Deep Generative Positive-Unlabeled Learning under Selection Bias

Byeonghu Na, Hyemi Kim, Kyungwoo Song, Weonyoung Joo, Yoon-Yeong Kim, and Il-Chul Moon
KAIST
Daejeon, Korea
{wp03052,khm0308,gtshs2,es345,yoonyeong.kim,icmoon}@kaist.ac.kr

ABSTRACT

Learning in the positive-unlabeled (PU) setting is prevalent in real world applications. Many previous works depend upon the *Selected Completely At Random* (SCAR) assumption to utilize unlabeled data, but the SCAR assumption is not often applicable to the real world due to selection bias in label observations. This paper is the first generative PU learning model without the SCAR assumption. Specifically, we derive the PU risk function without the SCAR assumption, and we generate a set of virtual PU examples to train the classifier. Although our PU risk function is more generalizable, the function requires PU instances that do not exist in the observations. Therefore, we introduce the VAE-PU, which is a variant of variational autoencoders to separate two latent variables that generate either features or observation indicators. The separated latent information enables the model to generate virtual PU instances. We test the VAE-PU on benchmark datasets with and without the SCAR assumption. The results indicate that the VAE-PU is superior when selection bias exists, and the VAE-PU is also competent under the SCAR assumption. The results also emphasize that the VAE-PU is effective when there are few positive-labeled instances due to modeling on selection bias.

CCS CONCEPTS

• **Computing methodologies** → **Learning latent representations**; **Semi-supervised learning settings**; **Neural networks**.

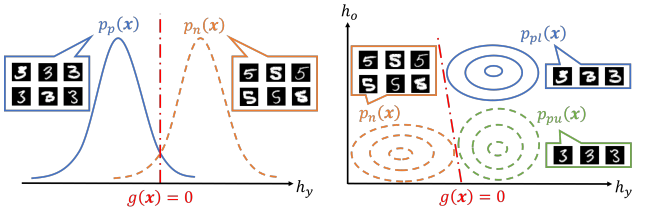
KEYWORDS

Positive-unlabeled learning; selection bias; variational autoencoders

ACM Reference Format:
Byeonghu Na, Hyemi Kim, Kyungwoo Song, Weonyoung Joo, Yoon-Yeong Kim, and Il-Chul Moon. 2020. Deep Generative Positive-Unlabeled Learning under Selection Bias. In *The 29th ACM International Conference on Information and Knowledge Management (CIKM '20), October 19–23, 2020, Virtual Event, Ireland*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3340531.3411971

CIKM '20, October 19–23, 2020, Virtual Event, Ireland

ACM ISBN 978-1-4503-6859-9/20/10…$15.00
https://doi.org/10.1145/3340531.3411971



(a) PU with SCAR **(b) PU with latent separation of observation and label**

Figure 1: Motivation of separation on the latent variable of label (h_y) and the latent variable of observation indicator (h_o) to overcome the Selected Completely At Random (SCAR) assumption. We illustrate the distribution that can be clarified from the data as the solid lines, and the unknown distribution from the data as the dotted lines. $g(x)$ is the label classifier that separates positive (p_p) and negative (p_n) distributions. If a bold font '3' is selectively labeled as positive, which violates SCAR, the data distributions of positive-labeled (p_{pl}) and positive-unlabeled (PU) (p_{pu}) become different. The previous PU learning with SCAR does not distinguish between p_{pl} and p_{pu} as (a). Therefore, our model takes into account the observation indicators of labeling, as well as the labels.

1 INTRODUCTION

Positive-unlabeled, or *PU*, learning is the learning of a binary classifier only with positively labeled data instances, with *positive* denoted as $y = +1$, which means that there are positive and negative instances without distinctions as *unlabeled*. This setting is different from an ordinary setting of the positive-negative, or *PN*, learning because PU learning does not provide a separate dataset to infer the distribution of negative instances, $p(x|y = -1)$. Although PU learning creates a challenge in estimating the distribution of negative cases, PU learning is a prevalent task in the field of recommendations [32], vision tasks [19], text classifications [20, 21], medical diagnosis predictions [4, 28], etc.

PU learning has been approached from two perspectives. The first approach assumes that the unlabeled instances consist of positive and negative instances by a proportion weight, so this approach provides a way in which to estimate a negative distribution from the inference made based on the unlabeled and positive distributions [17, 26, 27].

The second PU learning approach originates from utilizing the generative method to provide a set of *virtual* data instances. This approach has been a frequently used method for solving the open-set tasks [13, 22, 33] as well as data augmentations [1, 23, 30]. Because these venues share the characteristics of data scarcity, PU learning

can be viewed as a problem with negative example scarcity. For example, a variant of generative adversarial network (GAN) generates virtual negative instances to overcome negative data scarcity in the PU task [7].

Both PU learning approaches are built upon the assumption that positive labeling is imposed on randomly selected instances, which is known as the *Selected Completely At Random* (SCAR) assumption [6]. However, if we consider *selection bias* on labeled instances, the bias separates distribution on positive-unlabeled instances and distribution on positive-labeled instances [14].

Eventually, the selection bias invalidates SCAR in realistic scenarios. For example, in a recommendation system, purchased items are labeled as positive, and unlabeled items are considered to be negative. Now, let's introduce the concept of favorite items, which is a superset of the purchased items. If our recommendation system is intended to capture the favorite item set instead of the purchased items, the set of the unlabeled items does include the positive cases. This scenario indicates that the observed positive case, which is the purchased item, is not selected randomly from the favorite items. Here, the selection bias means that there is a reason, or bias, in selecting the purchased item from the favorite set. On the contrary, SCAR does not consider the existence of such a reason, so the purchase becomes an action related to random favorite items.

This paper introduces a new generative PU learning that does not depend on SCAR. To our knowledge, this is the first generative approach in the PU learning without SCAR. First, we derive the PU risk function without SCAR. Second, we need to generate the PU instances from a generative model under the new PU risk function. We designed our generative mechanism to be the latent separation between the positive-negative label and the label observation indicator (see Figure 1). Inherently, this model assumes that a data instance has two characteristics: one to determine its label and the other to determine whether it will be observed or not. Because the positive distributions depend on the observation indicator in the PU learning without SCAR, our model reflects the information of the observation indicator. The separated latent structure requires a new generative model, so we introduce the VAE-PU, which has two latent variables of the label and the observation indicator. Third, we present a joint learning structure of the VAE-PU and the risk function without SCAR, so the learning becomes the interaction between the virtual instance generations and the label classifier optimizations. Finally, we show that our method is the best performer in the PU learning task in the presence of selection bias.

2 PRELIMINARY

2.1 Model and Dataset Formulation

Before we review previous works and our model, we formulate PU learning as follows. This task assumes a binary classification of label $y \in \{+1, -1\}$ given input $x \in \chi \subset \mathbb{R}^D$. Let $p(x, y, o)$ be a joint probability distribution, where $o \in \{0, 1\}$ is an observation indicator. o is 1 if a label is observed, and 0 otherwise. The learning optimizes the parameters of the label classifier, $g : \mathbb{R}^D \to \mathbb{R}$, and the optimization is directed by a loss function, $l : \mathbb{R} \to \mathbb{R}_+$, from the margin of $yg(x)$. Eq. 1 specifies the risk function, which is aimed at reaching $y = \operatorname{sgn}(g(x))$, for a binary classification.

$$R(g) := \mathbb{E}_{(x,y)\sim p(x,y)}\left[l(yg(x))\right] \tag{1}$$

Here, we may select l from a set of variations, such as $l_{01}(z) = \frac{1-\operatorname{sgn}(z)}{2}$, $l_{\text{sig}}(z) = \frac{1}{1+e^z}$, etc. Now, the superscript of risk function R means the estimation from label classifier g, and the subscript of R is the origination of the data instance, which is true positive P, true negative N, or unlabeled U. For example, $R_P^+(g) := \mathbb{E}_{x\sim p(x|y=+1)}[l(+1 \times g(x))]$ is the risk function predicting the positive label (R^+) with a true positive instance (R_P).

PU learning is different from the conventional PN learning that the machine learning community is accustomed to, so we need to explicitly formulate the datasets due to potential confusion. First, PN learning assumes that the positive and the negative datasets are sampled independently, χ_P from $p_p(x) := p(x|y = +1)$ and χ_N from $p_n(x) := p(x|y = -1)$. Subsequently, the PN classification risk function becomes Eq. 2:

$$R_{\text{PN}}(g) := \pi_P R_P^+(g) + (1 - \pi_P)R_N^-(g) \tag{2}$$

where $\pi_P := p(y = +1)$ is a positive class prior. Second, in the dataset, PU learning differs from PN learning only by the removal of χ_N and by the adoption of the unlabeled dataset χ_U from $p(x)$ or $p(x|o = 0)$ (see section 2.3). This change in the dataset leads to a change in PU classification risk function $R_{\text{PU}}(g)$, which deviates from $R_{\text{PN}}(g)$.

