Weighted Contrastive Learning with Hard Negative Mining for Positive and Unlabeled Learning

Botai Yuan, Chen Gong, *Senior Member, IEEE,* Dacheng Tao, *Fellow, IEEE,* Jie Yang, *Senior Member, IEEE*

***Abstract*—Positive and Unlabeled (PU) learning aims to train a suitable classifier simply based on a set of positive data and unlabeled data. The state-of-the-art methods usually formulate PU learning as a cost-sensitive learning problem, in which every unlabeled example is treated as negative with modified class weights. However, existing methods fail to generate high-quality data representations, which brings about negative-prediction preference and performance decline. To overcome this problem, this paper proposes a novel algorithm dubbed "Weighted Contrastive Learning with Hard Negative Mining for Positive and Unlabeled Learning" (termed "WConPU"), which specifically designs a new prototypical contrastive strategy for gaining discriminative representations for PU learning. Specifically, our proposed WConPU consists of a contrastive learning module and a classifier training module, which can benefit from each other in an iterative manner. Moreover, a novel weighted contrastive objective function equipped with a prototype-based hard negative mining module is proposed to further enhance the representation quality. Theoretically, we show that our WConPU can be justified from the perspective of Expectation-Maximization (EM) algorithm. Empirically, we compare our method with state-of-the-art PU algorithms on a wide range of real-world benchmark datasets, and the experimental results firmly demonstrate the advantage of our proposed method over the existing PU learning approaches.**

***Index Terms*—Positive and Unlabeled Learning, Contrastive Learning, Hard Negative Mining.**

1 Introduction

RECENT years have witnessed a surge of research interest in Positive and Unlabeled learning (*i.e.*, PU learning) [1]–[8], of which the target is to train a binary classifier based on labeled positive data and unlabeled data. PU learning finds its usefulness in many real-world problems such as disease diagnosis [9], hyperspectral image classification [10], anomaly detection [11], [12], etc.

Numerous PU algorithms have been developed over the past decades [1], [5], [13], [14]. Early works usually follow a two-step strategy [15], [16], which first identifies reliable negatives from unlabeled data, and then uses (semi-)supervised learning to train a binary classifier with the reliable negative examples and labeled positive examples. Another prevalent research line is to formulate PU learning as a cost-sensitive learning problem [1], [14], [17], which directly treats unlabeled data as negative data with modified importance weights and has achieved state-of-the-art performance.

Although the cost-sensitive methods have achieved good results, they have some potential limitations, namely treating unlabeled examples as negative may lead the classifier to have a negative-prediction preference. This is because these methods fail to generate high-quality representations (see in Fig. 4), therefore they cannot discriminate the positive examples and negative examples within unlabeled data from the perspective of feature representation, and thus impairing the learning performance.

To ameliorate this issue, we propose a novel PU learning algorithm dubbed "**W**eighted **Con**trastive Learning with Hard Negative Mining for **P**ositive and **U**nlabeled Learning" (termed "WConPU"), which can well separate the positive data and negative data in representation space and facilitate the training of an unbiased classifier. The main strategy is to utilize prototypical contrastive learning to simultaneously utilize the supervision information as well as find discriminative representations of PU data. Specifically, two important steps (*i.e.*, *contrastive learning* and *classifier training*) alternate, so that they can benefit from each other via an iterative manner. In contrastive learning step, positive pairs and negative pairs are constructed for conducting contrastive learning to obtain discriminative data representations. To this end, the predicted labels output by classifier are employed, where the two examples with same predicted label constitute a positive pair, and the examples with dissimilar predicted labels are considered as a negative pair (Section 4.1). In classifier training step, a classifier is trained by utilizing the pseudo labels of examples, where each pseudo label is updated according to the similarity between the representation of corresponding example and the class prototype (Section 4.2). As such, a well-trained classifier obtained in classifier training step could help generate high-quality representations in the contrastive learning step, which will in turn facilitate the further improvement of the classifier. Our algorithm works in an end-to-end manner, which helps to generate discriminative feature representations for all training data for reliable PU classifier training. Theoretically, we demonstrate that the contrastive learning module and classifier training module can benefit from each other in an iterative manner through the perspective of Expectation-Maximization (EM) algorithm (Section 5), where the classifier training step corresponds to E-step, and the contrastive learning step

This research is supported by NSF of China (Nos: 62376153, 62336003, 12371510), NSF for Distinguished Young Scholar of Jiangsu Province (No: BK20220080), and the National Research Foundation, Singapore, and the CyberSG R&D Programme Office ("CRPO"), under the National Cybersecurity R&D Programme ("NCRP"), RIE2025 NCRP Funding Initiative (Award CRPO-GC1-NTU-002).

B. Yuan, C. Gong, and J. Yang are with the Department of Automation, the Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, Shanghai 200240, China. (e-mail: yuan_botai@sjtu.edu.cn, chen.gong@sjtu.edu.cn, jieyang@sjtu.edu.cn).

D. Tao is with the College of Computing and Data Science, Nanyang Technological University, Singapore. (e-mail: dacheng.tao@gmail.com).

Corresponding authors: J. Yang, C. Gong, and D. Tao.

corresponds to M-step.

To further enhance the discriminability of the obtained classifier, a novel weighted contrastive objective function along with a prototype-based hard negative mining module is proposed to further separate the positive examples and negative examples in the representation space. Hard negative examples correspond to those examples in a pair which belong to different classes but are quite close in the representation space. Since the ground-truth label of unlabeled data is not accessible in PU learning, the proposed hard negative mining module resorts to the class prototype to ensure the negativeness of the selected examples, and the Euclidean distance between the embeddings of two data points is employed to determine the hardness of the selected examples. With the mining of hard negative examples, our proposed contrastive objective function reweights the hard negative examples according to their dissimilarity with the anchor input, so that hard negative examples would be paid more attention during the gradient descent process. The novel weighted contrastive loss function could further help improve representation quality without introducing any extra trade-off parameters. Moreover, a label distribution alignment loss [18] is employed to further ameliorate the negative-prediction preference inherited by cost-sensitive methods such as [9], [14], and [19].

In summary, the main contributions of this paper are outlined as follows:

- **Methodology.** A novel PU learning algorithm termed WConPU is presented, which leverages contrastive learning to incorporate supervisory information and generate high-quality representations of PU data. Moreover, a novel weighted contrastive objective function and a prototype-based hard negative mining module are developed to enhance the quality of the learned representations.
- **Theory.** We justify that our WConPU algorithm can be understood from the perspective of Expectation-Maximization (EM) algorithm, which ensures the convergence of our WConPU towards an (locally) optimal solution.
- **Experiments.** Our WConPU method yields superior performance to existing typical PU learning approaches on various PU benchmark datasets. For example, we improve the best baseline method by 3.10% on SVHN [20] dataset.

2 Related Work

In this section, we review the prior works relevant to this paper, including PU learning and contrastive learning.

2.1 Positive and Unlabeled Learning

PU learning is proposed for the setting where only positive and unlabeled data are accessible for training, and the unlabeled data include both positive and negative examples [21], but their labels are unknown before training. PU learning [22] can be roughly attributed to two-step methods [15], [16], and cost-sensitive methods [9], [13], [14]. The two-step approaches first identify reliable negative examples from the unlabeled set, and then build a classifier on the positive set, reliable negatives, and the remaining unlabeled set. Such methods mainly differ in the ways to pick up reliable negatives. For example, "Classification from Positive, Unlabeled and Biased Negative Data" (PUbN) [23] pretrained a model with "Positive-Unlabeled Learning with Non-Negative Risk Estimator" (nnPU) [14] algorithm to recognize some reliable negative examples, and then combined positive risk, negative risk, and unlabeled risk to train the final classifier. Graph-based methods [24] assigned unlabeled data a pseudo label according to the distance between examples on a graph. Generative learning [25] has also been introduced to PU learning. For instance, "Generative adversarial positive-unlabeled learning" (GenPU) [26] utilized the framework of adversarial learning to train a negative example generator, and then incorporated the labeled positive data and generated negative data to train a classifier. HolisticPU [27] assigned pseudo-labels to unlabeled examples by identifying the unique predictive trend of each example.

Such sample selection methods might be vulnerable to the mis-identified negatives, and the current state-of-the-art result is usually achieved by cost-sensitive methods. The cost-sensitive approaches attempt to construct an unbiased or biased risk estimator by assigning the data points with different importance weights. Since the first unbiased PU (uPU) risk estimator [13] was proposed, numerous works have been done to further enhance the performance [9], [17]. The authors of uPU employed a convex surrogate loss to reduce the computational cost [1]. Since the empirical risk of training data could go negative when training a flexible deep neural network, the non-negative risk estimator, known as nnPU [14], was proposed. Besides, "Loss Decomposition and Centroid Estimation" (LDCE) [3] decomposed the loss function of corrupted negative examples into a label-independent term and a label-dependent term, where only the latter term is influenced by the label noise. Dist-PU [18] pursued the consistency between the distribution of predicted labels and the class prior and has achieved state-of-the-art performance. Recently, "Positive-Unlabeled Learning With Label Distribution Alignment" (PULDA) [28] extended Dist-PU [18] with a margin-based formulation, and involved the class prior estimation process. While the above PU learning methods assume that whether a positive instance is selected as labeled is irrelevant to its feature representation, namely instance-independent, some instance-dependent approaches were proposed, which assume that the labeling of a positive example depends on its feature. Representative methods include "Learning from Positive and Unlabeled Data with a Selection Bias" (PUSB) [19] and "Labeling Bias Estimation" (LBE) [4].

2.2 Contrastive Learning

Contrastive Learning (CL) is a discriminative model that achieves very impressive performance in self-supervised learning [29]–[33]. It pulls the examples in a positive pair close to each other, while driving the examples in a negative pair apart in the latent embedding space through contrastive loss [29], [30], [34]. The main difference between various contrastive learning approaches lies in their strategy for constructing

positive pairs. Chen *et al*. [29] extensively studied the effects of various data augmentation methods. MoCo [30], [35], [36] proposed a dictionary to maintain a negative sample set, therefore increasing the number of negative example pairs. "Bootstrap Your Own Latent" (BYOL) [37] consisted of an online network and a target network, which tried to get rid of the necessity of negative examples during the CL process. "Swapping Assignments between Views" (SwAV) [38] learned to predict the cluster assignment of one view from the representation of another view. "Simple Siamese" (SimSiam) [39] introduced Siamese networks to CL. Theoretically, Arora *et al*. [40] analyzed the effect of contrastive representation learning on a downstream classification task and provided a generalization bound for the standard contrastive objective function. Our work is also related to prototypical contrastive learning [41]–[44], where prototypes are used as representative features for classes. PCL [41] introduced prototypes as latent variables to help find the maximum-likelihood estimation of the network parameters in an Expectation-Maximization framework, which encourages representations to be closer to their prototypes. Different from the aforementioned self-supervised CL approaches, "Supervised Contrastive Learning" (SCL) [45] proposed a supervised contrastive objective that considers the examples in the same category as positive pairs to increase the utility of data.

It is worth mentioning that "Positive Unlabeled Contrastive Learning" (puNCE) [46] also introduces CL to PU learning. However, our WConPU has three main advantages over puNCE. Firstly, WConPU handles unlabeled examples in a more effective way. Compared with puNCE which simply labels each unlabeled example as positive and negative with different weights, WConPU leverages the output of classifier to provide precise supervisory information for contrasting unlabeled examples. Secondly, our WConPU implements CL in a more effective way. Compared with puNCE which runs in a two-step manner (*i.e.*, pre-training the encoder using a contrastive loss, and then linear probing with standard PU loss), our WConPU is end-to-end which directly utilizes prototypes to guide the updating of pseudo labels for classifier training. Thirdly, WConPU utilizes hard negative examples in a more effective way, as it incorporates hard negative mining which is not contained by puNCE to enhance the performance of CL. Above merits of WConPU lead to the improved results over puNCE (see the experimental results in Section 6).

Another similar work to our WConPU is PiCO [47], which is designed for partial label learning (PLL) [47]–[50]. Both PiCO and our WConPU share the similar idea of joint optimization of contrastive learning and classification. However, PiCO cannot be directly applied to PU learning because it is originally designed for partial label learning and cannot deal with PU data. For our proposed WConPU algorithm for PU learning, to effectively utilize the precious positive labels in PU learning, we design a contrastive loss consisted of a labeled positive part and an unlabeled part. Besides, PiCO constructs the positive peer set directly using predictions of the classifier, which may introduce noise and impair contrastive learning. In contrast, our approach introduces a self-adaptive threshold to refine the selection of positive pairs. Moreover, as the performance of the classifier in our WConPU depends heavily on the quality of learned representations, we propose a novel prototype-based hard negative mining module and a weighted contrastive loss to decrease the adverse impact of low-quality embeddings on final model performance. These advantages over PiCO critically enhance representation quality, ensuring a clear distinction between positive and negative data in the representation space, which is crucial for a well-performing PU classifier.

3 Problem Setting

In classical binary classification problem, let $\mathcal{X} \subseteq \mathbb{R}^d$ (d denotes dimension) and $\mathcal{Y} = \{0, 1\}$ be the input feature space and output label space, respectively. Let $p(\boldsymbol{x}, y)$ be the underlying joint density where $\boldsymbol{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$, and $p(\boldsymbol{x})$ be the marginal distribution of $p(\boldsymbol{x}, y)$, then the positive set and negative set for training a classifier can be denoted as:

$$\begin{aligned} X_{\mathrm{P}} &= \{\boldsymbol{x}_i\}_{i=1}^{n'_{\mathrm{P}}} \sim p_{\mathrm{P}}(\boldsymbol{x}), \\ X_{\mathrm{N}} &= \{\boldsymbol{x}_i\}_{i=1}^{n'_{\mathrm{N}}} \sim p_{\mathrm{N}}(\boldsymbol{x}), \end{aligned} \tag{1}$$

where n'_{P} and n'_{N} are the corresponding numbers of positive examples and negative examples, and $p_{\mathrm{P}}(\boldsymbol{x})$ and $p_{\mathrm{N}}(\boldsymbol{x})$ denote the marginal distributions of positive data and negative data accordingly. Let $\pi_{\mathrm{P}} = p(y = 1)$ be the positive class prior, then the whole set $X = X_{\mathrm{P}} \cup X_{\mathrm{N}}$ is generated as:

$$X = \{\boldsymbol{x}_i\}_{i=1}^{n} \sim p(\boldsymbol{x}), \tag{2}$$

$$p(\boldsymbol{x}) = \pi_{\mathrm{P}} \cdot p_{\mathrm{P}}(\boldsymbol{x}) + \pi_{\mathrm{N}} \cdot p_{\mathrm{N}}(\boldsymbol{x}), \tag{3}$$

where $n = n'_{\mathrm{P}} + n'_{\mathrm{N}}$ is the total number of training data, and $\pi_{\mathrm{N}} = 1 - \pi_{\mathrm{P}}$ denotes the negative class prior.

