Generative Adversarial Positive-Unlabeled Learning

Ming Hou[1], **Brahim Chaib-draa**[2], **Chao Li**[3], **Qibin Zhao**[14] *
[1] Tensor Learning Unit, Center for Advanced Intelligence Project, RIKEN, Japan
[2] Department of Computer Science and Software Engineering, Laval University, Canada
[3] Causal Inference Team, Center for Advanced Intelligence Project, RIKEN, Japan
[4] School of Automation, Guangdong University of Technology, China
ming.hou@riken.jp, brahim.chaib-draa@ift.ulaval.ca, chao.li.hf@riken.jp, qibin.zhao@riken.jp

Abstract

In this work, we consider the task of classifying binary positive-unlabeled (PU) data. Existing discriminative learning based PU models attempt to seek an optimal reweighting strategy for unlabeled (U) data, so that a decent decision boundary can be found. However, given limited positive (P) data, the conventional PU models tend to suffer from overfitting when adapted to very flexible deep neural networks. In contrast, we are the first to innovate a totally new paradigm to attack the binary PU task, from the perspective of generative learning by leveraging the powerful generative adversarial networks (GAN). Our generative positive-unlabeled (GenPU) framework incorporates an array of discriminators and generators that are endowed with different roles in simultaneously producing positive and negative realistic samples. We provide theoretical analysis to justify that, at equilibrium, GenPU is capable of recovering both positive and negative data distributions. Moreover, we show GenPU is generalizable and closely related to the semi-supervised classification. Given rather limited P data, experiments on both synthetic and real-world dataset demonstrate the effectiveness of our proposed framework. With infinite realistic and diverse samples generated from GenPU, a very flexible classifier can then be trained using deep neural networks.

1 Introduction

Positive-unlabeled (PU) classification [Denis *et al.*, 2005] has gained great popularity in dealing with limited partially labeled data and succeeded in a broad range of applications such as automatic label identification. Yet, PU can be used for the detection of outliers in an unlabeled dataset with knowledge only from a collection of inlier data [Hido *et al.*, 2008]. PU also finds its usefulness in 'one-vs-rest' classification task such as land-cover classification (urban vs non-urban) where non-urban data are too diverse to be labeled than urban data [Li *et al.*, 2011].

The most commonly used PU approaches for binary classification can typically be categorized, in terms of the way of handling U data, into two types [Kiryo *et al.*, 2017]. One type such as [Liu *et al.*, 2002; Li and Liu, 2003] attempts to recognize negative samples in the U data and then feed them to classical positive-negative (PN) models. However, these approaches depend heavily on the heuristic strategies and often yield a poor solution. The other type, including [Liu *et al.*, 2003; Lee and Liu, 2003], offers a better solution by treating U data to be N data with a decayed weight. Nevertheless, finding an optimal weight turns out to be quite costly. Most importantly, the classifiers trained based on above approaches suffer from a systematic estimation bias [Du Plessis *et al.*, 2015; Kiryo *et al.*, 2017].

Seeking for unbiased PU classifier, [Du Plessis *et al.*, 2014] investigated the strategy of viewing U data as a weighted mixture of P and N data [Elkan and Noto, 2008], and introduced an unbiased risk estimator by exploiting some non-convex symmetric losses, i.e., the ramp loss. Although cancelling the bias, the non-convex loss is undesirable for PU due to the difficulty of non-convex optimization. To this end, [Du Plessis *et al.*, 2015] proposed a more general risk estimator which is always unbiased and convex if the convex loss satisfies a linear-odd condition [Patrini *et al.*, 2016]. Theoretically, these authors argue that the estimator yields globally optimal solution, with more appealing learning properties than the non-convex counterpart. More recently, [Kiryo *et al.*, 2017] observed that the aforementioned unbiased risk estimators can go negative without bounding from the below, leading to serious overfitting when the classifier becomes too flexible. To fix this, they presented a non-negative biased risk estimator yet with favorable theoretical guarantees in terms of consistency, mean-squared-error reduction and estimation error. The proposed estimator is shown to be more robust against overfitting than previous unbiased ones. However, given limited P data, the overfitting issue still exists especially when very flexible deep neural network is applied.

Generative models, on the other hand, have the advantage in expressing complex data distribution. Apart from distribution density estimation, generative models are often applied to learn a function that is able to create more samples from the approximate distribution. Lately, a large body of successful deep generative models have emerged, especially generative adversarial networks (GAN) [Goodfellow *et al.*, 2014;

[*] the corresponding author

Salimans *et al.*, 2016]. GAN intends to solve the task of generative modeling by making two agents play a game against each other. One agent named generator synthesizes fake data from random noise; the other agent, termed as discriminator, examines both real and fake data and determines whether it is real or not. Both agents keep evolving over time and get better and better at their jobs. Eventually, the generator is forced to create synthetic data which is as realistic as possible to those from the training dataset.

Inspired by the tremendous success and expressive power of GAN, we tackle the binary PU classification task by resorting to generative modeling, and propose our generative positive-unlabeled (GenPU) learning framework. Building upon GAN, our GenPU model includes an array of generators and discriminators as agents in the game. These agents are devised to play different parts in simultaneously generating positive and negative real-like samples, and thereafter a standard PN classifier can be trained on those synthetic samples. Given a small portion of labeled P data as seeds, GenPU is able to capture the underlying P and N data distributions, with the capability to create infinite diverse P and N samples. In this way, the overfitting problem of conventional PU can be greatly mitigated. Furthermore, our GenPU is generalizable in the sense that it can be established by switching to different underlying GAN variants with distance measurements (i.e., Wasserstein GAN [Arjovsky *et al.*, 2017]) other than Jensen-Shannon divergence (JSD). As long as those variants are sophisticated to produce high-quality diverse samples, the optimal accuracy could be achieved by training a very deep neural networks.