2.2 PU Learning with SCAR

To reason $R_{\text{PU}}(g)$ further, we need a method to resolve R_P^+ and R_N^- with χ_U. SCAR is one solution to this resolution as shown in Eq. 3. Also, we provide a proposition that SCAR is equivalent to the case without the selection bias in PU learning.

Proposition 2.1. *The following statements are equivalent.*

$$\text{(a) } p(o = 1|x, y = +1) = p(o = 1|y = +1)\ \forall x \tag{3}$$

$$\text{(b) } p(x|y = +1, o = 1) = p(x|y = +1, o = 0) = p(x|y = +1)\ \forall x \tag{4}$$

Proof. Without loss of generality, we consider the $o = 1$ case.

$$(\Rightarrow)\quad p(x|y = +1, o = 1) = \frac{p(x, y = +1, o = 1)}{p(y = +1)p(o = 1|y = +1)}$$

$$= \frac{p(x, y = +1, o = 1)}{p(y = +1)p(o = 1|x, y = +1)} = \frac{p(x, y = +1)}{p(y = +1)} = p(x|y = +1)$$

$$(\Leftarrow)\quad p(o = 1|x, y = +1) = \frac{p(x, y = +1, o = 1)}{p(x|y = +1)p(y = +1)}$$

$$= \frac{p(x, y = +1, o = 1)}{p(x|y = +1, o = 1)p(y = +1)} = \frac{p(y = +1, o = 1)}{p(y = +1)}$$

$$= p(o = 1|y = +1)$$ □

Eq. 3 means that the positive-labeled (PL) instances are chosen from the positive instances independent of their feature information, x. Subsequently, the PL instances are identically distributed as the PU instances as shown in Eq. 4. Based on Proposition 2.1, SCAR is equivalent to the case without selection bias when the selection is the indication of the observation. Therefore, this paper's aim is to remove SCAR from the risk function.

Turning to the risk derivation, $R_{\text{PU}}(g)$, with SCAR, the unbiased PU (uPU) learning proposes the unbiased risk function of Eq. 1 [27].

$$R_{\text{uPU}}(g) := \pi_P R_P^+(g) - \pi_P R_P^-(g) + R_U^-(g) \tag{5}$$

$(1 - \pi)R_N^-(g) = R_U^-(g) - \pi R_P^-(g)$, which comes from the total law of probability, is applied to Eq. 5.

Furthermore, the non-negative PU (nnPU) learning resolves a potential overfitting problem of uPU [17].

$$R_{\mathrm{nnPU}}(g) := \pi_P R_P^+(g) + \max\left\{0, R_U^-(g) - \pi_P R_P^-(g)\right\} \tag{6}$$

The empirical estimator of $R_{\mathrm{uPU}}(g)$ approximates $R(g)$ as defined in $[0, \infty)$, yet the empirical estimator of $R_{\mathrm{uPU}}(g)$ is defined over $(-\infty, \infty)$. Therefore, $R_{\mathrm{nnPU}}(g)$ corrects this problem by introducing the rectifier by max operation.

Recently, generative models in PU learning have been developed, and these generative models still rely upon SCAR. GenPU is the first model to solve the PU task by generating *virtual* positive and negative instances with a GAN [7]. Similarly, PGAN uses a GAN to generate *virtual* negative instances to train a classifier [5]. PAN replaces the GAN generator with a classifier to produce *virtual* labels instead of *virtual* instances [9].

2.3 PU Learning without SCAR

Several works have proposed PU learning methods with a weaker assumption than SCAR. SAR-EM is based on SCAR without the *completely* part, by conditioning the selection of some features of the instances [3]. PUSB is a PU learning method under the *invariance of order* assumption [14]. The model of PUSB estimates density ratio scores using a pseudo empirical risk minimizer. The model determines the ranking of instances for being positive or negative, so the model can select only high-ranking instances for learning. These models still use a variant of $R_{\mathrm{nnPU}}(g)$.

There are two sampling schemes in PU learning with SCAR: the censoring scenario and the case-control scenario [6]. In the censoring scenario, the training set is drawn randomly from $p(x, y, o)$, and only x and o are recorded. In the case-control scenario, the PL set is drawn from $p_p(x)$, and the unlabeled set is drawn from $p(x)$. However, in PU learning without SCAR, we cannot guarantee that the PL set comes from $p_p(x)$ by Proposition 2.1. PUSB is based on the case-control scenario, so the PL set, or χ_{PL}, is drawn from $p_{pl}(x) := p(x|y = +1, o = 1)$, and the unlabeled set is drawn from $p(x)$. On the other hand, we focus on the censoring scenario that the unlabeled instances follow $p_u(x) := p(x|o = 0)$ to extract the observation information. Therefore, we define the unlabeled set, χ_U, which follows $p_u(x)$.

3 METHOD

This section introduces our generative model without SCAR, and this model is the first generative PU learning model without SCAR, to our knowledge. As we reviewed in Section 1, overcoming SCAR is important because SCAR means that there is no selection bias in the labeling, although there exists such bias in the real world.

We resolve this PU learning problem by separating the label and the observation so that they originate from two distinct latent variables. This suggests that our model embeds the latent information to determine the label regardless of the observation. Therefore, our model needs two specific capabilities: a classification without SCAR and a latent embedding with disentangled latent variables on labels and observations. To meet the capabilities, first, we define the risk function of the classifier that does not rely on SCAR, and second, we introduce a variant model of variational autoencoder (VAE) [16] for the disentanglement of labels and observations.

3.1 Risk Function without SCAR

This subsection derives the risk function without SCAR, which is the first derivation in the PU learning studies, to our knowledge. Our derivation starts from $R(g)$ in Eq. 1.

Theorem 3.1. *Let $p_{pl}(x) = p(x|y = +1, o = 1)$, $p_{pu}(x) = p(x|y = +1, o = 0)$, and $p_u(x) = p(x|o = 0)$. Then, the risk function of PU learning without SCAR, $R_{PU\neg SCAR}(g)$, can be expressed as*

$$\begin{aligned} R_{PU\neg SCAR}(g) =& p(y = +1, o = 1)\mathbb{E}_{x\sim p_{pl}(x)}[l(g(x))] \\ &+ p(y = +1, o = 0)\mathbb{E}_{x\sim p_{pu}(x)}[l(g(x)) - l(-g(x))] \\ &+ p(o = 0)\mathbb{E}_{x\sim p_u(x)}[l(-g(x))]. \end{aligned} \tag{7}$$

Proof.

$$\begin{aligned} &R_{\mathrm{PU}\neg\mathrm{SCAR}}(g) = \mathbb{E}_{x,y\sim p(x,y)}[l(yg(x))] \\ =&p(y = +1)\mathbb{E}_{x\sim p_p(x)}[l(g(x))] + p(y = -1)\mathbb{E}_{x\sim p_n(x)}[l(-g(x))] \\ =&p(y = +1)\mathbb{E}_{x\sim p_p(x)}[l(g(x))] + \{\mathbb{E}_{x\sim p(x)}[l(-g(x))] \\ &- p(y = +1)\mathbb{E}_{x\sim p_p(x)}[l(-g(x))]\} \\ =&p(y = +1)\mathbb{E}_{x\sim p_p(x)}[l(g(x)) - l(-g(x))] + \mathbb{E}_{x\sim p(x)}[l(-g(x))] \\ =&p(y = +1, o = 1)\mathbb{E}_{x\sim p_{pl}(x)}[l(g(x)) - l(-g(x))] \\ &+ p(y = +1, o = 0)\mathbb{E}_{x\sim p_{pu}(x)}[l(g(x)) - l(-g(x))] \\ &+ \mathbb{E}_{x\sim p(x)}[l(-g(x))] \\ =&p(y = +1, o = 1)\mathbb{E}_{x\sim p_{pl}(x)}[l(g(x))] \\ &+ p(y = +1, o = 0)\left\{\mathbb{E}_{x\sim p_{pu}(x)}[l(g(x)) - l(-g(x))]\right\} \\ &+ \left\{\mathbb{E}_{x\sim p(x)}[l(-g(x))] - p(o = 1)\mathbb{E}_{x\sim p_{pl}(x)}[l(-g(x))]\right\} \\ &(\because p(y = +1, o = 1) = p(o = 1) \text{ from assumption of PU learning}) \\ =&p(y = +1, o = 1)\mathbb{E}_{x\sim p_{pl}(x)}[l(g(x))] \\ &+ p(y = +1, o = 0)\left\{\mathbb{E}_{x\sim p_{pu}(x)}[l(g(x)) - l(-g(x))]\right\} \\ &+ p(o = 0)\mathbb{E}_{x\sim p_u(x)}[l(-g(x))] \\ &(\because \text{the toal law of probability}) \end{aligned}$$

□

Eventually, Eq. 7 is decomposed into three terms. The first term is classifying the PL instances into the positive set; the second term is classifying the PU instances into the positive set with a modified loss of the first term; and the last term is classifying the unlabeled instances into the negative set. Because we enforce the unlabeled instances to be negative by the last term, $p(o = 0)\mathbb{E}_{x\sim p_u(x)}[l(-g(x))]$, there should be a loss reduction to anticipate the PU cases, so we reduce the loss by $-p(y = +1, o = 0)\mathbb{E}_{x\sim p_{pu}(x)}[l(-g(x))]$, which is a part of the second loss term.