When it comes to PU learning, only a few labeled positive data and some unlabeled data is accessible. In this paper, we consider the case-control scenario [22], where two sets of data are sampled independently from $p_{\mathrm{P}}(\boldsymbol{x})$ and $p(\boldsymbol{x})$ as:

$$X_{\mathrm{L}} = \{\boldsymbol{x}_i\}_{i=1}^{n_{\mathrm{P}}} \sim p_{\mathrm{P}}(\boldsymbol{x}), \tag{4}$$

$$X_{\mathrm{U}} = \{\boldsymbol{x}_i\}_{i=1}^{n_{\mathrm{U}}} \sim p_{\mathrm{U}}(\boldsymbol{x}) = p(\boldsymbol{x}), \tag{5}$$

where n_{P} and n_{U} are the amounts of labeled positive examples and unlabeled examples, respectively, while X_{L} and X_{U} represent the labeled positive set and unlabeled set correspondingly. The aim of PU learning is to learn a classifier $f : \mathcal{X} \to [0, 1]^2$ with the training set $X_{\mathrm{PU}} = X_{\mathrm{L}} \cup X_{\mathrm{U}}$, such that the example $\boldsymbol{x}_i$ can obtain the correct label $\hat{y}_i = \arg\max_{j \in \{0,1\}} f^j(\boldsymbol{x}_i)$ assigned by f. For each $\boldsymbol{x}_i \in X_{\mathrm{PU}}$, we use o_i to indicate whether it is labeled as positive (*i.e.*, $o_i = 1$) or unlabeled (*i.e.*, $o_i = 0$).

4 Methodology

In this section, we describe our proposed WConPU framework in detail (see Fig. 1). Briefly speaking, the training of WConPU iterates between the steps of contrastive learning and classifier training. Specifically, contrastive learning module uses the output of classifier to determine the positive pairs and employs a novel weighted contrastive objective function which leverages hard negative examples to further improve the discriminability of obtained data representations. The hard

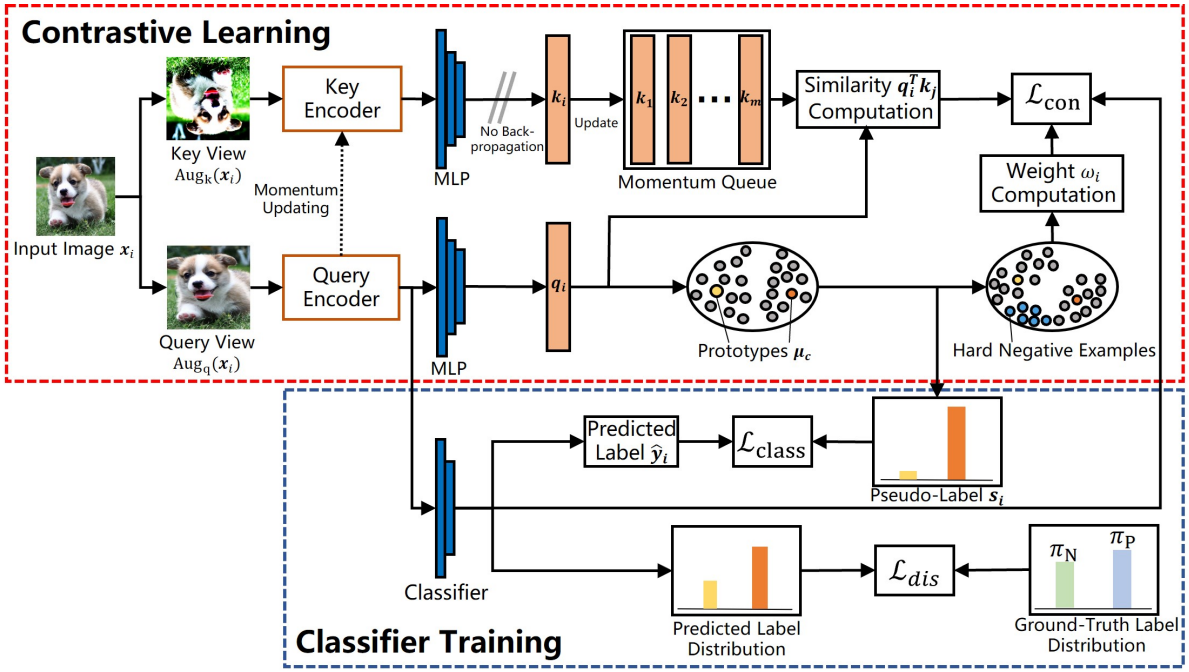


Fig. 1. Pipeline of WConPU. In contrastive learning module, given an input image $\boldsymbol{x}_i$, data augmentation is performed to create a query view $\mathrm{Aug}_q(\boldsymbol{x}_i)$ and a key view $\mathrm{Aug}_k(\boldsymbol{x}_i)$. Then a query embedding $\boldsymbol{q}_i$ and a key embedding $\boldsymbol{k}_i$ are generated, respectively, where the key embedding will be stored in a momentum queue. The query embedding $\boldsymbol{q}_i$ is used to update the class prototypes $\boldsymbol{\mu}_c$, which are utilized to construct the hard negative set of the input example. The hard negative examples will provide weight ω_i for our weighted contrastive objective function $\mathcal{L}_{\text{con}}$. The output of the classifier helps guide the construction of positive pairs for the contrastive learning module. In classifier training module, the class prototypes $\boldsymbol{\mu}_c$ are used to update the pseudo label $\boldsymbol{s}_i$, which is used to calculate the classification loss $\mathcal{L}_{\text{class}}$ with the predicted label $\hat{y}_i$. Moreover, a label distribution loss $\mathcal{L}_{\text{dis}}$ is employed to encourage the alignment of predicted labels with class prior. The contrastive learning module and classifier training module work in a collaborative manner iteratively.

negative examples are decided by a novel prototype-based hard negative mining module (Section 4.1). For classifier training, we employ the prototypes of different classes to update the pseudo labels of unlabeled examples for training the classifier. Besides, a label distribution alignment loss is deployed to further rectify the prediction bias inherited by cost-sensitive methods (Section 4.2).

4.1 Positive and Unlabeled Contrastive Learning

As mentioned above, CL is adopted by our method to enhance the discriminability of the obtained data representations. Since the dataset for PU learning contains precious positive labels, we should take measures to fully utilize them in a contrastive manner. Therefore, we propose a CL framework which is particularly effective under the setting of PU learning. Specifically, our framework follows the popular setup of MoCo [30] and SCL [45]. Given an anchor input $\boldsymbol{x}_i$ from the training set, a query view and a key view are generated through a weak augmentation $\mathrm{Aug}_q(\boldsymbol{x}_i)$ and a strong augmentation $\mathrm{Aug}_k(\boldsymbol{x}_i)$. The two augmented images are then respectively fed into a query network $g_q(\cdot)$ and a key network $g_k(\cdot)$, yielding a pair of ℓ_2-normalized embeddings $\boldsymbol{q}_i = g_q(\mathrm{Aug}_q(\boldsymbol{x}_i))$ and $\boldsymbol{k}_i = g_k(\mathrm{Aug}_k(\boldsymbol{x}_i))$. The key network shares the same architecture with the query network, and uses a momentum update aided by the query network. Formally, denoting the parameters of $g_q(\cdot)$ as θ_q and those of $g_k(\cdot)$ as θ_k, we update θ_k by:

$$\theta_k := \lambda\theta_k + (1-\lambda)\theta_q, \tag{6}$$

where $\lambda \in [0,1)$ is a momentum coefficient. Moreover, a momentum queue $\mathcal{Q} = \{\boldsymbol{k}_1, \boldsymbol{k}_2, ..., \boldsymbol{k}_m\}$ storing the most recent key embeddings is maintained, where m is the size of the momentum queue, and $\mathcal{Q}$ is updated chronologically during the training procedure. Meanwhile, we additionally save and update the ℓ_2-normalized prototype embedding $\boldsymbol{\mu}_c$ for $c \in \{0,1\}$. Here the prototype corresponds to a representative embedding for a group of semantically similar examples [41], and the computation of it will be introduced later. Given a mini-batch of examples $B = \{\boldsymbol{x}_i\}_{i=1}^b$, where b corresponds to the size of the current mini-batch, we have the following contrastive embedding pool:

$$\mathcal{A} = \mathcal{B}_q \cup \mathcal{B}_k \cup \mathcal{Q}, \tag{7}$$

where $\mathcal{B}_q = \{g_q(\mathrm{Aug}_q(\boldsymbol{x}_i)) | \boldsymbol{x}_i \in B\}$ and $\mathcal{B}_k = \{g_k(\mathrm{Aug}_k(\boldsymbol{x}_i)) | \boldsymbol{x}_i \in B\}$ contain the embeddings generated from the query view and key view of the current mini-batch, respectively.

Positive Peer Set Construction. The main challenge of introducing supervised CL to PU learning lies in constructing the positive peer set of anchor input $\boldsymbol{x}_i$ (see in Fig. 2). Here positive peers of $\boldsymbol{x}_i$ are the embeddings which constitute positive pairs with the anchor input $\boldsymbol{x}_i$. The true positive peer set of anchor input $\boldsymbol{x}_i$ is defined as:

$$\mathcal{P}(\boldsymbol{x}_i) = \{\boldsymbol{k}' | \boldsymbol{k}' \in \mathcal{A}(\boldsymbol{x}_i), y' = y_i\}, \tag{8}$$

where $\mathcal{A}(\boldsymbol{x}_i) = \mathcal{A} \backslash \{\boldsymbol{q}_i\}$, y_i represents the ground-truth label of $\boldsymbol{x}_i$, and y' represents the ground-truth label of the corresponding training example related to $\boldsymbol{k}'$.

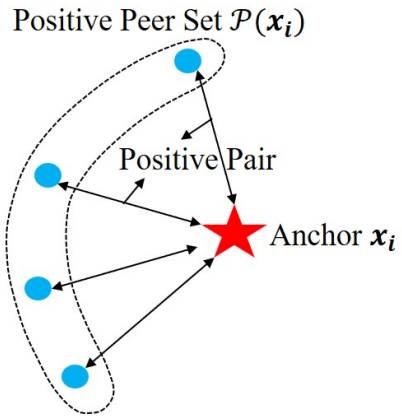


Fig. 2. Illustration of positive peer set $\mathcal{P}(\boldsymbol{x}_i)$. The positive peer set of the anchor input $\boldsymbol{x}_i$ is comprised of embeddings that form positive pairs with the anchor input $\boldsymbol{x}_i$. That is to say, these embeddings in $\mathcal{P}(\boldsymbol{x}_i)$ are associated with the examples that have the same ground-truth label as the anchor input $\boldsymbol{x}_i$.

However, the absence of ground-truth labels of unlabeled examples makes it impossible to obtain the true positive peer set of anchor input $\boldsymbol{x}_i$ in PU learning. An intuitive solution is to use the predicted labels $\hat{y}_i = \arg\max_{j\in\{0,1\}} f^j(\text{Aug}_q(\boldsymbol{x}_i))$ rendered by classifier to decide positive pairs. However, the result of classifier is unreliable during the early training stage, therefore directly using its output to construct positive pairs may erroneously identify some negative pairs as positive. To tackle this issue, we employ a "Self-Adaptive Threshold" (SAT) [51] to remove the uncertain positive pairs. When training starts, the threshold is low to accept more possibly correct examples into training. As the model becomes "stronger", the threshold adaptively increases to filter out possibly incorrect positive pairs to ensure the quality of embeddings. At iteration t, given a mini-batch of examples $\{\boldsymbol{x}_i\}_{i=1}^b$, where b corresponds to the batch size, by denoting $\boldsymbol{z}_i = \text{softmax}(f(\boldsymbol{x}_i))$ as the softmax output for the classifier $f(\cdot)$, the global threshold τ_t is updated as:

$$\tau_t = \lambda\tau_{t-1} + (1-\lambda)\frac{1}{b}\sum\nolimits_{i=1}^{b}\max(\boldsymbol{z}_i), \tag{9}$$

where $\lambda \in [0,1)$ is the momentum coefficient mentioned above. SAT also introduces a local threshold to modulate the global threshold in a class-specific fashion, which is:

$$\tilde{p}_t(c) = \lambda\tilde{p}_{t-1}(c) + (1-\lambda)\frac{1}{b}\sum\nolimits_{i=1}^{b}\boldsymbol{z}_i(c), \tag{10}$$

where $c \in \{0,1\}$ denotes the Class c, and $\boldsymbol{z}_i(c)$ denotes the c-th entry of $\boldsymbol{z}_i$. Both τ_t and $\tilde{p}_t(c)$ are initialized as $\frac{1}{2}$ at the beginning of the training process. By integrating the global threshold in Eq. (9) and the local threshold in Eq. (10), the final SAT $\tau_t(c)$ is given as:

$$\tau_t(c) = \frac{\tilde{p}_t(c)}{\max\{\tilde{p}_t(c) : c \in \{0,1\}\}} \cdot \tau_t. \tag{11}$$

For each $x_i \in X_\text{L}$, its positive peer set is denoted as $\mathcal{P}_\text{P}(\boldsymbol{x}_i)$. Similarly, for each $x_i \in X_\text{U}$, its positive peer set is denoted as $\mathcal{P}_\text{U}(\boldsymbol{x}_i)$. With the introduction of SAT, the positive peer set of $\boldsymbol{x}_i$ is decided as:

$$\begin{aligned}\mathcal{P}_\text{P}(\boldsymbol{x}_i) &= \{\boldsymbol{k}' | \boldsymbol{k}' \in \mathcal{A}(\boldsymbol{x}_i), \hat{y}' = 1, \max(z') \geq \tau_t(1)\} \cup X'_\text{L}, \\ \mathcal{P}_\text{U}(\boldsymbol{x}_i) &= \{\boldsymbol{k}' | \boldsymbol{k}' \in \mathcal{A}(\boldsymbol{x}_i), \hat{y}' = \hat{y}_i, \max(z') \geq \tau_t(\hat{y}')\},\end{aligned} \tag{12}$$

where $\boldsymbol{z}' = \text{softmax}(f(\boldsymbol{x}'))$ is the softmax result of classifier's output for the training example corresponding to $\boldsymbol{k}'$, $\hat{y}' = \arg\max_j f^j(\text{Aug}_q(\boldsymbol{x}'))$ is the predicted label for the corresponding training example related to $\boldsymbol{k}'$, and $X'_\text{L} = \{\boldsymbol{q}_i | \boldsymbol{q}_i = g_q(\text{Aug}_q(\boldsymbol{x}_i)), \boldsymbol{x}_i \in X_\text{L}\}$ comprises of the embeddings of labeled positive examples.

