frontiers in Neuroscience

ORIGINAL RESEARCH
published: 14 January 2021
doi: 10.3389/fnins.2020.615756

A Heterogeneous Spiking Neural Network for Unsupervised Learning of Spatiotemporal Patterns

Xueyuan She, Saurabh Dash, Daehyun Kim and Saibal Mukhopadhyay*

Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, United States

This paper introduces a heterogeneous spiking neural network (H-SNN) as a novel, feedforward SNN structure capable of learning complex spatiotemporal patterns with spike-timing-dependent plasticity (STDP) based unsupervised training. Within H-SNN, hierarchical spatial and temporal patterns are constructed with convolution connections and memory pathways containing spiking neurons with different dynamics. We demonstrate analytically the formation of long and short term memory in H-SNN and distinct response functions of memory pathways. In simulation, the network is tested on visual input of moving objects to simultaneously predict for object class and motion dynamics. Results show that H-SNN achieves prediction accuracy on similar or higher level than supervised deep neural networks (DNN). Compared to SNN trained with back-propagation, H-SNN effectively utilizes STDP to learn spatiotemporal patterns that have better generalizability to unknown motion and/or object classes encountered during inference. In addition, the improved performance is achieved with 6x fewer parameters than complex DNNs, showing H-SNN as an efficient approach for applications with constrained computation resources.

Keywords: spiking neural network, unsupervised STDP learning, multi-objective prediction, spatiotemporal data processing, neuromorphic computing

OPEN ACCESS

Edited by:
Mostafa Rahimi Azghadi, James Cook University, Australia

Reviewed by:
Chankyu Lee, Purdue University, United States
Timothée Masquelier, Centre National de la Recherche Scientifique (CNRS), France

****Correspondence:***
Xueyuan She
xshe@gatech.edu

Specialty section:
This article was submitted to Neuromorphic Engineering, a section of the journal Frontiers in Neuroscience

Received: *09 October 2020*
Accepted: *11 December 2020*
Published: *14 January 2021*

Citation:
She X, Dash S, Kim D and Mukhopadhyay S (2021) A Heterogeneous Spiking Neural Network for Unsupervised Learning of Spatiotemporal Patterns. Front. Neurosci. 14:615756. doi: 10.3389/fnins.2020.615756

1. INTRODUCTION

Spiking neural network (SNN) (Maass, 1997; Gerstner and Kistler, 2002b; Pfeiffer and Pfeil, 2018) is a dynamical system with bio-inspired neurons (Izhikevich, 2003) and learning behaviors (Dan and Poo, 2006; Caporale and Dan, 2008). In rate-encoded SNN, inputs are represented as spike trains with varying (pixel dependent) frequency and patterns within input spikes can be learned without supervision using spike-timing-dependent plasticity (STDP) (Bell et al., 1997; Magee and Johnston, 1997; Gerstner and Kistler, 2002a). The event-driven nature of SNN operation also promises high energy-efficiency during real-time inference (Roy et al., 2019). Over the years, SNN has shown success in spatial processing such as image classification. Although, many large scale spatial SNNs depend on conversion from DNN (Diehl et al., 2015; Rueckauer et al., 2017; Sengupta et al., 2019) or supervised training (Lee et al., 2016; Nicola and Clopath, 2017; Huh and Sejnowski, 2018), more recently, unsupervised learning has shown promising results (Lee et al., 2018; Srinivasan and Roy, 2019). However, SNNs for processing temporal or spatiotemporal data are still primarily based on recurrent connections (DePasquale et al., 2016; Bellec et al., 2018) and use supervised training (Stromatias et al., 2017; Wu et al., 2018), leading to high network complexity for processing spatiotemporal data and high demand for labeled data. Recently, SNN has also been explored for optical flow applications, such as Spike-FlowNet (Lee et al., 2020), which

is based on self-supervision; with STDP-based learning, it is possible to implement SNN for optical flow (Paredes-Vallés et al., 2020), but learning is limited to only short-term temporal patterns.

In this paper, we present a novel heterogeneous SNN (H-SNN) architecture that exploits inherent dynamics of spiking neurons for unsupervised learning of complex spatiotemporal patterns. Our key innovation is to use feedforward connections of spiking neurons with different dynamics to represent memory of different time-scales and learn temporal patterns. This eliminates the need for recurrent connection present in state-of-the-art SNNs used for temporal learning (DePasquale et al., 2016; Bellec et al., 2018; Wu et al., 2018; Lee et al., 2020). Moreover, we adapt spiking convolution modules (Kheradpisheh et al., 2018) to the network architecture. As a result, H-SNN is able to extract distinguishing temporal and spatial features from spatiotemporal inputs using only STDP based unsupervised training. To the best of our knowledge, H-SNN is the first STDP-learned multi-objective SNN for predicting object class and motion. More specifically, the following key contributions are made:

- We propose a novel spiking network architecture and STDP based unsupervised learning process for predicting object class and motion dynamic from spatiotemporal inputs.
- We design neurons with different dynamics and network layers with crossover connections to hierarchically form short and long term memory used to learn complex temporal patterns.
- We present spiking convolutional network with local/cross-depth inhibition to learn motion invariant spatial features from augmented dataset.
- We analytically show that heterogeneous neuronal dynamics and multi-path connection can represent distinguishable features of temporal patterns without any recurrent connection.

The effectiveness of H-SNN is demonstrated with computer vision tasks involving dynamic data set, considering that many neural network applications involve perception of objects in motion (Marković et al., 2014; Baca et al., 2018). **Figure 1** illustrates examples of such applications, where a robotic arm is interacting with a rolling object, and where an unmanned aerial vehicle (UAV) is analyzing a moving car. The object motion can be a mixture of translation and rotation, both with constant or changing speed. A neural network is required to perform two tasks: identify class of the object and understand dynamic of the motion. In real world implementations where unforeseen conditions might be encountered, the neural network must also be able to correctly classify known objects with unseen motion and recognize known motion of unknown objects. Thus, for networks that rely on camera as input, the constant transformation of pixel-level information makes the preceding problem challenging.

In our experiment, we first test the networks on a human gesture dataset captured by event camera (Amir et al., 2017). Datasets consist of frame sequences with various linear (translation) and angular (rotation) motions created from an aerial image dataset (Xia et al., 2017) and Fashion-MNIST (Xiao et al., 2017) are also tested for multi-objective prediction. We compare H-SNN to deep learning approaches including 3D Convolutional Neural Network (CNN) and CNN combined with recurrent neural network (RNN), as well as SNN trained with back-propagation. Results show that H-SNN has comparable (or better) prediction accuracy than baselines while having similar or less network parameters.

2. MATERIALS AND METHODS

2.1. Deep Learning for Learning Spatiotemporal Patterns

The spatiotemporal input data can be processed using deep neural network (DNN) with 3D convolution kernel (Tran et al., 2015) or recurrent connection (Donahue et al., 2015). However, in those spatiotemporal networks, transformation invariance is not explicitly imposed. To achieve transformation equivalent feature extraction, several DNN architectures have been proposed (Cheng et al., 2016; Weiler et al., 2018; Zhang, 2019). While it is possible to apply those designs in the spatiotemporal network, it has been demonstrated that DNNs are invariant only to images with very similar transformation as that in the training set (Azulay and Weiss, 2018), which indicates that DNNs generalize poorly when learning feature transformations. As a result, an increased amount of network parameters are required for conventional DNN to achieve transformation invariant classification. For example, in Weiler et al. (2018), each pre-defined rotation angle requires an additional set of CNN

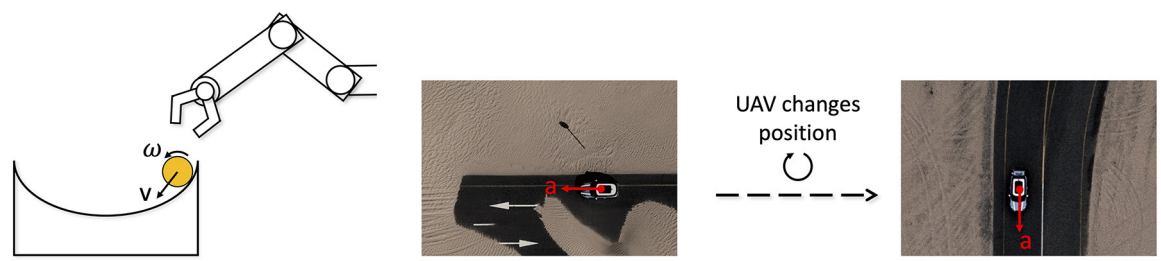

FIGURE 1 | Objects with mixed dynamics in computer vision applications: robots interacting with a rolling ball in a convex **(Left)**; an UAV analyzing movement of a vehicle **(Right)**.

filters; in data augmentation based approaches such as Cheng et al. (2016), more parameters are needed to learn all the rotated features. Together with the combination of 3D kernel or recurrent connections, this leads to large complexity for spatiotemporal networks.

2.2. Spiking Neuron and Synapses

Spiking neural network uses biologically plausible neuron and synapse models that can exploit temporal relationship between spiking events (Moreno-Bote and Drugowitsch, 2015; Lansdell and Kording, 2019). There are different models that are developed to capture the firing pattern of real biological neurons. We choose to use Leaky Integrate-and-Fire (LIF) model in this work described by:

$$\tau_m \frac{dv}{dt} = a + R_m I - v \text{ and } v = v_{reset}, \text{ if } v > v_{threshold} \quad (1)$$

Here, R_m is membrane resistance and $\tau_m = R_m C_m$ is the time constant with C_m being membrane capacitance. a is a parameter used to adjust neuron behavior during simulation. I is the sum of current from all synapses that connects to the neuron. A spike is generated when membrane potential v cross threshold and the neuron enters refractory period, during which the neuron can not spike again.