Whereas Eq. 7 is the theoretic derivation, Eq. 8 is the empirical estimator of $R_{PU\neg\mathrm{SCAR}}(g)$.

$$\begin{aligned} \hat{R}_{PU\neg\mathrm{SCAR}}(g) =& \frac{\pi_{PL}}{|\chi_{PL}|}\sum_{x^{(pl)}\in\chi_{PL}} l\left(g\left(x^{(pl)}\right)\right) + \frac{\pi_{PU}}{|\tilde{\chi}_{PU}|}\sum_{\tilde{x}^{(pu)}\in\tilde{\chi}_{PU}} l\left(g\left(\tilde{x}^{(pu)}\right)\right) \\ &+ \max\left\{\underbrace{0, -\frac{\pi_{PU}}{|\tilde{\chi}_{PU}|}\sum_{\tilde{x}^{(pu)}\in\tilde{\chi}_{PU}} l\left(-g\left(\tilde{x}^{(pu)}\right)\right) + \frac{\pi_U}{|\chi_U|}\sum_{x^{(u)}\in\chi_U} l\left(-g\left(x^{(u)}\right)\right)}_{(*)}\right\} \end{aligned} \tag{8}$$

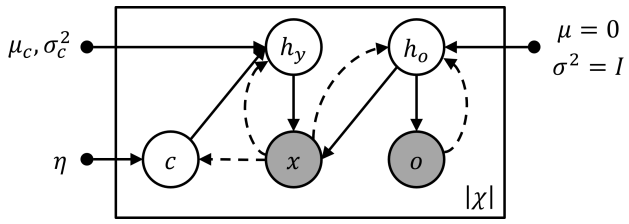


Figure 2: The graphical model of the VAE-PU. The solid lines denote the generative model p and the dashed lines denote the variational approximation q to p. The gray and white circles denote the observed variables and latent variables, respectively. $|\chi|$ is the number of entire data instances.

where $\tilde{\chi}_{PU}$ is the PU set drawn from $p_{pu}(x)$; π_* is the prior of dataset χ_*; and $|\chi_*|$ is the number of instances in dataset χ_*. Our setting assumes that positive class prior $\pi_P = p(y = +1)$ is known. Using the given π_P, we set other priors as $\pi_{PL} = \frac{|\chi_{PL}|}{|\chi|}$, $\pi_{PU} = \pi_P - \pi_{PL}$, and $\pi_U = 1 - \pi_{PL}$ where $\chi = \chi_{PL} \cup \chi_U$. There exist various methods for estimating the class prior in PU learning with SCAR [2, 25] and without SCAR [10]. Section 4 provides a sensitivity analysis for the estimation scenarios of π_P.

We use the max operation in $\hat{R}_{PU\neg\mathrm{SCAR}}(g)$ by following nnPU [17]. Although our risk function is a new contribution, as in nnPU, without the max operation, minimizing $\hat{R}_{PU\neg\mathrm{SCAR}}(g)$ with a deep neural network may fall into the overfitting problem because it is defined over $(-\infty, \infty)$. Specifically, $(*$ in Eq. 8$) \geq 0$ does not always satisfy, whereas the corresponding risk function terms in Eq. 7 are non-negative by Eq. 9.

$$p(o=0)\mathbb{E}_{x\sim p_u(x)}[l(-g(x))] - p(y=+1, o=0)\mathbb{E}_{x\sim p_{pu}(x)}[l(-g(x))] = p(y=-1)\mathbb{E}_{x\sim p_n(x)}[l(-g(x))] \geq 0 \quad (9)$$

To solve this problem, we also use a non-negative empirical estimation for $\hat{R}_{PU\neg\mathrm{SCAR}}(g)$.

Given $\hat{R}_{\mathrm{PU}\neg\mathrm{SCAR}}(g)$, it is now necessary to estimate the distribution of the PU cases due to the terms with $\tilde{\chi}_{PU}$. Our approach virtually generates such PU instances through a VAE structure customized for PU learning.

3.2 Variational PU Learning

To utilize the risk function in Eq. 7, we introduce the VAE-PU with two latent variables corresponding to the sources of the label and the observation. The VAE model can be represented as a probabilistic graphical model as well as a neural network structure, so we illustrate the model as Figures 2 and 3. The observed random variables are the feature vector (x) and the observation indicator (o). The latent random variables consist of the latent variable on label (h_y) and the latent variable on observation indicator (h_o).

3.2.1 Clustering on Virtual Positive and Negative Labels. Although the VAE-PU is designed to separate two latent variables of the labels and the observations, the VAE-PU introduces a clustering mechanism to resolve the positive cases in the unknown dataset. The challenge of PU learning comes from the positive cases buried in the unknown set, and we conjecture that the positive instances, regardless of the observation, would share the same latent variable of h_y. Because there is no supervision in χ_U, we seek a solution

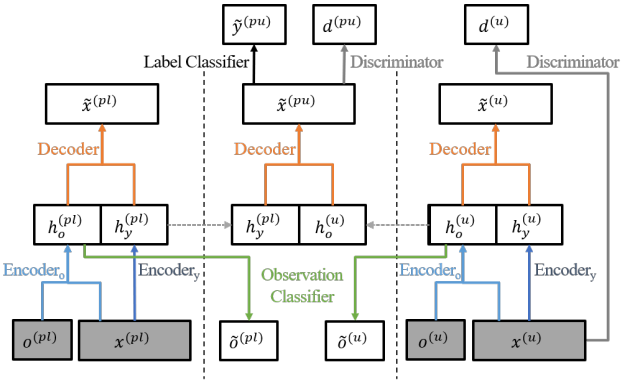


Figure 3: The neural network architecture of the VAE-PU. The solid lines represent the neural networks, and the dashed lines mean a copy of the variable. The superscript of the variable represents the dataset: (pl) is the PL set, (u) is the unlabeled set, and (pu) is the PU set.

from unsupervised learning, so we assign the instances of x to two clusters: positive and negative clusters. Therefore, we introduce the clustering mechanism in the VAE-PU, and c is the latent variable on cluster assignment. Fundamentally, c becomes a potential indication of an instance being positive even if the instance belongs to χ_U. Because c distinguishes the positive and the negative clusters, c is correlated to h_y, so the labeling can also be affected by c (see Figure 2). Technically, we model a prior with two Gaussian distributions in the VAE-PU, so the suggested clustering becomes fully integrated into the objective function of the VAE-PU.

3.2.2 Learning Objective of the VAE-PU. Our objective function of training the VAE-PU is the following:

$$\min_G \max_D (-\mathcal{L}_{\mathrm{ELBO}} + \alpha\mathcal{L}_{\mathrm{adv}} + \beta\mathcal{L}_{\mathrm{label}}) \quad (10)$$

where α and β are hyperparameters for balancing the losses; G consists of the encoder, the decoder, and the observation classifier of the VAE-PU; and D is the discriminator for adversarial regularization, which we discuss in Section 3.4. $\mathcal{L}_{\mathrm{ELBO}}$ is the function of the evidence lower bound (ELBO), which specifies how much the VAE-PU can represent the observed data; $\mathcal{L}_{\mathrm{adv}}$ is the adversarial generation loss, which adjusts the generated PU instances to yield an unlabeled attribute; and $\mathcal{L}_{\mathrm{label}}$ is the label loss, which adjusts the generated PU instances to yield a positive attribute.

