Generating Relevant Counter-Examples from a Positive Unlabeled Dataset for Image Classification

Florent Chiaroni, Ghazaleh Khodabandelou, Mohamed-Cherif Rahal, Nicolas Hueber, Frédéric Dufaux

Counter-Examples Generation from a Positive Unlabeled Image Dataset

Florent Chiaroni[a,c,*], Ghazaleh Khodabandelou[b], Mohamed-Cherif Rahal[a], Nicolas Hueber[d], Frederic Dufaux[c]

[a] *VEDECOM Institute, Department of delegated driving (VEH08), Perception team, 23 bis Allee des Marronniers, 78000, Versailles, France, {florent.chiaroni, mohamed.rahal}@vedecom.fr*

[b] *University of Paris-Est, Laboratoire Images, Signaux et Systèmes Intelligents (LISSI), 120 Rue Paul Armangot, 94400, Vitry-sur-Seine, France ghazaleh.khodabandelou@u-pec.fr*

[c] *Université Paris-Saclay, CNRS, CentraleSupélec, Laboratoire des signaux et systèmes, 3 rue Joliot Curie, 91190, Gif-sur-Yvette, France, {florent.chiaroni, frederic.dufaux}@l2s.centralesupelec.fr*

[d] *French-German Research Institute of Saint-Louis (ISL), ELSI team, 5 Rue du General Cassagnou, 68300, Saint-Louis, France, nicolas.hueber@isl.eu*

Abstract

This paper considers the problem of positive unlabeled (PU) learning. In this context, we propose a two-stage GAN-based model. More specifically, the main contribution is to incorporate a biased PU risk within the standard GAN discriminator loss function. In this manner, the discriminator is constrained to steer the generator to converge towards the unlabeled samples distribution while diverging from the positive samples distribution. Consequently, the proposed model, referred to as D-GAN, exclusively learns the counter-examples distribution without prior knowledge. Experimental results on simple and complex image datasets demonstrate that our approach outperforms state-of-the-art PU methods without prior by overcoming issues such as sensitivity to prior knowledge or first-stage overfitting.

Keywords: Generative Adversarial Networks (GANs), generative models, semi-supervised learning, partially supervised learning, deep learning

1. Introduction

Nowadays, the number of available labeled datasets dedicated to perception applications such as image classification [Russakovsky et al., 2015] and semantic scene understanding [Cordts et al.,

*Corresponding author: Florent Chiaroni (permanent address: f.chiaroni@net.estia.fr)

2016] has considerably augmented. However, when learning methods trained on these datasets are applied on real data, their performances are likely to deteriorate. Consequently, it is necessary to use a dataset specialized for the given target application. It turns out that it can be easy to get unlabeled data in some applications domains such as autonomous driving. Positive Unlabeled (PU) learning, also called *partially supervised classification* [Liu et al., 2002], enables to use these unlabeled data in combination with labeled samples of our class of interest: the positive class. The interest is that unlabeled data can contain relevant counter-examples, also called negative examples[1]. The difficulty is that unlabeled data can also contain a fraction π_P of unlabeled positive examples, as illustrated in Fig. 1. Sansone et al. [2018] enumerates several learning problems which can be addressed in this way such as the challenging information retrieval task.

(a) (b) (c)

Figure 1: Illustration of MNIST minibatch training examples for Positive Negative (PN) and Positive Unlabeled (PU) learning. If we consider the class *5* as our class of interest, the positive class, and the class *3* as the negative class representing counter-examples, then (a), (b) and (c) respectively represent positive, negative and unlabeled minibatch examples. While a PN training requires labeled Positive and labeled Negative examples, a PU training only requires labeled Positive and Unlabeled examples. The unlabeled minibatch in (c) is composed by a fraction π_P of positive examples, and a fraction $(1 - \pi_P)$ of negative examples. π_P is considered as a prior knowledge which is potentially unknown in real-world applications.

Several PU learning methods exist. While some of them are adapted to time series [de Carvalho Pagliosa and de Mello, 2018] or text classification, for instance by using non-negative matrix factorization [Li et al., 2016], we focus in this work on those that are applied to image classification

[1]We use the term *example* to design a single instance (i.e. item, observation) included in a sample set of data following a given distribution.

using deep neural networks. They are generally classified into two categories. The former is censoring PU learning, formalized by Elkan and Noto [2008] and recently improved by Northcutt et al. [2017]. The latter is case-control PU learning, introduced by Ward et al. [2009], formalized by du Plessis et al. [2014], and then consecutively improved by Du Plessis et al. [2015] and Kiryo et al. [2017] to reduce the training computational cost and alleviate the overfitting issue. In the context of the proposed approach, we focus our attention in this work on the recently presented GAN-based PU approaches. Thus we classify PU learning approaches into the two following groups suggested by Kiryo et al. [2017]: one-stage and two-stage PU methods.

