frontiers | Frontiers in Computer Science

TYPE Original Research
PUBLISHED 24 May 2023
DOI 10.3389/fcomp.2023.1114186

OPEN ACCESS

EDITED BY
Chaofeng Zhang,
Advanced Institute of Industrial Technology,
Japan

REVIEWED BY
Chong Di,
Qilu University of Technology, China
Jianwen Xu,
Muroran Institute of Technology, Japan

*CORRESPONDENCE
Shenghong Li
✉ shli@sjtu.edu.cn

RECEIVED 02 December 2022
ACCEPTED 20 April 2023
PUBLISHED 24 May 2023

CITATION
Yan Z, Li S, Duan Z and Zhao Y (2023) Robust deep semi-supervised learning with label propagation and differential privacy.
Front. Comput. Sci. 5:1114186.
doi: 10.3389/fcomp.2023.1114186

Robust deep semi-supervised learning with label propagation and differential privacy

Zhicong Yan[1], Shenghong Li[1]*, Zhongli Duan[2] and Yuanyuan Zhao[3]

[1]Department of Cyber Science and Engineering, Shanghai Jiao Tong University, Shanghai, China, [2]School of Art and Design, Zhengzhou Institute of Industrial Application Technology, Zhengzhou University, Zhengzhou, Hebei, China, [3]School of Information Science and Engineering, Hangzhou Normal University, Hangzhou, Zhejiang, China

Semi-supervised learning (SSL) methods provide a powerful tool for utilizing abundant unlabeled data to strengthen standard supervised learning. Traditional graph-based SSL methods prevail in classical SSL problems for their intuitional implementation and effective performance. However, they encounter troubles when applying to image classification followed by modern deep learning, since the diffusion algorithms face the curse of dimensionality. In this study, we propose a simple and efficient SSL method, combining a graph-based SSL paradigm with differential privacy. We aim at developing coherent latent feature space of deep neural networks so that the diffusion algorithm in the latent space can give more precise predictions for unlabeled data. Our approach achieves state-of-the-art performance on the Cifar10, Cifar100, and Mini-imagenet benchmark datasets and obtains an error rate of 18.56% on Cifar10 using only 1% of all labels. Furthermore, our approach inherits the benefits of graph-based SSL methods with a simple training process and can be easily combined with any network architecture.

KEYWORDS

deep semi-supervised learning, label propagation, differential privacy, robust learning, mixup data augmentation

1. Introduction

Deep neural networks (DNN) have become the first choice for computer vision applications due to their prominent performance and flexibility. However, the harsh requirement for precisely annotated data has largely constrained the wide application of deep learning. It is generally known that the collection and annotation of large-scale data are extremely costly and time-consuming in some professional industries (e.g., healthcare, finance, and manufacturing). Therefore, semi-supervised learning (SSL), which utilizes abundant unlabeled data in deep learning applications, has become an important research trend in the field of artificial intelligence (Chapelle et al., 2006; Tarvainen and Valpola, 2017; Iscen et al., 2019).

Various approaches to SSL in terms of image classification have been proposed in recent years based on some prototypical assumptions. For instance, the manifold assumption, which states that high dimensional data usually lie on low dimensional manifolds, leads to some consistency-based SSL methods. Other two well-known assumptions, cluster assumption, and low-density separation assumption have also inspired some types of research. To summarize, two main research directions show great promise.

One direction explores generative model-based approaches. For instance, VAE (Kingma and Welling, 2013), generative adversarial network (GAN) (Goodfellow et al., 2014), and normalizing flows (Kobyzev et al., 2020) can establish a low-dimensional hidden variable capturing the manifold of the input data, and then, Bayesian inference can be applied to optimize the posterior probability of both labeled and unlabeled examples (Kingma et al., 2014; Makhzani et al., 2015; Rasmus et al., 2015; Maaløe et al., 2016). However, GAN is known to be extremely difficult in generating high-resolution images, despite a large amount of research in recent years (Radford et al., 2015; Gulrajani et al., 2017; Brock et al., 2018), making these approaches difficult to scale to large and complex dataset (Yu et al., 2019). Furthermore, the extensive computation cost in training generative models makes these approaches less practical in real-world applications (Brock et al., 2018).

Another direction tries to exert proper regularizations on the classifier using unlabeled data. Those regularizations could be summarized into two categories as follows: one is consistency regularization, where two similar images or two networks with related parameters are encouraged to have similar network outputs (Sajjadi et al., 2016; Tarvainen and Valpola, 2017; Miyato et al., 2019). Another type is based on the data graph. In traditional machine learning, manifold assumption-based algorithms usually establish a graph to describe the manifold structure, then employ the graph Laplacian to induce smoothness on the data manifold, such as *Harmonic function* (HF) (Zhu et al., 2003), *Label Propagation* (LP) (Zhou et al., 2003; Gong et al., 2015), and *Manifold Regularization* (MR) (Belkin et al., 2006).

Those two types of semi-supervised methods have their strengths and weakness, respectively. In terms of consistency-based regularization, those methods only consider the perturbations around each data point, ignoring the connections between data points. Therefore, they do not fully utilize the data structure, such as manifolds or clusters. This artifact could be avoided if the data structure is taken into consideration using graph-based methods, which define convex optimization problems and have closed form of solutions (Zhou et al., 2003; Gong et al., 2015; Tu et al., 2015). On the contrary, the performance of the graph-based methods will degrade if the input data graph cannot satisfy the following conditions (Belkin et al., 2006): capturing the manifold structure of the input space and representing the similarity between two data points. Some traditional research aims at improving the graph quality (Jebara et al., 2009). However, it is extremely difficult to capture the manifold of high-dimensional image data, causing poor performance on image recognition tasks (Kamnitsas et al., 2018; Luo et al., 2018; Li et al., 2020, 2022; Ren et al., 2022).

Motivated by the above observation, we introduce differential privacy (Dwork and Roth, 2014) and mixup data augmentation (Zhang et al., 2018) in the graph-based SSL method. For both labeled and unlabeled data points, we force the predictions of network changes linearly in the vector from one data point to another, which forces the middle features to change linearly as well. We also employ differential privacy, to further boost the consistency of latent feature space by adding random noise to its latent representation layers. We observe that such regularization results in a more compact and coherent latent feature space given by the network and leads to a high-quality graph that captures the data manifold more accurately. Compared with the consistency-based methods (Sajjadi et al., 2016; Tarvainen and Valpola, 2017; Miyato et al., 2019), the proposed method demonstrate dominant performance with fewer label and much higher convergence speed, which means it can achieve the same performance with fewer computational cost. Compared with previous graph-based SSL methods, the proposed method tends to form more coherent latent feature space and achieves higher performance.

