. It is interesting to note that our group sparsity models include a drag force that is proportional to the velocity, as opposed to the *textbook* model that includes the square of velocity that is predicted for a constant drag coefficient. However, in reality the drag coefficient decreases with velocity, as shown in **Figure 1**, which may contribute to the force being proportional to velocity. Even when a higher fidelity drag model is used—a model containing rational terms missing from and poorly

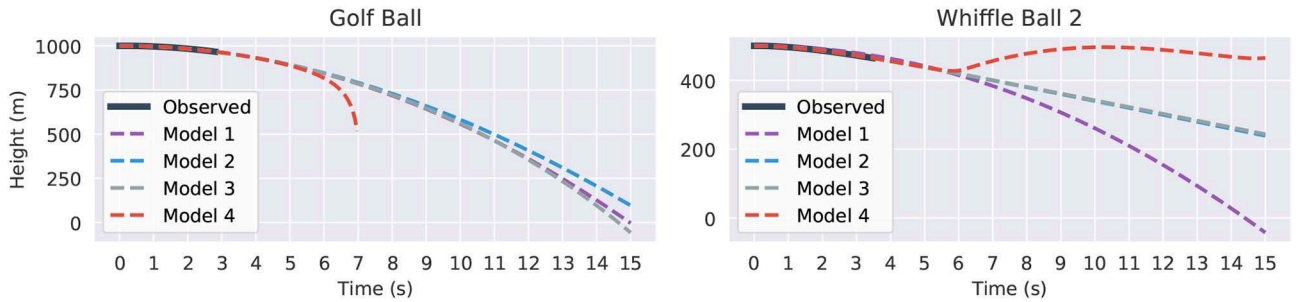


FIGURE 10 | Fifteen seconds forecasted trajectories for the Golf Ball (left) and Whiffle Ball 2 (right) based on the second drop. Part of the graph of Model 4 (red) is omitted in the Golf Ball plot because it diverged to $-\infty$.

approximated by the polynomial library functions—to collect measurements uncorrupted by noise, SINDy struggles to identify coherent dynamics. In general SINDy may not exhibit optimal performance if not equipped with a library of functions in which dynamics can be represented sparsely. We emphasize that although the learned models tend to fit the data relatively well, it would be a mistake to assume that they would retain their accuracy for Reynolds numbers larger than those present in the training data. In particular we should expect the models to have trouble extrapolating beyond the drag crisis where the dynamics change considerably. This weakness is inherent in virtually all machine learning models; their performance is best when they are applied to data similar to what they have already seen and dubious when applied in novel contexts. That is to say they excel at interpolation, but are often poor extrapolators.

Collecting a richer set of data should enable the development of refined models with more accurate drag physics[3], and this is the subject of future work. In particular, it would be interesting to collect data for spheres falling from greater heights, so that they reach terminal velocity. It would also be interesting to systematically vary the radius, mass, surface roughness, and porosity, for example to determine non-dimensional parameters. Finally, performing similar tests in other fluids, such as water, may also enable the discovery of added mass forces, which are quite small in air. Such a dataset would provide a challenging motivation for future machine learning techniques.

We were able to draw upon previous fluid dynamics research to establish a "ground truth" model against which to compare the models proposed by SINDy. However, in less mature application areas one may not be fortunate enough to have a theory-backed set of reference equations, making it challenging to assess the quality of learned models. Many methods in numerical analysis come equipped with *a priori* or *a posteriori* error estimators or convergence results to give one an idea of the size of approximation errors. Similarly, in statistics goodness of fit estimators exist to help guide practitioners about what type of performance they should expect from various models. A comprehensive study into whether similar techniques could be adopted for application to SINDy would be an interesting topic for future research efforts.

We believe that it is important to draw a parallel between great historical scientific breakthroughs, such as the discovery of a universal gravitational constant, and modern approaches in machine learning. Although computational learning algorithms are becoming increasingly powerful, they face many of the same challenges that human scientists have faced for centuries. These challenges include trade offs between model fidelity and the quality and quantity of data, with inaccurate measurements degrading our ability to disambiguate various physical effects. With noisy data, one can only expect model identification techniques to uncover the dominant, leading-order effects, such as gravity and simple drag; for subtler effects, more accurate measurement data is required. Modern learning architectures are often also prone to overfitting without careful cross-validation and regularization, and models that are both interpretable and generalizable come at a premium. Typically the regularization encodes some basic human assumption, such as sparse regularization, which promotes parsimony in models. More fundamentally, it is not always clear what should be measured, what terms should be modeled, and what parameters should be varied to isolate the effect one wishes to study. Historically, this type of scientific inquiry has been driven by human curiosity and intuition, which will be critical elements if machine intelligence is to advance scientific discovery.

DATA AVAILABILITY STATEMENT

The code and datasets used in this study are available at https://github.com/briandesilva/discovery-of-physics-from-data.

AUTHOR CONTRIBUTIONS

BS: implementation, numerical experiments, results analysis, and manuscript writing. DH: falling object experiments and scientific advice.

[3]We note that in order to properly resolve these more complex drag dynamics with SINDy the candidate library would likely need to be enriched.

SB and JK: scientific advice, results analysis, and manuscript writing.

ACKNOWLEDGMENTS

We would like to acknowledge funding support from the Defense Advanced Research Projects Agency (DARPA PA-18-01-FP-125) and the Air Force Office of Scientific Research (FA9550-18-1-0200 for SB and FA9550-17-1-0329 for JK).

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/frai.2020.00025/full#supplementary-material

REFERENCES

Achenbach, E. (1972). Experiments on the flow past spheres at very high reynolds numbers. *J. Fluid Mech.* 54, 565–75. doi: 10.1017/S0022112072000874

Achenbach, E. (1974). Vortex shedding from spheres. *J. Fluid Mech.* 62, 209–21. doi: 10.1017/S0022112074000644

Adler, C. G., and Coulter, B. L. (1978). Galileo and the tower of pisa experiment. *Am. J. Phys.* 46, 199–201. doi: 10.1119/1.11165

Bartlett, D., Goldhagen, P., and Phillips, E. (1970). Experimental test of coulomb's law. *Phys. Rev. D* 2:483. doi: 10.1103/PhysRevD.2.483

Battaglia, P., Pascanu, R., Lai, M., Rezende, D. J., and Kavukcuoglu, K. (2016). "Interaction networks for learning about objects, relations and physics," in *Advances in Neural Information Processing Systems*, (Long Beach, CA), 4502–10.

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., et al. (2018). Relational inductive biases, deep learning, and graph networks. *arXiv* 1806.01261.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. New York, NY: Springer.

Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. K. (1987). Occam's razor. *Inform. Process. Lett.* 24, 377–380. doi: 10.1016/0020-0190(87)90114-1

Bongard, J., and Lipson, H. (2007). Automated reverse engineering of nonlinear dynamical systems. *Proc. Natl. Acad. Sci. U.S.A.* 104, 9943–9948. doi: 10.1073/pnas.0609476104

Breiman, L. (2001). Statistical modeling: the two cultures (with comments and a rejoinder by the author). *Stat. Sci.* 16, 199–231. doi: 10.1214/ss/1009213726

Bridewell, W., Langley, P., Todorovski, L., and Džeroski, S. (2008). Inductive process modeling. *Mach. Learn.* 71, 1–32. doi: 10.1007/s10994-007-5042-6

Brunton, S. L., Proctor, J. L., and Kutz, J. N. (2016). Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proc. Natl. Acad. Sci. U.S.A.* 113, 3932–3937. doi: 10.1073/pnas.1517384113

Calvert, J. (1972). Some experiments on the flow past a sphere. *Aeronaut. J.* 76, 248–250.

Champion, K., Zheng, P., Aravkin, A. Y., Brunton, S. L., and Kutz, J. N. (2019). A unified sparse optimization framework to learn parsimonious physics-informed models from data. *arXiv* 1906.10612.

Chang, M. B., Ullman, T., Torralba, A., and Tenenbaum, J. B. (2016). A compositional object-based approach to learning physical dynamics. *arXiv* 1612.00341.

Christensen, R. S., Teiwes, R., Petersen, S. V., Uggerhøj, U. I., and Jacoby, B. (2014). Laboratory test of the galilean universality of the free fall experiment. *Phys. Educ.* 49:201. doi: 10.1088/0031-9120/49/2/201

Cooper, L. (1936). *Aristotle, Galileo, and the Tower of Pisa*, New York, NY: Cornell University Press Ithaca.

Cross, R., and Lindsey, C. (2014). Measuring the drag force on a falling ball. *Phys. Teacher* 52, 169–170. doi: 10.1119/1.4865522

Dam, M., Brøns, M., Juul Rasmussen, J., Naulin, V., and Hesthaven, J. S. (2017). Sparse identification of a predator-prey system from simulation data of a convection model. *Phys. Plasmas* 24:022310. doi: 10.1063/1.4977057

Domingos, P. (1999). The role of occam's razor in knowledge discovery. *Data Mining Knowl. Discov.* 3, 409–425. doi: 10.1023/A:1009868929893

Falconer, I. (2017). No actual measurement…was required: Maxwell and cavendish's null method for the inverse square law of electrostatics. *Stud. Hist. Philos. Sci. A* 65, 74–86. doi: 10.1016/j.shpsa.2017.05.001

Goff, J. E. (2013). A review of recent research into aerodynamics of sport projectiles. *Sports Eng.* 16, 137–154. doi: 10.1007/s12283-013-0117-z

Hoffmann, M., Fröhner, C., and Noé,, F. (2019). Reactive SINDy: discovering governing reactions from concentration data. *J. Chem. Phys.* 150:025101. doi: 10.1063/1.5066099

Kaewsutthi, C., and Wattanakasiwich, P. (2011). "Student learning experiences from drag experiments using high-speed video analysis," in *Proceedings of The Australian Conference on Science and Mathematics Education (formerly UniServe Science Conference)*, (Melbourne), Vol. 17.

Kaiser, E., Kutz, J. N., and Brunton, S. L. (2018). Sparse identification of nonlinear dynamics for model predictive control in the low-data limit. *Proc. R. Soc. Lond. A* 474:2219. doi: 10.1098/rspa.2018.0335

Kepler, J. (2015). *Astronomia Nova. Pragae 1609*. Green Lion Press.

Lai, Z., and Nagarajaiah, S. (2019). Sparse structural system identification method for nonlinear dynamic systems with hysteresis/inelastic behavior. *Mech. Syst. Signal Process.* 117, 813–42. doi: 10.1016/j.ymssp.2018.08.033

Loiseau, J.-C., and Brunton, S. L. (2018). Constrained sparse Galerkin regression. *J. Fluid Mech.* 838, 42–67. doi: 10.1017/jfm.2017.823

Loiseau, J.-C., Noack, B. R., and Brunton, S. L. (2018). Sparse reduced-order modeling: sensor-based dynamics to full-state estimation. *J. Fluid Mech.* 844, 459–90. doi: 10.1017/jfm.2018.147

Magarvey, R., and MacLatchy, C. (1965). Vortices in sphere wakes. *Can. J. Phys.* 43, 1649–56. doi: 10.1139/p65-154

Mangan, N. M., Brunton, S. L., Proctor, J. L., and Kutz, J. N. (2016). Inferring biological networks by sparse identification of nonlinear dynamics. *IEEE Trans. Mol. Biol. Multiscale Commun.* 2, 52–63. doi: 10.1109/TMBMC.2016.2633265

Mangan, N. M., Kutz, J. N., Brunton, S. L., and Proctor, J. L. (2017). Model selection for dynamical systems via sparse regression and information criteria. *Proc. R. Soc. A* 473, 1–16. doi: 10.1098/rspa.2017.0009

Maxwell, J. C. (1873). *A Treatise on Electricity and Magnetism*, Vol. 1. Oxford: Clarendon Press.

Mehta, R. D. (1985). Aerodynamics of sports balls. *Annu. Rev. Fluid Mech.* 17, 151–189. doi: 10.1146/annurev.fl.17.010185.001055

Mehta, R. D. (2008). "Sports ball aerodynamics," in *Sport Aerodynamics* (Vienna: Springer), 229–331. doi: 10.1007/978-3-211-89297-8_12

Moller, W. (1938). Experimentelle untersuchung zur hydromechanick der hugel. *Phys. Z.* 35, 57–80.

Mrowca, D., Zhuang, C., Wang, E., Haber, N., Fei-Fei, L. F., Tenenbaum, J., et al. (2018). "Flexible neural representation for physics prediction," in *Advances in Neural Information Processing Systems*, (Montréal), 8799–8810.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. Cambridge, MA: MIT Press.

Newman, J. (1977). *Marine Hydrodynamics*. Cambridge, MA: The MIT Press.

Newton, I. (1999). *The Principia: Mathematical Principles of Natural Philosophy*. Berkeley, CA: University of California Press.

Owen, J. P., and Ryu, W. S. (2005). The effects of linear and quadratic drag on falling spheres: an undergraduate laboratory. *Eur. J. Phys.* 26:1085. doi: 10.1088/0143-0807/26/6/016

Peters, C. H. F., and Knobel, E. B. (1915). *Ptolemy's Catalogue of Stars: A Revision of the Almagest*. Washington, DC: Physics and Chemistry in Space.

Ptolemy, C. (2014). *The Almagest: Introduction to the Mathematics of the Heavens*. Santa Fe, NM: Green Lion Press.

Raissi, M. (2018). Deep hidden physics models: Deep learning of nonlinear partial differential equations. *J. Mach. Learn. Res.* 19, 932–55. Available online at: http://jmlr.org/papers/v19/18-046.html.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. (2017a). Machine learning of linear differential equations using gaussian processes. *J. Comput. Phys.* 348, 683–93. doi: 10.1016/j.jcp.2017.07.050

Raissi, M., Perdikaris, P., and Karniadakis, G. E. (2017b). Physics informed deep learning (part I): data-driven solutions of nonlinear partial differential equations. *arXiv* 1711.10561.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. (2017c). Physics informed deep learning (part II): data-driven discovery of nonlinear partial differential equations. *arXiv* 1711.10566.

Rudy, S., Alla, A., Brunton, S. L., and Kutz, J. N. (2019). Data-driven identification of parametric partial differential equations. *SIAM J. Appl. Dyn. Syst.* 18, 643–60. doi: 10.1137/18M1191944

Rudy, S. H., Brunton, S. L., Proctor, J. L., and Kutz, J. N. (2017). Data-driven discovery of partial differential equations. *Sci. Adv.* 3:e1602614. doi: 10.1126/sciadv.1602614

Savitzky, A., and Golay, M. J. E. (1964). Smoothing and differentiation of data by simplified least squares procedures. *Anal. Chem.* 36, 1627–39. doi: 10.1021/ac60214a047

Schaeffer, H. (2017). Learning partial differential equations via data discovery and sparse optimization. *Proc. R. Soc. A* 473:20160446. doi: 10.1098/rspa.2016.0446

Schaeffer, H., and McCalla, S. G. (2017). Sparse model selection via integral terms. *Phys. Rev. E* 96:023302. doi: 10.1103/PhysRevE.96.023302

Schaeffer, H., Tran, G., and Ward, R. (2018). Extracting sparse high-dimensional dynamics from limited data. *SIAM J. Appl. Math.* 78, 3279–95. doi: 10.1137/18M116798X

Schmidt, M., and Lipson, H. (2009). Distilling free-form natural laws from experimental data. *Science* 324, 81–85. doi: 10.1126/science.1165893

Segre, M. (1980). The role of experiment in galileo's physics. *Archiv. Hist. Exact Sci.* 23, 227–252. doi: 10.1007/BF00357045

Smits, A. J., and Ogg, S. (2004). "Aerodynamics of the golf ball," in *Biomedical Engineering Principles in Sports* (New York, NY: Springer), 3–27. doi: 10.1007/978-1-4419-8887-4_1

Sorokina, M., Sygletos, S., and Turitsyn, S. (2016). Sparse identification for nonlinear optical communication systems: SINO method. *Opt. Express* 24, 30433–30443. doi: 10.1364/OE.24.030433

Su, W., Bogdan, M., and Candes, E. (2017). False discoveries occur early on the lasso path. *Ann. Stat.* 45, 2133–2150. doi: 10.1214/16-AOS1521

Sznitman, J., Stone, H. A., Smits, A. J., and Grotberg, J. B. (2017). Teaching the falling ball problem with dimensional analysis. *Eur. J. Phys. Educ.* 4, 44–54.

