GradPU: Positive-Unlabeled Learning via Gradient Penalty and Positive Upweighting

Songmin Dai, Xiaoqiang Li*, Yue Zhou, Xichen Ye, Tong Liu,

School of Computer Engineering and Science, Shanghai University, China
{laodar, xqli, yuez, yexichen0930, tong_liu}@shu.edu.cn

Abstract

Positive-unlabeled learning is an essential problem in many real-world applications with only labeled positive and unlabeled data, especially when the negative samples are difficult to identify. Most existing positive-unlabeled learning methods will overfit the positive class to some extent due to the existence of unidentified positive samples. This paper first analyzes the overfitting problem and proposes to bound the generalization errors via Wasserstein distances. Based on that, we develop a simple yet effective positive-unlabeled learning method, GradPU, which consists of two key ingredients: A gradient-based regularizer that penalizes the gradient norms in the interpolated data region, which improves the generalization of positive class; An unnormalized upweighting mechanism that assigns larger weights to those positive samples that are hard, not-well-fitted and less frequently labeled. It enforces the training error of each positive sample to be small and increases the robustness to the labeling bias. We evaluate our proposed GradPU on three datasets: MNIST, FashionMNIST, and CIFAR10. The results demonstrate that GradPU achieves state-of-the-art performance on both unbiased and biased positive labeling scenarios.

Introduction

Positive-Unlabeled Learning (PUL) aims to learn a binary classifier with only labeled positive (P) and unlabeled (U) data, where the U data consist of both unidentified P and unidentified negative (N) data. It naturally arises in many real-world applications where a certain class of data is costly to be annotated or difficult to be identified (Bekker and Davis 2020). Such applications include disease gene identification (Yang et al. 2012), remote sensing data (Li, Guo, and Elkan 2010), recommender systems (Ren, Ji, and Zhang 2014), and semi-supervised anomaly detection (Zhang et al. 2018b). In recent years, it has also been used to improve the performance of object detection (Yang, Liang, and Carin 2020; Guo et al. 2021), contrastive learning (Chuang et al. 2020), and reward learning (Xu and Denil 2021).

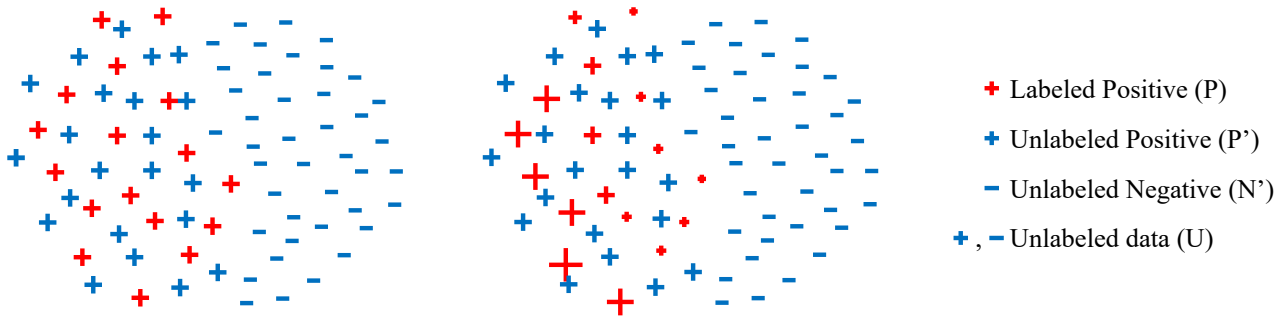
Most previous PUL works can be classified into two categories: sample-selection method (Zhang and Zuo 2009; Luo et al. 2021) and cost-sensitive method (Kiryo et al. 2017; Du Plessis, Niu, and Sugiyama 2015). The sample-selection methods heavily rely on the heuristic algorithm for filtering out or downweighting the positive data in U data (denoted as P′ data) while keeping the negative data in U data (denoted as N′ data). The cost-sensitive methods view the classification risk using all U data as the negative class as a biased estimation of the ideal PN risk, and they correct the bias by canceling out the extra risk caused by P′ data using labeled P data.

Despite these successes, we find the overfitting of P class caused by the conflicting fitting targets of the P and P′ data is not fully solved so far. To address it, we derive more useful generalization error bounds via Wasserstein distances. Based on the theoretical analysis about the generalization error, we propose a gradient-based regularization and an unnormalized positive upweighting mechanism. The gradient-based regularization penalizes the gradient norms in the regions between P and U samples, which leads to a better generalization of the P class. The positive upweighting mechanism assigns larger weights to those positive samples that are hard, not-well-fitted, and less frequently labeled. It forces each labeled P sample to have a small training error although the nearby P′ samples could hinder this. It is often assumed the labeled positive are **S**elected **C**ompletely **A**t **R**andom (SCAR) for most PUL methods (Kiryo et al. 2017; Chen et al. 2020; Bekker and Davis 2018; Hou et al. 2018). However, in many practical situations, the distribution of labeled P data may differ from the P′ due to a selection bias (Kato, Teshima, and Honda 2018; Na et al. 2020; Hammoudeh and Lowd 2020). See Figure 1 for a depiction of these two scenarios. The proposed positive upweighting mechanism can compensate for the selection bias and make our method also work on biased positive labeling scenarios. The proposed method is easy to implement and proved effective by experimental results.

Related Work

Let $X \in \mathbb{R}^d$ and $Y \in \{+1, -1\}$ be the input and output random variables. Let $p(x, y)$ be the underlying joint density of (X, Y). Let $p_p(x)$ denote the positive marginal density $p(x|Y = +1)$ and $p_n(x)$ denote negative marginal density $p(x|Y = -1)$. Let $\pi_p = p(Y = +1)$ and $\pi_n = p(Y = -1) = 1 - \pi_p$ be the class-prior probabilities. In PUL, the training dataset $\mathcal{X}$ consists of a positive set $\mathcal{X}_p$ and an unlabeled set $\mathcal{X}_u$, where each unlabeled example x has the probability

*Corresponding author



(a) w/ SCAR assumption (b) w/o SCAR assumption (c) Data notations

Figure 1: An illustration of the labeling mechanism assumptions in PUL. We use the sizes of markers to reflect the selection bias for labeling, where the larger marker indicates a higher selection probability. (a) PU dataset with SCAR assumption. Each positive sample is selected with equal probability for labeling. (b) PU dataset without SCAR assumption. The positive samples are selected with a labeling bias. (c) The data notations that we use throughout the paper. In this paper, we use a gradient-based regularization to address the overfitting problem of P class caused by conflicting fitting targets of the P and P′ data, and use a positive upweighting mechanism to enforce the training error of P data to be small. As a byproduct, the latter mechanism makes our method robust to the labeling bias.

