frontiers in Neurorobotics

ORIGINAL RESEARCH
published: 24 March 2022
doi: 10.3389/fnbot.2022.859610

Generative Adversarial Training for Supervised and Semi-supervised Learning

Xianmin Wang[1], *Jing Li*[1,2,3]*, *Qi Liu*[1], *Wenpeng Zhao*[1], *Zuoyong Li*[2] and *Wenhao Wang*[3]

[1] Institute of Artificial Intelligence and Blockchain, Guangzhou University, Guangzhou, China, [2] Fujian Provincial Key Laboratory of Information Processing and Intelligent Control, Minjiang University, Fuzhou, China, [3] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China

Neural networks have played critical roles in many research fields. The recently proposed adversarial training (AT) can improve the generalization ability of neural networks by adding intentional perturbations in the training process, but sometimes still fail to generate worst-case perturbations, thus resulting in limited improvement. Instead of designing a specific smoothness function and seeking an approximate solution used in existing AT methods, we propose a new training methodology, named Generative AT (GAT) in this article, for supervised and semi-supervised learning. The key idea of GAT is to formulate the learning task as a minimax game, in which the perturbation generator aims to yield the worst-case perturbations that maximize the deviation of output distribution, while the target classifier is to minimize the impact of this perturbation and prediction error. To solve this minimax optimization problem, a new adversarial loss function is constructed based on the cross-entropy measure. As a result, the smoothness and confidence of the model are both greatly improved. Moreover, we develop a trajectory-preserving-based alternating update strategy to enable the stable training of GAT. Numerous experiments conducted on benchmark datasets clearly demonstrate that the proposed GAT significantly outperforms the state-of-the-art AT methods in terms of supervised and semi-supervised learning tasks, especially when the number of labeled examples is rather small in semi-supervised learning.

Keywords: neural networks, adversarial training, generative AT, worst-case perturbations, smoothness function, trajectory-preserving-based alternating update strategy

Edited by:
Song Deng,
Nanjing University of Posts and Telecommunications, China

Reviewed by:
Yi He,
Old Dominion University, United States
Lina Yao,
University of New South Wales, Australia

****Correspondence:***
Jing Li
lijing@gzhu.edu.cn

Received: *21 January 2022*
Accepted: *25 February 2022*
Published: *24 March 2022*

Citation:
Wang X, Li J, Liu Q, Zhao W, Li Z and Wang W (2022) Generative Adversarial Training for Supervised and Semi-supervised Learning. Front. Neurorobot. 16:859610. doi: 10.3389/fnbot.2022.859610

1. INTRODUCTION

Neural networks have launched a profound reformation in various fields, such as intelligent driving (Feng et al., 2021), neuro-inspired computing (Zhang et al., 2020; Deng et al., 2021b), smart health (Khan et al., 2021), and human computer interaction (Deng et al., 2021a; Pustejovsky and Krishnaswamy, 2021; Fang et al., 2022). However, in practical classification and regression applications (Wu et al., 2021a), since the number of training examples is finite, the error rate calculated by the training examples may be considerably deviated from the one by test examples. This fact causes the overfitting problem (Wu et al., 2021b), which greatly impacts the generalization performance of neural networks. In order to prevent the neural networks from overfitting, one popular approach is to augment the loss function by introducing a regularization term, which encourages the model to be less dependent on the empirical risk for the finite training examples.

Based on Bayesian theory, this regularization term can be interpreted as a prior distribution reflecting the preconceived notion of the model (Bishop and Nasser, 2006; Wu et al., 2020). Accordingly, the prior distribution of a model is usually assumed to be smooth. That is to say, the outputs of a naturally occurring system tend to be smooth with respect to the spatial or temporal inputs (Wahba, 1990). This assumption indicates that the data points close to each other should be highly likely to infer the same predictions. Unfortunately, recent studies show that most of the neural networks suffer from misclassifying some data points that have only small differences from the correctly classified data points (Goodfellow et al., 2014b; Strauss et al., 2017; Yuan et al., 2019). These misclassified data points are called the adversarial examples, which are crafted by the addition of some imperceptive perturbations to the natural examples in the input space.

To overcome the problem that the neural networks are vulnerable to small but malicious perturbations, adversarial training (AT) is proposed (Goodfellow et al., 2014b; Wang et al., 2019; Cui et al., 2021; Zhang et al., 2022). AT aims to smooth the model outputs by penalizing the deviations caused by the adversarial perturbations. The major challenge of AT is how to accurately estimate such perturbations that alter the output distribution around the input data points. To this end, several perturbation-based methods have been proposed by solving an internal optimization problem at the current status of the model. For instance, random AT (RAT) (Zheng et al., 2016) improves the model smoothness by adding the randomly generated perturbations to the input data. These perturbed data points are encouraged to produce the same prediction given by its corresponding unperturbed versions. Since the perturbations around the input appear in random directions, RAT is referred to as an isotropic smoothing approach. However, it is shown that the isotropic smoothing makes the model particularly sensitive to adversarial examples (Szegedy et al., 2013; Goodfellow et al., 2014b). Based on this consideration, Goodfellow et al. (2014b) proposed a standard AT (SAT). SAT is an anisotropic method that smoothes the output distribution by making the model robust against perturbations in a specific direction. This specific direction in the input space is called the adversarial direction, in which the output of the model is the most sensitive. To identify the perturbations in the adversarial direction, SAT first formulates an objective function based on the differences between the prediction and correct labels and then solves this function with an efficient Frank-Wolfe optimizer. SAT requires the use of labels when calculating the adversarial perturbations. Hence, SAT cannot be applied to the regime of semi-supervised learning. Virtual AT (VAT) (Miyato et al., 2018) extends the notion of SAT in the sense that it defines the adversarial direction without label information, and thus can be applied to both supervised and semi-supervised learning tasks. We observe that in order to generate the adversarial perturbations, the existing AT methods explicitly define a smoothness function to regularize the neural networks. This leads to two limitations. First, it is extremely difficult to find a universal smoothness function due to the various output patterns and distance metrics. Second, there is no analytical solution to such a box-constrained function. Consequently, a numerical method is generally used to seek an approximate solution, which greatly affects the performance of identifying the worst-case adversarial perturbations.

Different from previous methodologies, we propose a novel AT methodology, named generative AT (GAT) in this article, to improve the smoothness of output distribution of neural networks for the supervised and semi-supervised learning tasks. The objective of the proposed GAT is to train the target classifier such that it not only achieve the minimum prediction error but also has the best robustness against the adversarial perturbations. To this aim, we formalize the regularizing process as a minimax game. To be specific, we exploit the cross entropy method to construct a new *adversarial loss* function. Moreover, we develop an effective alternating update strategy to optimize the challenging non-convex problems. The experimental results tested on benchmark datasets show that the proposed GAT obtains the empirical equilibrium point and state-of-the-art performance.