Now, the optimization problem becomes a *min-max* game due to the adversarial generation loss. It should be noted that the maximization problem of the discriminator is affected only by the adversarial generation loss. The variational learning in the adversarial framework has been frequently used to measure the sample similarity [18, 31].

Maximizing $\mathcal{L}_{\mathrm{ELBO}}$ forces the model to compress and to reconstruct the observed variables. Minimizing $\mathcal{L}_{\mathrm{adv}}$ guides the discriminator not to distinguish the unlabeled instances and the virtual PU instances from the VAE-PU. With the maximization problem of the discriminator, the virtual PU instances become similar to the unlabeled instances in the ideal learning process. Minimizing $\mathcal{L}_{\mathrm{label}}$ requires the virtual PU instances to be estimated as a positive instance by the label classifier. Note that the label classifier

is not updated by Eq. 10, which becomes the responsibility of the alternating learning processes in Algorithm 1. To sum up, the optimization problem maximizes the likelihood of the model with two regularization terms, which enable the virtual PU instances to have the positive and the unlabeled characteristics.

3.3 Generative Process and Variational Inference

3.3.1 Generative Process. The VAE-PU assumes that the PU instances can be generated from a cluster in the latent space. Here, the generative process indicates that there are two clusters that represent *positive* when $c = 1$ and *negative* when $c = 0$, with prior probability η. After the selection of the two clusters, the latent information on the label is sampled from the cluster density, which is a Gaussian distribution with mean μ_c and variance σ_c^2. Meanwhile, the latent value on the observation indicator is sampled from a standard Gaussian distribution. After the sampling of two latent variables, the latent values are concatenated 1) to sample the feature information that we can observe in the dataset using decoder f, and 2) to sample the observation indicator using observation classifier f_o. The generative process of the VAE-PU is summarized below.

(1) $c \sim \text{Bern}(\eta)$
(2) $h_y|c \sim \mathcal{N}(\mu_c, \sigma_c^2 I)$
(3) $h_o \sim \mathcal{N}(0, I)$
(4)(a) $x|h_y, h_o \sim \text{Bern}(f(h_y, h_o))$ if x is binary
(b) $x|h_y, h_o \sim \mathcal{N}(f_\mu(h_y, h_o), f_{\sigma^2}(h_y, h_o)I)$ if x is real-valued
(5) $o|h_o \sim \text{Bern}(f_o(h_o))$

Note that I is the identity matrix, and f_μ and f_{σ^2} are the Gaussian mean and the Gaussian variance estimators of the decoder when x is a real value, respectively.

3.3.2 Inference on Label and Observation Latent Variables. The graphical model can be inferred with the variational method that matches our neural network autoencoder structure, as well. To follow the variational inference, we introduce a variational distribution $q(h_y, h_o, c|x, o)$ that can be factorized by the mean-field approximation (11).

We assume that the label-related latent variables, h_y and c, are independent of the label observation, o, given the features, x, of the instance (12). Although this independence assumption comes from PU learning, we inherit the conventional classifier assumption, for example, the latent of the label, h_y, generates label y, and the latent variable of the observation, h_o, determines the observation indicator, o. Overall, we factorize the variational distribution below.

$$q(h_y, h_o, c|x, o) = q(h_y|x, o)q(h_o|x, o)q(c|x, o) \tag{11}$$
$$= q(h_y|x)q(h_o|x, o)q(c|x) \tag{12}$$

Also, by the generative process of the VAE-PU, the model distribution can be factorized as below.

$$p(h_y, h_o, c, x, o) = p(c)p(h_y|c)p(h_o)p(o|h_o)p(x|h_y, h_o) \tag{13}$$

The definition of the variational distribution in Eq. 12 leads to the derivation of ELBO in Eq. 14.

$$\begin{aligned}
\log p(x, o) &= \log \int_{h_y}\int_{h_o}\sum_c q(h_y, h_o, c|x, o)\frac{p(h_y, h_o, c, x, o)}{q(h_y, h_o, c|x, o)}dh_o dh_y \\
&\geq \int_{h_y}\int_{h_o}\sum_c q(h_y, h_o, c|x, o)\log\frac{p(h_y, h_o, c, x, o)}{q(h_y, h_o, c|x, o)}dh_o dh_y \\
&= \mathbb{E}_{q(h_y,h_o,c|x,o)}[\log p(c) + \log p(h_y|c) + \log p(h_o) + \log p(o|h_o) \\
&\quad + \log p(x|h_y, h_o) - \log q(h_y|x) - \log q(h_o|x, o) - \log q(c|x)] \\
&=: \mathcal{L}^*_{\text{ELBO}}(x, o)
\end{aligned} \tag{14}$$

3.3.3 Neural Network from ELBO. We design the neural network structure, see Figure 3, by following the ELBO function of Eq. 14. Specifically, the variational distribution of q becomes the encoder structure as an encoder of $q(h_y|x)$ with the input of x, and the other encoder of $q(h_o|x, o)$ with the input of x and o. Next, the model distribution of p becomes the decoder structure as the decoder of $p(x|h_y, h_o)$ generating $\tilde{x}$. Similarly to the decoder, we introduce observation classifier $p(o|h_o)$ to generate the observation indicator, $\tilde{o}$. Finally, Eq. 15 is the formulation of the closed-form cluster assignment probability, $q(c|x)$.

Proposition 3.2 shows that Eq. 15 maximizes the ELBO function with respect to $q(c|x)$. This formulation is motivated by VaDE [12], but we provide proof since the ELBO function of our model is different from VaDE.

Proposition 3.2. *The maximum of $\mathcal{L}^*_{ELBO}(x, o)$ with respect to $q(c|x)$ is achieved when*

$$q(c|x) = p(c|h_y) = \frac{p(c)p(h_y|c)}{\sum_{c'\in\{0,1\}} p(c')p(h_y|c')}. \tag{15}$$

Proof.

$$\begin{aligned}
&\mathcal{L}^*_{\text{ELBO}}(x, o) \\
&= \int_{h_y}\int_{h_o}\sum_c q(h_y, h_o, c|x, o)\log\frac{p(h_y, h_o, c, x, o)}{q(h_y, h_o, c|x, o)}dh_o dh_y \\
&= \int_{h_y}\int_{h_o}\sum_c q(h_y|x)q(h_o|x, o)q(c|x)\times \\
&\qquad\qquad \log\frac{p(h_y)p(c|h_y)p(h_o)p(o|h_o)p(x|h_y, h_o)}{q(h_y|x)q(h_o|x, o)q(c|x)}dh_o dh_y \\
&= \int_{h_y}\int_{h_o} q(h_y|x)q(h_o|x, o)\left\{\sum_c q(c|x)\log\frac{p(c|h_y)}{q(c|x)}\right\}dh_o dh_y + B \\
&= \int_{h_y}\int_{h_o} q(h_y|x)q(h_o|x, o)D_{KL}(q(c|x) \parallel p(c|h_y))dh_o dh_y + B
\end{aligned} \tag{16}$$

B denotes the constant term with respect to c for simplicity, and D_{KL} denotes the Kullback-Leibler (KL) divergence. We use Eq. 12 for the factorization of the variational model and Eq. 17, which is derived by the chain rule of probability and conditional independence of variables, for the factorization of the generative model.

$$\begin{aligned}
&p(h_y, h_o, c, x, o) \\
&= p(h_y)p(c|h_y)p(h_o|h_y, c)p(o|h_o, h_y, c)p(x|o, h_o, h_y, c) \\
&= p(h_y)p(c|h_y)p(h_o)p(o|h_o)p(x|h_y, h_o)
\end{aligned} \tag{17}$$

From Eq. 16, we need to minimize the KL divergence of $q(c|x)$ and $p(c|h_y)$ to maximize the $\mathcal{L}^*_{\text{ELBO}}(x, o)$, so $\mathcal{L}^*_{\text{ELBO}}(x, o)$ is maximized

when $q(c|x) = p(c|h_y)$. Additionally, $p(c|h_y) = \frac{p(c)p(h_y|c)}{\sum_{c' \in \{0,1\}} p(c')p(h_y|c')}$ by Bayes' theorem. □

From the generative process and the neural network formulation, we formulate the ELBO estimator of whole data instances, denoted $\hat{\mathcal{L}}_{\mathrm{ELBO}}$, using the stochastic gradient variational Bayes estimator and the reparametrization trick [12, 16].