Tanevski, J., Simidjievski, N., Todorovski, L., and Džeroski, S. (2017). "Process-based modeling and design of dynamical systems," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (Springer), 378–382. doi: 10.1007/978-3-319-71273-4_35

Tanevski, J., Todorovski, L., and Džeroski, S. (2016). Learning stochastic process-based models of dynamical systems from knowledge and data. *BMC Syst. Biol.* 10:30. doi: 10.1186/s12918-016-0273-4

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *J. R. Stat. Soc. B Methodol.* 58, 267–288. doi: 10.1111/j.2517-6161.1996.tb02080.x

Tran, G., and Ward, R. (2016). Exact recovery of chaotic systems from highly corrupted data. *arXiv* 1607.01067.

Watters, N., Zoran, D., Weber, T., Battaglia, P., Pascanu, R., and Tacchetti, A. (2017). "Visual interaction networks: learning a physics simulator from video," in *Advances in Neural Information Processing Systems*, (Long Beach, CA), 4539–4547.

White, F. M., and Chul, R. (2011). *Fluid Mechanics, 2011*. New-York, NY: MacGraw-Hill.

Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., et al. (2008). Top 10 algorithms in data mining. *Knowl. Inform. Syst.* 14, 1–37. doi: 10.1007/s10115-007-0114-2

Yuan, M., and Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *J. R. Stat. Soc. B Stat. Methodol.* 68, 49–67. doi: 10.1111/j.1467-9868.2005.00532.x

Zheng, P., Askham, T., Brunton, S. L., Kutz, J. N., and Aravkin, A. Y. (2018). A unified framework for sparse relaxed regularized regression: Sr3. *IEEE Access* 7, 1404–423. doi: 10.1109/ACCESS.2018.2886528

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Physics

ORIGINAL RESEARCH
published: 25 June 2020
doi: 10.3389/fphy.2020.00200

Automated Discovery of Local Rules for Desired Collective-Level Behavior Through Reinforcement Learning

*Tiago Costa, Andres Laan, Francisco J. H. Heras and Gonzalo G. de Polavieja**

Collective Behavior Laboratory, Champalimaud Research, Lisbon, Portugal

Complex global behavior patterns can emerge from very simple local interactions between many agents. However, no local interaction rules have been identified that generate some patterns observed in nature, for example the rotating balls, rotating tornadoes and the full-core rotating mills observed in fish collectives. Here we show that locally interacting agents modeled with a minimal cognitive system can produce these collective patterns. We obtained this result by using recent advances in reinforcement learning to systematically solve the inverse modeling problem: given an observed collective behavior, we automatically find a policy generating it. Our agents are modeled as processing the information from neighbor agents to choose actions with a neural network and move in an environment of simulated physics. Even though every agent is equipped with its own neural network, all agents have the same network architecture and parameter values, ensuring in this way that a single policy is responsible for the emergence of a given pattern. We find the final policies by tuning the neural network weights until the produced collective behavior approaches the desired one. By using modular neural networks with modules using a small number of inputs and outputs, we built an interpretable model of collective motion. This enabled us to analyse the policies obtained. We found a similar general structure for the four different collective patterns, not dissimilar to the one we have previously inferred from experimental zebrafish trajectories; but we also found consistent differences between policies generating the different collective pattern, for example repulsion in the vertical direction for the more three-dimensional structures of the sphere and tornado. Our results illustrate how new advances in artificial intelligence, and specifically in reinforcement learning, allow new approaches to analysis and modeling of collective behavior.

Keywords: collective behavior, multi agent reinforcement learning, deep learning, interpretable artificial intelligence, explainable artificial intelligence

OPEN ACCESS

Edited by:
Raul Vicente,
Max Planck Institute for Brain Research, Germany

Reviewed by:
Chengyi Xia,
Tianjin University of Technology, China
Francisco Martinez-Gil,
University of Valencia, Spain

****Correspondence:***
Gonzalo G. de Polavieja
gonzalo.polavieja@neuro.fchampalimaud.org

Specialty section:
This article was submitted to Interdisciplinary Physics, a section of the journal Frontiers in Physics

Received: *21 November 2019*
Accepted: *05 May 2020*
Published: *25 June 2020*

Citation:
Costa T, Laan A, Heras FJH and de Polavieja GG (2020) Automated Discovery of Local Rules for Desired Collective-Level Behavior Through Reinforcement Learning. Front. Phys. 8:200. doi: 10.3389/fphy.2020.00200

1. INTRODUCTION

Complex collective phenomena can emerge from simple local interactions of agents who lack the ability to understand or directly control the collective [1–14]. Examples include cellular automata for pattern generation [3, 6, 10], self-propelled particles (SPP) [2, 4, 5, 7, 11, 13], and ant colony models for collective foraging and optimization [8, 12].

If in one of such systems we observe a particular collective configuration, how can we infer the local rules that produced it? Researchers have relied on the heuristic known as the modeling cycle

[15, 16]. The researcher first proposes a set of candidate local rules based on some knowledge of the sensory and motor capabilities of the agents. The rules are then numerically simulated and the results compared with the desired outcome. This cycle is repeated, subsequently changing the rules until an adequate match between simulated trajectories and the target collective configuration is found.

Studies in collective behavior might benefit from a more systematic method to find local rules based on known global behavior. Previous work has considered several approaches. Several authors have started with simple parametric rules of local interactions and then tuned the parameters of the interaction rules via evolutionary algorithms based on task-specific cost functions [17, 18]. When the state space for the agents is small, a more general approach is to consider tabular rules, which specify a motor command for every possible sensory state [19].

These approaches have limitations. Using simple parametric rules based on a few basis functions produces models with limited expressibility. Tabular mapping has limited generalization ability. As an alternative not suffering from these problems, neural networks have been used as the function approximator [20–22]. Specifically, neural network based Q-learning has been used to study flocking strategies [23] and optimal group swimming strategies in turbulent plumes [24]. Q-learning can however run into optimization problems when the number of agents is large [25]. Learning is slow if we use Q-functions of collective states (e.g., the location and orientation of all agents) and actions, because the dimensionality scales with the number of agents. When implementing a separate Q-function for the state and action of each agent, the learning problem faced by each agent is no longer stationary because other agents are also learning and changing their policies simultaneously [26]. This violates the assumptions of Q-learning and can lead to oscillations or sub-optimal group level solutions [27].

Despite these difficulties, very recent work using inverse reinforcement learning techniques has been applied to find interaction rules in collectives [28, 29]. These approaches approximate the internal reward function each agent is following, and require experimental trajectories for all individuals in the collective. Here, we follow a different approach in which we aim at finding a single policy for all the agents in the collective, and with the only requirement of producing a desired collective configuration.

Our approach includes the following technical ingredients. We encode the local rule as a sensorimotor transformation, mathematically expressed as a parametric policy, which maps the agent's local state into a probability distribution over an agent's actions. As we are looking for a single policy, all agents have the same parametric policy, with the same parameter values, identically updated to maximize a group level objective function (total reward during a simulated episode) representing the desired collective configuration. A configuration of high reward was searched for directly, without calculating a group-level value function and thus circumventing the problem of an exploding action space. For this search, we use a simple algorithm of the class of Evolution Strategies (ES), which are biologically-inspired algorithms for black-box optimization [30, 31]. We could have chosen other black-box optimization algorithms instead, such as particle swarm algorithms [32]. However, this ES algorithm has recently been successful when solving Multi-Agent RL problems [33], and when training neural-network policies for hard RL problems [34].

We applied this approach to find local rules for various experimentally observed schooling patterns in fish. Examples include the rotating ball, the rotating tornado [35], the full-core rotating mill [36], and the hollow-core mill [37]. To our knowledge, with the exception of the hollow-core mill [38, 39], these configurations have not yet been successfully modeled using SPP models [11].

2. METHODS

We placed the problem of obtaining local rules of motion (policies) that generate the desired collective patterns in the reinforcement learning framework [40]. As usual in reinforcement learning, agents learn by maximizing a reward signal obtained from their interaction with the environment in a closed-loop manner, i.e., the learning agents's actions influence its later inputs **Figure 1**. To describe this interaction, it is necessary to specify a model of the agents and the environment, a reward function, a policy representation and a method to find the gradient of the reward function with respect to the parameters of the policy. Both the environment update and the reward are history-independent, and thus can be described in the framework of multi-agent Markov decision processes [41]. We describe the four components (agent and environment model, reward function, policy parameterization, and learning algorithm) in the following subsections.

2.1. A Model of the Agent and the Environment

We model fish as point particles moving trough a viscous three-dimensional environment. In this section, we explain how we update the state of each and every fish agent.

Let us define a global reference frame, with Z axis parallel to the vertical, and X and Y in a horizontal plane. Length is expressed in an arbitrary unit, which we call body length (BL) because it corresponds to the body length of agents in the retina experiments that we describe in the **Supplementary Text**.

In this reference frame, we consider a fish agent moving with a certain velocity. We describe this velocity as three numbers: the speed V, elevation angle θ (i.e., its inclination angle is $\frac{\pi}{2} - \theta$) and azimuth angle ϕ. In the next time step, we update the X, Y, and Z coordinates of the fish as

$$X(t+1) = X(t) + \delta\, V(t) \cos\phi(t) \cos\theta(t), \tag{1}$$

$$Y(t+1) = Y(t) + \delta\, V(t) \sin\phi(t) \cos\theta(t), \tag{2}$$

$$Z(t+1) = Z(t) + \delta\, V(t) \sin\theta(t), \tag{3}$$

where δ corresponds to the duration of a time step (see **Table 1** and **Table S2** for parameter values).

The elevation angle, azimuth angle change, and speed change are updated based on three outputs of the policy network, p_1,

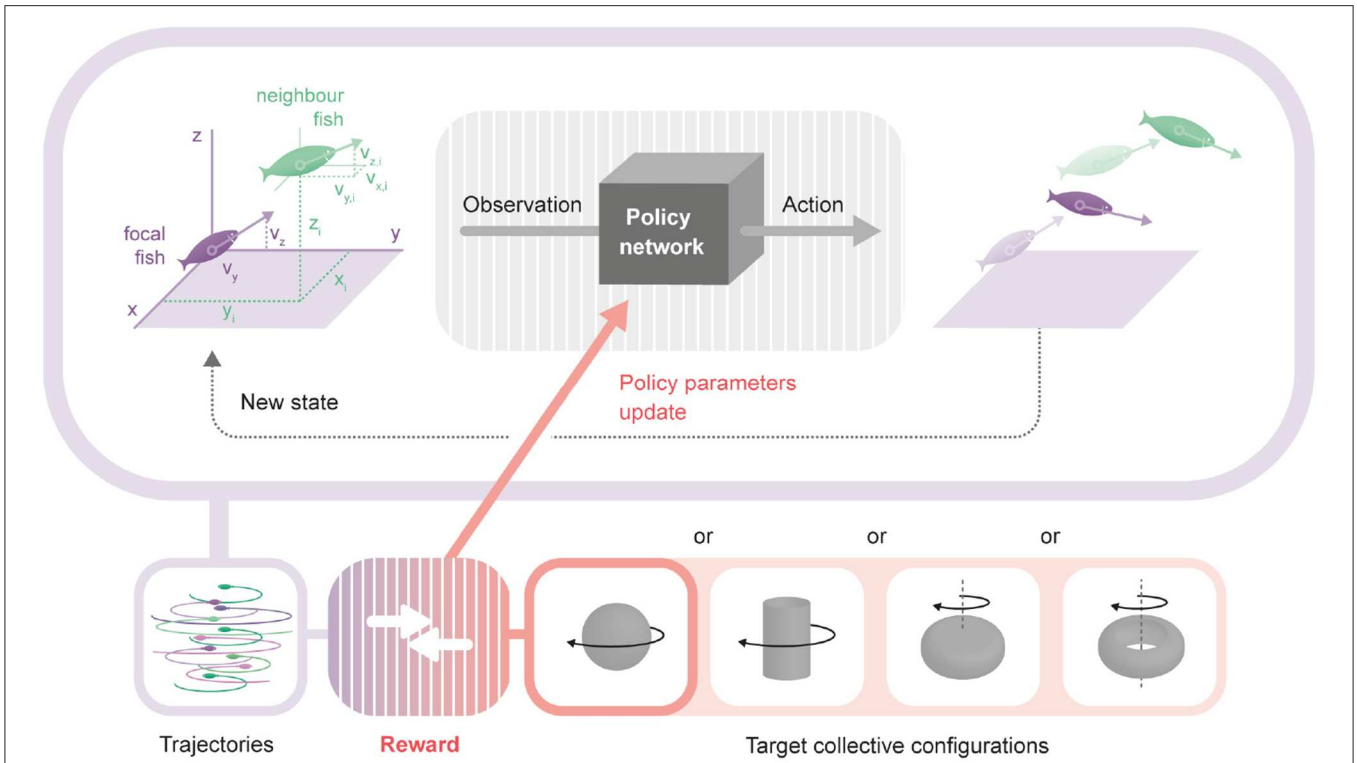


FIGURE 1 | Framework to obtain an interaction model producing a desired collective behavior. **Upper:** The policy network is a neural network that transforms the observation of an agent into its action. Its parameters are the same for all agents and chosen to produce a target collective behavior. For each focal fish (purple), the observation used to generate actions is formed by: the horizontal and vertical components of the focal fish, plus the three components of the relative position and three components of the absolute velocity of each of the neighbors. In the diagram, a single example neighbor fish is shown (green). From the observations of each focal fish, the policy network generates an action for each fish, modifying their heading and speed in the next time step according to the physics imposed for the environment. The new locations, velocities and orientations define a new state. **Lower:** Repeating the loop described above a number of times, we obtain a trajectory for each fish. We compare the trajectories with the selected desired configuration (rotating sphere, rotating tornado, rotating hollow-core mill and rotating full-core mill) producing a reward signal that it is used to change the parameters of the policy network.

TABLE 1 | Environment parameters described in the methods.

α (viscous drag)	1
ΔV_{max}	6.75 BL s^{-1}
$\Delta\phi_{max}$	$\frac{10\pi}{16}$ rad s^{-1}
θ_{max}	$\frac{\pi}{3}$ rad

p_2, and p_3, each bounded between 0 and 1. The three outputs of the policy network are independently sampled at time t from a distribution determined by the observation at time step t (see section 2.3).

The azimuth, ϕ, is updated using the first output of the policy, p_1:

$$\phi(t+1) = \phi(t) + \delta\,\Delta\phi_{max}\,2\left(p_1 - \frac{1}{2}\right), \tag{4}$$

with $\Delta\phi_{max}$ the maximum change in orientation per unit time, and δ is the time step duration.

The elevation angle, θ, is calculated based on the second output of the policy network, p_2, as

$$\theta(t) = \theta_{max}\,2\left(p_2 - \frac{1}{2}\right). \tag{5}$$

where the maximum elevation is θ_{max}

Finally, the speed change is the sum of two components: a linear viscous drag component (with parameter α) and an active propulsive thrust determined by the third output of the policy network, p_3,

$$V(t+1) = V(t) + \delta\left(\Delta V_{max}\,p_3 - \alpha V(t)\right). \tag{6}$$

The parameter ΔV_{max} is the maximum active change of speed of a fish. This equation for the change in velocity captures that deceleration in fish is achieved through the passive action of viscous forces [42].

At the beginning of each simulation, we initialize the positions and velocities of all fish randomly (see **Supplementary Text** for details). The same state update Equations (1)–(6) are applied

identically to all fish while taking into account that each one has a different position, speed, orientation and neighbors.

2.2. Reward Function

In our simulations, the final behavior to which the group converges is determined by the reward function. We aim to model four different collective behaviors, all of which have been observed in nature. These behaviors are called the rotating ball [16], the rotating tornado [35], the rotating hollow core mill [11, 37], and the rotating full core mill [36].

At each time step, the configuration of agents allows to compute an instantaneous group level reward $r(t)$. The objective of the reinforcement learning algorithm is to find ways to maximize the reward R in the episode, which is the sum of $r(t)$ over the N steps of simulation time. The instantaneous reward $r(t)$ is composed of several additive terms. In this section we will explain the terms used in the instantaneous reward function for the rotating ball, and in **Supplementary Text** we give mathematical expressions for the terms corresponding to the four collective structures.

The first term is composed of collision avoidance rewards, r_c. It provides an additive negative reward for every pair of fish (i, j) based on their mutual distance $d_{i,j}$. Specifically, for each neighbor we use a step function that is zero if $d_{i,j} > D_c$ and -1 otherwise. This term is meant to discourage the fish from moving too close to one another.

The second term is an attraction reward, r_a, which is negative and proportional to the sum of the cubed distances of all fish from the center of mass of the group. This attraction reward will motivate the fish to stay as close to the center of mass as possible while avoiding mutual collisions due to the influence of the collision reward. Together with r_c, it promotes the emergence of a dense fish ball.

