Classification from Positive, Unlabeled and Biased Negative Data

Yu-Guan Hsieh [*1] **Gang Niu** [2] **Masashi Sugiyama** [2 3]

Abstract

In binary classification, there are situations where negative (N) data are too diverse to be fully labeled and we often resort to positive-unlabeled (PU) learning in these scenarios. However, collecting a non-representative N set that contains only a small portion of all possible N data can often be much easier in practice. This paper studies a novel classification framework which incorporates such biased N (bN) data in PU learning. We provide a method based on empirical risk minimization to address this PUbN classification problem. Our approach can be regarded as a novel example-weighting algorithm, with the weight of each example computed through a preliminary step that draws inspiration from PU learning. We also derive an estimation error bound for the proposed method. Experimental results demonstrate the effectiveness of our algorithm in not only PUbN learning scenarios but also ordinary PU learning scenarios on several benchmark datasets.

1. Introduction

In conventional binary classification, examples are labeled as either positive (P) or negative (N), and we train a classifier on these labeled examples. On the contrary, positive-unlabeled (PU) learning addresses the problem of learning a classifier from P and unlabeled (U) data, without the need of explicitly identifying N data (Elkan & Noto, 2008; Ward et al., 2009).

PU learning finds its usefulness in many real-world problems. For example, in one-class remote sensing classification (Li et al., 2011), we seek to extract a specific land-cover class from an image. While it is easy to label examples of this specific land-cover class of interest, examples not belonging to this class are too diverse to be exhaustively annotated. The same problem arises in text classification, as it is difficult or even impossible to compile a set of N samples that provides a comprehensive characterization of everything that is not in the P class (Liu et al., 2003; Fung et al., 2006). Besides, PU learning has also been applied to other domains such as outlier detection (Hido et al., 2008; Scott & Blanchard, 2009), medical diagnosis (Zuluaga et al., 2011), or time series classification (Nguyen et al., 2011).

By carefully examining the above examples, we find out that the most difficult step is often to collect a fully representative N set, whereas only labeling a small portion of all possible N data is relatively easy. Therefore, in this paper, we propose to study the problem of learning from P, U and biased N (bN) data, which we name PUbN learning hereinafter. We suppose that in addition to P and U data, we also gather a set of bN samples, governed by a distribution distinct from the true N distribution. As described previously, this can be viewed as an extension of PU learning, but such bias may also occur naturally in some real-world scenarios. For instance, let us presume that we would like to judge whether a subject is affected by a particular disease based on the result of a physical examination. While the data collected from the patients represent rather well the P distribution, healthy subjects that request the examination are in general biased with respect to the whole healthy subject population.

We are not the first to be interested in learning with bN data. In fact, both Li et al. (2010) and Fei & Liu (2015) attempted to solve similar problems in the context of text classification. Li et al. (2010) simply discarded N samples and performed ordinary PU classification. It was also mentioned in the paper that bN data could be harmful. Fei & Liu (2015) adopted another strategy. The authors considered even gathering unbiased U data is difficult and learned the classifier from only P and bN data. However, their method is specific to text classification because it relies on the use of effective similarity measures to evaluate similarity between documents (refer to Supplementary Material D.5 for a deeper discussion and an empirical comparison with our method). Therefore, our work differs from these two in that the classifier is trained simultaneously on P, U and bN data, without resorting to domain-specific knowledge. The presence of U data allows us to address the problem from a statistical viewpoint, and thus the proposed method can be

[1]École Normale Supérieure, Paris, France [2]RIKEN, Tokyo, Japan [3]The University of Tokyo, Tokyo, Japan. Correspondence to: Yu-Guan Hsieh <yu-guan.hsieh@ens.fr>.

[*] Most of the work done during internship at RIKEN.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

applied to any PUbN learning problem in principle.

In this paper, we develop an empirical risk minimization-based algorithm that combines both PU learning and importance weighting to solve the PUbN classification problem. We first estimate the probability that an example is sampled into the P or the bN set. Based on this estimate, we regard bN and U data as N examples with instance-dependent weights. In particular, we assign larger weights to U examples that we believe to appear less often in the P and bN sets. P data are treated as P examples with unity weight but also as N examples with usually small or zero weight whose actual value depends on the same estimate.

The contributions of the paper are three-fold:

1. We formulate the PUbN learning problem as an extension of PU learning and propose an empirical risk minimization-based method to address the problem. We also theoretically establish an estimation error bound for the proposed method.
2. We experimentally demonstrate that the classification performance can be effectively improved thanks to the use of bN data during training. In other words, PUbN learning yields better performance than PU learning.
3. Our method can be easily adapted to ordinary PU learning. Experimentally we show that the resulting algorithm allows us to obtain new state-of-the-art results on several PU learning tasks.

Relation with Semi-supervised Learning. With P, N and U data available for training, our problem setup may seem similar to that of semi-supervised learning (Chapelle et al., 2010; Oliver et al., 2018). Nonetheless, in our case, N data are biased and often represent only a small portion of the whole N distribution. Therefore, most of the existing methods designed for the latter cannot be directly applied to the PUbN classification problem. Furthermore, our focus is on deducing a risk estimator using the three sets of data, with U data in particular used to compensate the sampling bias in N data. On the other hand, in semi-supervised learning the main concern is often how U data can be utilized for regularization (Grandvalet & Bengio, 2005; Belkin et al., 2006; Miyato et al., 2016; Laine & Aila, 2017). The two should be compatible and we believe adding such regularization to our algorithm can be beneficial in many cases.

Relation with Dataset Shift. PUbN learning can also be viewed as a special case of dataset shift[1] (Quionero-Candela et al., 2009) if we consider that P and bN data are drawn from the training distribution while U data are drawn from the test distribution. Covariate shift (Shimodaira, 2000; Sugiyama & Kawanabe, 2012) is another special case of dataset shift that has been studied intensively. In the covariate shift problem setting, training and test distributions have the same class conditional distribution and only differ in the marginal distribution of the independent variable. One popular approach to tackle this problem is to reweight each training example according to the ratio of the test density to the training density (Huang et al., 2007; Sugiyama et al., 2008). Nevertheless, simply training a classifier on a reweighted version of the labeled set is not sufficient in our case since there may be examples with zero probability to be labeled, and it is therefore essential to involve U samples in the second step of the proposed algorithm. It is also important to notice that the problem of PUbN learning is intrinsically different from that of covariate shift and neither of the two is a special case of the other.

Finally, source component shift (Quionero-Candela et al., 2009) is also related. It assumes that data are generated from several different sources and the proportions of these sources may vary between training and test times. In many practical situations, this is indeed what causes our collected N data to be biased. However, its definition is so general that we are not aware of any universal method which addresses this problem without explicit model assumptions on data distribution.

2. Problem Setting

In this section, we briefly review the formulations of PN, PU and PNU classification and introduce the problem of learning from P, U and bN data.

2.1. Standard Binary Classification

Let $\boldsymbol{x} \in \mathbb{R}^d$ and $y \in \{+1, -1\}$ be random variables following an unknown probability distribution with density $p(\boldsymbol{x}, y)$. Let $g : \mathbb{R}^d \to \mathbb{R}$ be an arbitrary decision function for binary classification and $\ell : \mathbb{R} \to \mathbb{R}_+$ be a loss function of margin $yg(\boldsymbol{x})$ that usually takes a small value for a large margin. The goal of binary classification is to find g that minimizes the classification risk:

$$R(g) = \mathbb{E}_{(\boldsymbol{x},y)\sim p(\boldsymbol{x},y)}[\ell(yg(\boldsymbol{x}))], \tag{1}$$

where $\mathbb{E}_{(\boldsymbol{x},y)\sim p(\boldsymbol{x},y)}[\cdot]$ denotes the expectation over the joint distribution $p(\boldsymbol{x}, y)$. When we care about classification accuracy, ℓ is the zero-one loss $\ell_{01}(z) = (1 - \mathrm{sign}(z))/2$. However, for ease of optimization, ℓ_{01} is often substituted with a surrogate loss such as the sigmoid loss $\ell_{\mathrm{sig}}(z) = 1/(1 + \exp(z))$ or the logistic loss $\ell_{\log}(z) = \ln(1 + \exp(-z))$ during learning.

[1] Dataset shift refers to any case where training and test distributions differ. The term sample selection bias (Heckman, 1979; Zadrozny, 2004) is sometimes used to describe the same thing. However, strictly speaking, sample selection bias actually refers to the case where training instances are first drawn from the test distributions and then a subset of these data is systematically discarded due to a particular mechanism.

In standard supervised learning scenarios (PN classification), we are given P and N data that are sampled independently from $p_{\mathrm{P}}(\boldsymbol{x}) = p(\boldsymbol{x} \mid y = +1)$ and $p_{\mathrm{N}}(\boldsymbol{x}) = p(\boldsymbol{x} \mid y = -1)$ as $\mathcal{X}_{\mathrm{P}} = \{\boldsymbol{x}_i^{\mathrm{P}}\}_{i=1}^{n_{\mathrm{P}}}$ and $\mathcal{X}_{\mathrm{N}} = \{\boldsymbol{x}_i^{\mathrm{N}}\}_{i=1}^{n_{\mathrm{N}}}$. Let us denote by $R_{\mathrm{P}}^+(g) = \mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{P}}(\boldsymbol{x})}[\ell(g(\boldsymbol{x}))]$, $R_{\mathrm{N}}^-(g) = \mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{N}}(\boldsymbol{x})}[\ell(-g(\boldsymbol{x}))]$ partial risks and $\pi = p(y = 1)$ the P prior. We have the equality $R(g) = \pi R_{\mathrm{P}}^+(g) + (1 - \pi)R_{\mathrm{N}}^-(g)$. The classification risk (1) can then be empirically approximated from data by

$$\hat{R}_{\mathrm{PN}}(g) = \pi \hat{R}_{\mathrm{P}}^+(g) + (1 - \pi)\hat{R}_{\mathrm{N}}^-(g),$$

where $\hat{R}_{\mathrm{P}}^+(g) = \frac{1}{n_{\mathrm{P}}} \sum_{i=1}^{n_{\mathrm{P}}} \ell(g(\boldsymbol{x}_i^{\mathrm{P}}))$ and $\hat{R}_{\mathrm{N}}^-(g) = \frac{1}{n_{\mathrm{N}}} \sum_{i=1}^{n_{\mathrm{N}}} \ell(-g(\boldsymbol{x}_i^{\mathrm{N}}))$. By minimizing $\hat{R}_{\mathrm{PN}}(g)$ we obtain the ordinary empirical risk minimizer $\hat{g}_{\mathrm{PN}}$.

2.2. PU Classification

In PU classification, instead of N data $\mathcal{X}_{\mathrm{N}}$ we have only access to $\mathcal{X}_{\mathrm{U}} = \{x_i^{\mathrm{U}}\}_{i=1}^{n_{\mathrm{U}}} \sim p(\boldsymbol{x})$ a set of U samples drawn from the marginal density $p(\boldsymbol{x})$. Several effective algorithms have been designed to address this problem. Liu et al. (2002) proposed the S-EM approach that first identifies reliable N data in the U set and then run the Expectation-Maximization (EM) algorithm to build the final classifier. The biased support vector machine (Biased SVM) introduced by Liu et al. (2003) regards U samples as N samples with smaller weights. Mordelet & Vert (2014) solved the PU problem by aggregating classifiers trained to discriminate P data from a small random subsample of U data. An ad hoc algorithm designed for linear classifiers, treating the U set as an N set influenced by label noise, was proposed in (Shi et al., 2018).

Recently, attention has also been paid on the unbiased risk estimator proposed by du Plessis et al. (2014; 2015). The key idea is to use the following equality:

$$(1 - \pi)R_{\mathrm{N}}^-(g) = R_{\mathrm{U}}^-(g) - \pi R_{\mathrm{P}}^-(g),$$

where $R_{\mathrm{U}}^-(g) = \mathbb{E}_{x\sim p(\boldsymbol{x})}[\ell(-g(\boldsymbol{x}))]$ and $R_{\mathrm{P}}^-(g) = \mathbb{E}_{x\sim p_{\mathrm{P}}(\boldsymbol{x})}[\ell(-g(\boldsymbol{x}))]$. This equality is acquired by exploiting the fact $p(\boldsymbol{x}) = \pi p_{\mathrm{P}}(\boldsymbol{x}) + (1 - \pi)p_{\mathrm{N}}(\boldsymbol{x})$. As a result, we can approximate the classification risk (1) by

$$\hat{R}_{\mathrm{PU}}(g) = \pi \hat{R}_{\mathrm{P}}^+(g) - \pi \hat{R}_{\mathrm{P}}^-(g) + \hat{R}_{\mathrm{U}}^-(g), \tag{2}$$

where $\hat{R}_{\mathrm{P}}^-(g) = \frac{1}{n_{\mathrm{P}}} \sum_{i=1}^{n_{\mathrm{P}}} \ell(-g(\boldsymbol{x}_i^{\mathrm{P}}))$ and $\hat{R}_{\mathrm{U}}^-(g) = \frac{1}{n_{\mathrm{U}}} \sum_{i=1}^{n_{\mathrm{U}}} \ell(-g(\boldsymbol{x}_i^{\mathrm{U}}))$. We then minimize $\hat{R}_{\mathrm{PU}}(g)$ to obtain another empirical risk minimizer $\hat{g}_{\mathrm{PU}}$. Note that as the loss is always positive, the classification risk (1) that $\hat{R}_{\mathrm{PU}}(g)$ approximates is also positive. However, Kiryo et al. (2017) pointed out that when the model of g is too flexible, that is, when the function class $\mathcal{G}$ is too large, $\hat{R}_{\mathrm{PU}}(\hat{g}_{\mathrm{PU}})$ indeed goes negative and the model severely overfits the training data. To alleviate overfitting, the authors observed that $R_{\mathrm{U}}^-(g) - \pi R_{\mathrm{P}}^-(g) = (1 - \pi)R_{\mathrm{N}}^-(g) \geq 0$ and proposed the non-negative risk estimator for PU learning:

$$\tilde{R}_{\mathrm{PU}}(g) = \pi \hat{R}_{\mathrm{P}}^+(g) + \max\{0, \hat{R}_{\mathrm{U}}^-(g) - \pi \hat{R}_{\mathrm{P}}^-(g)\}. \tag{3}$$

In terms of implementation, stochastic optimization was used and when $r = \hat{R}_{\mathrm{U}}^-(g) - \pi \hat{R}_{\mathrm{P}}^-(g)$ becomes smaller than some threshold value $-\beta$ for a mini-batch, they performed a step of gradient ascent along ∇r to make the mini-batch less overfitted.