The main contributions of this article are summarized as follows:

- We formulate the regularizing for the learning task as a minimax game according to the outputs of the target classifier from the natural example and its adversarial version derived by a perturbation generator. As the game approaches the empirical equilibrium, the target classifier achieves the best performance.
- A new *adversarial loss* function is constructed based on the *cross entropy* method, which not only accurately reflects the deviation caused by the perturbation but also efficiently assesses the confidence of network output.
- An effective alternating update strategy based on trajectory preserving is proposed to control the minimax optimization training to be stable.
- The proposed GAT regularizes the model without label information, hence it can be applied to the supervised and semi-supervised learning tasks.

It is worth emphasizing that our method differs from any one of the generative-model-based AT methods (Kingma et al., 2014; Maaløe et al., 2016; Salimans et al., 2016; Dai et al., 2017). This family of methods is considered to be an improvement of Generative Adversarial Network (GAN), in the sense that the target classifier in their frameworks is the extension of the GAN's discriminator serving for distinguishing the natural and generated examples. For our method, the discriminator is not the target classifier; instead, it is manually designed according to the outputs of the target classifier over the natural example and its adversarial version.

2. PROBLEM SETTING AND RELATED WORKS

Without loss of generality, we consider the classification tasks in a semi-supervised setting. Let $x \in \mathcal{X} = R^I$ be the input vector with I-dimension and $y \in \mathcal{Y} = Z^K$ be the one-hot vector of labels with K categories. $\mathcal{D}^l = \left\{x^l_{(i)}, y^l_{(i)} | i = 1, ..., N^l\right\}$ and $\mathcal{D}^{ul} =$

$\left\{x^{ul}_{(j)}|j=1,...,N^{ul}\right\}$ denote the labeled and unlabeled dataset, where N^l and N^{ul} are the number of labeled and unlabeled examples. AT regularizes the neural network such that both the natural and perturbed examples output the intended predictions. That is, we aim to learn a mapping $\mathbb{F}: X \to [0,1]^K$ parameterized with $\theta \in \Theta$ *via* solving the following optimization problem

$$\min\left\{\mathcal{L}_S\left(\mathcal{D}^l,\theta\right)+\lambda\cdot\mathcal{L}_R\left(D^l,D^{ul},\theta\right)\right\}. \tag{1}$$

The symbol $\mathcal{L}_S$ in Equation 1 represents the *supervised loss* over the labeled dataset, which can be expanded as

$$\mathcal{L}_S=\mathbb{E}_{(x^l,y^l)\sim\mathcal{D}^l}\Gamma\left(y^l,\mathbf{F}_\theta\left(x^l\right)\right), \tag{2}$$

where $\mathbf{F}_\theta\left(x^l\right)$ denotes the output distribution vector of the neural network on the input x^l given the model parameter θ, y^l is the one-hot vector of the true label for x^l. The operator $\Gamma(\cdot,\cdot)$ denotes the distance measure used to evaluate the similarity of two distributions. A common choice of Γ for the supervised cost $\mathcal{L}_S$ is the measure of *cross entropy*. $\mathcal{L}_R$ is the *adversarial loss*, which is served as a regularization term for promoting the smoothness of the model. The *adversarial loss* plays an important role in enhancing the generalization performance while the number of labeled examples is small relative to the number of the whole training examples (i.e., $N^l << N^{ul}+N^l$). λ is a non-negative value that controls the relative balance between the *supervised loss* and the *adversarial loss*.

Many approaches are presented to construct $\mathcal{L}_R$ based on the smoothness assumption, which can be generally represented in a framework as

$$\mathcal{L}_R=\mathbb{E}_{x\sim\mathcal{D}}\Gamma\left(\mathbf{F}_\theta\left(x;\xi\right),\tilde{\mathbf{F}}_{\theta'}\left(x;\xi'\right)\right), \tag{3}$$

where x is sampled from the dataset $\mathcal{D}$ which consists of both labeled and unlabel examples. $\Gamma\left(\mathbf{F}_\theta\left(x;\xi\right),\tilde{\mathbf{F}}_{\theta'}\left(x;\xi'\right)\right)$ is termed as the smoothness function, which is comprised of a teacher model $\mathbf{F}_\theta(x;\xi)$ and a student model $\tilde{\mathbf{F}}_{\theta'}\left(x;\xi'\right)$. The teacher model is parameterized with parameter θ and perturbation ξ, while the student model is parameterized with parameter θ' and perturbation ξ'. The goal of $\mathcal{L}_R$ is to improve the model's smoothness by forcing the student model to follow the teacher model. That is to say, the output distributions yielded by $\tilde{\mathbf{F}}$ is supported to be consistent with the outputs derived by $\mathbf{F}$. To this end, the teacher model, student model, and similarity measure are required to be carefully crafted for formulating an appropriate smoothness function against the perturbation of the input and the variance of the parameters. Based on the implementations of this smoothness function, some typical AT approaches can be explicitly defined.

Random Adversarial Training: In RAT, random noises are introduced in the student model instead of the teacher model, and the parameters of the student model are shared with the teacher model. Moreover, L_2 distance is used to measure the similarity of the output distributions derived by $\tilde{\mathbf{F}}$ and $\mathbf{F}$ on the whole training examples. That is, $\theta'=\theta$, $\xi'\sim\mathcal{N}(0,1)$, $\xi=0$, and $\mathcal{D}=\mathcal{D}^{ul}\bigcup\mathcal{D}^l$ for Equation 3.

Adversarial Training With Π-Model: In contrast to RAT, Π-model introduces random noises to both the teacher model and student model, i.e., $\xi',\xi\sim\mathcal{N}(0,1)$. The reason for this is based on the assumption that predictions yielded by natural example may itself be an outlier, hence it is reasonable to make two noisy predictions learn from each other. In this case, optimizing the smoothness function for Π-model is equivalent to minimizing the prediction variance of the classifier (Luo et al., 2018).

Standard Adversarial Training: Instead of adding random noises to the teacher/student model, the perturbation adopted in SAT is some imperceptible noise that is carefully designed to fool the neural network. The *adversarial loss* $\mathcal{L}_R^{sat}$ of SAT can be written as

$$\begin{aligned}\mathcal{L}_R^{sat}&=\mathbb{E}_{(x^l,y^l)\sim\mathcal{D}^l}\mathrm{KL}\left(y^l||\tilde{\mathbf{F}}_\theta\left(x^l;\xi_{adv}\right)\right)\\ s.t.\quad \xi_{adv}&=\underset{\xi;\|\xi\|\le\varepsilon}{\arg\max}\,\mathrm{KL}\left(y^l||\tilde{\mathbf{F}}_\theta\left(x^l;\xi\right)\right),\end{aligned} \tag{4}$$

where the operator $\mathrm{KL}(\cdot||\cdot)$ denotes the similarity measure of *Kullback-Leibler (K-L) divergence*. ξ_{adv} denotes adversarial perturbation which is added into x^l to make the output distribution of the student model most greatly deviate y^l. ε is a prior constant that controls the perturbation strength. Note that the teacher model, in this case, is degenerated into the one-hot vector of the true label. Generally, we cannot obtain the exact adversarial direction of ξ_{adv} in a closed form. Hence, a linear approximation of this objective function is applied to approximate the adversarial perturbation. For ℓ_∞ norm, the adversarial perturbation ξ_{adv} can be efficiently approximated by using the famous fast gradient sign method (FGSM) (Madry et al., 2017). That is,

$$\xi_{adv}\approx\varepsilon\cdot\mathrm{sign}\left(\nabla_{x^l}\mathrm{KL}\left(y^l||\tilde{\mathbf{F}}_\theta\left(x^l;\xi\right)\right)\right). \tag{5}$$

Some alternative invariants such as the iterative gradient sign method (IGSM) (Tramèr et al., 2017) and the momentum IGSM (M-IGSM) (Dong et al., 2018) are available to solve the objective function. By adding adversarial perturbations to the student model, SAT obtains better generalization performance than RAT and Π-model. Unfortunately, SAT can only be applied in supervised learning tasks since it has to use the labeled examples to compute the *adversarial loss*.