The third term in the instantaneous reward, r_r, is added to promote rotation. We calculate for each fish i its instantaneous angular rotation about the center of mass in the $X - Y$ plane, Ω_i. The rotation term, r_r, is the sum of beta distributions of that angular rotation across all fish.

The fourth and final term, r_v, penalizes slow configurations. It is a step function that is 0 if the mean speed is above $V_{\min}$ and -1 otherwise. $V_{\min}$ is small enough to have a negligible effect in the trained configuration, but large enough to prevent the agents from not moving. As such, this last term encourages the agents to explore the state-action space by preventing them from remaining still.

The reward functions designed to encourage the emergence of a rotating tornado and the rotating mills are described in the **Supplementary Text** but they in general consist of similar terms.

Unlike previous work in which each agent is trying to maximize an internal reward function [28, 29], we defined the reward functions globally. Although each agent is observing and taking actions independently, the collective behavior is achieved by rewarding the coordination of all the agents, and not their individual behaviors.

2.3. The Policy Network

We parameterize our policy as a modular neural network with sigmoid activation functions, **Figure 2**. In our simulations, all fish in the collective are equipped with an individual neural network. Each network receives the state observed by the corresponding agent and outputs an action that will update its own position and velocity.

All the networks have the same weight values, but variability in the individual behaviors is still assured for two reasons. First, we use stochastic policies, which makes sense biologically, because the same animal can react differently to the same stimulus. In addition, a stochastic policy enables a better exploration of the state-action space [43]. Second, different fish will still act following different stochastic distributions if they have different surroundings. In the next section we will describe these networks and the implementation of stochasticity in the policy.

2.3.1. Inputs and Outputs

At each time step, the input to the network is information about the agent surroundings. For each focal fish, at every time step we consider an instantaneous frame of reference centered on the focal fish, with the z axis parallel to the global Z axis, and with the y axis along the projection of the focal velocity in the $X - Y$ plane. For each neighbor, the variables considered are x_i, y_i, z_i (the components of the neighbor i position in the new frame of reference) and $v_{x,i}$, $v_{y,i}$, $v_{z,i}$ (the components of the neighbor velocity in the new frame of reference). In addition, we also use v_y and v_z (the components of the focal fish velocity in the new frame of reference). Please note that the frame is centered in the focal fish, but it does not move nor rotate with it, so all speeds are the same as in the global frame of reference.

The policy network outputs three numbers, p_1, p_2, and p_3 (see next section for details), that are then used to update the agent's azimuth, elevation angle and speed, respectively.

2.3.2. Modular Structure

To enable interpretability, we chose a modular structure for the policy neural network. Similar to our previous work [44], we chose a network architecture with two modules, **Figure 2**. Both modules have the 8 inputs we detailed above and two hidden layers of 100 and 30 neurons.

The first module, the pairwise-interaction module, contains 6 output neurons, **Figure 2A**. For each neighbor i, they produce 6 outputs $\{p^{\mu}_{j,i}, p^{\sigma}_{j,i}\}_{j=1,2,3}$. The outputs are symmetrized with respect to reflections in the $x - y$ and $y - z$ planes, with the exception of $p_{1,i}$ (mean azimuth angle change, anti-symmetrized with respect to the $x - y$ plane) and $p_{2,i}$ (mean elevation change, anti-symmetrized with respect to the $y - z$ plane).

The previous values, $\{p^{\mu}_{j,i}, p^{\sigma}_{j,i}\}_{j=1,2,3}$, encode the mean and the scaled variance of three normal distributions (clipped between 0 and 1) used to sample three variables,

$$p_{j,i} \sim N(p^{\mu}_{j,i},\ 0.48\, p^{\sigma}_{j,i} + 0.02). \tag{7}$$

For each neighbor, i, we sample values of $p_{1,i}, p_{2,i}$ and $p_{3,i}$ independently from the respective distributions, **Figure 2B**.

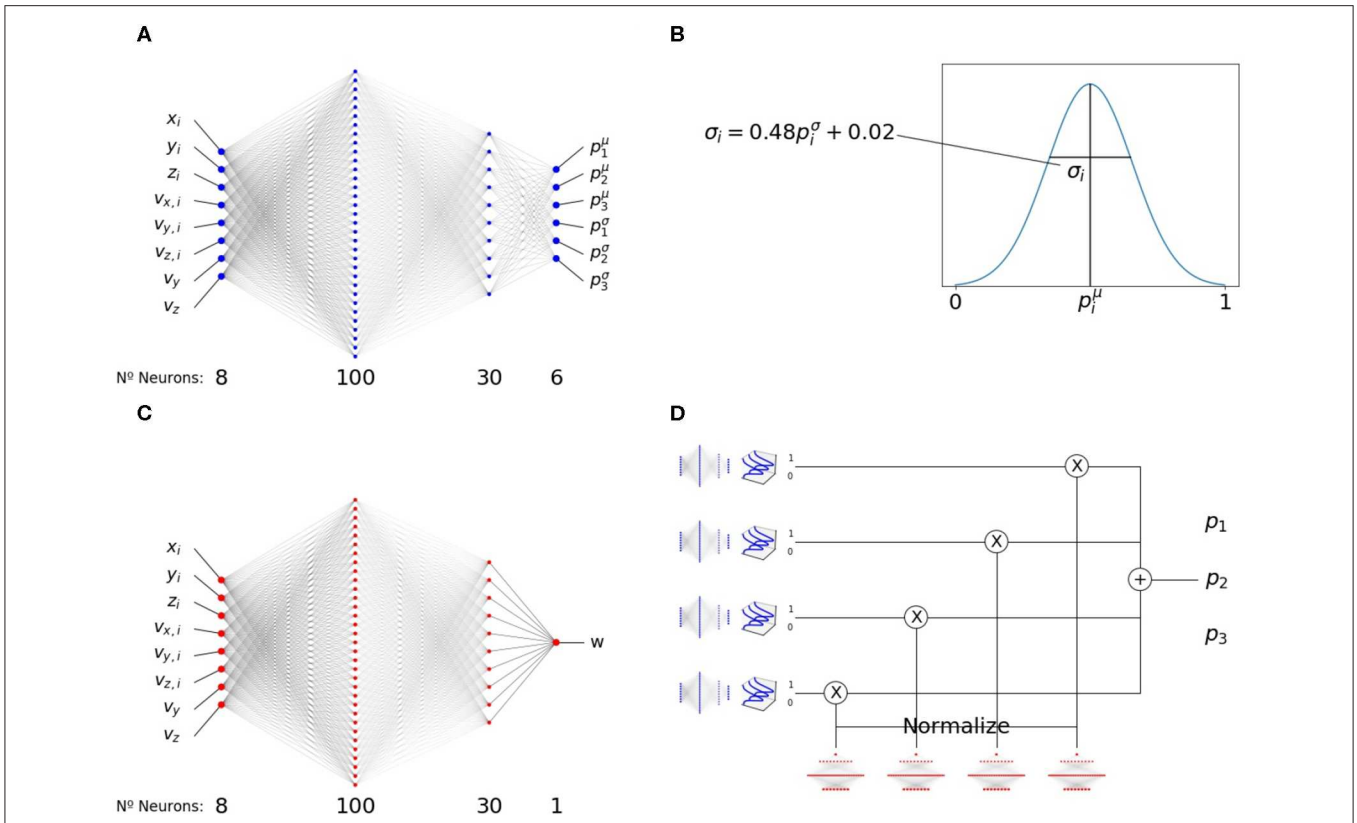


FIGURE 2 | Modular structure of the policy network. **(A)** Pair-interaction module, receiving asocial variables, v_y and v_z, and social variables, $x_i, y_i, z_i, v_{x,i}, v_{y,i}, v_{z,i}$, from a single neighbor i, and outputting 6 numbers (three mean values and three variance values). **(B)** Three numbers are sampled from three clipped Gaussian distributions with mean and variances given by the output of the pair-interaction module. **(C)** Aggregation module. Same structure as A, but with a single output that is passed through an exponential function to make it positive. **(D)** Complete structure of the modular network, showing how the outputs of the pair-interaction and aggregation modules are integrated. The resulting output, p_1, p_2, p_3, determines the heading and speed of the focal fish in the next time step.

The second module, the aggregation module, has a single output, W (**Figure 2C**). It is symmetrized with respect to both the $x-y$ and $y-z$ planes. It is clipped between -15 and 15 and there is an exponential non-linearity after the single-neuron readout signal to make it positive.

The final output combines both modules, **Figure 2D**. It is calculated by summing the outputs of the pairwise-interaction modules applied to the set of all neighbors, $\mathcal{I}$, using the outputs of the aggregation module as normalized weights,

$$P = \sum_{i\in\mathcal{I}} P_i \frac{W_i}{\sum_{j\in\mathcal{I}} W_j} \tag{8}$$

where we combined $p_{1,i}, p_{2,i}$, and $p_{3,i}$ as components of a vector P_i. The final outputs used to update the dynamics of the agent, p_1, p_2, p_3 are the components of P.

Everywhere in this paper, the set of neighbors considered, $\mathcal{I}$, consists of all the other fish in the same environment as the focal. Even if this is the case, note that the introduction of the aggregation module acts as a simulated attention that selects which neighbors are more relevant for a given state and policy.

2.4. Optimizing the Neural Network Parameters

Following previous work [33, 34], we improved the local rule using an "Evolution Strategies" algorithm [30, 31]. The text in this section is an explanation of its main elements.

Let us denote by $\vec{\omega}$ the neural network weights at every iteration of the algorithm, and by $R(\vec{\omega})$ the reward obtained in that iteration (sum during the episode). A change in parameters that improves the reward can be obtained by following the gradient with small steps (gradient ascent),

$$\vec{\omega} \leftarrow \vec{\omega} + \lambda \vec{\nabla} R(\vec{\omega}), \tag{9}$$

where λ is the learning rate (see **Supplementary Text** for the values of hyper-parameters in our simulations). Repeating this gradient ascent on the reward function, we approach the desired collective behavior over time. As we explain below, we perform this gradient while co-varying the parameters of all agents. In contrast, the naive application of policy gradient would be equivalent to performing the gradient with respect to the parameters of one of the agents, keeping the parameters of others constant. This could produce learning inefficiencies or even failure to find the desired policy.

We estimate the gradient numerically from the rewards of many simulations using policy networks with slightly different parameters. We first sample K vectors $\vec{\epsilon}_i$ independently from a spherical normal distribution of mean 0 and standard deviation σ, with as many dimensions as parameters in the model. We define $\overrightarrow{\epsilon_{i+K}} = -\vec{\epsilon}_i$, $i = 1,...,K$. We calculate $2K$ parameter vectors $\vec{\omega}_i = \vec{\omega} + \vec{\epsilon}_i$. We then conduct a single simulation of a fish collective —all agents in the same environment sharing the same values $\vec{\omega}_i$— and we record the reward R_i. Then, we use $\frac{1}{2\sigma^2 K}\sum_i \vec{\epsilon}_i R_i$ as an approximation of $\vec{\nabla}R(\vec{\omega})$ to perform gradient ascent [34].

$$\vec{\omega} \leftarrow \vec{\omega} + \lambda \frac{1}{2\sigma^2 K}\sum_{i=1}^{2K} \vec{\epsilon}_i R_i \tag{10}$$

We refer to **Figure 3** for four example training runs using the algorithm, and to **Figure 4** for the pseudo-code of the algorithm.

2.5. Measurements of XY-Plane Interaction

As in previous work [44], we described changes in the azimuth angle using the approximate concepts of attraction, repulsion, and alignment. We define the attraction-repulsion and the alignment score as useful quantifications of these approximate concepts. Please note that these scores are not related to reward.

We obtained the attraction-repulsion and alignment scores from a centered and scaled version of p_1^μ:

$$\hat{p}_{1,i}(\phi_i) = 2\left(p^\mu_{1,i}(\phi_i) - \frac{1}{2}\right) \tag{11}$$

where we chose to only explicitly highlight its dependence with the relative neighbor orientation in the XY plane, ϕ_i. This relative neighbor orientation can be calculated as the difference of the azimuth angle of the neighbor and the azimuth angle of the focal fish.

Attraction-repulsion score is defined by averaging $\hat{p}_{1,i}$ over all possible relative orientations of the neighbor in the XY plane.

$$\text{sign}(x)\langle \hat{p}_{1,i}(\phi_i)\rangle_{\phi_i\in[-\pi,\pi)}, \tag{12}$$

We would say there is attraction (repulsion) when the score is positive (negative).

The alignment score is defined as

$$\max_{\phi_i\in[-\pi,\pi)}\{\hat{p}_{1,i}(\phi_i)\,\text{sign}(\phi_i)\} - \max_{\phi_i\in[-\pi,\pi)}\{-\hat{p}_{1,i}(\phi_i)\,\text{sign}(\phi_i)\}. \tag{13}$$

As in [44], we arbitrarily decided that alignment is dominant (and thus that the point is in the alignment area) if $\hat{p}_{1,i}$ changes sign when changing the relative orientation of the neighbor in the XY plane, ϕ_i. Otherwise, it is in an attraction or repulsion area, depending on the sign of the attraction-repulsion score [44]. Under this definition, repulsion (attraction) areas are the set of possible relative positions of the neighbor which would make a focal fish turn away (toward) the neighbor, independently of the neighbor orientation relative to the focal fish

3. RESULTS

To simulate collective swimming, we equipped all fish with an identical neural network. At each time step, the neural network analyzes the surroundings of each fish and produces an action for that fish, dictating change in its speed and turning, **Figure 1**. Under such conditions, the neural network encodes a local rule and by varying the weights within the network, we can modify the nature of the local rule and thus the resulting group level dynamics.

As in previous work [44], we enabled interpretability by using a neural network built from two modules with a few inputs and few outputs each, **Figure 2**. A pairwise-interaction module outputs turning and change of speed with information from a single neighbor, i, at a time. It is composed of two parts. The first part outputs in a deterministic way mean values and variances for each of the three parameters encoding turning and change of speed, **Figure 2A**. The second part consists in sampling each parameter, $p_{1,i}, p_{2,i}, p_{3,i}$, from a clipped Gaussian distribution with the mean and variance given by the outputs of the first part, **Figure 2B**.

An aggregation module outputs a single positive number expressing the importance carried by the signal of each neighbor, **Figure 2C**. The final outputs of the complete modular neural network are obtained by summing the results of the pairwise-interaction module, weighting them by the normalized outputs of the aggregation module, **Figure 2D**. The final outputs, p_1, p_2 and p_3, determine the motor command. We perform these computations for each agent, and use the outputs to determine the position and speed of each agent in the next time step (Equations 1–3).

We introduced a reward function, measuring how similar are the produced trajectories to the desired group behavior (see section 2 for details). We used one of four different reward functions to encourage the emergence of one of four different collective configurations, all of which have been observed in natural groups of fish. These patterns are the rotating ball, the rotating tornado, the rotating hollow core mill and the rotating full core mill.

We used evolutionary strategies to gradually improve the performance of the neural network at the task of generating the desired collective configurations. The value of the reward function increased gradually during training for all four patterns, **Figure 3**. After several thousand time steps, the reward plateaued and the group collective motion was visually highly similar to the desired one (see **Supplementary Videos 1–4**). We tested that the agents learned to generate the desired collective configurations by using independent quantitative quality indexes (**Supplementary Text** and **Figure S1**). They show that agents learn first to come together into a compact formation, and then to move in the right way, eventually producing the desired configurations. We also checked that the configurations are formed also when the number of agents is different to the number used in training (see **Supplementary Videos 5–8** for twice as many agents as those used in training).