In SNN, two neurons connected by one synapse are referred to as pre-synaptic neuron and post-synaptic neuron. Conductance of the synapse determines how strongly two neurons are connected and learning can be achieved through modulating the conductance following an algorithm named spike-timing-dependent-plasticity (STDP) (Gerstner et al., 1993; Gerstner and Kistler, 2002a), which has been applied in SNNs designed for computer vision applications (Masquelier and Thorpe, 2007; Diehl and Cook, 2015). There are two types of synaptic weight modulation in STDP: long-term potentiation (LTP) and long-term depression (LTD). More specifically, LTP is triggered when post-synaptic neuron spikes closely after a pre-synaptic neuron spike, indicating a causal relationship between the two events. On the other hand, when a post-synaptic neuron spikes before pre-synaptic spike arrives or without receiving a pre-synaptic spike at all, the synapse goes through LTD. There is an exponential relationship between the time difference of spikes Δt (Bi and Poo, 2001) and the current level of conductance (Querlioz et al., 2013). In this work, the magnitude of LTP and LTD is determined as follows:

$$\Delta G_p = \alpha_p e^{-\Delta t(G-G_{min})/(\tau_{pot}(G_{max}-G_{min}))} \quad (2)$$

$$\Delta G_d = \alpha_d e^{-\Delta t(G_{max}-G)/(\tau_{dep}(G_{max}-G_{min}))} \quad (3)$$

In the functions above, ΔG_p is the magnitude of LTP actions, and ΔG_d is the magnitude of LTD actions. α_p, α_d, G_{max}, and G_{min} are parameters that are tuned based on specific network configurations. τ_{dep} and τ_{pot} are time constant parameters. Δt is determined by subtracting the arrival time of the pre-synapse spike from that of the post-synapse spike ($t_{post} - t_{pre}$).

2.3. The Proposed Method

2.3.1. Heterogeneous Neuron Dynamics

Prior SNN designs mostly focus on using neurons with the same dynamics through the network. In contrast, H-SNN combines neurons with different dynamics to facilitate learning and memory of different length separately in a heterogeneous network. As neurons are used as the basic element of information retention, long and short retention length can be achieved if neurons have different decay rates. For simplicity, let $b = -\frac{1}{\tau_m}$ and $c = \frac{R_m}{\tau_m}$. Consider a spiking neuron that receives input current I at $t = 0$, time for its membrane potential to decay to v_{reset} is:

$$t_{decay} = \frac{1}{b} ln(v_{reset} + \frac{a}{b}) - \frac{1}{b} ln(v_{reset} + \frac{a}{b} + cI) \quad (4)$$

Equation (4) suggests that, by adjusting the parameters a, b and c in Equation (1), different information retention period can be achieved. Particularly, in H-SNN, three types of spiking neurons are used:

- **Learner neuron**, which has a balanced decay rate and input response designed to optimize STDP learning, is similar to neurons used in previous works (Querlioz et al., 2013; Lee et al., 2018; Srinivasan and Roy, 2019). Its parameters are referred to as $\{a_{ln}, b_{ln}, c_{ln}\}$.
- **Short-term neuron**, with parameters $\{a_{stn}, b_{stn}, c_{stn}\}$, has $a_{stn} < a_{ln}$ and $b_{stn} > b_{ln}$ to create higher decay rate. It is used to extract short term patterns from input features.
- **Long-term neuron**, with parameters $\{a_{ltn}, b_{ltn}, c_{ltn}\}$, has lower decay rate than learner neuron, as $a_{ltn} > a_{ln}$ and $b_{ltn} < b_{ln}$. It is designed for long term pattern recognition.

Since membrane potential of long term memory neuron decays slower, under the same input signal, it can potentially produce spike frequency that dominates the short term memory neuron when the two are placed in parallel. To prevent this, $c_{stn} > c_{ltn}$ is used. **Figure 2A** shows responses of different neurons to pre-synaptic frequency. Long-term neuron is able to respond to lower frequency input, due to its lower decay rate, but its post-synaptic frequency increases slowly compared to the short-term neuron. **Figure 2B** shows that short-term and long-term neurons gain membrane potential faster than learner neuron with a given input current, but they also exhibit different decay rates when input current is zero.

2.3.2. Network Architecture

The architecture of H-SNN is shown in **Figure 3**. Three types of modules are connected by three types of data flow paths between network layers. Spike signal from each layer's memory module is sent through perception path to deeper layers. Learning path connects memory module to learner module in the next layer to enable STDP learning. The learned synapse conductance is transferred from learner module to memory module in the same layer. Modules are built with neurons of specific dynamics as

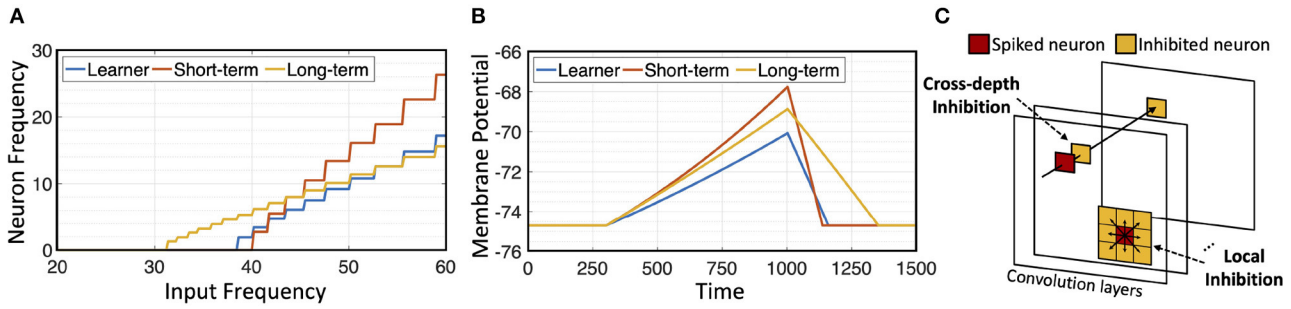

FIGURE 2 | Neurons and inhibition: **(A)** Neuron response to input frequency; **(B)** neuron decay rate; **(C)** illustration of cross-depth and local inhibition.

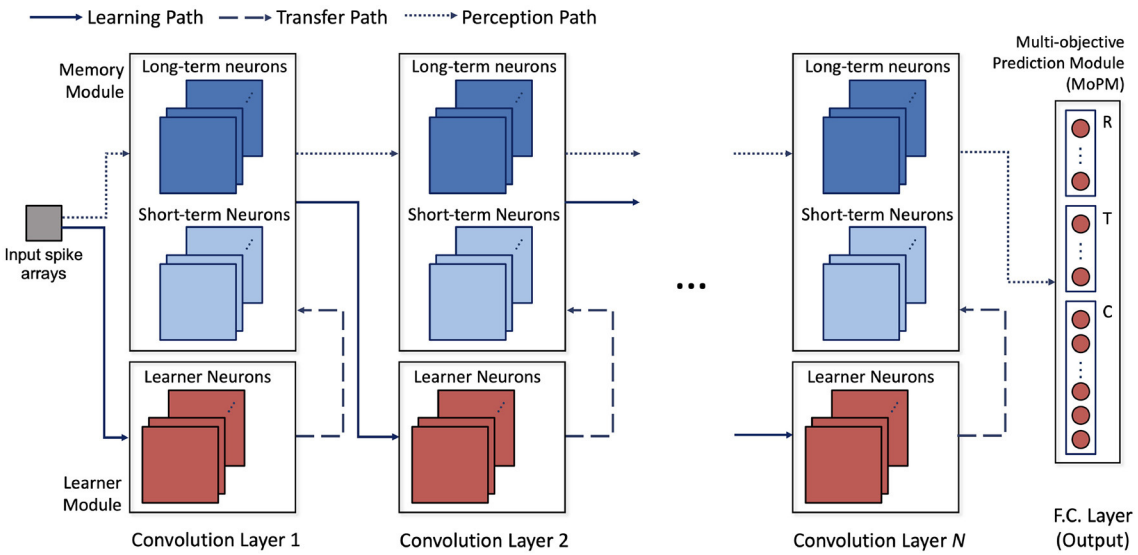

FIGURE 3 | Architecture of the proposed network; network operation involves data flow of three paths: learning path where STDP learning is used to modulate synapse conductance, transfer path where learned features are transferred to long and short term neurons, and perception path, where perceived input spatiotemporal features are encoded in spikes.

mentioned before. Each convolution layer contains a learner module and a memory module. Two inhibition schemes are implemented within the convolutional layers: cross-depth, and local (**Figure 2C**). Cross-depth inhibition is implemented to create competition between neurons with the same receptive field. This prevents more than one kernel from learning the same pattern. Local inhibition, where the spike of one neuron inhibits surrounding neurons in the same depth, is used to help the network to better detect and learn translation invariant features.

Learner module is responsible for facilitating STDP learning. All the spiking neurons in this module are learner neurons, and local inhibition is combined with cross-depth inhibition. For memory module, a combination of long-term and short-term neurons are used. Synapses in memory module are used only for perception thus not modified by STDP learning. Cross-depth inhibition is not implemented to accelerate neuron response and mitigate the diminishing spike frequency issue. The last layer is a multi-objective prediction module (MoPM) with each neuron fully connected to the previous layer. As will be discussed later, MoPM is fine-tuned with supervision. To facilitate the common conversion process, MoPM consists of all standard learner neurons. Inside the MoPM, neurons are indexed as $N_{i,j}$ where i represents one objective (section) and each j represent one label (class) within that section. Here, section-lateral inhibition is implemented, which means that spiking of neuron $N_{i,j}$ sends inhibition signal to all neurons in section i while other sections are unaffected. This enables H-SNN to simultaneously generate prediction for independent objectives.

2.3.3. Learning Process

For each input sequence, all frames are converted to a 2D array of spike train; the frequency of spike train for a pixel is proportional to the pixel's intensity (rate encoding). The network receives one frame at a time and observes it for a period (t_{train} chosen based on input frequency range) to generate sufficient spiking events. After all frames are learned the process repeats for the next sequence.

The learning of the network proceeds layer-wise. During learning of layer 1, neurons in the learner module receive input spikes and perform STDP learning. The learned conductance matrix is transferred to long-term and short-term neurons. Next, learning of layer 2 begins. In this step, memory neurons in layer 1 receive input spikes and generate spikes for the learner neurons in layer 2. After the learner neurons complete learning of all training sequences, the conductance matrix is transferred to layer 2 memory neurons. This process repeats for all convolution layers. While in one layer, the learner neurons exhibit different neuron activity with the long-term and short-term neurons, such layer-wise learning process allows the next layer to learn the changed spiking pattern using STDP. Since multi-layer SNN experiences diminishing spiking frequency along network layers, threshold of neurons in the memory module are scaled down to produce higher output spiking frequency. The scaling factor for each layer is tuned as network hyperparameter, and its value is kept uniform within one layer to prevent distortion of output pattern. During training of the final fully connected layer (MoPM), the perception path produces spikes for each training sequence and spike frequency of all memory neurons in convolution layer n is calculated. The last layer is trained with a conventional stochastic gradient descent method to predict multiple objectives. The objective function is to minimize binary cross entropy loss between label vectors that are one-hot encoded for each prediction target and the layer output.