Virtual Adversarial Training: Different from SAT, the key idea of VAT is to define the *adversarial loss* based on the output distribution inferred on the unlabeled examples. In this regard, the *adversarial loss* $\mathcal{L}_R^{vat}$ of VAT can be written as

$$\begin{aligned}\mathcal{L}_R^{vat}&=\mathbb{E}_{x\sim\mathcal{D}^l\cup\mathcal{D}^{ul}}\mathrm{KL}\left(\mathbf{F}_\theta(x)||\tilde{\mathbf{F}}_\theta\left(x;\xi_{adv}\right)\right)\\ s.t.\quad \xi_{adv}&=\underset{\xi;\|\xi\|\le\varepsilon}{\arg\max}\,\mathrm{KL}\left(\mathbf{F}_\theta(x)||\tilde{\mathbf{F}}_\theta\left(x;\xi\right)\right).\end{aligned} \tag{6}$$

To obtain the adversarial perturbation ε_{adv}, Miyato et al. (2018) proposed to approximate the objective function with a second-order Taylor's expansion at $\varepsilon=0$. That is,

$$\xi_{adv}\approx\underset{\xi;\|\xi\|\le\varepsilon}{\arg\max}\,\frac{1}{2}\xi^T H(x,\theta)\,\xi, \tag{7}$$

where H is a Hessian matrix which is defined by $H(x,\theta) = \nabla\nabla_{\xi}\mathrm{KL}\left(\mathbf{F}_{\theta}(x)\,||\,\tilde{\mathbf{F}}_{\theta}(x;\xi)\right)$. This binomial optimization is an eigenvalue problem that can be solved using power iteration algorithm. Since VAT acquires the adversarial perturbation in the absence of label information, this method is applicable to both supervised and semi-supervised learning.

3. THE PROPOSED METHOD

Adversarial training methods regularize the neural network *via* forcing the output distribution to be robust against adversarial examples. To obtain intentional perturbations, the existing AT methods require to explicitly define a smoothness function to compute the perturbations. Due to the non-convex characteristic of the smoothness function, the existing AT methods usually fail to generate worst-case perturbation by approximation analysis. To tackle this problem, we propose a novel AT framework termed GAT for improving the smoothness of the neural network, where the worst-case perturbation of the input is generated by a generator. In the following sections, we construct our framework by answering two central questions: (1) how to formulate the loss function with the perturbation generator and target classifier and (2) how to effectively optimize this loss function during the training process.

3.1. GAT Loss Based on Minimax Game

In our framework, two neural networks are considered, i.e., the target classifier $\mathbf{T}_{\theta}(x)$ parameterized with θ and the perturbation generator $\mathbf{G}_{\varphi}(x)$ parameterized with φ. In our framework, the target classifier is the optimization objective that will be required eventually. The perturbation generator is constructed by an auto-encoder-like neural network. Specifically, the perturbation generator can be defined as a mapping $\mathbf{G}_{\varphi}:\mathcal{X} \rightarrow \mathcal{X}$, which takes a natural example in $\mathcal{X}$ and then transforms it into an imperceptible perturbation in the same space $\mathcal{X}$. For ℓ_{∞} norm, such constraints can be represented as

$$\forall x, \left\|\mathbf{G}_{\varphi}(x)\right\|_{\infty} \leq \varepsilon, \tag{8}$$

where ε is the perturbation bounds that controls the adversarial strength. To implement the constraints indicated by Equation 8, the activation function of the last layer in $\mathbf{G}_{\varphi}$ is particularly defined as $\varepsilon \cdot tanh(\cdot)$. Then, the generated perturbation is added into the corresponding natural example to composite an adversarial example.

The goal of $\mathbf{G}_{\varphi}$ is to find a perturbation that most deviates the current inferred output of the target classifier from the status quo, while $\mathbf{T}_{\theta}(x)$ is to minimize the prediction error for the natural example as well as the deviation caused by such perturbation. This problem can be formulated as a minimax game and the loss function of which can be formulated as

$$\begin{aligned}\min_{\theta}\max_{\varphi}\quad & \mathbb{E}_{(x^l,y^l)\sim\mathcal{D}^l}\Gamma_S\left(y^l, \mathbf{T}_{\theta}\left(x^l\right)\right) \\ & + \lambda\cdot\mathbb{E}_{x\sim\mathcal{D}^l\cup\mathcal{D}^{ul}}\Gamma_R\left(\mathbf{T}_{\theta}(x), \mathbf{T}_{\theta}\left(\mathbf{G}_{\varphi}(x)+x\right)\right).\end{aligned} \tag{9}$$

Equation 9 is referred to as the GAT loss, which is comprised of a *supervised loss* $\mathcal{L}_S$ and an *adversarial loss* $\mathcal{L}_R$. $\mathcal{L}_S$ is determined by labeled examples, while $\mathcal{L}_R$ is independent of the labels and served as a regularization term smoothing the model. The parameter λ controls the balance of $\mathcal{L}_S$ and $\mathcal{L}_R$. For the maximization and minimization loop of the minimax game, φ and θ are the parameters required to be optimized. Since $\mathcal{L}_R$ is defined over the whole data set, our method is applicable to semi-supervised learning. Note that for the *adversarial loss*, the target classifier $\mathbf{T}_{\theta}(x)$ is considered as the teacher model, while the compound function of $\mathbf{T}_{\theta}\left(\mathbf{G}_{\varphi}(x)+x\right)$ is served as the student model.