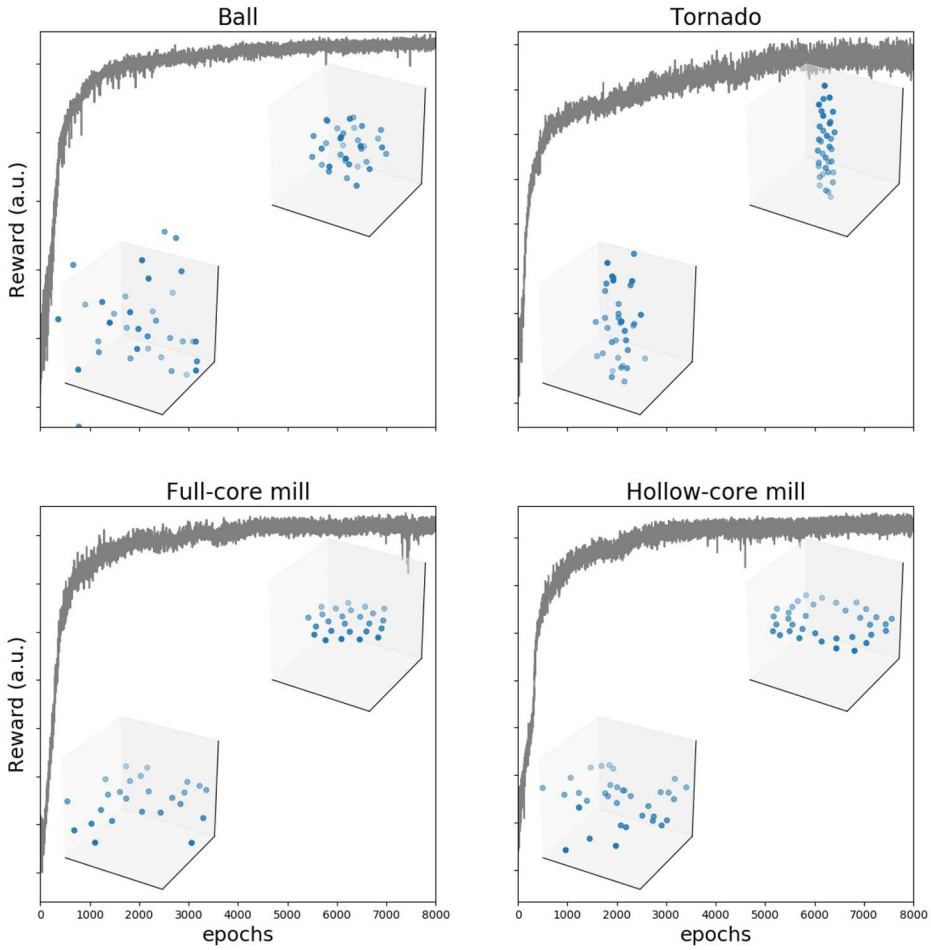


FIGURE 3 | Training makes reward to increase and group behavior to converge to the desired configuration. Reward as a function of number of training epochs in an example training cycle for each of the four configurations. In each of the examples, we show two frames (agents shown as blue dots) from the generated trajectories, one early in the training process (100 epochs) and a second after the reward plateaus (8,000 epochs).

3.1. Description of Rotating Ball Policy

Here, we use the low dimensionality of each module in terms of inputs and outputs to describe the policy with meaningful plots. We describe here the policy of the rotating ball (**Figure 5**). The equivalent plots for the other three configurations can be found in **Figures S2–S4**.

The pairwise-interaction module outputs three parameters for each focal fish, all bounded between 0 and 1. The first one, p_1, determines the change in azimuth, that is, rotations in the XY plane (**Figure 5**, first column). To further reduce dimensionality in the plots, we simplify the description of the policy in this XY plane by computing attraction-repulsion and orientation scores (see Equations 12, 13, section 2). These scores quantify the approximate concepts of attraction, repulsion and orientation. In **Figure 5**, we plot the alignment score in the areas where alignment is dominant, and the attraction-repulsion score otherwise.

The attraction areas give the neighbor positions in this XY plane which make a focal fish (located at $x_i = y_i = 0$) swim toward the neighbor, independently of the neighbor orientation (**Figure 5**, first column, orange). The focal fish turns toward the neighbor if the neighbor is far (> 1.5 BL). Repulsion areas are the relative positions of the neighbor which make a focal fish swim away from the neighbor (**Figure 5**, first column, purple). If the neighbor is closer than 1 BL, but not immediately in front or at the back of the focal, the focal tends to turn away from the neighbor (purple). In the areas where alignment is dominant (alignment areas), we plotted the alignment score (**Figure 5**, first column, gray). If the neighbor is at an intermediate distance, or in front or in the back of the focal fish, the focal fish tends to orient with respect to the neighbor in the XY plane.

The second parameter, p_2, determines the elevation angle (**Figure 5**, second column). p_2 is 0.5 on average, and thus elevation angle is zero on average, when the neighbor is on the same XY plane as the focal (**Figure 5**, second column, second row). When the neighbor is above, the focal agent tends to choose a negative elevation when the neighbor is close (**Figure 5**, second column, blue), and a positive elevation if it is far (red). The

$\vec{\omega}^0 \leftarrow$Xavier initialization
for $i = 1, ..., n$ epochs **do**
  for $j = 1, ..., K$ random perturbation **do**
    sample $\epsilon_j \leftarrow \sigma \overrightarrow{N(0, I)}$
    reset environment to random initialization $\mathcal{S}_{i,j}$
    compute R_j^+, reward of simulation with parameter $\vec{\omega}^{i-1} + \epsilon_j$
    reset environment to previous initialization $\mathcal{S}_{i,j}$
    compute R_j^-, reward of simulation with parameter $\vec{\omega}^{i-1} - \epsilon_j$
  end for
  normalize $\{R_j\}_{1,...K}$
  $\vec{\omega}^i \leftarrow \vec{\omega}^{i-1} + \lambda \frac{1}{2\sigma^2 K} \sum \vec{\epsilon}_j (R_j^+ - R_j^-)$
  anneal learning rate: $\lambda \leftarrow \gamma_\lambda \lambda$
  anneal s.d. of perturbations: $\sigma \leftarrow \gamma_\sigma \sigma$
end for

FIGURE 4 | Evolution strategies algorithm.

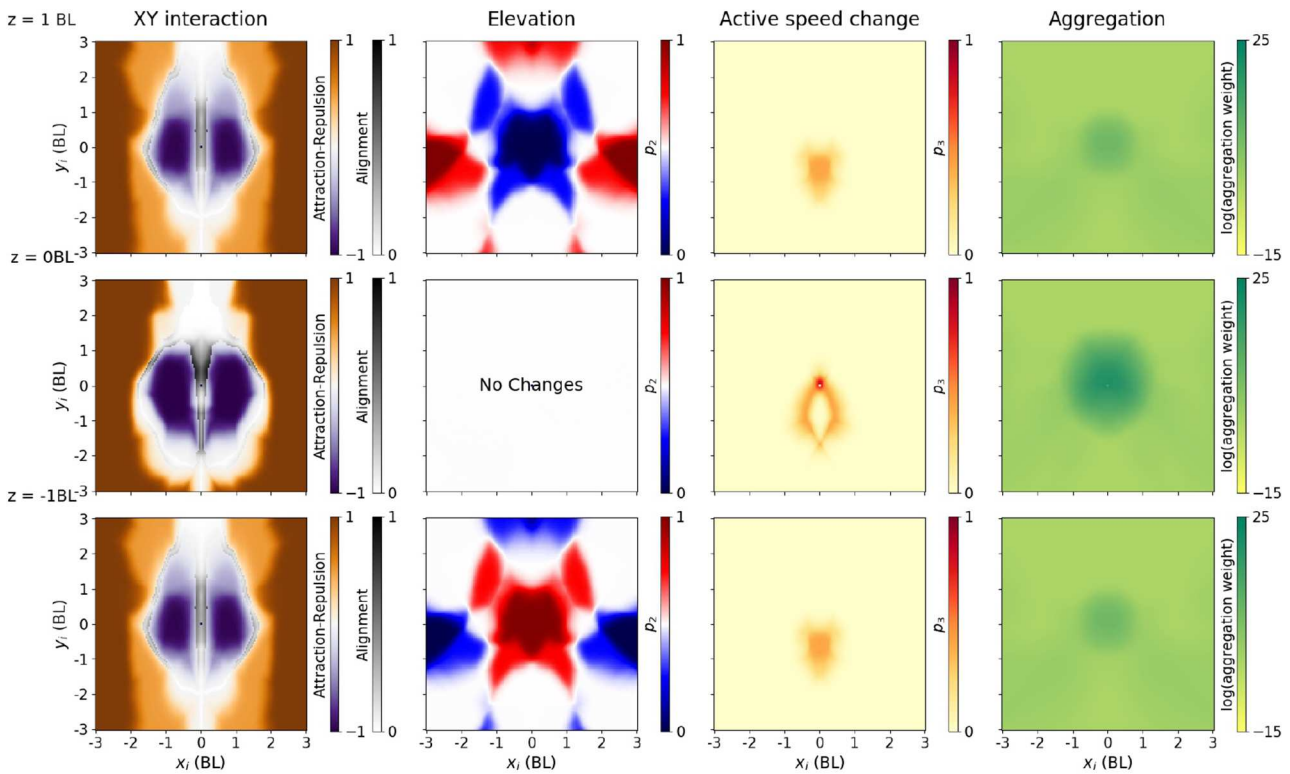


FIGURE 5 | Policy producing a rotating ball, as a function of neighbor relative location. Each output (three from the pair-interaction, one from the aggregation) is shown in a different column. All columns have three diagrams, with the neighbor 1 BL above (top row), in the same *XY* plane (middle row) and 1 BL below the focal (bottom row). Speed of both the focal and neighbor has been fixed to the median in each configuration. In addition, in columns 2–4, we plot the average with respect to an uniform distribution over all possible relative neighbor orientations (from $-\pi$ to π). **First column:** Interaction in the XY plane (change of azimuth). Instead of plotting p_1, we explain interaction using the approximate notions of alignment (gray), attraction (orange), and repulsion (purple) areas, as in [44]. Alignment score (gray) measures how much the azimuth changes when changing the neighbor orientation angle, and it is computed only in the orientation areas (see Methods, Section 2.5). Attraction (orange) and repulsion scores (purple) measure how much the azimuth change when averaging across all relative orientation angles, and we plot it only outside orientation areas. **Second column:** Elevation angle, through the mean value of the p_2 parameter. Blue areas indicate that the focal fish will move downwards ($p_2 < 0.5$), while red areas indicate that the focal fish will move upwards ($p_2 > 0.5$). **Third column:** Active change in speed, through the mean value of the p_3 parameter. Darker areas (large mean p_3) indicate increase in speed, and lighter areas indicate passive coast. **Fourth column:** Output of the aggregation module. Neighbors in the darker areas weight more in the aggregation.

opposite happens when the neighbor is below: the focal agent tends to choose a positive elevation when the neighbor is close, and a negative elevation if it is far (**Figure 5**, second column, third row).

The third parameter, p_3, determines the active speed change (**Figure 5**, third column). The active speed change is small, except if the neighbor is in a localized area close and behind the focal.

The aggregation module outputs a single positive output, determining the weight of each neighbor in the final aggregation. In the rotating ball policy, the neighbors that are weighted the most in the aggregation are the ones closer than 1 BL from the focal (**Figure 5**, third column). Neighbors located in a wide area behind the focal, but not exactly behind it, are assigned a moderate weight.

Note that the aggregation module is not constrained to produce a local spatial integration, since the network has access to every neighboring fish. However, we can observe how an aggregation module like the one shown for the rotating ball (**Figure 5**, third column) would preferably select a subset of closest neighbors—i.e., integration is local in space. This is also true for the other configurations: for each desired collective configuration we could always find policies with local integration (e.g., **Figures S2–S4**).

3.2. Description of Policy Differences Between Configurations

In the previous section, we described the policy we found to best generate a rotating ball. The policies we found that generate the other three configurations have many similarities and some consistent differences, **Figure 6**. In this section, we will highlight these differences.

The policy generating a tornado has an attraction-repulsion pattern somewhere in between the rotating ball and the full core milling (**Figure 6**, first row). The major feature that distinguishes this policy from the others is a strong repulsion along the z-axis, i.e., the area of the plot where the focal fish changes elevation to move away from the neighbor is larger (**Figure 6**, second column, second row, blue area). This was expected, as to form the tornado the agents need to spread along the z-axis much more than they do in the other configurations.

The policy generating a full-core mill has an increased repulsion area, particularly in the frontal and frontal-lateral areas (**Figure 6**, third column, first row, purple areas). The policy generating a hollow-core mill has an increased alignment area, weakening the repulsion area (**Figure 6**, fourth column, first row, gray area). It also has an increased area with a high change in velocity (**Figure 6**, fourth column, third row, red area). Both mill policies have an extended aggregation area, especially the hollow core (**Figure 6**, third and fourth column, fourth row), and lack an area where the focal fish changes elevation to move away from the neighbor (**Figure 6**, third and fourth column, second row, red area). The almost absence of repulsion in the z-direction makes both mills to be 2D structures, whereas repulsion in the sphere and tornado makes them 3D.

The highlighted differences between policies are robust (see **Figure 5** and the Supplementary Material for **Figures S2–S4** and **Figures S6–S13** to see the robustness of the results in different runs). For instance, we have compared the policies by restricting the fish to move at the median speed is each configuration. The highlighted differences are still valid when the fish move with a common median speed across configurations (**Figure S5**).

3.3. Adding a Retina to the Agents

In the preceding section, the observations made by each agent were simple variables like position or velocities of neighbors. This simplification aided analysis, but animals do not receive external information in this way but by sensory organs.

We checked whether we could achieve the group configuration we have studied when the input to the policy for each agent is the activation of an artificial retina observing the other agents. The retina is modeled using a three-dimensional ray-tracing algorithm: from each agent, several equidistant rays project until they encounter a neighbor, or up to a maximum ray length r. The state, i.e., the input to the policy, is the list of the ray lengths. Information about the relative velocity was also given as input to the policy by repeating this computation with the current position and orientation of the focal, but the previous position of all the other agents. See **Supplementary Text** for a detailed description.

We approximated the policy using a single fully-connected network. Using the interaction and attention modules described in section 2.3.2 would not have added interpretability in this case, because the number of inputs is too large. By using the same evolutionary strategy, we were able to obtain a decision rule leading to the desired collective movement configurations (**Supplementary Videos 9–12**).

Although these configurations were qualitatively similar to the ones we obtained with the modular network (**Figure S14**), the average inter-agent distance was greater. This resulted in less compact configurations (**Tables S3, S5**). This effect might be the result of the increased complexity and decreased accuracy of the inputs given to the policy by the retina, which causes the agents to avoid each other more than in simulations without retina.

4. DISCUSSION

We have applied evolutionary strategies (ES) to automatically find local rules able to generate desired group level movement patterns. Namely, we found local rules that generate four complex collective motion patterns commonly observed in nature [35, 45]. Three of these four patterns have, to our knowledge, not yet been generated using self-propelled particle models with local rules.

We used neural networks as approximators of the policy, the function mapping the local state to actions. The naive use of a neural network would produce a black-box model, that can be then analyzed with different *post-hoc* explainability strategies (inversion [46], saliency analysis [47], knowledge distillation [48], etc.; see [49, 50] for reviews). Instead, as we did in [44], we designed and trained a model that is inherently transparent (interpretable). This alternative improves our confidence in the model understandability: the model is its own exact explanation [51].

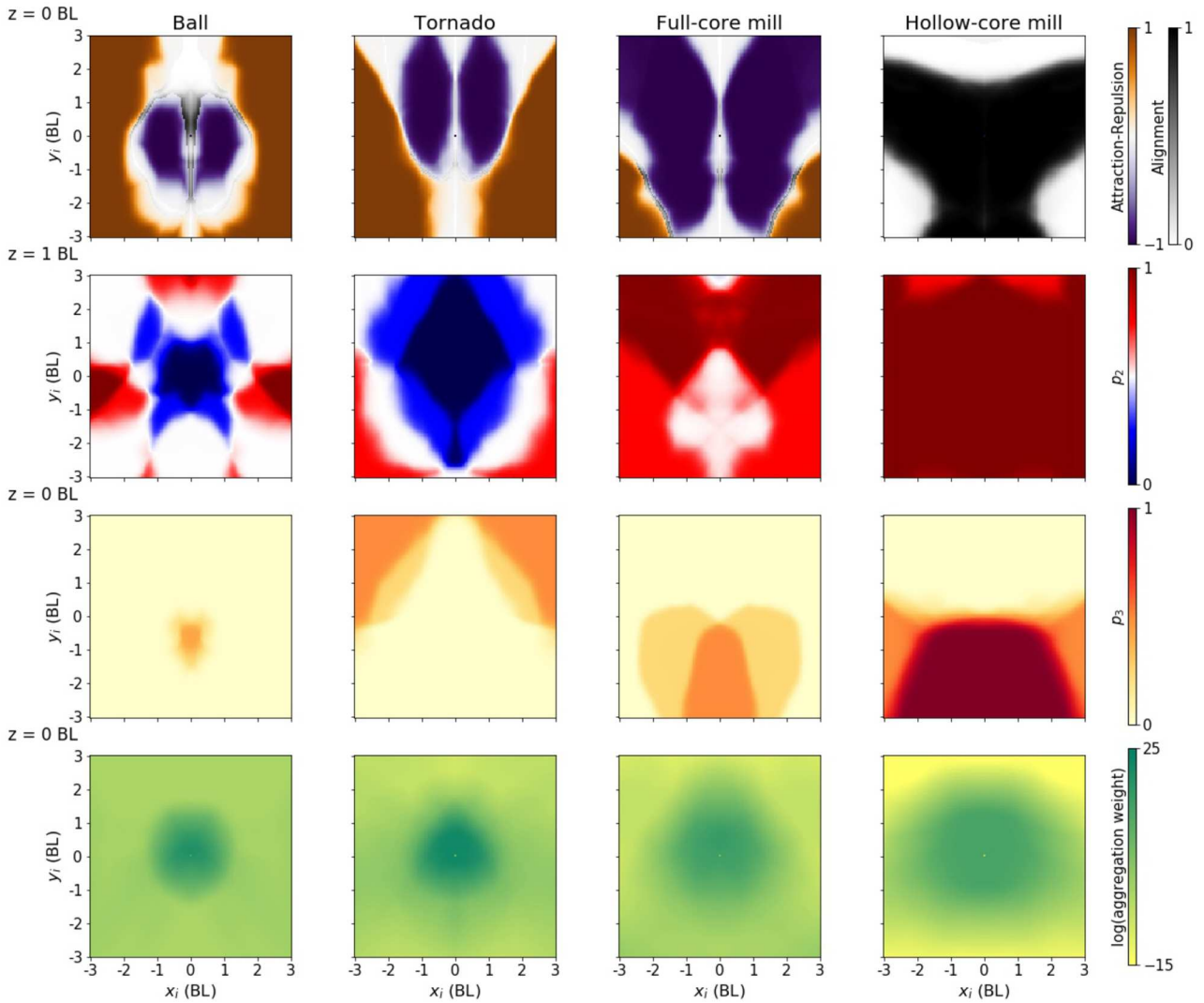


FIGURE 6 | Policies producing different configurations. Each column corresponds to one of the four desired configurations. Each panel is the equivalent of the middle row in **Figure 5**, except in the case of the elevation (second column), that is equivalent to the panel in the top row. For each configuration, the speed of the focal and neighbor has been fixed to the configuration median: 0.42, 0.32, 0.75, and 1.54 BL/s, respectively.