Our main contribution (i) we are the first (to our knowledge) to invent a totally new paradigm to effectively solve the PU task through deep generative models; (ii) we provide theoretical analysis to prove that, at equilibrium, our model is capable of learning both positive and negative data distributions; (iii) we experimentally show the effectiveness of the proposed model given limited P data on both synthetic and real-world dataset; (iiii) our method can be easily extended to solve the semi-supervised classification, and also opens a door to new solutions of many other weakly supervised learning tasks from the aspect of generative learning.

2 Preliminaries

2.1 positive-unlabeled (PU) classification

Given as input d-dimensional random variable $\mathbf{x} \in \mathbb{R}^d$ and scalar random variable $y \in \{\pm 1\}$ as class label, and let $p(\mathbf{x}, y)$ be the *joint density*, the *class-conditional densities* are:

$$p_p(\mathbf{x}) = p(\mathbf{x}|y=1) \qquad p_n(\mathbf{x}) = p(\mathbf{x}|y=-1),$$

while $p(\mathbf{x})$ refers to as the unlabeled *marginal density*. The standard PU classification task [Ward *et al.*, 2009] consists of a positive dataset $\mathcal{X}_p$ and an unlabeled dataset $\mathcal{X}_u$ with i.i.d samples drawn from $p_p(\mathbf{x})$ and $p(\mathbf{x})$, respectively:

$$\mathcal{X}_p = \{\mathbf{x}_p^i\}_{i=1}^{n_p} \sim p_p(\mathbf{x}) \qquad \mathcal{X}_u = \{\mathbf{x}_u^i\}_{i=1}^{n_u} \sim p(\mathbf{x}).$$

Due to the fact that the unlabeled data can be regarded as a mixture of both positive and negative samples, the marginal density turns out to be

$$p(\mathbf{x}) = \pi_p p(\mathbf{x}|y=1) + \pi_n p(\mathbf{x}|y=-1), \tag{1}$$

where $\pi_p = p(y=1)$ and $\pi_n = 1 - \pi_p$ are denoted as *class-prior probability*, which is usually unknown in advance and can be estimated from the given data [Jain *et al.*, 2016]. The objective of PU task is to train a classifier on $\mathcal{X}_p$ and $\mathcal{X}_u$ so as to classify the new unseen pattern $\mathbf{x}^{new}$.

In contrast to PU classification, positive-negative (PN) classification assumes all negative samples,

$$\mathcal{X}_n = \{\mathbf{x}_n^i\}_{i=1}^{n_n} \sim p_n(\mathbf{x}),$$

are labeled, so that the classifier can be trained in an ordinary supervised learning fashion.

2.2 generative adversarial networks (GAN)

GAN, originated in [Goodfellow *et al.*, 2014], is one of the most recent successful generative models that is equipped with the power of producing distributional outputs. GAN obtains this capability through an adversarial competition between a generator G and a discriminator D that involves optimizing the following minimax objective function:

$$\min_G \max_D \mathcal{V}(G, D) = \min_G \max_D \; \mathbb{E}_{\mathbf{x}\sim p_x(\mathbf{x})} \log(D(\mathbf{x})) + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})} \log(1 - D(G(\mathbf{z}))), \tag{2}$$

where $p_x(\mathbf{x})$ represents true data distribution; $p_z(\mathbf{z})$ is typically a simple prior distribution (e.g., $\mathcal{N}(0,1)$) for latent code $\mathbf{z}$, while a generator distribution $p_g(\mathbf{x})$ associated with G is induced by the transformation $G(\mathbf{z})\colon \mathbf{z} \to \mathbf{x}$.

To find the optimal solution, [Goodfellow *et al.*, 2014] employed simultaneous stochastic gradient descent (SGD) for alternately updating D and G. The authors argued that, given the optimal D, minimizing G is equivalent to minimizing the distribution distance between $p_x(\mathbf{x})$ and $p_g(\mathbf{x})$. At convergence, GAN has $p_x(\mathbf{x}) = p_g(\mathbf{x})$.

3 Generative PU Classification

3.1 notations

Throughout the paper, $\{p_p(\mathbf{x}), p_n(\mathbf{x}), p(\mathbf{x})\}$ denote the positive data distribution, the negative data distribution and the entire data distribution, respectively. $\{D_p, D_u, D_n\}$ are referred to as the positive, unlabeled and negative discriminators, while $\{G_p, G_n\}$ stand for positive and negative generators, targeting to produce real-like positive and negative samples. Correspondingly, $\{p_{gp}(\mathbf{x}), p_{gn}(\mathbf{x})\}$ describe the positive and negative distributions induced by the generator functions $G_p(\mathbf{z})$ and $G_n(\mathbf{z})$.

3.2 proposed GenPU model

We build our GenPU model upon GAN by leveraging its potentiality in producing realistic data, with the goal of identification of both positive and negative distributions from P and U data. Then, a decision boundary can be made by training standard PN classifier on the generated samples. Figure 1 illustrates the architecture of the proposed framework.