2.3. PNU Classification

In semi-supervised learning (PNU classification), P, N and U data are all available. An abundance of works have been dedicated to solving this problem. Here we in particular introduce the PNU risk estimator proposed by Sakai et al. (2017). By directly leveraging U data for risk estimation, it is the most comparable to our method. The PNU risk is simply defined as a linear combination of PN and PU/NU risks. Let us just consider the case where PN and PU risks are combined, then for some $\gamma \in [0, 1]$, the PNU risk estimator is expressed as

$$\begin{aligned}\hat{R}_{\mathrm{PNU}}^{\gamma}(g) &= \gamma \hat{R}_{\mathrm{PN}}(g) + (1 - \gamma)\hat{R}_{\mathrm{PU}}(g) \\ &= \pi \hat{R}_{\mathrm{P}}^+(g) + \gamma(1 - \pi)\hat{R}_{\mathrm{N}}^-(g) \\ &\quad + (1 - \gamma)(\hat{R}_{\mathrm{U}}^-(g) - \pi \hat{R}_{\mathrm{P}}^-(g)).\end{aligned} \tag{4}$$

We can again consider the non-negative correction by forcing the term $\gamma(1 - \pi)\hat{R}_{\mathrm{N}}^-(g) + (1 - \gamma)(\hat{R}_{\mathrm{U}}^-(g) - \pi \hat{R}_{\mathrm{P}}^-(g))$ to be non-negative. In the rest of the paper, we refer to the resulting algorithm as non-negative PNU (nnPNU) learning (see Supplementary Material D.4 for an alternative definition of nnPNU and the corresponding results).

2.4. PUbN Classification

In this paper, we study the problem of PUbN learning. It differs from usual semi-supervised learning in the fact that labeled N data are not fully representative of the underlying N distribution $p_{\mathrm{N}}(\boldsymbol{x})$. To take this point into account, we introduce a latent random variable s and consider the joint distribution $p(\boldsymbol{x}, y, s)$ with constraint $p(s = +1 \mid \boldsymbol{x}, y = +1) = 1$. Equivalently, $p(y = -1 \mid \boldsymbol{x}, s = -1) = 1$. Let $\rho = p(y = -1, s = +1)$. Both π and ρ are assumed known throughout the paper. In practice they often need to be estimated from data (Jain et al., 2016; Ramaswamy et al., 2016; du Plessis et al., 2017). In place of ordinary N data we collect a set of bN samples

$$\mathcal{X}_{\mathrm{bN}} = \{\boldsymbol{x}_i^{\mathrm{bN}}\}_{i=1}^{n_{\mathrm{bN}}} \sim p_{\mathrm{bN}}(\boldsymbol{x}) = p(\boldsymbol{x}|y = -1, s = +1).$$

For instance, in text classification, if our bN data is composed of a small set of all possible N topics, $s = +1$ means

that a sample is either from these topics that make up the bN set or in the P class. The goal remains the same: we would like to minimize the classification risk (1).

3. Method

In this section, we propose a risk estimator for PUbN classification and establish an estimation error bound for the proposed method. Finally we show how our method can be applied to PU learning as a special case when no bN data are available.

3.1. Risk Estimator

Let $R^-_{\mathrm{bN}}(g) = \mathbb{E}_{x\sim p_{\mathrm{bN}}(\boldsymbol{x})}[\ell(-g(\boldsymbol{x}))]$ and $R^-_{s=-1}(g) = \mathbb{E}_{x\sim p(\boldsymbol{x}|s=-1)}[\ell(-g(\boldsymbol{x}))]$. Since $p(\boldsymbol{x}, y=-1) = p(\boldsymbol{x}, y=-1, s=+1) + p(\boldsymbol{x}, s=-1)$, we have

$$R(g) = \pi R^+_{\mathrm{P}}(g) + \rho R^-_{\mathrm{bN}}(g) + (1-\pi-\rho)R^-_{s=-1}(g). \quad (5)$$

The first two terms on the right-hand side of the equation can be approximated directly from data by $\hat{R}^+_{\mathrm{P}}(g)$ and $\hat{R}^-_{\mathrm{bN}}(g) = \frac{1}{n_{\mathrm{bN}}}\sum_{i=1}^{n_{\mathrm{bN}}} \ell(-g(\boldsymbol{x}^{\mathrm{bN}}_i))$. We therefore focus on the third term $\bar{R}^-_{s=-1}(g) = (1-\pi-\rho)R^-_{s=-1}(g)$. Our approach is mainly based on the following theorem. We relegate all proofs to the Supplementary Material.

Theorem 1. *Let $\sigma(\boldsymbol{x}) = p(s=+1 \mid \boldsymbol{x})$. For all $\eta \in [0,1]$ and $h : \mathbb{R}^d \to [0,1]$ satisfying the condition $h(\boldsymbol{x}) > \eta \Rightarrow \sigma(\boldsymbol{x}) > 0$, the risk $\bar{R}^-_{s=-1}(g)$ can be expressed as*

$$\begin{aligned}\bar{R}^-_{s=-1}(g) = {}& \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[\mathbb{1}_{h(\boldsymbol{x})\le\eta}\,\ell(-g(\boldsymbol{x}))(1-\sigma(\boldsymbol{x}))] \\ &+ \pi\,\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{P}}(\boldsymbol{x})}\left[\mathbb{1}_{h(\boldsymbol{x})>\eta}\,\ell(-g(\boldsymbol{x}))\frac{1-\sigma(\boldsymbol{x})}{\sigma(\boldsymbol{x})}\right] \\ &+ \rho\,\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{bN}}(\boldsymbol{x})}\left[\mathbb{1}_{h(\boldsymbol{x})>\eta}\,\ell(-g(\boldsymbol{x}))\frac{1-\sigma(\boldsymbol{x})}{\sigma(\boldsymbol{x})}\right]. \end{aligned} \quad (6)$$

In the theorem, $\bar{R}^-_{s=-1}(g)$ is decomposed into three terms, and when the expectation is substituted with the average over training samples, these three terms are approximated respectively using data from $\mathcal{X}_{\mathrm{U}}$, $\mathcal{X}_{\mathrm{P}}$ and $\mathcal{X}_{\mathrm{bN}}$. The choice of h and η is thus very crucial because it determines what each of the three terms tries to capture in practice. Ideally, we would like h to be an approximation of σ. Then, for $\boldsymbol{x}$ such that $h(\boldsymbol{x})$ is close to 1, $\sigma(\boldsymbol{x})$ is close to 1, so the last two terms on the right-hand side of the equation can be reasonably evaluated using $\mathcal{X}_{\mathrm{P}}$ and $\mathcal{X}_{\mathrm{bN}}$ (i.e., samples drawn from $p(\boldsymbol{x} \mid s=+1)$). On the contrary, if $h(\boldsymbol{x})$ is small, $\sigma(\boldsymbol{x})$ is small and such samples can be hardly found in $\mathcal{X}_{\mathrm{P}}$ or $\mathcal{X}_{\mathrm{bN}}$. Consequently the first term appeared in the decomposition is approximated with the help of $\mathcal{X}_{\mathrm{U}}$. Finally, in the empirical risk minimization paradigm, η becomes a hyperparameter that controls how important U data is against P and bN data when we evaluate $\bar{R}^-_{s=-1}(g)$. The larger η is, the more attention we would pay to U data.

One may be curious about why we do not simply approximate the whole risk using only U samples, that is, set η to 1. There are two main reasons. On one hand, if we have a very small U set, which means $n_{\mathrm{U}} \ll n_{\mathrm{P}}$ and $n_{\mathrm{U}} \ll n_{\mathrm{bN}}$, approximating a part of the risk with labeled samples should help us reduce the estimation error. This may seem unrealistic but sometimes unbiased U samples can also be difficult to collect (Ishida et al., 2018). On the other hand, more importantly, we have empirically observed that when the model of g is highly flexible, even a sample regarded as N with small weight gets classified as N in the latter stage of training and performance of the resulting classifier can thus be severely degraded. Introducing η alleviates this problem by avoiding treating all U data as N samples.

As σ is not available in reality, we propose to replace σ by its estimate $\hat{\sigma}$ in (6). We further substitute h with the same estimate and obtain the following expression:

$$\begin{aligned}\bar{R}^-_{s=-1,\eta,\hat{\sigma}}(g) = {}& \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[\mathbb{1}_{\hat{\sigma}(\boldsymbol{x})\le\eta}\,\ell(-g(\boldsymbol{x}))(1-\hat{\sigma}(\boldsymbol{x}))] \\ &+ \pi\,\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{P}}(\boldsymbol{x})}\left[\mathbb{1}_{\hat{\sigma}(\boldsymbol{x})>\eta}\,\ell(-g(\boldsymbol{x}))\frac{1-\hat{\sigma}(\boldsymbol{x})}{\hat{\sigma}(\boldsymbol{x})}\right] \\ &+ \rho\,\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{bN}}(\boldsymbol{x})}\left[\mathbb{1}_{\hat{\sigma}(\boldsymbol{x})>\eta}\,\ell(-g(\boldsymbol{x}))\frac{1-\hat{\sigma}(\boldsymbol{x})}{\hat{\sigma}(\boldsymbol{x})}\right]. \end{aligned}$$

We notice that $\bar{R}_{s=-1,\eta,\hat{\sigma}}$ depends both on η and $\hat{\sigma}$. It can be directly approximated from data by

$$\begin{aligned}\hat{\bar{R}}_{s=-1,\eta,\hat{\sigma}}(g) = {}& \frac{1}{n_{\mathrm{U}}}\sum_{i=1}^{n_{\mathrm{U}}}\left[\mathbb{1}_{\hat{\sigma}(\boldsymbol{x}^{\mathrm{U}}_i)\le\eta}\,\ell(-g(\boldsymbol{x}^{\mathrm{U}}_i))(1-\hat{\sigma}(\boldsymbol{x}^{\mathrm{U}}_i))\right] \\ &+ \frac{\pi}{n_{\mathrm{P}}}\sum_{i=1}^{n_{\mathrm{P}}}\left[\mathbb{1}_{\hat{\sigma}(\boldsymbol{x}^{\mathrm{P}}_i)>\eta}\,\ell(-g(\boldsymbol{x}^{\mathrm{P}}_i))\frac{1-\hat{\sigma}(\boldsymbol{x}^{\mathrm{P}}_i)}{\hat{\sigma}(\boldsymbol{x}^{\mathrm{P}}_i)}\right] \\ &+ \frac{\rho}{n_{\mathrm{bN}}}\sum_{i=1}^{n_{\mathrm{bN}}}\left[\mathbb{1}_{\hat{\sigma}(\boldsymbol{x}^{\mathrm{bN}}_i)>\eta}\,\ell(-g(\boldsymbol{x}^{\mathrm{bN}}_i))\frac{1-\hat{\sigma}(\boldsymbol{x}^{\mathrm{bN}}_i)}{\hat{\sigma}(\boldsymbol{x}^{\mathrm{bN}}_i)}\right]. \end{aligned}$$

We are now able to derive the empirical version of Equation (5) as

$$\hat{R}_{\mathrm{PUbN},\eta,\hat{\sigma}}(g) = \pi\hat{R}^+_{\mathrm{P}}(g) + \rho\hat{R}^-_{\mathrm{bN}}(g) + \hat{\bar{R}}^-_{s=-1,\eta,\hat{\sigma}}(g). \quad (7)$$

3.2. Practical Implementation

To complete our algorithm, we need to be able to estimate σ and find appropriate η. Given that the value of η can be hard to tune, we introduce another intermediate hyperparameter τ and choose η such that $\#\{x \in \mathcal{X}_{\mathrm{U}} \mid \hat{\sigma}(x) \le \eta\} = \lfloor \tau(1-\pi-\rho)n_{\mathrm{U}} \rfloor$, where $\lfloor\cdot\rfloor$ is the floor function. The number $\tau(1-\pi-\rho)$ is then the portion of unlabeled samples that are involved in the second step of our algorithm. Intuitively, we can set a higher τ and include more U samples in the minimization of (7) when we have a good

Algorithm 1 PUbN Classification

Input: data $(\mathcal{X}_{\mathrm{P}}, \mathcal{X}_{\mathrm{bN}}, \mathcal{X}_{\mathrm{U}})$, hyperparameter τ
Step 1:
Compute $\hat{\sigma}$ by minimizing an nnPU risk involving $\mathcal{X}_{\mathrm{P}}$, $\mathcal{X}_{\mathrm{bN}}$ as P data and $\mathcal{X}_{\mathrm{U}}$ as U data
Step 2:
Initialize model parameter θ of g
Choose $\mathcal{A}$ a SGD-like stochastic optimization algorithm
Set η such that

$$\#\{x \in \mathcal{X}_{\mathrm{U}} \mid \hat{\sigma}(x) \leq \eta\} = \lfloor \tau(1-\pi-\rho)n_{\mathrm{U}} \rfloor$$

for $i = 1 \ldots$ **do**
  Shuffle $(\mathcal{X}_{\mathrm{P}}, \mathcal{X}_{\mathrm{bN}}, \mathcal{X}_{\mathrm{U}})$ into M mini-batches
  for each mini-batch $(\mathcal{X}_{\mathrm{P}}^j, \mathcal{X}_{\mathrm{bN}}^j, \mathcal{X}_{\mathrm{U}}^j)$ **do**
    Compute the corresponding $\hat{R}_{\mathrm{PUbN},\eta,\hat{\sigma}}(g)$
    Use $\mathcal{A}$ to update θ with the gradient information $\nabla_\theta \hat{R}_{\mathrm{PUbN},\eta,\hat{\sigma}}(g)$
  end for
end for
Return: θ minimizing the validation loss

estimate $\hat{\sigma}$ and otherwise we should prefer a smaller τ to reduce the negative effect that can be caused by the use of $\hat{\sigma}$ of poor quality. The use of validation data to select the final τ should also be prioritized as what we do in the experimental part.