$p(x)$. For PU datasets with SCAR assumption, each labeled positive example has the probability $p_p(x)$. In this paper, we will further decompose the $\mathcal{X}_u$ into an unlabeled positive set $\mathcal{X}_{p'}$ and an unlabeled negative set $\mathcal{X}_{n'}$.

Let $g : \mathbb{R}^d \to \mathbb{R}$ denote an arbitrary decision function belong to the hypothesis space $\mathcal{G}$ and $\ell : \mathbb{R} \times \pm 1 \to \mathbb{R}$ denote the loss function measuring the error of $g(x)$ to the label. Let $R_D^y(g) := E_{x\sim P_D}\ell(g(x), y)$ denote a risk that the data from P_D are regarded as class y, it can be approximated by the empirical risk $\hat{R}_D^y(g) = \frac{1}{|\mathcal{X}_D|}\sum_{x\in\mathcal{X}_D}\ell(g(x), y)$ computed on a dataset $\mathcal{X}_D$ sampled from P_D.

Naive Classifier A simple and naive method to learn a PUL classifier (Neelakantan, Roth, and McCallum 2015) is treating all unlabeled data as negative class samples and training it in a standard binary classification way. We name this method the naive classifier. The empirical risk used for training can be written as:

$$\hat{R}_{\text{naive}}(g) = \frac{1}{|\mathcal{X}_p|}\sum_{x\in\mathcal{X}_p}\ell(g(x), +1) + \frac{1}{|\mathcal{X}_u|}\sum_{x\in\mathcal{X}_u}\ell(g(x), -1).$$

Negative Selection Based To remove the unreliable negative and train a better PUL classifier in the standard binary classification way. Many methods (Zhang and Zuo 2009; Zhang et al. 2019; Luo et al. 2021) propose to identify the possible negative data in the U data or downweight the unreliable one. We view these approaches as instance-wise reweighted binary classification and summarize them into:

$$\hat{R}_{NS}(g) = \frac{1}{|\mathcal{X}_p|}\sum_{x\in\mathcal{X}_p}\ell(g(x), +1) + \frac{1}{|\mathcal{X}_u|}\sum_{x\in\mathcal{X}_u} w_u(x)\ell(g(x), -1),$$

where the $w_u(x)$ denotes the selection or reweighting function for U data. Most of the early negative selection-based works heavily rely on the heuristics to identify the negative (Zhang and Zuo 2009). Recently, some modern techniques such as Generative Adversarial Networks (GAN) (Hu et al. 2021) and reinforcement learning (Luo et al. 2021) are used to generate or select the possible negative data. However, they are often costly and unstable for training.

Loss Correction Based In contrast, the cost-sensitive methods (Kiryo et al. 2017; Du Plessis, Niu, and Sugiyama 2014) treat the risk of the naive classifier as a biased one (relative to the ideal PN classifier) and propose to correct the estimation bias. The ideal PN risk $\pi_p R_p^+(g) + (1-\pi_p)R_n^-(g)$ defined on the P and N distributions can be calculated using the unbiased risk $R_{\text{uPU}}(g) = \pi_p R_p^+(g) + R_u^-(g) - \pi_p R_p^-(g)$ defined on the P and U distribution, since $R_u^-(g) = \pi_p R_p^-(g) + (1-\pi_p)R_n^-(g)$ under SCAR assumption. The unbiased risk $R_{\text{uPU}}(g)$ can be approximated using the training data $\mathcal{X}_p$ and $\mathcal{X}_u$:

$$\hat{R}_{\text{uPU}}(g) = \pi_p\hat{R}_p^+(g) + \hat{R}_u^-(g) - \pi_p\hat{R}_p^-(g). \quad (1)$$

nnPU (Kiryo et al. 2017) found that $\hat{R}_u^-(g) - \pi_p\hat{R}_p^-(g)$ can often be negative, which is a signal for overfitting. So they proposed to optimize the following non-negative unbiased empirical risk:

$$\hat{R}_{\text{nnPU}}(g) = \pi_p\hat{R}_p^+(g) + \max\{0, \hat{R}_u^-(g) - \pi_p\hat{R}_p^-(g)\}. \quad (2)$$

Method

In this section, we first analyze the problem of existing methods. Then we propose a new way to bound the generalization error. Based on that, we propose to combine a gradient-based regularization and a positive upweighting strategy to solve the PUL problem.

The Problem of Existing PUL Methods

The common issue of most existing works is that the fitting targets of P and P′ samples can often be conflicted, which

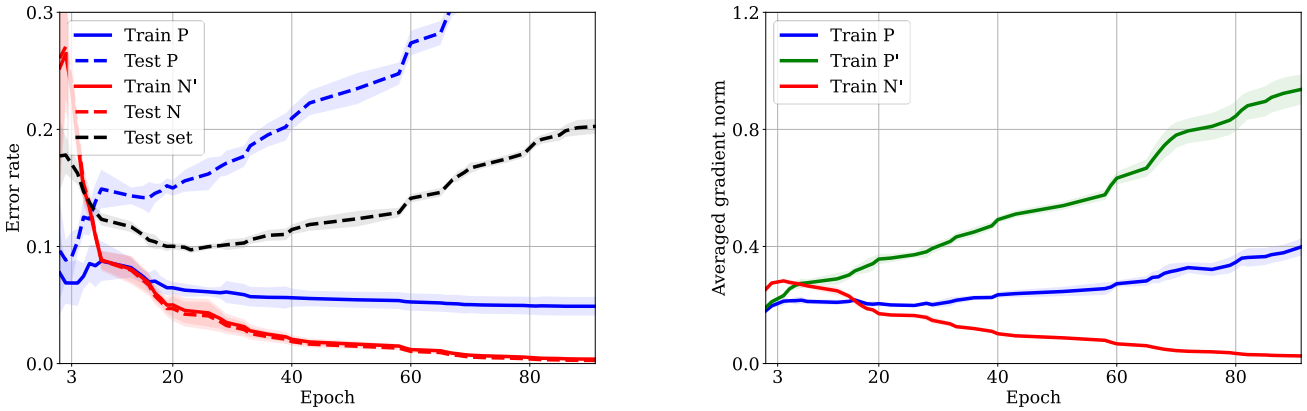


Figure 2: The training statistics of a naive classifier on MNIST dataset. Left: The test error rate of N class decreases with a very similar curve as the training error rate of N′ data. By contrast, the test error rate of the P class starts to grow after the ∼3th epoch despite the training error rate of the P data continuing to decrease. Right: Correspondingly, the mean gradient norms (the output with respect to input, $\mathrm{E}_x[\|\nabla_x g(x)\|_2^1]$) of the training P data (and P′ data) grows much faster than those of training N′ data.

will lead to the overfitting of P class.