One-stage PU methods such as the unbiased PU method (uPU) [Du Plessis et al., 2015] and the non-negative PU method (nnPU) [Kiryo et al., 2017] consist of training a classifier using an unbiased risk directly on the PU dataset. These methods have the advantage to need only one training of the classifier. However, they require dataset prior knowledge and consequently uPU and nnPU need to be combined with an approach estimating the prior knowledge [Christoffel et al., 2016]. Consequently, they are critically sensitive to slight prior variations per minibatch, as shown experimentally in Section 4.3.1.

Two-stage PU methods prepare during the first stage a Positive Negative (PN) dataset. Some noisy labeled learning strategies consist of detecting automatically the most plausible mislabeled examples [Ekambaram et al., 2016] through an iterative process in order to accelerate the manual PN labeling. Some other methods can prepare automatically a PN dataset without human supervision from end-to-end. For instance, Rank Pruning method (RP) [Northcutt et al., 2017] firstly estimates the prior such that it can select only the examples considered as the most confident, in order to substitute the unlabeled samples for the second-stage training of the classifier. RP achieves two-stage state-of-the-art performances without prior knowledge. However, it can only exploit a sub part of the PU dataset during training. This can curb its prediction performances on complex datasets like CIFAR-10. Recently, a new subcategory of two-stage PU methods has appeared: GAN-based PU methods. They address the PU learning challenge by producing, thanks to an adversarial training [Goodfellow et al., 2014], generated samples from a PU dataset during the first step. Then, they are used to train a standard Positive Negative (PN) classifier during the second

step.

We discuss in more details these PU methods, namely uPU [Du Plessis et al., 2015], nnPU [Kiryo et al., 2017], RP [Northcutt et al., 2017], GenPU [Hou et al., 2018] and PGAN [Chiaroni et al., 2018] in the related work Section 2.

We can nonetheless already make the following remarks, motivating the design of the proposed approach. Unbiased methods [Kiryo et al., 2017], [Du Plessis et al., 2015], and GenPU [Hou et al., 2018] are by definition sensitive to the prior knowledge in order to deal with a PU dataset. Conversely, whereas the two-stage censoring methods, such as RP [Northcutt et al., 2017], do not require prior information, they suffer from generalization and unsteadiness problems due to their selective process. The PGAN method is the first that does not need prior knowledge nor a selective process, thus preserving a low sensitivity to prior knowledge combined with a training stability. However, as mentioned in the PGAN work, it inherently suffers from first-stage overfitting. Based on these considerations, we propose in this work a novel GAN-based model, referred to as Divergent-GAN (D-GAN), to overcome the latter issue while preserving the PGAN advantages. To the best of our knowledge, we are the first to propose a GAN-based method to capture exclusively the unlabeled negative samples distribution from a PU dataset without prior knowledge. More specifically, our contribution is the following:

- We propose to incorporate a biased PU learning loss function inside the original GAN [Goodfellow et al., 2014] discriminator loss function. The intuition behind it is to have the generative model solving the PU learning problem formulated in the discriminator loss function. In this way, the generator learns the distribution of the examples which are both unlabeled and not positive, namely the negative ones included in the unlabeled dataset.

Consequently, the proposed D-GAN framework compares favorably with PU learning state-of-the-art performances on simple MNIST [LeCun et al., 1998] and complex CIFAR-10 [Krizhevsky and Hinton, 2009] image datasets. The proposed framework code is available [2].

The remaining of this paper is structured as follow. Section 2 presents previous PU learning

[2]D-GAN code for RGB images of 64×64 pixels is provided in supplementary material.

approaches. Section 3 describes the proposed method. Section 4 presents the corresponding experimental results. Finally, in Section 5, we draw conclusions and discuss perspectives.