In brief, GenPU framework is an analogy to a minimax game comprising of two generators $\{G_p, G_n\}$ and three discriminators $\{D_p, D_u, D_n\}$. Guided by the adversarial supervision of $\{D_p, D_u, D_n\}$, $\{G_p, G_n\}$ are tasked with synthesizing positive and negative samples that are indistinguishable with the real ones drawn from $\{p_p(\mathbf{x}), p_n(\mathbf{x})\}$, respectively. As being their competitive opponents, $\{D_p, D_u, D_n\}$

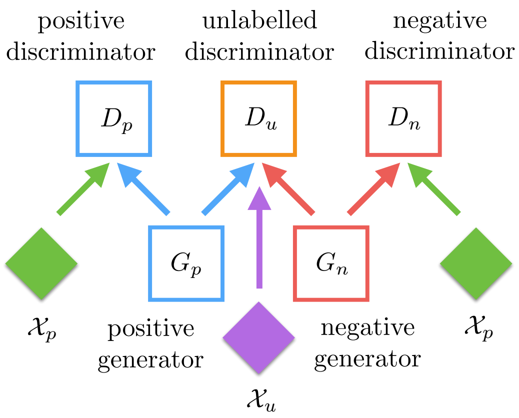

Figure 1: Our GenPU framework. D_p receives as inputs the real positive examples from $\mathcal{X}_p$ and the synthetic positive examples from G_p; D_n receives as inputs the real positive examples from $\mathcal{X}_p$ and the synthetic negative examples from G_n; D_u receives as inputs real unlabeled examples from $\mathcal{X}_u$, synthetic positive examples from G_p as well as synthetic negative examples from G_n at the same time. Associated with different loss functions, G_p and G_n are designated to generate positive and negative examples, respectively.

are devised to play distinct roles in instructing the learning process of $\{G_p, G_n\}$.

More formally, the overall GenPU objective function can be decomposed, in views of G_p and G_n, as follows:

$$\min_{\{G_p,G_n\}} \max_{\{D_p,D_u,D_n\}} \mathcal{V}(G,D) = \pi_p \min_{G_p} \max_{\{D_p,D_u\}} \mathcal{V}_{G_p}(G,D) + \pi_n \min_{G_n} \max_{\{D_u,D_n\}} \mathcal{V}_{G_n}(G,D), \quad (3)$$

where π_p and π_n corresponding to G_p and G_n are the priors for positive class and negative class, satisfying $\pi_p + \pi_n = 1$. Here, we assume π_p and π_n are predetermined and fixed.

The first term linked with G_p in (3) can be further split into two standard GAN components GAN_{G_p,D_p} and GAN_{G_p,D_u}:

$$\min_{G_p} \max_{\{D_p,D_u\}} \mathcal{V}_{G_p}(G,D) = \lambda_p \min_{G_p} \max_{D_p} \mathcal{V}_{G_p,D_p}(G,D) + \lambda_u \min_{G_p} \max_{D_u} \mathcal{V}_{G_p,D_u}(G,D), \quad (4)$$

where λ_p and λ_u are the weights balancing the relative importance of effects between D_p and D_u. In particular, the value functions of GAN_{G_p,D_p} and GAN_{G_p,D_u} are

$$\mathcal{V}_{G_p,D_p}(G,D) = \mathbb{E}_{\mathbf{x}\sim p_p(\mathbf{x})} \log(D_p(\mathbf{x})) + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})} \log(1 - D_p(G_p(\mathbf{z}))) \quad (5)$$

and

$$\mathcal{V}_{G_p,D_u}(G,D) = \mathbb{E}_{\mathbf{x}\sim p_u(\mathbf{x})} \log(D_u(\mathbf{x})) + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})} \log(1 - D_u(G_p(\mathbf{z}))). \quad (6)$$

On the other hand, the second term linked with G_n in (3) can also be split into GAN components, namely GAN_{G_n,D_u} and GAN_{G_n,D_n}:

$$\min_{G_n} \max_{\{D_u,D_n\}} \mathcal{V}_{G_p}(G,D) = \lambda_u \min_{G_n} \max_{D_u} \mathcal{V}_{G_n,D_u}(G,D) + \lambda_n \min_{G_n} \max_{D_n} \mathcal{V}_{G_n,D_n}(G,D), \quad (7)$$

whose weights λ_u and λ_n control the trade-off between D_u and D_n. GAN_{G_n,D_u} also takes the form of the standard GAN with the value function

$$\mathcal{V}_{G_n,D_u}(G,D) = \mathbb{E}_{\mathbf{x}\sim p_u(\mathbf{x})} \log(D_u(\mathbf{x})) + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})} \log(1 - D_u(G_n(\mathbf{z}))). \quad (8)$$

In contrast to the 'zero-sum' loss applied elsewhere, the optimization of GAN_{G_n,D_n} is given by first maximizing the objective

$$D_n^\star = \arg\max_{D_n} \mathbb{E}_{\mathbf{x}\sim p_p(\mathbf{x})} \log(D_n(\mathbf{x})) + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})} \log(1 - D_n(G_n(\mathbf{z}))), \quad (9)$$

then plugging $D_n^\star$ into the following equation (10), leading to a value function of G_n as

$$\mathcal{V}_{G_n,D_n^\star}(G, D_n^\star) = -\mathbb{E}_{\mathbf{x}\sim p_p(\mathbf{x})} \log(D_n^\star(\mathbf{x})) - \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})} \log(1 - D_n^\star(G_n(\mathbf{z}))). \quad (10)$$

Intuitively, (5)-(6) indicate G_p, co-supervised under both D_p and D_u, endeavours to minimize the distance between the induced distribution $p_{gp}(\mathbf{x})$ and positive data distribution $p_p(\mathbf{x})$, while striving to stay around within the whole data distribution $p(\mathbf{x})$. In fact, G_p tries to deceive both discriminators by simultaneously maximizing D_p's and D_u's outputs on fake positive samples. As a result, the loss terms in (5) and (6) jointly guide $p_{gp}(\mathbf{x})$ gradually moves towards and finally settles to $p_p(\mathbf{x})$ of $p(\mathbf{x})$.