To summarize, there are two key contributions of our study as follows:

- We propose a simple but effective regularization method that can be applied to a graph-based SSL framework to regularize the latent space of deep neural network during the training process so that the graph-based SSL framework can work better with high-dimensional image datasets.
- We experimentally show that the proposed method achieves significant performance in improvement over the previous graph-based SSL method (Iscen et al., 2019) on SSL standard benchmarks and demonstrates competitive results to other state-of-the-art SSL methods.

2. Related work

In this section, we roughly categorized the recent advances in SSL into generative methods and graph-based methods and briefly introduce these two types of methods.

2.1. Generative SSL methods

Instead of directly estimating posterior probability $p(y|x)$, the generative methods pay attention to learning the class distributions $P(x|y)$ or the joint distribution $p(x, y) = p(x|y)p(y)$, to compute the posterior probability using Bayes' formula. In this framework, SSL can be modeled as a missing data problem while the unlabeled data can be used to optimize the marginal distribution $p(x) = \sum_{y\in Y} p(x|y)p(y)$. The joint log-likelihood on both labeled data set D_L and unlabeled data set D_U is naturally considered an objective function (Chapelle et al., 2006) as follows:

$$\sum_{(x_i,y_i)\in D_L} \log \pi_{y_i} p(x_i|y_i,\theta) + \sum_{x_i\in D_U} log \sum_{y=1}^{M} \pi_y p(x_i|y,\theta), \tag{1}$$

where $\pi_y = p(y|\theta)$ is class prior. Previous studies under this framework use auto-encoder (Rasmus et al., 2015) or variational auto-encoder (Kingma and Welling, 2013) to model $p(x, y)$. Unfortunately, since the neural network has excessive representational power, optimizing marginal distribution cannot guarantee to achieve the right joint distribution, making those approaches perform less well in large datasets.

To better estimate $p(x|y)$, generative adversarial network (GAN) (Goodfellow et al., 2014) has been introduced to the SSL framework. GAN is well-known for high-quality realistic image generation, Radford et al. (2015) employed fake examples from a conditional GAN as additional training data. Salimans et al. (2016) strengthened the discriminator to classify input data as well as distinguish fake examples from real data. Besides, BadGAN (Dai et al., 2017) argued that generating low-quality examples which lie in low-density areas between different classes can better guide the classifier position its the decision boundary. However, those approaches are based on practice and lack theoretical analysis, showing weak performance compared with newly emerged approaches.

2.2. Graph-based SSL methods

Graph-based methods operate on a weighted graph $G = (V, E)$ with adjacency matrix A, where the vertex set V is composed of all data samples denoted as $D = D_L \cup D_U$, and the elements of adjacency matrix A_{ij} are based on similarities between vertices $x_i, x_j \in V$. The smoothness assumption states that close data points should have similar predictions. Label Propagation (LP) iteratively propagated the class posterior of each node to its neighbors, faster through a short connection between data nodes, until a global equilibrium is reached. Zhou et al. (2003) showed that the same solution is arrived at by enforcing the smoothness term or equivalently minimizing the energy on the graph as follows:

$$R(\mathrm{f}) = \frac{1}{2}\sum_{i,j} A_{ij}\left(\mathrm{f}_i - \mathrm{f}_j\right)^2 = \mathrm{f}^{\mathrm{T}}\Delta \mathrm{f} \qquad (2)$$

Here, f_i is the pseudo label of the i-th data sample, $\Delta = D - A$ is the traditional graph Laplacian where D is a diagonal matrix with $D_{ii} = \sum_{j=1}^{n} A_{ij}$. If f is replaced with f(X) which is the output of a parameterized function on all data samples, the R(f) reaches a graph Laplacian regularizer which forces the function f to be harmonic. Some modifications to Equation (2) are made to mitigate the effect of outliers in the graph (Gong et al., 2015; Tu et al., 2015). An inevitable drawback of these approaches is that their performance largely relies on the quality of the input graph.

Our study is inspired by a series of recent methods which utilize the LP with a dynamically constructed graph in the optimization process. Acting like EM algorithms, those methods alternate between the two steps, the first step is to use the embeddings obtained by the deep neural network to construct the nearest neighbor graph, and then LP is performed on this graph to infer pseudo-label for the unlabeled images. After that, the network is trained using both labeled and pseudo-labeled data. In addition to the time cost by LP, this method just uses standard back-propagation methods to train the deep neural network, making it fast and efficient. Luo et al. (2018) used the graph constructed from embeddings obtained by the teacher model which is acting better than the student model. Kamnitsas et al. (2018) utilize LP to infer clusters in network middle representation, then encourage points in the same cluster to be closer. Iscen et al. (2019) introduce entropy as an uncertainty measure of pseudo-labels, then reduce the cost of uncertain examples. All of those approaches construct the graph actively in the optimization process to improve the quality of pseudo-labels.

3. The proposed method

The common challenge of graph-based methods is the need for a well-behaved graph that captures the geometry manifold of input data. In the following, we formalize our approach and emphasize our efforts in improving the quality of the graph. The key motivation of our method is to form a better network representation for better graph construction.

3.1. Problem formulation

We assume that the input space is $\mathcal{X} \subseteq \mathbb{R}^N$. We have a collection of l labeled samples $X_L := \{x_1, ..., x_l\}$ with $x_i \in \mathcal{X}$, their labels are given by $Y_L = \{y_1, ..., y_l\}$ with $y_i \in C$, where $C = \{1, ..., c\}$ is the set of discrete labels for c classes. In addition, u extra samples $X_U = \{x_{l+1}, ..., x_{l+u}\}$ are given without any label information. The whole set of samples is denoted as $D = X_L \cup X_U$. The transductive goal in SSL is to find the possible label set $\hat{Y}$ for all unlabeled samples, while the inductive goal is to find a classifier $f : \mathcal{X} \mapsto \mathbb{R}^c$ which can generalize well on unseen samples by utilizing all samples D and label Y_L. In this study, we focus on the inductive settings and use the convolutional neural network (CNN) as the classifier in our experiments.