In the *naive classifier*, the P′ samples are directly assigned with contradictory labels to the P samples.

In the *unbiased PU classifier*, $\hat{R}_u^-(g)$ in Equation 1 can be decomposed into $\pi_p\hat{R}_{p'}^-(g) + (1-\pi_p)\hat{R}_{n'}^-(g)$. Then we can find the $\pi_p\hat{R}_{p'}^-(g)$ is a conflicting term with the $-\pi_p\hat{R}_p^-(g)$.

In the *non-negative PU classifier*, the empirical risk $\hat{R}_{\text{nnPU}}(g)$ in Equation 2 can be rewritten into $\pi_p\hat{R}_p^+(g) + \max\{0, (1-\pi_p)\hat{R}_{n'}^-(g) + \pi_p\hat{R}_{p'}^-(g) - \pi_p\hat{R}_p^-(g)\}$. The optimizing of the term $\pi_p\hat{R}_{p'}^-(g) - \pi_p\hat{R}_p^-(g)$ with conflicting fitting target will be turned off when $\hat{R}_u^-(g) - \pi_p\hat{R}_p^-(g) < 0$. However, when N′ is not well-fitted (e.g. in the early training stage or with some hard negatives), $\hat{R}_{n'}^-$ can become large, and the conflicting terms will continue to be optimized until it is smaller than $-(1-\pi_p)\hat{R}_{n'}^-$.

For *negative selection-based method*, a perfect selection or reweighting function may require itself to be already a perfect PUL classifier. Otherwise, removing or downweighting an excessive number of U samples is often required to ensure that the remaining samples provide reliable supervision. However, this may result in a portion of N′ samples (especially the hard negatives) also being removed. If we try to keep more U data, the potential residual P′ data will again contribute a conflicting target with P data.

The Overfitting of P Class. Theoretical analysis in previous works often relies on the generalization bounds derived from the Rademacher complexity defined on the hypothesis space $\mathcal{G}$ and training set $\mathcal{X}$. It is often assumed that the Rademacher complexity will be bounded and decrease as the dataset increases. However, (Arpit et al. 2017) pointed out that DNN can always fit random labels even with various traditional regularizers. This indicates that when fitting conflicting targets of P and P', the flexible DNN using the traditional regularizers can still reach into a very complex function space, resulting in the overfitting of P class. We verify this by training a naive classifier and show the error rates during training in Figure 2 (Left), where we can find the overfitting problem occurs mainly in the P class rather than the N class. The overfitting problems (P class only) can also be found for nnPU and other methods, see Appendix Section 2.1.

Bounding the Generalization Error via Wasserstein Distance

To better understand and solve the overfitting of P class and further ensure both the generalization error of the P and N class can be well-bounded, we derive more useful generalization error bounds via Wasserstein distance and provide insights for improving the performance of the P class.

Definition 1 *Let $W_1(P_D, \mathcal{X}_D)$ denote the 1^{th}-Wasserstein distance between a distribution P_D and the uniform distribution over a set $\mathcal{X}_D = \{x_i | x_i \sim P_D\}_{i=1}^{N_D}$. The Kantorovich-Rubinstein duality (Villani 2009) states that $W_1(P_D, \mathcal{X}_D) = \sup_{f:\rho(f)\leq 1}\{\mathbb{E}_{x\sim P_D}[f(x)] - \mathbb{E}_{x\sim\mathcal{X}_D}[f(x)]\}$, where $\rho(f)$ is the minimal Lipschitz constant for f.*

$W_1(P_D, \mathcal{X}_D)$ can be viewed as a metric measuring how close $\mathcal{X}_D$ is to P_D. By Wasserstein law of large numbers (Van Handel 2014), the expectation of W_1 distance between a distribution and its sampled set will go to 0, as the sample size increases. Since $R_D(g) = \mathbb{E}_{x\sim P_D}[(l\circ g)(x)], \hat{R}_D(g) = \mathbb{E}_{x\sim\mathcal{X}_D}[(l\circ g)(x)]$ and $W_1(P_D, \mathcal{X}_D)$ can be written as $\frac{1}{K}\sup_{f:\rho(f)\leq K}\{\mathbb{E}_{x\sim P_D}[f(x)] - \mathbb{E}_{x\sim\mathcal{X}_D}[f(x)]\}$, we have the following lemma:

Lemma 1 *If the composition of a loss function ℓ and a decision function g is Lipschitz continuous, then we can bound the gap between the risk $R_D(g)$ and the empirical risk $\hat{R}_D(g)$. (See (Lopez and Jog 2018; Rodríguez Gálvez et al. 2021) for similar but expectation forms):*

$$R_D(g) - \hat{R}_D(g) \leq \rho(\ell\circ g) W_1(P_D, \mathcal{X}_D). \tag{3}$$

By applying Equation 3 in the scenarios where both P and N′ are correctly labeled while P′ are wrongly labeled, we can

bound the generalization errors of the P and N classes by the following theorem.

Theorem 1 *(Generalization Error Bounds) For the absolute value loss function $\ell(g, y) = |g(x) - y|$, and any Lipschitz continuous decision function $g : \mathbb{R}^d \to [-1, 1]$, assuming the labeled positive samples are selected completely at random, the generalization errors of P class $R_p^+(g)$ and N class $R_n^-(g)$ can be bounded:*

$$2 - \hat{R}_{p'}^-(g) - \rho(g) W_1(P_p, \mathcal{X}_{p'}) \leq R_p^+(g) \leq \hat{R}_p^+(g) + \rho(g) W_1(P_p, \mathcal{X}_p)$$

and,

$$R_n^-(g) \leq \hat{R}_{n'}^-(g) + \rho(g) W_1(P_n, \mathcal{X}_{n'}) \quad (4)$$

See Appendix Section 1 for the proof and analysis on more general cases (other types of loss function).