Estimating σ. If we regard s as a class label, the problem of estimating σ is then equivalent to training a probabilistic classifier separating the classes with $s = +1$ and $s = -1$. Upon noting that $(\pi+\rho)\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|s=+1)}[\ell(\epsilon g(x))] = \pi\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{P}}(\boldsymbol{x})}[\ell(\epsilon g(x))] + \rho\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{bN}}(\boldsymbol{x})}[\ell(\epsilon g(x))]$ for $\epsilon \in \{+1,-1\}$, it is straightforward to apply nnPU learning with availability of $\mathcal{X}_{\mathrm{P}}$, $\mathcal{X}_{\mathrm{bN}}$ and $\mathcal{X}_{\mathrm{U}}$ to minimize $\mathbb{E}_{(\boldsymbol{x},s)\sim p(\boldsymbol{x},s)}[\ell(sg(\boldsymbol{x}))]$. In other words, here we regard $\mathcal{X}_{\mathrm{P}}$ and $\mathcal{X}_{\mathrm{bN}}$ as P and $\mathcal{X}_{\mathrm{U}}$ as U, and attempt to solve a PU learning problem by applying nnPU. Since we are interested in the class-posterior probabilities, we minimize the risk with respect to the logistic loss and apply the sigmoid function to the output of the model to get $\hat{\sigma}(\boldsymbol{x})$. However, the above risk estimator accepts any reasonable $\hat{\sigma}$ and we are not limited to using nnPU for computing $\hat{\sigma}$. For example, the least-squares fitting approach proposed by Kanamori et al. (2009) for direct density ratio estimation can also be adapted to solving the problem.

To handle large datasets, it is preferable to adopt stochastic optimization algorithms to minimize $\hat{R}_{\mathrm{PUbN},\eta,\hat{\sigma}}(g)$.

3.3. Estimation Error Bound

Here we establish an estimation error bound for the proposed method. Let $\mathcal{G}$ be the function class from which we find a function. The Rademacher complexity of $\mathcal{G}$ for the samples of size n drawn from $q(\boldsymbol{x})$ is defined as

$$\mathfrak{R}_{n,q}(\mathcal{G}) = \mathbb{E}_{\mathcal{X}\sim q^n}\mathbb{E}_{\xi}\left[\sup_{g\in\mathcal{G}}\frac{1}{n}\sum_{x_i\in\mathcal{X}}\xi_i g(\boldsymbol{x}_i)\right],$$

where $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ and $\xi = \{\xi_1, \ldots, \xi_n\}$ with each $\boldsymbol{x}_i$ drawn from $q(\boldsymbol{x})$ and ξ_i as a Rademacher variable (Mohri et al., 2012). In the following we will assume that $\mathfrak{R}_{n,q}(\mathcal{G})$ vanishes asymptotically as $n \to \infty$. This holds for most of the common choices of $\mathcal{G}$ if proper regularization is considered (Bartlett & Mendelson, 2002; Golowich et al., 2018). Assume additionally the existence of $C_g > 0$ such that $\sup_{g\in\mathcal{G}} \|g\|_\infty \leq C_g$ as well as $C_\ell > 0$ such that $\sup_{|z|\leq C_g} \ell(z) \leq C_\ell$. We also assume that ℓ is Lipschitz continuous on the interval $[-C_g, C_g]$ with a Lipschitz constant L_ℓ.

Theorem 2. *Let $g^* = \arg\min_{g\in\mathcal{G}} R(g)$ be the true risk minimizer and $\hat{g}_{\mathrm{PUbN},\eta,\hat{\sigma}} = \arg\min_{g\in\mathcal{G}} \hat{R}_{\mathrm{PUbN},\eta,\hat{\sigma}}(g)$ be the PUbN empirical risk minimizer. We suppose that $\hat{\sigma}$ is a fixed function independent of data used to compute $\hat{R}_{\mathrm{PUbN},\eta,\hat{\sigma}}(g)$ and $\eta \in (0,1]$. Let $\zeta = p(\hat{\sigma}(\boldsymbol{x}) \leq \eta)$ and $\epsilon = \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[|\hat{\sigma}(\boldsymbol{x}) - \sigma(\boldsymbol{x})|^2]$. Then for any $\delta > 0$, with probability at least $1-\delta$,*

$$\begin{aligned} &R(\hat{g}_{\mathrm{PUbN},\eta,\hat{\sigma}}) - R(g^*) \\ &\leq 4L_\ell\mathfrak{R}_{n_{\mathrm{U}},p}(\mathcal{G}) + \frac{4\pi L_\ell}{\eta}\mathfrak{R}_{n_{\mathrm{P}},p_{\mathrm{P}}}(\mathcal{G}) + \frac{4\rho L_\ell}{\eta}\mathfrak{R}_{n_{\mathrm{bN}},p_{\mathrm{bN}}}(\mathcal{G}) \\ &\quad + 2C_\ell\sqrt{\frac{\ln(6/\delta)}{2n_{\mathrm{U}}}} + \frac{2\pi C_\ell}{\eta}\sqrt{\frac{\ln(6/\delta)}{2n_{\mathrm{P}}}} + \frac{2\rho C_\ell}{\eta}\sqrt{\frac{\ln(6/\delta)}{2n_{\mathrm{bN}}}} \\ &\quad + 2C_\ell\sqrt{\zeta\epsilon} + \frac{2C_\ell}{\eta}\sqrt{(1-\zeta)\epsilon}. \end{aligned}$$

Theorem 2 combined with the Borel-Cantelli lemma implies that as $n_{\mathrm{P}} \to \infty$, $n_{\mathrm{bN}} \to \infty$ and $n_{\mathrm{U}} \to \infty$, the inequality $\limsup R(\hat{g}_{\mathrm{PUbN},\eta,\hat{\sigma}}) - R(g^*) \leq 2C_\ell\sqrt{\zeta\epsilon} + 2(C_\ell/\eta)\sqrt{(1-\zeta)\epsilon}$ holds almost surely. Furthermore, if there is $C_{\mathcal{G}} > 0$ such that $\mathfrak{R}_{n,q}(\mathcal{G}) \leq C_{\mathcal{G}}/\sqrt{n}$ [2], the convergence of $[(R(\hat{g}_{\mathrm{PUbN},\eta,\hat{\sigma}}) - R(g^*)) - (2C_\ell\sqrt{\zeta\epsilon} + 2(C_\ell/\eta)\sqrt{(1-\zeta)\epsilon})]^+$ to 0 is in $\mathcal{O}_p(1/\sqrt{n_{\mathrm{P}}} + 1/\sqrt{n_{\mathrm{bN}}} + 1/\sqrt{n_{\mathrm{U}}})$, where $\mathcal{O}_p$ denotes the order in probability and $[\cdot]^+ = \max\{0, \cdot\}$. As for ϵ, knowing that $\hat{\sigma}$ is also estimated from data in practice [3], apparently its value depends on both the estimation algorithm and the number of samples that are involved in the estimation process. For example, in our approach we applied nnPU with the logistic loss to obtain $\hat{\sigma}$, so the excess risk can be written as $\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\mathrm{KL}(\sigma(\boldsymbol{x})\|\hat{\sigma}(\boldsymbol{x}))$, where by abuse of notation $\mathrm{KL}(p\|q) = p\ln(p/q) + (1-$

[2] For instance, this holds for linear-in-parameter model class $\mathcal{F} = \{f(\boldsymbol{x}) = \boldsymbol{w}^\top\phi(\boldsymbol{x}) \mid \|\boldsymbol{w}\| \leq C_{\boldsymbol{w}}, \|\phi\|_\infty \leq C_\phi\}$, where $C_{\boldsymbol{w}}$ and C_ϕ are positive constants (Mohri et al., 2012).

[3] These data, according to theorem 2, must be different from those used to evaluate $\hat{R}_{\mathrm{PUbN},\eta,\hat{\sigma}}(g)$. This condition is however violated in most of our experiments. See Supplementary Material D.3 for more discussion.

$p)\ln((1-p)/(1-q))$ denotes the KL divergence between two Bernouilli distributions with parameters respectively p and q. It is known that $\epsilon = \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[|\hat{\sigma}(\boldsymbol{x}) - \sigma(\boldsymbol{x})|^2] \leq (1/2)\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\mathrm{KL}(\sigma(\boldsymbol{x})\|\hat{\sigma}(\boldsymbol{x}))$ (Zhang, 2004). The excess risk itself can be decomposed into the sum of the estimation error and the approximation error. Kiryo et al. (2017) showed that under mild assumptions the estimation error part converges to zero when the sample size increases to infinity in nnPU learning. It is however impossible to get rid of the approximation error part which is fixed once we fix the function class $\mathcal{G}$. To circumvent this problem, we can either resort to kernel-based methods with universal kernels (Zhang, 2004) or simply enlarge the function class when we get more samples.

3.4. PU Learning Revisited

In PU learning scenarios, we only have P and U data and bN data are not available. Nevertheless, if we let y play the role of s and ignore all the terms related to bN data, our algorithm is naturally applicable to PU learning. Let us name the resulting algorithm PUbN\N, then

$$\hat{R}_{\mathrm{PUbN\backslash N},\eta,\hat{\sigma}}(g) = \pi\hat{R}^{+}_{\mathrm{P}}(g) + \hat{\bar{R}}^{-}_{y=-1,\eta,\hat{\sigma}}(g),$$

where $\hat{\sigma}$ is an estimate of $p(y=+1 \mid \boldsymbol{x})$ and

$$\bar{R}^{-}_{y=-1,\eta,\hat{\sigma}}(g) = \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[\mathbb{1}_{\hat{\sigma}(\boldsymbol{x})\leq\eta}\,\ell(-g(\boldsymbol{x}))(1-\hat{\sigma}(\boldsymbol{x}))] + \pi\,\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{P}}(\boldsymbol{x})}\left[\mathbb{1}_{\hat{\sigma}(\boldsymbol{x})>\eta}\,\ell(-g(\boldsymbol{x}))\frac{1-\hat{\sigma}(\boldsymbol{x})}{\hat{\sigma}(\boldsymbol{x})}\right].$$

PUbN\N can be viewed as a variant of the traditional two-step approach in PU learning which first identifies possible N data in U data and then perform ordinary PN classification to distinguish P data from the identified N data. However, being based on state-of-the-art nnPU learning, our method is more promising than other similar algorithms. Moreover, by explicitly considering the posterior $p(y=+1 \mid \boldsymbol{x})$, we attempt to correct the bias induced by the fact of only taking into account confident negative samples. The benefit of using an unbiased risk estimator is that the resulting algorithm is always statistically consistent, i.e., the estimation error converges in probability to zero as the number of samples grows to infinity.

4. Experiments

In this section, we experimentally investigate the proposed method and compare its performance against several baseline methods.

4.1. Basic Setup

We focus on training neural networks with stochastic optimization. For simplicity, in an experiment, $\hat{\sigma}$ and g always use the same model and are trained for the same number of epochs. All models are learned using AMSGrad (Reddi et al., 2018) as the optimizer and the logistic loss as the surrogate loss unless otherwise specified. In all the experiments, an additional validation set, equally composed of P, U and bN data, is sampled for both hyperparameter tuning and choosing the model parameters with the lowest validation loss among those obtained after every epoch. Regarding the computation of the validation loss, we use the PU risk estimator (2) with the sigmoid loss for g and an empirical approximation of $\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[|\hat{\sigma}(\boldsymbol{x})-\sigma(\boldsymbol{x})|^2] - \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[\sigma(\boldsymbol{x})^2]$ for $\hat{\sigma}$ (see Supplementary Material B).

4.2. Effectiveness of the Algorithm

We assess the performance of the proposed method on three benchmark datasets: MNIST, CIFAR-10 and 20 Newsgroups. Experimental details are given in Supplementary Material C. To recapitulate, for the three datasets we respectively use a 4-layer ConvNet, PreAct ResNet-18 (He et al., 2016) and a 3-layer fully connected neural network. On 20 Newsgroups text features are generated thanks to the use of ELMo word embedding (Peters et al., 2018). Since all the three datasets are originally designed for multiclass classification, we group different categories together to form a binary classification problem.