Weighted Contrastive Learning with Hard Negative Mining. Simply introducing the above positive-pair-based contrastive learning to PU learning is not enough to get high-quality representations. Recently, hard negative mining [52]–[55] was proposed to construct hard yet important negative pairs for further boosting the performance of CL, which is helpful to obtain discriminative data representations. Therefore, here we propose a novel weighted contrastive objective function equipped with a prototype-based hard negative mining module to help CL in generating more distinguishable representations. As mentioned in [52], a hard negative pair should satisfy two requirements, namely: 1) *True negativeness*, which means that the labels of two data points in a hard negative pair should be different; 2) *Hardness*, which means that the most useful hard negative examples are the ones of which the embeddings are similar to the query embedding. To measure the hardness of each example, we first propose a dissimilarity metric $DisSim(\cdot,\cdot)$ according to the Euclidean distance between two embeddings, namely:

$$DisSim(\boldsymbol{q}_i, \boldsymbol{k}_j) = \frac{1}{4}\left\| \frac{\boldsymbol{q}_i}{\|\boldsymbol{q}_i\|} - \frac{\boldsymbol{k}_j}{\|\boldsymbol{k}_j\|} \right\|^2, \tag{13}$$

where $\boldsymbol{q}_i = g_q(\text{Aug}_q(\boldsymbol{x}_i))$, $\boldsymbol{k}_j = g_k(\text{Aug}_k(\boldsymbol{x}_j))$, and $\|\cdot\|$ denotes the ℓ_2-norm of a vector. Obviously, $DisSim(\cdot,\cdot) \in [0,1]$.

In PU learning, the ground-truth label of unlabeled data is not accessible, so the "true negativeness" requirement mentioned above is difficult to satisfy exactly. Since a prototype is a representative embedding for a group of semantically similar examples [41], we resort to the similarity between the embedding of each example and the corresponding class prototype $\boldsymbol{\mu}_c$ to construct the hard negative set $\mathcal{B}_i^\text{neg}$, which contains the embeddings constituting hard negative pairs with query embedding $\boldsymbol{q}_i$. Denoting $Q_{\frac{1}{4}}(\boldsymbol{x}_i)$ as the first quartile of the values $\{DisSim(\boldsymbol{q}_i, \boldsymbol{k}_1), DisSim(\boldsymbol{q}_i, \boldsymbol{k}_2), ..., DisSim(\boldsymbol{q}_i, \boldsymbol{k}_m)\}$ by ascending order, then the hard negative set of $\boldsymbol{x}_i$ is given by:

$$\mathcal{B}_i^\text{neg} = \{\boldsymbol{k}_j | \boldsymbol{k}_j \in \mathcal{Q}, \tilde{y}_i \neq \tilde{y}_j, DisSim(\boldsymbol{q}_i, \boldsymbol{k}_j) \leq Q_{\frac{1}{4}}(\boldsymbol{x}_i)\}, \tag{14}$$

where $\tilde{y}_i = \arg\max_{c\in\{0,1\}} \boldsymbol{q}_i^\top \boldsymbol{\mu}_c$, and $\tilde{y}_j = \arg\max_{c\in\{0,1\}} \boldsymbol{k}_j^\top \boldsymbol{\mu}_c$. In above Eq. (14), the query embedding $\boldsymbol{q}_i$ and the key embedding $\boldsymbol{k}_j$ belong to different class prototypes, so they satisfy the *true negativeness* requirement. Meanwhile, the selected two embeddings are with extremely small dissimilarity value (*i.e.*, $DisSim(\boldsymbol{q}_i, \boldsymbol{k}_j) \leq Q_{\frac{1}{4}}(\boldsymbol{x}_i)$), which meets the *hardness* requirement.

To leverage the hard negative examples obtained from the hard negative mining step, we propose a novel weighted contrastive objective function which is specially designed for PU learning. Given an example $\boldsymbol{x}_i$, if it is labeled as positive (*i.e.*, $o_i = 1$), the weighted contrastive loss for it is given as:

$$\mathcal{L}_{\text{con}}^{\text{p}}(\boldsymbol{x}_i) = -\frac{1}{|\mathcal{P}_{\text{P}}(\boldsymbol{x}_i)|} \sum_{\boldsymbol{k}_+ \in \mathcal{P}_{\text{P}}(\boldsymbol{x}_i)} \tilde{\mathcal{L}}(\boldsymbol{q}_i, \boldsymbol{k}_+), \tag{15}$$

where $|\cdot|$ denotes the size of a set. Otherwise (*i.e.*, $o_i = 0$), the weighted contrastive loss for $\boldsymbol{x}_i$ is:

$$\mathcal{L}_{\text{con}}^{\text{u}}(\boldsymbol{x}_i) = -\frac{1}{|\mathcal{P}_{\text{U}}(\boldsymbol{x}_i)|} \sum_{\boldsymbol{k}_+ \in \mathcal{P}_{\text{U}}(\boldsymbol{x}_i)} \tilde{\mathcal{L}}(\boldsymbol{q}_i, \boldsymbol{k}_+), \tag{16}$$

where the $\tilde{\mathcal{L}}(\boldsymbol{q}_i, \boldsymbol{k}_+)$ in Eq. (15) and Eq. (16) is given as:

$$\log \frac{\exp(\boldsymbol{q}_i^\top \boldsymbol{k}_+ / \rho)}{\sum\limits_{\boldsymbol{k}' \in \mathcal{A}(\boldsymbol{x}_i) \backslash \mathcal{B}_i^{\text{neg}}} \exp(\boldsymbol{q}_i^\top \boldsymbol{k}' / \rho) + \sum\limits_{\boldsymbol{k}_j \in \mathcal{B}_i^{\text{neg}}} \omega_j \exp(\boldsymbol{q}_i^\top \boldsymbol{k}_j / \rho)}, \tag{17}$$

where $\omega_j = 1/DisSim(\boldsymbol{q}_i, \boldsymbol{k}_j)$ is the weight of the j-th hard negative pair, and $\rho \geq 0$ is the temperature coefficient. Following [45], $\{\boldsymbol{k} | \boldsymbol{k} \in \{\mathcal{A}(\boldsymbol{x}_i) \backslash \boldsymbol{k}_+\}\}$ is deemed as negative peer set, in which every element corresponds to the embedding of an example constituting the negative pair with anchor input $\boldsymbol{x}_i$.

Note that above proposed contrastive objective functions (*i.e.*, Eq. (15) and Eq. (16)) reweight the hard negative examples according to their dissimilarity with the anchor input. To be specific, the hard negative examples that are more similar to the anchor input should be paid more attention in the contrastive process as they are easily confused, so their weights ω_j should be enlarged. This weighted contrastive loss helps push hard negative examples away in the representation space to improve the discriminability of obtained data representations. The empirical study in Section 6.3 will show that our weighted contrastive loss noticeably improves the performance of the PU classifier without reweighting operation.

By combining Eq. (15) and Eq. (16), the weighted **con**trastive loss $\mathcal{L}_{\text{con}}$ in our method is expressed as:

$$\mathcal{L}_{\text{con}} = \frac{1}{n} \left[\sum_{\boldsymbol{x}_i \in X_{\text{L}}} \mathcal{L}_{\text{con}}^{\text{p}}(\boldsymbol{x}_i) + \sum_{\boldsymbol{x}_i \in X_{\text{U}}} \mathcal{L}_{\text{con}}^{\text{u}}(\boldsymbol{x}_i) \right]. \tag{18}$$

4.2 Classifier Training

Since the above-mentioned contrastive learning module can generate discriminative representations which contain implicit semantic information, we propose a prototype-based pseudo labeling mechanism on unlabeled data to enrich the supervision information for classifier training.

For each image $\boldsymbol{x}_i$ in the training set X_{PU}, we assign it a vector $\boldsymbol{s}_i \in [0,1]^2$ as the pseudo label. For the labeled positive data, their pseudo labels are exactly the one-hot label vectors and will not update during the training procedure. In contrast, the pseudo labels of unlabeled data are updated in a moving average manner which will be later introduced. With the help of pseudo labels, we can train a classifier $f : \mathcal{X} \to [0,1]^2$ by using cross-entropy loss. The **class**ification loss $\mathcal{L}_{\text{class}}$ is thus given by:

$$\mathcal{L}_{\text{class}} = \frac{1}{n} \sum\nolimits_{i=1}^{n} \sum\nolimits_{j=0}^{1} -s_{i,j} \log(z_{i,j}), \tag{19}$$

where $z_{i,j}$ is the j-th entry of the softmax result of $f(\boldsymbol{x}_i)$, and $s_{i,j}$ denotes the j-th entry of the pseudo label $\boldsymbol{s}_i$.

Pseudo Label Updating. Note that the prototypes mentioned in Section 4.1 actually contain precious semantic information, so the pseudo label of each example can be decided according to its nearest prototype. However, this method might bring unstableness to the network, as the prototypes might be inaccurate during early training stage. Therefore, we adopt a moving average manner to update the pseudo labels. Specifically, we first initialize the pseudo label for $\boldsymbol{x}_i$ according to the class prior, which is:

$$\boldsymbol{s}_i = \begin{cases} [0,1]^\top, & \text{if } o_i = 1, \\ [\pi_{\text{N}}, \pi_{\text{P}}]^\top, & \text{if } o_i = 0, \end{cases} \tag{20}$$

where $\pi_{\text{P}} = p(y=1)$ is the positive class prior, and $\pi_{\text{N}} = 1 - \pi_{\text{P}}$. Afterwards, the pseudo label is updated after each iteration, namely:

$$\boldsymbol{s}_i := \alpha \boldsymbol{s}_i + (1-\alpha) \boldsymbol{h}_i, \tag{21}$$

where $\alpha \in [0,1)$ controls the updating speed of pseudo label, and $\boldsymbol{h}_i$ is a one-hot vector given by:

$$h_{i,j} = \begin{cases} 1, & \text{if } j = \arg\max_{c \in \{0,1\}} \boldsymbol{q}_i^\top \boldsymbol{\mu}_c, \\ 0, & \text{else}, \end{cases} \tag{22}$$

where $\boldsymbol{\mu}_c$ is the prototype corresponding to the Class c, and $h_{i,j}$ is the j-th element of the one-hot vector $\boldsymbol{h}_i$. Note that according to Eq. (20) and Eq. (21), the updated pseudo labels naturally satisfy $\sum_{j=0}^{1} s_{i,j} = 1$.

Since each prototype incorporates implicit semantic information, this moving average mechanism would encourage pseudo label to gradually move towards the correct one. Meanwhile, because $\boldsymbol{h}_i$ is in a one-hot form, the pseudo label $\boldsymbol{s}_i$ would eventually get close to a one-hot vector, which carries confident label information and would facilitate the training of the classifier. The empirical study in Section 6.7 further proves the effectiveness of our prototype-based pseudo label updating mechanism.

Prototype Updating. The intuitive solution to update the class prototype is to average the representations of data points according to the output of the classifier after every epoch. However, this would bring two side effects, namely: 1) A large size of training data would incur heavy computational cost; and 2) Epoch-level updating mechanism might bring undesirable confirmation bias, which means that the classifier will mistakenly classify an example into an incorrect class with a high confidence. To ameliorate these issues, we also employ a moving average manner to update each class prototype, which is:

$$\boldsymbol{\mu}_c := \text{Normalize}(\lambda \boldsymbol{\mu}_c + (1-\lambda) \boldsymbol{q}_i), \tag{23}$$

where $c = \arg\max_{j \in \{0,1\}} f^j(\text{Aug}_q(\boldsymbol{x}_i))$, $\lambda \in [0,1)$ is the momentum coefficient mentioned above, and the Normalize$(\cdot)$ in Eq. (23) is ℓ_2-normalization operation.