Classfiers with smaller $\rho(g)$ generalize better. The Theorem 1 tells us, if the P and N′ samples can be well fitted by training algorithm (thus $\hat{R}_p^+(g)$ and $\hat{R}_{n'}^-(g)$ can be small), the generalization error upper bounds will be dominated by $\rho(g) W_1(P_p, \mathcal{X}_p)$ and $\rho(g) W_1(P_n, \mathcal{X}_{n'})$. Considering the W_1 distances are constants for a given PU dataset, a smaller $\rho(g)$ will lead to tighter generalization error upper bounds of P and N classes, and thus the generalization performance of P and N classes will tend to be better.

Explaining the overfitting of P class. We argue that the P and P′ samples with conflicting fitting targets will force g to output different values, e.g. ± 1, for every P sample x_p and its closest P′ neighbors, making $\rho(g) \geq 2/\min_{x_{p'} \in \mathcal{X}_{p'}, x_{p'} \neq x_p} \|x_p - x_{p'}\|$. Since the P samples and P′ samples come from the same data manifold, the denominator could be very small, which makes the $\rho(g)$ become very large and the generalization error is less bounded. We can see in Figure 2 (Right), that large gradient norms (an indicator for the local minimal Lipschitz constant) will appear around the P class data, but not the N class data. This may provide an explanation for the different generalization gaps of the P and N classes in Figure 2 (Left).

Classfiers with smaller $\rho(g)$ fit P samples harder. By Theorem 1, the empirical risk of P data $\hat{R}_p^+(g)$ has a lower bound $2 - \hat{R}_{p'}^-(g) - \rho(g) W_1(P_p, \mathcal{X}_{p'}) - \rho(g) W_1(P_p, \mathcal{X}_p)$. For a given PU dataset, the W_1 distances are constants. A small enough $\rho(g)$ will make the lower bound dominated by $2 - \hat{R}_{p'}^-(g)$. That means the fitting of P′ samples as -1 will hinder the fitting of P samples as +1. For example, when the training error of P′ samples $\hat{R}_{p'}^-(g)$ decreases to 0, the training error of P samples $\hat{R}_p^+(g)$ will have a large lower bound close to 2 (the max possible value for the expectation of $\ell(g, y)$).

The previous analysis motivates us to improve the test performance of the P class by penalizing the minimal Lipschitz constant of the decision function g and enforcing small training errors for P data.

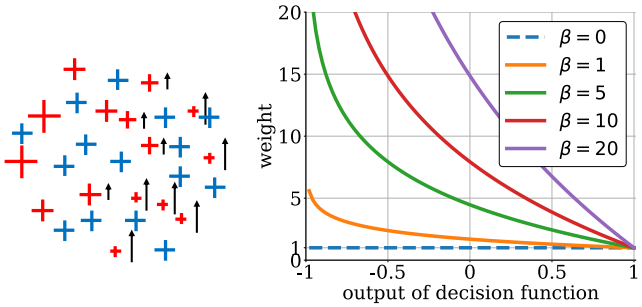


Figure 3: Illustration of the positive upweighting strategy. Left: When $\rho(g)$ is small enough, the fitting of P samples (red) around the regions dominated by P′ samples (blue) is hard, since the surrounding P′ samples have the opposite labels. We increase the weights of the not-well-fitted P examples. Right: In detail, we add a weight $w_p \geq 1$ (Equation 6) to the loss of each P sample based on its prediction. Setting $\beta > 0$ will increases the weight of those positive samples with small output values. The total weight of training P samples will thus not be normalized to 1 after scaling by $\frac{1}{|\mathcal{X}_p|}$.

Gradient-based Regularization

To penalize the $\rho(g)$ of a decision function g parameterized by deep neural networks, we will adopt a technique named gradient penalty that is widely used to enforce Lipschitz continuity on the discriminator in training generative adversarial networks (Gulrajani et al. 2017; Arjovsky and Bottou 2017; Miyato et al. 2018; Kurach et al. 2018). In detail, we perform a gradient penalty in the region between P and U data by

$$L_{GP}(\mathcal{X}; g) = E_{x \sim \text{int}(\mathcal{X}_p, \mathcal{X}_u)}[\|\nabla_x g(x)\|^2], \quad (5)$$

where $\text{int}(\mathcal{X}_p, \mathcal{X}_u) = \{\tilde{\boldsymbol{x}} | \tilde{\boldsymbol{x}} = \lambda \boldsymbol{x}_p + (1 - \lambda) \boldsymbol{x}_u, \lambda \in [0, 1], \boldsymbol{x}_p \in \mathcal{X}_p, \boldsymbol{x}_u \in \mathcal{X}_u\}$. Assuming that almost all N class data and P class data can be distinguished by a function with a small $\rho(g)$ that doesn't result in overfitting, we can use a moderate gradient penalty strength to improve the generalization of P class while maintaining the capability to correctly classify the P and N class. As far as we know, despite the popularity of gradient penalty in training GANs, there are few explorations about it in the PUL domain (also the related domains, e.g. noisy labeled learning, which also contains conflicting fitting targets).

Unnormalized Positive Upweighting

To enforce that the training error of P samples is small, we propose to upweight the P data affected by P′ data. We calculate an unnormalized weight via the output of the decision function on each P sample:

$$w_p(x; \beta) = 1 - \beta \log\left(\frac{1 + g(x)}{2}\right), \quad (6)$$

where $\beta > 0$ is a parameter controlling the upweighting strength. The upweighting mechanism aims to increase the weights of those positive samples that are less frequently labeled. Intuitively, it focuses more on hard and not-well-fitted samples like Focal Loss (Lin et al. 2017). See Figure 3 for

Algorithm 1: Training procedure of GradPU classifier.