2.3.4. Memory Pathway - Hierarchical Memory Formation in H-SNN

Within the perception path, long-term and short-term neurons in different layers are connected with crossover connections. This establishes different memory pathways as shown by the red, gray and blue lines in **Figure 4**. More specifically, we define a memory pathway as one trace of stacked connection of long-term and short-term neurons from the first memory module to the last.The memory pathways in H-SNN have a wide range of time scales. Connections consist of entirely short-term neurons, long-term neurons, and mixture of both types of neurons create memory pathways of shortest, longest, and intermediate time scales, respectively.

To better illustrate this, an example is shown in **Figure 4**. The memory pathways enable hierarchical learning of temporal patterns for the target spike train shown at the top of the figure. Learner neurons in the first layer extract correlation information from the immediate past (single spike in **Figure 4**) and transfer such knowledge to neurons in memory pathway (two different compositions of a single spike in **Figure 4**). The memory of layer 1 creates a higher level of temporal abstraction of input features that are transmitted as spike inputs to the learner neurons in layer 2. Hence, learner neuron in layer 2 learns compositions of the higher level temporal patterns perceived by memory modules in layer 1 (see bottom of **Figure 4** for illustration). This process is repeated throughout the learning process creating a hierarchical learning of temporal features

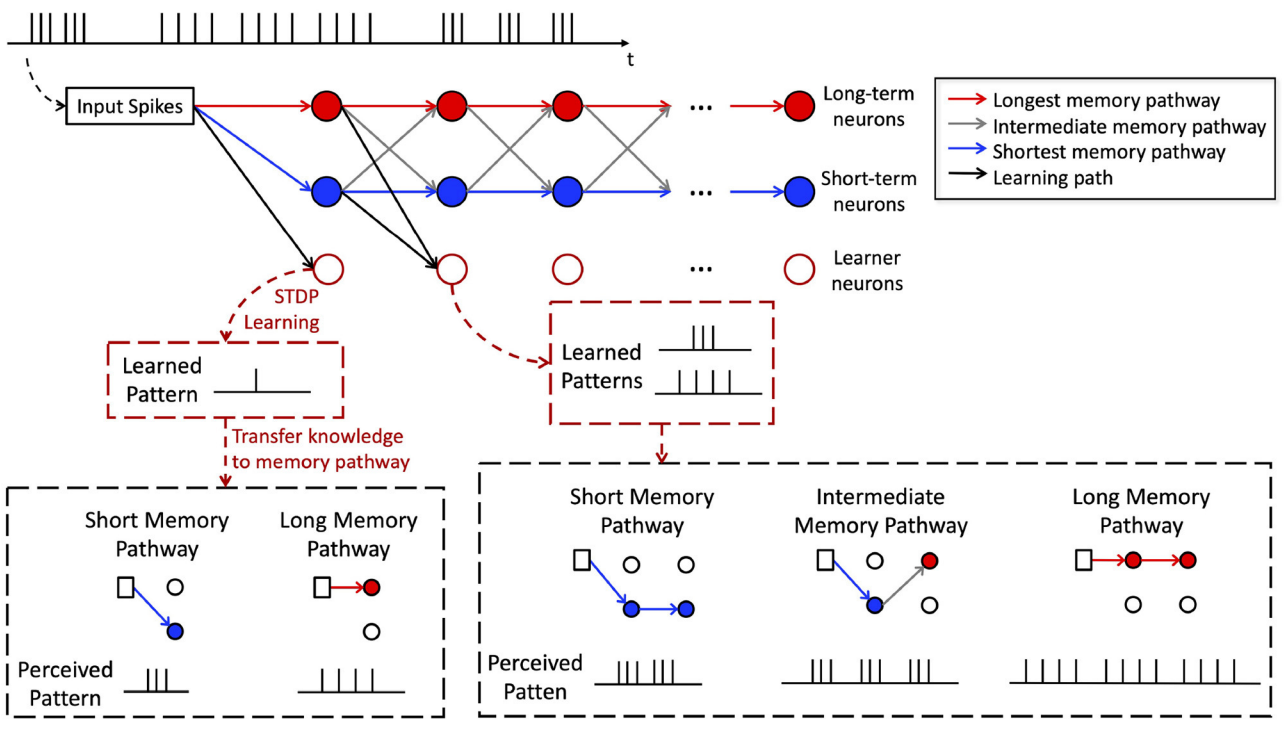

FIGURE 4 | Illustration of memory pathway and hierarchical learning in H-SNN.

with learner neurons in the last layer learning highest level temporal features over the longest time-window. In other words, the equivalent STDP learning window expands along network layers, with long memory pathways creating the faster expansion and short memory pathways creating the slower expansion. Temporal features of different scales can therefore be learned without supervision.

2.4. Spatiotemporal Pattern Representation in H-SNN

For H-SNN to predict object class and motion dynamic, memory modules needs to transform the input sequences to output space that contains spatial (motion invariant) features of target classes and spatiotemporal features of target dynamics. The last layer of H-SNN trained with SGD can statistically correlate those attributes in the reduced dimension to output targets. As H-SNN uses spatial architecture based on spiking convolutional module, H-SNN models spatial features similar to a traditional CNN. However, unlike the global optimization of parameters using gradient descent in CNN, H-SNN uses local, unsupervised STDP to learn spatial patterns in a layer-by-layer fashion, such as shown in prior works (Kheradpisheh et al., 2018; Srinivasan and Roy, 2019).

A network must memorize distinguishable information over various time scales to model temporal patterns. Since H-SNN does not use recurrent connections, we analytically show that the feed-forward connections of spiking neurons with different dynamics can represent various temporal patterns, through proving two properties of H-SNN: output of different memory pathways is distinguishable and different memory pathways can retain temporal information of different length.

We first show that memory pathways have different response function, i.e. different mapping from input spike pattern to output spike pattern (Lemma 2.1). As information is represented by spike frequency (rate encoding), this proves that each memory pathway has distinguishable information (Lemma 2.2). Next, we demonstrate that different memory pathways represent information over different time scales (Lemma 2.4). The following steps show that the H-SNN can represent temporal patterns without using recurrent connections.

2.4.1. Neuron Decay Rate

First, consider the LIF neuron as described by (1), as reproduced here with parameters b and c:

$$dv/dt = a + bv + cI \text{ and } v = v_{reset}, \text{ if } v > v_{threshold}$$

Without input current, the differential equation for membrane potential of neuron m can be re-written a first order separable ordinary differential equation, and its solution leads to:

$$\frac{1}{a + bv_m} dv_m/dt = 1$$

$$v_m(t) = \frac{1}{b} e^{bt+Cb} - \frac{a}{b} \tag{5}$$

Here, C is the constant of integration. A single input spike with current I drives membrane potential to:

$$v_m^{ltn} = v_{reset} + cI$$

Consider initial condition at $t = 0$ with $v_m = v_{reset} + cI$, value of integration constant C can be found, and by substituting it (5) can be rewritten:

$$C = \frac{1}{b} ln(bv_{reset} + bcI + ab)$$

$$v_m^{ltn}(t) = (v_{reset} + cI + \frac{a}{b}) e^{bt} - \frac{a}{b}$$

After neuron m received the input spike, the time it takes for the membrane potential to decay to $v_m^{ltn} = v_{reset}$ is:

$$t_{decay} = \frac{1}{b} ln(v_{reset} + \frac{a}{b}) - \frac{1}{b} ln(v_{reset} + \frac{a}{b} + cI)$$

It can be observed that decay rate is different for the three types of neuron and their parameters $\{a, b, c\}$ differ.

2.4.2. Distinguishability of Memory Pathways

We first derive the spike response function of memory pathways, which formulates output spike frequency given input spike frequency. Based on this, we show the distinguishability of memory pathways by analyzing their response functions.

Lemma 2.1. *Let $\{n_1, n_2, ...n_m\}$ be a list of sequential connected spiking neurons that have uniform refractory period r, and minimal number of input spike needed to reach spiking threshold $\{\gamma_1, \gamma_2, ...\gamma_m\}$. With f_{in} being the input spike frequency to n_1, the output spike frequency of n_m is:*

$$f_{n_m} = (r + \frac{\prod_{i=1}^{m} \gamma_i}{f_{in}} + r \sum_{k=0}^{m-2} \prod_{j=0}^{k} \gamma_{m-j})^{-1}$$

Proof. Spiking event at time t^i can be marked with Dirac delta function $\delta(t - t^i)$. For a spiking neuron m, input current from all pre-synaptic neurons is $I_m = \sum_{n,i} G_{mn} \delta(t - t_n^i)$, where G_{nm} is the synapse conductance between neuron n and m, and the sum is over all pre-synaptic spiking events. Integration of the input current can be performed for each pre-synaptic spike. With initial state at $t = 0$ with $v = v_{reset}$, solving (1) leads to:

$$v_m(t) = v_{reset} e^{bt} - \frac{a}{b}(1 - e^{bt}) + ce^{bt} \sum_n \sum_i G_{mn} \int_0^t \delta(t - t_n^i) e^{-bt} dt \tag{6}$$

Now consider neuron m with one pre-synaptic neuron that has output frequency f_{in} and $t^0 = 0$, the time t^γ for neuron m to reach spiking state is $v_m(t^\gamma) >= v_{th}$, where v_{th} is the threshold voltage. Since the membrane potential increases at time of t^i and decays otherwise, t^γ will be one instance of t^i. When we set $t = t^i$ and expand the equation for v_m, we have:

$$v_m(t^i) = (v_{reset} + \frac{a}{b}) e^{bt^i} - \frac{a}{b} + cGe^{bt^i - bt^1} + cGe^{bt^i - bt^2} + ... + cG \tag{7}$$

The post-synaptic neuron m has output spike frequency that can be found with:

$$f_{out} = F(f_{in}) = \frac{1}{T + r} = \frac{1}{(\gamma / f_{in}) + r} = \frac{f_{in}}{\gamma + r f_{in}} \tag{8}$$

where r is the refractory period, and $t^{\gamma} = \{\min t^{\gamma} \mid v_m(t^{\gamma}) \geq v_{th}\}$. $F(f_{in})$ is referred to as the neuron response function. Consider a long-term neuron and a short-term neuron connected in series with one input neuron at spiking frequency f_{in}, the response function would be a composition of two neurons' response function:

$$(F_{stn} \circ F_{ltn})(f_{in}) = (\frac{\gamma_{ltn}\gamma_{stn}}{f_{in}} + \gamma_{stn} r + r)^{-1} \tag{9}$$

By doing the same composition for more neurons, we can generalize the pattern, and response function of memory pathway ($\Phi(f_{in})$) that consists of m neurons can be derived as follows:

$$f_{n_m} = \Phi(f_{in}) = (r + \frac{\prod_{i=1}^{m} \gamma_i}{f_{in}} + r \sum_{k=0}^{m-2} \prod_{j=0}^{k} \gamma_{m-j})^{-1} \tag{10}$$

The proof is complete.