Baselines. When $\mathcal{X}_{\mathrm{bN}}$ is given, two baseline methods are considered. The first one is nnPNU adapted from (4). In the second method, named as PU→PN, we train two binary classifiers: one is learned with nnPU while we regard s as the class label, and the other is learned from $\mathcal{X}_{\mathrm{P}}$ and $\mathcal{X}_{\mathrm{bN}}$ to separate P samples from bN samples. A sample is classified in the P class only if it is so classified by the two classifiers. When $\mathcal{X}_{\mathrm{bN}}$ is not available, nnPU is compared with the proposed PUbN\N.

Sampling bN Data. To sample $\mathcal{X}_{\mathrm{bN}}$, we suppose that the bias of N data is caused by a latent prior probability change (Sugiyama & Storkey, 2007; Hu et al., 2018) in the N class. Let $z \in \mathcal{Z} = \{1,\ldots,S\}$ be some latent variable which we call a latent category, where S is a constant. It is assumed

$$p(\boldsymbol{x} \mid z, y=-1) = p(\boldsymbol{x} \mid z, y=-1, s=+1),$$
$$p(z \mid y=-1) \neq p(z \mid y=-1, s=+1).$$

In the experiments, the latent categories are the original class labels of the datasets. Concrete definitions of $\mathcal{X}_{\mathrm{bN}}$ with experimental results are summarized in Table 1.

Results. Overall, our proposed method consistently achieves the best or comparable performance in all the scenarios, including those of standard PU learning. Additionally, using bN data can effectively help improving the classification performance. However, the choice of algorithm is essential. Both nnPNU and the naive PU→PN are

Table 1. Mean and standard deviation of misclassification rates over 10 trials for MNIST, CIFAR-10 and 20 Newsgroups under different choices of P class and bN data sampling strategies. For a same learning task, different methods are compared using the same 10 random samplings. Underlines denote that with the use of bN data the method leads to an improvement of performance according to the 5% t-test. Boldface indicates the best method in each task.

† Biased N data uniformly sampled from the indicated latent categories.

⋆ Probabilities that a sample of $\mathcal{X}_{\text{bN}}$ belongs to the latent categories [1, 3, 5, 7, 9] / [bird, cat, deer, dog, frog, horse] / [sci., soc., talk.] are [0.03, 0.15, 0.3, 0.02, 0.5] / [0.1, 0.02, 0.2, 0.08, 0.2, 0.4] / [0.1, 0.5, 0.4].

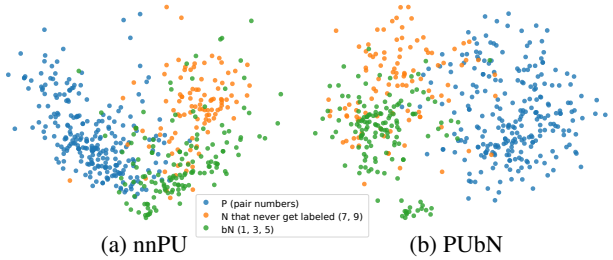
Dataset	P	biased N	ρ	nnPU/nnPNU	PUbN(\N)	PU→PN
MNIST	0, 2, 4, 6, 8	Not given	NA	5.76 ± 1.04	$\mathbf{4.64 \pm 0.62}$	NA
		1, 3, 5 †	0.3	5.33 ± 0.97	4.05 ± 0.27	$\mathbf{4.00 \pm 0.30}$
		9 > 5 > others ⋆	0.2	4.60 ± 0.65	3.91 ± 0.66	$\mathbf{3.77 \pm 0.31}$
CIFAR-10	Airplane, automobile, ship, truck	Not given	NA	12.02 ± 0.65	$\mathbf{10.70 \pm 0.57}$	NA
		Cat, dog, horse †	0.3	10.25 ± 0.38	$\mathbf{9.71 \pm 0.51}$	10.37 ± 0.65
		Horse > deer = frog > others ⋆	0.25	9.98 ± 0.53	$\mathbf{9.92 \pm 0.42}$	10.17 ± 0.35
CIFAR-10	Cat, deer, dog, horse	Not given	NA	23.78 ± 1.04	$\mathbf{21.13 \pm 0.90}$	NA
		Bird, frog †	0.2	22.00 ± 0.53	$\mathbf{18.83 \pm 0.71}$	19.88 ± 0.62
		Car, truck †	0.2	22.00 ± 0.74	$\mathbf{20.19 \pm 1.06}$	21.83 ± 1.36
20 Newsgroups	alt., comp., misc., rec.	Not given	NA	14.67 ± 0.87	$\mathbf{13.30 \pm 0.53}$	NA
		sci.†	0.21	14.69 ± 0.46	$\mathbf{13.10 \pm 0.90}$	13.58 ± 0.97
		talk.†	0.17	14.38 ± 0.74	$\mathbf{12.61 \pm 0.75}$	13.76 ± 0.66
		soc. > talk. > sci.⋆	0.1	14.41 ± 0.76	$\mathbf{12.18 \pm 0.59}$	12.92 ± 0.51

able to leverage bN data to enhance classification accuracy in only relatively few tasks. In the contrast, the proposed PUbN successfully reduce the misclassification error most of the time.

Clearly, the performance gain that we can benefit from the availability of bN data is case-dependent. On CIFAR-10, the greatest improvement is achieved when we regard mammals (i.e. cat, deer, dog and horse) as P class and drawn samples from latent categories bird and frog as labeled negative data. This is not surprising because birds and frogs are more similar to mammals than vehicles, which makes the classification harder specifically for samples from these two latent categories. By explicitly labeling these samples as N data, we allow the classifier to make better predictions for these difficult samples.

4.3. Illustration on How the Presence of bN Data Help

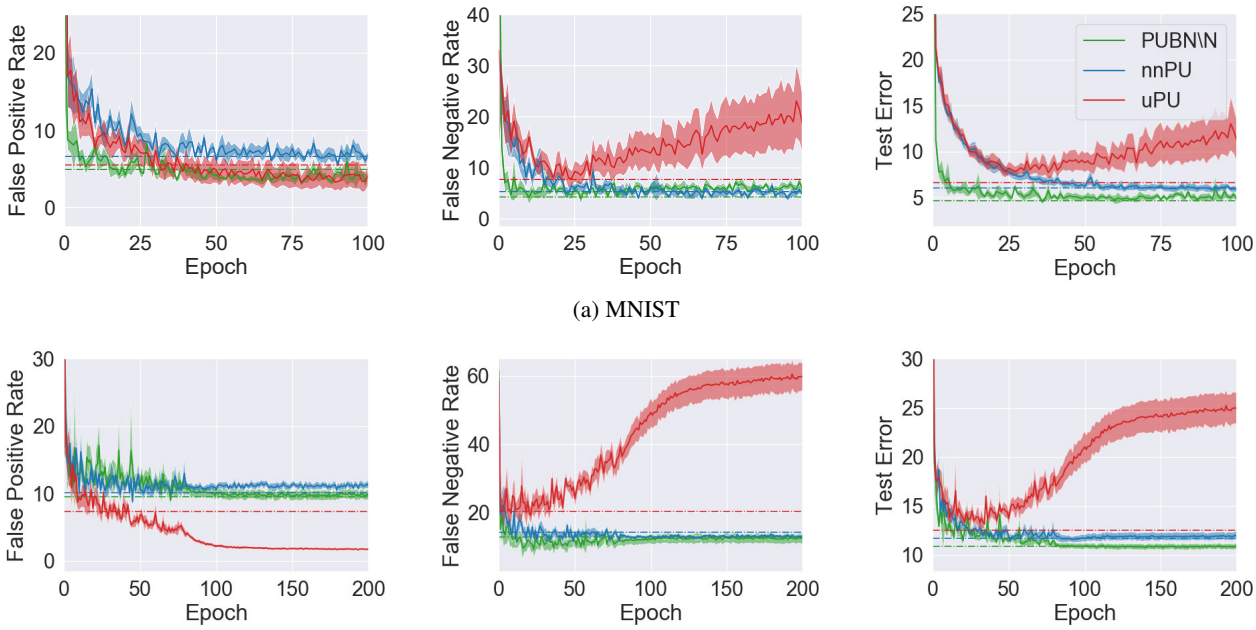
Through experiments we have demonstrated that the presence of bN data effectively helps learning a better classifier. Here we would like to provide some intuition for the reason behind this. Let us consider the MNIST learning task where $\mathcal{X}_{\text{bN}}$ is uniformly sampled from the latent categories 1, 3 and 5. We project the representations learned by the classifier (i.e., the activation values of the last hidden layer of the neural network) into a 2D plane using PCA for both nnPU and PUbN algorithms, as shown in Figure 1.



(a) nnPU (b) PUbN

Figure 1. PCA embeddings of the representations learned by the nnPU and PUbN classifiers for 500 samples from the test set in the MNIST learning task where $\mathcal{X}_{\text{bn}}$ is uniformly sampled from latent categories 1, 3 and 5.

For both nnPU and PUbN classifiers, the first two principal components account around 90% of variance. We can therefore presume that the figure depicts fairly well the learned representations. Thanks to the use of bN data, in the high-level feature space 1, 3, 5 and P data are further pushed away when we employ the proposed PUbN learning algorithm, and we are always able to separate 7, 9 from P to some extent. This explains the better performance which is achieved by PUbN learning and the benefit of incorporating bN data into the learning process.



(a) MNIST

(b) CIFAR-10, vehicles as P class

Figure 2. Comparison of uPU, nnPU and PUbN\N over two of the four PU learning tasks. For each task, means and standard deviations are computed based on the same 10 random samplings. Dashed lines indicate the corresponding values of the final classifiers (recall that at the end we select the model with the lowest validation loss out of all epochs).

4.4. Why Does PUbN\N Outperform nnPU ?

Our method, specifically designed for PUbN learning, naturally outperforms other baseline methods in this problem. Nonetheless, Table 1 equally shows that the proposed method when applied to PU learning, achieves significantly better performance than the state-of-the-art nnPU algorithm. Here we numerically investigate the reason behind this phenomenon with help of the first two PU tasks of the table.

Besides nnPU and PUbN\N, we compare with unbiased PU (uPU) learning (2). Both uPU and nnPU are learned with the sigmoid loss, learning rate 10^{-3} for MNIST and initial learning rate 10^{-4} for CIFAR-10, as uPU learning is unstable with the logistic loss. The other parts of the experiments remain unchanged. On the test sets we compute the false positive rates, false negative rates and misclassification errors for the three methods and plot them in Figure 2. We first notice that PUbN\N still outperforms nnPU trained with the sigmoid loss. In fact, the final performance of the nnPU classifier does not change much when we replace the logistic loss with the sigmoid loss.

In (Kiryo et al., 2017), the authors observed that uPU overfits training data with the risk going to negative. In other words, a large portion of U samples are classified as N. This is confirmed in our experiments by an increase of false negative rate and decrease of false positive rate. nnPU remedies the problem by introducing the non-negative risk estimator (3). While the non-negative correction successfully prevents false negative rate from going up, it also causes more N samples to be classified as P compared to uPU. However, since the gain in terms of false negative rate is enormous, at the end nnPU achieves a lower misclassification error. By further identifying possible N samples after nnPU learning, we expect that our algorithm can yield lower false positive rate than nnPU without misclassifying too many P samples as N as in the case of uPU. Figure 2 suggests that this is effectively the case. In particular, we observe that on MNIST, our method achieves the same false positive rate as uPU whereas its false negative rate is comparable to nnPU.

5. Conclusion

This paper studies the PUbN classification problem, where a binary classifier is trained on P, U and bN data. The proposed method is a two-step approach inspired from both PU learning and importance weighting. The key idea is to attribute appropriate weights to each example for evaluation of the classification risk using the three sets of data. We theoretically established an estimation error bound for the proposed risk estimator and experimentally showed that our approach successfully leveraged bN data to improve the classification performance on several real-world datasets. A variant of our algorithm was able to achieve state-of-the-art results in PU learning.

Acknowledgements

MS was supported by JST CREST Grant Number JPMJCR1403.

References

Bartlett, P. L. and Mendelson, S. Rademacher and gaussian complexities: Risk bounds and structural results. *JMLR*, 3(Nov):463–482, 2002.

Belkin, M., Niyogi, P., and Sindhwani, V. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *JMLR*, 7(Nov):2399–2434, 2006.

Chapelle, O., Schlkopf, B., and Zien, A. *Semi-Supervised Learning*. The MIT Press, 1st edition, 2010.

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

du Plessis, M. C., Niu, G., and Sugiyama, M. Analysis of learning from positive and unlabeled data. In *NeurIPS*, pp. 703–711, 2014.

du Plessis, M. C., Niu, G., and Sugiyama, M. Class-prior estimation for learning from positive and unlabeled data. *Maching Learning*, 106(4):463–492, April 2017. ISSN 0885-6125.

Elkan, C. and Noto, K. Learning classifiers from only positive and unlabeled data. In *KDD*, 2008.

Fei, G. and Liu, B. Social media text classification under negative covariate shift. In *Proc. of EMNLP*, pp. 2347–2356, 2015.

Fung, G. P. C., Yu, J. X., Lu, H., and Yu, P. S. Text classification without negative examples revisit. *TKDE*, 18(1): 6–20, 2006.

Golowich, N., Rakhlin, A., and Shamir, O. Size-independent sample complexity of neural networks. In *COLT*, pp. 297–299, 2018.