Input: A training set $\mathcal{X} = \mathcal{X}_p \cup \mathcal{X}_u$, upweighting parameter $\beta_{\max}$, gradient penalty parameter α
Output: Model parameter θ of the binary classifier g

for $t = 1, 2, ..., N$ **do**
  Shuffle and divide dataset into I mini-batches $\{\mathcal{X}^i | \mathcal{X}^i = (\mathcal{X}_p^i, \mathcal{X}_u^i)\}$
  for $i = 1, 2, ..., I$ **do**
    Sample a mini-batch of $\boldsymbol{\lambda} \sim U(0, 1)$ with batch size $B = \max(|\mathcal{X}_p^i|, |\mathcal{X}_u^i|)$
    Resample both $\mathcal{X}_p^i$ and $\mathcal{X}_u^i$ into the common batch size B, denoted as $\tilde{\mathcal{X}}_p^i$ and $\tilde{\mathcal{X}}_u^i$
    Construct a mini-batch of interpolated data by $\boldsymbol{\lambda}\tilde{\mathcal{X}}_p^i + (1 - \boldsymbol{\lambda})\tilde{\mathcal{X}}_u^i$
    Compute the current upweight parameter $\beta = \frac{t \cdot I + i}{N \cdot I} \beta_{\max}$
    Update θ with $\nabla_\theta \hat{R}_{\text{GradPU}}(g)$ by Equation 7
  end for
end for

an explanation. Unlike negative-selection based methods, we do not remove or explicitly downweight the U data. Therefore, all N′ samples can be more sufficiently fitted by enough training iterations and model capacity. We highlight that the larger weights for the less frequently labeled positive samples make the upweighting function $w_p(x; \beta)$ compensate a non-uniform labeling frequency, thus enabling the proposed method to be better applied to biased positive labeling scenarios.

Overall Training Objective

Putting the gradient-based regularization and the unnormalized upweight mechanism together, the overall training objective of GradPU is to minimize:

$$\begin{aligned}\hat{R}_{\text{GradPU}} = &\frac{1}{|\mathcal{X}_p|} \sum_{x \in \mathcal{X}_p} w_p(x; \beta) \ell(g(x), +1) + \\ &\frac{1}{|\mathcal{X}_u|} \sum_{x \in \mathcal{X}_u} \ell(g(x), -1) + \alpha L_{\text{GP}}(\mathcal{X}; g),\end{aligned} \tag{7}$$

where $\alpha > 0$ is a parameter controlling the strength of gradient penalty. The loss function is simply chosen as $\ell(g(x), y) = |y - g(x)|$, and the output of g is restricted to $[-1, 1]$ via a $\tanh$ activation function. In practice, we find using the gradient before the $\tanh$ activation for penalizing is better. Besides, an initial large value for β will make the training unstable (maybe caused by the imbalanced total weights for positive and negative terms). So we gradually and linearly increase the β from 0 to a predefined max value $\beta_{\max}$. The description of the overall training procedure can be found in Algorithm 1.

Experiments

In this section, we first evaluate our proposed GradPU on MNIST, FashionMNIST, and CIFAR-10. We then provide ablation studies on the gradient-based regularizer and unnormalized upweighting. We also demonstrate the robustness of GradPU with different sizes of the labeled set. Finally, we test the performance of GradPU on experiments with selection bias settings. Following previous PUL works, we construct positive-unlabeled datasets by adapting the commonly used, fully labeled, and multi-class datasets.

MNIST: The MNIST dataset contains 10 classes of gray-scale handwritten digit images with 28×28 pixels. It consists of 60,000 training images and 10,000 test images. As in (Chen et al. 2020), we choose odd numbers 1, 3, 5, 7, and 9 as the positive class while other numbers as the negative class.

FashionMNIST: The FashionMNIST dataset contains 10 classes of gray-scale fashion product images. Like MNIST, it consists of 60,000 training images and 10,000 test images, with the size of 28×28 pixels. The classes with indexes 0, 2, 4, 6, and 8 are used as the positive class and others as the negative class. In the following, we will also use FMNIST to denote FashionMNIST.

CIFAR10: The CIFAR10 dataset consists of colored images with sizes of 32×32 pixels. There are 50,000 training images and 10,000 test images. The vehicles classes ("airplane", "automobile", "ship", "truck") are chosen as positive class, and the animal classes ("bird", "cat", "deer", "dog", "frog", "horse") are chosen as negative class (Kiryo et al. 2017; Chen et al. 2020; Kato, Teshima, and Honda 2018).

Implementation Details

For fair comparisons, we follow the classifier architectures in (Kiryo et al. 2017; Chen et al. 2020), an MLP with four fully connected hidden layers of 300 neurons for MNIST and FashionMNIST, and a 13-layers convolutional neural network for CIFAR-10. We remove the batch normalization layers since it is found to be incompatible with the gradient-based regularizer (Kurach et al. 2018). We use the Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. For all trials, we use the batch size of 250 and gradually decrease the learning rate to 0.1 times of its initial value. We report the mean accuracies and deviations over three runs. For MNIST and FashionMNIST, we train the models for 100 epochs, while for CIFAR10, we train the models for 200 epochs. We tune hyper-parameters such as weight decay, learning rate, the gradient penalty strength α, and the upweighting strength $\beta_{\max}$ using a validation set. More experimental details can be found in Appendix Section 3.

Dataset

Evaluation under SCAR Setting

Most existing PUL methods are evaluated under the SCAR assumption. We follow the protocol widely used by randomly selecting a small portion of positive instances from the original training data as labeled positive $\mathcal{X}_p$, 500 instances as the validation set $\mathcal{X}_{\text{val}}$, and the remaining instances as unlabeled data $\mathcal{X}_u$. The test set $\mathcal{X}_{\text{test}}$ is adapted from the original test set by relabeling the data into binary classes. We first compute the real class prior based on the dataset and use it for methods requiring a class prior. We report the experimental results for