3.2. Overview

Given a randomly initialized neural network f parameterized by θ, we introduce a new optimization process for SSL that can be summarized as follows. First, we perform pre-training of the network using only labeled data to warm it up, where we introduce mixup data argumentation. Then, we start the iterative SSL training process, perform label propagation to infer pseudo-labels, and optimize the network with both labeled and pseudo-labeled data. We point out two critical improvements compared with previous approaches: mixup regularization with pseudo-labeled data and deformed graph Laplacian-based label propagation. In addition, we incorporate the pseudo-label certainty and class-balancing strategy from the study by Iscen et al. (2019) into our approach. A graphical view of the proposed approach is shown in Figure 1.

3.3. Supervised mixup pre-training

In the early stage of the training process, the neural network f is composed of randomly initialized weight parameters. The output of the neural network is chaotic and has little semantic information about the input images. In previous studies, the network is pre-trained using the labeled samples by minimizing supervised cost only. Standard optimization techniques are employed in this procedure. The optimization target is the expectation of loss function ℓ over the labeled data distribution P_D as follows:

$$R_{su}(f_\theta) = \mathop{\mathbb{E}}_{(x,y)\sim D_L}\left[\ell\left(f_\theta(x), y\right)\right], \qquad (3)$$

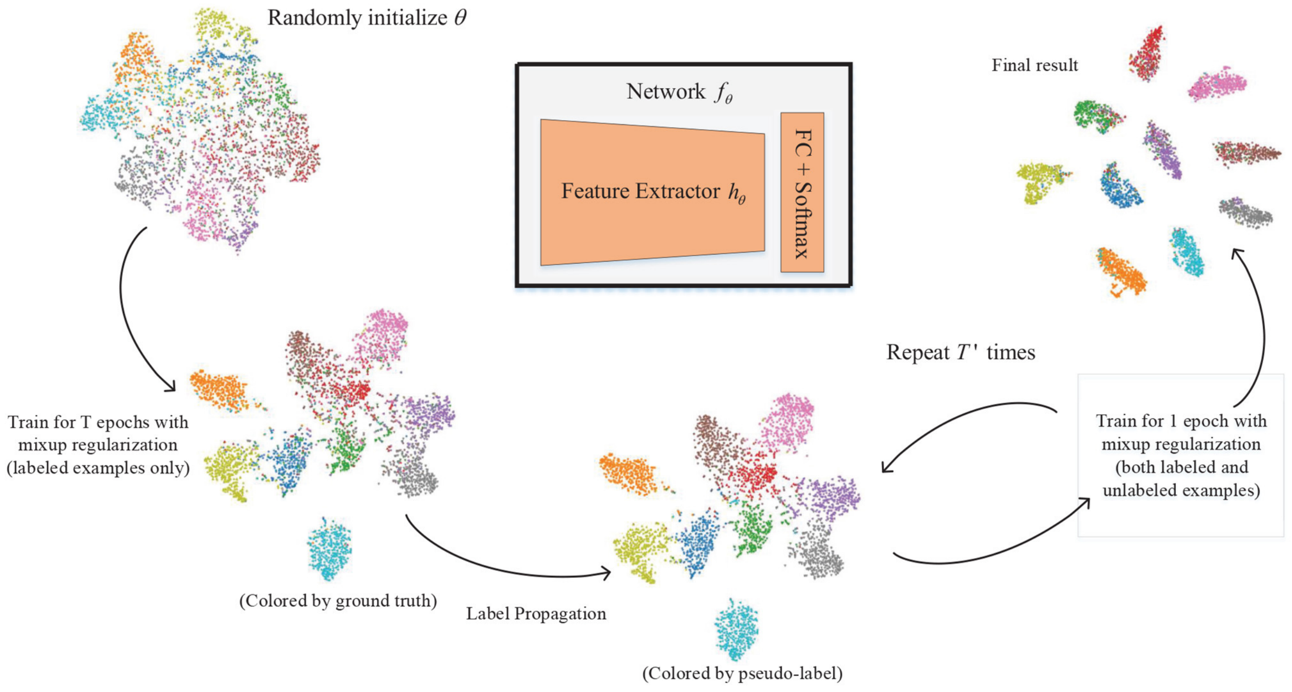
Randomly initialize θ
Network f_θ
Feature Extractor h_θ
FC + Softmax
Final result
Train for T epochs with mixup regularization (labeled examples only)
(Colored by ground truth)
Label Propagation
Repeat T' times
Train for 1 epoch with mixup regularization (both labeled and unlabeled examples)
(Colored by pseudo-label)

FIGURE 1
Overview of the proposed method. The colored points are the t-SNE visualization of the feature vectors extracted from Cifar10 train data by the same deep neural network in different training stages (we squeeze the 128-D feature vectors into 2-D plane coordinates using t-SNE). Starting from a randomly initialized network, we first train it with labeled data ($l = 1,000$) only and differential privacy to form a primitive network representation. Then, we perform label propagation and train the network on the entire dataset, with both handcraft labels and pseudo-labels. We repeat this process for T' times until convergence.

where $D_L(x, y) = \frac{1}{l}\sum_{i=1}^{l} \delta(x = x_i, y = y_i)$ is a point distribution that is employed to estimate the true data distribution P_D when labeled data set X_L and correspondent label Y_L are given. With fewer labeled data in the SSL setting, the point distribution is hardly able to estimate the true data distribution P_D, causing the model overfitting and degeneration. Previous studies take measures to mitigate this problem, using a larger learning rate and fewer pre-training epochs (Iscen et al., 2019). Other regularization techniques such as dropout and weight decay are adopted by Luo et al. (2018) and Miyato et al. (2019).