Label Distribution Alignment. To further rectify the negative-prediction preference caused by the absence of clear negative labels in the training set X_{PU}, in this paper, we resort to the label distribution loss proposed in Dist-PU [18] to rectify this prediction bias. Dist-PU solves the PU learning from a label distribution perspective, in which the expected risk of conventional binary classifier R can be formulated as:

$$\begin{aligned} R =& \pi_{\mathrm{P}}\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{P}}(\boldsymbol{x})}\left[\mathbb{1}(\hat{y},1)\right] + \pi_{\mathrm{N}}\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{N}}(\boldsymbol{x})}\left[\mathbb{1}(\hat{y},0)\right] \\ =& \pi_{\mathrm{P}}\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{P}}(\boldsymbol{x})}\left[1-\hat{y}\right] + \pi_{\mathrm{N}}\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{N}}(\boldsymbol{x})}\left[\hat{y}\right] \\ =& \pi_{\mathrm{P}}\underbrace{\left|\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{P}}(\boldsymbol{x})}\left[\hat{y}\right]-1\right|}_{R_{\mathrm{P}}} + \pi_{\mathrm{N}}\underbrace{\left|\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{N}}(\boldsymbol{x})}\left[\hat{y}\right]\right|}_{R_{\mathrm{N}}}, \end{aligned} \tag{24}$$

where $\hat{y} = \arg\max_{j\in\{0,1\}} f^j(\boldsymbol{x})$ denotes the predicted label of the classifier, $\mathbb{E}(\cdot)$ denotes the expectation, and $\mathbb{1}(\hat{y}, y)$ corresponds to the zero-one loss. The value of $\mathbb{1}(\hat{y}, y)$ is 1 if $\hat{y} \neq y$, and 0 otherwise. Then Dist-PU concentrates on the difference between the expectation of the predicted labels and that of the underlying ground-truth labels, namely:

$$\begin{aligned} &\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\left[\hat{y}\right] - \mathbb{E}_{(\boldsymbol{x},y)\sim p(\boldsymbol{x},y)}\left[y\right] \\ =& \pi_{\mathrm{P}}\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{P}}(\boldsymbol{x})}\left[\hat{y}\right] - \pi_{\mathrm{P}} + \pi_{\mathrm{N}}\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{N}}(\boldsymbol{x})}\left[\hat{y}\right] \\ =& -\pi_{\mathrm{P}}R_{\mathrm{P}} + \pi_{\mathrm{N}}R_{\mathrm{N}}. \end{aligned} \tag{25}$$

By substituting Eq. (25) into Eq. (24), an equivalent form of R in a label distribution alignment manner can be derived as:

$$R = 2\pi_{\mathrm{P}}R_{\mathrm{P}} + \left|\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{U}}(\boldsymbol{x})}\left[\hat{y}\right] - \pi_{\mathrm{P}}\right|. \tag{26}$$

By denoting $z_{i,j}$ as the j-th entry of the softmax result of $f(\boldsymbol{x}_i)$, the label **dis**tribution loss $\mathcal{L}_{\mathrm{dis}}$ is expressed as:

$$\mathcal{L}_{\mathrm{dis}} = 2\pi_{\mathrm{P}}\left|\frac{1}{n_{\mathrm{P}}}\sum_{\boldsymbol{x}_i\in X_{\mathrm{L}}} z_{i,1} - 1\right| + \left|\frac{1}{n_{\mathrm{U}}}\sum_{\boldsymbol{x}_i\in X_{\mathrm{U}}} z_{i,1} - \pi_{\mathrm{P}}\right|. \tag{27}$$

This loss function forces the distribution of predicted labels to align with the class prior, and thus further overcoming the negative-prediction preference inherited by existing cost-sensitive methods such as [14], [17], and [19].

4.3 Overall Framework

Finally, we aggregate the above-mentioned loss functions as the total objective of our proposed method, which is:

$$\mathcal{L}_{\mathrm{WConPU}} = \mathcal{L}_{\mathrm{class}} + \gamma_0\mathcal{L}_{\mathrm{con}} + \gamma_1\mathcal{L}_{\mathrm{dis}}, \tag{28}$$

where γ_0 and γ_1 are non-negative tunable hyper-parameters controlling the importance of each module. The pseudo-code of our complete algorithm is shown in Algorithm 1

To summarize, two key components of our framework (*i.e.*, contrastive learning and classifier training) work in a collaborative fashion. Contrastive learning benefits from the classifier training in constructing precise positive peer sets, which play an essential role in generating discriminative data representations. Such good representations in turn aid classifier training in obtaining accurate pseudo labels, which are critical in training a well-performed classifier. These two components can benefit from each other in an iterative manner.

Algorithm 1: Pseudo-code of WConPU (one epoch).

Input: Training set X_{PU}, positive class prior π_{P}, classifier f parameterized by θ_f, query network g_q, key network g_k, momentum queue $\mathcal{Q}$, initialized pseudo labels $\boldsymbol{s}_i$ associated with $\boldsymbol{x}_i$ in X_{PU}, initialized class prototypes $\boldsymbol{\mu}_c$ ($c \in \{0,1\}$), trade-off parameters γ_0, γ_1, α, and λ.
Output: The optimal parameters θ_f^* for the classifier f
for $iter = 1, 2, \ldots,$ **do**
  Sample a mini-batch B from X_{PU}
  `// query and key embeddings generation`
  $\mathcal{B}_q = \{\boldsymbol{q}_i = g_q(\mathrm{Aug}_q(\boldsymbol{x}_i)) | \boldsymbol{x}_i \in B\}$
  $\mathcal{B}_k = \{\boldsymbol{k}_i = g_k(\mathrm{Aug}_k(\boldsymbol{x}_i)) | \boldsymbol{x}_i \in B\}$
  $\mathcal{A} = \mathcal{B}_q \cup \mathcal{B}_k \cup \mathcal{Q}$
  `// contrastive learning and classifier training`
  for $\boldsymbol{x}_i \in B$ **do**
    Update prototype $\boldsymbol{\mu}_c$ with Eq. (23)
    Generate positive peer set of $\boldsymbol{x}_i$ with Eq. (12)
    Implement hard negative mining with Eq. (14)
    Update pseudo labels $\boldsymbol{s}_i$ with Eq. (21)
  end
  `// network updating`
  Minimize loss $\mathcal{L}_{\mathrm{WConPU}} = \mathcal{L}_{\mathrm{class}} + \gamma_0\mathcal{L}_{\mathrm{con}} + \gamma_1\mathcal{L}_{\mathrm{dis}}$
  Update g_k in a momentum manner with Eq. (6)
end

5 Model Justification

Inspired by PiCO [47], we present justification on why our WConPU can provide precise supervisory information for the PU classifier from the perspective of EM algorithm. Since the ground-truth labels of labeled positive examples are directly available in PU learning, here we primarily concentrate on the unlabeled examples in the following analysis.

For the simplicity of analysis, we consider that in each iteration, all unlabeled examples are involved, namely $\mathcal{A} = \{g_q(\boldsymbol{x}_i) | \boldsymbol{x}_i \in X_{\mathrm{U}}\}$. In the remainder of this paper, the subscript i associated with $\boldsymbol{x}$ is omitted to simplify the mathematics if no confusion will be incurred. Then, the contrastive loss in Eq. (18) for unlabeled examples will be transformed to:

$$\begin{aligned} \hat{\mathcal{L}}_{\mathrm{con}} =& \frac{1}{n_{\mathrm{U}}}\sum_{\boldsymbol{x}\in X_{\mathrm{U}}}\left\{-\frac{1}{|\mathcal{P}(\boldsymbol{x})|}\sum_{\boldsymbol{k}_+\in\mathcal{P}(\boldsymbol{x})}\log\frac{\exp(\boldsymbol{q}^\top\boldsymbol{k}_+/\rho)}{\sum_{\boldsymbol{k}'\in\mathcal{A}(\boldsymbol{x})}\exp(\boldsymbol{q}^\top\boldsymbol{k}'/\rho)}\right\} \\ =& \frac{1}{n_{\mathrm{U}}}\sum_{\boldsymbol{x}\in X_{\mathrm{U}}}\left\{-\frac{1}{|\mathcal{P}(\boldsymbol{x})|}\sum_{\boldsymbol{k}_+\in\mathcal{P}(\boldsymbol{x})}(\boldsymbol{q}^\top\boldsymbol{k}_+/\rho)\right\} \\ &+ \frac{1}{n_{\mathrm{U}}}\sum_{\boldsymbol{x}\in X_{\mathrm{U}}}\left\{\log\sum_{\boldsymbol{k}'\in\mathcal{A}(\boldsymbol{x})}\exp(\boldsymbol{q}^\top\boldsymbol{k}'/\rho)\right\} \\ =& \mathcal{L}_{\mathrm{alignment}} + \mathcal{L}_{\mathrm{uniform}}, \end{aligned} \tag{29}$$

where $\mathcal{P}(\boldsymbol{x})$ corresponds to the positive peer set of example $\boldsymbol{x}$. The *uniformity* term $\mathcal{L}_{\mathrm{uniform}}$ aims to maximally preserve the information of the data, which has been analyzed in [56]. Here we focus on the analysis of the *alignment* term $\mathcal{L}_{\mathrm{alignment}}$, which encourages the encoder to generate similar features to similar examples [56]. Since we construct the positive peer set of $\boldsymbol{x}$ based on the predicted label $\hat{y} = \arg\max_{j\in\{0,1\}} f^j(\boldsymbol{x})$

in Eq. (12), we first divide X_{U} into two subsets $\mathcal{S}_c \in X_{\mathrm{U}}$ with $c = 0, 1$, where $\mathcal{S}_c = \{\boldsymbol{x} | \arg\max_{j\in\{0,1\}} f^j(\boldsymbol{x}) = c, \boldsymbol{x} \in X_{\mathrm{U}}\}$. To be more specific, each subset $\mathcal{S}_c$ contains the examples with the same predicted label, resulting in the reformulation of the term $\mathcal{L}_{\text{alignment}}$, which is:

$$\begin{aligned}\mathcal{L}_{\text{alignment}} &= \frac{1}{n_{\mathrm{U}}} \sum_{\boldsymbol{x}\in X_{\mathrm{U}}} \frac{1}{|\mathcal{P}(\boldsymbol{x})|} \sum_{\boldsymbol{k}_+\in\mathcal{P}(\boldsymbol{x})} (||\boldsymbol{q} - \boldsymbol{k}_+||^2 - 2)/(2\rho) \\ &= \frac{1}{2\rho n_{\mathrm{U}}} \sum_{\mathcal{S}_c\in X_{\mathrm{U}}} \frac{1}{|\mathcal{S}_c|} \sum_{\boldsymbol{x},\boldsymbol{x}'\in\mathcal{S}_c} ||g_q(\boldsymbol{x}) - g_q(\boldsymbol{x}')||^2 + K \\ &= \frac{1}{\rho n_{\mathrm{U}}} \sum_{\mathcal{S}_c\in X_{\mathrm{U}}} \sum_{\boldsymbol{x}\in\mathcal{S}_c} ||g_q(\boldsymbol{x}) - \hat{\boldsymbol{\mu}}_c||^2 + K,\end{aligned} \tag{30}$$

where K is a constant and $\hat{\boldsymbol{\mu}}_c = \frac{1}{|\mathcal{S}_c|}\sum_{\boldsymbol{x}\in\mathcal{S}_c} g_q(\boldsymbol{x})$ represents the mean center of $\mathcal{S}_c$. The derivation of Eq. (30) is provided in Appendix A.1.

Now we are ready to justify our WConPU from the perspective of EM algorithm. In E-step, the classifier assigns each example to one specific cluster. In M-step, minimizing Eq. (30) enforces the learned representations towards their cluster center. The detailed interpretation is provided below:

E-Step. Given an encoder $g_q(\cdot)$ parameterized by θ_q, our goal is to find the network parameters θ_q that maximizes the log-likelihood function below:

$$\begin{aligned}\theta_q^* &= \arg\max_{\theta_q} \sum_{\boldsymbol{x}\in X_{\mathrm{U}}} \log p(\boldsymbol{x}|\theta_q) \\ &= \operatorname{argmax}_{\theta_q} \sum_{\boldsymbol{x}\in X_{\mathrm{U}}} \log \sum_{y=0}^{1} p(\boldsymbol{x}, y|\theta_q) \\ &= \operatorname{argmax}_{\theta_q} \sum_{\boldsymbol{x}\in X_{\mathrm{U}}} \log \sum_{y=0}^{1} Q(y) \frac{p(\boldsymbol{x}, y|\theta_q)}{Q(y)} \\ &\geq \operatorname{argmax}_{\theta_q} \sum_{\boldsymbol{x}\in X_{\mathrm{U}}} \sum_{y=0}^{1} Q(y) \log \frac{p(\boldsymbol{x}, y|\theta_q)}{Q(y)},\end{aligned} \tag{31}$$

where $Q(y)$ denotes some distribution over all unlabeled examples ($y \in \{0, 1\}$), and $\sum_{y=0}^{1} Q(y) = 1$. The last step of derivation uses Jensen's inequality. To make the inequality hold with equality, we require $\frac{p(\boldsymbol{x},y|\theta_q)}{Q(y)}$ to be a constant. Therefore, we have:

$$Q(y) = \frac{p(\boldsymbol{x}, y|\theta_q)}{\sum_{y=0}^{1} p(\boldsymbol{x}, y|\theta_q)} = \frac{p(\boldsymbol{x}, y|\theta_q)}{p(\boldsymbol{x}|\theta_q)} = p(y|\boldsymbol{x}, \theta_q), \tag{32}$$

which corresponds to the posterior class probability. In WConPU, we estimate the posterior class probability with the output of the classifier. Specifically, given the predicted label $\hat{y} = \arg\max_{j\in\{0,1\}} f^j(\boldsymbol{x})$, we have $Q(y) = \mathbb{1}(\hat{y} = y)$.