In addition, the operator $\Gamma_S(\cdot,\cdot)$ and $\Gamma_R(\cdot,\cdot)$ are the similarity measures for $\mathcal{L}_S$ and $\mathcal{L}_R$, respectively. Here, Γ_R is crucial for the construction of *adversarial loss*. Instead of using *K-L divergence* to define the *adversarial loss* as VAT/SAT does, we exploit *cross entropy* measures to formulate the *adversarial loss* function. There are two beneficial effects for this implementation. First, *cross entropy* overcomes the problem of zero avoiding, an inward nature for the *K-L divergence*(Bishop and Nasser, 2006). Second, since *cross entropy* can be represented as the sum of *K-L divergence* and *information entropy*, $\mathcal{L}_R$ not only implies the deviation of the output distributions, but also signifies the confidence of the prediction of the target classifier. In particular, by substituting Γ_R with *cross entropy* in Equation 9, $\mathcal{L}_R$ in GAT loss can be rewritten as

$$\begin{aligned}&\mathrm{CE}\left(\mathbf{T}_{\theta}(x), \mathbf{T}_{\theta}\left(\mathbf{G}_{\varphi}(x)+x\right)\right) \\ &\quad = \mathrm{KL}\left(\mathbf{T}_{\theta}(x)\,||\,\mathbf{T}_{\theta}\left(\mathbf{G}_{\varphi}(x)+x\right)\right) + \mathrm{H}\left(\mathbf{T}_{\theta}(x)\right),\end{aligned} \tag{10}$$

where the operator $\mathrm{CE}(\cdot,\cdot)$ and $\mathrm{H}(\cdot)$ denote *cross entropy* and *information entropy*. In Equation 10, $\mathrm{KL}\left(\mathbf{T}_{\theta}(x)\,||\,\mathbf{T}_{\theta}\left(\mathbf{G}_{\varphi}(x)+x\right)\right)$ is termed as smoothness term, which reflects the deviation of the output distributions, while $\mathrm{H}\left(\mathbf{T}_{\theta}(x)\right)$ is termed as confidence term, which indicates the confidence of the output distribution. Moreover, we observed that the confidence term is independent with parameter φ. Hence, for the maximization loop of the minimax game, maximizing $\mathcal{L}_R$ requires to maximize the smoothness term only. Whereas, for the minimization loop, minimizing $\mathcal{L}_R$ requires to minimize both the smoothness term and confidence term. Note that minimizing the confidence term facilitates boosting of the prediction confidence of the neural network. Thus, our *adversarial loss* has the effect of entropy minimization proposed in Grandvalet and Bengio (2004) and Sajjadi et al. (2016).

3.2. Alternating Update Process Based on Trajectory Preserving

Figure 1 depicts the framework of GAT, in which two neural networks are required to be optimized, i.e., the target classifier $\mathcal{T}$ and the perturbation generator $\mathcal{G}$. $\mathcal{G}$ takes natural example x from the full dataset comprising of both the labeled and unlabeled examples and generates a perturbation $\mathbf{G}_{\varphi}(x)$. Then, $\mathbf{G}_{\varphi}(x)$ is appended into x to composite an adversarial example. Both the adversarial example and its corresponding natural example are fed into $\mathcal{T}$ for constructing the *adversarial loss* $\mathcal{L}_R$. Meanwhile, labeled example x^l sampled from the labeled dataset is input to $\mathcal{T}$ for formulating the *supervised loss* $\mathcal{L}_S$.

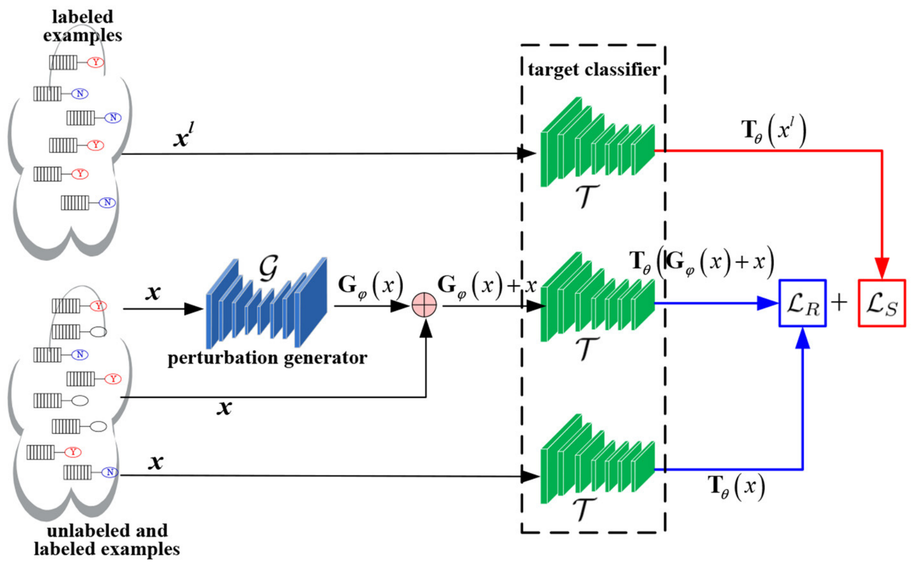

FIGURE 1 | The overall framework of Generative AT (GAT).

The objective of our framework is to find stable θ and φ such that $\mathcal{G}$ maximizes the GAT loss for the given fixed θ, while $\mathcal{T}$ minimizes the GAT loss for the given fixed φ. Due to the non-linear constraint of the perturbation and non-convex properties of the loss function, this optimization problem is very challenging. Inspired by the training pattern of GAN (Goodfellow et al., 2014a) and some common tricks in reinforcement learning (Mnih et al., 2015), we propose to optimize the GAT loss by an alternative updating procedure and stabilize this procedure based on trajectory preserving.

First, we decompose the minimax optimization problem into the inner loop and outer loop. The inner loop aims to derive an optimal φ for maximizing the loss, while the outer loop aims to obtain an optimal θ for minimizing the loss. Due to the fact that the parameter φ in the inner loop is independent of the *supervised loss* during the maximizing procedure, then the optimal φ of $\mathcal{G}$ under the fixed θ can be written as Equation 11. Meanwhile, the optimal θ of $\mathcal{T}$ under the given fixed φ can be represented as Equation 12.

$$\varphi = \arg\max_{\varphi} \mathbb{E}_{x\sim\mathcal{D}^l\cup\mathcal{D}^{ul}} \mathrm{CE}\left(\mathbf{T}_\theta(x), \mathbf{T}_\theta\left(x+\mathbf{G}_\varphi(x)\right)\right), \quad (11)$$

$$\begin{aligned}\theta = \arg\min_{\theta} & \mathbb{E}_{(x^l,y^l)\sim\mathcal{D}^l} \mathrm{CE}\left(y^l, \mathbf{T}_\theta\left(x^l\right)\right) + \\ & \lambda\cdot\mathbb{E}_{x\sim\mathcal{D}^l\cup\mathcal{D}^{ul}} \mathrm{CE}\left(\mathbf{T}_\theta(x), \mathbf{T}_\theta\left(x+\mathbf{G}_\varphi(x)\right)\right).\end{aligned} \quad (12)$$

Second, since the perturbation generator and the target classifier are assumed to be neural networks, the parameters θ and φ in Equations 11 and 12 can be calculated by stochastic-gradient-based methods (Liu et al., 2021; Jin et al., 2022). A traditional solution to this minimax problem is to alternatively update φ by gradient ascent over the full dataset and update θ by gradient descent over the labeled dataset. However, since the number of labeled training examples is small, both φ and θ are not easy to converge in practice. We develop a trajectory preserving strategy to tackle this problem. In our method, for each epoch of alternating, we update φ using gradient ascent and record the update trajectories of φ. Then, based on these trajectories, we retrieve the intermediate parameter φ' by executing a pseudo-update procedure for φ. Finally, we update θ by gradient descent under the given φ'.