Equations (8)-(10) suggest G_n, when facing both D_u and D_n, struggles to make the induced $p_{gn}(\mathbf{x})$ stay away from $p_p(\mathbf{x})$, and also makes its effort to force $p_{gn}(\mathbf{x})$ to lie within $p(\mathbf{x})$. To achieve this, the objective in (10) favors G_n to produce negative examples; this in turn helps D_n to maximize the objective in (9) to separate positive training samples from fake negative samples rather than confusing D_n. Notice that, in the value function (10), G_n is designed to minimize D_n's output instead of maximizing it when feeding D_n with fake negative samples. Consequently, D_n will send uniformly negative feedback to G_n. In this way, the gradient information derived from negative feedback drives down $p_{gn}(\mathbf{x})$ near the positive data region $p_p(\mathbf{x})$. In the meantime, the gradient signals from D_u increase $p_{gn}(\mathbf{x})$ outside the positive region but still restricting $p_{gn}(\mathbf{x})$ in the true data distribution $p(\mathbf{x})$. This crucial effect will eventually push $p_{gn}(\mathbf{x})$ away from $p_p(\mathbf{x})$ but towards $p_n(\mathbf{x})$.

3.3 theoretical analysis

Theoretically, suppose all the $\{G_p, G_n\}$ and $\{D_p, D_u, D_n\}$ have enough capacity. Then the following results show that, at Nash equilibrium point of (3), the minimal JSD between the distributions induced by $\{G_p, G_n\}$ and the data distributions $\{p_p(\mathbf{x}), p_n(\mathbf{x})\}$ are achieved, respectively, i.e., $p_{gp}(\mathbf{x}) = p_p(\mathbf{x})$ and $p_{gn}(\mathbf{x}) = p_n(\mathbf{x})$. Meanwhile, the JSD between the distribution induced by G_n and data distribution $p_p(\mathbf{x})$ is maximized, i.e., $p_{gn}(\mathbf{x})$ almost never overlaps with $p_p(\mathbf{x})$.

Proposition 1. *Given fixed generators G_p, G_n and known class prior π_p, the optimal discriminators D_p, D_u and D_n*

for the objective in equation (3) *have the following forms*:

$$D_p^\star(\boldsymbol{x}) = \frac{p_p(\boldsymbol{x})}{p_p(\boldsymbol{x}) + p_{gp}(\boldsymbol{x})},$$

$$D_u^\star(\boldsymbol{x}) = \frac{p(\boldsymbol{x})}{p(\boldsymbol{x}) + \pi_p p_{gp}(\boldsymbol{x}) + \pi_n p_{gn}(\boldsymbol{x})}$$

and

$$D_n^\star(\boldsymbol{x}) = \frac{p_p(\boldsymbol{x})}{p_p(\boldsymbol{x}) + p_{gn}(\boldsymbol{x})}.$$

Proof. Assume that all the discriminators D_p, D_u and D_n can be optimized in functional space. Differentiating the objective $\mathcal{V}(G, D)$ in (3) w.r.t. D_p, D_u and D_n and equating the functional derivatives to zero, we can obtain the optimal $D_p^\star$, $D_u^\star$ and $D_n^\star$ as described above. □

Theorem 2. *Suppose the data distribution $p(\boldsymbol{x})$ in the standard PU learning setting takes form of $p(\boldsymbol{x}) = \pi_p p_p(\boldsymbol{x}) + \pi_n p_n(\boldsymbol{x})$, where $p_p(\boldsymbol{x})$ and $p_n(\boldsymbol{x})$ are well-separated. Given the optimal $D_p^\star$, $D_u^\star$ and $D_n^\star$, the minimax optimization problem with the objective function in* (3) *obtains its optimal solution if*

$$p_{gp}(\boldsymbol{x}) = p_p(\boldsymbol{x}) \ \ \textit{and} \ \ p_{gn}(\boldsymbol{x}) = p_n(\boldsymbol{x}), \qquad (11)$$

with the objective value of $-(\pi_p \lambda_p + \lambda_u)\log(4)$.

Proof. Substituting the optimal $D_p^\star$, $D_u^\star$ and $D_n^\star$ into (3), the objective can be rewritten as follows:

$$\begin{aligned}
\mathcal{V}(G, D^\star) = \pi_p \cdot \{\lambda_p \cdot [\mathbb{E}_{\mathbf{x}\sim p_p(\mathbf{x})} \log(\frac{p_p(\mathbf{x})}{p_p(\mathbf{x}) + p_{gp}(\mathbf{x})})& \\
+ \mathbb{E}_{\mathbf{x}\sim p_{gp}(\mathbf{x})} \log(\frac{p_{gp}(\mathbf{x})}{p_p(\mathbf{x}) + p_{gp}(\mathbf{x})})]& \\
+ \lambda_u \cdot [\mathbb{E}_{\mathbf{x}\sim p_u(\mathbf{x})} \log(\frac{p(\mathbf{x})}{p(\mathbf{x}) + \pi_p p_{gp}(\mathbf{x}) + \pi_n p_{gn}(\mathbf{x})})& \\
+ \mathbb{E}_{\mathbf{x}\sim p_{gp}(\mathbf{x})} \log(\frac{\pi_p p_{gp}(\mathbf{x}) + \pi_n p_{gn}(\mathbf{x})}{p(\mathbf{x}) + \pi_p p_{gp}(\mathbf{x}) + \pi_n p_{gn}(\mathbf{x})})]\}& \\
+ \pi_n \cdot \{\lambda_u \cdot [\mathbb{E}_{\mathbf{x}\sim p_u(\mathbf{x})} \log(\frac{p(\mathbf{x})}{p(\mathbf{x}) + \pi_p p_{gp}(\mathbf{x}) + \pi_n p_{gn}(\mathbf{x})})& \\
+ \mathbb{E}_{\mathbf{x}\sim p_{gn}(\mathbf{x})} \log(\frac{\pi_p p_{gp}(\mathbf{x}) + \pi_n p_{gn}(\mathbf{x})}{p(\mathbf{x}) + \pi_p p_{gp}(\mathbf{x}) + \pi_n p_{gn}(\mathbf{x})})]& \\
- \lambda_n \cdot [\mathbb{E}_{\mathbf{x}\sim p_p(\mathbf{x})} \log(\frac{p_p(\mathbf{x})}{p_p(\mathbf{x}) + p_{gn}(\mathbf{x})})& \\
+ \mathbb{E}_{\mathbf{x}\sim p_{gn}(\mathbf{x})} \log(\frac{p_{gn}(\mathbf{x})}{p_p(\mathbf{x}) + p_{gn}(\mathbf{x})})]\}.& \qquad (12)
\end{aligned}$$