M-Step. In M-step, our goal is to maximize the likelihood stated in Eq. (31) under the assumption that the posterior class probability is already known. We will show that under some mild assumptions, minimizing Eq. (30) is equivalent to maximizing a lower bound of likelihood expressed in Eq. (31), and the corresponding theorem is:

Theorem 1. *Assume data from the same class in the representation space follow a d-variate von Mises-Fisher (vMF) distribution. The probabilistic density of this distribution is given by* $f(\boldsymbol{x}|\bar{\boldsymbol{\mu}}_c, \kappa) = c_d(\kappa)e^{\kappa\bar{\boldsymbol{\mu}}_c^\top g_q(\boldsymbol{x})}$*, where* $\bar{\boldsymbol{\mu}}_c = \hat{\boldsymbol{\mu}}_c/||\hat{\boldsymbol{\mu}}_c||$ *is the mean direction,* κ *is the concentration parameter, and* $c_d(\kappa)$ *is the normalization factor. We further assume a uniform class prior* $p(y = 0) = p(y = 1) = 1/2$*. Denoting* $n_c = |\mathcal{S}_c|$ $(c = 0, 1)$ *as the size of each subset, minimizing Eq.* (30) *and maximizing the likelihood in Eq.* (31) *is equivalent to maximizing* R_1 *and* R_2 *below correspondingly, which are:*

$$\begin{aligned}R_1 &= \sum_{\mathcal{S}_c\in X_{\mathrm{U}}} \frac{n_c}{n_{\mathrm{U}}} ||\hat{\boldsymbol{\mu}}_c||^2, \\ R_2 &= \sum_{\mathcal{S}_c\in X_{\mathrm{U}}} \frac{n_c}{n_{\mathrm{U}}} ||\hat{\boldsymbol{\mu}}_c||,\end{aligned} \tag{33}$$

with $R_1 \leq R_2$ *strictly holds.*

The proof of Theorem 1 is provided in Appendix A.2. Theorem 1 indicates that minimizing the alignment term in Eq. (30) is equivalent to maximizing a lower bound of likelihood in Eq. (31). The lower bound is tight when $||\hat{\boldsymbol{\mu}}_c||$ is close to 1, which implies that the examples with the same predicted label (*i.e.*, $\boldsymbol{x} \in \mathcal{S}_c$) are strongly concentrated in the representation space. By examining Eq. (33), we observe that the contrastive loss encourages the norm of $\hat{\boldsymbol{\mu}}_c$ to be large. Besides, as shown in Fig. 4, our WConPU is indeed capable of generating compact clusters. Hence, minimizing the contrastive loss is equivalent to maximizing the likelihood in Eq. (31).

Since our WConPU can be understood from the perspective of EM algorithm, the contrastive learning module and classifier training module in our WConPU can benefit from each other in an iterative manner, and thus allowing our WConPU to converge to a (local) optima.

6 Experiment

To demonstrate the effectiveness of our proposed WConPU, in this section, we perform exhaustive experiments on several publicly available benchmark datasets and real-world datasets.

6.1 Experimental Settings

Datasets. We conduct experiments on four popular benchmark datasets including CIFAR-10, SVHN, STL-10, and Alzheimer datasets. More details of the datasets are provided below as well as in Table I.

- **CIFAR-10** dataset [57] consists of colored images in 10 different classes including "airplane", "automobile", "bird", etc. In our experiments, we take the images of vehicles (*i.e.*, "airplanes", "automobiles", "ships", "trucks") as the positive class and treat the images of animals (*i.e.*, "birds", "cats", "deer", "dogs", "frogs", "horses") as the negative class.
- **SVHN** dataset [20] is a collection of colored images of street view house numbers. The even numbers (*i.e.*, "0", "2", "4", "6", "8") are regarded as positive class and the odd numbers (*i.e.*, "1", "3", "5", "7", "9") are regarded as negative class.
- **STL-10** dataset [58] collects 10 classes of colored images, such as "airplanes", "birds", and "cars". We build

TABLE I
SUMMARY OF EMPLOYED DATASETS.

Dataset	Input Image Size	n_{P}	n_{U}	# Testing Data	π_{P}	Positive Class	Backbone
CIFAR-10	$3 \times 32 \times 32$	1,000	50,000	10,000	0.4	Vehicles (*i.e.*, Class ID: 0, 1, 8, 9)	13-layer CNN
SVHN	$3 \times 32 \times 32$	1,000	73,257	26,032	0.46	Even (*i.e.*, Class ID: 0, 2, 4, 6, 8)	13-layer CNN
STL-10	$3 \times 96 \times 96$	500	5,000	8,000	0.4	Vehicles (*i.e.*, Class ID: 1, 3, 9, 10)	ResNet-18 [59]
Alzheimer	$3 \times 224 \times 224$	769	5,121	1,279	0.5	Alzheimer's Disease	ResNet-50 [59]

a PU dataset by treating the vehicles (*i.e.*, "airplanes", "cars", "ships", "trucks") as the positive class, and the animals (*i.e.*, "birds", "cats", "deer", "dogs", "horses", "monkeys") as the negative class.

- **Alzheimer**[1] dataset contains the MRI images for identifying the Alzheimer's Disease. The MRI images of demented patients are recognized as positive class and the MRI images of healthy people are recognized as negative class.

Baseline Methods. We choose eleven state-of-the-art PU learning algorithms for comparison, which are listed as follows:

- **uPU** [13] designs an unbiased risk estimator for PU learning.
- **nnPU** [14] overcomes the overfitting problem by forcing the risk estimator of negative class to be non-negative.
- **RP** [60] ranks the training data by confidence and selects the most confident examples as positive or negative.
- **PUSB** [19] proposes a threshold estimation algorithm to deal with the selection bias during the labeling process.
- **PUbN** [23] first pretrains a model with nnPU algorithm to classify some reliable positive data, negative data, and unlabeled data, and then minimizes a risk approximated by the above three partitions.
- **Self-PU** [9] employs several self-supervised techniques to extend the learning capability of the previous PU model.
- **aPU** [61] deals with the arbitrary positive shift between source and target distributions.
- **VPU** [62] introduces a variational principle to relax the requirement of class prior.
- **ImbPU** [17] redesigns the nnPU estimator to enable the learning from imbalanced data.
- **Dist-PU** [18] forces the distribution of predicted labels to align with that of ground-truth ones.
- **puNCE** [46] first pretrains the encoder using a contrastive loss, and then do linear probing with nnPU algorithm.
- **PiCO** [47] introduces a prototypical label disambiguation algorithm for addressing the partial label learning (PLL) problem. Note that in our experiments, we incorporate $\mathcal{L}_{\mathrm{dis}}$ into PiCO to make it adaptable to PU conditions.

Evaluation Metrics. For each compared method, we report the results on the test set in terms of six metrics for a comprehensive evaluation, including accuracy (ACC), Precision (Prec.), Recall (Rec.), F1 measure (F1), Area Under ROC Curve (AUC), and Average Precision (AP). On each dataset, every algorithm was independently implemented five times with different labeled positive examples, and the mean and standard deviation of the results on each metric are recorded. Besides, the paired t-test with 95% confidence level is also conducted to see whether our method is statistically better than the comparators.

Implementation Details. Our proposed WConPU method was implemented via PyTorch on a Geforce RTX 3090Ti GPU. The backbone networks of comparators on every dataset are summarized in Table I. For our WConPU, by following [30], [45], the projection head after the contrastive encoder is a 2-layer multilayer perceptron (MLP) of which the output is an 128-dimensional embedding. Since the query view and key view requires weak and strong augmentation correspondingly, we respectively employ SimAugment [45] and RandAugment [63] for such augmentations. For the size of the queue storing key embeddings, it is set to 4096 for Alzheimer dataset, while it is 8192 for other datasets. For pseudo label updating, we set α to 0.9. The momentum coefficient λ is set to 0.999 and the temperature parameter ρ is set to 0.07. We use a standard SGD optimizer with a momentum of 0.9 as the optimizer with a cosine annealing scheduler, where the initial learning rate is set as 1×10^{-2}. The batch size is 8 for Alzheimer dataset, and 256 for other datasets. By following [23], [27], [64], an additional clean validation set (10% of the training examples) is sampled for trade-off parameter selection. Note that the clean validation set does not contribute to the final training process. The trade-off parameters γ_0 and γ_1 are properly selected from $\{1\times10^{-3}, 1\times10^{-2}, 1\times10^{-1}, 1\}$ via grid search, respectively. We do not employ the early stopping technique but instead report the results of the 800th epoch in iterative training of all compared methods over all datasets.

6.2 Comparison with State-of-the-art Methods

In this section, we evaluate the classification performance of the proposed method by comparing it with the baseline methods mentioned in Section 6.1. The performance of all baseline methods and our WConPU on the adopted four benchmark datasets is summarized in Table II. According to the results, we have the following observations:

- On all four datasets, our proposed WConPU outperforms all competitors by a noticeable margin in terms of most metrics. In particular, we improve the best baseline method by 3.10% on SVHN dataset. The comparison result further validates the effectiveness of our proposed method.
- The standard deviation of the results of our WConPU across different trials is relatively small when compared with most of the other methods, so our method is robust

[1]Dubey, S. Alzheimer's Dataset. Available online: https://www.kaggle.com/tourist55/alzheimers-dataset-4-class-of-images

TABLE II
COMPARISON RESULTS OF VARIOUS METHODS ON CIFAR-10, SVHN, STL-10, AND ALZHEIMER DATASETS. THE BEST AND SECOND BEST VALUES ARE BOTH HIGHLIGHTED. ✓DENOTES THAT OUR WCONPU IS SIGNIFICANTLY BETTER THAN THE CORRESPONDING METHODS REVEALED BY THE PAIRED T-TEST WITH CONFIDENCE LEVEL 95%.