We used a modular policy network, composed by two modules. Each module is an artificial neural network with thousands of parameters, and therefore it is a flexible universal function approximator. However, we can still obtain insight, because each module implements a function with low number of inputs and outputs that we can plot [44]. Similar to what we obtained from experimental trajectories of zebrafish [44], we found the XY-interaction to be organized in areas of repulsion, orientation, and attraction, named in order of increasing distance from the focal fish. We were also able to describe differences between the policies generating each of the configurations.

To find the local rules generating the desired configurations, we used a systematic version of the collective behavior modeling cycle [15]. The traditional collective behavior modeling cycle begins with the researcher proposing a candidate rule and tuning it through a simulation based feedback process. Here, we parameterize the local rule as a neural network. Since neural networks are highly expressive function approximators which can capture a very diverse set of possible local rules, our method automates the initial process of finding a set of candidate rules. As in the case of the traditional modeling cycle, our method also relies on a cost function (the reward function) and numerical simulations to measure the quality of a proposed rule. The process of rule adaptation is automated by following a gradient of the reward function with respect to neural network weights. Just like the modeling cycle, our method uses iterations that gradually improve the policy until it converges to a satisfactory solution.

There are theoretical guarantees for convergence in tabular RL, or when linear approximators are used for the value functions

[40]. However, these theoretical guarantees do not normally extend to the case where neural networks are used as function approximators [52], nor to multi-agent RL [25]. Here we report an application where the ES algorithm was able to optimize policies parameterized by deep neural networks in multi-agent environments. To our knowledge, there were no theoretical guarantees for convergence for our particular setting.

The method we have proposed could have several other interesting applications. In cases where it is possible to record rich individual level data sets of collective behavior, it can be possible to perform detailed comparisons between the rules discovered by our method and the ones observed in experiments [44, 53]. The method could also be applicable to answer more hypothetical questions such as what information must be available in order for a certain collective behavior to emerge. Animals may interact in a variety of ways including visual sensory networks [54], vocal communication [55], chemical communication [56] and environment modification (stigmergy) [57]. Animals also have a variety of cognitive abilities such as memory and varying sensory thresholds. By removing or incorporating such capabilities into the neural networks it is now possible to theoretically study the effects these factors have on collective behavior patterns.

Here we relied on an engineered reward function because the behaviors we were modeling have not yet been recorded in quantitative detail. In cases where trajectory data is available, detailed measures of similarity with observed trajectories can be used as a reward [33, 58]. Moreover, we can use adversarial classifiers to automatically learn these measures of similarity [19, 59]. Further interesting extensions could include creating diversity within the group by incorporating several different neural networks into the collective and studying the emergence of behavioral specialization and division of labor [60].

The present work may be used as a normative framework when the rewards used represent important biological functions. While prior work using analytic approaches has been successful for simple scenarios [42, 61], the present approach can extend them to situations in which no analytic solution can be obtained.

DATA AVAILABILITY STATEMENT

The datasets were generated using the software from https://gitlab.com/polavieja_lab/rl_collective_behaviour.

AUTHOR CONTRIBUTIONS

AL and GP devised the project. TC, AL, FH, and GP developed and verified analytical methods. TC wrote the software, made computations and plotted results with supervision from AL, FH, and GP. All authors discussed the results and contributed to the writing of the manuscript.

FUNDING

This work was supported by Fundação para a Ciência e a Tecnologia (www.fct.pt) PTDC/NEU-SCC/0948/2014 (to GP and contract to FH), Congento (congento.org) LISBOA-01-0145-FEDER-022170, NVIDIA (nvidia.com) (FH and GP), and Champalimaud Foundation (fchampalimaud.org) (GP). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

ACKNOWLEDGMENTS

We are grateful to Francisco Romero-Ferrero and Andreas Gerken for discussions.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fphy.2020.00200/full#supplementary-material

Supplementary Video 1 | Simulation of the agents trained to adopt a Rotating ball, with the modular deep networks. Here the number of agents is the same as the number of agents used in the training.

Supplementary Video 2 | Simulation of the agents trained to adopt a Tornado, with the modular deep networks. Here the number of agents is the same as the number of agents used in the training.

Supplementary Video 3 | Simulation of the agents trained to adopt a Full core milling, with the modular deep networks. Here the number of agents is the same as the number of agents used in the training.

Supplementary Video 4 | Simulation of the agents trained to adopt a Hollow core milling, with the modular deep networks. Here the number of agents is the same as the number of agents used in the training.

Supplementary Video 5 | Simulation of the agents trained to adopt a Rotating ball, with the modular deep networks. Here the number of agents is 70 while the number of agents used in training is 35.

Supplementary Video 6 | Simulation of the agents trained to adopt a Tornado, with the modular deep networks. Here the number of agents is 70 while the number of agents used in training is 35.

Supplementary Video 7 | Simulation of the agents trained to adopt a Full core milling, with the modular deep networks. Here the number of agents is 70 while the number of agents used in training is 25.

Supplementary Video 8 | Simulation of the agents trained to adopt a Hollow core milling, with the modular deep networks. Here the number of agents is 70 while the number of agents used in training is 35.

Supplementary Video 9 | Simulation of the agents trained to adopt a Rotating ball, when the network received the activation of a simulated retina as input.

Supplementary Video 10 | Simulation of the agents trained to adopt a tornado, when the network received the activation of a simulated retina as input.

Supplementary Video 11 | Simulation of the agents trained to adopt a full core milling, when the network received the activation of a simulated retina as input.

Supplementary Video 12 | Simulation of the agents trained to adopt a hollow core milling, when the network received the activation of a simulated retina as input.

REFERENCES

1. Smith A. *An Inquiry into the Nature and Causes of the Wealth of Nations*. Vol. 2. London: W. Strahan and T. Cadell (1776).
2. Aoki I. A simulation study on the schooling mechanism in fish. *Bull Japan Soc Sci Fish*. (1982) **48**:1081–8. doi: 10.2331/suisan.48.1081
3. Wolfram S. Statistical mechanics of cellular automata. *Rev Modern Phys*. (1983) **55**:601. doi: 10.1103/RevModPhys.55.601
4. Vicsek T, Czirók A, Ben-Jacob E, Cohen I, Shochet O. Novel type of phase transition in a system of self-driven particles. *Phys Rev Lett*. (1995) **75**:1226. doi: 10.1103/PhysRevLett.75.1226
5. Helbing D, Molnar P. Social force model for pedestrian dynamics. *Phys Rev E*. (1995) **51**:4282. doi: 10.1103/PhysRevE.51.4282
6. Chopard B, Droz M. *Cellular Automata*. Cambridge, MA: Springer (1998).
7. Levine H, Rappel WJ, Cohen I. Self-organization in systems of self-propelled particles. *Phys Rev E*. (2000) **63**:017101. doi: 10.1103/PhysRevE.63.017101
8. Bonabeau E, Dorigo M, Theraulaz G. Inspiration for optimization from social insect behaviour. *Nature*. (2000) **406**:39. doi: 10.1038/35017500
9. Corning PA. The re-emergence of "emergence": a venerable concept in search of a theory. *Complexity*. (2002) **7**:18–30. doi: 10.1002/cplx.10043
10. Wolfram S. *A New Kind of Science*. Vol. 5. Champaign, IL: Wolfram media (2002).
11. Couzin ID, Krause J, James R, Ruxton GD, Franks NR. Collective memory and spatial sorting in animal groups. *J Theor Biol*. (2002) **218**:1. doi: 10.1006/jtbi.2002.3065
12. Dorigo M, Birattari M. Ant colony optimization. In: Sammut C, Webb GI, editors. *Encyclopedia of Machine Learning*. Boston, MA: Springer (2011). p. 36–9. doi: 10.1007/978-0-387-30164-8_22
13. Vicsek T, Zafeiris A. Collective motion. *Phys Rep*. (2012) **517**:71–140. doi: 10.1016/j.physrep.2012.03.004
14. Xia C, Ding S, Wang C, Wang J, Chen Z. Risk analysis and enhancement of cooperation yielded by the individual reputation in the spatial public goods game. *IEEE Syst J*. (2016) **11**:1516–25. doi: 10.1109/JSYST.2016.2539364
15. Sumpter DJ, Mann RP, Perna A. The modelling cycle for collective animal behaviour. *Interface Focus*. (2012) **2**:764–73. doi: 10.1098/rsfs.2012.0031
16. Lopez U, Gautrais J, Couzin ID, Theraulaz G. From behavioural analyses to models of collective motion in fish schools. *Interface Focus*. (2012) **2**:693–707. doi: 10.1098/rsfs.2012.0033
17. Ioannou CC, Guttal V, Couzin ID. Predatory fish select for coordinated collective motion in virtual prey. *Science*. (2012) **337**:1212–5. doi: 10.1126/science.1218919
18. Hein AM, Rosenthal SB, Hagstrom GI, Berdahl A, Torney CJ, Couzin ID. The evolution of distributed sensing and collective computation in animal populations. *Elife*. (2015) **4**:e10955. doi: 10.7554/eLife.10955
19. Li W, Gauci M, Groß R. Turing learning: a metric-free approach to inferring behavior and its application to swarms. *Swarm Intell*. (2016) **10**:211–43. doi: 10.1007/s11721-016-0126-1
20. Goodfellow I, Bengio Y, Courville A, Bengio Y. *Deep Learning*. Vol. 1. Cambridge: MIT Press (2016).
21. LeCun Y, Bengio Y, Hinton G. Deep learning. *Nature*. (2015) **521**:436. doi: 10.1038/nature14539
22. Schmidhuber J. Deep learning in neural networks: an overview. *Neural Netw*. (2015) **61**:85–117. doi: 10.1016/j.neunet.2014.09.003
23. Durve M, Peruani F, Celani A. Learning to flock through reinforcement. *arXiv preprint arXiv:191101697* (2019).
24. Verma S, Novati G, Koumoutsakos P. Efficient collective swimming by harnessing vortices through deep reinforcement learning. *Proc Natl Acad Sci USA*. (2018) **115**:5849–54. doi: 10.1073/pnas.1800923115
25. Bu L, Babu R, De Schutter B. A comprehensive survey of multiagent reinforcement learning. *IEEE Trans Syst Man Cybernet C*. (2008) **38**:156–72. doi: 10.1109/TSMCC.2007.913919
26. Burk F. *A Garden of Integrals*. Vol. 31. Washington, DC: MAA (2007).
27. Nowé A, Vrancx P, De Hauwere YM. Game theory and multi-agent reinforcement learning. In: Wiering M, van Otterlo M, editors. *Reinforcement Learning*. Berlin; Heidelberg: Springer (2012). p. 441–70. doi: 10.1007/978-3-642-27645-3_14
28. Pinsler R, Maag M, Arenz O, Neumann G. Inverse reinforcement learning of bird flocking behavior. In: *ICRA Workshop*. (2018).
29. Fahad M, Chen Z, Guo Y. Learning how pedestrians navigate: a deep inverse reinforcement learning approach. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Madrid (2018). p. 819–26. doi: 10.1109/IROS.2018.8593438
30. Rechenberg I, Eigen M. *Evolutionsstrategie; Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog [Stuttgart-Bad Cannstatt] (1973).
31. Wierstra D, Schaul T, Peters J, Schmidhuber J. Natural evolution strategies. In: *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*. Hong Kong: IEEE (2008). p. 3381–7. doi: 10.1109/CEC.2008.4631255
32. Eberhart R, Kennedy J. Particle swarm optimization. In: *Proceedings of the IEEE International Conference on Neural Networks*. Perth, WA: Citeseer (1995). p. 1942–8.
33. Shimada K, Bentley P. Learning how to flock: deriving individual behaviour from collective behaviour with multi-agent reinforcement learning and natural evolution strategies. In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. New York, NY: ACM (2018). p. 169–70. doi: 10.1145/3205651.3205770
34. Salimans T, Ho J, Chen X, Sidor S, Sutskever I. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:170303864* (2017).
35. Parrish JK, Viscido SV, Grunbaum D. Self-organized fish schools: an examination of emergent properties. *Biol Bull*. (2002) **202**:296–305. doi: 10.2307/1543482
36. Tunstrøm K, Katz Y, Ioannou CC, Huepe C, Lutz MJ, Couzin ID. Collective states, multistability and transitional behavior in schooling fish. PLoS Comput Biol. (2013) **9**:e1002915. doi: 10.1371/journal.pcbi.1002915
37. Parrish JK, Edelstein-Keshet L. Complexity, pattern, and evolutionary trade-offs in animal aggregation. *Science*. (1999) **284**:99–101. doi: 10.1126/science.284.5411.99
38. Strömbom D. Collective motion from local attraction. *J Theor Biol*. (2011) **283**:145–51. doi: 10.1016/j.jtbi.2011.05.019
39. Calovi DS, Lopez U, Ngo S, Sire C, Chaté H, Theraulaz G. Swarming, schooling, milling: phase diagram of a data-driven fish school model. *N J Phys*. (2014) **16**:015026. doi: 10.1088/1367-2630/16/1/015026
40. Sutton RS, Barto AG. *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press (1998). doi: 10.1016/S1474-6670(17)38315-5
41. Boutilier C. Planning, learning and coordination in multiagent decision processes. In: *Proceedings of the 6th Conference on Theoretical Aspects of Rationality and Knowledge*. Morgan Kaufmann Publishers Inc. San Francisco, CA (1996) p. 195–210.
42. Laan A, de Sagredo RG, de Polavieja GG. Signatures of optimal control in pairs of schooling zebrafish. *Proc R Soc B*. (2017) **284**:20170224. doi: 10.1098/rspb.2017.0224
43. Plappert M, Houthooft R, Dhariwal P, Sidor S, Chen RY, Chen X, et al. Parameter space noise for exploration. *arXiv preprint arXiv:170601905* (2017).
44. Heras FJH, Romero-Ferrero F, Hinz RC, de Polavieja GG. Deep attention networks reveal the rules of collective motion in zebrafish. *PLoS Comput. Biol.* (2019) **15**:e1007354. doi: 10.1371/journal.pcbi.1007354
45. Sumpter DJ. *Collective Animal Behavior*. New Jersey, NJ: Princeton University Press (2010) doi: 10.1515/9781400837106
46. Mahendran A, Vedaldi A. Understanding deep image representations by inverting them. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Boston, MA (2015). p. 5188–96. doi: 10.1109/CVPR.2015.7299155
47. Baehrens D, Schroeter T, Harmeling S, Kawanabe M, Hansen K, MÃžller KR. How to explain individual classification decisions. *J Mach Learn Res.* (2010) **11**:1803–31. doi: 10.5555/1756006.1859912
48. Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:150302531* (2015).
49. Fan F, Xiong J, Wang G. On interpretability of artificial neural networks. *arXiv preprint arXiv:200102522*. (2020).
50. Arrieta AB, Díaz-Rodríguez N, Del Ser J, Bennetot A, Tabik S, Barbado A, et al. Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*. (2020) **58**:82–115. doi: 10.1016/j.inffus.2019.12.012