Combining the intermediate terms associated with λ_u using the fact $\pi_p + \pi_n = 1$, we reorganize (12) and arrive at

$$\begin{aligned}
G^\star &= \arg\min_G \mathcal{V}(G, D^\star) \\
&= \arg\min_G \ \pi_p \cdot \lambda_p \cdot [2 \cdot \mathrm{JSD}(p_p \,\|\, p_{gp}) - \log(4)] \\
&\quad + \lambda_u \cdot [2 \cdot \mathrm{JSD}(p \,\|\, \pi_p p_{gp} + \pi_n p_{gn}) - \log(4)] \\
&\quad - \pi_n \cdot \lambda_n \cdot [2 \cdot \mathrm{JSD}(p_p \,\|\, p_{gn}) - \log(4)], \qquad (13)
\end{aligned}$$

which peaks its minimum if

$$p_{gp}(\mathbf{x}) = p_p(\mathbf{x}), \qquad (14)$$

$$\pi_p p_{gp}(\mathbf{x}) + \pi_n p_{gn}(\mathbf{x}) = p(\mathbf{x}) \qquad (15)$$

and for almost every $\mathbf{x}$ except for those in a zero measure set

$$p_p(\mathbf{x}) > 0 \Rightarrow p_{gn}(\mathbf{x}) = 0, \ p_{gn}(\mathbf{x}) > 0 \Rightarrow p_p(\mathbf{x}) = 0. \quad (16)$$

The solution to $G = \{G_p, G_n\}$ must jointly satisfy the conditions described in (14)-(16), which implies (11) and leads to the minimum objective value of $-(\pi_p \lambda_p + \lambda_u)\log(4)$. □

The theorem reveals that approaching to Nash equilibrium is equivalent to jointly minimizing $\mathrm{JSD}(p \,\|\, \pi_p p_{gp} + \pi_n p_{gn})$ and $\mathrm{JSD}(p_p \,\|\, p_{gp})$ and maximizing $\mathrm{JSD}(p_p \,\|\, p_{gn})$ at the same time, thus exactly capturing p_p and p_n.

3.4 connection to semi-supervised classification

The goal of semi-supervised classification is to learn a classifier from positive, negative and unlabeled data. In such context, besides training sets $\mathcal{X}_p$ and $\mathcal{X}_u$, a partially labeled negative set $\mathcal{X}_n$ is also available, with samples drawn from negative data distribution $p_n(\mathbf{x})$.

In fact, the very same architecture of GenPU can be applied to the semi-supervised classification task by just adapting the standard GAN value function to G_n, then the total value function turns out to be

$$\begin{aligned}
&\mathcal{V}(G, D) = \\
&\pi_p\{\lambda_p[\mathbb{E}_{\mathbf{x}\sim p_p(\mathbf{x})} \log(D_p(\mathbf{x})) + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})} \log(1 - D_p(G_p(\mathbf{z})))] \\
&+ \lambda_u[\mathbb{E}_{\mathbf{x}\sim p_u(\mathbf{x})} \log(D_u(\mathbf{x})) + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})} \log(1 - D_u(G_p(\mathbf{z})))]\} \\
&+ \\
&\pi_n\{\lambda_u[\mathbb{E}_{\mathbf{x}\sim p_u(\mathbf{x})} \log(D_u(\mathbf{x})) + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})} \log(1 - D_u(G_n(\mathbf{z})))] \\
&+ \lambda_n[\mathbb{E}_{\mathbf{x}\sim p_n(\mathbf{x})} \log(D_n(\mathbf{x})) + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})} \log(1 - D_n(G_n(\mathbf{z})))]\}.
\end{aligned}$$

With above formulation, D_n discriminates the negative training samples from the synthetic negative samples produced by G_n. Now G_n intends to fool D_n and D_u simultaneously by outputting realistic examples, just like G_p does for D_p and D_u. Being attracted by both $p(\mathbf{x})$ and $p_n(\mathbf{x})$, the induced distribution $p_{gn}(\mathbf{x})$ gradually approaches to $p_n(\mathbf{x})$ and finally recovers the true distribution $p_n(\mathbf{x})$ of $p(\mathbf{x})$. Theoretically, it is not hard to show the optimal G_p and G_n, at convergence, give rise to $p_{gp}(\mathbf{x}) = p_p(\mathbf{x})$ and $p_{gn}(\mathbf{x}) = p_n(\mathbf{x})$.

4 Experimental Results

We show the efficacy of our framework by conducting experiments on synthetic and real-world images datasets. For real data, the approaches including oracle PN, unbiased PU (UPU) [Du Plessis *et al.*, 2015], non-negative PU (NNPU) [Kiryo *et al.*, 2017] [1] are selected for comparison. Specifically, the Oracle PN means all the training labels are available for all the P and N data, whose performance is just used as a reference for other approaches. For all the methods, the true classprior π_p is assumed to be known in advance. Regarding the weights of GenPU, for simplicity, we set $\lambda_u = 1$ and freely tune λ_p and λ_n on the validation set via grid search over the range like [... 0.01, 0.02, ... 0.1, 0.2, ..., 1, 2, ...].