In this study, we introduce mixup (Zhang et al., 2018) in the pre-training procedure. Instead of simple point distribution, mixup proposes a vicinal distribution to estimate P_D, whose generative process is summarized as follows:

$$\begin{aligned} (x_i, y_i), (x_j, y_j) &\sim D_L, \\ \lambda &\sim Beta(\alpha, \alpha), \\ x &:= \lambda x_i + (1-\lambda)x_j, \\ y &:= \lambda y_i + (1-\lambda)y_j, \end{aligned} \tag{4}$$

where α is a hyperparameter $\alpha \in (0, \infty)$, which controls the strength of interpolation between data pairs. Notably, D_M will degrade to D_L as $\alpha \rightarrow 0$. We denote the mixup distribution as $D_M(D_L, \alpha)$. By replacing the D_L with mixup distribution in Equation (3), we have as follows:

$$R_{su}(f) = \mathop{\mathbb{E}}_{(x,y)\sim D_M(D_L,\alpha_{su})}\left[\ell\left(f_\theta(x), y\right)\right]. \tag{5}$$

In a nutshell, we just randomly sample two data points each time and perform standard supervised training using the mixed data point. Notably, the interpolation between y_i and y_j is an interpolation between two one-hot encoded probability vectors. This modification does not influence the optimization process, which means any optimizer and network architecture can be applied with this regularization.

We summarize the benefits of introducing mixup mainly in two-folds as follows:

(1) Overfitting problem is greatly alleviated. While the output posterior probability of the classifier is forced to transit linearly from class to class, the decision boundaries between classes are pushed into the intermediate area, reducing the number of undesirable results when predicting outside the train examples. The experiments will demonstrate the test error of the pre-trained network is reduced by introducing mixup regularization.

(2) The internal representation of the classifier is encouraged to transit linearly as well as the output, leading to abstract representations in smooth and coherent feature space, and then more accurate label

propagation is preformed based on sample distance in feature space.

3.4. Label propagation via nearest neighbor graph

Given a classification network f with pre-trained parameters θ, we clarify how to infer the possible label for unlabeled examples, which is employed to guide the SSL optimization process in the following subsection. First, we construct a graph based on network internal representations of all examples, and then, we apply label propagation on the graph to infer pseudo labels.

In most cases, the deep neural network can be seen as a sequence of non-linear layers or transformations, with each transformation giving an internal representation of the input. While the low-level representation captures more details of the input image, the high-level representation contains more semantic information about the input image, and the last layer maps the input feature into class probabilities. In a nutshell, network f can be decomposed as $f = g \circ h$ where $h: \mathcal{X} \mapsto \mathbb{R}^d$ is a feature extractor and $g: \mathbb{R}^d \mapsto \mathbb{R}^c$ is the output layer usually consisting of a fully-connected layer with softmax. We take h as a low-dimensional feature extractor and denote the feature vector for the i-th example as $v_i := h(x_i)$. We extract all features set of D as $V = \{v_1, .., v_l, v_{l+1}, ..., v_n\}$ for similarity computation.

The next is to construct a graph on N data nodes. While computing the full $N \times N$ affinity matrix A may be intractable, we approximate it by constructing a nearest neighbor graph and only count the similarity between nodes and their k nearest neighbors. Thus, we create a graph with a sparse affinity matrix $A \in \mathbb{R}^{n \times n}$ and its elements as follows:

$$a_{ij} := \begin{cases} s(v_i, v_j), & \text{if } i \neq j \wedge v_i \in \mathrm{NN}_k(v_j) \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where $\mathrm{NN}_k(v_i)$ denotes the set of k nearest neighbors of v_i in D, and s is the similarity function. The choice of s is quite flexible. Since we need to approximate the semantic similarity of two instances, we adopt the Gaussian similarity function $s(v_i, v_j) = \exp\left(\|v_i - v_j\|^2/\sigma^2\right)$ with hyperparameter σ. Notably, approximate nearest neighbor (ANN) algorithms can be applied to accelerate the graph construction for large N.

Hereafter, let $W := A + A^{\mathrm{T}}$ be the symmetric affinity matrix and $\mathcal{W} = D^{-1/2} W D^{-1/2}$ be its normalized counterpart, D is the diagonal degree matrix, in which the element is defined by $D_{ii} := \sum_{j=1}^{n} W_{ij}$. Further more, the volume of the graph is formulated as $\nu = \sum_{i=1}^{n} d_{ii}$.

After defining those parameters, we are going to describe our LP algorithm by defining input and output as two $n \times c$ matrix Y, Z. Y is the matrix of the given label with rows $\{y_1, ..., y_l, y_{l+1}, ...y_n\}$, where the first l rows are corresponding one-hot encoded labels of each labeled example, and the rest are all zero vectors. Z is the desired class posterior probabilities which are solved by minimizing the following cost function:

$$\begin{aligned} \min_{Z} Q(Z) =& \frac{1}{2}\Big[\beta \sum_{i,j=1}^{n} w_{ij} \Big\| \frac{z_i}{\sqrt{d_{ii}}} - \frac{z_j}{\sqrt{d_{jj}}} \Big\|^2 \\ &+ \gamma \sum_{i=1}^{n} (1 - d_{ii}/\nu)\|z_i\|^2 + \sum_{i=1}^{n} \|z_i - y_i\|^2\Big] \\ =& \frac{1}{2}\Big[\beta tr(Z^{\mathrm{T}} \Delta Z) + \gamma tr(Z^{\mathrm{T}}(I - D/\nu)Z) \\ &+ \|Z - Y\|_F^2\Big]. \end{aligned} \tag{7}$$

Here, z_i is the i-th row of matrix Z, $\Delta = I - \mathcal{W}$ is the normalized graph Laplacian and $\|\cdot\|_F$ is the Frobenius norm. The first term encourages smoothness where similar examples tend to induce the same predictions, and the last term attempts to maintain predictions for labeled examples (Zhou et al., 2003). In addition, the outlier which is indicated by a lower degree d_{ii} is forced to have a weak label in the second term. Thus, the degree of smoothness and weakness of outliers is controlled by weight parameters β and γ individually. To find the optimal solution of Z, we set the derivative of Equation (7) with respect to Z to 0 and obtain as follows:

$$\beta \Delta Z^* + \gamma (I - D/\nu) Z^* + Z^* - Y = 0. \tag{8}$$

Thus, the optimal Z is defined as follows:

$$Z^* = \left(I + \beta\Delta + \gamma(I - D/\nu)\right)^{-1} Y. \tag{9}$$

Let $k_1 = \frac{\beta}{1+\beta+\gamma}, k_2 = \frac{\gamma}{(1+\beta+\gamma)\nu}$ and ignore the constant part in Equation (9). We have as follows:

$$Z^* = \left(I - k_1\mathcal{W} - k_2 D\right)^{-1} Y. \tag{10}$$

Directly computing Z^* by Equation (10) is often intractable for large n because the inverse matrix $\left(I - k_1\mathcal{W} - k_2 D\right)^{-1}$ is not sparse, Instead, we use the conjugate gradient (CG) method to solve the linear system as follows:

$$\left(I - k_1\mathcal{W} - k_2 D\right)^{-1} Z = Y. \tag{11}$$

This method is applicable because $\left(I - k_1\mathcal{W} - k_2 D\right)^{-1}$ is a positive-definite matrix. The CG method has been adopted in many LP applications (Zhou et al., 2003; Gong et al., 2015; Tu et al., 2015; Iscen et al., 2019). Finally, the pseudo-label for an unlabeled example is given as follows:

$$\hat{y}_i = \arg\max_{j} z_{ij}. \tag{12}$$

Equation (11) is a hard assignment by evaluating the most confident class of each example; however, the contrast between classes can reflect the certainty of each example. Following Iscen et al. (2019), we associate a measure of confidence to each unlabeled example of calculating the entropy of Z:

$$w_i := 1 - \frac{H(\hat{z}_i)}{\log(c)}, \tag{13}$$

where $\hat{z}_i$ is the normalized counterpart of z_i, in other words, $\hat{z}_{ij} = z_{ij}/\sum_k z_{ik}$, function $H: \mathbb{R}^c \mapsto \mathbb{R}$ is the entropy function.

3.5. Mixup regularization with differential privacy

Given the pseudo-label and confidence measure of all unlabeled data, we associate them with each example and denote the point distribution of unlabeled data as $D_U(x, y, w) = \frac{1}{u}\sum_{i=l}^{l+u}\delta(x = x_i, y = \hat{y}_i, w = w_i)$. To take the confidence coefficient into account, we propose the mixup distribution of unlabeled data as D_{MU} whose generative process is summarized as follows:

$$\begin{aligned} (x_i, \hat{y}_i, w_i), (x_j, \hat{y}_j, w_j) &\sim D_U, \\ \lambda &\sim Beta(\alpha, \alpha), \\ x &:= \lambda x_i + (1-\lambda)x_j, \\ \hat{y} &:= \lambda \hat{y}_i + (1-\lambda)\hat{y}_j, \\ w &:= \lambda w_i + (1-\lambda)w_j, \end{aligned} \tag{14}$$

In this generative process, the input data are interpolated randomly between two examples, while the pseudo-label and confidence score are interpolated with the same proportion as well.

The abundant unlabeled data are used in the training process by minimizing the following cost along with labeled data.

$$R_{unsu}(f_\theta) = \mathop{\mathbb{E}}_{(x,\hat{y},w)\sim D_{MU}(D_U,\alpha_{unsu})}\left[w\ell\big(f_\theta(x), \hat{y}\big)\right]. \tag{15}$$

While the interpolation of labeled data can help form better network representation, in the unsupervised part, introducing interpolation of unlabeled data can lead to two benefits as follows: (1) The decision boundaries are pushed far away from unlabeled data, which are a desired property of low-density separation assumption. The model is forced to make neutral predictions in the middle zone of different samples or namely different clusters. (2) The clusters in hidden space are encouraged to have only one class of pseudo label, respectively. Considering the clusters in hidden space, if data points in one cluster are pseudo-labeled by two different classes, the mixup loss term will tear this cluster apart. The middle of this cluster is both encouraged to have neutral predictions as the interpolation of edge points or encouraged to have clear predictions as the middle points of the cluster.

Moreover, we employ differential privacy by directly adding noise to the latent representation of the deep neural network in the training progress:

$$f^*_\theta(x) = g(h(x) + \epsilon), \tag{16}$$

where ϵ is randomly sampled from $\mathcal{N}(0, \sigma^2)$. This procedure is inspired by the Ladder network (Rasmus et al., 2015). By adding noise to its latent representation, the neural network will have more resistance to the dataset bias and form a more coherent latent space.

Finally, we finetune the network by minimizing the following objective function using both labeled and unlabeled data:

$$R_{all}(f_\theta, D_L, D_U) = R_{su}(f_\theta, D_L) + \lambda R_{unsu}(f^*_\theta, D_U), \tag{17}$$

where λ is a coefficient that controls the effects of the unsupervised term.

3.6. Iterative training

We summarize our approach with the above definitions. Given a convolution neural network f with randomly initialized weights θ, we begin by training the network with mixup regularization for T epochs using the supervised loss term (Equation 5), and then, we start the following iterative process. First, we extract feature vector set V on the entire training set X and construct a nearest neighbor graph by computing the adjacency matrix via Equation (6). Second, we perform label propagation by solving the linear system (Equation 11) and assign pseudo-label and confidence score by Equations (12) and (13). Finally, we train the network for one epoch by minimizing the cost (Equation 17) on both labeled and unlabeled data set. This iterative process is repeated for T' epochs.

The whole training process is summarized in Algorithm 1, where procedure *Optimize*() refers to the mini-batch optimization of the given loss term for one epoch. In our experiment, we randomly sample a mini-batch of data and perform mixup interpolation within this mini-batch, and this strategy is used to reduce I/O consumption and report no harm to the result in the study by Zhang et al. (2018). The procedure *NearestNeighborGraph*() refers to the nearest neighbor graph construction based on the feature vector set V and the computing of edge value in the graph.

4. Experiments

In this section, we conduct our experiments with several standard image datasets commonly used in image classification. We first describe the datasets and our implementation details, and then, we compare the proposed method with the state-of-the-art methods. Finally, we conduct an ablation study to give a deep investigation into our method.

```
$\theta \leftarrow$ initialize randomly;
for epoch $\in [1, ..., T]$ do
    $\theta \leftarrow Optimize(R_S(X_L, Y_L, \theta))$;
end for
for epoch $\in [1, ..., T']$ do
    for $i \in 1, ..., n$ do
        $v_i \leftarrow h_\theta(x_i)$;
    end for
    $A \leftarrow NearestNeighborGraph(V)$ ;
    $W \leftarrow A + A^{\mathrm{T}}$;
    $\Delta \leftarrow I - D^{-1/2}WD^{-1/2}$;
    $Z \leftarrow$ solve with CG;
    for $i \in 1, ..., n$ do
        $\hat{y}_i \leftarrow \arg\max_j z_{ij}$;
        $w_i \leftarrow 1 - H(\hat{z}_i)/log(c)$;
    end for
    $\theta \leftarrow Optimize(R_{all}(X_L, Y_L, X_U, \hat{Y}_U, \theta))$;
end for
```

Algorithm 1. Mini-batch training with LP for SSL.