$$\hat{\mathcal{L}}^{\star}_{\mathrm{ELBO}}(x,o) = \frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{D}\left(x|_i \log \tilde{x}^{(l)}|_i + (1-x|_i)\log(1-\tilde{x}^{(l)}|_i)\right)$$
$$-\frac{1}{2}\sum_{c\in\{0,1\}}\gamma_c\sum_{j=1}^{J_y}\left(\log\sigma_c^2|_j + \frac{\sigma^2_{E_y} + (\mu_{E_y}|_j - \mu_c|_j)^2}{\sigma_c^2|_j}\right)$$
$$+\frac{1}{2}\sum_{j=1}^{J_y}\left(1+\log\sigma^2_{E_y}|_j\right) + \frac{1}{2}\sum_{j=1}^{J_o}\left(1+\log\sigma^2_{E_o}|_j - \sigma^2_{E_o}|_j - \mu^2_{E_o}|_j\right)$$
$$+\left(\sum_{c\in\{0,1\}}\gamma_c\log\frac{\pi_c}{\gamma_c}\right) + o\log(\tilde{o}) + (1-o)\log(1-\tilde{o})$$

$$\hat{\mathcal{L}}_{\mathrm{ELBO}} = \frac{1}{|\chi^{\star}|}\sum_{(x,o)\in\chi^{\star}}\hat{\mathcal{L}}^{\star}_{\mathrm{ELBO}}(x,o) \tag{18}$$

where L is the number of Monte Carlo samples for each feature vector; D is the dimension of the features; J_y is the dimension of the latent h_y; J_o is the dimension of the latent h_o; $\tilde{x}^{(l)}$ is the lth output of the decoder; $\tilde{o}$ is the output of the observation classifier; γ_c denotes $q(c|x)$, which is computed by Eq. 15; $*|_i$ denotes the ith element of $*$; μ_{E_*} and σ_{E_*} denote the output mean and variance of the encoder of $*$; and $\chi^{\star}$ is the dataset, which has the feature vectors of χ and the corresponding observation indicator.

3.3.4 Generation of Positive-Unlabeled Instances. Given the VAE-PU structure, it is feasible to generate the set of virtual PU instances, $\tilde{\chi}_{PU}$. To generate the virtual PU instances, first, we match the PL instances and the unlabeled instances. For example, we find the nearest latent vector of h_y in the unlabeled set. Second, we extract the latent variable on label from the PL instances, denoted $h_y^{(pl)}$, and the latent variable on observation indicator from the unlabeled instances, denoted $h_o^{(u)}$. Then, $h_y^{(pl)}$ and $h_o^{(u)}$ are concatenated to form the latent variable of the PU instances. Finally, by putting the concatenated vectors in the decoder, we obtain the feature vector of PU instances, denoted $\tilde{x}^{(pu)}$. Figure 3 describes the generation forward path of PU instances, $\tilde{x}^{(pu)}$, at the final decoder layer.

3.4 Adversarial Generation Loss

The virtual PU instances can imitate the feature distribution more closely by utilizing an adversarial structure and our label classifier. We introduce an adversarial generation loss $\mathcal{L}_{\mathrm{adv}}$ that uses a discriminator D.

$$\mathcal{L}_{\mathrm{adv}} = \mathbb{E}_{x\sim p_u(x)}\left[\log D\left(x\right)\right] + \mathbb{E}_{x\sim p_{pu}(x)}\left[\log\left(1-D\left(x\right)\right)\right] \tag{19}$$

The discriminator D minimizes $\mathcal{L}_{\mathrm{adv}}$ to distinguish the unlabeled instances and the virtual instances from the VAE-PU. On the other hand, the VAE-PU maximizes $\mathcal{L}_{\mathrm{adv}}$ to fool the discriminator. By this *min-max* optimization, the virtual instances from the VAE-PU tend to be similar to the instances of the unlabeled dataset. Using the given unlabeled dataset χ_U and the virtual PU dataset $\tilde{\chi}_{PU}$, we can estimate Eq. 19 by the empirical estimator.

$$\hat{\mathcal{L}}_{\mathrm{adv}} = \frac{1}{|\chi_U|}\sum_{x^{(u)}\in\chi_U}\log D\left(x^{(u)}\right) + \frac{1}{|\tilde{\chi}_{PU}|}\sum_{\tilde{x}^{(pu)}\in\tilde{\chi}_{PU}}\log\left(1-D\left(\tilde{x}^{(pu)}\right)\right) \tag{20}$$

3.5 Label Loss

To generate the set of PU instances, $\tilde{\chi}_{PU}$, with the positive property, we also introduce a label loss $\mathcal{L}_{\mathrm{label}}$ that uses a label classifier g.

$$\mathcal{L}_{\mathrm{label}} = \mathbb{E}_{x\sim p_{pu}(x)}\left[l\left(g\left(x\right)\right)\right] \tag{21}$$

The label classifier g provides the loss of a positive margin. By minimizing $\mathcal{L}_{\mathrm{label}}$, the virtual PU instances that are generated by the VAE-PU have a positive property. We can also estimate Eq. 21 by the empirical estimator using the virtual PU dataset $\tilde{\chi}_{PU}$.

$$\hat{\mathcal{L}}_{\mathrm{label}} = \frac{1}{|\tilde{\chi}_{PU}|}\sum_{\tilde{x}^{(pu)}\in\tilde{\chi}_{PU}} l\left(g\left(\tilde{x}^{(pu)}\right)\right) \tag{22}$$

Using the adversarial generation loss and the label loss, $\tilde{x}^{(pu)}$ becomes similar to the unlabeled instance and has a positive property.

3.6 Learning of Label Classifier and VAE-PU

Algorithm 1 specifies a learning process of the label classifier and the VAE-PU. The learning process of the label classifier with our new risk function, Eq. 7, and the VAE-PU is composed of two alternating steps. This alternation is inherently necessary because the virtual example generation needs to be tuned by the label classifier, and vice versa.

Our first step is the *min-max* optimization of the ELBO estimator from the VAE-PU (18), the adversarial generation loss estimator (20), and the label loss estimator (22) with respect to the VAE-PU and the discriminator.

$$\min_G \max_D (-\hat{\mathcal{L}}_{\mathrm{ELBO}} + \alpha\hat{\mathcal{L}}_{\mathrm{adv}} + \beta\hat{\mathcal{L}}_{\mathrm{label}}) \tag{23}$$

The second step starts from the sampled virtual PU instances, which enable the Monte-Carlo estimation of the risk function without SCAR (8), and the second step optimizes the label classifier in Line 11 of Algorithm 1.

$$\min_g \hat{R}_{PU\neg\mathrm{SCAR}}(g) \tag{24}$$

3.7 Complexity Analysis

We compare the time complexity for baselines and our model. Table 1 provides the list of baselines. The time complexity for uPU, nnPU, GenPU, PUbN\N, and PAN is $O\left(\left(n_{PL}+n_U\right)DH\right)$, where D is the dimension of the feature vectors, H is the dimension of the embedding layer, and n_{PL} and n_U are the numbers of the PL instances and unlabeled instances, respectively. PUSB has two stages: training a network and finding a threshold. Training a network takes $O\left(\left(n_{PL}+n_U\right)DH\right)$, and finding a threshold by ranking the instances takes $O\left(\left(n_{PL}+n_U\right)^2\right)$. Therefore, PUSB takes $O\left(\left(n_{PL}+n_U\right)\left(\left(n_{PL}+n_U\right)+DH\right)\right)$.

Our model also has two stages: training the VAE-PU and training the label classifier. In both training procedures, the evaluation of networks takes $O\left(\left(n_{PL}+n_U\right)DH\right)$. We also need to generate virtual PU instances. Because we find the appropriate unlabeled

Algorithm 1: The learning algorithm of the label classifier and the VAE-PU

Input: Positive-labeled dataset χ_{PL}, unlabeled dataset χ_U, batch size of positive-labeled dataset m_{pl} and unlabeled dataset m_u, loss trade-off parameters α and β, and pre-trained parameters π, μ_c, and σ_c

Output: VAE-PU parameters, discriminator parameters, and label classifier parameters

1 **while** *not converged* **do**
2 **for** *VAE-PU training iterations* **do**
3 sample minibatch of positive-labeled examples $\left\{x_i^{(pl)}\right\}_{i=1}^{m_{pl}}$ from χ_{PL}
4 sample minibatch of unlabeled examples $\left\{x_i^{(u)}\right\}_{i=1}^{m_u}$ from χ_U
5 generate virtual positive-unlabeled examples $\left\{\tilde{x}_i^{(pu)}\right\}_{i=1}^{m_{pl}} =: \tilde{\chi}_{PU}$
6 update discriminator with the adversarial generation loss $\hat{\mathcal{L}}_{\text{adv}}$
7 update VAE-PU (encoder, decoder, observation classifier) with Eq. 23
8 **end for**
9 **for** *label classifier training iterations* **do**
10 obtain $\left\{x_i^{(pl)}\right\}_{i=1}^{m_{pl}}$, $\left\{x_i^{(u)}\right\}_{i=1}^{m_u}$, and $\left\{\tilde{x}_i^{(pu)}\right\}_{i=1}^{m_{pl}}$ by Line 3-5
11 update the label classifier with Eq. 24
12 **end for**
13 **end while**

instances for each PL instance by comparing the distances in the latent space, this takes $O\left(n_{PL}n_U H^*\right)$, where H^* is the dimension of the latent variables. However, there exists a method that takes constant average time to find the approximate nearest neighbor [29]. Then, we reduce the time complexity of generating instances as $O\left(n_{PL}H^*\right)$. Therefore, our model takes $O\left(\left(n_{PL}+n_U\right)DH\right)$ because H^* is usually smaller than H.