Lemma 2.2. *The response function (10) for different memory pathways are distinct from each other.*

Proof. Consider memory pathway formed by two neurons $p_1 = \{n_{ltn}, n_{stn}\}$ with $\gamma_{ltn}, \gamma_{stn}$ connected in order, and another memory pathway formed by $p_2 = \{n_{stn}, n_{ltn}\}$ with $\gamma_{stn}, \gamma_{ltn}$ connected in order. The response function of the two memory pathways is:

$$\begin{aligned} \Phi_{p_1}(f_{in}) &= (\frac{\gamma_{ltn}\gamma_{stn}}{f_{in}} + \gamma_{stn} r + r)^{-1} \text{ and} \\ \Phi_{p_2}(f_{in}) &= (\frac{\gamma_{stn}\gamma_{ltn}}{f_{in}} + \gamma_{ltn} r + r)^{-1} \end{aligned} \tag{11}$$

It can be observed that $\Phi_{p_1} \neq \Phi_{p_2}$, and that response function (10) is dependent on the order of γ. Each memory pathway in H-SNN has sequentially connected neurons that form an ordered list of γ. Since all memory pathways have different connection sequences, which leads to different sequences of γ, response function Φ is different for each memory pathway. The proof is complete.

2.4.3. Retention Length of Memory Pathways

Here, we first prove the existence of cut-off frequency of spiking neurons, followed by analysis of retention length of memory pathways.

Lemma 2.3. *There exists a cut-off pre-synaptic spike frequency to a post-synaptic neuron below which the post-synaptic neuron cannot spike.*

Proof. From the membrane potential equation derived in the main paper for Lemma 3.1:

$$v_m(t^i) = (v_{reset} + \frac{a}{b}) e^{bt^i} - \frac{a}{b} + cGe^{bt^i - bt^1} + cGe^{bt^i - bt^2} + ... + cG$$

Since $\Delta t = \frac{1}{f} = t^{i+1} - t^i$, subtracting membrane potential values at two consecutive t_i provides:

$$\begin{aligned} \Delta v_m = v_m(t^{i+1}) - v_m(t^i) &= (v_{reset} + \frac{a}{b}) e^{bt^{i+1}} - (v_{reset} + \frac{a}{b}) e^{bt^i} \\ &+ cGe^{bt^{i+1} - bt^1} \end{aligned} \tag{12}$$

setting t^1 to zero, we have:

$$\Delta v_m = (v_{reset} + \frac{a}{b}) e^{bt^i} (e^{b\Delta t} - 1) + cGe^{bt^i} e^{b\Delta t}$$

$$\Delta v_m = ((v_{reset} + \frac{a}{b})(e^{b\Delta t} - 1) + cGe^{b\Delta t}) e^{bt^i} \tag{13}$$

As the term $((v_{reset} + \frac{a}{b})(e^{b\Delta t} - 1) + cGe^{b\Delta t})$ does not depend on t^i, v_m is either strictly increasing, staying the same or decreasing with higher t^i. This indicates that, when $\Delta v_m \leq 0$ the post-synaptic neuron can never spike regardless of how many pre-synaptic spike it receives. For post-synaptic neuron with specific parameters and synapse conductance G, $((v_{reset} + \frac{a}{b})(e^{b\Delta t} - 1) + cGe^{b\Delta t})$ depends on Δt. Since $\Delta t = \frac{1}{f}$, there exists a value of pre-synaptic frequency f below which the post-synaptic neuron cannot spike. The proof is complete.

Lemma 2.4. *The duration for which each memory pathway can store information is different.*

Proof. Given a memory pathway consist of neurons $\{n_1, n_2, ...n_m\}$ connected in order with input spike to n_1 at frequency f_{in}^0. There exists a cutoff frequency f_c^i for each neuron n_i. If input spike frequency to n_i is lower than f_c^i, the neuron cannot spike. This value can be mapped to original input frequency f_{in}^0 with (10) of the memory pathway ending at n_{i-1}: $f_c^i = \Phi n_{i-1}(f_{in}^0)$. The mapped cutoff frequency is referred to as f_{mapped}^i. Since a neuron must receive at lease one input spike to reach threshold, along memory pathway f_{mapped}^i decreases, or holds the same. Therefore, the longest duration T that the memory pathway can hold information is $T = (f_{mapped}^m)^{-1}$. Using (10), we have: $f_c^m = \Phi n_{m-1}(f_{mapped}^m)$, which leads to:

$$f_{mapped}^m = (\frac{1}{f_c^m} - r - r \sum_{k=0}^{m-3} \prod_{j=0}^{k} \gamma_{m-1-j})^{-1} \prod_{i=1}^{m-1} \gamma_i \tag{14}$$

$$T = (\frac{1}{f_c^m} - r - r \sum_{k=0}^{m-3} \prod_{j=0}^{k} \gamma_{m-1-j})(\prod_{i=1}^{m-1} \gamma_i)^{-1} \tag{15}$$

Following the same process used in proof of Lemma 2.2, we can show that T is different for each memory pathway. The proof is complete.

3. RESULTS

3.1. Parameters and Simulation Configurations

In this work we use a GPU accelerated SNN simulator we developed as an open source project named ParallelSpikeSim (She et al., 2019). The manually tuned neuron parameters are: for learner neurons, a is –4.1, b is –0.01, c is 0.31, and r (refractory period) is 20 ms; for short-term neurons, a is –5.1, b is –0.02, c is 0.45, and r is 10 ms; for long-term neuron, a is –1.6, b is –0.001, c is 0.16, and r is 10 ms. Values of STDP parameters are: $\alpha_p = 0.1, \alpha_d = 0.03, G_{max} = 1.0$, $G_{min} = 0$, $\tau_{pot} = 10$ ms, and $\tau_{dep} = 80$ ms. To prevent early convergence of synaptic conductance and allow the network to effectively learn the entire dataset, the values of α_p and α_d are chosen to be relatively small, and training data is shuffled for both class and motion categories. In terms of simulation, the unit timestep is set to be 1 ms. Input frames are converted to spike trains with pixel intensity proportional to spike frequency range $f_{min}^{input} = 0$ Hz and $f_{max}^{input} = 100$ Hz. Time spent on each frame is $t_{train} = 300$ ms. All neuron states are reset to default value after learning of each sequence.

3.2. Quantitive Analysis

Comparing t_{decay} when m is a long-term neuron and when m is a short-term neuron, starting from the same membrane potential level, the ratio of the time it takes for two neurons to decay to v_{reset} is:

$$\frac{t_{decay}^{ltn}}{t_{decay}^{stn}} \approx 2.4$$

From (13), we can find the cut-off frequency value by setting $\Delta v_m = 0$, which gives:

$$(v_{reset} + \frac{a}{b})(e^{b\Delta t} - 1) = -cGe^{b\Delta t}$$

$$\Delta t = \frac{1}{b} ln(\frac{v_{reset} + a/b}{v_{reset} + a/b - cG})$$

In terms of cut-off frequency,

$$f_0 = \frac{1}{\Delta t} = \frac{b}{ln(\frac{v_{reset}+a/b}{v_{reset}+a/b-cG})}$$

Comparing three neuron types with values of parameters $\{a, b, c\}$ as shown before, under different synapse conductance, the cut-off frequency of each neuron is shown in **Table 1**. It can be observed that, for all three neuron types, the cut-off frequency has a strong dependency on synapse conductance.

3.3. Baseline Networks

Five DNNs are implemented to represent baselines for spatiotemporal processing, namely, (i) a simple 3D CNN (Tran et al., 2015) referred to as 3D CNN-α, which has similar layer configurations as H-SNN, (ii) 3D CNN-β, a more complex 3D CNN with more layers and parameters, (iii) 3D MobileNetV2 and (iv) 3D ShuffleNetV2 as implemented in Köpüklü et al. (2019). The fifth baseline for comparison is an implementation of CNN+LSTM (Donahue et al., 2015). To prevent overfitting, for 3D CNN-α and 3D CNN-β dropout layers are applied; for all DNN baselines, early stopping for training are used. In Wu et al. (2018), spatiotemporal back-propagation is shown for SNN and the trained network is tested for dynamic dataset. We implement this design with two variants as additional bio-inspired baseline networks. The first variant is referred to as BP-SNN, which has the same convolution layer configuration as H-SNN but does not use neurons of different dynamics. Based on the original BP-SNN structure, we implement refractory period to the neurons, and modified each layer to include neurons with long and short memory, similar to H-SNN. This second variant is referred to as BP-SNN-LS.

3.4. Network Complexity and Energy Dissipation

The configurations of convolution layers and network parameter number are shown in **Table 2**. The tested H-SNN has 0.74 million parameters. 3D CNN-α has similar complexity as H-SNN with 0.83 million parameters and BP-SNN has the same number of parameters as H-SNN. On the other hand, 3D CNN-β and CNN+LSTM contains 4.5 million and 3.7 million parameters. 3D MobileNetV2 and 3D ShuffleNetV2 has 0.5x complexity (Köpüklü et al., 2019) and contains more parameters

TABLE 1 | Cut-off frequency f_0 (Hz) of post-synaptic neuron with one pre-synaptic neuron as input at different synapse conductance.

Neuron type	G = 0.1	G = 0.2	G = 0.3	G = 0.4	G = 0.5	G = 0.6
Learner	105.8	52.9	35.2	26.4	21.1	17.6
Short-term	78.9	39.5	26.3	19.7	15.8	13.1
Long-term	63.5	31.7	21.1	15.8	12.7	10.6

TABLE 2 | Network configurations.