4.1. Datasets

We conduct experiments on three datasets, such as Cifar10, Cifar100, and Mini-imagenet. Cifar10 is widely used in related study, and Mini-imagenet is adopted by Iscen et al. (2019), to evaluate the proposed method on a large-scale dataset. Those datasets are commonly used in SSL setting by randomly taking a certain amount of labels and all image data to train the network and evaluate on the test set for fair a comparison with fully supervised methods, while the use of other labels in the training process is forbidden.

4.1.1. Cifar10, Cifar100

Cifar10 and Cifar100 datasets (Krizhevsky, 2009) are adopted in the evaluation process of previous SSL methods. The two datasets consist of small images of size 32 × 32. The training set of Cifar10 contains 50 k images, and its test set contains 10 k images, collected from 10 classes. Similar to Cifar10, Cifar100 has 50 and 10 k images for training and test, respectively, instead, Cifar100 collects images from 100 classes. For Cifar10, we randomly choose 50, 100, 200, and 400 images from each class, as the labeled images in our evaluation corresponding to 500, 1,000, 2,000, and 4,000 labels in total. For each class, we also choose 500 images as the validation images and employ the best model in validation to get the final performance on the test dataset. Following the common practice, we repeat the selection process 10 times, and for each time, we run the algorithm once on the dataset split and report mean error and standard deviation of test accuracy.

4.1.2. Mini-imagenet

Mini-imagenet was proposed by Gidaris and Komodakis (2018) for a few-shot learning evaluation, which is a simplified version of the Imagenet dataset. We adopt the same setting as the study by Iscen et al. (2019). Mini-imagenet consists of 100 classes with 600 images in each class, we randomly choose 500 images per class for the training set and use the remaining 100 images for testing.

4.2. Implementation details

4.2.1. Networks

We adopt a "13-layer" network for experiments on Cifar10 and Cifar100, which is a baseline used in all experiments in Table 1, and Resnet-18 is employed for experiments on Mini-imagenet. All of those networks consist of a feature extractor h_θ, followed by a linear classification layer. The l_2-normalization after the feature extractor in the study by Iscen et al. (2019) is canceled, which reported slightly harmful performance since we employ the Mahalanobis distance between features instead of the dot product as the similarity function.

4.2.2. Hyper-parameters

The following hyper-parameters are adopted in all experiments. First, we train the model with labeled data for 30 epochs, then we finetune the model with all data for 270 epochs for the experiments on Cifar10 and Cifar100 and 370 epochs for the experiments on Mini-imagenet. The training is performed using the SGD optimizer in all experiments. The learning rate is decayed from 0.1 to 0 with cosine annealing (Loshchilov and Hutter, 2016), and the momentum and weight decay parameters are set to 0.9 and 0.0001, respectively.

For the three hyperparameters, k, k_1, k_2 introduced in Section 3.5, and we set $k = 10$ in Equation (6) for fast graph construction and set $k_1 = 0.99, k_2 = 0.0005$ in Equation (11), where we implement the CG algorithm using the python sci-kit package. Other two hyperparameter mixup coefficients α_{su} and α_{unsu} are set to 1.0 in all our experiments. We set the value of λ in Equation (17) to 10 for all experiments.

TABLE 1 Comparison with state-of-the-art methods on Cifar10 using 13-layer ConvNet network architecture.

Nb. labels	500 labels	1,000 labels	2,000 labels	4,000 labels
Nb. images	50,000 images	50,000 images	50,000 images	50,000 images
Supervised w/o. mixup	46.22 ± 2.93	33.09 ± 1.13	24.32 ± 0.34	17.75 ± 0.15
Supervised w. mixup	44.65 ± 1.01	34.84 ± 1.37	24.86 ± 0.42	16.89 ± 0.16
BadGAN† (Dai et al., 2017)	–	–	–	14.41 ± 0.30
VAT† (Miyato et al., 2019)	–	–	–	11.36 ± 0.34
MT† (Tarvainen and Valpola, 2017)	–	27.36 ± 1.30	15.73 ± 0.31	12.31 ± 0.28
SWA† (Athiwaratkun et al., 2019)	–	15.58 ± 0.12	11.02 ± 0.23	9.05 ± 0.21
LP† (Iscen et al., 2019)	32.40 ± 1.80	22.02 ± 0.88	15.66 ± 0.35	12.69 ± 0.29
LP+MT† (Iscen et al., 2019)	24.02 ± 2.44	16.93 ± 0.70	13.22 ± 0.29	10.61 ± 0.28
ICT† (Verma et al., 2019)	–	15.48 ± 0.78	**9.26 ± 0.09**	**7.29 ± 0.09**
Ours	**18.56 ± 1.58**	**14.74 ± 0.55**	10.14 ± 0.30	8.58 ± 0.27

The error rate is reported over 10 runs. †Denotes scores reported in previous studies. Bold values means the best result under the described experimental settings.

TABLE 2 Comparison with the state-of-the-art methods on Cifar100 using 13-layer ConvNet network architecture.

Nb. labels	4,000 labels	10,000 labels
Nb. images	50,000 images	50,000 images
Supervised w/o. mixup	51.82 ± 0.51	39.81 ± 0.53
Supervised w. mixup	52.43 ± 0.43	38.53 ± 0.29
LP† (Iscen et al., 2019)	46.20 ± 0.76	38.43 ± 1.88
LP+MT† (Iscen et al., 2019)	43.73 ± 0.20	35.92 ± 0.47
Ours	**38.87 ± 0.43**	**32.15 ± 0.25**

The error rate is reported over 10 runs. †Denotes scores reported in previous studies. Bold values means the best result under the described experimental settings.

TABLE 3 Comparison with state-of-the-art methods on Mini-imagenet using the resnet-18 network.

Dataset	4,000 labels	10,000 labels
Nb. labeled images	50,000 images	50,000 images
Supervised	63.57 ± 0.59	48.25 ± 0.33
LP+MT†	70.29 ± 0.81	57.58 ± 1.47
Ours	**48.86 ± 0.11**	**40.08 ± 0.93**
Fully supervised with all labels	31.97 ± 1.46	31.97 ± 1.46

The error rate is reported over three runs. †Denotes scores reported in previous studies. Bold values means the best result under the described experimental settings.