4 EXPERIMENT

4.1 Experiment Settings

4.1.1 Datasets and Baselines. All benchmark datasets are not binary classes, so we construct the positive (P) and the negative (N) classes for three benchmark datasets [7, 17].

(1) MNIST. We perform two tasks: '3 vs 5' and 'Even vs Odd' ('E vs O'). For the '3 vs 5' task, we sample 5,000 '3' digit images as positive and 5,000 '5' digit images as negative. For the 'E vs O' task, we use all MNIST images by labeling even numbers as positive and odd numbers as negative. **(2) CIFAR-10.** The P class is composed of 'airplane,' 'automobile,' 'ship,' and 'truck,' and the N class consists of 'bird,' 'cat,' 'deer,' 'dog,' 'frog,' and 'horse'. **(3) 20 Newsgroups.** The P class consists of 'alt.,' 'comp.,' 'misc.,' and 'rec.' topics, and the N class includes the documents of 'sci,., 'soc.,' and 'talk.' topics.

On CIFAR-10, we use the pre-trained embedding vectors from [11] to extract the abstract information of images. On 20 Newsgroups, we use pre-trained embedding vectors from [24] to extract the abstract information of documents.

Table 1: The list of baselines and our model.

Model	Architecture	SCAR
uPU [27]	Classifier only	Yes
nnPU [17]	Classifier only	Yes
PUbN\N [8]	Classifier only	Yes
GenPU [7]	GAN	Yes
PAN [9]	Classifier & Discriminator	Yes
PUSB [14]	Classifier only	No
VAE-PU (Ours)	VAE & GAN	No

Our major comparison aspect is the impact of SCAR on the PU learning. Therefore, we keep SCAR in the *Random* setting and remove SCAR in the *Bias* setting. Here, the *Random* setting labels the positive instances without further distinctions besides the above positive criteria. Hence, we label 100 images of '3' in MNIST, 1,000 images of 'E' in MNIST, 100 images of vehicle images in CIFAR-10, and 1,000 documents of positive topics in 20 Newsgroups. In contrast, the *Bias* setting selects instances to label with a selection bias. Therefore, we label 100 images for *bold* '3' in MNIST, 1,000 images for *bold* 'E' in MNIST, 100 images for only *automobile* and *airplane* images in CIFAR-10, and documents for only articles with *mac* and *score* in 20 Newsgroups.

We use six baseline models, as specified in Table 1. We select three risk-based PU learning models and two generative PU learning models. Also, we select PUSB as another baseline model, which anticipates the selection bias.

4.1.2 VAE-PU Architectures. For MNIST and CIFAR-10, we use a 500-D 2-layer fully connected neural network for the encoder and the decoder of the VAE-PU, a fully connected neural network without the hidden layer for the observation classifier of the VAE-PU, a 256-D 1-layer fully connected neural network for the discriminator, and a 300-D 4-layer fully connected neural network for the label classifier. For 20 Newsgroups, we use a 500-D 4-layer fully connected neural network for the encoder and the decoder of the VAE-PU, a 2-D 1-layer fully connected neural network for the observation classifier of the VAE-PU, a 256-D 1-layer fully connected neural network for the discriminator, and a 300-D 2-layer fully connected neural network for the label classifier. Note that we use the same architecture of the label classifier for all models for each dataset except uPU, in which we experienced overfitting.

We use the Adam optimizer [15], and we utilize the sigmoid loss as the surrogate loss of the label classifier for all models. We provide a true positive class prior π_P for all models. Similar to VaDE, we pre-train the encoder and the decoder of the VAE-PU using the reconstruction loss [12]. Also, we use the Gaussian mixture model on h_y to find the hyperparameters of our model prior to the experiment. Finally, we train the encoder, the decoder, the observation classifier, and the discriminator at the initial stage of learning. After a preset number of iterations, we train these structures and the label classifier, alternatively.

4.2 Results and Analysis

Table 2 provides the average accuracy of baselines and our model by replicating the 10 times. For *Random*, which holds SCAR, our model shows comparable performances among the baselines. For

Table 2: Average accuracy and standard deviation of each dataset (%). The experiment was repeated 10 times for each dataset and model. The bold numbers indicate the best accuracy, and the underlineed numbers indicate that the p-value is larger than 0.05 against the best accuracy based on a t-test.

Labeling	Data	uPU	nnPU	PUbN\N	GenPU	PAN	PUSB	VAE-PU (Ours)
Random (With SCAR)	MNIST 35	88.67 ± 0.30	**93.70 ± 0.77**	93.31 ± 0.90	89.90 ± 1.49	92.58 ± 0.99	93.04 ± 0.56	92.50 ± 0.66
	MNIST EO	85.71 ± 0.08	93.98 ± 0.45	94.72 ± 0.22	76.25 ± 1.73	93.43 ± 0.16	**95.26 ± 0.25**	94.21 ± 0.26
	CIFAR-10	60.88 ± 3.94	91.85 ± 0.48	91.69 ± 0.63	69.65 ± 9.03	92.18 ± 1.10	**92.44 ± 0.42**	92.00 ± 0.66
	20News	60.92 ± 4.03	86.99 ± 0.58	**88.05 ± 0.17**	69.79 ± 7.65	87.51 ± 0.29	82.85 ± 1.32	85.40 ± 0.82
Bias (Without SCAR)	MNIST 35	64.09 ± 0.50	63.91 ± 4.62	73.76 ± 1.34	73.54 ± 10.5	67.29 ± 1.44	66.55 ± 1.54	**76.25 ± 2.36**
	MNIST EO	62.75 ± 0.16	67.44 ± 1.35	70.10 ± 1.66	66.03 ± 1.09	67.49 ± 1.27	67.68 ± 0.65	**73.29 ± 1.50**
	CIFAR-10	59.86 ± 3.65	75.06 ± 0.98	76.59 ± 0.27	62.41 ± 5.71	74.62 ± 2.63	72.96 ± 0.87	**78.19 ± 2.17**
	20News	54.95 ± 4.36	68.38 ± 2.68	79.13 ± 2.13	59.05 ± 8.93	73.91 ± 0.19	78.29 ± 1.39	**79.66 ± 1.68**

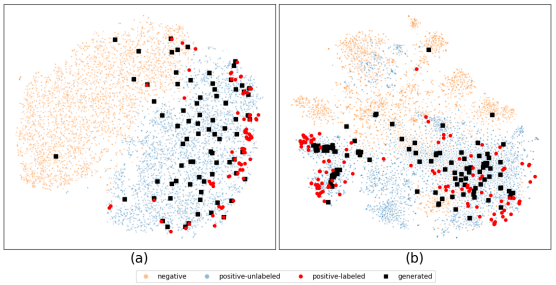


Figure 4: t-SNE visualizations in *Bias* on (a) MNIST '3 vs 5' and (b) 20 Newsgroups.

Bias, which does not keep SCAR, our model shows the best performance among the baselines. Although PUSB is a model without SCAR, the accuracy is lower than for other baselines in *Bias*. The low performance of PUSB could originate from the invalidated invariance of order assumption. On the other hand, PUSB shows good performance in *Random* since the invariance of order assumption implies SCAR inherently.