2. Related work

The PU learning problem consists of trying to distinguish positive samples from negative ones by using a PU dataset. Let $X \in \mathbb{R}^m$ be the input random variable and $Y \in \{0, 1\}$ its associated label. X can be a positive X_P, negative X_N or unlabeled X_U sample which respectively follow the distributions $p_P = p(X|Y = 0)$, $p_N = p(X|Y = 1)$ and $p_U = (1 - \pi_P) \cdot p_N + \pi_P \cdot p_P$. The unknown prior $\pi_P \in (0, 1)$ represents the fraction of unlabeled positive examples included in the unlabeled dataset.

Previous works on PU learning [Denis, 1998] consider the entire distribution of the unlabeled examples as negative. In this way, all the negative examples, present in the unlabeled dataset, are always considered as negative. However, concerning the positive examples, it implies associating two contradictory labels to the distribution of positive examples in unknown proportions depending on the π_P value. Thus, training directly a classifier with positive and unlabeled data provokes a bias in the training estimator, which is not present during a standard positive negative training. This bias can limit prediction performances of the learning model.

Several strategies have been proposed to solve this drawback such as unbiased methods [du Plessis et al., 2014], [Du Plessis et al., 2015], [Kiryo et al., 2017], pruning method [Northcutt et al., 2017], and more recently GAN based methods [Hou et al., 2018], [Chiaroni et al., 2018]. However, those strategies still present some issues including prior knowledge sensitivity, training unsteadiness, or overfitting problems.

We present in this section different state-of-the-art methods and their respective drawbacks that we aim at overcoming with the proposed GAN-based PU framework.

2.1. Unbiased methods

In order to palliate a biased training, the authors of unbiased techniques [du Plessis et al., 2014], [Du Plessis et al., 2015], [Kiryo et al., 2017] suggest to avoid the estimator bias by adding some terms in the training loss function. Then, the classifier behaves as if it is trained with a positive

negative dataset. The authors firstly used a non convex loss function [du Plessis et al., 2014], which then has been adapted for convex loss functions [Du Plessis et al., 2015] in order to reduce the computational burden. Subsequently, it was proposed to overcome the training overfitting by adding a binary condition (an "if" condition) in the training loss function [Kiryo et al., 2017].

These methods exploit the prior π_P in the empirical training loss function. However, we observe that the empirical prior value $\hat{\pi}_P$ per batch of small size (minibatch) is slightly different to the global dataset prior π_P, as its standard deviation depends on the minibatch size, such that:

$$\hat{\pi}_P = \pi_P + \alpha, \tag{1}$$

with $\alpha \sim p_\alpha(m)$, where p_α is the probability distribution of the noise α depending on the minibatch size m, as shown in Figure 2. We observe that the worst case scenario is when π_P is close to the value 0.5, combined with a small batch size. The cases where π_P is higher than 0.5 behave symmetrically to the cases where π_P is smaller than 0.5.

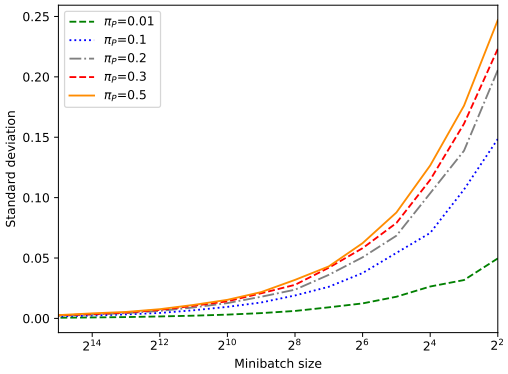


Figure 2: Standard deviation per minibatch of the global prior π_P as a function of the minibatch size, for a given uniformly mixed dataset composed by 60000 examples. More specifically, we have randomly extracted minibatches from a simulated unlabeled MNIST dataset depending on π_P. The ground truth for labels present in each simulated unlabeled minibatch has been preserved, such that we have been able to estimate how the prior knowledge of a given unlabeled minibatch can vary in terms of standard deviation, depending on the minibatch size.

In our case, we want to train a deep learning model using the stochastic gradient descent (SGD) optimization technique, which is known to be relevant with batches of small size. So the theoretical formulation of unbiased techniques cannot be maintained using SGD with small batch sizes. We will show empirically that in practice, unbiased techniques are highly sensitive to the minibatch

size in terms of prediction performances, as they are theoretically sensitive to the prior π_P.

It turns out that it is possible to avoid this limitation with two-stage approaches.

2.2. Two-stage approaches

Two-stage approaches mainly consist of preparing during the first-stage a positive negative (PN) training dataset which will then be used to directly train a standard classifier during the second stage. One interest of those approaches is that they are not sensitive to the prior knowledge variation. Consequently, they are compatible with the use of minibatches, and thus are suitable when applying SGD optimization.