4.3. Comparison with state-of-the-art methods

In this section, we present a comparison with the state-of-the-art methods. We choose representative methods from three categories, such as generative SSL methods (BadGAN; Dai et al., 2017), consistency-based SSL methods [VAT; (Miyato et al., 2019), MT (Tarvainen and Valpola, 2017), and ICT (Verma et al., 2019)], and graph-based SSL methods (LP; Iscen et al., 2019). The performance of various methods on three datasets is represented in Tables 1–3, respectively.

The proposed method outperforms other methods with the same network architecture. On the Cifar10 dataset, our method achieves a significant error rate reduction (∼ 20%) compared with our precedent method (Iscen et al., 2019), showing that our method exactly amends its weakness and successfully mitigates the performance gap between the graph-based SSL framework and other SSL methods. Compared with the best consistency-based method (Verma et al., 2019) to our knowledge, our method performs slightly weaker with 4,000 labels in total but outperforms it with fewer labeled images. This shows the advantage of traditional graph-based SSL learning that it can make more effective utilization of available labels, which are still applicable when it comes to modern deep learning architecture. We also try to use even fewer labels to evaluate the robustness of our method. Holding 500 labels (∼ 1% of all), our method still achieves 18.56% error rate on the Cifar10 dataset.

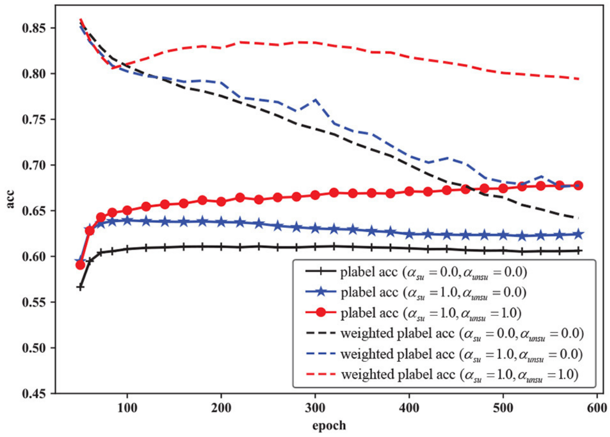

FIGURE 2
Pseudo-label accuracy and weighted pseudo-label accuracy with different mixup conditions on Mini-imagenet (10,000 labels are given in the training process, and accuracy is calculated according to ground truth). $\alpha = 0.0$ means no mixup operation and $\alpha = 1.0$ means mixup coefficient, λ is drawn from a uniform distribution. The results show that the applied regularization in both two losses greatly improves the pseudo-label accuracy during the iterative training process.

4.4. Ablation studies

We conduct ablation studies to investigate the impact of mixup regularization on the pseudo-labels. To access the quality of pseudo-labels, accuracy is an important indicator. Despite this, we utilize the confidence score to calculate the weighted accuracy of pseudo-labels: $Acc_{weighted} = \frac{1}{u}\sum_{i=1}^{u} w_i \delta(\hat{y}_i = y_i)$. During the iterative optimization process, if the models are not capable of correcting wrong pseudo-labels, the confidence of those mistakes will increase and eventually get close to 1, leading to $Acc_{weighted} \approx Acc$. The weighted accuracy indicator can reflect if the model really learns something useful from unlabeled images or if it just remembers the pseudo-labels. Figure 2 shows the progress of accuracy and the weighted accuracy of pseudo-labels throughout the training process. The experiments are conducted on Mini-imagenet with 100 labels per class. The results show that if no mixup regularization is applied or only applying mixup regularization on labeled data, the accuracy of pseudo-labels only increases in the beginning and tends to be stable in the following epochs, while the weighted accuracy curve keeps declining until getting close to the accuracy curve. These results imply that without regularization, the deep neural network just remembers the pseudo-labels due to its excessive representational ability. In contrast, our regularization method successfully alleviates such undesired phenomenon, and the accuracy of pseudo-labels is keep increasing during the training process.

In Table 4, we compare the performance of Mini-imagenet. The results show that our method greatly reduces the error rate compared with the baseline method even on a high-resolution image dataset. To investigate the effectiveness of differential privacy in the proposed method, we vary the noise scale from 0 to 1.0 and report the performance on different datasets in Table 5. The

TABLE 4 Impact of mixup regularization on pair of labeled data points or unlabeled data points.

Nb. labels Nb. images	10,000 labels 50,000 images
LP+MT†	57.35
LP+MT	48.07
LP+MT+Su.mixup	44.39
LP+MT+Su.mixup+Unsu.mixup	39.43

The error rate is reported on Mini-imagenet with 10,000 labels.

TABLE 5 Impact of varying the noise scale σ.

Noise scale	Cifar10	Cifar100	Mini-imagenet
0	12.31	37.98	43.72
0.01	10.24	35.32	41.30
0.1	8.58	32.15	39.65
1	8.69	31.90	39.61

results clearly show that the added noise reduces the error rate of the final model.

5. Conclusion and future work

In this study, we present a simple but effective regularization method in the graph-based SSL framework. Based on the previously proposed method that extends the traditional graph-based SSL framework for modern deep learning of image recognition, our study further strengthens this research line by introducing two critical measures as follows: imposing regularization on the latent space of the deep neural network and preventing the outlier data points from hurting the label propagation process. We show that our approach is effective and practical in utilizing unlabeled images via evaluation on both simple datasets of Cifar10 and Cifar100 and complex datasets with high resolution (Mini-imagenet). Furthermore, our method is computationally efficient and easy to implement the experiment on Mini-imagenet costs approximately 5h using a single NVIDIA 1080TI GPU. Our study also demonstrates differential privacy, which is an effective technique to constrain the excessive representation power of deep neural networks. Future study includes designing more delicate and effective regularization techniques in the SSL framework to further mitigate the performance gap between semi-supervised learning and supervised learning with all labels.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

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

Funding

This research study was funded in part by the Defence Industrial Technology Development Program under grant No. JCKY2020604B004.

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.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Athiwaratkun, B., Finzi, M., Izmailov, P., and Wilson, A. G. (2019). "There are many consistent explanations of unlabeled data: why you should average," in *Proceedings of the International Conference on Learning Representations (ICLR)* (New Orleans, LA).