[1] The software codes for UPU and NNPU are downloaded from https://github.com/kiryor/nnPUlearning

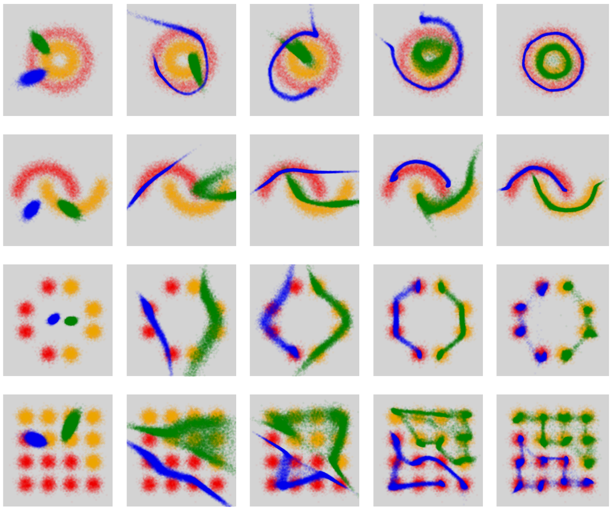

Figure 2: Evolution of the positive samples (in green) and negative samples (in blue) produced by GenPU. The true positive samples (in orange) and true negative samples (in red) are also illustrated.

4.1 synthetic simulation

We begin our test with a toy example to visualize the learning behaviors of our GenPU. The training samples are synthesized using concentric circles, Gaussian mixtures and half moons functions with Gaussian noises added to the data (standard deviation is 0.1414). The training set contains 5000 positive and 5000 negative samples, which are then partitioned into 500 positively labelled and 9500 unlabeled samples. We establish the generators with two fully connected hidden layers and the discriminators with one hidden layer. There are 128 ReLU units contained in all hidden layers. The dimensionality of the input latent code is set to 256. Figure 2 depicts the evolution of positive and negative samples produced by GenPU through time. As expected, in all the scenarios, the induced generator distributions successfully converge to the respective true data distributions given limited P data. Notice that the Gaussian mixtures cases demonstrate the capability of our GenPU to learn a distribution with multiple submodes.

Operation	Feature Maps	Nonlinearity
$G_p(\mathbf{z}), G_n(\mathbf{z}): \mathbf{z} \sim \mathcal{N}(0, I)$	100	
fully connected	256	leaky relu
fully connected	256	leaky relu
fully connected	256/784	tanh
$D_p(\mathbf{x}), D_n(\mathbf{x})$	256/784	
fully connected	1	sigmoid
$D_u(\mathbf{x})$	256/784	
fully connected	256	leaky relu
fully connected	256	leaky relu
fully connected	1	sigmoid
leaky relu slope	0.2	
mini-batch size for $\mathcal{X}_p, \mathcal{X}_u$	50, 100	
learning rate	0.0003	
optimizer	Adam($0.9, 0.999$)	
weight, bias initialization	0, 0	

Table 1: Specifications of network architecture and hyperparameters for USPS/MNIST dataset.

4.2 mnist and usps dataset

Next, the evaluation is carried out on MNIST [LeCun *et al.*, 1998] and USPS [LeCun *et al.*, 1990] datasets. For MNIST, we each time select a pair of digits to construct the P and N sets, each of which consists of $5,000$ training points. The specifics for architecture and hyperparameters are described in Table 1. To be challenging, the results of the most visually similar digit pairs, such as '3' vs '5' and '8' vs '3', are recorded in Table 1. The best accuracies are shown with the number of labeled positive examples N_l ranging from 100 to 1. Obviously, our method outperforms UPU in all the cases. We also observe our GenPU achieves better than or comparable accuracy to NNPU when the number of labeled samples is relatively large (i.e., $N_l = 100$). However, when the labeled samples are insufficient, for instance $N_l = 5$ of the '3' vs '5' scenario, the accuracy of GenPU slightly decreases from 0.983 to 0.979, which is in contrast to that of NNPU drops significantly from 0.969 to 0.843. Spectacularly, GenPU still remains highly accurate even if only one labeled sample is provided whereas NNPU fails in this situation.

Figure 3 reports the training and test errors of the classifiers for distinct settings of N_l. When N_l is 100, UPU suffers from a serious overfitting to training data, whilst both NNPU and GenPU perform fairly well. As N_l goes small (i.e., 5), NNPU also starts to overfit. It should be mentioned that the negative training curve of UPU (in blue) is because the unbiased risk estimators [Du Plessis *et al.*, 2014; 2015] contain negative loss term which is unbounded from the below. When the classifier becomes very flexible, the risk can be arbitrarily negative [Kiryo *et al.*, 2017]. Additionally, the rather limited N_l cause training processes of both UPU and NNPU behave unstable. In contrast, GenPU avoids overfitting to small training P data by restricting the models of D_p and D_n from being too complex when N_l becomes small (see Table 1). For visualization, Figure 4 demonstrates the generated digits with only one labeled '3', together with the projected distributions induced by G_p and G_n. In Table 3, similar results can be obtained on USPS data.