Grandvalet, Y. and Bengio, Y. Semi-supervised learning by entropy minimization. In *NeurIPS*, pp. 529–536, 2005.

He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. In *ECCV*, pp. 630–645. Springer, 2016.

Heckman, J. J. Sample selection bias as a specification error. *Econometrica*, 47(1):153–161, 1979.

Hido, S., Tsuboi, Y., Kashima, H., Sugiyama, M., and Kanamori, T. Inlier-based outlier detection via direct density ratio estimation. In *ICDM*, pp. 223–232. IEEE, 2008.

Hu, W., Niu, G., Sato, I., and Sugiyama, M. Does distributionally robust supervised learning give robust classifiers? In *ICML*, pp. 2034–2042, 2018.

Huang, J., Gretton, A., Borgwardt, K. M., Schölkopf, B., and Smola, A. J. Correcting sample selection bias by unlabeled data. In *NeurIPS*, pp. 601–608, 2007.

Ishida, T., Niu, G., and Sugiyama, M. Binary classification from positive-confidence data. In *NeurIPS*, pp. 5921–5932, 2018.

Jain, S., White, M., and Radivojac, P. Estimating the class prior and posterior from noisy positives and unlabeled data. In *NeurIPS*, pp. 2693–2701, 2016.

Kanamori, T., Hido, S., and Sugiyama, M. A least-squares approach to direct importance estimation. *JMLR*, 10: 1391–1445, 2009.

Kiryo, R., Niu, G., du Plessis, M. C., and Sugiyama, M. Positive-unlabeled learning with non-negative risk estimator. In *NeurIPS*, pp. 1675–1685, 2017.

Laine, S. and Aila, T. Temporal ensembling for semi-supervised learning. In *ICLR*, 2017.

Li, W., Guo, Q., and Elkan, C. A positive and unlabeled learning algorithm for one-class classification of remote-sensing data. *TGRS*, 49(2):717–725, 2011.

Li, X.-L., Liu, B., and Ng, S.-K. Negative training data can be harmful to text classification. In *Proc. of EMNLP*, pp. 218–228, 2010.

Liu, B., Lee, W. S., Yu, P. S., and Li, X. Partially supervised classification of text documents. In *ICML*, volume 2, pp. 387–394, 2002.

Liu, B., Dai, Y., Li, X., Lee, W. S., and Yu, P. S. Building text classifiers using positive and unlabeled examples. In *ICDM*, pp. 179–186. IEEE, 2003.

Miyato, T., Maeda, S.-i., Koyama, M., Nakae, K., and Ishii, S. Distributional smoothing with virtual adversairal training. In *ICLR*, 2016.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. *Foundations of machine learning*. The MIT press, 2012.

Mordelet, F. and Vert, J.-P. A bagging svm to learn from positive and unlabeled examples. *Pattern Recognition Letters*, 37:201–209, 2014.

Nguyen, M. N., Li, X.-L., and Ng, S.-K. Positive unlabeled learning for time series classification. In *IJCAI*, volume 11, pp. 1421–1426, 2011.

Oliver, A., Odena, A., Raffel, C. A., Cubuk, E. D., and Goodfellow, I. Realistic evaluation of deep semi-supervised learning algorithms. In *NeurIPS*, pp. 3239–3250. 2018.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. Deep contextualized word representations. In *Proc. of NAACL*, 2018.

Quionero-Candela, J., Sugiyama, M., Schwaighofer, A., and Lawrence, N. D. *Dataset Shift in Machine Learning*. The MIT Press, 2009.

Ramaswamy, H., Scott, C., and Tewari, A. Mixture proportion estimation via kernel embeddings of distributions. In *ICML*, pp. 2052–2060, 2016.

Reddi, S. J., Kale, S., and Kumar, S. On the convergence of adam and beyond. In *ICLR*, 2018.

Rücklé, A., Eger, S., Peyrard, M., and Gurevych, I. Concatenated power mean word embeddings as universal cross-lingual sentence representations. *arXiv preprint arXiv:1803.01400*, 2018.

Sakai, T., du Plessis, M. C. d., Niu, G., and Sugiyama, M. Semi-supervised classification based on classification from positive and unlabeled data. In *ICML*, volume 70, pp. 2998–3006, 2017.

Scott, C. and Blanchard, G. Novelty detection: Unlabeled data definitely help. In *AISTATS*, pp. 464–471, 2009.

Shalev-Shwartz, S. and Ben-David, S. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

Shi, H., Pan, S., Yang, J., and Gong, C. Positive and unlabeled learning via loss decomposition and centroid estimation. In *IJCAI*, pp. 2689–2695, 2018.

Shimodaira, H. Improving predictive inference under covariate shift by weighting the log-likelihood function. *JMLR*, 90(2):227–244, 2000.

Sugiyama, M. and Kawanabe, M. *Machine Learning in Non-Stationary Environments: Introduction to Covariate Shift Adaptation*. The MIT Press, Cambridge, Massachusetts, USA, 2012.

Sugiyama, M. and Storkey, A. J. Mixture regression for covariate shift. In *NeurIPS*, pp. 1337–1344, 2007.

Sugiyama, M., Nakajima, S., Kashima, H., Buenau, P. V., and Kawanabe, M. Direct importance estimation with model selection and its application to covariate shift adaptation. In *NeurIPS*, pp. 1433–1440, 2008.

Ward, G. A., Hastie, T. J., Barry, S. T., Elith, J., and Leathwick, J. R. Presence-only data and the em algorithm. *Biometrics*, 65 2:554–63, 2009.

Zadrozny, B. Learning and evaluating classifiers under sample selection bias. In *ICML*, pp. 903–910, 2004.

Zhang, T. Statistical behavior and consistency of classification methods based on convex risk minimization. *Annals of Statistics*, pp. 56–85, 2004.

Zuluaga, M., Hush, D., J F Delgado Leyton, E., Hernandez Hoyos, M., and Orkisz, M. Learning from only positive and unlabeled data to detect lesions in vascular ct images. In *MICCAI*, volume LNCS 6893, pp. 9–16, 2011.

Classification from Positive, Unlabeled and Biased Negative Data: Supplementary Material

Appendix

A. Proofs

A.1. Proof of Theorem 1

We notice that $(1-\pi-\rho)p(\boldsymbol{x} \mid s=-1) = p(\boldsymbol{x}, s=-1)$ and that when $h(\boldsymbol{x}) > \eta$, we have $p(s=+1 \mid \boldsymbol{x}) = \sigma(\boldsymbol{x}) > 0$, which allows us to write $p(s=-1 \mid \boldsymbol{x}) = (p(s=-1 \mid \boldsymbol{x})/p(s=+1 \mid \boldsymbol{x}))p(s=+1 \mid \boldsymbol{x})$. We can thus decompose $\bar{R}^{-}_{s=-1}(g)$ as following:

$$\begin{aligned}
\bar{R}^{-}_{s=-1}(g) &= \int \ell(-g(\boldsymbol{x}))p(\boldsymbol{x}, s=-1)\,dx \\
&= \int \mathbb{1}_{h(\boldsymbol{x})\leq\eta}\,\ell(-g(\boldsymbol{x}))p(\boldsymbol{x}, s=-1)\,dx \\
&\quad + \int \mathbb{1}_{h(\boldsymbol{x})>\eta}\,\ell(-g(\boldsymbol{x}))p(\boldsymbol{x}, s=-1)\,dx \\
&= \int \mathbb{1}_{h(\boldsymbol{x})\leq\eta}\,\ell(-g(\boldsymbol{x}))\frac{p(\boldsymbol{x}, s=-1)}{p(\boldsymbol{x})}p(\boldsymbol{x})\,dx \\
&\quad + \int \mathbb{1}_{h(\boldsymbol{x})>\eta}\,\ell(-g(\boldsymbol{x}))\frac{p(\boldsymbol{x}, s=-1)}{p(\boldsymbol{x}, s=+1)}p(\boldsymbol{x}, s=+1)\,dx.
\end{aligned}$$

By writing $p(\boldsymbol{x}, s=-1) = p(s=-1 \mid \boldsymbol{x})p(\boldsymbol{x}) = (1-\sigma(\boldsymbol{x}))p(\boldsymbol{x})$ and $p(\boldsymbol{x}, s=+1) = p(s=+1 \mid \boldsymbol{x})p(\boldsymbol{x}) = \sigma(\boldsymbol{x})p(\boldsymbol{x})$, we have

$$\begin{aligned}
\bar{R}^{-}_{s=-1}(g) &= \int \mathbb{1}_{h(\boldsymbol{x})\leq\eta}\,\ell(-g(\boldsymbol{x}))(1-\sigma(\boldsymbol{x}))p(\boldsymbol{x})\,dx \\
&\quad + \int \mathbb{1}_{h(\boldsymbol{x})>\eta}\,\ell(-g(\boldsymbol{x}))\frac{1-\sigma(\boldsymbol{x})}{\sigma(\boldsymbol{x})}p(\boldsymbol{x}, s=+1)\,dx.
\end{aligned}$$

We obtain Equation (6) after replacing $p(\boldsymbol{x}, s=+1)$ by $\pi p(x \mid y=+1) + \rho p(x \mid y=-1, s=+1)$.

A.2. Proof of Theorem 2

For $\hat{\sigma}$ and η given, let us define

$$R_{\mathrm{PUbN},\eta,\hat{\sigma}}(g) = \pi R^{+}_{\mathrm{P}}(g) + \rho R^{-}_{\mathrm{bN}}(g) + \bar{R}^{-}_{s=-1,\eta,\hat{\sigma}}(g).$$

The following lemma establishes the uniform deviation bound from $\hat{R}_{\mathrm{PUbN},\eta,\hat{\sigma}}$ to $R_{\mathrm{PUbN},\eta,\hat{\sigma}}$.

Lemma 1. *Let $\hat{\sigma} : \mathbb{R}^d \to [0,1]$ be a fixed function independent of data used to compute $\hat{R}_{\mathrm{PUbN},\eta,\hat{\sigma}}$ and $\eta \in (0,1]$. For any $\delta > 0$, with probability at least $1-\delta$,*

$$\begin{aligned}
&\sup_{g\in\mathcal{G}} |\hat{R}^{-}_{\mathrm{PUbN},\eta,\hat{\sigma}}(g) - R_{\mathrm{PUbN},\eta,\hat{\sigma}}(g)| \\
&\quad \leq 2L_\ell \mathfrak{R}_{n_{\mathrm{U}},p}(\mathcal{G}) + \frac{2\pi L_\ell}{\eta}\mathfrak{R}_{n_{\mathrm{P}},p_{\mathrm{P}}}(\mathcal{G}) + \frac{2\rho L_\ell}{\eta}\mathfrak{R}_{n_{\mathrm{bN}},p_{\mathrm{bN}}}(\mathcal{G}) + C_\ell\sqrt{\frac{\ln(6/\delta)}{2n_{\mathrm{U}}}} + \frac{\pi C_\ell}{\eta}\sqrt{\frac{\ln(6/\delta)}{2n_{\mathrm{P}}}} + \frac{\rho C_\ell}{\eta}\sqrt{\frac{\ln(6/\delta)}{2n_{\mathrm{bN}}}}.
\end{aligned}$$

Proof. For ease of notation, let

$$R_{\mathrm{P}}(g) = \mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{P}}(\boldsymbol{x})}\left[\ell(g(\boldsymbol{x})) + \mathbb{1}_{\hat{\sigma}(\boldsymbol{x})>\eta}\,\ell(-g(\boldsymbol{x}))\frac{1-\hat{\sigma}(\boldsymbol{x})}{\hat{\sigma}(\boldsymbol{x})}\right],$$

$$R_{\mathrm{bN}}(g) = \mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{bN}}(\boldsymbol{x})}\left[\ell(-g(\boldsymbol{x}))\left(1 + \mathbb{1}_{\hat{\sigma}(\boldsymbol{x})>\eta}\frac{1-\hat{\sigma}(\boldsymbol{x})}{\hat{\sigma}(\boldsymbol{x})}\right)\right],$$

$$R_{\mathrm{U}}(g) = \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\left[\mathbb{1}_{\hat{\sigma}(\boldsymbol{x})\leq\eta}\,\ell(-g(\boldsymbol{x}))(1-\hat{\sigma}(\boldsymbol{x}))\right],$$

$$\hat{R}_{\mathrm{P}}(g) = \frac{1}{n_{\mathrm{P}}}\sum_{i=1}^{n_{\mathrm{P}}}\left[\ell(g(\boldsymbol{x}_i^{\mathrm{P}})) + \mathbb{1}_{\hat{\sigma}(\boldsymbol{x}_i^{\mathrm{P}})>\eta}\,\ell(-g(\boldsymbol{x}_i^{\mathrm{P}}))\frac{1-\hat{\sigma}(\boldsymbol{x}_i^{\mathrm{P}})}{\hat{\sigma}(\boldsymbol{x}_i^{\mathrm{P}})}\right],$$

$$\hat{R}_{\mathrm{bN}}(g) = \frac{1}{n_{\mathrm{bN}}}\sum_{i=1}^{n_{\mathrm{bN}}}\left[\ell(-g(\boldsymbol{x}_i^{\mathrm{bN}}))\left(1 + \mathbb{1}_{\hat{\sigma}(\boldsymbol{x}_i^{\mathrm{bN}})>\eta}\frac{1-\hat{\sigma}(\boldsymbol{x}_i^{\mathrm{bN}})}{\hat{\sigma}(\boldsymbol{x}_i^{\mathrm{bN}})}\right)\right],$$

$$\hat{R}_{\mathrm{U}}(g) = \frac{1}{n_{\mathrm{U}}}\sum_{i=1}^{n_{\mathrm{U}}}\left[\mathbb{1}_{\hat{\sigma}(\boldsymbol{x}_i^{\mathrm{U}})\leq\eta}\,\ell(-g(\boldsymbol{x}_i^{\mathrm{U}}))(1-\hat{\sigma}(\boldsymbol{x}_i^{\mathrm{U}}))\right].$$