Belkin, M., Niyogi, P., and Sindhwani, V. (2006). Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. *J. Mach. Learn. Res.* 7, 2399–2434.

Brock, A., Donahue, J., and Simonyan, K. (2018). "Large scale GAN training for high fidelity natural image synthesis," in *Proceedings of the International Conference on Learning Representations (ICLR)* (Vancouver, BC).

Chapelle, O., Schölkopf, B., and Zien, A. (2006). *Semi-Supervised Learning*. The MIT Press. doi: 10.7551/mitpress/9780262033589.001.0001

Dai, Z., Yang, Z., Yang, F., Cohen, W. W., and Salakhutdinov, R. R. (2017). "Good semi-supervised learning that requires a bad GAN," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (Long Beach, CA), *Vol. 30*.

Dwork, C., and Roth, A. (2014). The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.* 9, 211–407. doi: 10.1561/0400000042

Gidaris, S., and Komodakis, N. (2018). "Dynamic few-shot visual learning without forgetting," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Salt Lake City, UT), 4367–4375. doi: 10.1109/CVPR.2018.00459

Gong, C., Liu, T., Tao, D., Fu, K., Tu, E., and Yang, J. (2015). Deformed graph Laplacian for semisupervised learning. *IEEE Trans. Neural Netw. Learn. Syst.* 26, 2261–2274. doi: 10.1109/TNNLS.2014.2376936

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., et al. (2014). "Generative adversarial nets," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (Montreal, QC).

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. (2017). "Improved training of wasserstein GANs," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (Long Beach, CA), *Vol. 30*.

Iscen, A., Tolias, G., Avrithis, Y., and Chum, O. (2019). "Label propagation for deep semi-supervised learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. (Long Beach, CA). doi: 10.1109/CVPR.2019.00521

Jebara, T., Wang, J., and Chang, S.-F. (2009). "Graph construction and b-matching for semi-supervised learning," in *Proceedings of the International Conference on Machine Learning (ICML)* (Montreal, QC), 441–448. doi: 10.1145/1553374.1553432

Kamnitsas, K., Castro, D., Le Folgoc, L., Walker, I., Tanno, R., Rueckert, D., et al. (2018). "Semi-supervised learning via compact latent space clustering," in *International Conference on Machine Learning* (Stockholm), 2459–2468.

Kingma, D. P., Mohamed, S., Jimenez Rezende, D., and Welling, M. (2014). "Semi-supervised learning with deep generative models," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (Montreal, QC), *Vol. 27*.

Kingma, D. P., and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Kobyzev, I., Prince, S. J., and Brubaker, M. A. (2020). Normalizing flows: an introduction and review of current methods. *IEEE Trans. Pattern Anal. Mach. Intell.* 43, 3964–3979. doi: 10.1109/TPAMI.2020.2992934

Krizhevsky, A. (2009). *Learning multiple layers of features from tiny images* (Master's thesis). University of Toronto, Toronto, ON, Canada.

Li, G., Ota, K., Dong, M., Wu, J., and Li, J. (2020). Desvig: decentralized swift vigilance against adversarial attacks in industrial artificial intelligence systems. *IEEE TII* 16, 3267–3277. doi: 10.1109/TII.2019.2951766

Li, G., Wu, J., Li, S., Yang, W., and Li, C. (2022). Multi-tentacle federated learning over software-defined industrial internet of things against adaptive poisoning attacks. *IEEE Trans. Indus. Inform.* 19, 1260–1269. doi: 10.1109/TII.2022.3173996

Loshchilov, I., and Hutter, F. (2016). SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*.

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/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* Salt Lake City, UT. doi: 10.1109/CVPR.2018.00927

Maaløe, L., Sønderby, C. K., Sønderby, S. K., and Winther, O. (2016). "Auxiliary deep generative models," in *Proceedings of the International Conference on Machine Learning (ICML)* (New York, NY), *Vol. 48*, 1445–1453.

Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., and Frey, B. (2015). Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*.

Miyato, T., Maeda, S.-I., Koyama, M., and Ishii, S. (2019). 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

Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.

Rasmus, A., Berglund, M., Honkala, M., Valpola, H., and Raiko, T. (2015). "Semi-supervised learning with ladder networks," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (Montreal, QC), *Vol. 28*.

Ren, G., Wu, J., Li, G., Li, S., and Guizani, M. (2022). Protecting intellectual property with reliable availability of learning models in ai-based cybersecurity services. *IEEE TDSC* 1–18. doi: 10.1109/TDSC.2022.3222972

Sajjadi, M., Javanmardi, M., and Tasdizen, T. (2016). "Regularization with stochastic transformations and perturbations for deep semi-supervised learning," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (Barcelona), 1163–1171.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). "Improved techniques for training GANs," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (Barcelona), *Vol. 29*.

Tarvainen, A., and Valpola, H. (2017). "Mean teachers are better role models: weight-averaged consistency targets improve semi-supervised deep learning results," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (Long Beach, CA), 1195–1204.

Tu, E., Yang, J., Kasabov, N., and Zhang, Y. (2015). Posterior distribution learning (PDL): a novel supervised learning framework using unlabeled samples to improve classification performance. *Neurocomputing* 157, 173–186. doi: 10.1016/j.neucom.2015.01.020

Verma, V., Lamb, A., Kannala, J., Bengio, Y., and Lopez-Paz, D. (2019). "Interpolation consistency training for semi-supervised learning," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)* (Macao), 3635–3641. doi: 10.24963/ijcai.2019/504

Yu, B., Wu, J., Ma, J., and Zhu, Z. (2019). "Tangent-normal adversarial regularization for semi-supervised learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (Long Beach, CA), 10676–10684. doi: 10.1109/CVPR.2019.01093

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. (2018). "Mixup: beyond empirical risk minimization," in *Proceedings of the International Conference on Learning Representations (ICLR)* (Vancouver, BC).

Zhou, D., Bousquet, O., Lal, T. N., Weston, J., and Olkopf, B. S. (2003). "Learning with local and global consistency," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)* (Vancouver, BC).

Zhu, X., Ghahramani, Z., and Lafferty, J. D. (2003). "Semi-supervised learning using gaussian fields and harmonic functions," in *Proceedings of the International Conference on Machine Learning (ICML)* (Washington, DC), 912–919.