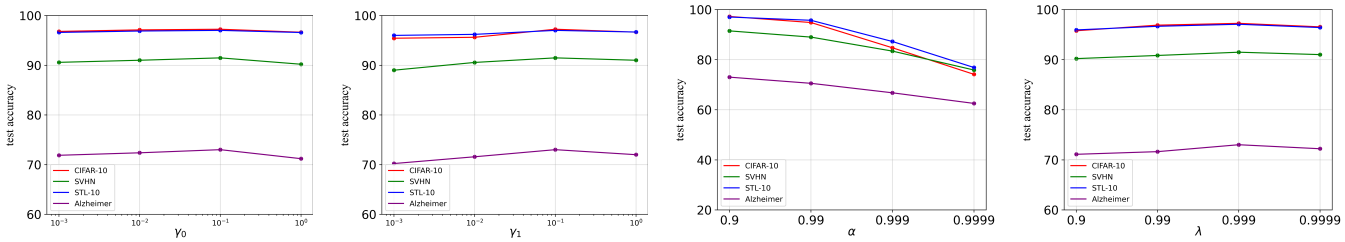
Dataset	Method	ACC (%)	Prec. (%)	Rec. (%)	F1 (%)	AUC (%)	AP (%)
CIFAR-10	uPU [13]	88.41±0.41 ✓	87.21±2.09 ✓	83.02±1.98 ✓	85.12±0.43 ✓	94.98±0.62 ✓	92.71±1.08 ✓
	nnPU [14]	88.91±0.43 ✓	86.21±1.02 ✓	86.03±1.22 ✓	86.11±0.49 ✓	95.13±0.55 ✓	92.51±1.32 ✓
	RP [60]	88.74±0.16 ✓	86.02±1.05 ✓	85.72±1.61 ✓	85.93±0.30 ✓	95.21±0.23 ✓	93.01±0.61 ✓
	PUSB [19]	88.97±0.39 ✓	86.15±0.56 ✓	86.22±0.45 ✓	86.18±0.51 ✓	95.15±0.50 ✓	92.44±1.34 ✓
	PUbN [23]	89.83±0.30 ✓	87.85±0.98 ✓	86.56±1.87 ✓	87.18±0.54 ✓	94.44±0.35 ✓	91.28±1.11 ✓
	Self-PU [9]	89.31±0.56 ✓	86.26±0.76 ✓	87.22±2.16 ✓	86.77±1.12 ✓	95.52±0.46 ✓	93.31±1.02 ✓
	aPU [61]	89.09±0.44 ✓	86.31±1.33 ✓	86.33±0.71 ✓	86.41±0.41 ✓	95.11±0.39 ✓	92.42±1.22 ✓
	VPU [62]	87.89±0.56 ✓	86.71±1.46 ✓	82.88±2.93 ✓	84.42±1.12 ✓	94.55±0.46 ✓	92.02±0.69 ✓
	ImbPU [17]	89.43±0.42 ✓	86.72±0.89 ✓	86.91±0.78 ✓	86.77±0.56 ✓	95.53±0.26 ✓	93.33±0.69 ✓
	Dist-PU [18]	91.88±0.52 ✓	89.87±1.09 ✓	89.84±0.81 ✓	89.85±0.62 ✓	96.92±0.45 ✓	95.49±0.72 ✓
	puNCE [46]	95.34±0.24 ✓	95.11±0.88 ✓	93.43±0.45 ✓	94.21±0.44 ✓	98.59±0.53 ✓	98.45±0.63
	PiCO [47]	95.64±0.12 ✓	94.89±0.76 ✓	93.97±0.47 ✓	94.75±0.49 ✓	98.67±0.44 ✓	98.22±0.91
	WConPU	97.22±0.15	96.87±0.54	96.02±0.32	96.43±0.29	99.49±0.22	99.25±0.34
SVHN	uPU [13]	83.35±0.45 ✓	87.11±2.39 ✓	75.93±2.68 ✓	81.12±0.56 ✓	91.93±0.62 ✓	90.22±1.11 ✓
	nnPU [14]	83.88±0.45 ✓	86.78±1.15 ✓	77.25±1.42 ✓	82.01±0.58 ✓	92.02±0.52 ✓	90.28±1.38 ✓
	RP [60]	81.73±0.15 ✓	84.01±1.01 ✓	76.12±1.51 ✓	80.10±0.32 ✓	89.75±0.23 ✓	87.99±0.56 ✓
	PUSB [19]	83.99±0.41 ✓	86.81±0.51 ✓	78.01±0.51 ✓	82.11±0.51 ✓	91.89±0.52 ✓	90.31±1.34 ✓
	PUbN [23]	84.89±0.30 ✓	88.26±0.98 ✓	83.57±1.87 ✓	83.16±0.54 ✓	92.03±0.35 ✓	91.89±1.11 ✓
	Self-PU [9]	84.12±0.72 ✓	86.16±0.78 ✓	79.22±2.35 ✓	82.55±1.06 ✓	91.73±0.58 ✓	90.99±1.01 ✓
	aPU [61]	84.01±0.52 ✓	86.29±1.30 ✓	81.21±0.79 ✓	82.33±0.56 ✓	91.56±0.42 ✓	90.66±1.23 ✓
	VPU [62]	76.89±0.48 ✓	79.56±1.41 ✓	75.36±2.84 ✓	73.31±0.91 ✓	85.78±0.41 ✓	83.35±0.73 ✓
	ImbPU [17]	84.20±0.46 ✓	86.69±0.87 ✓	81.18±0.82 ✓	82.99±0.56 ✓	91.79±0.27 ✓	91.21±0.45 ✓
	Dist-PU [18]	85.96±0.33 ✓	89.06±0.89 ✓	84.36±0.76 ✓	83.66±0.56 ✓	92.92±0.49 ✓	92.29±0.88 ✓
	puNCE [46]	88.39±0.35 ✓	90.35±0.92 ✓	83.81±1.99 ✓	87.01±0.55 ✓	94.87±0.35 ✓	93.87±0.92 ✓
	PiCO [47]	89.01±0.34 ✓	90.47±0.79 ✓	85.74±0.64 ✓	87.51±0.44 ✓	95.58±0.54 ✓	94.32±0.63 ✓
	WConPU	91.49±0.29	93.77±0.67	87.54±0.71	90.45±0.35	96.97±0.59	96.82±0.37
STL-10	uPU [13]	93.13±0.42 ✓	90.42±1.08 ✓	92.62±1.28 ✓	91.51±0.62 ✓	97.95±0.56 ✓	97.26±1.21 ✓
	nnPU [14]	93.38±0.42 ✓	91.20±1.01 ✓	92.34±1.03 ✓	91.77±0.58 ✓	97.69±0.51 ✓	95.99±1.18 ✓
	RP [60]	92.88±0.56 ✓	92.87±1.35 ✓	89.18±1.88 ✓	90.97±0.45 ✓	92.15±0.18 ✓	95.58±2.29 ✓
	PUSB [19]	93.65±0.16 ✓	92.06±0.52 ✓	92.06±0.42 ✓	92.06±0.33 ✓	98.06±0.52 ✓	97.21±1.13 ✓
	PUbN [23]	94.01±0.31 ✓	93.01±0.98 ✓	93.11±1.01 ✓	92.98±0.54 ✓	98.20±0.35 ✓	97.66±1.47 ✓
	Self-PU [9]	93.73±0.28 ✓	92.12±1.01 ✓	92.61±1.82 ✓	92.22±1.09 ✓	91.98±0.22 ✓	96.88±1.12 ✓
	aPU [61]	93.41±0.45 ✓	91.15±1.24 ✓	92.55±0.83 ✓	91.52±0.88 ✓	97.85±0.66 ✓	96.23±1.03 ✓
	VPU [62]	91.51±0.65 ✓	93.01±0.66 ✓	84.00±2.56 ✓	88.28±1.52 ✓	96.40±0.82 ✓	95.53±0.73 ✓
	ImbPU [17]	93.88±0.81 ✓	92.25±1.12 ✓	91.66±0.83 ✓	92.01±0.54 ✓	97.98±0.72 ✓	97.33±1.02 ✓
	Dist-PU [18]	94.73±0.31 ✓	93.35±1.01 ✓	93.47±0.81 ✓	93.41±0.41 ✓	98.54±0.71	97.96±1.01 ✓
	puNCE [46]	95.13±0.22 ✓	94.09±0.55 ✓	94.95±0.82 ✓	94.51±0.51 ✓	98.66±0.24 ✓	98.23±0.69 ✓
	PiCO [47]	95.55±0.23 ✓	94.36±0.42 ✓	95.12±0.81 ✓	94.75±0.44 ✓	98.78±0.15 ✓	98.55±0.34
	WConPU	97.02±0.21	95.53±0.41	97.42±0.91	96.35±0.26	99.58±0.12	99.46±0.21
Alzheimer	uPU [13]	68.42±2.22 ✓	69.71±3.44	67.33±5.18 ✓	68.63±1.73 ✓	73.99±2.72 ✓	70.12±2.98
	nnPU [14]	68.21±2.15 ✓	68.09±2.21 ✓	71.01±5.88 ✓	68.11±2.99 ✓	71.99±3.01 ✓	70.01±2.21 ✓
	RP [60]	62.03±2.85 ✓	63.11±3.77 ✓	66.23±9.86 ✓	61.99±6.03 ✓	66.32±2.99 ✓	64.10±2.11 ✓
	PUSB [19]	69.19±2.41 ✓	70.11±1.88	69.43±2.13 ✓	69.41±2.15 ✓	74.66±2.42 ✓	70.12±1.64 ✓
	PUbN [23]	70.00±1.02 ✓	69.43±2.25	74.22±6.01 ✓	71.18±2.89 ✓	74.98±0.89 ✓	69.66±1.63 ✓
	Self-PU [9]	70.79±0.73 ✓	69.55±2.51	75.51±4.99 ✓	72.10±1.02 ✓	75.85±1.68 ✓	71.79±3.63
	aPU [61]	68.41±1.71 ✓	66.23±0.88 ✓	75.71±6.21 ✓	71.01±3.06	73.66±2.44 ✓	70.23±3.33
	VPU [62]	66.51±0.61 ✓	64.89±1.01 ✓	75.18±3.71 ✓	71.01±0.98 ✓	72.99±0.91 ✓	71.21±0.65 ✓
	ImbPU [17]	68.19±0.69 ✓	67.34±2.31 ✓	71.24±6.21 ✓	68.79±1.81 ✓	73.69±0.75 ✓	70.56±0.97 ✓
	Dist-PU [18]	71.57±0.62 ✓	68.48±1.16 ✓	80.09±5.10	73.74±1.64	77.13±0.69	73.33±1.47
	puNCE [46]	70.59±0.77 ✓	68.99±1.56 ✓	75.99±6.11 ✓	71.55±1.11 ✓	75.55±1.03 ✓	71.23±1.66
	PiCO [47]	71.94±0.71 ✓	69.59±1.12	79.01±5.03	73.92±1.02	77.59±0.78	72.17±0.97
	WConPU	73.02±0.66	70.87±2.42	79.12±4.99	74.23±0.76	78.55±1.07	72.66±1.07

to different selections of initial labeled positive examples in X_{L}.

- The PU algorithms based on contrastive learning (*i.e.*, our WConPU and puNCE) exhibit superior performance when compared with the methods that do not incorporate contrastive learning, such as nnPU, Self-PU, and Dist-PU. This supports our argument that utilizing high-quality representations is beneficial for training a well-performed PU classifier.
- Although both PiCO and our WConPU employ a joint optimization approach combining contrastive learning with classification, our WConPU significantly outper-

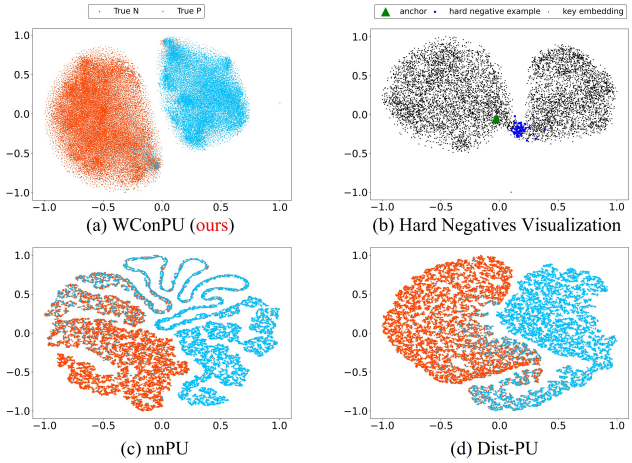
TABLE III
ABLATION STUDY OF WCONPU ON CIFAR-10 WITH $n_{\mathrm{P}} = 1k$.

Variant	Setting	ACC (%)	Prec. (%)	Rec. (%)	F1 (%)	AUC (%)	AP (%)
I	WConPU w/o $\mathcal{L}_{\mathrm{con}}$	82.15±2.25	90.82±1.99	61.60±6.11	73.41±1.73	92.55±2.31	88.86±1.01
II	WConPU w/o pseudo label updating	90.03±1.01	90.33±1.33	85.01±1.33	87.01±1.06	95.45±0.99	93.33±0.87
III	WConPU w/o $\mathcal{L}_{\mathrm{dis}}$	94.23±1.02	95.85±1.21	89.03±0.76	92.33±0.72	98.51±0.56	97.19±0.83
IV	WConPU w/o hard negative mining	95.87±0.81	95.91±0.45	93.21±0.87	94.76±1.35	98.93±0.64	98.65±0.45
V	WConPU	97.22±0.15	96.87±0.54	96.02±0.32	96.43±0.29	99.49±0.22	99.25±0.34



(a) (b) (c) (d)

Fig. 3. Parametric sensitivity analysis on CIFAR-10, SVHN, STL-10, and Alzheimer datasets. (a), (b), (c), and (d) show the impact of different values of γ_0, γ_1, α, and λ on test accuracy, respectively.



(a) WConPU (ours) (b) Hard Negatives Visualization

(c) nnPU (d) Dist-PU

Fig. 4. t-SNE [65] visualization of learned representations of nnPU, Dist-PU and our WConPU on the training data of CIFAR-10, where "Vehicles" are deemed as positive class. Compared with nnPU and Dist-PU, the proposed WConPU significantly improves the embedding quality. Meanwhile, the subfigure (b) shows the t-SNE visualization of hard negative examples.

forms PiCO on all four datasets in terms of most metrics. This superior performance can be attributed to our methodological innovations such as prototype-based hard negative mining module and weighted contrastive loss, which enable our WConPU algorithm to leverage contrastive learning in a more effective way than PiCO.

The main reason for the above merits is that our algorithm can generate high-quality representations, which guarantees the performance stability and superiority of the induced PU classifier when compared with other popular PU learning methodologies.

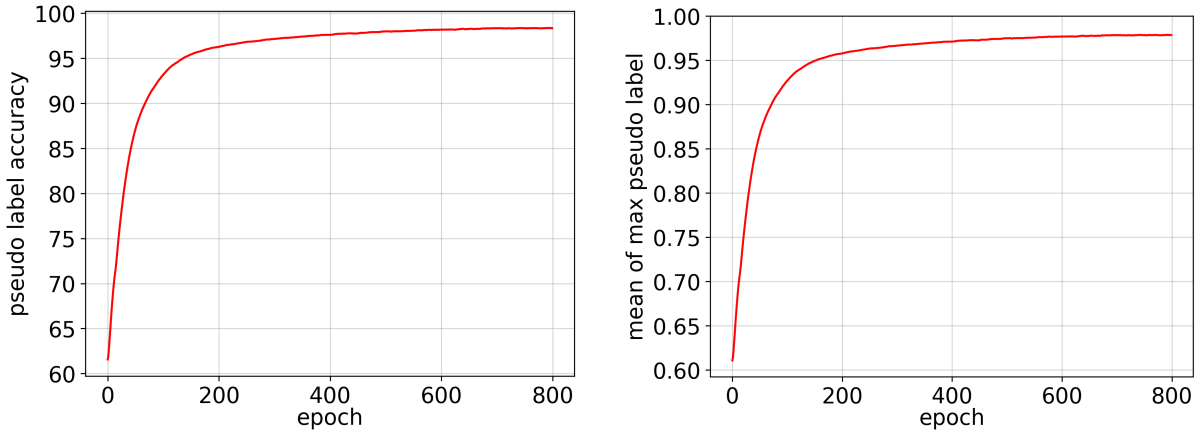
6.3 Ablation Studies

In order to analyze the impact of each module in WConPU, *i.e.*, weighted contrastive loss, pseudo label updating, prototype-based hard negative mining, and label distribution alignment, we conduct ablation studies on CIFAR-10 and investigate the algorithmic performance. The results are shown in Table III.

Effect of Contrastive Learning. From the Table III, we can see that the contrastive learning module contributes significantly to WConPU. To be specific, without $\mathcal{L}_{\mathrm{con}}$ (see Variant I), the performance of WConPU decreases dramatically. The reason is that without CL, we cannot obtain high-quality representations, and the prototype-based pseudo label updating module also cannot work properly.

Effect of Prototype-based Pseudo Label Updating and $\mathcal{L}_{\mathrm{dis}}$. In Variant II, we make the pseudo labels unchanged during the training process. Without the pseudo label updating, the supervisory information for classifier training is weak, so the model predictions on unlabeled data will show high-entropy and low-confidence. In Variant III, we treat all unlabeled examples as negative, and then use the cross entropy loss to replace $\mathcal{L}_{\mathrm{dis}}$. Without $\mathcal{L}_{\mathrm{dis}}$, some positive examples would be wrongly deemed as negative, which impairs the performance of WConPU.

Effect of Hard Negative Mining. In Variant IV, we remove the hard negative mining module from our WConPU model, and our weighted contrastive objective function would degenerate to traditional SCL loss function. From Table III, we can see that there is a noticeable decline in the performance of WConPU without hard negative mining module. The reason is that without this module, the model cannot separate positive and negative examples well in the embedding space, and this will bring challenges to the training of a well-performed classifier. For better understanding the impact of hard negative mining on further improving the representation quality, we show the t-SNE [65] visualization result of hard negative examples in Fig. 4(b).



(a) Pseudo label accuracy

(b) Mean of the max value of pseudo labels

Fig. 5. Pseudo label updating process on CIFAR-10 with $n_{\mathrm{P}} = 1k$. The subfigure (a) shows the accuracy curve of pseudo labels during the training process, and the subfigure (b) shows the mean of the max value of pseudo labels, which is calculated by $\frac{1}{n}\sum_{i=1}^{n}\max(\boldsymbol{s}_i)$.

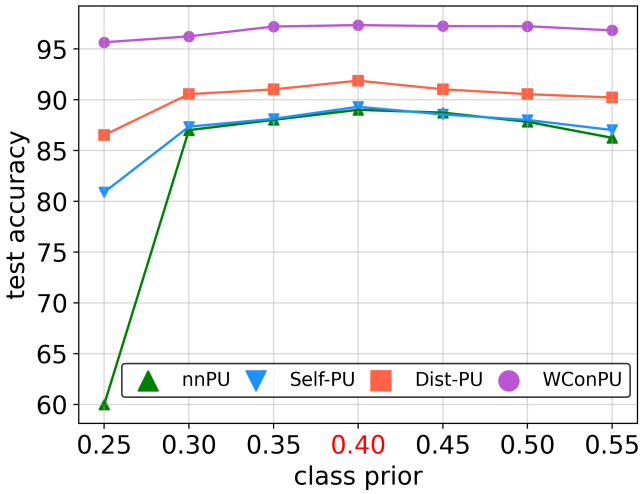


Fig. 6. Sensitivity analysis regarding class prior on CIFAR-10 dataset. The real value of positive class prior π_{P} is 0.4.