The implementation details of the proposed trajectory preserving training procedure are illustrated in **Algorithm 1**,

Algorithm 1: Trajectory preserving training process.

```
Initialize randomly $\theta$
for $epoch = 1 : E$ do
    Create empty list $L$
    Initialize randomly $\varphi_0$
    for $t = 0 : T$ do
        Sample batch $\{x_i^{(t)}\}$ of size $M$ from $\mathcal{D}^{ul} \cup \mathcal{D}^l$
        Store $(\{x_i^{(t)}\}, \varphi^{(t)})$ into the list $L$
        Update $\varphi^{(t+1)}$ by gradient ascent (Equation 13)
    end
    for $t = 0 : T$ do
        Retrieve $(\{x_i^{(t)}\}, \varphi^{(t)})$ from the list $L$
        Pseudo-update $\varphi'$ by gradient ascent (Equation 14)
        Sample batch $\{(x_j^l, y_j^l)\}$ of size $N$ from $\mathcal{D}^l$
        Update $\theta$ by gradient descent (Equation 15)
    end
end
return $\theta$
```

where E is the number of training epochs, T is the maximum iterations in each epochs. Equations 13 and 14 represent the updating and pseudo-updating for φ by gradient ascent. Equation 15 describes the updating process for θ by gradient descent. α_g and α_t are the learning rate for the perturbation generator and target classifier, respectively.

$$\varphi^{(t+1)} = \varphi^{(t)} + \alpha_g \nabla_{\varphi^{(t)}} \frac{1}{M} \sum_{i=1}^{M} \mathrm{CE}\left(\mathbf{T}_\theta\left(x_i^{(t)}\right),\ \mathbf{T}_\theta\left(x_i^{(t)} + \mathbf{G}_{\varphi^{(t)}}\left(x_i^{(t)}\right)\right)\right) \tag{13}$$

$$\varphi' = \varphi^{(t)} + \alpha_g \nabla_{\varphi^{(t)}} \frac{1}{M} \sum_{i=1}^{M} \mathrm{CE}\left(\mathbf{T}_\theta\left(x_i^{(t)}\right),\ \mathbf{T}_\theta\left(x_i^{(t)} + \mathbf{G}_{\varphi^{(t)}}\left(x_i^{(t)}\right)\right)\right) \tag{14}$$

$$\begin{aligned}\theta =& \theta - \alpha_t \nabla_\theta \left\{ \frac{1}{N} \sum_{j=1}^{N} \mathrm{CE}\left(y_j^l, \mathbf{T}_\theta\left(x_j^l\right)\right) + \frac{\lambda}{M} \sum_{i=1}^{M} \mathrm{CE}\left(\mathbf{T}_\theta\left(x_i^{(t)}\right),\right.\right. \\ & \left.\left.\mathbf{T}_\theta\left(x_i^{(t)} + \mathbf{G}_{\varphi'}\left(x_i^{(t)}\right)\right)\right)\right\}.\end{aligned} \tag{15}$$

4. EXPERIMENTS

To validate the performance of our method on supervised and semi-supervised task, we carried out experiments on synthetic datasets and practical benchmarks by comparing with various strong competitors.

4.1. Supervised Learning on a Synthetic Dataset

This section tests the supervised learning performance of our method for binary classification problems using two well-known synthetic datasets, i.e., the "Moons" dataset (termed as M-dataset) and the "Circles" dataset (termed as C-dataset). The data points in the two datasets are sampled uniformly from two trajectories over the space of R^2 and embedded linearly into 100-dimension vector space. Each dataset contains 16 training data points and 1,000 testing points. **Figures 4**, **5** provide the visualizations for M-dataset and N-dataset, where the red circles and blue triangles separately stand for the training examples with labels 1 and 0. The target classifier used in this experiment is a neural network with one hidden layer comprised of 100 hidden units, where ReLU and softmax activation function are applied to the hidden units and output units. We compare our method with some popular AT methods, such as SAT (Goodfellow et al., 2014b), RAT (Zheng et al., 2016), and VAT (Miyato et al., 2018). These AT methods and the proposed GAT are conducted under the setting of $\lambda = 1$ and $\epsilon = 0.2$. Particularly, the perturbation generator in our method has three hidden layers with the unit number 128, 64, and 128, respectively.

Since the number of the training examples is extremely small compared to the input dimension, the target classifier for binary classification is very vulnerable to the problem of overfitting. **Figures 2A,B** depict the transitions of the accuracy rates for the target classifier with the GAT regularization and without this regularization (termed as Plain NN). It can be observed that the training accuracy of Plain NN and GAT achieved 100% for the two datasets. Nevertheless, the test accuracy rate of GAT is noticeably higher than that of Plain NN. Although our method suffers from some fluctuations with the accuracy rate at the initial stage of the training process, the test accuracy rate of our method finally achieves a stable value after a few iterations, thanks to the trajectory preserving training strategy. **Figure 3** visualizes the output distributions of the trained target classifier on the M-dataset and C-dataset with our method and Plain NN. We can observe that compared to plain NN, GAT provides more flat regions for the landscape of the output distribution. This phenomenon indicates that our method is conducive to the smoothness of the model in the sense that flat surfaces of the landscape imply small deviations of the output.

Moreover, we plot the contours of the target classifier's predictions for label 1 on the two synthetic datasets by various regularization methods. As shown in **Figures 4**, **5**, the black line in each plot stands for the contour of value 0.5, which is usually used as the decision boundary for the binary classification tasks. From these figures, we can see that the L_2 regularization method fails to acquire correct decision boundary on both the M-dataset and C-dataset, hence, many false predictions are produced by this method. RAT obtains convincing decision boundary for M-dataset, but it generates an unreasonable decision boundary for C-dataset. Among these methods, only SAT, VAT, and

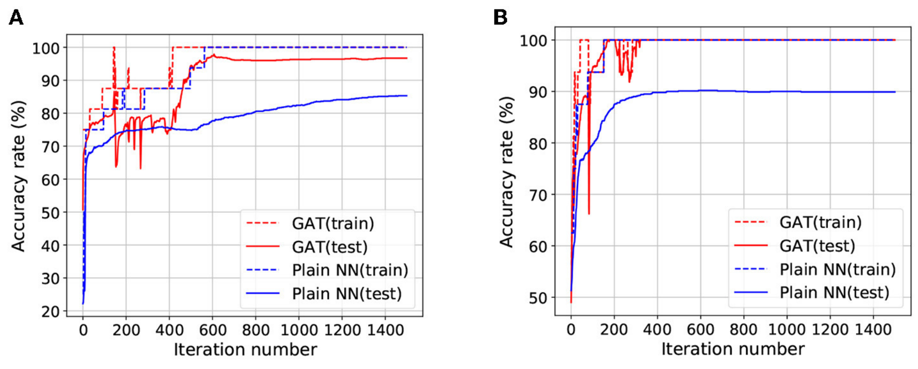

FIGURE 2 | The transition curves of accuracy rates by Plain NN and the proposed GAT on M-dataset and C-dataset. **(A)** Plots the results for M-dataset, **(B)** plots the results for C-dataset.