51. Rudin C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nat Mach Intell*. (2019) **1**:206–15. doi: 10.1038/s42256-019-0048-x
52. Tsitsiklis J, Van Roy B. *An Analysis of Temporal-Difference Learning with Function Approximation*. Technical Report LIDS-P-2322. Laboratory for Information and Decision Systems (1996).
53. Herbert-Read JE, Perna A, Mann RP, Schaerf TM, Sumpter DJ, Ward AJ. Inferring the rules of interaction of shoaling fish. *Proc Natl Acad Sci USA*. (2011) **108**:18726–31. doi: 10.1073/pnas.1109355108
54. Rosenthal SB, Twomey CR, Hartnett AT, Wu HS, Couzin ID. Revealing the hidden networks of interaction in mobile animal groups allows prediction of complex behavioral contagion. *Proc Natl Acad Sci USA*. (2015) **112**:4690–5. doi: 10.1073/pnas.1420068112
55. Jouventin P. *Visual and Vocal Signals in Penguins, Their Evolution and Adaptive Characters*. Boston, MA: Fortschritte der Verhaltensforschung (1982).
56. Sumpter DJ, Beekman M. From nonlinearity to optimality: pheromone trail foraging by ants. *Anim Behav*. (2003) **66**:273–80. doi: 10.1006/anbe.2003.2224
57. Theraulaz G, Bonabeau E. A brief history of stigmergy. *Artif Life*. (1999) **5**:97–116. doi: 10.1162/106454699568700
58. Cazenille L, Bredéche N, Halloy J. Automatic calibration of artificial neural networks for zebrafish collective behaviours using a quality diversity algorithm. In: *Biomimetic and Biohybrid Systems - 8th International Conference, Living Machines*. Nara (2019). p. 38–50. doi: 10.1007/978-3-030-24741-6_4
59. Ho J, Ermon S. Generative adversarial imitation learning. In: *Advances in Neural Information Processing Systems*. Red Hook, NY (2016) p. 4565–73.
60. Robinson GE. Regulation of division of labor in insect societies. *Annu Rev Entomol*. (1992) **37**:637–65. doi: 10.1146/annurev.en.37.010192.003225
61. Arganda S, Pérez-Escudero A, de Polavieja GG. A common rule for decision making in animal collectives across species. *Proc Natl Acad Sci USA*. (2012) **109**:20508–13. doi: 10.1073/pnas.1210664109

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Physics

ORIGINAL RESEARCH
published: 13 August 2020
doi: 10.3389/fphy.2020.00297

Input Redundancy for Parameterized Quantum Circuits

Francisco Javier Gil Vidal[1]* *and Dirk Oliver Theis*[1,2]*

[1] *Theoretical Computer Science, University of Tartu, Tartu, Estonia,* [2] *Ketita Labs OÜ, Tartu, Estonia*

One proposal to utilize near-term quantum computers for machine learning are Parameterized Quantum Circuits (PQCs). There, input is encoded in a quantum state, parameter-dependent unitary evolution is applied, and ultimately an observable is measured. In a hybrid-variational fashion, the parameters are trained so that the function assigning inputs to expectation values matches a target function. The *no-cloning principle* of quantum mechanics suggests that there is an advantage in redundantly encoding the input several times. In this paper, we prove lower bounds on the number of redundant copies that are necessary for the expectation value function of a PQC to match a given target function. We draw conclusions for the architecture design of PQCs.

Keywords: parameterized quantum circuits, quantum neural networks, near-term quantum computing, lower bounds, input encoding

OPEN ACCESS

Edited by:
Vctor M. Eguíluz, Institute of Interdisciplinary Physics and Complex Systems (IFISC), Spain

Reviewed by:
Gian Luca Giorgi, Institute of Interdisciplinary Physics and Complex Systems (IFISC), Spain
Wilson Rosa De Oliveira, Federal Rural University of Pernambuco, Brazil

***Correspondence:**
Francisco Javier Gil Vidal
francisco.javier.gil.vidal@ut.ee
Dirk Oliver Theis
dotheis@ut.ee

Specialty section:
This article was submitted to Quantum Computing, a section of the journal Frontiers in Physics

Received: *22 May 2020*
Accepted: *30 June 2020*
Published: *13 August 2020*

Citation:
Gil Vidal FJ and Theis DO (2020) Input Redundancy for Parameterized Quantum Circuits. Front. Phys. 8:297. doi: 10.3389/fphy.2020.00297

1. INTRODUCTION

Quantum Information Processing proposes to exploit quantum physical phenomena for the purpose of data processing. Conceived in the early 80's [1, 2], recent breakthroughs in building controllable quantum mechanical systems have led to an explosion of activity in the field.

Building quantum computers is a formidable challenge—but so is designing algorithms which, when implemented on them, are able to exploit the advantage that quantum computing is widely believed by experts to have over classical computing on some computational tasks. A particularly compelling endeavor is to make use of *near-term quantum computers*, which suffer from limited size and the presence of debilitating levels of quantum noise. The field of algorithm design for Noisy Intermediate-Scale Quantum (NISQ) computers has scrambled over the last few years to identify fields of computing, paradigms of employing quantum information processing, and commercial use-cases in order to profit from recent progress in building programmable quantum mechanical devices—limited as they may be at present [3].

One use-case area where quantum advantage might materialize in the near term is that of Artificial Intelligence [3, 4]. The hope is best reasoned for generative tasks: several families of probability distributions have been theoretically proven to admit quantum algorithms for efficiently sampling from them, while no classical algorithm is able or is known to be able to perform that sampling task. Boson sampling is probably the most widely known of these sampling tasks, even though the advantage does not seem to persist in the presence of noise (cf. [5]); examples of some other sampling procedures can be found in references [6, 7].

Promising developments have also been made available in the case of quantum circuits that can be iteratively altered by manipulation of one or several parameters: Du et al. [8] consider so-called *Parameterized Quantum Circuits (PQCs)* and find that they, too, yield a theoretical advantage for generative tasks. PQCs are occasionally referred to as *Quantum Neural Networks (QNNs)* (e.g., in [9]) when aspects of non-linearity are emphasized, or as *Variational Quantum Circuits* [10]. We stick to the term PQC in this paper, without having in mind excluding QNNs or VQCs.

The PQC architectures which have been considered share some common characteristics, but an important design question is how the input data is presented. Input data refers either to a feature vector, or to output of another layer of a larger, potentially hybrid quantum-classical neural network. The fundamental choice is whether to encode digitally or in the amplitudes of a quantum state. Digital encoding usually entails preparing a quantum register in states $|b^x\rangle$, where $b^x \in \{0,1\}^n$ is binary encoding of input datum x. Encoding in the amplitudes of a quantum state, on the other hand, refers to preparing a an n-bit quantum register in a state of the form $|\phi^x\rangle := \sum_{j=0}^{2^n-1} \phi_j(x)|j\rangle$, where ϕ_j, $j = 0,\ldots,2^n-1$ is a family of encoding functions which must ensure that $|\phi^x\rangle$ is a quantum state for each x, i.e., that $\sum_j |\phi_j(x)|^2 = 1$ holds for all x. We refer the reader to the discussion of these concepts in Schuld and Petruccione [11] for further details.

The present paper deals with redundancy in the input data, i.e., giving the same datum several times. The most straightforward concept here is that of "tensorial" encoding [12]. Here, several quantum registers are prepared in a state which is the tensor product of the corresponding number of *identical* copies of a data-encoding state, i.e., $|\phi^x\rangle\otimes\cdots\otimes|\phi^x\rangle$. For example, Mitarai et al. [13], propose the following construction: To encode a real number x close to 0, they choose the state

$$|\phi^x\rangle = R_y(\arcsin(x)/2)|0\rangle = \sin(\arcsin(x)/2)|0\rangle + \cos(\arcsin(x)/2)|1\rangle, \quad (1)$$

where $R_y(\theta) := e^{-i\theta\sigma_Y/2}$ is the 1-qubit Pauli rotation around the Y-axis (and σ_Y the Pauli matrix). But then, to construct a PQC that is able to learn polynomials of degree n in a single variable, they encode the polynomial variable x into n identical copies, $\bigotimes_{j=1}^n |\phi^x\rangle$. It is noteworthy, and the starting point of our research, that the number of times that the input, x, is encoded redundantly, depends on the complexity of the learning task.

Encoding the input several times redundantly, as in tensorial encoding, is probably motivated by the quantum no-cloning principle. While classical circuits and classical neural networks can have *fan-out*—the output of one processing node (gate, neuron, …) can be the input to several others—the no-cloning principle of quantum mechanics forbids to duplicate data which is encoded in the amplitudes of a quantum state. This applies to PQCs, and, specifically, to the input that is fed into a PQC, if the input is encoded in the amplitudes of input states.

1.1. The Research Presented in This Paper

The no-cloning principle suggests that duplicating input data redundantly is unavoidable. The research presented in this paper aims to lower bound *how often* the data has to be redundantly encoded, if a given function is to be learned. The novelty in this paper lies in establishing that these lower bounds are possible. For that purpose, the cases for which we prove lower bounds are natural, but not overly complex, thus highlighting the principle over the application.

The objects of study of this paper are PQCs of the following form. The input consists of a single real number x, which is encoded into amplitudes by applying a multi-qubit Hamiltonian evolution of the form $e^{-i\eta(x)H}$ at one point (*no redundancy*), or several points in the quantum circuit. The function η and Hamiltonian H may be different at the different points the quantum circuit.

Hence, our definition of "input" is quite general, and allows, for example, that the input is given in the middle of a quantum circuit—mimicking the way how algorithms for fault-tolerant quantum computing operate on continuous data: the subroutine for accessing the data be called repeatedly; cf., e.g., the description of the input oracles in van Apeldoorn and Gilyén [14]. It should be pointed out, however, that general state preparation procedures as in Harrow et al. [15] and Schuld et al. [12] cannot not be studied with the tools of this paper, because they apply many operations with parameters derived from a *collection* of inputs, instead of a *single* input.

Our lower bound technique is based on Fourier analysis.

1.2. Example

Take, as example, the parameterized quantum circuits of Mitarai et al. [13] mentioned above. Comparing with (1) shows: The single real input x close to 0 is prepared by performing, at n different positions in the quantum circuit, Hamiltonian evolution $e^{-i\eta_j(x)H_j}$ with $\eta_j(x) := \arcsin(x)$, and $H_j := \sigma_Y/2$, for $j = 1,\ldots,n$.

We say that the input x to the quantum circuits of Mitarai et al. are encoded with *input redundancy n*—meaning, the input is given n times.

The example highlights the ostensible wastefulness of giving the same data n times, and the question naturally arises whether a more clever application of possibly different rotations would have reduced the amount of input redundancy.

In the case of Mitarai et al.'s example, it can easily be seen—from algebraic arguments involving the quantum operations which are performed—that, in order to produce a polynomial of degree n, redundancy n is best possible for the particular way of encoding the value x *by applying the Pauli rotation to distinct qubits*, we leave that to the reader. However, already the question whether by re-using the same qubit a less "wasteful" encoding could have been achieved is quite not so easy. Our Fourier analysis based techniques give lower bounds for more general encodings, in particular, for applying arbitrary single-qubit Pauli rotations to an arbitrary set of qubits at arbitrary time during the quantum circuit.

Figure 1 next page shows the schematic of quantum circuits with input x. The setup resembles that of a neural network layer. The j'th "copy" of the input is made available in the quantum circuit by, at some time, performing the unitary operation $e^{2\pi i\eta_j(x)H_j}$ on one qubit, where $\eta_j(x) = \varphi(a_jx + b_j)$, for an "activation function" φ. (We switch here to adding the factor 2π, to be compatible with our Fourier approach).

In the above-mentioned example in Mitarai et al. [13], the activation function is $\varphi := \arcsin$. **Figure 1** aims at making clear that the input can be encoded by applying different unitary operations to different qubits, or to the same qubit several times, or any combination of these possibilities. Generalization

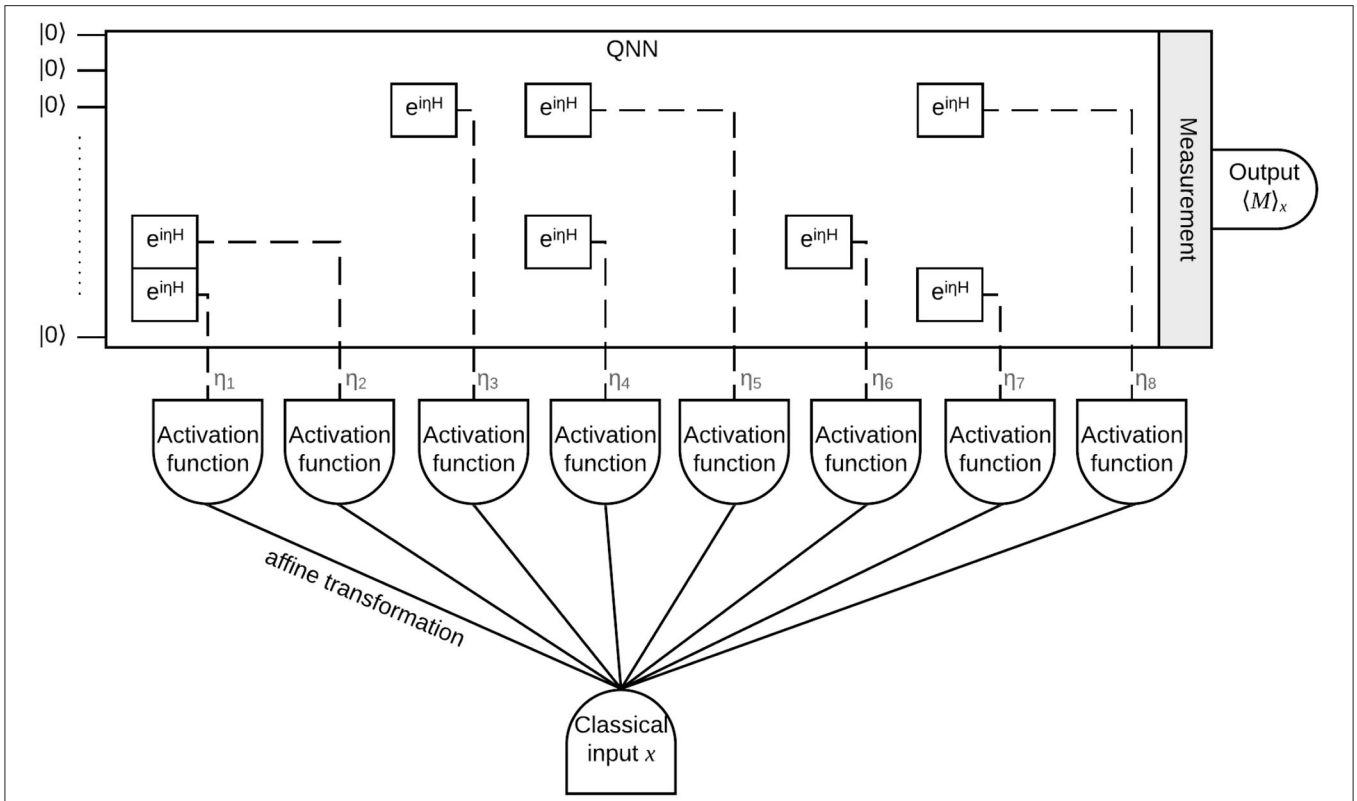


FIGURE 1 | Schematic for the PQCs we consider. The classical input x, after being subjected to a transformations $\eta_j(x) = \varphi(a_j x + b_j)$ with "activation function" φ, is fed into the QNN/PQC through Hamiltonian evolution operations $e^{2\pi i \eta_j(x) H_j}$. Quantum operations which do not participate in entering the input data into the quantum circuit are not shown; these include the operations which depend on the *training parameters* θ.

of our results to several inputs is straightforward, if the activation functions in **Figure 1** have a single input.

1.3. Our Results

As hinted above, our intention with this paper is to establish, in two natural examples, the possibility of proving lower bounds on input redundancy. The first example is what we call "linear" input encoding, where the activation function is $\varphi(x) = x$. The second example is Mitarai et al. [13] approach, where the activation function is $\varphi(x) = \arcsin(x)$.

For both examples, we prove lower bounds on the input redundancy in terms of linear-algebraic complexity measures of the target function. We find the lower bounds to be logarithmic, and the bounds are tight.