2.2.1. Pruning approach

Rank Pruning (RP) method [Northcutt et al., 2017] is a two-stage technique. It first estimates the prior π_P and exploits it to prune the dataset in order to capture only a subset corresponding to the most confident positive and negative samples. Then, during the second stage it considers this subset as a cleanly labeled positive negative dataset to train a classifier. While not requiring prior knowledge in input, RP achieves state-of-the-art results for information retrieval in the One-vs-Rest task on simple datasets such as MNIST. However, by using a pruning strategy, RP can miss some relevant training examples not included in the selected subset of training. As a consequence, this can limit its generalization, as will be shown experimentally in Table 2, where RP is shown to be relatively unstable when compared to GAN-based approaches in terms of prediction performances. Using only a training subset is also a weakness on complex datasets like CIFAR-10, where a large training dataset is preferable to obtain better results.

Some approaches have been recently proposed by exploiting GANs benefits, achieving high prediction scores over the same PU learning tasks.

2.2.2. GAN-based approaches

GAN-based PU approaches represent a recent subcategory of two-stage PU methods, as proposed in GenPU [Hou et al., 2018] and PGAN [Chiaroni et al., 2018]. The interest of using GANs is twofold. First, GANs enable relevant data augmentation, as proposed for instance in [Zhu et al., 2018] for dealing with unbalanced label distributions for emotion classification. Second, it allows for

the use of high-level feature metrics to evaluate generated samples quality, thanks to the adversarial training. This can make capturing a target distribution easier.

In this PU learning context, the generated samples replace the unlabeled ones by learning on the latter as PGAN [Chiaroni et al., 2018], or on both unlabeled and positive labeled ones as GenPU [Hou et al., 2018]. Both methods exploit GANs benefits, but the functioning are different and they are not suitable under the same datasets conditions.

GenPU [Hou et al., 2018] is based on the original GAN convergence [Goodfellow et al., 2014], such that: $\pi_P \cdot p_{G_P} + (1 - \pi_P) \cdot p_{G_N} \rightarrow p_U$, with p_{G_P} the distribution of positive samples generated by the generator G_P, p_{G_N} the distribution of the negative samples generated by the generator G_N, and p_U the distribution of real unlabeled samples. In practice, GenPU is an interesting PU method on simple datasets with few positive labeled samples, and it generates relevant counter-examples. However, its adversarial training of five learning models instead of two as in the original GAN framework [Goodfellow et al., 2014] to address *standard* PU learning challenge[3] is more computational demanding and not necessary to generate relevant counter-examples. Moreover, using five models amplifies the mode collapse issue, and the corresponding training optimization functions need three additional sets of hyper-parameters combined with prior knowledge. This is impractical in the context of real applications where hyper-parameters may have to be tuned on limited computational resources to adapt the model for a given application dataset.

PGAN [Chiaroni et al., 2018] is trained to converge towards the unlabeled dataset distribution during the first step. During the second step, it exploits GANs imperfections for capturing the unlabeled distribution, such that the generated distribution at the adversarial equilibrium is still separable from the unlabeled samples distribution by a classifier. It presents a relatively steadier behaviour and better prediction performances than the two-stage baseline RP method on the complex RGB image dataset CIFAR-10 without prior knowledge. However, it is less suitable for relatively simpler datasets like MNIST. The problem is that the generated samples are all considered as negative samples by the classifier. But this is possible only if the generated samples

[3]We use the term *standard* to refer to the case where we have enough positive labeled examples (at least 100), such that the difficulty is mainly the ability to exploit counter-examples included in the unlabeled set.

distribution converges close enough towards the unlabeled samples distribution, while not matching it. If the PGAN first-stage performs as expected theoretically by [Goodfellow et al., 2014], then the PGAN classification second stage falls back into the initial PU learning problem.

Our proposed approach, presented in Sec. 3, overcomes previously enumerated PU methods shortcomings, to address the *standard* PU learning task, as summarized in Table 1.

Methods	D-GAN (proposed)	PGAN [Chiaroni et al., 2018]	GenPU [Hou et al., 2018]	RP [Northcutt et al., 2017]	nnPU [Kiryo et al., 2017]
No need of prior knowledge	√	√		√	
No first-stage overfitting	√		√	√	√
Generalizable over complex datasets	√	√			
Able to generate relevant counter-examples	√		√		
Training stability using SGD	√	√		√	
Original GAN architecture	√	√			
Code availability	√			√	√

Table 1: Summary of presented state-of-the-art methods advantages and drawbacks compared to the proposed D-GAN approach. A void cell means that the mentioned criterion is not applicable with the corresponding method.