From the sub-additivity of the supremum operator, we have

$$\begin{aligned}&\sup_{g\in\mathcal{G}}|\hat{R}^{-}_{\mathrm{PUbN},\eta,\hat{\sigma}}(g) - R_{\mathrm{PUbN},\eta,\hat{\sigma}}(g)|\\ &\quad\leq \pi\sup_{g\in\mathcal{G}}|\hat{R}_{\mathrm{P}}(g) - R_{\mathrm{P}}(g)| + \rho\sup_{g\in\mathcal{G}}|\hat{R}_{\mathrm{bN}}(g) - R_{\mathrm{bN}}(g)| + \sup_{g\in\mathcal{G}}|\hat{R}_{\mathrm{U}}(g) - R_{\mathrm{U}}(g)|.\end{aligned}$$

As a consequence, to conclude the proof, it suffices to prove that with probability at least $1-\delta/3$, the following bounds hold separately:

$$\sup_{g\in\mathcal{G}}|\hat{R}_{\mathrm{P}}(g) - R_{\mathrm{P}}(g)| \leq \frac{2L_\ell}{\eta}\mathfrak{R}_{n_{\mathrm{P}},p_{\mathrm{P}}}(\mathcal{G}) + \frac{C_\ell}{\eta}\sqrt{\frac{\ln(6/\delta)}{2n_{\mathrm{P}}}}, \tag{8}$$

$$\sup_{g\in\mathcal{G}}|\hat{R}_{\mathrm{bN}}(g) - R_{\mathrm{bN}}(g)| \leq \frac{2L_\ell}{\eta}\mathfrak{R}_{n_{\mathrm{bN}},p_{\mathrm{bN}}}(\mathcal{G}) + \frac{C_\ell}{\eta}\sqrt{\frac{\ln(6/\delta)}{2n_{\mathrm{bN}}}}, \tag{9}$$

$$\sup_{g\in\mathcal{G}}|\hat{R}_{\mathrm{U}}(g) - R_{\mathrm{U}}(g)| \leq 2L_\ell\mathfrak{R}_{n_{\mathrm{U}},p}(\mathcal{G}) + C_\ell\sqrt{\frac{\ln(6/\delta)}{2n_{\mathrm{U}}}}. \tag{10}$$

Below we prove (8). (9) and (10) are proven similarly.

Let $\phi_{\boldsymbol{x}}:\mathbb{R}\to\mathbb{R}_+$ be the function defined by $\phi_{\boldsymbol{x}}: z\mapsto \ell(z) + \mathbb{1}_{\hat{\sigma}(\boldsymbol{x})>\eta}\,\ell(-z)((1-\hat{\sigma}(\boldsymbol{x}))/\hat{\sigma}(\boldsymbol{x}))$. For $\boldsymbol{x}\in\mathbb{R}^d, g\in\mathcal{G}$, since $\ell(g(\boldsymbol{x}))\in[0,C_\ell]$, $\ell(-g(\boldsymbol{x}))\in[0,C_\ell]$ and $\mathbb{1}_{\hat{\sigma}(\boldsymbol{x})>\eta}((1-\hat{\sigma}(\boldsymbol{x}))/\hat{\sigma}(\boldsymbol{x}))\in[0,(1-\eta)/\eta]$, we always have $\phi_{\boldsymbol{x}}(g(\boldsymbol{x}))\in[0,C_\ell/\eta]$. Following the proof of Theorem 3.1 in (Mohri et al., 2012), it is then straightforward to show that with probability at least $1-\delta/3$, it holds that

$$\sup_{g\in\mathcal{G}}|\hat{R}_{\mathrm{P}}(g) - R_{\mathrm{P}}(g)| \leq 2\,\mathbb{E}_{\mathcal{X}_{\mathrm{P}}\sim p_{\mathrm{P}}^{n_{\mathrm{P}}}}\mathbb{E}_\theta\left[\sup_{g\in\mathcal{G}}\frac{1}{n_{\mathrm{P}}}\sum_{i=1}^{n_{\mathrm{P}}}\theta_i\phi_{\boldsymbol{x}_i}(g(\boldsymbol{x}_i))\right] + \frac{C_\ell}{\eta}\sqrt{\frac{\ln(6/\delta)}{2n_{\mathrm{P}}}},$$

where $\theta = \{\theta_1,\ldots,\theta_{n_{\mathrm{P}}}\}$ and each θ_i is a Rademacher variable.

Also notice that for all $\boldsymbol{x}$, $\phi_{\boldsymbol{x}}$ is a (L_ℓ/η)-Lipschitz function on the interval $[-C_g, C_g]$. By using a modified version of Talagrad's concentration lemma (specifically, Lemma 26.9 in (Shalev-Shwartz & Ben-David, 2014)), we can show that, when the set $\mathcal{X}_{\mathrm{P}}$ is fixed, we have

$$\mathbb{E}_\theta\left[\sup_{g\in\mathcal{G}}\frac{1}{n_{\mathrm{P}}}\sum_{i=1}^{n_{\mathrm{P}}}\theta_i\phi_{\boldsymbol{x}_i}(g(\boldsymbol{x}_i))\right] \leq \frac{L_\ell}{\eta}\mathbb{E}_\theta\left[\sup_{g\in\mathcal{G}}\frac{1}{n_{\mathrm{P}}}\sum_{i=1}^{n_{\mathrm{P}}}\theta_i g(\boldsymbol{x}_i)\right].$$

In particular, as the inequality deals with empirical Rademacher complexity, the dependence of $\phi_{\boldsymbol{x}}$ on $\boldsymbol{x}$ would not be an issue. In fact, with $\boldsymbol{x}$ being fixed, the indicator function $\mathbb{1}_{\hat{\sigma}(\boldsymbol{x})>\eta}$ is nothing but a constant and its discontinuity has nothing to do with the Lipschitz continuity of $\phi_{\boldsymbol{x}}$. We obtain Equation (8) After taking expectation over $\mathcal{X}_{\mathrm{P}} \sim p_{\mathrm{P}}^{n_{\mathrm{P}}}$. □

However, what we really want to minimize is the true risk $R(g)$. Therefore, we also need to bound the difference between $R_{\mathrm{PUbN},\eta,\hat{\sigma}}(g)$ and $R(g)$, or equivalently, the difference between $\bar{R}^{-}_{s=-1,\eta,\hat{\sigma}}(g)$ and $\bar{R}^{-}_{s=-1}(g)$.

Lemma 2. *Let $\hat{\sigma} : \mathbb{R}^d \to [0,1]$, $\eta \in (0,1]$, $\zeta = p(\hat{\sigma} \leq \eta)$ and $\epsilon = \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[|\hat{\sigma}(\boldsymbol{x}) - \sigma(\boldsymbol{x})|^2]$. For all $g \in \mathcal{G}$, it holds that*

$$|\bar{R}^{-}_{s=-1,\eta,\hat{\sigma}}(g) - \bar{R}^{-}_{s=-1}(g)| \leq C_\ell \sqrt{\zeta\epsilon} + \frac{C_\ell}{\eta}\sqrt{(1-\zeta)\epsilon}.$$

Proof. One one hand, we have

$$\begin{aligned}\bar{R}^{-}_{s=-1}(g) = &\underbrace{\int \mathbb{1}_{\hat{\sigma}(\boldsymbol{x})\leq\eta}\, \ell(-g(\boldsymbol{x}))(1-\sigma(\boldsymbol{x}))p(\boldsymbol{x})\,dx}_{A_1} \\ &+ \underbrace{\int \mathbb{1}_{\hat{\sigma}(\boldsymbol{x})>\eta}\, \ell(-g(\boldsymbol{x}))(1-\sigma(\boldsymbol{x}))p(\boldsymbol{x})\,dx}_{B_1}.\end{aligned}$$

On the other hand, we can express $\bar{R}^{-}_{s=-1,\eta,\hat{\sigma}}(g)$ as

$$\begin{aligned}\bar{R}^{-}_{s=-1,\eta,\hat{\sigma}}(g) &= \int \mathbb{1}_{\hat{\sigma}(\boldsymbol{x})\leq\eta}\, \ell(-g(\boldsymbol{x}))(1-\hat{\sigma}(\boldsymbol{x}))p(\boldsymbol{x})\,dx \\ &\quad + \int \mathbb{1}_{\hat{\sigma}(\boldsymbol{x})>\eta}\, \ell(-g(\boldsymbol{x}))\frac{1-\hat{\sigma}(\boldsymbol{x})}{\hat{\sigma}(x)}p(\boldsymbol{x}, s=+1)\,dx. \\ &= \underbrace{\int \mathbb{1}_{\hat{\sigma}(\boldsymbol{x})\leq\eta}\, \ell(-g(\boldsymbol{x}))(1-\hat{\sigma}(\boldsymbol{x}))p(\boldsymbol{x})\,dx}_{A_2} \\ &\quad + \underbrace{\int \mathbb{1}_{\hat{\sigma}(\boldsymbol{x})>\eta}\, \ell(-g(\boldsymbol{x}))(1-\hat{\sigma}(\boldsymbol{x}))\frac{\sigma(\boldsymbol{x})}{\hat{\sigma}(\boldsymbol{x})}p(\boldsymbol{x})\,dx}_{B_2}.\end{aligned}$$

The last equality follows from $p(\boldsymbol{x}, s=+1) = \sigma(\boldsymbol{x})p(\boldsymbol{x})$. As $|\bar{R}^{-}_{s=-1,\eta,\hat{\sigma}}(g) - \bar{R}^{-}_{s=-1}(g)| \leq |A_1 - A_2| + |B_1 - B_2|$, it is sufficient to derive bounds for $|A_1 - A_2|$ and $|B_1 - B_2|$ separately. For $|B_1 - B_2|$, we write

$$\begin{aligned}|B_1 - B_2| &\leq \int \mathbb{1}_{\hat{\sigma}(\boldsymbol{x})>\eta}\, \ell(-g(\boldsymbol{x}))\frac{|\hat{\sigma}(\boldsymbol{x}) - \sigma(\boldsymbol{x})|}{\hat{\sigma}(\boldsymbol{x})}p(\boldsymbol{x})\,dx \\ &\leq \frac{C_\ell}{\eta}\int \mathbb{1}_{\hat{\sigma}(\boldsymbol{x})>\eta}|\hat{\sigma}(\boldsymbol{x}) - \sigma(\boldsymbol{x})|p(\boldsymbol{x})\,dx \\ &\leq \frac{C_\ell}{\eta}\left(\int \mathbb{1}^2_{\hat{\sigma}(\boldsymbol{x})>\eta}p(\boldsymbol{x})\,dx\right)^{\frac{1}{2}}\left(\int |\hat{\sigma}(\boldsymbol{x}) - \sigma(\boldsymbol{x})|^2 p(\boldsymbol{x})\,dx\right)^{\frac{1}{2}} \\ &= \frac{C_\ell}{\eta}\sqrt{(1-\zeta)\epsilon}\end{aligned}$$

From the second to the third line we use the Cauchy-Schwarz inequality. $|A_1 - A_2| \leq C_\ell\sqrt{\zeta\epsilon}$ can be proven similarly, which concludes the proof. □

Combining lemma 1 and lemma 2, we know that with probability at least $1-\delta$, the following holds:

$$
\begin{aligned}
&\sup_{g\in\mathcal{G}}|\hat{R}^{-}_{\mathrm{PUbN},\eta,\hat{\sigma}}(g)-R(g)| \\
&\quad\le 2L_\ell\mathfrak{R}_{n_\mathrm{U},p}(\mathcal{G})+\frac{2\pi L_\ell}{\eta}\mathfrak{R}_{n_\mathrm{P},p_\mathrm{P}}(\mathcal{G})+\frac{2\rho L_\ell}{\eta}\mathfrak{R}_{n_\mathrm{bN},p_\mathrm{bN}}(\mathcal{G}) \\
&\qquad+C_\ell\sqrt{\frac{\ln(6/\delta)}{2n_\mathrm{U}}}+\frac{\pi C_\ell}{\eta}\sqrt{\frac{\ln(6/\delta)}{2n_\mathrm{P}}}+\frac{\rho C_\ell}{\eta}\sqrt{\frac{\ln(6/\delta)}{2n_\mathrm{bN}}}+C_\ell\sqrt{\zeta\epsilon}+\frac{C_\ell}{\eta}\sqrt{(1-\zeta)\epsilon}.
\end{aligned}
$$

Finally, with probability at least $1-\delta$,

$$
\begin{aligned}
&R(\hat{g}_{\mathrm{PUbN},\eta,\hat{\sigma}})-R(g^*) \\
&\quad=(R(\hat{g}_{\mathrm{PUbN},\eta,\hat{\sigma}})-\hat{R}^{-}_{\mathrm{PUbN},\eta,\hat{\sigma}}(\hat{g}_{\mathrm{PUbN},\eta,\hat{\sigma}})) \\
&\qquad+(\hat{R}^{-}_{\mathrm{PUbN},\eta,\hat{\sigma}}(\hat{g}_{\mathrm{PUbN},\eta,\hat{\sigma}})-\hat{R}^{-}_{\mathrm{PUbN},\eta,\hat{\sigma}}(g^*))+(\hat{R}^{-}_{\mathrm{PUbN},\eta,\hat{\sigma}}(g^*)-R(g^*)) \\
&\quad\le\sup_{g\in\mathcal{G}}|\hat{R}^{-}_{\mathrm{PUbN},\eta,\hat{\sigma}}(g)-R(g)|+0+\sup_{g\in\mathcal{G}}|\hat{R}^{-}_{\mathrm{PUbN},\eta,\hat{\sigma}}(g)-R(g)| \\
&\quad\le 4L_\ell\mathfrak{R}_{n_\mathrm{U},p}(\mathcal{G})+\frac{4\pi L_\ell}{\eta}\mathfrak{R}_{n_\mathrm{P},p_\mathrm{P}}(\mathcal{G})+\frac{4\rho L_\ell}{\eta}\mathfrak{R}_{n_\mathrm{bN},p_\mathrm{bN}}(\mathcal{G}) \\
&\qquad+2C_\ell\sqrt{\frac{\ln(6/\delta)}{2n_\mathrm{U}}}+\frac{2\pi C_\ell}{\eta}\sqrt{\frac{\ln(6/\delta)}{2n_\mathrm{P}}}+\frac{2\rho C_\ell}{\eta}\sqrt{\frac{\ln(6/\delta)}{2n_\mathrm{bN}}}+2C_\ell\sqrt{\zeta\epsilon}+\frac{2C_\ell}{\eta}\sqrt{(1-\zeta)\epsilon}.
\end{aligned}
$$

The first inequality uses the definition of $\hat{g}_{\mathrm{PUbN},\eta,\hat{\sigma}}$.