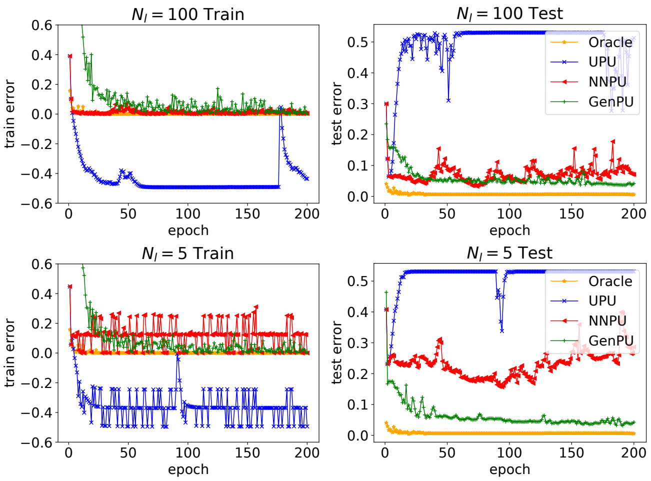

Figure 3: Training error and test error of deep PN classifiers on MINST for the pair '3' vs '5' with distinct N_l. Top: (a) and (b) for $N_l = 100$. Bottom: (c) and (d) for $N_l = 5$.

MNIST	'3' vs. '5'				'8' vs. '3'			
$N_p : N_u$	Oracle PN	UPU	NNPU	GenPU	Oracle PN	UPU	NNPU	GenPU
100 : 9900	0.993	0.914	0.969	**0.983**	0.994	0.932	0.974	**0.982**
50 : 9950	0.993	0.854	0.966	**0.982**	0.994	0.873	0.965	**0.979**
10 : 9990	0.993	0.711	0.866	**0.980**	0.994	0.733	0.907	**0.978**
5 : 9995	0.993	0.660	0.843	**0.979**	0.994	0.684	0.840	**0.976**
1 : 9999	0.993	0.557	0.563	**0.976**	0.994	0.550	0.573	**0.972**

Table 2: The accuracy comparison on MNIST for $N_l \in \{100, 50, 10, 5, 1\}$.

USPS	'3' vs '5'			'8' vs '3'		
$N_l : N_u$	UPU	NNPU	GenPU	UPU	NNPU	GenPU
50 : 1950	0.890	**0.965**	**0.965**	0.900	**0.965**	0.945
10 : 1990	0.735	0.880	**0.955**	0.725	0.920	**0.935**
5 : 1995	0.670	0.830	**0.950**	0.630	0.865	**0.925**
1 : 1999	0.540	0.610	**0.940**	0.555	0.635	**0.920**

Table 3: The accuracy comparison on USPS for $N_l \in \{50, 10, 5, 1\}$.

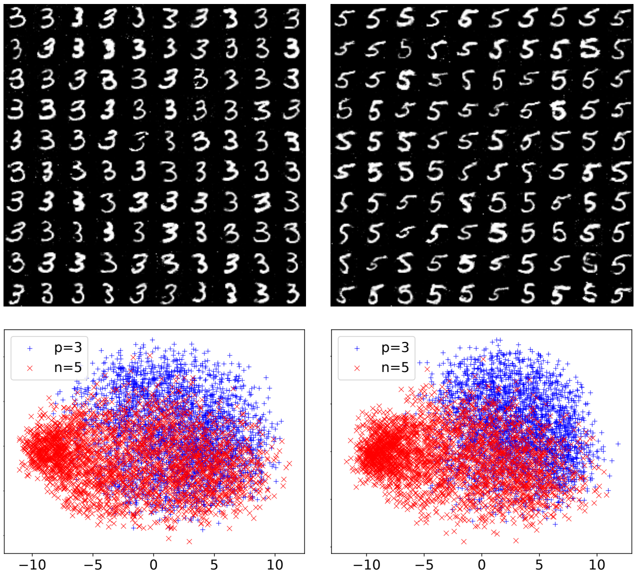

Figure 4: Top: visualization of positive (left) and negative (right) digits generated using one positive '3' label. Bottom: projected distributions of '3' vs '5', with ground truth (left) and generated (right).

4.3 celeb-a dataset

We are also interested in how GenPU performs on the real-life image set. In this set, the data is taken from CelebA dataset [Liu *et al.*, 2015] and resized to 64×64. We aim at classifying female from male using partially labeled male face images. To this end, the first $20,000$ male and $20,000$ female faces in CelebA are chosen as training set and the last $1,000$ faces are used as test set. Then, $2,000$ out of $20,000$ male faces are are randomly selected as positively labeled data. The architectures for generators and discriminators follow the design of the improved WGAN [Gulrajani *et al.*, 2017]. Figure 5 illustrates the generated male and female faces, indicating GenPU is capable of producing visually appealing and diverse images that belong to the correct categories. A deep PN classifier is then trained on the synthetic images and achieves the accuracy of 87.9 which is better than 86.8 of NNPU and 62.5 of UPU.

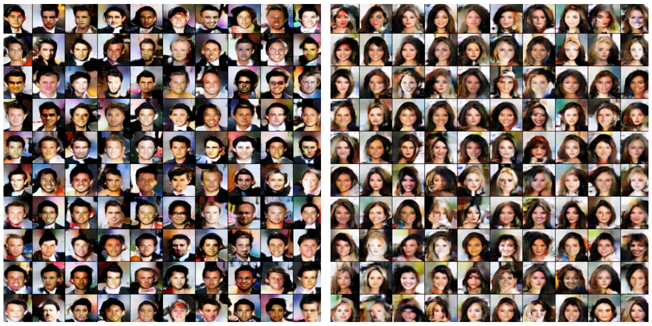

Figure 5: Visualization of male (left) and female (right) faces generated by GenPU on CelebA 64 × 64 data.