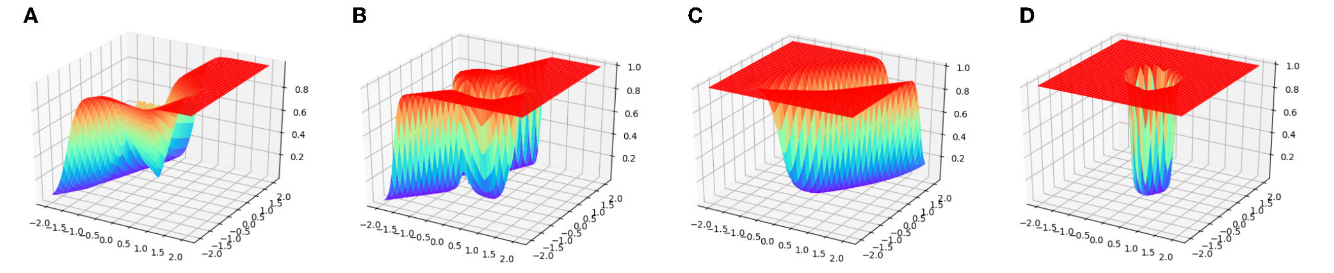

FIGURE 3 | The visualization of model distributions of GAT and Plain NN on the synthetic datasets. **(A,B)** Show the distribution surface on *M*-dataset, **(C,D)** show the distribution surface on *C*-dataset, where flat surface regions implicate small output deviations.

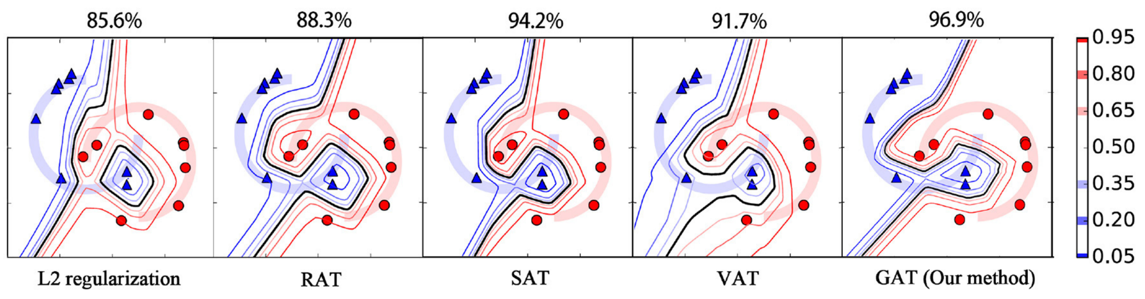

FIGURE 4 | The contour of output confidences for label 1 on *M*-dataset with various regularization methods. The red circles and blue triangles represent the data points with labels 1 and 0, respectively. The decision boundaries with different confidences are plotted with different colored contours. Note that the black line represents the contour of probability value 0.5, which is usually served as the decision boundary for the binary classification task. The accuracy rate of each method for the test examples is displayed above the panel.

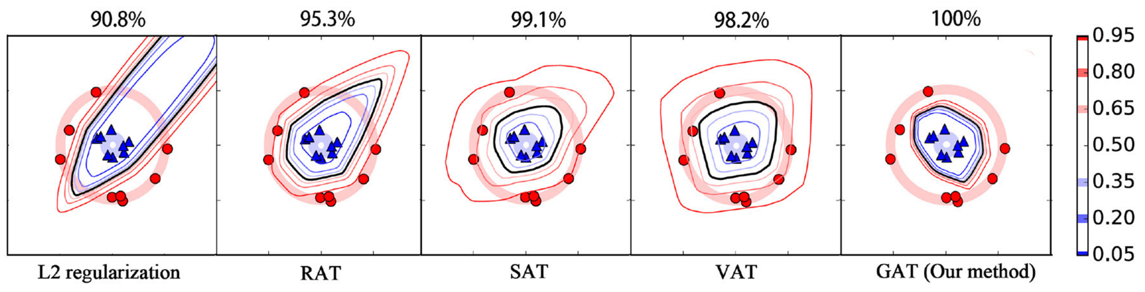

FIGURE 5 | The contour of output confidence for label 1 on *C*-dataset with various regularization methods. The detailed illustrations for this figure can be referred to the caption of **Figure 4**.

our method yield applicable decision boundary for both the *M*-dataset and *C*-dataset, because these methods employ an anisotropic way to smooth the classifier. Compared to RAT and VAT, the decision boundaries of our method for different contour values are more compact. This phenomenon illustrates that our method can provide more confidence predictions for the new instances, thanks to the *cross entropy* measure for the adversarial loss. Our method also achieves the highest test accuracy rate against its competitors on both the *M*-dataset and *C*-dataset.

4.2. Supervised Learning on the Benchmark Dataset

In this section, we evaluate the performance of our methods on the MNIST dataset for a supervised learning scenario. The origin 60,000 training examples are split into 50,000 training examples and 10,000 test examples. The target classifier is made up of four hidden dense layers, whose unit numbers are 1200, 600, 300, and 150, respectively. The input dimension of the target classifier is 784 and the output dimension is 10. For each method, we use the setting of hyper-parameters that exhibits the best performance on the test dataset to train the neural network and record their test errors. The perturbation generator in our method is comprised of hidden layers whose unit numbers are 1200, 600, 300, and 600, respectively. The control parameters of the methods by our implementations are set $\lambda = 1$ and $\epsilon = 0.2$. We compare our method with some typical AT methods on the MNIST dataset for supervised learning task. To verify the capability of the trajectory preserving strategy, we also conducted an ablation experiment for GAT-woTP, a method using the proposed GAT framework but Without Trajectory Preserving strategy during the training. The test error rates of these methods are reported in **Table 1**. The experimental results demonstrate that our method surpasses the previous state-of-the-art AT methods by a large margin. Moreover, our method also outperforms advanced generation-based algorithms such as Ladder network and CatGAN. Besides, note that the error rate obtained by our method is much lower than that acquired by GAT-woTP. This is because the trajectory preserving strategy is benefit to ensure the stability of the training process. Without this strategy, GAT is usually difficult to achieve a favorable convergent point during the training.

4.3. Semi-supervised Learning on Benchmark Dataset

This section validates the effectiveness of our method for semi-supervised learning tasks on three popular benchmarks of MNIST, SVHN, and CIFAR-10. According to the experimental setups in Miyato et al. (2018), we take a test dataset with fixed size 1,000 from the training examples and train the classifier under four sizes of the labeled dataset, i.e., $N_l = \{100, 600, 1000, 3000\}$, where N_l is size of the dataset. The rest instances of the training examples are served as unlabeled examples. Then, we record the test errors under different values of N_l. For our method, we use a mini-batch of size 64 to calculate the *supervised loss* in Equation 11 and a mini-batch of size 256 to calculate the *adversarial loss* in Equation 12. The control parameters of the methods by our implementations are set at $\lambda = 1$ and $\epsilon = 0.2$. To test the performance of the trajectory preserving strategy for semi-supervised learning, we make several ablation experiments for GAT-woTP which is described in Section 4.2. For the reason that SAT can only be applied to supervised learning task, the results of SAT have not been reported in these experiments.