6.4 Embedding Visualization

In this part, we compare the t-SNE visualization results of nnPU [14], Dist-PU [18], and our WConPU, as nnPU is a classical PU method, and Dist-PU is the latest state-of-the-art model. The results are shown in Fig. 4. From the figure, we have two observations: 1) As a representative of cost-sensitive methods, nnPU performs poorly in separating positive examples and negative examples in the representation space; and 2) Although Dist-PU solves PU learning problem from a label distribution perspective and can generate relatively better representations than nnPU, it still performs worse than our WConPU. To summarize, with the employment of CL, our WConPU can generate more discriminative representations than other methods, which is the key to the superior performance of our method. Moreover, Fig. 4(b) shows that our prototype-based hard negative mining module can successfully mine the hard negative examples during the training process, and our weighted contrastive loss can leverage the hard negative examples to generate well-separated representations.

6.5 Parametric Sensitivity

In our WConPU model, there are four trade-off parameters γ_0, γ_1, α, and λ that should be pre-tuned manually. In this section, we analyze the parametric sensitivity of our WConPU to these parameters on CIFAR-10, SVHN, STL-10, and Alzheimer datasets. Specifically, we examine the test accuracy by varying one of γ_0, γ_1, α, and λ, and meanwhile fixing the other parameters to a constant value. By changing γ_0 and γ_1 from 0.001 to 1, α and λ from 0.9 to 0.9999, the results on CIFAR-10, SVHN, STL-10, and Alzheimer datasets are shown in Fig. 3.

From the curves presented in Fig. 3, we find that these four parameters are critical for our WConPU to achieving good performance. To be specific, the performance of our WConPU is relatively stable when γ_0, γ_1, and λ vary within a wide range. For γ_1, the performance will slightly drop if it is set within $[0.001, 0.01]$. In contrast, Fig. 3(c) reveals that $\alpha = 0.9$ usually leads to high classification accuracy. Therefore, α is consistently set to 0.9 throughout our experiments.

6.6 Sensitivity to Class Prior

Since our proposed WConPU needs the estimated class prior π_{P} as input, here we study the impact of inaccurate input class prior on our model in Fig. 6. Besides, other three existing methods (*i.e.*, nnPU, Self-PU, and Dist-PU) which also rely on class prior are also incorporated for comparison. From Fig. 6, we can see that even with misspecified class prior, our WConPU still considerably outperforms other competitors which operate under ground-truth class prior. Moreover, the performance of our WConPU is much more stable than other competitors, especially when the prior is underestimated. This further demonstrates the practicality of WConPU. This lower sensitivity to class prior can be attributed to two factors: 1) The side effects brought by misspecified class prior could be offset by the high-quality representations originated from

contrastive learning module; and 2) Our pseudo label updating mechanism works in a moving-average manner, which can prevent our model from being trapped in bad local optima caused by inaccurate class prior.

6.7 Qualitative Analysis of Pseudo Labels

To show the effectiveness of our prototype-based pseudo label updating strategy, we show the pseudo label updating process in Fig. 5. Specifically, Fig. 5(a) shows the accuracy curve of pseudo labels, and Fig. 5(b) shows the mean of the maximal value of pseudo labels, which is calculated by $\frac{1}{n}\sum_{i=1}^{n}\max(\boldsymbol{s}_i)$. From the figure, we can see that the pseudo labels would gradually move towards the ground-truth labels and converge to nearly one-hot vectors during the training procedure. This further confirms that our pseudo label updating mechanism can provide high-quality supervisory information for classifier training.

7 Conclusion

In this paper, we propose a new PU learning algorithm termed "WConPU" which utilizes the contrastively learned prototypes to guide the learning of the classifier. The advantages of WConPU are three-fold: 1) The employment of contrastive learning guarantees the generation of discriminative representations; 2) The novel weighted contrastive objective function equipped with a prototype-based hard negative mining module further improves the quality of representations; and 3) The pseudo label updating strategy takes full advantage of the implicit semantic information of prototypes, which guides the classifier to make confident and precise predictions. Due to the above reasons, our method has shown superior performance to various state-of-the-art PU learning methods on diverse datasets.

References

[1] M. C. du Plessis, G. Niu, and M. Sugiyama, "Convex formulation for learning from positive and unlabeled data," in *International Conference on Machine Learning*, vol. 37, 2015, pp. 1386–1394.

[2] C. Gong, T. Liu, J. Yang, and D. Tao, "Large-margin label-calibrated support vector machines for positive and unlabeled learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3471–3483, 2019.

[3] C. Gong, H. Shi, T. Liu, C. Zhang, J. Yang, and D. Tao, "Loss decomposition and centroid estimation for positive and unlabeled learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 3, pp. 918–932, 2021.

[4] C. Gong, Q. Wang, T. Liu, B. Han, J. You, J. Yang, and D. Tao, "Instance-dependent positive and unlabeled learning with labeling bias estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 8, pp. 4163–4177, 2022.

[5] X. Li and B. Liu, "Learning from positive and unlabeled examples with different data distributions," in *European Conference on Machine Learning*, vol. 3720, 2005, pp. 218–229.

[6] E. Sansone, F. De Natale, and Z. Zhou, "Efficient training for positive unlabeled learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 11, pp. 2584–2598, 2019.

[7] H. Shi, S. Pan, J. Yang, and C. Gong, "Positive and unlabeled learning via loss decomposition and centroid estimation," in *International Joint Conferences on Artificial Intelligence*, 2018, pp. 2689–2695.

[8] S. Zhang, A. Zhu, G. Zhu, Z. Wei, and K. Li, "Building fake review detection model based on sentiment intensity and PU learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 10, pp. 6926–6939, 2023.

[9] X. Chen, W. Chen, T. Chen, Y. Yuan, C. Gong, K. Chen, and Z. Wang, "Self-pu: Self boosted and calibrated positive-unlabeled training," in *International Conference on Machine Learning*, vol. 119, 2020, pp. 1510–1519.

[10] W. Li, Q. Guo, and C. Elkan, "A positive and unlabeled learning algorithm for one-class classification of remote-sensing data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 2, pp. 717–725, 2011.

[11] J. Zhang, Z. Wang, J. Meng, Y. Tan, and J. Yuan, "Boosting positive and unlabeled learning for anomaly detection with multi-features," *IEEE Transactions on Multimedia*, vol. 21, no. 5, pp. 1332–1344, 2019.

[12] J. Zhang, Z. Wang, J. Yuan, and Y. Tan, "Positive and unlabeled learning for anomaly detection with multi-features," in *ACM International Conference on Multimedia*, 2017, pp. 854–862.

[13] M. C. du Plessis, G. Niu, and M. Sugiyama, "Analysis of learning from positive and unlabeled data," in *Advances in Neural Information Processing Systems*, 2014, pp. 703–711.

[14] R. Kiryo, G. Niu, M. C. du Plessis, and M. Sugiyama, "Positive-unlabeled learning with non-negative risk estimator," in *Advances in Neural Information Processing Systems*, 2017, pp. 1675–1685.

[15] B. Liu, W. S. Lee, P. S. Yu, and X. Li, "Partially supervised classification of text documents," in *International Conference on Machine Learning*, 2002, pp. 387–394.

[16] H. Yu, J. Han, and K. C. Chang, "PEBL: web page classification without negative examples," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 1, pp. 70–81, 2004.

[17] G. Su, W. Chen, and M. Xu, "Positive-unlabeled learning from imbalanced data," in *International Joint Conferences on Artificial Intelligence*, 2021, pp. 2995–3001.

[18] Y. Zhao, Q. Xu, Y. Jiang, P. Wen, and Q. Huang, "Dist-pu: Positive-unlabeled learning from a label distribution perspective," in *International Conference on Computer Vision and Pattern Recogintion*, 2022, pp. 14 441–14 450.

[19] M. Kato, T. Teshima, and J. Honda, "Learning from positive and unlabeled data with a selection bias," in *International Conference on Learning Representations*, 2019.

[20] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[21] F. Denis, "PAC learning from positive statistical queries," in *Algorithmic Learning Theory*, vol. 1501, 1998, pp. 112–126.

[22] J. Bekker and J. Davis, "Learning from positive and unlabeled data: a survey," *Machine Learning*, vol. 109, no. 4, pp. 719–760, 2020.

[23] Y. Hsieh, G. Niu, and M. Sugiyama, "Classification from positive, unlabeled and biased negative data," in *International Conference on Machine Learning*, vol. 97, 2019, pp. 2820–2829.

[24] J. C. Carnevali, R. G. Rossi, E. E. Milios, and A. de Andrade Lopes, "A graph-based approach for positive and unlabeled learning," *Information Sciences*, vol. 580, pp. 655–672, 2021.

[25] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[26] M. Hou, B. Chaib-draa, C. Li, and Q. Zhao, "Generative adversarial positive-unlabelled learning," in *International Joint Conference on Artificial Intelligence*, 2018, pp. 2255–2261.

[27] X. Wang, W. Wan, C. Geng, S. Li, and S. Chen, "Beyond myopia: Learning from positive and unlabeled data through holistic predictive trends," in *Advances in Neural Information Processing Systems*, 2023.

[28] Y. Jiang, Q. Xu, Y. Zhao, Z. Yang, P. Wen, X. Cao, and Q. Huang, "Positive-unlabeled learning with label distribution alignment," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 12, pp. 15 345–15 363, 2023.

[29] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," in *International Conference on Machine Learning*, vol. 119, 2020, pp. 1597–1607.

[30] K. He, H. Fan, Y. Wu, S. Xie, and R. B. Girshick, "Momentum contrast for unsupervised visual representation learning," in *International Conference on Computer Vision and Pattern Recogintion*, 2020, pp. 9726–9735.

[31] Y. Liu, Z. Li, S. Pan, C. Gong, C. Zhou, and G. Karypis, "Anomaly detection on attributed networks via contrastive self-supervised learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 6, pp. 2378–2392, 2021.

[32] G. Bai, W. Xi, X. Hong, X. Liu, Y. Yue, and S. Zhao, "Robust and rotation-equivariant contrastive learning," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2023.

[33] G. Wu, J. Jiang, and X. Liu, "A practical contrastive learning framework for single-image super-resolution," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–12, 2023.

[34] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *International Conference on Computer Vision and Pattern Recogintion*, 2006, pp. 1735–1742.

[35] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *arXiv preprint arXiv:2003.04297*, 2020.

[36] X. Chen, S. Xie, and K. He, "An empirical study of training self-supervised vision transformers," in *International Conference on Computer Vision*, 2021, pp. 9620–9629.

[37] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar *et al.*, "Bootstrap your own latent - A new approach to self-supervised learning," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 21 271–21 284.

[38] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 9912–9924.

[39] X. Chen and K. He, "Exploring simple siamese representation learning," in *International Conference on Computer Vision and Pattern Recogintion*, 2021, pp. 15 750–15 758.

[40] N. Saunshi, O. Plevrakis, S. Arora, M. Khodak, and H. Khandeparkar, "A theoretical analysis of contrastive unsupervised representation learning," in *International Conference on Machine Learning*, vol. 97, 2019, pp. 5628–5637.

[41] J. Li, P. Zhou, C. Xiong, and S. C. H. Hoi, "Prototypical contrastive learning of unsupervised representations," in *International Conference on Learning Representations*, 2021.

[42] H. Liu, F. Zhang, X. Zhang, S. Zhao, J. Sun, H. Yu, and X. Zhang, "Label-enhanced prototypical network with contrastive learning for multi-label few-shot aspect category detection," in *Conference on Knowledge Discovery and Data Mining*. ACM, 2022, pp. 1079–1087.

[43] A. Das, Y. Xian, D. Dai, and B. Schiele, "Weakly-supervised domain adaptive semantic segmentation with prototypical contrastive learning," in *International Conference on Computer Vision and Pattern Recognition*. IEEE, 2023, pp. 15 434–15 443.

[44] R. Gupta, A. Roy, C. Christensen, S. Kim, S. Gerard, M. Cincebeaux, A. Divakaran, T. Grindal, and M. Shah, "Class prototypes based contrastive learning for classifying multi-label and fine-grained educational videos," in *International Conference on Computer Vision and Pattern Recognition*. IEEE, 2023, pp. 19 923–19 933.

[45] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 18 661–18 673.

[46] A. Acharya, S. Sanghavi, L. Jing, B. Bhushanam, D. Choudhary, M. Rabbat, and I. Dhillon, "Positive unlabeled contrastive learning," *arXiv preprint arXiv:2206.01206*, 2023.

[47] H. Wang, R. Xiao, Y. Li, L. Feng, G. Niu, G. Chen, and J. Zhao, "Pico: Contrastive label disambiguation for partial label learning," in *International Conference on Learning Representations*, 2022.

[48] T. Cour, B. Sapp, and B. Taskar, "Learning from partial labels," *Journal of Machine Learning Research*, vol. 12, pp. 1501–1536, 2011.

[49] N. Xu, J. Lv, and X. Geng, "Partial label learning via label enhancement," in *AAAI Conference on Artificial Intelligence*, 2019, pp. 5557–5564.

[50] H. Wang, R. Xiao, Y. Li, L. Feng, G. Niu, G. Chen, and J. Zhao, "Pico+: Contrastive label disambiguation for robust partial label learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 5, pp. 3183–3198, 2024.

[51] Y. Wang, H. Chen, Q. Heng, W. Hou, Y. Fan, Z. Wu, J. Wang, M. Savvides, T. Shinozaki, B. Raj, B. Schiele, and X. Xie, "Freematch: Self-adaptive thresholding for semi-supervised learning," in *International Conference on Learning Representations*, 2023.

[52] J. D. Robinson, C. Chuang, S. Sra, and S. Jegelka, "Contrastive learning with hard negative samples," in *International Conference on Learning Representations*, 2021.