5 Discussion

One key factor to the success of GenPU relies on the capability of underlying GAN in generating diverse samples with high quality standard. Only in this way, the ideal performance could be achieved by training a flexible classifier on those samples. However, it is widely known that the perfect training of the original GAN is quite challenging. GAN suffers from issues of mode collapse and mode oscillation, especially when high-dimensional data distribution has a large number of output modes. For this reason, the similar issue of the original GAN may happen to our GenPU when a lot of output submodes exist. Since it has empirically been shown that the original GAN equipped with JSD inclines to mimic the mode-seeking process towards convergence. Fortunately, our framework is very flexible in the sense that it can be established by switching to different underlying GAN variants with more effective distance metrics (i.e., integral probability metric (IPG)) other than JSD (or f-divergence). By doing so, the possible issue of the missing modes can be greatly reduced. Another possible solution is to extend single generator G_p (G_n) to multiple generators $\{G_p^i\}_{i=1}^I$ $(\{G_n^j\}_{i=1}^J)$ for the positive (negative) class, also by utilizing the parameter sharing scheme to leverage common information and reduce the computational load. Another possible solution is to extend single generator G_p (G_n) to multiple generators $\{G_p^i\}_{i=1}^I$ $(\{G_n^j\}_{i=1}^J)$ for the positive (negative) class, also by utilizing the parameter sharing scheme to leverage common information and reduce the computational load.

Regarding other application domains, one direction of future work is to use GenPU for text classification. However, applying GAN to generating sequential semantically meaningful text is challenging, since GAN has difficulty in directly generating sequences of discrete tokens [Goodfellow *et al.*, 2016]. More sophisticated underlying GANs need to be developed for this purpose.

6 Acknowledgments

This work was partially supported by JSPS KAKENHI (Grant No. 17K00326), NSFC China (Grant No. 61773129) and JST GREST (Grant No. JPMJCR1784).

References

[Arjovsky *et al.*, 2017] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223, 2017.

[Denis *et al.*, 2005] François Denis, Rémi Gilleron, and Fabien Letouzey. Learning from positive and unlabeled examples. *Theoretical Computer Science*, 348(1):70–83, 2005.

[Du Plessis *et al.*, 2014] Marthinus C Du Plessis, Gang Niu, and Masashi Sugiyama. Analysis of learning from positive and unlabeled data. In *Advances in Neural Information Processing Systems*, pages 703–711, 2014.

[Du Plessis *et al.*, 2015] Marthinus Du Plessis, Gang Niu, and Masashi Sugiyama. Convex formulation for learning from positive and unlabeled data. In *International Conference on Machine Learning*, pages 1386–1394, 2015.

[Elkan and Noto, 2008] Charles Elkan and Keith Noto. Learning classifiers from only positive and unlabeled data. In *SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 213–220. ACM, 2008.

[Goodfellow *et al.*, 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[Goodfellow *et al.*, 2016] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*. MIT press Cambridge, 2016.

[Gulrajani *et al.*, 2017] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein GANs. In *Advances in Neural Information Processing Systems*, pages 5769–5779, 2017.

[Hido *et al.*, 2008] Shohei Hido, Yuta Tsuboi, Hisashi Kashima, Masashi Sugiyama, and Takafumi Kanamori. Inlier-based outlier detection via direct density ratio estimation. In *International Conference on Data Mining*, pages 223–232. IEEE, 2008.

[Jain *et al.*, 2016] Shantanu Jain, Martha White, and Predrag Radivojac. Estimating the class prior and posterior from noisy positives and unlabeled data. In *Advances in Neural Information Processing Systems*, pages 2693–2701, 2016.

[Kiryo *et al.*, 2017] Ryuichi Kiryo, Gang Niu, Marthinus C du Plessis, and Masashi Sugiyama. Positive-unlabeled learning with non-negative risk estimator. In *Advances in Neural Information Processing Systems*, pages 1674–1684, 2017.

[LeCun *et al.*, 1990] Yann LeCun, Bernhard E Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne E Hubbard, and Lawrence D Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems*, pages 396–404, 1990.

[LeCun *et al.*, 1998] Yann LeCun, Corinna Cortes, and Christopher JC Burges. The MNIST database of handwritten digits. *http://yann.lecun.com/exdb/mnist/*, 1998.

[Lee and Liu, 2003] Wee Sun Lee and Bing Liu. Learning with positive and unlabeled examples using weighted logistic regression. In *International Conference on Machine Learning*, pages 448–455, 2003.

[Li and Liu, 2003] Xiaoli Li and Bing Liu. Learning to classify texts using positive and unlabeled data. In *International Joint Conference on Artifical Intelligence*, pages 587–592, 2003.

[Li *et al.*, 2011] Wenkai Li, Qinghua Guo, and Charles Elkan. 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, 2011.

[Liu *et al.*, 2002] Bing Liu, Wee Sun Lee, Philip S Yu, and Xiaoli Li. Partially supervised classification of text documents. In *International Conference on Machine Learning*, pages 387–394, 2002.

[Liu *et al.*, 2003] Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip S Yu. Building text classifiers using positive and unlabeled examples. In *International Conference on Data Mining*, pages 179–186. IEEE, 2003.

[Liu *et al.*, 2015] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *International Conference on Computer Vision*, pages 3730–3738, 2015.

[Patrini *et al.*, 2016] Giorgio Patrini, Frank Nielsen, Richard Nock, and Marcello Carioni. Loss factorization, weakly supervised learning and label noise robustness. In *International Conference on Machine Learning*, pages 708–717, 2016.

[Salimans *et al.*, 2016] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.

[Ward *et al.*, 2009] Gill Ward, Trevor Hastie, Simon Barry, Jane Elith, and John R Leathwick. Presence-only data and the EM algorithm. *Biometrics*, 65(2):554–563, 2009.