Dataset	Naive	uPU	nnPU	VPU	SelfPU	GradPU	Sup.
MNIST 1k	92.9 ± 0.5	92.5 ± 0.4	93.6 ± 0.2	93.9 ± 0.5	94.2 ± 0.5	$\mathbf{96.4 \pm 0.4}$	96.7 ± 0.2
MNIST 3k	95.1 ± 0.3	95.2 ± 0.3	95.8 ± 0.3	95.8 ± 0.4	96.0 ± 0.3	$\mathbf{97.7 \pm 0.2}$	97.8 ± 0.2
CIFAR10 1k	88.1 ± 0.7	88.0 ± 0.6	88.6 ± 0.4	88.6 ± 0.3	89.7 ± 0.2	$\mathbf{90.1 \pm 0.2}$	90.5 ± 0.5
CIFAR10 3k	91.0 ± 0.5	90.4 ± 0.6	90.9 ± 0.2	90.5 ± 0.1	90.8 ± 0.2	$\mathbf{91.9 \pm 0.1}$	92.9 ± 0.3
FMNIST 1k	94.6 ± 0.4	94.9 ± 0.2	95.7 ± 0.2	95.3 ± 0.7	-	$\mathbf{96.3 \pm 0.2}$	96.6 ± 0.2
FMNIST 3k	96.0 ± 0.3	96.1 ± 0.3	96.3 ± 0.4	96.5 ± 0.1	-	$\mathbf{96.7 \pm 0.1}$	97.2 ± 0.2

Table 1: Test accuracy on benchmark datasets. The "Sup." refers to the supervised binary classifier trained using $\mathcal{X}_p$ and $\mathcal{X}_{n'}$. "1k" and "3k" denote the number of labeled P samples for training. Bold font indicates the highest performance among the non-supervised methods.

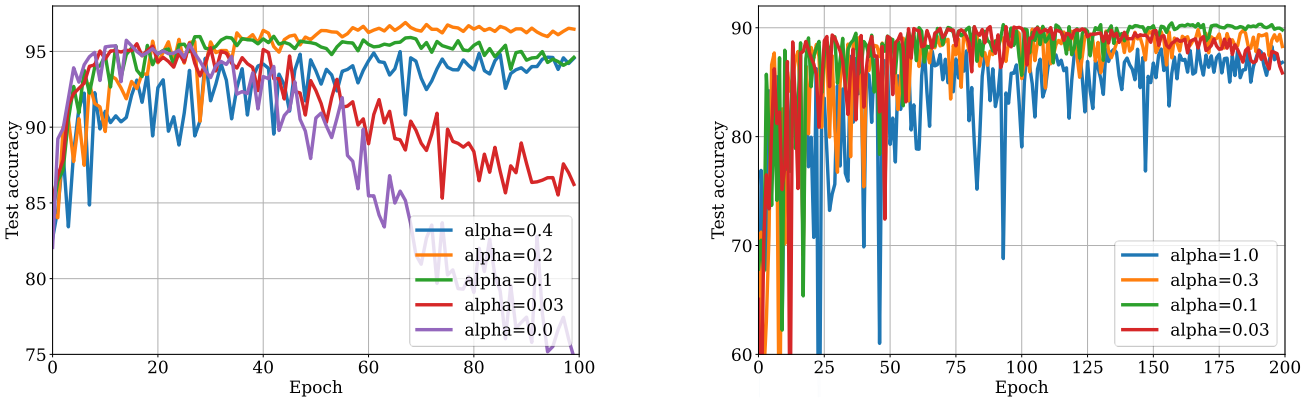


Figure 4: Study of the gradient penalty strength α on MNIST (left) and CIFAR10 (right) with 1000 labeled positive samples.

both cases $|\mathcal{X}_p| = 1,000$ and $|\mathcal{X}_p| = 3,000$ in Table 1. We can see that GradPU outperforms the baseline models in all datasets and achieves performance very close to the supervised classifier trained using $\mathcal{X}_p$ and $\mathcal{X}_{n'}$ in some cases. Other evaluation protocols (about $|\mathcal{X}_p|$, positive classes, classifier architectures, and the validation set) and the corresponding experimental results can be found in Appendix Section 2.

Ablation Study

Effects of Regularization Strength The penalty strength on gradient norm is controlled by the parameter α. The larger α is, the smoother decision function the model will learn. We study the effect of α and report the test accuracies during the training process of GradPU on MNIST and CIFAR10. As shown in Figure 4, a moderate α (0.2 for MNIST and 0.1 for CIFAR10) can avoid overfitting during the training process and achieve the best performance. In contrast, a much smaller or larger α leads to a reasonable overfitting or underfitting phenomena.

Comparison with Other Regularizers To verify the superiority of the gradient-based regularizer over the traditional regularizers, we also compare our gradient-based regularization with the following regularization methods: network width, weight decay, dropout, and mixup (Zhang et al. 2018a). Considering early stopping is also a well-known technique to avoid memorization, we report both the last accuracy and the best accuracy during training. For all training except the weight decay regularizer itself, we use the default weight decay 5e-4. Learning rates are searched in range of {4e-5, 1e-4, 4e-4, 1e-3}. The search ranges of regularization strengths are as follows: model width, {300, 150, 75, 35}; weight decay, {1e-6, 1e-4, 1e-3, 1e-2, 1e-1}; dropout, {0.2, 0.4, 0.6, 0.8}; input noise, {0.05, 0.1, 0.2, 0.4}; mixup, {0.1, 0.3, 1.0, 3.0, 10.0}. The results are reported in Table 2. The gradient-based regularization always outperforms all other regularization methods.

Unnormalized Positive Upweighting We show the accuracies under various strengths of upweighting $\beta_{\max}$ in Table 3. As we can see, moderate (non-zero) upweighting strengths are useful in all three datasets. Since the upweighting mechanism is designed to alleviate the resistance effect of the P′ data to the fitting of the P class, a PU dataset with a smaller proportion of P′ data in U data is expected to require a smaller upweighting strength. This can be verified by the observation that the best upweighting strength in CIFAR10 is smaller than those in MNSIT and FashionMNIST.

Robustness to the Number of Labeled Positives

We also investigate the performance of GradPU with various numbers of labeled positives on MNIST and FashionMNIST, specifically, $|\mathcal{X}_p|$ ranges in $\{1000, 2000, 4000, 8000, 16000\}$. We calculate the mean accuracies and deviations on three independent runs. The results are shown in Figure 5. We can see that GradPU can surpass the baseline under various positive labeling proportion cases. It performs comparably to the supervised classifier trained using $\mathcal{X}_p$ and $\mathcal{X}_{n'}$ when the labeled positive number is bigger than 4000 on MNIST.