We perform t-SNE to analyze where the generated PU data instances are located in the data distribution on the *Bias* setting. In Figure 4, the orange circles are the negative instances, and the blue circles are the PU instances. Here, the red circles are the PL instances that do not cover the entire blue distribution of p_p. In spite of these biased observations on the positive labels, the black squares are the virtually generated PU instances from the VAE-PU, so the p_{pu} could be estimated without SCAR. In some results, the PL instances are concentrated in some clusters because the PL instances rarely exist in the rest of the clusters. Therefore, the model has difficulty generating PU instances in the rest of the clusters. That being said, if the model has PL instances from non-concentrated clusters, the model is able to capture the PU distribution of these clusters.

Also, we inspect the t-SNE visualization to ensure the separation of h_y and h_o. Figure 5 shows that h_y is appropriately divided in *Random* and *Bias*. Additionally, h_o is not distinguishable between the labeled and the unlabeled instances in *Random*, but it is well differentiated in the labeled and the unlabeled instances in *Bias*. The results imply that h_y contains the label information and h_o contains the observation information, as intended.

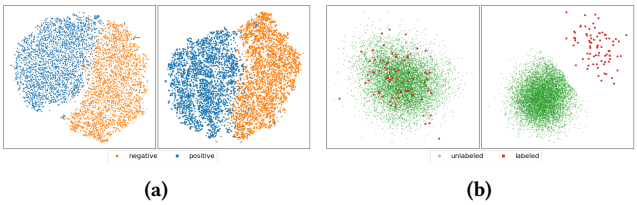
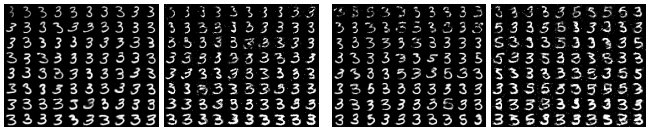


Figure 5: t-SNE visualizations of the label latent variable (h_y) and the observation indicator latent variable (h_o) on the MNIST '3 vs 5' task. (a) h_y for *Random* (left) and *Bias* (right); (b) h_o for *Random* (left) and *Bias* (right).



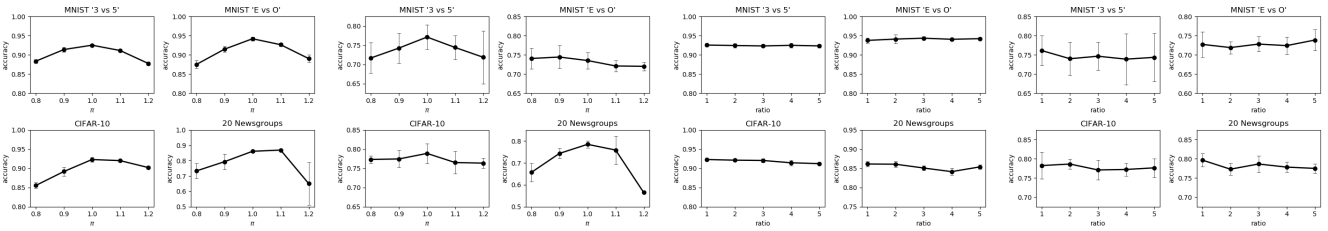
(a) VAE-PU (*Random*) **(b) GenPU (*Random*)** **(c) VAE-PU (*Bias*)** **(d) GenPU (*Bias*)**

Figure 6: Generated images comparison of VAE-PU and GenPU on the MNIST '3 vs 5' task.

Particularly, we compare our model to GenPU because both are recent models solving the PU learning from the generative modeling perspective. As our performance gain comes from the virtual example from p_{pu}, we compare the utilized virtual examples from the two models. Figure 6 demonstrates two results. First, there is no distinction between the generated images in *Random*. Second, there is a distinction between the generated images in *Bias* because GenPU intensely samples the bold font of '3' due to the selective labeling on the bold fonts. On the other hand, VAE-PU samples the whole variations of '3' because the label and the observation latent variables are separated.

Figure 8 shows the change of the PU risk value, Eq. 8, over the training of the VAE-PU and GenPU. For each model, we put the generated instances as PU data into Eq. 8. The VAE-PU can minimize the PU risk more than GenPU in *Random* and *Bias*. This verifies that the generated data from GenPU faces difficulty in capturing the PU data distribution without SCAR.

We experiment with the sensitivity of our model by the number of PL instances. Figure 9 represents that our generative model



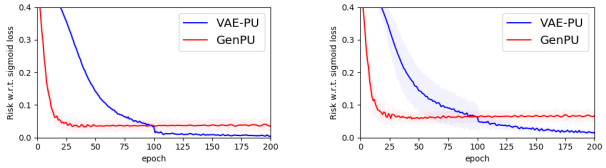
(a) Varying the number of generated samples in *Random*

(b) Varying the number of generated samples in *Bias*

(c) Varying π_P in *Random*

(d) Varying π_P in *Bias*

Figure 7: Average accuracy and standard deviation of varying (a-b) the ratio of the number of generated samples to the number of given PL samples on each dataset and (c-d) π_P on each dataset. The experiment was repeated 10 times for each dataset and model.



(a) *Random*

(b) *Bias*

Figure 8: The means and standard deviations of training PU¬SCAR risk function per epoch based on the 10 repeated experiments of the MNIST '3 vs 5' task.

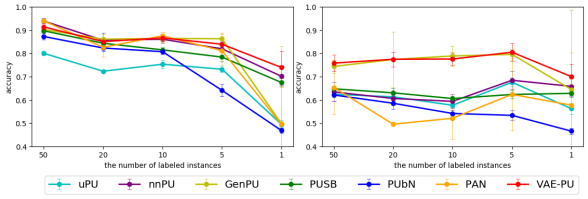


Figure 9: Average accuracy and standard deviation of varying the number of PL instances in the MNIST '3 vs 5' task for (left) *Random* and (right) *Bias*. The experiment was repeated 10 times for each dataset and model.

(a) *Random*

(b) *Bias*

Figure 10: Generated images from given PL images (Column 1, red box) of each experiment on the MNIST '3 vs 5' task. The columns represent the generated image of the experiment with (Column 2) $\alpha = 0$, $\beta = 0$; (Column 3) $\alpha = \alpha^*$, $\beta = 0$; (Column 4) $\alpha = 0$, $\beta = \beta^*$; and (Column 5) $\alpha = \alpha^*$, $\beta = \beta^*$.

performs well in *Random* and *Bias* with few PL instances. GenPU, another generative model of the PU learning, also has higher performance than the classifier-based models. Additionally, most of the models increase performance when there are five PL instances compared to ten PL instances in the *Bias* setting. We conjecture that the performance drop comes from the selection bias that is less affected when the number of PL instances is extremely small. Therefore, we see the accuracy of the *Bias* setting is similar to that of the *Random* setting in cases with much fewer PL instances.

We control the hyperparameter π_P to conduct a sensitivity analysis for misestimation of π_P. From the setting of sensitivity analysis in nnPU [17], we experiment with five cases of π_P: $0.8\pi_P^*$, $0.9\pi_P^*$, π_P^*, $1.1\pi_P^*$, and $1.2\pi_P^*$, where π_P^* is a true positive class prior. Other experiment settings are the same as before. Figure 7a and 7b show the results of sensitivity analysis for misestimation of π_P. In the *Bias* setting, the over-misestimation hurts more than the underestimation because the over-misestimation of π_P directly increases the influence of the generated PU instances, since $\pi_{PU} = \pi_P - \pi_{PL}$ in our setting.

We experiment on the over-sampling of the generated PU instances, $\tilde{x}^{(pu)}$, to see the impact. Figure 7c and 7d represent that there is no significant difference in the results depending on the number of generated PU samples for all experiments.

Figure 10 provides an ablation study to find the influence of the adversarial generation loss and the label loss. We compare the generated images of four cases, adjusting the hyperparameters α and β in Eq. 23: (1) $\alpha = 0$, $\beta = 0$; (2) $\alpha = \alpha^*$, $\beta = 0$; (3) $\alpha = 0$, $\beta = \beta^*$; and (4) $\alpha = \alpha^*$, $\beta = \beta^*$, where α^* and β^* are the proper hyperparameters found through the hyperparameter tuning. In the *Random* setting, all cases generate images similar to the given PL images because the PL distribution and PU distribution are similar in the *Random* setting. On the other hand, in the *Bias* setting, the generated images become different for each experiment. When we remove the adversarial loss and the label loss (column 2 in Figure 10b), the generated images are similar to the given PL images. When we only add the adversarial generation loss (column 3 in Figure 10b), the generated images are relatively thin digit images. In particular, some generated images are thin '5' images because the adversarial generation loss forces sampling images similar to the unlabeled images. With both adversarial generation loss and label loss, the model generates thin '3' images, as we expected. Therefore, we have found that the label loss helps the generated images be positive.