[53] L. Xu, J. Lian, W. X. Zhao, M. Gong, L. Shou, D. Jiang, X. Xie, and J.-R. Wen, "Negative sampling for contrastive representation learning: A review," *arXiv preprint arXiv:2206.00212*, 2022.

[54] W. Zhang and K. Stratos, "Understanding hard negatives in noise contrastive estimation," in *North American Chapter of the Association for Computational Linguistics*, 2021, pp. 1090–1101.

[55] S. Zhang, M. Liu, J. Yan, H. Zhang, L. Huang, X. Yang, and P. Lu, "M-mix: Generating hard negatives via multi-sample mixing for contrastive learning," in *ACM Conference on Knowledge Discovery and Data Mining*, 2022, pp. 2461–2470.

[56] T. Wang and P. Isola, "Understanding contrastive representation learning through alignment and uniformity on the hypersphere," in *International Conference on Machine Learning*, vol. 119, 2020, pp. 9929–9939.

[57] A. Krizhevsky, "Learning multiple layers of features from tiny images," University of Toronto, Tech. Rep., 2009.

[58] A. Coates, A. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *Artificial Intelligence and Statistics*, 2011, pp. 215–223.

[59] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *International Conference on Computer Vision and Pattern Recogintion*, 2016, pp. 770–778.

[60] C. G. Northcutt, T. Wu, and I. L. Chuang, "Learning with confident examples: Rank pruning for robust classification with noisy labels," in *Conference on Uncertainty in Artificial Intelligence*, 2017.

[61] Z. Hammoudeh and D. Lowd, "Learning from positive and unlabeled data with arbitrary positive shift," in *Advances in Neural Information Processing Systems*, 2020.

[62] H. Chen, F. Liu, Y. Wang, L. Zhao, and H. Wu, "A variational approach for learning from positive and unlabeled data," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 14 844–14 854.

[63] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le, "Randaugment: Practical automated data augmentation with a reduced search space," in *International Conference on Computer Vision and Pattern Recognition*, 2020, pp. 702–703.

[64] Z. Zhu, L. Wang, P. Zhao, C. Du, W. Zhang, H. Dong, B. Qiao, Q. Lin, S. Rajmohan, and D. Zhang, "Robust positive-unlabeled learning via noise negative sample self-correction," in *Conference on Knowledge Discovery and Data Mining*. ACM, 2023, pp. 3663–3673.

[65] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of Machine Learning Research*, vol. 9, no. 11, 2008.

APPENDIX A
PROOF AND DERIVATION

In this appendix, we provide the proofs and mathematical derivations for some theoretical findings in Section 5.

A.1 Derivation of Eq. (30)

By denoting $n_c = |\mathcal{S}_c|$ $(c = 0, 1)$ as the size of each subset, the derivation of the second equality in Eq. (30) suffices to show that $\frac{1}{2n_c}\sum_{\boldsymbol{x},\boldsymbol{x}'\in\mathcal{S}_c}||g_q(\boldsymbol{x}) - g_q(\boldsymbol{x}')||^2 =$

$\sum_{\boldsymbol{x}\in\mathcal{S}_c}||g_q(\boldsymbol{x})-\hat{\boldsymbol{\mu}}_c||^2$. The detailed derivation is given below:

$$\begin{aligned}
&\frac{1}{2n_c}\sum_{\boldsymbol{x},\boldsymbol{x}'\in\mathcal{S}_c}||g_q(\boldsymbol{x})-g_q(\boldsymbol{x}')||^2\\
=&\frac{1}{2n_c}\sum_{\boldsymbol{x}\in\mathcal{S}_c}\sum_{\boldsymbol{x}'\in\mathcal{S}_c}\left(||g_q(\boldsymbol{x})||^2-2g_q(\boldsymbol{x})^\top g_q(\boldsymbol{x}')+||g_q(\boldsymbol{x}')||^2\right)\\
=&\frac{1}{n_c}\sum_{\boldsymbol{x}\in\mathcal{S}_c}\left(n_c-g_q(\boldsymbol{x})^\top\left(\sum_{\boldsymbol{x}'\in\mathcal{S}_c}g_q(\boldsymbol{x}')\right)\right)\\
=&\frac{1}{n_c}\sum_{\boldsymbol{x}\in\mathcal{S}_c}\left(n_c-g_q(\boldsymbol{x})^\top\left(n_c\hat{\boldsymbol{\mu}}_c\right)\right)\\
=&n_c-\left(\sum_{\boldsymbol{x}\in\mathcal{S}_c}g_q(\boldsymbol{x})\right)^\top\hat{\boldsymbol{\mu}}_c\\
=&n_c(1-||\hat{\boldsymbol{\mu}}_c||^2)\\
=&\left(n_c-2\left(\sum_{\boldsymbol{x}\in\mathcal{S}_c}g_q(\boldsymbol{x})\right)^\top\hat{\boldsymbol{\mu}}_c+n_c||\hat{\boldsymbol{\mu}}_c||^2\right)\\
=&\sum_{\boldsymbol{x}\in\mathcal{S}_c}\left(||g_q(\boldsymbol{x})||^2-2g_q(\boldsymbol{x})^\top\hat{\boldsymbol{\mu}}_c+||\hat{\boldsymbol{\mu}}_c||^2\right)\\
=&\sum_{\boldsymbol{x}\in\mathcal{S}_c}||g_q(\boldsymbol{x})-\hat{\boldsymbol{\mu}}_c||^2.
\end{aligned}\tag{34}$$

A.2 Proof of Theorem 1

By ignoring the constant $-\sum_{\boldsymbol{x}\in X_{\mathrm{U}}}\sum_{y=0}^{1}Q(y)\log Q(y)$ in Eq. (31), maximizing the log-likelihood function in Eq. (31) can be written as:

$$\begin{aligned}
&\max_{\theta_q}\sum_{\boldsymbol{x}\in X_{\mathrm{U}}}\sum_{y=0}^{1}Q(y)\log p(\boldsymbol{x},y|\theta_q)\\
=&\max_{\theta_q}\sum_{\boldsymbol{x}\in X_{\mathrm{U}}}\sum_{y=0}^{1}Q(y)\log p(\boldsymbol{x}|y,\theta_q)p(y)\\
=&\max_{\theta_q}\sum_{\boldsymbol{x}\in X_{\mathrm{U}}}\sum_{y=0}^{1}\mathbb{1}(\hat{y}_i=y)\log p(\boldsymbol{x}|y,\theta_q)\\
=&\max_{\theta_q}\sum_{\mathcal{S}_c\in X_{\mathrm{U}}}\sum_{\boldsymbol{x}\in\mathcal{S}_c}\log p(\boldsymbol{x}|y=c,\theta_q)\\
=&\max_{\theta_q}\sum_{\mathcal{S}_c\in X_{\mathrm{U}}}\sum_{\boldsymbol{x}\in\mathcal{S}_c}\left(\kappa\bar{\boldsymbol{\mu}}_c^\top g_q(\boldsymbol{x})+\log\left(c_d(\kappa)\right)\right)\\
=&\max_{\theta_q}\sum_{\mathcal{S}_c\in X_{\mathrm{U}}}\frac{n_c}{n_{\mathrm{U}}}||\hat{\boldsymbol{\mu}}_c||,
\end{aligned}\tag{35}$$

where $n_c=|\mathcal{S}_c|$.

Besides, minimizing Eq. (30) is equivalent to:

$$\begin{aligned}
&\min_{\theta_q}\sum_{\mathcal{S}_c\in X_{\mathrm{U}}}\sum_{\boldsymbol{x}\in\mathcal{S}_c}||g_q(\boldsymbol{x})-\hat{\boldsymbol{\mu}}_c||^2\\
=&\min_{\theta_q}\sum_{\mathcal{S}_c\in X_{\mathrm{U}}}\sum_{\boldsymbol{x}\in\mathcal{S}_c}(||g_q(\boldsymbol{x})||^2-2g_q(\boldsymbol{x})^\top\hat{\boldsymbol{\mu}}_c+||\hat{\boldsymbol{\mu}}_c||^2)\\
=&\min_{\theta_q}\sum_{\mathcal{S}_c\in X_{\mathrm{U}}}(n_c-n_c||\hat{\boldsymbol{\mu}}_c||^2)\\
=&\max_{\theta_q}\sum_{\mathcal{S}_c\in X_{\mathrm{U}}}\frac{n_c}{n_{\mathrm{U}}}||\hat{\boldsymbol{\mu}}_c||^2.
\end{aligned}\tag{36}$$

Note that all the embeddings are ℓ_2-normalized and thus $\|\hat{\boldsymbol{\mu}}_c\|\in[0,1]$. Therefore, this theorem is proved.

APPENDIX B
MORE EXPERIMENTAL RESULTS

B.1 Different Hard Negative Mining Strategies

TABLE IV
TEST ACCURACY OF DIFFERENT HARD NEGATIVE MINING STRATEGIES ON CIFAR-10, SVHN, AND STL-10 DATASETS.

Dataset	Method	Accuracy
CIFAR-10	WConPU	97.22 ± 0.15
	Pseudo Label-based	96.72 ± 0.19
SVHN	WConPU	91.49 ± 0.29
	Pseudo Label-based	91.11 ± 0.32
STL-10	WConPU	97.02 ± 0.21
	Pseudo Label-based	96.79 ± 0.21

In our WConPU, the hard negative set is constructed based on the similarity between the embedding of each example and the corresponding class prototype. Nevertheless, an alternative approach involves selecting the hard negative examples based on pseudo labels. Therefore, to evaluate the effectiveness of these two distinct hard negative mining strategies, we have made a comprehensive comparison on CIFAR-10, SVHN, and STL-10 datasets. From Table IV, we can see that the test accuracy achieved through pseudo label-based hard negative mining is lower than that of the method proposed by our WConPU. This is because the prototypes exhibit better reliability than predicted labels at the initial stage of training procedure.

B.2 Different Prototype Updating Strategies

TABLE V
TRAINING TIME (MIN/EPOCH) AND TEST ACCURACY OF DIFFERENT PROTOTYPE UPDATING METHODS ON CIFAR-10, SVHN, AND STL-10 DATASETS.

Dataset	Method	Time	Accuracy
CIFAR-10	WConPU	0.45	97.22 ± 0.15
	Re-compute	0.63	97.17 ± 0.18
SVHN	WConPU	0.54	91.49 ± 0.29
	Re-compute	0.75	91.32 ± 0.25
STL-10	WConPU	0.49	97.02 ± 0.21
	Re-compute	0.68	96.98 ± 0.23

An alternative strategy for updating prototypes is to re-compute the prototypes by averaging embeddings of all training examples at the end of each epoch. A comparative analysis is conducted between this technique and the method proposed in our WConPU. From Table V, we can see that our WConPU method not only achieves competitive results with the re-compute method, but also surpasses the re-compute method significantly in terms of efficiency.

Botai Yuan received his bachelor's degree from Huazhong University of Science and Technology (HUST) in 2021. Currently, he is pursuing his Ph.D. degree in Shanghai Jiao Tong University (SJTU) under the supervision of Prof. Jie Yang and Prof. Chen Gong. His research interests mainly lie in Positive and Unlabeled Learning.

Chen Gong (Senior Member, IEEE) received his dual doctoral degree from Shanghai Jiao Tong University (SJTU) and University of Technology Sydney (UTS), respectively. Currently, he is a full professor of Shanghai Jiao Tong University. His research interests mainly include machine learning, data mining, and learning-based vision problems. He has published more than 130 technical papers at prominent journals and conferences such as JMLR, IEEE T-PAMI, IEEE T-NNLS, IEEE T-IP, IEEE T-CYB, IEEE T-CSVT, IEEE T-MM, IEEE T-ITS, ACM T-IST, ICML, NeurIPS, ICLR, CVPR, ICCV, AAAI, IJCAI, ICDM, etc. He serves as the associate editor for IEEE T-CSVT, Neural Networks, and NePL, and also the Area Chair or Senior PC member of several top-tier conferences such as AAAI, IJCAI, ICML, ICLR, ECML-PKDD, AISTATS, ICDM, ACM MM, etc. He won the "Excellent Doctorial Dissertation Award" of Chinese Association for Artificial Intelligence, "Young Elite Scientists Sponsorship Program" of China Association for Science and Technology, "Wu Wen-Jun AI Excellent Youth Scholar Award", and the Scientific Fund for Distinguished Young Scholars of Jiangsu Province. He was also selected as the "Global Top Chinese Young Scholars in AI" released by Baidu, and "World's Top 2% Scientists" released by Stanford University.

Dacheng Tao (Fellow, IEEE) is currently a distinguished University professor in the College of Computing and Data Science with Nanyang Technological University. He mainly applies statistics and mathematics to artificial intelligence and data science, and his research is detailed in one monograph and more than 200 publications in prestigious journals and proceedings, leading conferences, with best paper awards, best student paper awards, and test-of-time awards. His publications have been cited more than 142 K times and he has an h-index 180+ in Google Scholar. He received the 2015 and 2020 Australian Eureka Prize, the 2018 IEEE ICDM Research Contributions Award, and the 2021 IEEE Computer Society McCluskey Technical Achievement Award. He is a Fellow of the Australian Academy of Science, AAAS, ACM, and IEEE.

Jie Yang (Senior Member, IEEE) received a bachelor's degree in Automatic Control in Shanghai Jiao Tong University (SJTU), where a master's degree in Pattern Recognition and Intelligent System was achieved three years later. In 1994, he received Ph.D. at Department of Computer Science, University of Hamburg, Germany. Now he is the Professor and Director of Institute of Image Processing and Pattern recognition in Shanghai Jiao Tong University. He is the principal investigator of more than 30 national and ministry scientific research projects in image processing, pattern recognition, data mining, and artificial intelligence. He has published six books, more than five hundreds of articles in national or international academic journals and conferences. Google citation 26000, H-index 83. Up to now, he has supervised 6 postdoctoral, 48 doctors and 70 masters, awarded six research achievement prizes from ministry of Education, China and Shanghai municipality. He has owned 48 patents. Three Ph.D. dissertation he supervised was evaluated as "National Best Ph.D. Dissertation" in 2009, in 2017, and in 2019. He has been chairman and keynote speaker of more than 10 international conferences.