To the best of our knowledge, our results give the first quantitative lower bounds on input redundancy. These lower bounds, as well as other conclusions derived from our constructions, should directly influence design decisions for quantum neural network architectures.

1.4. Paper Organization

In the next section we review the background on the PQC model underlying our results. Sections 3 and 4 contain the results on linear and arcsine input encoding, respectively. We close with a discussion and directions of future work.

2. BACKGROUND

2.1. MiNKiF PQCs

We now describe parameterized quantum circuits (PQCs) in more detail. Denote by

$$U_H(\alpha)\colon \rho \mapsto e^{-2\pi i \alpha H} \rho e^{2\pi i \alpha H} \tag{2}$$

the quantum operations of an evolution with Hamiltonian H (operating on some set of qubits); the 2π factor is just a convenience for us and introduces no loss of generality. Following, in spirit, Mitarai et al. [13], in this paper we consider quantum circuits which apply quantum operations each of which is one of the following:

1. An operation as in (2), with a parameter $\alpha := \eta$ which will encode input, x (i.e., η is determined by x);
2. An operation as in (2), with a parameter $\alpha := \theta$ which will be "trained" (we refer to these parameters as the *training parameters*);
3. Any quantum operation not defined by any parameter (although its effect can *depend* on θ, η, e.g., via dependency on measurement results).

Denote the concatenated quantum operation by $\mathcal{E}(\eta, \theta)$. Now let M be an observable, and consider its expectation value on the state which results if the parameterized quantum circuit is

applied to a fixed input state ρ_0, e.g., $\rho_0 := |0\rangle\langle 0|$. We denote the expectation value with parameters set to η, θ by $f(\eta, \theta)$:

$$f\colon \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}\colon (\eta, \theta) \mapsto = \operatorname{tr}(M\mathcal{E}(\eta, \theta)\rho_0). \tag{3}$$

The PQCs could have multiple outputs, but we do not consider that in this paper. We refer to PQCs of this type as *MiNKiF* PQCs, as [13] realized the fundamental property

$$\partial_{\theta_j} f(\eta, \theta) = \pi\left(f(\eta, \theta + \tfrac{1}{4}e_j) - f(\eta, \theta - \tfrac{1}{4}e_j)\right), \tag{4}$$

where e_j is the vector with a 1 in position j and 0 otherwise. This equation characterizes trigonometric functions. (The same relation holds obviously for derivatives in η_j direction.)

The setting we consider in this paper is the following.

- The training parameters, θ, have been trained perfectly and are thus ignored, in other words, omitting the θ argument, we conveniently consider f to be a function defined on $\mathbb{R}^n$ (instead of on $\mathbb{R}^n \times \mathbb{R}^m$);
- The inputs, x, are real numbers;
- The parameters η of f are determined by x, i.e., η is replaced by $(\varphi(a_1 x + b_1), \ldots, \varphi(a_n x + b_n))$, where $a, b \in \mathbb{R}^n$; in other words, we study the function

$$\mathbb{R} \to \mathbb{R}\colon x \mapsto f(\varphi(a_1 x + b_1), \ldots, \varphi(a_n x + b_n)).$$

We allow a, b to depend on the target function[1].

This setting is restrictive only in as far as the input is one-dimensional; the reason for this restriction is that this paper aims to introduce and demonstrate a concept, and not be encyclopedic or obtain the best possible results.

This setting clearly includes the versions of amplitude encoding discussed in the introduction by applying operations $U_{H_j}(\varphi(a_j x + b_j))$ to $|0\rangle\langle 0|$ states (of appropriately many qubits) for several j's, with suitable H_j's. However, the setting is more general in that it doesn't restrict to encode the input near the beginning of a quantum circuit, indeed, the order of the types of quantum operations is completely free.

To summarize, we study the functions

$$\begin{aligned} &f\colon \mathbb{R} \to \mathbb{R}\colon \eta \mapsto f(\eta) && \text{(5a)}\\ &:= \operatorname{tr}\Big(M\, V_n\, U_n(\eta_n)\, V_{n-1}\, U_{n-1}(\eta_{n-1})\ \ldots\ V_1\, U_1(\eta_1)\, V_0\, \rho_0\Big) \end{aligned}$$

where

$$U_1 := U_{H_1}, \ldots, U_n := U_{H_n} \text{for Hamiltonians } H_j,\ j = 1, \ldots, n. \tag{5b}$$

and

$$\mathbb{R} \to \mathbb{R}\colon x \mapsto f(\eta(x)) \tag{6}$$

where $\eta\colon \mathbb{R} \to \mathbb{R}^n\colon x \mapsto \varphi(a_1 x + b_1), \ldots, \varphi(a_n x + b_n)$. Then we ask the question: How large is the space of the $x\colon f(\eta(x))$, for a fixed activation function φ, but variable vectors $a, b \in \mathbb{R}^n$?

[1] Cf. Remarks 6 and 12. Indeed, our analysis suggests that the a, b should be training parameters if the goal is to achieve high expressivity; see the Conclusions.

2.2. Fourier Calculus on MiNKiF Circuits

This paper builds on the simple observation of [16] that, under assumptions which are reasonable for near-term gate-based quantum computers, the Fourier spectrum, in the sense of the Fourier transform of tempered distributions, is finite and can be understood from the eigenvalues of the Hamiltonians. In particular, if, for each of the Hamiltonians H_j, $j = 1, \ldots, n$, the differences of the eigenvalue of H_j are integer multiples of a positive number κ_j, then $\eta \mapsto f(\eta)$ is periodic.

Take, for example, the case of Pauli rotations ($e^{-2\pi i \sigma_\star/2}$ in our notation): There, each of the H_j is of the form $\sigma_{u_j}/2$ (with $u_j \in \{x, y, z\}$). The eigenvalues of H_j are $\pm 1/2$, the eigenvalue differences are $0, \pm 1$, and $f\colon \mathbb{R}^n \to \mathbb{R}$ is 1-periodic[2] in every parameter, with Fourier spectrum contained in

$$\mathbb{Z}_3^n := \{0, \pm 1\}^n. \tag{7}$$

More generally, if the H_j have eigenvalues, say, $\lambda_j^{(0)} \in \mathbb{R}$ and $\lambda_j^{(s)} = \lambda_j^{(0)} + s$ for $s = 1, \ldots, K_j$, then the eigenvalue differences are $\{-K_j, \ldots, K_j\}$, and $f\colon \mathbb{R}^n \to \mathbb{R}$ is 1-periodic in every parameter, with Fourier spectrum contained in $\prod_{j=1}^n \{-K_j, \ldots, K_j\}$.

We refer to [16] for the (easy) details. In this paper, focusing on the goal of demonstrating the possibility to prove lower bounds on the input redundancy, we mostly restrict to 2-level Hamiltonians with eigenvalue difference 1 (such as one-half times a tensor product of Pauli matrices), which gives us the nice Fourier spectrum (7), commenting on other spectra only *en passant*.

For easy reference, we summarize the discrete Fourier analysis properties of the expectation value functions that we consider in the following remark. The proof of the equivalence of the three conditions is contained in the above discussions, except for the existence of a quantum circuit for a given multi-linear trigonometric polynomial, for which we defer [17], as it is not the topic of this paper.

REMARK 1. *The following three statements are equivalent. If they hold, we refer to the function as an expectation value function, for brevity (suppressing the condition on the eigenvalues of the Hamiltonians). The input redundancy of the function is n.*

1. *The function f is of the form 5, where the H_j, $j = 1, \ldots, n$, have eigenvalues $\pm 1/2$.*
2. *The function f is a real-valued function $\mathbb{R}^n \to \mathbb{R}$ which is 1-periodic in every parameter, and its Fourier spectrum is contained in $\mathbb{Z}_3^n$. Hence,*

$$f(\eta) = \sum_{w \in \mathbb{Z}_3^n} \hat{f}(w) e^{2\pi i w \bullet \eta} \tag{8}$$

where $w \bullet \eta := \sum_{j=1}^n \eta_j w_j$ is the dot product (computed in $\mathbb{R}$), and $\hat{f}$ the usual periodic Fourier transform of f, i.e., $\hat{f}(w) = \int_{[0,1]^n} e^{-2\pi i w \bullet \eta} f(\eta)\, d\eta$.

[2] This is where the factor 2π in the exponent is used.

3. *The function f is a multi-linear polynomial in the sine and cosine functions, i.e.,*

$$f(\eta) = \sum_{\tau\in\{1,\cos,\sin\}^n} \tilde{f}_\tau \prod_{j=1}^{n} \tau_j(2\pi\eta_j), \tag{9}$$

(where "1" under the sum denotes the all-1 function).

3. LINEAR INPUT ENCODING

We start discussing the case where the input parameters are affine functions of the input variable, e.g., $\varphi = \mathrm{id}$ and $\eta(x) = x\cdot a + b$ for some $a, b \in \mathbb{R}^n$, so the input redundancy is n.

For $a \in \mathbb{R}^n$ define

$$K_a := \left\{ w \bullet a \;\middle|\; w \in \mathbb{Z}_3^n \right\}, \tag{10a}$$

$$\mathrm{spread}(a) := \frac{1}{2}|K_a \setminus \{0\}|, \tag{10b}$$

with $|\cdot|$ denoting set cardinality; we refer to spread(a) as the *spread* of a. We point the reader to the fact that K_a is symmetric around $0 \in \mathbb{R}$ and $0 \in K_a$, so that the spread is a non-negative integer.

For every $k \in \mathbb{R}$, consider the function

$$\chi_k \colon \mathbb{R} \to \mathbb{C} \colon t \mapsto e^{2\pi i k t}. \tag{11}$$

These functions are elements of the vector space $\mathbb{C}^{\mathbb{R}}$ of all complex-valued functions on the real line. We note the following well-known fact.

LEMMA 2. *The functions χ_k, $k \in \mathbb{R}$, defined in (11) are linearly independent (in the algebraic sense, i.e., every finite subset is linearly independent).*

Moreover, for every $x_0 \in \mathbb{R}$ and $\varepsilon > 0$, the restrictions of these functions to the interval $]x_0-\varepsilon, x_0+\varepsilon[$ are linearly independent.

Proof: We refer the reader to **Appendix 1** for the first statement and only prove the second one.

Suppose that for some finite set $K \subset \mathbb{R}$ and complex numbers α_k, $k \in K$ we have $g(z) := \sum_{j=1}^{m} \alpha_j \chi_{k_j}(z) = 0$ for all $z \in]x_0-\varepsilon, x_0+\varepsilon[$. Since g is analytic and non-zero analytic functions can only vanish on a discrete set, we then must also have $g(z) = 0$ for all $z \in \mathbb{C}$. This means that the linear dependence on an interval implies linear dependence on the whole real line. This proves the second statement, and the proof of Lemma 2 is completed.

We can now give the definition of the quantity which will lower-bound the input redundancy for linear input encoding.

DEFINITION 3. *The Fourier rank of a function $h \colon \mathbb{R} \to \mathbb{R}$ at a point $x_0 \in \mathbb{R}$ is the infimum of the numbers r such that there exists an $\varepsilon > 0$, a set $K \subset \mathbb{R}\setminus\{0\}$ of size $2r$, and coefficients $\alpha_k \in \mathbb{C}$, $k \in \{0\}\cup K$ such that*

$$h(x) = \sum_{k\in\{0\}\cup K} \alpha_k\, \chi_k(x) \quad \text{for all } x \in]x_0-\varepsilon, x_0+\varepsilon[. \tag{12}$$

Note that the Fourier rank can be infinite, and if it is finite, then it is a non-negative integer. Indeed, from $h^* = h$ it follows that $\sum_{k\in\{0\}\cup K} \alpha_k\chi_k = \sum_{k\in\{0\}\cup K} \alpha_k^*\chi_{-k}$, so that by the linear independence of the χ's (Lemma 2) we have $\alpha_{-k} = \alpha_k^*$, which means that in a minimal representation of h, the set K is symmetric around $0 \in \mathbb{R}$.

EXAMPLES.

- Constant functions have Fourier rank 0 at every point.
- The trigonometric functions $x \mapsto \cos(\kappa x + \phi)$, with $\kappa \neq 0$, have Fourier rank 1 at every point.
- Trigonometric polynomials of degree d, $x \mapsto \sum_{j=0}^{d} \alpha_j \cos^j(\kappa_j x + \phi_j)$, have Fourier rank d at every point, if $\alpha_d \neq 0, \kappa_d \neq 0$.
- The function $x \mapsto \left|\sin(\pi x)\right|$ has Fourier rank 1 at every $x_0 \in \mathbb{R}\setminus\mathbb{Z}$ and infinite Fourier rank at the points $x_0 \in \mathbb{Z}$.
- The function $x \mapsto x$ has infinite Fourier rank at every point.

THEOREM 4. *Let f be an expectation value function, i.e., as in Remark 1. Moreover, let $a, b \in \mathbb{R}^n$, and $h \colon \mathbb{R} \to \mathbb{R} \colon x \mapsto f(x\cdot a + b)$. For every $x_0 \in \mathbb{R}$, the Fourier rank of h at x_0 is less than or equal to the spread of a.*

Proof: With the preparations above, this is now a piece of cake. Let $x_0 \in \mathbb{R}$ and set $\varepsilon := 1$. With K_a as defined in (10), for $x \in]x_0-\varepsilon, x_0+\varepsilon[$, we have

$$\begin{aligned} h(x) = f(x\cdot a + b) &= \sum_{w\in\mathbb{Z}_3^n} \hat{f}(w) e^{2\pi i w\bullet(x\cdot a+b)} && \text{[Remark 12]} \\ &= \sum_{w\in\mathbb{Z}_3^n} \hat{f}(w) e^{2\pi i w\bullet b} e^{2\pi i x\cdot w\bullet a} \\ &= \sum_{k\in K_a} \Big(\sum_{\substack{w\in\mathbb{Z}_3^n,\\ w\bullet a = k}} \hat{f}(w) e^{2\pi i w\bullet b} \Big)\, e^{2\pi i x\cdot k} \\ &= \sum_{k\in K_a} \alpha_k\, \chi_k(x), \end{aligned}$$

where we let

$$\alpha_k := \sum_{\substack{w\in\mathbb{Z}_3^n,\\ w\bullet a = k}} \hat{f}(w) e^{2\pi i w\bullet b}$$

This shows that h has a representation as in (12) with $K := K_a \setminus \{0\}$. It follows that the Fourier rank of h is bounded from above by $|K_a|/2 = \mathrm{spread}\, a$. This completes the proof of Theorem 4.

The theorem allows us to give the concrete lower bounds for the input redundancy.

COROLLARY 5. *Let h be a real-valued function defined in some neighborhood of a point $x_0 \in \mathbb{R}$.*

Suppose that in a neighborhood of x_0, h is equal to an expectation value function with linear input encoding, i.e., there is an n, a function f as in Remark 1, vectors $a, b \in \mathbb{R}^n$, and an $\varepsilon > 0$ such that $h(x) = f(x\cdot a + b)$ holds for all $x \in]x_0-\varepsilon, x_0+\varepsilon[$.

The input redundancy, n, is greater than or equal to $\log_3(r+1)$, where r is the Fourier rank of h at x_0.

To represent a function h by a MiNKiF PQC with linear input encoding in a tiny neighborhood of a given point x_0, the input redundancy must be at least the logarithm of the Fourier rank of h at x_0.

Proof of Corollary 5: For every $a \in \mathbb{R}^n$, we have $|K_a| \leq 3^n$, by the definition of K_a, and hence $\text{spread}(a) \leq (3^n - 1)/2$.

We allow that a, b are chosen depending on h (see the Remark 6 below). Theorem 4 gives us the inequality

$$r \leq \max_a \text{spread}(a) \leq (3^n - 1)/2,$$

which implies $n \geq \log_3(2r + 1) \geq \log_3(r + 1)$, as claimed. (We put the $+1$ to make the expression well-defined for $r = 0$.) This concludes the proof of Corollary 5.

REMARK 6. *If the entries of a are all equal up to sign, then we have* $\text{spread}(a) = n$. *It can be seen that if the entries of a are chosen uniformly at random in* $[0, 1]$, *then* $\text{spread}(a) = (3^n - 1)/2$. *Hence, it seems that some choices for a are better than others. Moreover, looking into the proof of Theorem 4 again, we see that the χ_k, $k \in K_a$, must suffice to represent (or approximate) the target function, and that the entries of b play a role in which coefficients α_k can be chosen for a given a. Hence, it is plausible that the choices of a, b should depend on h.*

REMARK 7. *Our restriction to Hamiltonians with two eigenvalues leads to the definition of the spread in (10). If the set of eigenvalue distances of the Hamiltonian encoding the input η_j is $D_j \subset \mathbb{R}$, then, for the definition of the spread, we must put this:*

$$K_a := \left\{ w \bullet a \;\middle|\; w \in \prod_{j=1}^{n} D_j \right\}.$$

Theorem 4 and Corollary 5 remain valid, with essentially the same proofs, but with a higher base for the logarithm.