	MNIST 1k		MNIST 3k		FMNIST 1k	
	best	last	best	last	best	last
Model width	92.3 ± 0.2	83.0 ± 0.7	96.1 ± 0.1	95.5 ± 0.3	95.7 ± 0.2	92.4 ± 0.6
Weight decay	93.9 ± 0.3	90.6 ± 0.8	95.8 ± 0.1	95.6 ± 0.2	95.6 ± 0.2	93.9 ± 0.6
Dropout	92.6 ± 0.7	84.7 ± 1.7	96.2 ± 0.2	95.3 ± 0.8	95.8 ± 0.2	92.9 ± 0.6
Input noise	93.4 ± 0.1	91.8 ± 0.5	96.2 ± 0.2	95.6 ± 0.8	95.9 ± 0.1	95.5 ± 0.4
Mixup	94.2 ± 0.3	90.2 ± 0.5	96.3 ± 0.1	93.3 ± 0.5	95.8 ± 0.4	93.9 ± 0.5
GP on int($\mathcal{X}$)	$\mathbf{96.4 \pm 0.2}$	$\mathbf{95.9 \pm 0.5}$	$\mathbf{97.7 \pm 0.2}$	$\mathbf{97.3 \pm 0.2}$	$\mathbf{96.4 \pm 0.1}$	$\mathbf{96.2 \pm 0.2}$

Table 2: Comparision with various regularization methods. Results of both the last and best epoch are shown. We remark that all the trials use the proposed upweighting strategy, the best searched learning rates, and the best regularization strengths.

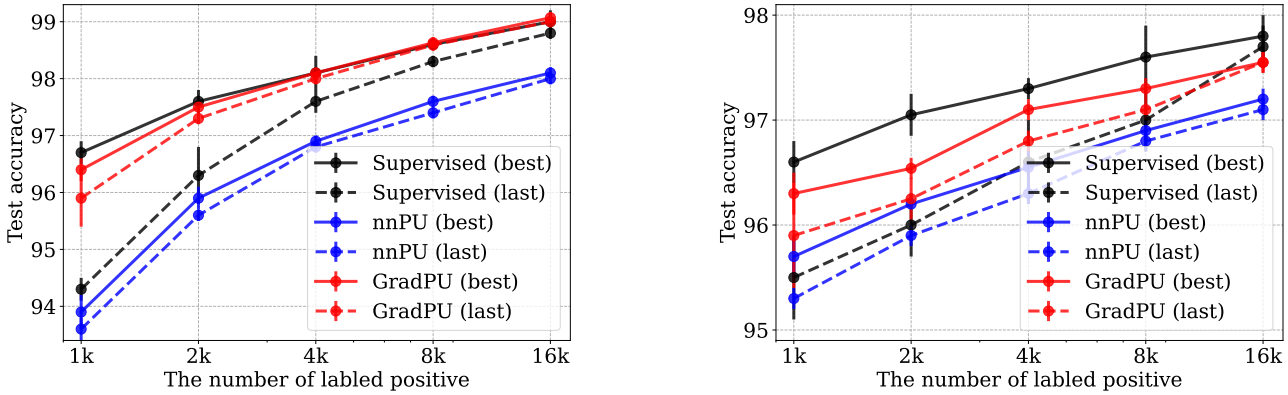


Figure 5: Accuracies of GradPU with various labeled positive sizes on MNIST (left) and FashionMNIST (right).

$\beta_{\max}$	MNIST 1k ($\pi_p \approx 0.5$)	FMNIST 1k ($\pi_p \approx 0.5$)	CIFAR10 1k ($\pi_p \approx 0.4$)
0.0	94.8 ± 0.4	95.9 ± 0.3	89.7 ± 0.3
1.0	95.7 ± 0.2	96.0 ± 0.2	$\mathbf{90.1 \pm 0.1}$
3.0	96.0 ± 0.4	96.1 ± 0.3	$\mathbf{90.1 \pm 0.2}$
10.0	96.2 ± 0.3	$\mathbf{96.3 \pm 0.2}$	89.6 ± 0.4
30.0	$\mathbf{96.4 \pm 0.4}$	96.2 ± 0.1	89.1 ± 0.3

Table 3: Effectiveness of positive upweighting mechanism.

	MNIST 1k	FMNIST 1k	CIFAR10 1k
nnPU	89.7 ± 0.9	94.2 ± 0.2	87.1 ± 0.7
nnPUSB	91.5 ± 0.9	95.1 ± 0.3	87.3 ± 0.6
GradPU$_{\beta=0}$	81.7 ± 1.2	93.9 ± 0.5	79.7 ± 0.8
GradPU$_{\alpha=0}$	90.5 ± 0.6	94.7 ± 0.7	84.1 ± 0.6
GradPU	$\mathbf{94.1 \pm 0.3}$	$\mathbf{96.0 \pm 0.3}$	$\mathbf{88.6 \pm 0.5}$

Table 4: Accuracy of the proposed GradPU with a selection bias on the positive labels, comparing with nnPU, nnPUSB, and ablated GradPUs.

Evaluation under Selection Bias Setting

We evaluate the GradPU on the scenario without SCAR assumption using MNIST, FashionMNIST, and CIFAR10 datasets. To simulate a labeling bias, we obtain the labeled positive data by labeling some instances of the positive data following a probability induced from a well-trained logistic regression model as PUSB(Kato, Teshima, and Honda 2018), see Appendix Section 3 for more details. We compare the GradPU with nnPU, nnPUSB, and ablated GradPUs. As shown in Table 4, the GradPU outperforms the existing methods on all datasets. Both the gradient penalty and upweighting mechanism are effective in such a scenario. Removing the upweighting mechanism ($\beta_{\max} = 0$) results in a significant performance drop, which verifies that the upweighting mechanism is useful and essential for handling the selection bias.

Conclusion

In this paper, we first presented theoretical analyses about the naive classifier and the corresponding overfitting problem of P class. Based on that, we then proposed a novel PUL method named GradPU, which combines a gradient-based regularizer and a positive upweighting strategy. The gradient-based regularizer improves the generalization of P class, and the positive upweighting mechanism ensures each training P sample is well-fitted. The GradPU is easy to implement, can be used to train flexible DNN models, and works on both unbiased and biased positive labeling scenarios. Experimental results show that GradPU outperforms the previous PUL methods on MNIST, FashionMNIST, and CIFAR10. In the future, we will explore the GradPU in the application domains and investigate the generality of the gradient-based regularizer on other topics with conflicting targets.

Acknowledgements

This work is supported in part by Shanghai science and technology committee under grant No. 21511100600. We appreciate the High Performance Computing Center of Shanghai University, and Shanghai Engineering Research Center of Intelligent Computing System for providing the computing resources and technical support.