B. Validation Loss for Estimation of σ

In terms of validation we want to choose the model for $\hat{\sigma}$ such that $J_0(\hat{\sigma})=\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[|\hat{\sigma}(\boldsymbol{x})-\sigma(\boldsymbol{x})|^2]$ is minimized. Since $\sigma(\boldsymbol{x})p(\boldsymbol{x})=p(\boldsymbol{x},s=+1)$, we have

$$
\begin{aligned}
J_0(\hat{\sigma})&=\int(\hat{\sigma}(\boldsymbol{x})-\sigma(\boldsymbol{x}))^2p(\boldsymbol{x})\,dx \\
&=\int\hat{\sigma}(\boldsymbol{x})^2p(\boldsymbol{x})\,dx-2\int\hat{\sigma}(\boldsymbol{x})p(\boldsymbol{x},s=+1)\,dx+\int\sigma(\boldsymbol{x})^2p(\boldsymbol{x})\,dx.
\end{aligned}
$$

The last term does not depend on $\hat{\sigma}$ and can be ignored if we want to identify $\hat{\sigma}$ achieving the smallest $J(\hat{\sigma})$. We denote by $J(\hat{\sigma})$ the sum of the first two terms. The middle term can be further expanded using

$$
\int\hat{\sigma}(\boldsymbol{x})p(\boldsymbol{x},s=+1)\,dx=\pi\int\hat{\sigma}(\boldsymbol{x})p(\boldsymbol{x}\mid y=+1)\,dx+\rho\int\hat{\sigma}(\boldsymbol{x})p(\boldsymbol{x}\mid y=-1,s=+1)\,dx.
$$

The validation loss of an estimation $\hat{\sigma}$ is then defined as

$$
\hat{J}(\hat{\sigma})=\frac{1}{n_\mathrm{U}}\sum_{i=1}^{n_\mathrm{U}}\hat{\sigma}(\boldsymbol{x}_i^\mathrm{U})^2-\frac{2\pi}{n_\mathrm{P}}\sum_{i=1}^{n_\mathrm{P}}\hat{\sigma}(\boldsymbol{x}_i^\mathrm{P})-\frac{2\rho}{n_\mathrm{bN}}\sum_{i=1}^{n_\mathrm{bN}}\hat{\sigma}(\boldsymbol{x}_i^\mathrm{bN}).
$$

It is also possible to minimize this value directly to acquire $\hat{\sigma}$. In our experiments we decide to learn $\hat{\sigma}$ by nnPU for a better comparison between different methods.

C. Detailed Experimental Setting

C.1. From Multiclass to Binary Class

In the experiments we work on multiclass classification datasets. Therefore it is necessary to define the P and N classes ourselves. MNIST is processed in such a way that pair numbers 0, 2, 4, 6, 8 form the P class and impair numbers 1, 3, 5,

7, 9 form the N class. Accordingly, $\pi = 0.49$. For CIFAR-10, we consider two definitions of the P class. The first one corresponds to a quite natural task that aims to distinguish vehicles from animals. Airplane, automobile, ship and truck are therefore defined to be the P class while the N class is formed by bird, cat, deer, dog, frog and horse. For the sake of diversity, we also study another task in which we attempt to distinguish the mammals from the non-mammals. The P class is then formed by cat, deer, dog, and horse while the N class consists of the other six categories. We have $\pi = 0.4$ in the two cases. As for 20 Newsgroups, alt., comp., misc. and rec. make up the P class whereas sci., soc. and talk. make up the N class. This gives $\pi = 0.56$.

C.2. Training, Validation and Test Set

For the three datasets, we use the standard test examples as a held-out test set. The test set size is thus of 10000 for MNIST and CIFAR-10, and 7528 for 20 Newsgroups. Regarding the training set, we sample 500, 500 and 6000 P, bN and U training examples for MNIST and 20 Newsgroups, and 1000, 1000 and 10000 P, bN and U training examples for CIFAR-10. The validation set is always five times smaller than the training set.

C.3. 20 Newsgroups Preprocessing

The original 20 Newsgroups dataset contains raw text data and needs to be preprocessed into text feature vectors for classification. In our experiments we borrow the pre-trained ELMo word embedding (Peters et al., 2018) from https://allennlp.org/elmo. The used 5.5B model was, according to the website, trained on a dataset of 5.5B tokens consisting of Wikipedia (1.9B) and all of the monolingual news crawl data from WMT 2008-2012 (3.6B). For each word, we concatenate the features from the three layers of the ELMo model, and for each document, as suggested by Rücklé et al. (2018), we concatenate the average, minimum, and maximum computed along the word dimension. This results in a 9216-dimensional feature vector for a single document.

C.4. Models and Hyperparameters

Shared The nnPU threshold parameter β and the weight decay are respectively fixed at 0 and 10^{-4}. Other hyperparameters including $\tau \in \{0.5, 0.7, 0.9\}$, $\gamma \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and learning rate are selected with validation data.

MNIST For MNIST, we use a standard ConvNet with ReLU. This model contains two 5x5 convolutional layers and one fully-connected layer, with each convolutional layer followed by a 2x2 max pooling. The channel sizes are 5-10-40. The model is trained for 100 epochs with each minibatch made up of 10 P, 10 bN (if available) and 120 U samples. The learning rate is selected from the range $\alpha \in \{10^{-2}, 10^{-3}\}$.

CIFAR-10 For CIFAR-10, we train PreAct ResNet-18 (He et al., 2016) for 200 epochs and the learning rate is divided by 10 after 80 epochs and 120 epochs. This is a common practice and similar adjustment can be found in (He et al., 2016). The minibatch size is 1/100 of the number of training samples, and the initial learning rate is chosen from $\{10^{-2}, 10^{-3}\}$.

20 Newsgroups For 20 Newsgroups, with the extracted features, we simply train a multilayer perceptron with two hidden layers of 300 neurons for 50 epochs. We use basically the same hyperparameters as for MNIST except that the learning rate α is selected from $\{5 \cdot 10^{-3}, 10^{-3}, 5 \cdot 10^{-4}\}$.

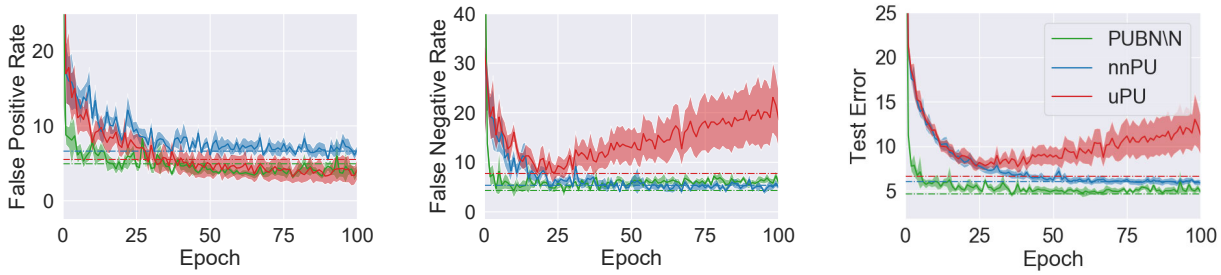
D. Additional Experiments

D.1. Why Does PUbN\N Outperform nnPU ?

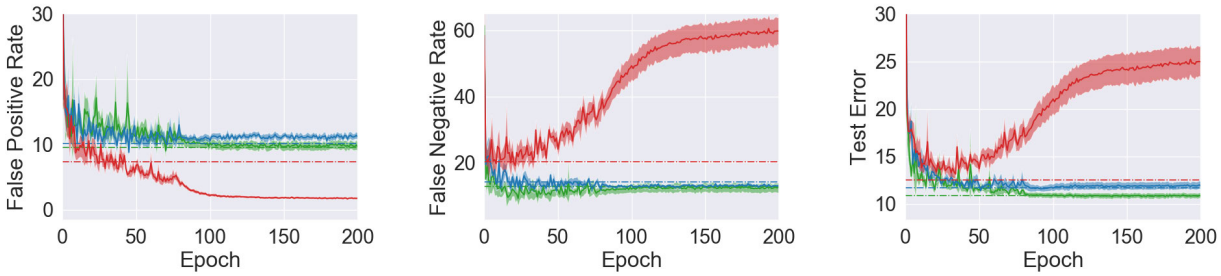
Here we complete the results presented in Section 4.4 with the plots on the other two PU learning tasks (Figure 3). We recall that we compare between PUbN\N, nnPU and uPU learning, and that both uPU and nnPU are learned with the sigmoid loss, learning rate 10^{-3} for MNIST and initial learning rate 10^{-4} for CIFAR-10. The learning rate is 10^{-4} for 20 Newsgroups.

D.2. Influence of η and ρ

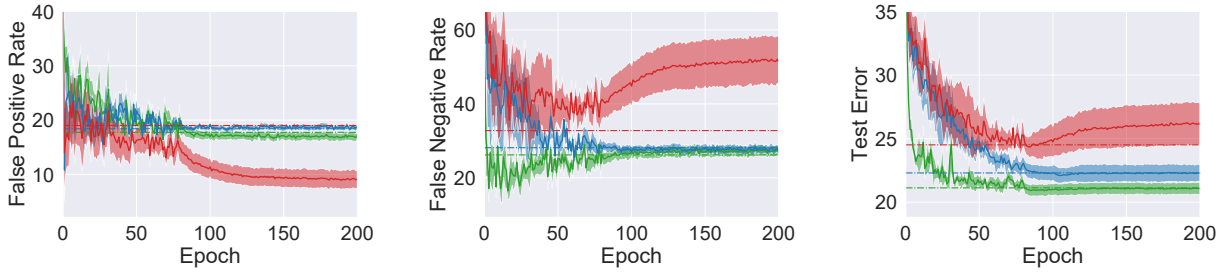
In the proposed algorithm we introduce η to control how $\bar{R}_{s=-1}(g)$ is approximated from data and assume that $\rho = p(y = -1, s = +1)$ is given. Here we conduct experiments to see how our method is affected by these two factors. To assess the influence of η, from Table 1 we pick four learning tasks and we choose τ from $\{0.5, 0.7, 0.9, 2\}$ while all the other hyperparameters are fixed. Similarly, to simulate the case where ρ is misspecified, we replace it by $\rho' \in \{0.8\rho, \rho, 1.2\rho\}$ in



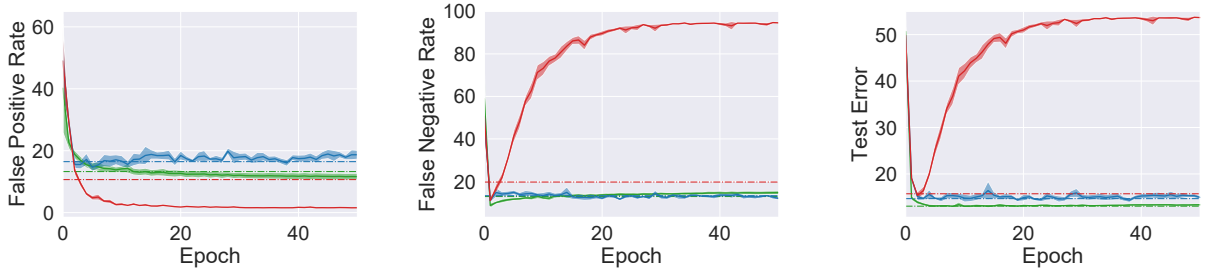
(a) MNIST



(b) CIFAR-10, vehicles as P class



(c) CIFAR-10, mammals as P class



(d) 20 Newsgroups

Figure 3. Comparison of uPU, nnPU and PUbN\N over the four PU learning tasks. For each task, means and standard deviations are computed based on the same 10 random samplings. Dashed lines indicate the corresponding values of the final classifiers (recall that at the end we select the model with the lowest validation loss out of all epochs).

our learning method and run experiments with all hyperparameters being fixed to a certain value. However, we still use the true ρ to compute η from τ to ensure that we always use the same number of U samples in the second step of the algorithm independent of the choice of ρ'.