5 CONCLUSION

We propose a generative PU learning method, the VAE-PU, without the SCAR assumption. When we do not assume SCAR, the risk function requires positive-unlabeled instances that do not exist in the observations. Therefore, we design a deep generative model to virtually generate the unobserved data of positive-unlabeled

instances. Technically, we derive the risk function without SCAR. Another technical contribution is separating latent variables on the label and the observation indicator through the VAE-PU, so the generated examples can be utilized to derive the risk function of the PU learning. To our knowledge, this is the first generative model without SCAR in the PU learning community. This generative approach is naturally called for because the generation is necessary for solving such biased observations.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their valuable comments and helpful suggestions on our manuscript. This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF), funded by the Ministry of Education (NRF-2019M3F2A1072239).

REFERENCES

[1] Antreas Antoniou, Amos Storkey, and Harrison Edwards. 2017. Data Augmentation Generative Adversarial Networks. arXiv:1711.04340 [stat.ML]

[2] Jessa Bekker and Jesse Davis. 2018. Estimating the class prior in positive and unlabeled data through decision tree induction, In Proceedings of the 32th AAAI Conference on Artificial Intelligence. *Proceedings of the 32th AAAI Conference on Artificial Intelligence*, 2712–2719.

[3] Jessa Bekker, Pieter Robberechts, and Jesse Davis. 2020. Beyond the Selected Completely at Random Assumption for Learning from Positive and Unlabeled Data. In *Machine Learning and Knowledge Discovery in Databases*, Ulf Brefeld, Elisa Fromont, Andreas Hotho, Arno Knobbe, Marloes Maathuis, and Céline Robardet (Eds.). Springer International Publishing, Cham, 71–85.

[4] Luigi Cerulo, Charles Elkan, and Michele Ceccarelli. 2010. Learning gene regulatory networks from only positive and unlabeled data. *BMC Bioinformatics* 11, 1 (2010), 228. https://doi.org/10.1186/1471-2105-11-228

[5] F. Chiaroni, M. Rahal, N. Hueber, and F. Dufaux. 2018. Learning with A Generative Adversarial Network From a Positive Unlabeled Dataset for Image Classification. In *2018 25th IEEE International Conference on Image Processing (ICIP)*. 1368–1372. https://doi.org/10.1109/ICIP.2018.8451831

[6] Charles Elkan and Keith Noto. 2008. Learning Classifiers from Only Positive and Unlabeled Data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Las Vegas, Nevada, USA) *(KDD '08)*. Association for Computing Machinery, New York, NY, USA, 213–220. https://doi.org/10.1145/1401890.1401920

[7] Ming Hou, Brahim Chaib-Draa, Chao Li, and Qibin Zhao. 2018. Generative Adversarial Positive-Unlabeled Learning. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence* (Stockholm, Sweden) *(IJCAI'18)*. AAAI Press, 2255–2261.

[8] Yu-Guan Hsieh, Gang Niu, and Masashi Sugiyama. 2019. Classification from Positive, Unlabeled and Biased Negative Data. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.), Vol. 97. PMLR, Long Beach, California, USA, 2820–2829.

[9] Wenpeng Hu, Ran Le, Bing Liu, Feng Ji, Haiqing Chen, Dongyan Zhao, Jinwen Ma, and Rui Yan. 2020. Learning from Positive and Unlabeled Data with Adversarial Training. https://openreview.net/forum?id=HygPjlrYvB

[10] Shantanu Jain, Martha White, and Predrag Radivojac. 2016. Estimating the class prior and posterior from noisy positives and unlabeled data. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett (Eds.). 2685–2693.

[11] Xu Ji, Andrea Vedaldi, and João F. Henriques. 2019. Invariant Information Clustering for Unsupervised Image Classification and Segmentation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (2019), 9864–9873.

[12] Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. 2017. Variational Deep Embedding: An Unsupervised and Generative Approach to Clustering. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence* (Melbourne, Australia) *(IJCAI'17)*. AAAI Press, 1965–1972.

[13] Inhyuk Jo, Jungtaek Kim, Hyohyeong Kang, Yong-Deok Kim, and Seungjin Choi. 2018. Open Set Recognition by Regularising Classifier with Fake Data Generated by Generative Adversarial Networks. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2686–2690.

[14] Masahiro Kato, Takeshi Teshima, and Junya Honda. 2019. Learning from Positive and Unlabeled Data with a Selection Bias. In *International Conference on Learning Representations*.

[15] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.).

[16] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.).

[17] Ryuichi Kiryo, Gang Niu, Marthinus C du Plessis, and Masashi Sugiyama. 2017. Positive-Unlabeled Learning with Non-Negative Risk Estimator. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 1675–1685.

[18] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. 2016. Autoencoding beyond pixels using a learned similarity metric. In *Proceedings of The 33rd International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Maria Florina Balcan and Kilian Q. Weinberger (Eds.), Vol. 48. PMLR, New York, New York, USA, 1558–1566.

[19] W. Li, Q. Guo, and C. Elkan. 2011. A Positive and Unlabeled Learning Algorithm for One-Class Classification of Remote-Sensing Data. *IEEE Transactions on Geoscience and Remote Sensing* 49, 2 (Feb 2011), 717–725. https://doi.org/10.1109/TGRS.2010.2058578

[20] Xiaoli Li and Bing Liu. 2003. Learning to Classify Texts Using Positive and Unlabeled Data. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence* (Acapulco, Mexico) *(IJCAI'03)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 587–592.

[21] Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip S. Yu. 2003. Building Text Classifiers Using Positive and Unlabeled Examples. In *Proceedings of the Third IEEE International Conference on Data Mining (ICDM '03)*. IEEE Computer Society, USA, 179.

[22] Lawrence Neal, Matthew Olson, Xiaoli Fern, Weng-Keen Wong, and Fuxin Li. 2018. Open set learning with counterfactual images. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 613–628.

[23] Augustus Odena, Christopher Olah, and Jonathon Shlens. 2017. Conditional image synthesis with auxiliary classifier gans. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2642–2651.

[24] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, New Orleans, Louisiana, 2227–2237. https://doi.org/10.18653/v1/N18-1202

[25] Marthinus Plessis, Gang Niu, and Masashi Sugiyama. 2016. Class-prior Estimation for Learning from Positive and Unlabeled Data. *Machine Learning* (11 2016). https://doi.org/10.1007/s10994-016-5604-6

[26] Marthinus C. du Plessis, Gang Niu, and Masashi Sugiyama. 2014. Analysis of Learning from Positive and Unlabeled Data. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1* (Montreal, Canada) *(NIPS'14)*. MIT Press, Cambridge, MA, USA, 703–711.

[27] Marthinus Du Plessis, Gang Niu, and Masashi Sugiyama. 2015. Convex Formulation for Learning from Positive and Unlabeled Data. In *Proceedings of the 32nd International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Francis Bach and David Blei (Eds.), Vol. 37. PMLR, Lille, France, 1386–1394.

[28] Jing Ren, Qian Liu, John Ellis, and Jinyan Li. 2015. Positive-unlabeled learning for the prediction of conformational B-cell epitopes. *BMC Bioinformatics* 16, 18 (2015), S12. https://doi.org/10.1186/1471-2105-16-S18-S12

[29] Enrique [Vidal Ruiz]. 1986. An algorithm for finding nearest neighbours in (approximately) constant average time. *Pattern Recognition Letters* 4, 3 (1986), 145 – 157. https://doi.org/10.1016/0167-8655(86)90013-9

[30] Toan Tran, Trung Pham, Gustavo Carneiro, Lyle Palmer, and Ian Reid. 2017. A bayesian data augmentation approach for learning deep models. In *Advances in neural information processing systems*. 2797–2806.

[31] Yongqin Xian, Saurabh Sharma, Bernt Schiele, and Zeynep Akata. 2019. F-VAEGAN-D2: A Feature Generating Framework for Any-Shot Learning. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), 10267–10276.

[32] Jinfeng Yi, Cho-Jui Hsieh, Kush R. Varshney, Lijun Zhang, and Yao Li. 2017. Scalable Demand-Aware Recommendation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 2409–2418.

[33] Sergey Demyanov Zongyuan Ge and Rahil Garnavi. 2017. Generative OpenMax for Multi-Class Open Set Classification. In *Proceedings of the British Machine Vision Conference (BMVC)*, Gabriel Brostow Tae-Kyun Kim, Stefanos Zafeiriou and Krystian Mikolajczyk (Eds.). BMVA Press, Article 42, 12 pages. https://doi.org/10.5244/C.31.42