References

Arjovsky, M.; and Bottou, L. 2017. Towards principled methods for training generative adversarial networks. In *ICLR*.

Arpit, D.; Jastrzębski, S.; Ballas, N.; Krueger, D.; Bengio, E.; Kanwal, M. S.; Maharaj, T.; Fischer, A.; Courville, A.; Bengio, Y.; et al. 2017. A closer look at memorization in deep networks. In *ICML*, 233–242. PMLR.

Bekker, J.; and Davis, J. 2018. Learning from positive and unlabeled data under the selected at random assumption. In *Second International Workshop on Learning with Imbalanced Domains: Theory and Applications*, 8–22. PMLR.

Bekker, J.; and Davis, J. 2020. Learning from positive and unlabeled data: A survey. *Machine Learning*, 109(4): 719–760.

Chen, X.; Chen, W.; Chen, T.; Yuan, Y.; Gong, C.; Chen, K.; and Wang, Z. 2020. Self-pu: Self boosted and calibrated positive-unlabeled training. In *ICML*, 1510–1519. PMLR.

Chuang, C.-Y.; Robinson, J.; Lin, Y.-C.; Torralba, A.; and Jegelka, S. 2020. Debiased contrastive learning. *NeurIPS*, 33: 8765–8775.

Du Plessis, M.; Niu, G.; and Sugiyama, M. 2015. Convex formulation for learning from positive and unlabeled data. In *ICML*, 1386–1394. PMLR.

Du Plessis, M. C.; Niu, G.; and Sugiyama, M. 2014. Analysis of learning from positive and unlabeled data. *NeurIPS*, 27.

Gulrajani, I.; Ahmed, F.; Arjovsky, M.; Dumoulin, V.; and Courville, A. C. 2017. Improved training of wasserstein gans. *NeurIPS*, 30.

Guo, J.; Han, K.; Wu, H.; Zhang, C.; Chen, X.; Xu, C.; Xu, C.; and Wang, Y. 2021. Positive-Unlabeled Data Purification in the Wild for Object Detection. In *CVPR*, 2653–2662.

Hammoudeh, Z.; and Lowd, D. 2020. Learning from positive and unlabeled data with arbitrary positive shift. *NeurIPS*, 33: 13088–13099.

Hou, M.; Chaib-Draa, B.; Li, C.; and Zhao, Q. 2018. Generative adversarial positive-unlabeled learning. In *IJCAI*, 2255–2261.

Hu, W.; Le, R.; Liu, B.; and Ji, e. a. 2021. Predictive Adversarial Learning from Positive and Unlabeled Data. In *AAAI*, volume 35, 7806–7814.

Kato, M.; Teshima, T.; and Honda, J. 2018. Learning from positive and unlabeled data with a selection bias. In *ICLR*.

Kiryo, R.; Niu, G.; Du Plessis, M. C.; and Sugiyama, M. 2017. Positive-unlabeled learning with non-negative risk estimator. *NeurIPS*, 30.

Kurach, K.; Lucic, M.; Zhai, X.; Michalski, M.; and Gelly, S. 2018. The GAN Landscape: Losses, Architectures, Regularization, and Normalization. *ArXiv*, abs/1807.04720.

Li, W.; Guo, Q.; and Elkan, C. 2010. A positive and unlabeled learning algorithm for one-class classification of remote-sensing data. *IEEE transactions on geoscience and remote sensing*, 49(2): 717–725.

Lin, T.-Y.; Goyal, P.; Girshick, R. B.; He, K.; and Dollár, P. 2017. Focal Loss for Dense Object Detection. In *ICCV*.

Lopez, A. T.; and Jog, V. 2018. Generalization error bounds using Wasserstein distances. *2018 IEEE Information Theory Workshop (ITW)*, 1–5.

Luo, C.; Zhao, P.; Chen, C.; and Qiao, e. a. 2021. PULNS: Positive-Unlabeled Learning with Effective Negative Sample Selector. In *AAAI*, volume 35, 8784–8792.

Miyato, T.; Kataoka, T.; Koyama, M.; and Yoshida, Y. 2018. Spectral Normalization for Generative Adversarial Networks. In *ICLR*.

Na, B.; Kim, H.; Song, K.; and Joo, e. a. 2020. Deep generative positive-unlabeled learning under selection bias. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 1155–1164.

Neelakantan, A.; Roth, B.; and McCallum, A. 2015. Compositional Vector Space Models for Knowledge Base Completion. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 156–166.

Ren, Y.; Ji, D.; and Zhang, H. 2014. Positive unlabeled learning for deceptive reviews detection. In *EMNLP*, 488–498.

Rodríguez Gálvez, B.; Bassi, G.; Thobaben, R.; and Skoglund, M. 2021. Tighter expected generalization error bounds via wasserstein distance. In *NeurIPS*, volume 34.

Van Handel, R. 2014. Probability in high dimension. Technical report, PRINCETON UNIV NJ.

Villani, C. 2009. *Optimal transport: old and new*, volume 338. Springer.

Xu, D.; and Denil, M. 2021. Positive-Unlabeled Reward Learning. In *CoRL*, 205–219. PMLR.

Yang, P.; Li, X.-L.; Mei, J.-P.; Kwoh, C.-K.; and Ng, S.-K. 2012. Positive-unlabeled learning for disease gene identification. *Bioinformatics*, 28(20): 2640–2647.

Yang, Y.; Liang, K. J.; and Carin, L. 2020. Object Detection as a Positive-Unlabeled Problem. *arXiv preprint arXiv:2002.04672*.

Zhang, B.; and Zuo, W. 2009. Reliable Negative Extracting Based on kNN for Learning from Positive and Unlabeled Examples. *J. Comput.*, 4: 94–101.

Zhang, C.; Ren, D.; Liu, T.; Yang, J.; and Gong, C. 2019. Positive and Unlabeled Learning with Label Disambiguation. In *IJCAI*.

Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2018a. mixup: Beyond Empirical Risk Minimization. In *ICLR*.

Zhang, J.; Wang, Z.; Meng, J.; Tan, Y.-P.; and Yuan, J. 2018b. Boosting positive and unlabeled learning for anomaly detection with multi-features. *IEEE Transactions on Multimedia*, 21(5): 1332–1344.