4. ARCSINE INPUT ENCODING

We now consider the original situation of the example in Mitarai et al. [13], where the activation function is $\varphi = \arcsin$. More precisely, for $a, b \in \mathbb{R}^n$, we consider

$$\eta(x) := \arcsin((ax + b)/(2\pi)).$$

Abbreviating $s_j := a_j x + b_j$ and $c_j := \sqrt{1 - s_j^2}$ for $j = 1, \ldots, n$, Remark 13, gives us that the expectation value functions with arcsine input encoding are of the form

$$\begin{aligned} h(x) = f(\eta(x)) &= \sum_{\substack{S,C \subseteq [n] \\ S \cap C = \emptyset}} \tilde{f}_{S,C} \prod_{j \in S} s_j \prod_{j \in C} c_j \\ &= \sum_{\substack{S,C \subseteq [n] \\ S \cap C = \emptyset}} \tilde{f}_{S,C} \prod_{j \in S} (a_j x + b_j) \prod_{j \in C} \sqrt{1 - (a_j x + b_j)^2}, \quad (13) \end{aligned}$$

where we use the common shorthand $[n] := \{1, \ldots, n\}$, and set $\tilde{f}_{S,C} := \tilde{f}_{\tau(S,C)}$ with $\tau_j(S, C) = \sin$ if $j \in S$, $\tau_j(S, C) = \cos$ if $j \in C$, and $\tau_j(S, C) = \text{id}$ otherwise.

Consider a formal expression of the form

$$\mu_{S,C}^{(a,b)} := \prod_{j \in S} (a_j x + b_j) \prod_{j \in C} \sqrt{1 - (a_j x + b_j)^2} \quad (14)$$

where x is a variable (for arbitrary $a, b \in \mathbb{R}^n$ and $S, C \subseteq [n]$ with $S \cap C = \emptyset$). We call it an *sc-monomial* of degree $|S| + |C|$. An sc-monomial can be evaluated at points $x \in \mathbb{R}$ for which the expression under the square root is not a negative real number, i.e., in the interval

$$I_\mu := \bigcap_{j \in C} \left] \frac{-1 - b_j}{a_j}, \frac{+1 - b_j}{a_j} \right[\quad (15)$$

(which could be empty), and it defines an analytic function there. Note, though, that it can happen that an sc-monomial can be continued to an analytic function on a larger interval than I_μ. The obvious example where that happens is this: For $j, j' \in C$ with $j \neq j'$ we have $(a_j, b_j) = \pm(a_{j'}, b_{j'})$. In that case, the formal power series of the sc-monomial simplifies, and omitting the interval $\left] \frac{-1 - b_j}{a_j}, \frac{+1 - b_j}{a_j} \right[$ (also for j') from (15) makes the intersection larger.

The following technical fact can be shown (cf. [17]).

LEMMA 8. *Let $g = \sum_j \alpha_j \mu_j$ be a linear combination of sc-monomials with degrees at most d, and suppose that $\bigcap_j I_{\mu_j} \neq \emptyset$. If an analytic continuation of g to a function $\tilde{g} \colon \mathbb{R} \to \mathbb{R}$ exists, then $\tilde{g}$ is a polynomial of degree at most d.*

From this lemma, we obtain the following result.

COROLLARY 9. *Let $h \colon \mathbb{R} \to \mathbb{C}$ be an analytic function, and $x_0 \in \mathbb{R}$.*

Suppose that in a neighborhood of x_0, h is equal to an expectation value function with arcsine input encoding, i.e., there is an n, a function f as in Remark 1, vectors $a, b \in \mathbb{R}^n$, and an $\varepsilon > 0$ such that

1. $-1 \leq x \cdot a_j + b_j \leq +1$ *for all* $x \in]x_0 - \varepsilon, x_0 + \varepsilon[$, *and*
2. $h(x) = f(\arcsin(x \cdot a + b))$ *holds for all* $x \in]x_0 - \varepsilon, x_0 + \varepsilon[$.

Then h is a polynomial, and the input redundancy, n, is greater than or equal to the degree of h.

To represent a polynomial h by a MiNKiF PQC with arcsine input encoding in a tiny neighborhood of a given point x_0, the input redundancy must be at least the degree of h.

Proof of Corollary 9: Let us abbreviate $g \colon x \mapsto f(\arcsin(x \cdot a + b)) \colon]x_0 - \varepsilon, x_0 + \varepsilon[\to \mathbb{R}$. From the discussion above, we know that g is a linear combination of sc-monomials.

Both functions h and g are analytic, and they coincide on an interval. Hence, g has an analytic continuation, h, to the

real line so that Lemma 8 is applicable, and states that h is a polynomial with degree at most n. This completes the proof of Corollary 9.

As indicated in the introduction, in the special case which is considered in Mitarai et al. [13]—where the input amplitudes are stored (by rotations) in n distinct qubits before any other quantum operation is performed—this can be proved by looking directly at the effect of a Pauli transfer matrix on the mixed state vector in the Pauli basis. Our corollary shows that this effect persists no matter how the arcsine-encoded inputs are spread over the quantum circuit.

The corollary allows us to lower bound the input redundancy for some functions.

EXAMPLES. There is no PQC with arcsine input encoding that represents the function $x \mapsto \sin x$ (exactly) in a neighborhood any point. Indeed, the same holds for any analytic function defined on the real line which is not a polynomial: the exponential function, the sigmoid function, arcus tangens, ...

Unfortunately, from these impossibility results, no approximation error lower bounds can be derived. Indeed, in their paper [13], Mitarai et al. point out that, due to the $\sqrt{\cdot}$ terms, the functions represented by the expectation values can more easily represent a larger class of functions than polynomials.

To give lower bounds for the representation of functions which are not analytic on the whole real line, we proceed as follows. For fixed $n \geq 1$, $x_0 \in \mathbb{R}$ and $a, b \in \mathbb{R}^n$, denote by $M_{x_0}^{n;a,b}$ the vector space spanned by all functions of the form $]x_0 - \varepsilon, x_0 + \varepsilon[\to \mathbb{R} \colon x \mapsto f(\arcsin(x \cdot a + b))$ for an[3] $\varepsilon > 0$, where f ranges over all expectation value functions with input redundancy n, i.e., functions as in Remark 1, $a, b \in \mathbb{R}^n$ satisfy $-1 < a_j x_0 + b_j < +1$, and the arcsin is applied to each component of the vector.

PROPOSITION 10. *The vector space $M_{x_0}^{n;a,b}$ has dimension at most 3^n, and is spanned by the sc-monomials (14) of degree n.*

Proof: With a, b fixed, there are at most 3^n sc-monomials (14) of degree n, as $S, C \subseteq [n]$ and $S \cap C = \emptyset$ hold. Hence, the statement about the dimension follows from the fact that the elements of $M_{x_0}^{n;a,b}$ are generated by sc-monomials.

The fact that the sc-monomials generate the expectation value functions with arcsine input-encoding of redundancy n is just the statement of (13) above. This concludes the proof of Proposition 10.

We can now proceed in analogy to the case of linear input encoding. Let us define the *sc-rank* at x_0 of a function h defined in a neighborhood of x_0 as the infimum over all r for which there exist sc-monomials $\mu_1, \ldots, \mu_r$, an $\varepsilon > 0$, and real numbers $\alpha_1, \ldots, \alpha_r$ such that $x_0 \in \bigcap_j I_{\mu_u}$, and

$$h(x) = \sum_{j=1}^{r} \alpha_j \mu_j(x) \quad \text{for all } x \in]x_0 - \varepsilon, x_0 + \varepsilon[.$$

[3]Mathematically rigorously speaking, $M_{x_0}^{n;a,b}$ is the germ of functions at x_0.

Proposition 10 now directly implies the following result.

COROLLARY 11. *Let h be a real-valued function defined in some neighborhood of a point $x_0 \in \mathbb{R}$.*

Suppose that in a neighborhood of x_0, h is equal to an expectation value function with arcsine input encoding, i.e., there is an n, a function f as in Remark 1, vectors $a, b \in \mathbb{R}^n$, and an $\varepsilon > 0$ such that $h(x) = f(\arcsin(x \cdot a + b)/(2\pi))$ holds for all $x \in]x_0 - \varepsilon, x_0 + \varepsilon[$.

The input redundancy, n, is greater than or equal to $\log_3(r)$, where r is the sc-rank of h at x_0.

> To represent a function h by a MiNKiF PQC with arcsine input encoding in a tiny neighborhood of a given point x_0, the input redundancy must be at least the logarithm of the sc-rank of h at x_0.

We conclude the section with a note on the choice of the parameters a, b.

REMARK 12. *It can be seen [17] that the dimension of the space $M_{x_0}^n$ is 3^n, if a_j, b_j $j = 1, \ldots, n$ are chosen in general position, but only $O(n)$ if a is a constant multiple of the all-ones vector. Moreover, as indicated in Proposition 10, the basis elements which span the space depend on a, b, and hence the space $M_{x_0}^{n;a,b}$ will in general be different for different choices of a, b. Again, we find that it is plausible that the choices of a, b should depend on the target function.*

5. CONCLUSIONS AND OUTLOOK

To the best of our knowledge, our results give the first rigorous theoretical quantitative justification of a routine decision for the design of parameterized quantum circuit architectures: Input redundancy *must* be present if good approximations of functions are the goal.

Both activation functions we have considered give clear evidence that input redundancy is necessary, and grows at least logarithmically with the "complexity" of the function: The complexity of a function f with respect to a family $\mathcal{B}$ of "basis functions" is the number of functions from the family which are needed to obtain f as a linear combination. In our results, the function family $\mathcal{B}$ depends on the activation function. In the case of linear input encoding (activation function "identity"), the basis functions are trigonometric functions $t \mapsto e^{2\pi i k t}$, whereas for the arcsin activation function, we obtain the basis monomials (14) already used, in a weaker form, in Mitarai et al. [13].

From Remarks 6 and 12 we see that the weights a, b, i.e., the coefficients in the affine transformation links in **Figure 1**, should have to be variable in order to ensure a reasonable amount of expressiveness in the function represented by the quantum circuit. We use the term *variational input encoding* to refer to the concept of training the parameters involved in the encoding with other model parameters. A recent set of limited experiments [18] indicate that variational input

encoding improves the accuracy of Quantum Neural Networks in classification tasks.

While we emphasize the point that this paper demonstrates a concept—lower bounds for input redundancy can be proven—there are a few obvious avenues to improve our results.

Most importantly, our proofs rely on exactly representing a target function. This is an unrealistic scenario. The most pressing task is thus to give lower bounds on the input redundancy when an approximation of the target function with a desired accuracy $\varepsilon > 0$ in a suitable norm is sufficient.

Secondly, we thank an anonymous reviewer for pointing out to us that lower bounds for many more activation functions could be proved.

Finally, Remark 1 mentions that for every multi-linear trigonometric polynomial f, there is a PQC whose expectation value function is precisely f. It would be interesting to lower-bound a suitable quantum-complexity measure of the PQCs representing a function, e.g., circuit depth. While comparisons of the quantum vs. classical complexity of estimating expectation values have attracted some attention [19], to our knowledge, the same question in the "parameterized setting" has not been considered.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

AUTHOR CONTRIBUTIONS

DT contributed the realization that lower bounds could be obtained, and sketches of the proofs. FG contributed the details of the proofs and the literature overview. All authors contributed to the article and approved the submitted version.

FUNDING

This research was supported by the Estonian Research Council, ETAG (*Eesti Teadusagentuur*), through PUT Exploratory Grant #620. DT was partly supported by the Estonian Centre of Excellence in IT (EXCITE, 2014-2020.4.01.15-0018), funded by the European Regional Development Fund.

ACKNOWLEDGMENTS

This manuscript has been released as a pre-print on arXiv (arXiv:1901.11434).

REFERENCES

1. Feynman RP. Simulating physics with computers. *Int J Theor Phys*. (1982) **21**:467–88. doi: 10.1007/BF02650179
2. Manin II. *Vychislimoe i nevychislimoe*. Moscow: Sov. Radio (1980).
3. Mohseni M, Read P, Neven H, Boixo S, Denchev V, Babbush R, et al. Commercialize quantum technologies in five years. *Nat News*. (2017) **543**:171. doi: 10.1038/543171a
4. Perdomo-Ortiz A, Benedetti M, Realpe-Gómez J, Biswas R. Opportunities and challenges for quantum-assisted machine learning in near-term quantum computers. *Quant Sci Technol*. (2018) **3**:030502. doi: 10.1088/2058-9565/aab859
5. Neville A, Sparrow C, Clifford R, Johnston E, Birchall PM, Montanaro A, et al. Classical boson sampling algorithms with superior performance to near-term experiments. *Nat Phys*. (2017) **13**:1153. doi: 10.1038/nphys4270
6. Bremner MJ, Montanaro A, Shepherd DJ. Average-case complexity versus approximate simulation of commuting quantum computations. *Phys Rev Lett*. (2016) **117**:080501. doi: 10.1103/PhysRevLett.117.080501
7. Farhi E, Harrow AW. Quantum supremacy through the quantum approximate optimization algorithm. *arXiv*. (2016) 160 207674.
8. Du Y, Hsieh MH, Liu T, Tao D. The expressive power of parameterized quantum circuits. *arXiv*. (2018) 181011922.
9. Farhi E, Neven H. Classification with quantum neural networks on near term processors. *arXiv*. (2018) 180206002.
10. McClean JR, Romero J, Babbush R, Aspuru-Guzik A. The theory of variational hybrid quantum-classical algorithms. *New J Phys*. (2016) **18**:023023. doi: 10.1088/1367-2630/18/2/023023
11. Schuld M, Petruccione F. *Supervised Learning with Quantum Computers*. Vol. 17. Cham: Springer (2018).
12. Schuld M, Bocharov A, Svore K, Wiebe N. Circuit-centric quantum classifiers. *arXiv*. (2018) 180400633.
13. Mitarai K, Negoro M, Kitagawa M, Fujii K. Quantum circuit learning. *Phys Rev A*. (2018) **98**:032309. doi: 10.1103/PhysRevA.98.032309
14. van Apeldoorn J, Gilyén A. Improvements in quantum SDP-solving with applications. *arXiv*. (2018) 180405058.
15. Harrow AW, Hassidim A, Lloyd S. Quantum algorithm for linear systems of equations. *Phys Rev Lett*. (2009) **103**:150502. doi: 10.1103/PhysRevLett.103.150502
16. Vidal JG, Theis DO. Calculus on parameterized quantum circuits. *arXiv*. (2018) 181206323.
17. Vidal JG. *Analysis on Quantum Circuits*. Tartu: University of Tartu (2020).
18. Lei AW. *Comparisons of Input Encodings for Quantum Neural Networks*. Tartu: University of Tartu (2020).
19. Bravyi S, Gosset D, Movassagh R. Classical algorithms for quantum mean values. *arXiv*. (2019) 190911485.

Conflict of Interest: DT was employed by the company Ketita Labs OÜ.

The remaining author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

A. PROOF OF LEMMA 2

We have to prove that the functions χ_k, $k \in \mathbb{R}$, defined in (11) are linearly independent (in the algebraic sense, i.e., considering finite subsets of the functions at a time). There are several ways of proving this well-known fact; we give the proof that probably makes most sense to a physics readership: The Fourier transform (in the sense of tempered distributions) of the function χ_k is $\delta(k - *)$, the Dirac distribution centered on k. These generalized functions are clearly linearly independent for different values of k.

Advantages of publishing in Frontiers

OPEN ACCESS
Articles are free to read for greatest visibility and readership

FAST PUBLICATION
Around 90 days from submission to decision

HIGH QUALITY PEER-REVIEW
Rigorous, collaborative, and constructive peer-review

TRANSPARENT PEER-REVIEW
Editors and reviewers acknowledged by name on published articles

Frontiers
Avenue du Tribunal-Fédéral 34
1005 Lausanne | Switzerland

Visit us: www.frontiersin.org
Contact us: frontiersin.org/about/contact

REPRODUCIBILITY OF RESEARCH
Support open data and methods to enhance research reproducibility

DIGITAL PUBLISHING
Articles designed for optimal readership across devices

FOLLOW US
@frontiersin

IMPACT METRICS
Advanced article metrics track visibility across digital media

EXTENSIVE PROMOTION
Marketing and promotion of impactful research

LOOP RESEARCH NETWORK
Our network increases your article's readership