The results are reported in Table 2 and Table 3. We can see that the performance of the algorithm is sensitive to the choice of τ. With larger value of τ, more U data are treated as N data in PUbN learning, and consequently it often leads to higher

Table 2. Results on four different PUbN learning tasks when we vary the value of τ (and accordingly, η). Reported are means of false positive rates (FPR), false negative rates (FNR), misclassification rates (Error), and validation losses (VLoss) over 10 trials.

Dataset	P	biased N	τ	FPR	FNR	Error	VLoss
MNIST	0, 2, 4, 6, 8	1, 3, 5	0.5	4.79	4.32	4.56	10.11
			0.7	3.32	4.81	4.05	**9.15**
			0.9	3.29	4.40	**3.83**	9.30
			2	3.38	5.32	4.33	10.68
CIFAR-10	Airplane, automobile, ship, truck	Horse $>$ deer = frog $>$ others	0.5	8.31	12.35	**9.92**	**12.50**
			0.7	8.23	13.15	10.20	12.62
			0.9	7.54	14.68	10.40	13.08
			2	6.23	20.29	11.85	13.64
CIFAR-10	Cat, deer, dog, horse	Bird, frog	0.5	14.45	27.57	19.70	22.08
			0.7	13.20	27.27	**18.83**	**20.72**
			0.9	13.00	32.61	20.84	23.78
			2	11.67	31.49	19.60	22.52
20 Newsgroups	alt., comp., misc., rec.	soc. $>$ talk. $>$ sci.	0.5	11.28	12.90	**12.18**	**16.04**
			0.7	11.40	13.58	12.62	16.64
			0.9	10.09	16.70	13.79	16.90
			2	10.34	20.55	16.06	20.99

Table 3. Mean and standard deviation of misclassification rates over 10 trials on different PUbN learning tasks when we replace ρ by $\rho' \in \{0.8\rho, \rho, 1.2\rho\}$. Underlines indicate significant degradation of performance according to the 5% t-test.

Dataset	P	biased N	ρ'/ρ		
			0.8	1	1.2
MNIST	0, 2, 4, 6, 8	1, 3, 5	4.10 ± 0.39	4.05 ± 0.27	4.14 ± 0.45
		$9 > 5 >$ others	3.85 ± 0.55	3.91 ± 0.66	3.94 ± 0.54
CIFAR-10	Airplane, automobile, ship, truck	Cat, dog, horse	10.23 ± 0.59	9.71 ± 0.51	$\underline{10.32 \pm 0.57}$
		Horse $>$ deer = frog $>$ others	10.18 ± 0.40	9.92 ± 0.42	10.05 ± 0.59
CIFAR-10	Cat, deer, dog, horse	Bird, frog	18.94 ± 0.50	18.83 ± 0.71	19.06 ± 0.80
		Car, truck	20.39 ± 1.24	20.19 ± 1.06	19.92 ± 0.89
20 Newsgroups	alt., comp., misc., rec.	sci.	13.49 ± 0.61	13.10 ± 0.90	13.31 ± 1.05
		talk.	12.64 ± 0.69	12.61 ± 0.75	$\underline{13.77 \pm 0.85}$
		soc. $>$ talk. $>$ sci.	$\underline{12.90 \pm 0.79}$	12.18 ± 0.59	$\underline{12.74 \pm 0.35}$

false negative rate and lower false positive rate. The trade-off between these two measures is a classic problem in binary classification. In particular, when $\tau = 2$, a lot more U samples are involved in the computation of the PUbN risk (7), but this does not allow the classifier to achieve a better performance. We also observe that there is a positive correlation between the misclassification rate and the validation loss, which confirms that the optimal value of η can be chosen without need of unbiased N data.

Table 3 shows that in general slight misspecification of ρ does not cause obvious degradation of the classification performance. In fact, misspecification of ρ mainly affect the weights of each sample when we compute $\hat{R}_{\mathrm{PUbN},\eta,\hat{\sigma}}$ (due to the direct

Table 4. Mean and standard deviation of misclassification rates over 10 trials on different PUbN learning tasks with $\hat{\sigma}$ and g trained using either the same or different sets of data.

Dataset	P	biased N	Data for $\hat{\sigma}$ and g	
			Same	Different
MNIST	0, 2, 4, 6, 8	1, 3, 5	4.05 ± 0.27	3.71 ± 0.45
		$9 > 5 >$ others	3.91 ± 0.66	4.06 ± 0.36
CIFAR-10	Airplane, automobile, ship, truck	Cat, dog, horse	9.71 ± 0.51	10.00 ± 0.51
		Horse $>$ deer $=$ frog $>$ others	9.92 ± 0.42	9.66 ± 0.46
CIFAR-10	Cat, deer, dog, horse	Bird, frog	18.83 ± 0.71	18.52 ± 0.70
		Car, truck	20.19 ± 1.06	19.98 ± 0.93
20 Newsgroups	alt., comp., misc., rec.	sci.	15.61 ± 1.50	16.60 ± 2.38
		talk.	17.14 ± 1.87	15.80 ± 0.95
		soc. $>$ talk. $>$ sci.	15.93 ± 1.88	15.80 ± 1.91

presence of ρ in (7) and influence on estimating σ). However, as long as the variation of these weights remain in a reasonable range, the learning algorithm should yield classifiers with similar performances.

D.3. Estimating σ from Separate Data

Theorem 2 suggests that $\hat{\sigma}$ should be independent from the data used to compute $\hat{R}_{\mathrm{PUbN},\eta,\hat{\sigma}}$. Therefore, here we investigate the performance of our algorithm when $\hat{\sigma}$ and g are optimized using different sets of data. We sample two training sets and two validation sets in such a way that they are all disjoint. The size of a single training set and a single validation set is as indicated in Appendix C.2, except for 20 Newsgroups we reduce the number of examples in a single set by half. We then use different pairs of training and validation sets to learn $\hat{\sigma}$ and g. For 20 Newsgroups we also conduct standard experiments where $\hat{\sigma}$ and g are learned on the same data, whereas for MNIST and CIFAR-10 we resort to Table 1.

The results are presented in Table 4. Estimating σ from separate data does not seem to benefit much the final classification performance, despite the fact that it requires collecting twice more samples. In fact, $\hat{\bar{R}}^{-}_{s=-1,\eta,\hat{\sigma}}(g)$ is a good approximation of $\bar{R}^{-}_{s=-1,\eta,\hat{\sigma}}(g)$ as long as the function $\hat{\sigma}$ is smooth enough and does not possess abrupt changes between data points. With the use of non-negative correction, validation data and L2 regularization, the resulting $\hat{\sigma}$ does not overfit training data so this should always be the case. As a consequence, even if $\hat{\sigma}$ and g are learned on the same data, we are still able to achieve small generalization error with sufficient number of samples.

D.4. Alternative Definition of nnPNU

In subsection 2.3, we define the nnPNU algorithm by forcing the estimator of the whole N partial risk to be positive. However, notice that the term $\gamma(1-\pi)\hat{R}^{-}_{\mathrm{N}}(g)$ is always positive and the chances are that including it simply makes non-negative correction weaker and is thus harmful to the final classification performance. Therefore, here we consider an alternative definition of nnPNU where we only force the term $(1-\gamma)(\hat{R}^{-}_{\mathrm{U}}(g) - \pi\hat{R}^{-}_{\mathrm{P}}(g))$ to be positive. We plug the resulting algorithm in the experiments of subsection 4.2 and summarize the results in Table 5 in which we denote the alternative version of nnPNU by nnPU+PN since it uses the same non-negative correction as nnPU. The table indicates that neither of the two definitions of nnPNU consistently outperforms the other. It also ensures that there is always a clear superiority of our proposed PUbN algorithm compared to nnPNU despite its possible variant that is considered here.

D.5. More on Text Classification

Fei & Liu (2015) introduced CBS learning in the context of text classification. The idea is to transform document representation from the traditional n-gram feature space to a center-based similarity (CBS) space, in hope that this

Table 5. Mean and standard deviation of misclassification rates over 10 trials on different PUbN learning tasks for the two possible definitions of the nnPNU algorithm.

Dataset	P	biased N	nnPNU	nnPU + PN
MNIST	0, 2, 4, 6, 8	1, 3, 5	5.33 ± 0.97	5.68 ± 0.78
		$9 > 5 >$ others	4.60 ± 0.65	5.10 ± 1.54
CIFAR-10	Airplane, automobile, ship, truck	Cat, dog, horse	10.25 ± 0.38	10.87 ± 0.62
		Horse $>$ deer $=$ frog $>$ others	9.98 ± 0.53	10.77 ± 0.65
CIFAR-10	Cat, deer, dog, horse	Bird, frog	22.00 ± 0.53	21.41 ± 1.01
		Car, truck	22.00 ± 0.74	21.80 ± 0.74
20 Newsgroups	alt., comp., misc., rec.	sci.	14.69 ± 0.46	14.50 ± 1.32
		talk.	14.38 ± 0.74	14.71 ± 1.01
		soc. $>$ talk. $>$ sci.	14.41 ± 0.70	13.66 ± 0.72

could mitigate the adverse effect of N data being biased. They conducted experiments with SVMs and showed that the transformation could effectively help improving the classification performance. However, this process largely reduces the number of input features, and for us it is unclear whether CBS transformation would still be beneficial when it is possible to use other kinds of text features or more sophisticated models. On the contrary, we propose a training strategy that is a priori compatible with any extracted features and models. As mentioned in the introduction, another important difference between our work and CBS learning is that the latter does not assume availability of U data while the presence of U data is indispensable in our case.

Experimental Setup. Below we compare PUbN learning with CBS learning on 20 Newsgroups text classification experiments. The numbers of different types of examples that are used by each learning method are summarized in Table 6, where PbN denotes the case where only P and bN data are available. Notice that here we consider two different PbN learning settings, depending on the number of used bN samples. If we compare the two PbN settings with the PUbN setting, for one we add extra U samples and for the other we replace a part of bN samples by U samples. A second point to notice is that no validation data are used in the PbN settings. In fact, although we empirically observe that the performance of CBS learning is greatly influenced by the choice of the hyperparameters, from (Fei & Liu, 2015) it is unclear how validation data can be used for hyperparameter tuning. As a result, for CBS learning we simply report the best results that were achieved in our experiments. The reported values can be regarded as an upper bound on the performance of this method. By the way, SVM training itself usually does not require the use of validation data.

To make PUbN and CBS learning comparable, we use linear model and take normalized tf-idf vectors as input variables in PUbN learning. We also include another PbN baseline that directly trains a SVM on the normalized tf-idf feature vectors. Regarding CBS learning, we use Chi-Square feature selection as it empirically produces the best results. The number of retained features is considered as a hyperparameter and as mentioned above its value can in effect have great impact on the final result. Although Fei & Liu (2015) suggested using simultaneously unigram, bigram and trigram representations of each document, the use of bigrams and trigrams does not appear to provide any benefits in our experiments. Therefore for the results that are presented here we only use the unigram representation of a document.

Results. Table 7 affirms the superiority of PUbN learning, even in the case where the PbN and PUbN settings share the same total number of samples. Only in the third learning task with 1000 bN samples and without CBS transformation PbN learning slightly outperforms PUbN learning. This can be explained by the fact that in this learning task, though the collected N samples are biased, they still cover all the possible topics appearing in the N distribution.

CBS transformation does sometimes improve the classification accuracy, but the improvement is not consistent. We conjecture that CBS learning is not so effective here both because our P class contains several distinct topics, and because our N class is not so diverse compared with (Fei & Liu, 2015). Having multiple topics in P implies that there may not be a meaningful center for the P class in the feature space. On the other hand, in the results reported by Fei & Liu (2015) we can

Table 6. Number of samples in each set under different learning settings for supplementary 20 Newsgroups experiments that compare PUbN learning with PbN learning and (Fei & Liu, 2015).

	P train	bN train	U train	P validation	bN validation	U validation	Total
PUbN	500	320	500	100	80	100	1600
PbN 400	600	400	NA	NA	NA	NA	1000
PbN 1000	600	1000	NA	NA	NA	NA	1600

Table 7. Mean and standard deviation of misclassification rates over 10 trials for text classification tasks on 20 newsgroups to compare PUbN learning with CBS learning and a PbN baseline.

biased N	PUbN	PbN 400		PbN 1000	
		tf-idf	CBS	tf-idf	CBS
sci.	$\mathbf{13.06 \pm 0.69}$	25.31 ± 0.72	21.15 ± 0.73	19.68 ± 0.94	17.72 ± 0.74
talk.	$\mathbf{14.88 \pm 0.63}$	23.42 ± 0.44	22.73 ± 0.81	19.41 ± 0.50	19.59 ± 0.47
soc. $>$ talk. $>$ sci.	$\mathbf{13.96 \pm 0.61}$	19.82 ± 0.86	21.68 ± 0.84	$\mathbf{13.68 \pm 0.43}$	17.43 ± 0.53

see that the benefit of CBS learning becomes less significant when a large proportion of N topics can be found in the bN set. In particular, in the last learning task, CBS transformation even turns out to be harmful. We also observe that after CBS transformation, the classifier becomes less sensitive to both how bN data are sampled and the size of the bN set. This seems to suggest that CBS learning is the most beneficial when both the number of topics appeared in the bN set and the amount of bN data are very limited.