Model	Convolution layer configuration	Total parameter
3D CNN-α	Conv3D{[3x3x3,20],[3x3x3,32], [5x5x5,64], [5x5x5,64]}	0.83M
3D CNN-β	Conv3D{[3x3x3,32],[5x5x5,64], [3x3x3,96], [3x3x3,128]x2}	4.5M
3D MobileNetV2	(Köpüklü et al., 2019)	1.5M
3D ShuffleNetV2	(Köpüklü et al., 2019)	1.2M
CNN+LSTM	Conv2D{[3x3,64],[3x3,128],[5x5,256]}	3.7M
BP-SNN/BP-SNN-LS	Conv2D{[3x3,32],[3x3,64], [5x5,128], [7x7,40]}	0.74M
H-SNN	Conv2D{[3x3,32],[3x3,64], [5x5,128], [7x7,40]}	0.74M

than H-SNN. For CNN+LSTM, parameters in the CNN encoder is 2.7M, while that in the LSTM decoder is 1.0M.

For H-SNN, network memory consist of two main parts: (i) synapse conductance, which are trainable parameters, use 0.474M, and (ii) neuron state, which are non-trainable variables, use 0.267M to store all membrane potential. The number of H-SNN's trainable parameters is thus significantly less than 3D CNN-β and CNN+LSTM. Moreover, it is well-known that the event-driven nature of SNN assists in reducing network activation which in turn reduced energy dissipation of computation. Using method presented in Panda et al. (2020), we compute the energy advantage of H-SNN over DNN baselines during inference as follows: 1.0 for H-SNN, 1.55 for 3D CNN-α, and 3.37 for 3D ShuffleNetV2; 3D MobileNetV2, 3D CNN-β, and CNN+LSTM consumes 9.01×, 11.80× and 28.88× higher energy than H-SNN, respectively. BP-SNN and BP-SNN-LS uses similar energy as H-SNN.

3.5. Single-Objective Prediction

3.5.1. Experimental Details

To test the effectiveness of H-SNN in learning spatiotemporal patterns, both single-objective and multi-objective experiments are conducted. In the single-objective experiment, an event camera dataset of human gesture (Amir et al., 2017) is used. Here, individual events are superimposed onto frames with resolution of 128x128 over 20 ms window. Each generated sequence contains one hundred frames and the network learns to predict the type of action in each sequence. Three baseline networks are tested: 3D CNN-α as well as the more complex 3D MobilNetV2 and 3D ShuffleNetV2.

To study the feasibility of learning with less labeled data and benefit of unsupervised learning, three sets of experiments are performed on DNN baselines using different training set sizes: 100, 50, and 30% of the full training data set. In terms of H-SNN, two training configurations are tested: one that uses the same training set as DNN for STDP learning and SGD-based final layer tuning, referred to as **H-SNN**; the other, referred to as **H-SNN (full data)**, uses the full training set for STDP unsupervised learning, while SGD-based tuning uses the same set as DNN.

3.5.2. Results

As shown in **Table 3**, accuracy results from the three sets of experiments are listed. With full training dataset, accuracy of H-SNN is on a comparable level with 3D MobileNetV2 and 3D ShuffleNetV2 and outperforms 3D CNN-α. With less amount of labeled training data, all networks experience performance degradation. Among tested networks, H-SNN (full data) has the lowest accuracy decrease, while other networks lose around 7% from 100% training data to 30%. Such difference leads to the higher accuracy of H-SNN (full data) in low training data conditions, which indicates that unsupervised STDP learning of unlabeled data is effectively improving spatiotemporal pattern recognition in this particular task.

TABLE 3 | Accuracy result of event camera dataset for networks trained with 100, 50, and 30% of labeled training data.

Model	100%	50%	30%
3D CNN-α	92.8	90.5	86.3
3D MobileNetV2	97.0	94.2	90.4
3D ShuffleNetV2	97.3	95.4	90.1
H-SNN	96.2	93.8	**90.9**
H-SNN (full data)	96.2	**95.8**	**93.7**

H-SNN accuracy results that exceed all baselines are marked in bold.

3.6. Multi-Objective Prediction

3.6.1. Experimental Details

The second set of experiments is designed as a multi-objective computer vision task: the network observes a moving object as visual input to predict the class of the object and dynamic of its motion. We generate dataset with controlled motion dynamics by extracting objects from an aerial image dataset (Xia et al., 2017). A subset (20%) of the original training data for each object class is used with label to test the benefit of unsupervised learning. In order to generate transformation sequences, objects are placed on canvas and applied with translation and rotation, each with five possible dynamics: static, constant speed, accelerating, decelerating, and oscillating. For the training sequence, transformation dynamics are generated with parameters P_{train}. For the test sequence, objects are taken from the test set of Xia et al. (2017) and transformation dynamic parameters are drawn from Gaussian distribution with mean P_{train} and standard deviation σ_{ts}. As a second, non-aerial test case, we have performed similar experiments on Fashion-MNIST dataset which are 10 classes of apparel items with the dimension of 28 by 28, as objects. Training and test sequences are generated following the same process as discussed above, except that 10% of the original training data is used. For all generated sequences, training data is shuffled for both object class and motion dynamic.

In addition to regular training/inference setup, to understand the network's capability to learn class and motion independently, a total of three sets of experiments are conducted:

- ***Experiment 1: all-objective prediction*** In this regular training/inference setup, training set contains all classes and all possible transformation dynamics, and networks are tested for prediction of object class and its rotation/translation dynamics, which is either static, constant speed, accelerating, decelerating, or oscillating.
- ***Experiment 2: class-agnostic motion prediction*** In this experiment, networks are trained with sequences containing objects belonging to half of all classes, and tested for transformation dynamics prediction on the other half classes.
- ***Experiment 3: motion-agnostic class prediction*** Training sequences contain all object classes, but with only a subset of motion dynamics. Networks are tested for accuracy of class prediction but with unknown dynamics.

Experiment 1 is conducted on both the aerial and Fashion-MNIST dataset, while the other two are tested on the aerial dataset only. Examples of training/test sequences for the three

experiments are shown in **Figure 5**. All baseline networks as discussed in section 3.3 are tested.

3.6.2. Training Configurations

We test H-SNN with two training schemes for the aerial dataset. The first one, referred to as **H-SNN (1xU)**, uses 20% of the training set mentioned before for STDP learning and SGD-based final layer tuning. To study the advantage of training with unlabeled data, we create a second network, **H-SNN (5xU)**, that uses 5× more unlabeled data during STDP learning in SNN while the SGD-based tuning of the last year still uses the original labeled training set same as H-SNN (1xU). For Fashion-MNIST, **H-SNN (1xU)** uses sequences generated with 10% of original training set per class for unsupervised learning and final layer tuning; **H-SNN (10xU)** uses the whole training set for unsupervised learning and the same 600 objects per class sequences as used in H-SNN (1xU) for final layer supervised tuning. All baselines, including DNNs and SNN trained with back-propagation, are trained with the dataset used by H-SNN (1xU).

3.6.3. Aerial Footage Results

3.6.3.1. All-objective prediction

In all-objective prediction, results are measured by four metrics: accuracy for three separate targets and joint accuracy, which accounts for predictions that are correct for all three separate targets. Each objective's individual accuracy: class, rotation, and translation, is referred to as C, R, and T. Training accuracy for all networks are shown in **Table 4**. From the test accuracy

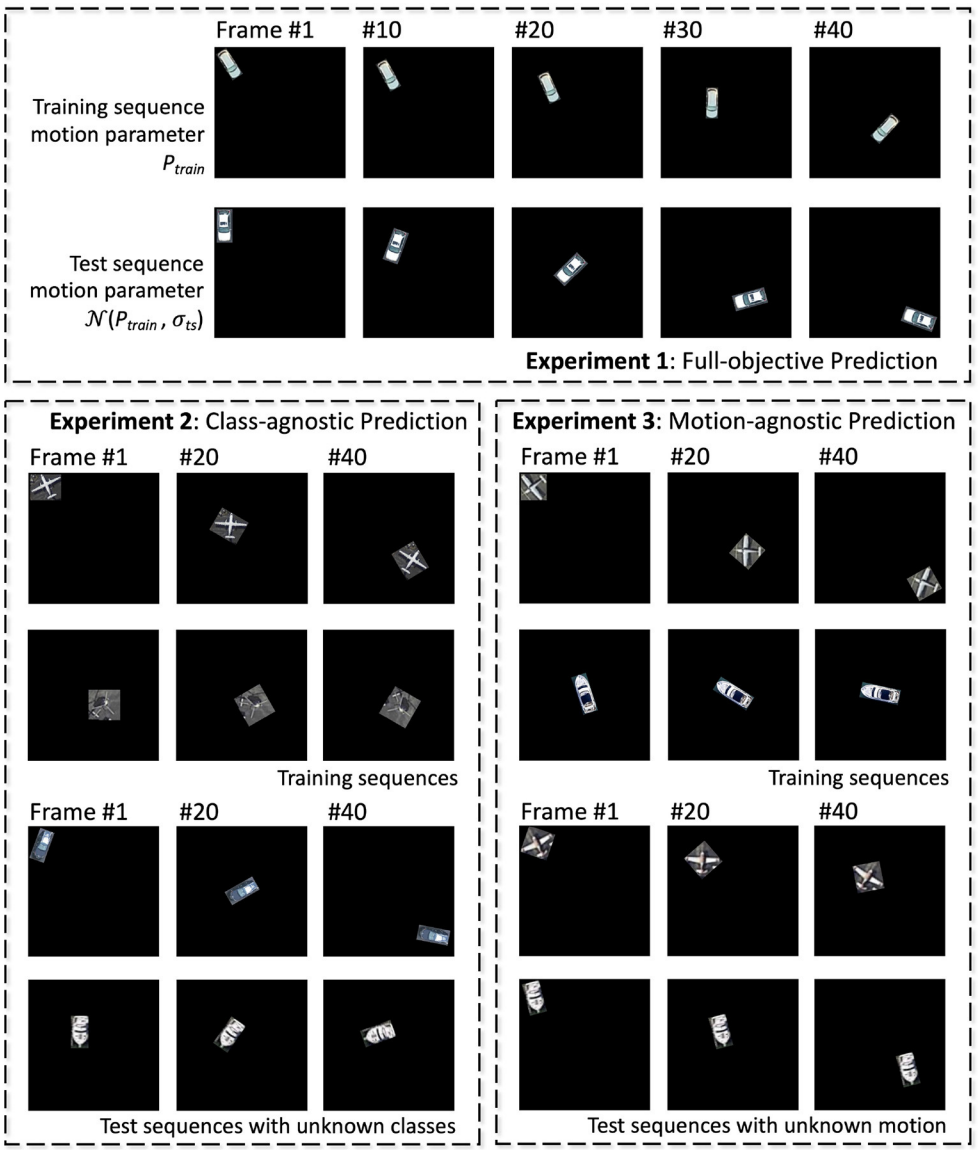

FIGURE 5 | Illustrations of training and test sequences for the three experiments of the aerial footage dataset.

results in **Table 5**, it can be observed that 3D MobileNetV2 and 3D ShuffleNetV2 show better accuracy than 3D CNN-α while they both fall behind 3D CNN-β and CNN+LSTM. BP-SNN-LS demonstrates better performance than BP-SNN in rotation and translation targets, while its class prediction is similar with BP-SNN.