TABLE 1 | Test error rates of various regularization methods for supervised learning task on MNIST dataset.

Method	Test error rate (%)
SVM (gaussian kernel)	1.40
Dropout	1.05
Maxout networks	0.94
DBM	0.79
Ladder network[†]	0.57
Conv-CatGAN[†]	0.48
Plain NN (Baseline)	1.15
RAT	0.85
SAT (L_∞)	0.78
VAT	0.66
GAT-woTP	0.65
GAT (Our method)	0.45

The upper panel refers to the experimental results reported in prior work, the error rates in the bottom panel are derived by our implementations. [†]Represents the generation-based methods.

TABLE 2 | Test error rates of semi-supervised learning methods on MNIST datasets.

Method	Test error rate (%)			
	$N_l = 100$	$N_l = 600$	$N_l = 1,000$	$N_l = 3,000$
SVM	23.44	8.85	7.77	4.21
EmbedNN	16.9	5.97	5.73	3.59
PEA	10.79	2.44	2.23	1.91
Conv-CatGAN[†]	1.93 (±0.01)	1.86 (±0.11)	1.73 (±0.18)	1.67 (±0.12)
Ladder networks[†]	1.06 (±0.37)	0.93 (±0.07)	0.84 (±0.08)	0.79 (±0.09)
Auxiliary DGM[†]	0.96 (±0.02)	0.90 (±0.05)	0.86 (±0.13)	0.78 (±0.05)
RAT	6.62 (±1.02)	3.75 (±0.14)	1.61 (±0.09)	1.51 (±0.08)
VAT	2.38 (±0.11)	1.38 (±0.08)	1.35 (±0.12)	1.28 (±0.07)
GAT-woTP	1.97 (±0.87)	1.66 (±0.85)	1.58 (±0.96)	1.32 (±0.65)
GAT (Our method)	0.90 (±0.11)	0.85 (±0.09)	0.83 (±0.17)	0.75 (±0.08)

N_l denotes the number of labeled examples for the training dataset.
The results in the upper panel are referred to the reports in prior work, the error rates in the bottom panel are derived by our implementations. [†]Represents the generation-based methods.

TABLE 3 | Test error rates (%) of semi-supervised learning methods on SVHN and CIFAR-10 datasets.

Method	SVHN	CIFAR-10
	$N_l = 1,000$	$N_l = 4,000$
Π-model	5.43 (±0.25)	16.55 (±0.29)
Mean teacher	5.21 (±0.21)	17.74 (±0.30)
ALI	7.41 (±0.65)	17.99 (±1.62)
Ban GAN[†]	4.25 (±0.03)	14.41 (±0.30)
Tripple GAN[†]	5.77 (±0.17)	16.99 (±0.36)
Improved GAN[†]	4.39 (±1.20)	16.20 (±1.60)
TNAR-LGAN (Small)[†]	4.25 (±0.09)	12.97 (±0.31)
TNAR-LGAN (Large)[†]	4.03 (±0.13)	12.76 (±0.04)
RAT (Small)	8.42 (±0.22)	18.58 (±0.26)
RAT (Large)	8.36 (±0.22)	18.23 (±0.16)
VAT (Small)	6.83 (±0.24)	14.87 (±0.13)
VAT (Large)	5.77 (±0.32)	14.18 (±0.38)
GAT-woTP (Small)	6.53 (±0.95)	14.36 (±1.03)
GAT-woTP (Large)	5.26 (±0.92)	14.02 (±0.88)
GAT (Our method, Small)	4.27 (±0.14)	12.96 (±0.15)
GAT (Our method, Large)	4.01 (±0.11)	12.81 (±0.13)

N_l represents the number of labeled examples in the training dataset. The results in the upper panel are referred to the reports in prior work, the results in the bottom panel are derived from our implementations. [†]Stands for the generation-based methods.

For the MNIST dataset, the structures of the target classifier and perturbation generator are identical to the structures employed in Section 4.2. **Table 2** lists the test error rates of the comparing semi-supervised learning methods for different values of N_l on MNIST. The experimental results show that our method achieves the lowest error rates among all the methods for different numbers of labeled examples. Moreover, our method significantly outperforms the state-of-the-art AT methods when the number of labeled examples is small. For the experiments on SVHN and CIFAR-10, two type of convolution neural networks (CNNs), named "Small" (Salimans et al., 2016) and "Large" (Laine and Aila, 2018), are employed as the target classifiers. More details about the settings and structures of the two CNNs can be referred to (Miyato et al., 2018). The structure of the perturbation generator in this experiment is the same as the one applied in the experiment for the MNIST dataset. The performance of various comparing methods for SVHN and CIFAR-10 is reported in **Table 3**. From the table, we can find that GAT obtains the best generalization capability for the SVHN dataset and achieves comparable performance to the state-of-the-art generation-based method such as TNAR-VAE for the CIFAR-10 dataset. In addition, GAT reaches lower error rates compared to GAT-woTP for all the three benchmarks, which verifies the favorable performance of the trajectory preserving strategy for stabilizing the training for our proposal.

5. CONCLUSION

In this article, a novel GAT framework has been proposed to improve the generalization performance of neural networks for both the supervised and semi-supervised learning tasks. In the proposed framework, the target classifier is regularized by letting the perturbation generator watch and move against the target classifier in a minimax game. We exploit the *cross entropy* to evaluate the output deviation for the regularization term such that the prediction of the target classifier can be reinforced. Furthermore, an effective alternating update method is developed to stably train the target classifier and perturbation generator. Numerous experiments are conducted on synthetic and real datasets and their results demonstrate the effectiveness of our proposal.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

AUTHOR CONTRIBUTIONS

XW contributed to the conception of the study, performed the data analyses, and wrote the manuscript. JL contributed significantly to analysis and manuscript preparation. QL, WZ, ZL, and WW performed the experiments.

FUNDING

This work was supported by the National Natural Science Foundation of China (Nos. 62072127 and 62002076), Project 6142111180404 supported by CNKLSTISS, Science and Technology Program of Guangzhou, China (Nos. 202002030131 and 201904010493), Guangdong Basic and Applied Basic Research Fund Joint Fund Youth Fund (No. 2019A1515110213), Open Fund Project of Fujian Provincial Key Laboratory of Information Processing and Intelligent Control (Minjiang University) (No. MJUKF-IPIC202101), Natural Science Foundation of Guangdong Province (No. 2020A1515010423).

REFERENCES

Bishop, C. M., and Nasser, M. N. (2006). *Pattern Recognition and Machine Learning, Vol. 4.* New York, NY: Springer.