With $\sigma_{ts} = 1$, H-SNN (1xU) predicts with good accuracy for motion dynamics and achieves a reasonable level of class prediction accuracy. This indicates that H-SNN is able to learn spatiotemporal patterns from moving objects and predicts for separate objectives based on the learned patterns. Comparing H-SNN (5xU) to H-SNN (1xU), unsupervised learning provides considerable performance increase for all targets. With increasing σ_{ts}, accuracy for class prediction does not experience drastic degradation, showing that visual features learned by the network have a high degree of transformation invariance.

In comparison with baseline networks, accuracy values where H-SNN exceeds all baselines are marked bold in **Table 5**. For $\sigma_{ts} = 1$, H-SNN (1xU) outperforms BP-SNN and 3D CNN variants except for 3D CNN-β, while H-SNN (5xU) shows accuracy on a par with 3D CNN-β and CNN+LSTM. The advantage of H-SNN (1xU) is more evident in predicting motion with high deviation, as it achieves better results than baseline networks. This indicates that H-SNN is able to generalize more effectively the transformation invariant/equivariant patterns. With extra unlabeled dataset used for SNN learning, H-SNN (5xU) outperforms all baselines networks in most metrics. BP-SNN shows comparable accuracy as H-SNN (1xU) for class prediction, while its performance for motion prediction is noticeably lower than H-SNN (1xU), especially at higher σ_{ts}. BP-SNN-LS has similar performance with H-SNN (1xU) while still outperformed by H-SNN (5xU).

Confusion matrices for $\sigma_{ts} = 5$ are shown in **Figure 6A**. For each of the two variants of H-SNN, results are presented with three matrics. The matrix on the left is for class prediction, the top right is for rotation and the bottom right is for translation. Horizontal axis is predicted label and vertical axis is target label, each marked with a number; for class prediction, 0-9 represents the 10 classes of objects; for rotation and translation, 0 is static, 1 is constant speed, 2 is acceleration, 3 is deceleration, and 4 is oscillation. Lighter color represents more instances. It can be observed that, in terms of class prediction, confusion matrices of H-SNN (1xU) and H-SNN (5xU) share similarities, while H-SNN (5xU) predicts with more consistency across all classes. For motion prediction, learning unlabeled data has different effect: errors of H-SNN (5xU) for rotation prediction is more concentrated in one dynamic, while its errors for translation prediction spreads out more evenly, compared to H-SNN (1xU).

TABLE 4 | Training accuracy of all-objective prediction with aerial footage dataset.

Model	Joint	{C,R,T}
3D CNN-α	80.5	90.8, 94.9, 93.4
3D CNN-β	84.8	91.3, 96.5, 96.3
3D MobileNetV2	88.4	92.4, 97.7, 97.9
3D ShuffleNetV2	87.8	92.6, 96.8, 97.9
CNN+LSTM	86.7	90.7, 97.5, 98.1
BP-SNN	87.4	89.5, 98.9, 98.7
BP-SNN-LS	85.8	90.4, 97.5, 97.3
H-SNN (1xU)	84.8	91.4, 96.5, 96.1
H-SNN (5xU)	89.6	92.3, 98.1, 99.0

3.6.3.2. Class-agnostic motion prediction

Table 6 lists accuracy of class-agnostic motion prediction. Each cell in the table contains accuracy for rotation (R) and translation (T). Result shows that the two H-SNN implementations are able to successfully predict motion dynamics of objects from unknown classes. Compared to motion dynamic accuracy from all-objective prediction, we observe that H-SNN experiences some degree of performance degradation. However, the decrease in accuracy is not drastic, especially for H-SNN (5xU). This shows that H-SNN is able to learn and predict motion dynamics independent of object's visual features on a certain level.

Among conventional deep networks, 3D CNN-β and CNN+LSTM show better accuracy than 3D CNN-α by a significant lead. 3D ShuffleNetV2 performs well in predicting

TABLE 5 | Test accuracy of all-objective prediction with aerial footage dataset.

	$\sigma_{ts} = 1$		$\sigma_{ts} = 3$		$\sigma_{ts} = 5$	
Model	**Joint**	**{C,R,T}**	**Joint**	**{C,R,T}**	**Joint**	**{C,R,T}**
3D CNN-α	51.3	64.5, 87.9, 90.4	23.2	58.9, 61.9, 63.5	16.0	56.1, 51.1, 55.8
3D CNN-β	58.6	67.1, 94.7, 92.2	31.6	64.9, 66.1, 73.7	25.3	62.8, 60.3, 66.9
3D MobileNetV2	54.7	66.9, 88.3, 92.7	24.2	64.1, 53.8, 70.1	18.0	61.9, 47.1, 62.0
3D ShuffleNetV2	53.8	64.8, 89.8, 92.5	26.1	63.2, 55.8, 73.9	19.9	60.8, 49.7, 65.9
CNN+LSTM	56.1	66.2, 92.8, 91.3	28.4	63.0, 68.3, 66.0	23.3	61.3, 63.8, 59.5
BP-SNN	49.5	65.1, 86.7, 88.3	23.6	61.7, 59.2, 64.6	17.4	60.4, 50.1, 57.3
BP-SNN-LS	58.1	66.7, 92.6, 94.1	32.5	60.1, 68.8, 78.7	26.7	58.3, 65.3, 70.2
H-SNN (1xU)	56.2	66.8, 91.0, 92.4	**34.0**	64.4, **70.5**, 74.8	**29.0**	**63.2**, **66.1**, 69.7
H-SNN (5xU)	**68.0**	**72.8**, 94.4, **98.8**	**44.6**	**69.5**, **78.4**, **81.8**	**32.9**	**66.4**, **68.3**, **72.6**

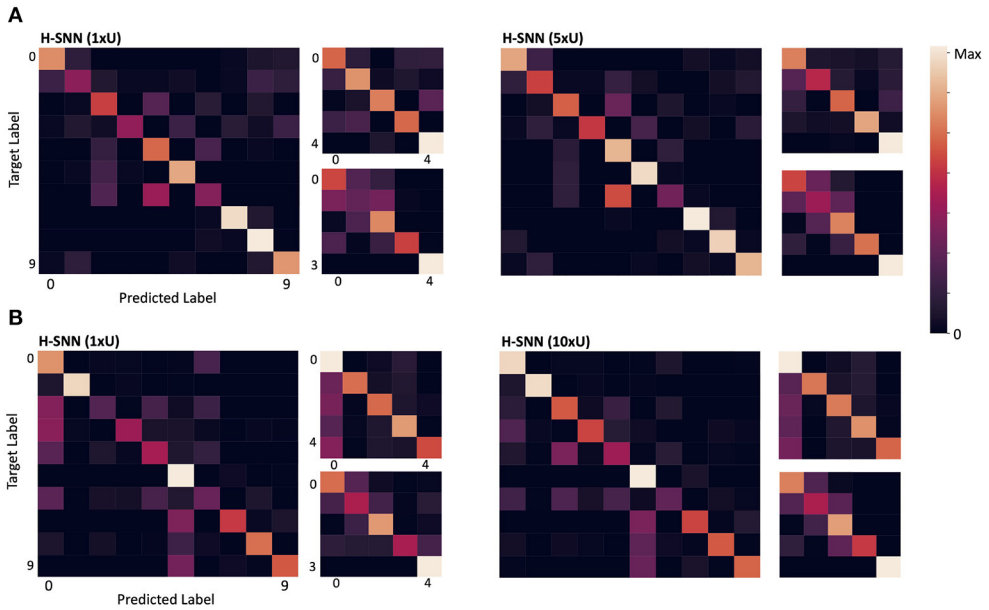

FIGURE 6 | Confusion matrix for **(A)** aerial image sequences and **(B)** Fashion-MNIST sequences; for each network, the left matrix is for class prediction, the top right for rotation and the bottom right for translation; darker color represents less instances.

TABLE 6 | {R,T} prediction accuracy for unknown classes with aerial footage dataset.

Model	$\sigma_{ts} = 1$ {R,T}	$\sigma_{ts} = 3$ {R,T}	$\sigma_{ts} = 5$ {R,T}
3D CNN-α	86.4, 88.7	60.7, 64.9	52.3, 56.3
3D CNN-β	95.2, 91.4	70.3, 65.3	61.4, 59.3
3D MobileNetV2	91.0, 83.6	67.1, 67.4	56.0, 61.6
3D ShuffleNetV2	97.5, 81.9	68.2, 62.2	67.6, 57.8
CNN+LSTM	93.1, 87.6	72.5, 66.7	63.1, 57.7
BP-SNN	84.7, 85.9	57.5, 64.2	51.4, 58.6
BP-SNN-LS	86.5, 94.8	62.3, 78.2	57.5, 60.4
H-SNN (1xU)	88.6, 91.0	68.0, 73.4	64.0, **63.2**
H-SNN (5xU)	92.3, **98.4**	**72.7**, **80.7**	66.3, **70.7**

H-SNN accuracy results that exceed all baselines are marked in bold.

TABLE 7 | Motion-agnostic class prediction: (left) configurations for class prediction test with unknown transformations where $\{s, c, a, d, o\}$: static, constant speed, accelerating, decelerating and oscillating; (right) accuracy of class prediction for different test cases with aerial footage dataset.

	Training dynamics	Test dynamics
Pair I	T: $\{s, a, d, o\}$	T: $\{c\}$
	R: $\{a, d\}$	R: $\{c, s, o\}$
Pair II	T: $\{a, d\}$	T: $\{c, s, o\}$
	R: $\{s, a, d, o\}$	R: $\{c\}$
Pair III	T: $\{a, d\}$	T: $\{c, s, o\}$
	R: $\{a, d\}$	R: $\{c, s, o\}$

Model	Pair I	Pair II	Pair III
3D CNN-α	52.7	50.1	45.4
3D CNN-β	57.5	52.5	51.9
3D MobileNetV2	51.9	47.1	50.6
3D ShuffleNetV2	59.1	56.2	59.3
CNN+LSTM	56.2	56.6	53.1
BP-SNN	49.3	48.2	43.0
BP-SNN-LS	51.2	53.6	55.4
H-SNN (1xU)	**60.4**	54.0	53.4
H-SNN (5xU)	**64.7**	**60.6**	58.9

H-SNN accuracy results that exceed all baselines are marked in bold.

rotation dynamic, while 3D MobileNetV2 shows good accuracy for translation dynamic. BP-SNN-LS has good performance for translation prediction, while the spatiotemporal patterns learned by BP-SNN are less generalizable, as its performance lags behind H-SNN and other baselines in this test. With increasing variation in transformation parameters, as observed in the $\sigma_{ts} = 3$ and $\sigma_{ts} = 5$ cases, accuracy of all networks degrade considerably. Accuracy of H-SNN (1xU) is on similar level with the more complex DNNs and BP-SNN-LS, and higher than 3D CNN-α and BP-SNN. H-SNN (5xU) shows comparable or better performance than best baseline performance. Similar to all-objective prediction, the advantage of H-SNN (5xU) is more evident for high σ_{ts} cases.

3.6.3.3. Motion-agnostic class prediction

The three training/test pairs of motion dynamics are shown in **Table 7** (left), and results for each pair is shown in **Table 7** (left). H-SNN is able to learn motion invariant spatial patterns as it predicts object classes with reasonable accuracy. For this task, accuracy of H-SNN (1xU) can again be improved further using more unlabeled data as shown in the H-SNN (5xU) results. This indicates a better generalization ability of H-SNN (5xU) for objects with unknown transformations.

In this test, 3D ShuffleNetV2 provides better performance than all other baselines. Compared to other networks, H-SNN (1xU) performs well for Pair I while keeping its advantage over 3D CNN-α and BP-SNN for all test cases. For Pair II, it is on a par with the best baseline results. H-SNN (5xU) achieves considerable improvement for all pairs by providing the best accuracy results except for Pair III, where it slightly falls behind 3D ShuffleNetV2. Hence, we observe that patterns learned from unlabeled data by STDP reduces the impact of unknown transformations to class prediction. For BP-SNN, degradation from all-objective prediction is significant, which indicates that the network has difficulty in learning spatial patterns invariant to unknown transformations. Compared to BP-SNN, BP-SNN-LS shows improvement in Pair II and Pair III while performs similarly in Pari I.

It is also worth noting that, compared to previously shown class prediction results, combinations of training/test dynamics affect network performance differently. For example, Pair II causes more degradation to 3D CNN-β than to CNN+LSTM while Pair I has similar influence to the two networks. For BP-SNN, Pair III shows to be most challenging. This indicates that the spatial patterns learned by networks generalize differently for unseen motion dynamics.

3.6.4. Fashion-MNIST Results

As shown in **Table 8**, both variants of H-SNN provide good accuracy for the three targets, indicating that the network is able to effectively learn the spatiotemporal patterns in moving apparel items, which have different visual features from the aerial footage dataset. H-SNN (10xU) shows higher accuracy than H-SNN (1xU) for class and translation motion prediction, while the improvement it achieves for rotation prediction is smaller.

From the confusion matrix in **Figure 6B**, the two H-SNN variants show similar profile for rotation and translation predictions while their class prediction differentiates. With unsupervised learning, H-SNN (10xU) is able to predict more accurately for classes that have high error rate, while the improvement on classes with lower error rate is less noticeable.

Among the baseline DNNs, 3D CNN-β shows good result at low σ_{ts} as it performs better than other deep networks and BP-SNN, and shares similar performance with CNN+LSTM in high σ_{ts} in terms of joint accuracy while each individual target differs. BP-SNN-LS shows considerable gain from BP-SNN and has the best performance among baseline networks. H-SNN (1xU) demonstrates similar accuracy as 3D CNN-β for $\sigma_{ts} = 1$ while its accuracy is lower than BP-SNN-LS. At higher σ_{ts}, H-SNN (1xU) shows comparative advantage: the prediction accuracy for multiple targets exceeds all baseline networks. This indicates that, for Fashion-MNIST, H-SNN is able to more effectively learn spatiotemporal patterns generalizable to different transformation dynamics. With unsupervised learning of extra unlabeled sequences, H-SNN (10xU) is able to make better prediction in almost all individual targets, and achieves joint accuracy considerably higher than baseline networks.

3.6.5. Impact of Training Data Size

In this section we investigate the impact of scaling labeled training data on H-SNN, with two types of unsupervised learning setups. For the aerial dataset, unsupervised learning and supervised training of H-SNN both use sequences generated with 20, 60, and 100% of the original dataset. For Fashion-MNIST, three levels of supervised training: 10, 50, and 100%, are tested on a network that learns 100% data without supervision. The results are shown in **Table 9**. For the aerial dataset, it can be observed that by increasing training data size the network experiences considerable improvement on performance. The gain in class prediction accuracy is higher than that in motion prediction and the improvement in general is higher for lower σ_{ts} cases. When the network always learns 100% unlabeled data, similar trend can also be observed as shown in the Fashion-MNIST result. However, the benefit from increasing training data size is smaller than in the aerial dataset, e.g., for $\sigma_{ts} = 1$, joint accuracy increased by around 27% for aerial image, while for Fashion-MNIST the gain is around 9%.

4. DISCUSSION

In this paper we present H-SNN as a novel spiking neural network design that is capable of learning spatiotemporal information with STDP. For H-SNN, no recurrent connection

TABLE 8 | Accuracy result for sequences generated with Fashion-MNIST.

	$\sigma_{ts} = 1$		$\sigma_{ts} = 3$		$\sigma_{ts} = 5$	
Model	**Joint**	**{C,R,T}**	**Joint**	**{C,R,T}**	**Joint**	**{C,R,T}**
3D CNN-α	54.3	70.4, 86.4, 89.2	19.0	50.2, 57.0, 66.5	10.9	43.9, 45.9, 54.4
3D CNN-β	63.1	72.5, 92.6, 94.0	28.4	61.0, 61.3, 75.9	18.2	58.9, 53.8, 57.4
3D MobileNetV2	57.6	70.0, 94.2, 87.3	22.6	58.3, 57.4, 67.6	13.8	53.8, 48.2, 53.1
3D ShuffleNetV2	61.0	68.7, 92.3, 96.2	23.9	53.4, 56.0, 79.8	12.1	43.2, 47.5, 59.0
CNN+LSTM	55.7	69.2, 89.6, 89.8	24.2	55.3, 74.7, 58.5	18.4	53.2, 64.1, 54.1
BP-SNN	54.5	68.1, 85.7, 93.4	19.5	47.0, 56.5, 73.5	14.3	50.6, 47.5, 59.3
BP-SNN-LS	70.0	76.8, 97.1, 94.1	37.2	66.0, 71.6, 78.7	26.0	62.5, 67.3, 61.7
H-SNN (1xU)	65.6	74.1, 96.2, 92.2	**38.9**	64.8, **83.7**, 71.8	**27.6**	59.2, **71.0**, **65.7**
H-SNN (10xU)	**73.9**	**79.4**, 96.8, **96.2**	**47.5**	**70.0**, **84.9**, **79.9**	**31.7**	**65.2**, **71.9**, **67.6**

H-SNN accuracy results that exceed all baselines are marked in bold.

TABLE 9 | Impact of training data size for aerial dataset (top 3 rows) with scaling unsupervised learning data size and Fashion-MNIST (bottom 3 rows) with fixed unsupervised learning data size.

	$\sigma_{ts} = 1$		$\sigma_{ts} = 3$		$\sigma_{ts} = 5$	
Training set	**Joint**	**{C,R,T}**	**Joint**	**{C,R,T}**	**Joint**	**{C,R,T}**
20%	56.2	66.8, 91.0, 92.4	34.0	64.4, 70.5, 74.8	29.0	63.2, 66.1, 69.7
60%	76.5	81.5, 95.2, 98.6	49.0	73.7, 81.7, 81.3	34.6	70.4, 69.0, 71.2
100%	83.5	86.3, 97.2, 99.5	54.7	77.1, 84.9, 83.5	37.8	72.9, 70.5, 73.5
10%	73.9	79.4, 96.8, 96.2	47.5	70.0, 84.9, 79.9	31.7	65.2, 71.9, 67.6
50%	77.8	81.1, 97.6, 98.3	51.5	72.3, 86.5, 82.3	34.7	68.7, 73.6, 68.6
100%	82.3	85.3, 97.9, 98.6	57.4	76.8, 87.1, 85.9	38.1	74.2, 74.0, 69.4

is needed due to the hierarchical formation of long and short memory. This makes it possible to implement a feedforward convolutional network that can be learned with STDP unsupervised training. The effectiveness of H-SNN design is confirmed through mathematical analysis and experiments.

Based on neuron design and network architecture, analysis of neuron dynamics is performed and formula of memory pathway response function is derived. We demonstrate that distinct response functions for different input spike frequency are present in H-SNN. Meanwhile, derivation of cut-off frequency of memory pathway shows that memory of different time scales can be achieved. H-SNN is thus analytically shown to be able to represent distinguishable temporal patterns.

We test H-SNN in computer vision tasks predicting for both single and multiple objectives, and demonstrate the effectiveness of H-SNN on dataset with different visual features and varying motion dynamics. H-SNN is compared with conventional DNN approaches including multiple variants of 3D-CNN, CNN with recurrent connections and SNN trained with back-propagation. Results show two main advantages of H-SNN. First, H-SNN has comparable accuracy with DNN using the same amount of training data. Meanwhile, with the addition of unlabeled data, H-SNN can be further optimized with unsupervised STDP learning and provides higher accuracy than conventional DNN and BP-SNN. The advantage over baselines is most significant when motion dynamic has high deviation from training set. This trend is observed for H-SNN with and without extra unlabeled data to learn. The second advantage is that, with unsupervised learning, H-SNN demonstrates better generalization ability to unknown motion or classes in motion-agnostic and class-agnostic tests. Compared to BP-SNN, H-SNN more effectively learns transformation invariant spatial patterns as well as the general spatiotemporal patterns in the dataset. The combination of long and short term neurons in BP-SNN-LS produces considerable improvement over the original BP-SNN in terms of motion dynamics prediction accuracy. However, the supervised training method of BP-SNN-LS does not have the ability to learn unlabeled data, thus cannot benefit from the same technique used for H-SNN (5xU) and H-SNN (10xU) to further improve SNN performance.