Cui, J., Liu, S., Wang, L., and Jia, J. (2021). "Learnable boundary guided adversarial training," in *Proceedings of the IEEE/CVF International Conference on Computer Vision* (Montreal, QC), 15721–15730.

Dai, Z., Yang, Z., Yang, F., Cohen, W. W., and Salakhutdinov, R. R. (2017). "Good semi-supervised learning that requires a bad gan," in *Advances in Neural Information Processing Systems,* eds I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Long Beach, CA: Curran Associates), 6510–6520.

Deng, S., Cai, Q., Zhang, Z., and Wu, X. (2021a). User behavior analysis based on stacked autoencoder and clustering in complex power grid environment. *IEEE Trans. Intell. Transp. Syst.* 1–15. doi: 10.1109/TITS.2021.3076607

Deng, S., Chen, F., Dong, X., Gao, G., and Wu, X. (2021b). Short-term load forecasting by using improved gep and abnormal load recognition. *ACM Trans. Internet Technol. (TOIT)* 21, 1–28. doi: 10.1145/3447513

Dong, Y., Liao, F., Pang, T., Su, H., Zhu, J., Hu, X., et al. (2018). "Boosting adversarial attacks with momentum," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Salt Lake City, UT), 9185–9193.

Fang, Y., Chen, P., and Han, T. (2022). Hint: harnessing the wisdom of crowds for handling multi-phase tasks. *Neural Comput. Appl.* 1–23. doi: 10.1007/s00521-021-06825-7

Feng, S., Yan, X., Sun, H., Feng, Y., and Liu, H. X. (2021). Intelligent driving intelligence test for autonomous vehicles with naturalistic and adversarial environment. *Nat. Commun.* 12, 1–14. doi: 10.1038/s41467-021-21007-8

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Bing, X., Warde-Farley, D., Ozair, S., et al. (2014a). "Generative adversarial nets," in *International Conference on Neural Information Processing Systems* (Montreal, QC).

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014b). Explaining and harnessing adversarial examples. *arXiv preprint* arXiv:1412.6572.

Grandvalet, Y., and Bengio, Y. (2004). "Semi-supervised learning by entropy minimization," in *International Conference on Neural Information Processing Systems* (Vancouver, BC).

Jin, L., Wei, L., and Li, S. (2022). Gradient-based differential neural-solution to time-dependent nonlinear optimization. *IEEE Trans. Autom. Control* 1. doi: 10.1109/TAC.2022.3144135

Khan, M. M., Mehnaz, S., Shaha, A., Nayem, M., and Bourouis, S. (2021). Iot-based smart health monitoring system for covid-19 patients. *Comput. Math. Methods Med.* 2021, 1–11. doi: 10.1155/2021/8591036

Kingma, D. P., Mohamed, S., Rezende, D. J., and Welling, M. (2014). "Semi-supervised learning with deep generative models," in *Advances in Neural Information Processing Systems, Vol. 2* (Cambridge, MA: MIT Press), 3581–3589.

Laine, S. M., and Aila, T. O. (2018). *Temporal Ensembling for Semi-Supervised Learning.* U.S. Patent App. 15/721,433.

Liu, M., Chen, L., Du, X., Jin, L., and Shang, M. (2021). Activated gradients for deep neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* 1–13. doi: 10.1109/TNNLS.2021.3106044

Luo, Y., Zhu, J., Li, M., Ren, Y., and Zhang, B. (2018). "Smooth neighbors on teacher graphs for semi-supervised learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Salt Lake City, UT), 8896–8905.

Maaløe, L., Sønderby, C. K., Sønderby, S. K., and Winther, O. (2016). Auxiliary deep generative models. *arXiv preprint* arXiv:1602.05473.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *arXiv preprint* arXiv:1706.06083.

Miyato, T., Maeda, S.-i., Koyama, M., and Ishii, S. (2018). Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE Trans. Pattern Anal. Mach. Intell.* 41, 1979–1993. doi: 10.1109/TPAMI.2018.2858821

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., et al. (2015). Human-level control through deep reinforcement learning. *Nature* 518, 529. doi: 10.1038/nature14236

Pustejovsky, J., and Krishnaswamy, N. (2021). Embodied human computer interaction. *KI-Künstliche Intelligenz* 35, 307–327.

Sajjadi, M., Javanmardi, M., and Tasdizen, T. (2016). Regularization with stochastic transformations and perturbations for deep semi-supervised learning.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). "Improved techniques for training gans," in *Advances in Neural Information Processing Systems* (Red Hook, NY: Curran Associates), 2234–2242.

Strauss, T., Hanselmann, M., Junginger, A., and Ulmer, H. (2017). Ensemble methods as a defense to adversarial perturbations against deep neural networks. *arXiv preprint* arXiv:1709.03423.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., et al. (2013). Intriguing properties of neural networks. *arXiv preprint* arXiv:1312.6199.

Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. (2017). Ensemble adversarial training: Attacks and defenses. *arXiv preprint* arXiv:1705.07204.

Wahba, G. (1990). Spline models for observational data. *Technometrics* 34, 113–114.

Wang, X., Li, J., Kuang, X., Tan, Y.-a., and Li, J. (2019). The security of machine learning in an adversarial setting: a survey. *J. Parallel Distrib. Comput.* 130, 12–23. doi: 10.1016/j.jpdc.2019.03.003

Wu, D., He, Y., Luo, X., and Zhou, M. (2021a). A latent factor analysis-based approach to online sparse streaming feature selection. *IEEE Trans. Syst. Man Cybern. Syst.* 1–15. doi: 10.1109/TSMC.2021.3096065

Wu, D., Luo, X., Shang, M., He, Y., Wang, G., and Wu, X. (2020). A data-characteristic-aware latent factor model for web services qos prediction. *IEEE Trans. Knowl. Data Eng.* 1. doi: 10.1109/TKDE.2020.3014302

Wu, D., Shang, M., Luo, X., and Wang, Z. (2021b). An l1-and-l2-norm-oriented latent factor model for recommender systems. *IEEE Trans. Neural Netw. Learn. Syst.* 1–14. doi: 10.1109/TNNLS.2021.3071392

Yuan, X., He, P., Zhu, Q., and Li, X. (2019). Adversarial examples: Attacks and defenses for deep learning. *IEEE Trans. Neural Netw. Learn. Syst.* 30, 2805–2824. doi: 10.1109/TNNLS.2018.2886017

Zhang, C., Li, J., Wu, J., Liu, D., Chang, J., and Gao, R. (2022). Deep recommendation with adversarial training. *IEEE Trans. Emerg. Top. Comput.* 1. doi: 10.1109/TETC.2022.3141422

Zhang, W., Gao, B., Tang, J., Yao, P., Yu, S., Chang, M.-F., Yoo, H.-J., Qian, H., and Wu, H. (2020). Neuro-inspired computing chips. *Nat. Electron.* 3, 371–382. doi: 10.1038/s41928-020-0435-7

Zheng, S., Song, Y., Leung, T., and Goodfellow, I. (2016). "Improving the robustness of deep neural networks via stability training," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Las Vegas, NV), 4480–4488.