In addition to prediction accuracy, the improved performance of H-SNN is achieved with much lower network complexity than conventional deep networks, and can be implemented in hardware with higher energy-efficiency. In conclusion, H-SNN provides an appealing solution for learning spatiotemporal patterns encountered in computer vision applications that have limited training data and/or constrained computing resources.

DATA AVAILABILITY STATEMENT

Publicly available datasets are used in this study. This data can be found here: https://captain-whu.github.io/DOTA/dataset.html, https://github.com/zalandoresearch/fashion-mnist.

AUTHOR CONTRIBUTIONS

XS developed the main concepts, performed simulation, and wrote the paper. All authors assisted in developing the concept and writing the paper.

FUNDING

This material is based on work sponsored by the Army Research Office and was accomplished under Grant Number W911NF-19-1-0447. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government.

REFERENCES

Amir, A., Taba, B., Berg, D., Melano, T., McKinstry, J., Di Nolfo, C., et al. (2017). "A low power, fully event-based gesture recognition system," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Honolulu, HI), 7243–7252.

Azulay, A., and Weiss, Y. (2018). Why do deep convolutional networks generalize so poorly to small image transformations? *arXiv [Preprint]* arXiv:1805.12177.

Baca, T., Hert, D., Loianno, G., Saska, M., and Kumar, V. (2018). "Model predictive trajectory tracking and collision avoidance for reliable outdoor deployment of unmanned aerial vehicles," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Madrid: IEEE), 6753–6760.

Bell, C. C., Han, V. Z., Sugawara, Y., and Grant, K. (1997). Synaptic plasticity in a cerebellum-like structure depends on temporal order. *Nature* 387, 278–281. doi: 10.1038/387278a0

Bellec, G., Salaj, D., Subramoney, A., Legenstein, R., and Maass, W. (2018). "Long short-term memory and learning-to-learn in networks of spiking neurons," in *Advances in Neural Information Processing Systems* (Montreal, QC), 787–797.

Bi, G., and Poo, M. (2001). Synaptic modification by correlated activity: Hebb's postulate revisited. *Annu. Rev. Neurosci.* 24, 139–166. doi: 10.1146/annurev.neuro.24.1.139

Caporale, N., and Dan, Y. (2008). Spike timing–dependent plasticity: a hebbian learning rule. *Annu. Rev. Neurosci.* 31, 25–46. doi: 10.1146/annurev.neuro.31.060407.125639

Cheng, G., Zhou, P., and Han, J. (2016). Learning rotation-invariant convolutional neural networks for object detection in VHR optical remote sensing images. *IEEE Trans. Geosci. Remote Sens.* 54, 7405–7415. doi: 10.1109/TGRS.2016.2601622

Dan, Y., and Poo, M.-M. (2006). Spike timing-dependent plasticity: from synapse to perception. *Physiol. Rev.* 86, 1033–1048. doi: 10.1152/physrev.00030.2005

DePasquale, B., Churchland, M. M., and Abbott, L. (2016). Using firing-rate dynamics to train recurrent networks of spiking model neurons. *arXiv preprint* arXiv:1601.07620.

Diehl, P. U., and Cook, M. (2015). Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Front. Comput. Neurosci.* 9:99. doi: 10.3389/fncom.2015.00099

Diehl, P. U., Neil, D., Binas, J., Cook, M., Liu, S.-C., and Pfeiffer, M. (2015). "Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing," in *2015 International Joint Conference on Neural Networks (IJCNN)* (Killarney: IEEE), 1–8.

Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., et al. (2015). "Long-term recurrent convolutional networks for visual recognition and description," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Boston, MA), 2625–2634.

Gerstner, W., and Kistler, W. M. (2002a). Mathematical formulations of hebbian learning. *Biol. Cybern.* 87, 404–415. doi: 10.1007/s00422-002-0353-y

Gerstner, W., and Kistler, W. M. (2002b). *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge: Cambridge University Press. doi: 10.1017/CBO9780511815706

Gerstner, W., Ritz, R., and van Hemmen, J. L. (1993). Why spikes? Hebbian learning and retrieval of time-resolved excitation patterns. *Biol. Cybern.* 69, 503–515. doi: 10.1007/BF00199450

Huh, D., and Sejnowski, T. J. (2018). "Gradient descent for spiking neural networks," in *Advances in Neural Information Processing Systems* (Montreal, QC), 1433–1443.

Izhikevich, E. M. (2003). Simple model of spiking neurons. *IEEE Trans. Neural Netw.* 14, 1569–1572. doi: 10.1109/TNN.2003.820440

Kheradpisheh, S. R., Ganjtabesh, M., Thorpe, S. J., and Masquelier, T. (2018). Stdp-based spiking deep convolutional neural networks for object recognition. *Neural Netw.* 99, 56–67. doi: 10.1016/j.neunet.2017.12.005

Köpüklü, O., Kose, N., Gunduz, A., and Rigoll, G. (2019). "Resource efficient 3d convolutional neural networks," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)* (Seoul: IEEE), 1910–1919.

Lansdell, B. J., and Kording, K. P. (2019). Spiking allows neurons to estimate their causal effect. *bioRxiv*. doi: 10.1101/253351

Lee, C., Kosta, A., Zhu, A. Z., Chaney, K., Daniilidis, K., and Roy, K. (2020). Spike-flownet: event-based optical flow estimation with energy-efficient hybrid neural networks. *arXiv preprint* arXiv:2003.06696.

Lee, C., Panda, P., Srinivasan, G., and Roy, K. (2018). Training deep spiking convolutional neural networks with stdp-based unsupervised pre-training followed by supervised fine-tuning. *Front. Neurosci.* 12:435. doi: 10.3389/fnins.2018.00435

Lee, J. H., Delbruck, T., and Pfeiffer, M. (2016). Training deep spiking neural networks using backpropagation. *Front. Neurosci.* 10:508. doi: 10.3389/fnins.2016.00508

Maass, W. (1997). Networks of spiking neurons: the third generation of neural network models. *Neural Netw.* 10, 1659–1671.

Magee, J. C., and Johnston, D. (1997). A synaptically controlled, associative signal for Hebbian plasticity in hippocampal neurons. *Science* 275, 209–213. doi: 10.1126/science.275.5297.209

Marković, I., Chaumette, F., and Petrović, I. (2014). "Moving object detection, tracking and following using an omnidirectional camera on a mobile robot," in *2014 IEEE International Conference on Robotics and Automation (ICRA)* (Hong Kong: IEEE), 5630–5635.

Masquelier, T., and Thorpe, S. J. (2007). Unsupervised learning of visual features through spike timing dependent plasticity. *PLoS Comput. Biol.* 3:e31. doi: 10.1371/journal.pcbi.0030031

Moreno-Bote, R., and Drugowitsch, J. (2015). Causal inference and explaining away in a spiking network. *Sci. Rep.* 5:17531. doi: 10.1038/srep17531

Nicola, W., and Clopath, C. (2017). Supervised learning in spiking neural networks with force training. *Nat. Commun.* 8, 1–15. doi: 10.1038/s41467-017-01827-3

Panda, P., Aketi, S. A., and Roy, K. (2020). Toward scalable, efficient, and accurate deep spiking neural networks with backward residual connections, stochastic softmax, and hybridization. *Front. Neurosci.* 14:653. doi: 10.3389/fnins.2020.00653

Paredes-Vallés, F., Scheper, K. Y. W., and De Croon, G. C. H. E. (2020). Unsupervised learning of a hierarchical spiking neural network for optical flow estimation: From events to global motion perception. *IEEE Trans. Pattern Anal. Mach. Intell.* 42, 2051–2064. doi: 10.1109/TPAMI.2019.2903179

Pfeiffer, M., and Pfeil, T. (2018). Deep learning with spiking neurons: opportunities and challenges. *Front. Neurosci.* 12:774. doi: 10.3389/fnins.2018.00774

Querlioz, D., Bichler, O., Dollfus, P., and Gamrat, C. (2013). Immunity to device variations in a spiking neural network with memristive nanodevices. *IEEE Trans. Nanotechnol.* 12, 288–295. doi: 10.1109/TNANO.2013.2250995

Roy, K., Jaiswal, A., and Panda, P. (2019). Towards spike-based machine intelligence with neuromorphic computing. *Nature* 575, 607–617. doi: 10.1038/s41586-019-1677-2

Rueckauer, B., Lungu, I.-A., Hu, Y., Pfeiffer, M., and Liu, S.-C. (2017). Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Front. Neurosci.* 11:682. doi: 10.3389/fnins.2017.00682

Sengupta, A., Ye, Y., Wang, R., Liu, C., and Roy, K. (2019). Going deeper in spiking neural networks: Vgg and residual architectures. *Front. Neurosci.* 13:95. doi: 10.3389/fnins.2019.00095

She, X., Long, Y., and Mukhopadhyay, S. (2019). Fast and low-precision learning in gpu-accelerated 635 spiking neural network. In 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE) 636 (IEEE), 450–455.

Srinivasan, G., and Roy, K. (2019). Restocnet: residual stochastic binary convolutional spiking neural network for memory-efficient neuromorphic computing. *Front. Neurosci.* 13:189. doi: 10.3389/fnins.2019.00189

Stromatias, E., Soto, M., Serrano-Gotarredona, T., and Linares-Barranco, B. (2017). An event-driven classifier for spiking neural networks fed with synthetic or dynamic vision sensor data. *Front. Neurosci.* 11:350. doi: 10.3389/fnins.2017.00350

Tran, D., Bourdev, L., Fergus, R., Torresani, L., and Paluri, M. (2015). "Learning spatiotemporal features with 3d convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision* (Santiago), 4489–4497.

Weiler, M., Hamprecht, F. A., and Storath, M. (2018). "Learning steerable filters for rotation equivariant cnns," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Salt Lake City, UT), 849–858.

Wu, Y., Deng, L., Li, G., Zhu, J., and Shi, L. (2018). Spatio-temporal backpropagation for training high-performance spiking neural networks. *Front. Neurosci.* 12:331. doi: 10.3389/fnins.2018.00331

Xia, G., Bai, X., Ding, J., Zhu, Z., Belongie, S. J., Luo, J., et al. (2017). DOTA: a large-scale dataset for object detection in aerial images. *CoRR* abs/1711.10398.

Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *arXiv*.

Zhang, R. (2019). "Making convolutional networks shift-invariant again," in *International Conference on Machine Learning* (Long Beach, CA), 7324–7334.

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.