3. Proposed Approach

In this section, we first briefly recall the main reasoning which motivates our research work. Next, we discuss some features of a biased PU risk. We then propose to incorporate this risk into a generic GAN framework in order to guide the generator convergence towards the negative samples distribution, denoted as p_N, included inside the unlabeled dataset distribution, denoted as p_U.

3.1. Motivation

In PU learning, if a classifier associates a given expected label value with positive examples, and in parallel associates a second distinct label value with unlabeled examples, then it is proven that the negative non-labeled examples are exclusively associated with the label of non-labeled examples [Denis, 1998]. Concerning GANs, it has been shown that the discriminator learning task influences directly the adversarial generator behaviour [Mao et al., 2017]. Based on these considerations, this work aims at incorporating a biased PU risk inside the traditional GAN discriminator cost function. This compels the discriminator D, to separate negative from positive distributions, which in turn guides the generator G, to exclusively learn the unlabeled counter-examples distribution from a PU dataset. As a matter of the fact, the proposed method is novel in the way it exclusively generates

relevant counter-examples without prior knowledge information, while preserving a standard GAN architecture.

Thereafter, we present the biased PU risk that we incorporate in the proposed GAN PU discriminator training loss function.

3.2. Biased PU risk

In what follows, we first explain the expected PU functionality to be incorporated into the GAN discriminator loss function. **Biased PU risk setting:** Let $D : \mathbb{R}^m \to [0, 1]$ be the decision function which is, later on, considered as the discriminator D, of the proposed framework network. We have $l(\hat{y}, y)$ such that $l : [0, 1] \times [0, 1] \to \mathbb{R}$ is the arbitrary cost function with the predicted output $\hat{y}$ of D for a given example and the corresponding label y as input. D is trained with a PU risk R_{PU} to predict the label value 1 for the unlabeled examples, and the label value 0 for the positive labeled ones such that:

$$R_{PU}(D) = \mathbb{E}_{x_U \sim \boldsymbol{p_U}}[l(D(x_U), 1)] + \mathbb{E}_{x_P \sim p_P}[l(D(x_P), 0)]. \tag{2}$$

Given the composition of the distribution p_U, we develop:

$$R_{PU}(D) = (1 - \pi_P) \cdot \mathbb{E}_{x_N \sim p_N}[l(D(x_N), 1)] + \pi_P \cdot \mathbb{E}_{x_P \sim p_P}[l(D(x_P), 1)] + \mathbb{E}_{x_P \sim p_P}[l(D(x_P), 0)]. \tag{3}$$

Counter-examples are correctly labeled: Decomposed in this way, the negative examples included in the unlabeled dataset are associated exclusively to the label value 1 for any π_P value, such that the negative training examples are all correctly labeled. When there is no overfitting on training positive examples, then one can assume that labeled and unlabeled positive examples follow the same distribution p_P, as mentioned in [Kiryo et al., 2017]. Since expectations are linear, p_P is associated to both contradictory labels 0 and 1 as below:

$$R_{PU}(D) = \mathbb{E}_{x_N \sim p_N}[(1 - \pi_P) l(D(x_N), 1)] + \mathbb{E}_{x_P \sim p_P}[\pi_P l(D(x_P), 1) + l(D(x_P), 0)]. \tag{4}$$

Positive samples distribution p_P is shifted away from the counter-examples dis-

tribution p_N: When defining the cost-function l as the binary cross-entropy H (Eq. 5) such that $l = H$, then we can demonstrate that the second term in the Equation 4 is equivalent to associating the positive distribution p_P with a unified biased intermediate label value δ. The binary cross-entropy H is defined as:

$$H(D(X), Y) = -Y log(D(X)) - (1 - Y) log(1 - D(X)), \tag{5}$$

where Y is the label value associated with the input X of D. If $l = H$, then concerning the second term of the Equation 4, we can demonstrate that:

$$\begin{aligned}
\pi_P H(D(x_P), 1) + 1H(D(x_P), 0) =& - \pi_P log(D(x_P)) - 1log(1 - D(x_P)) \\
=& - \pi_P log(D(x_P)) - (1 + \pi_P - \pi_P) log(1 - D(x_P)) \\
=& (1 + \pi_P) \cdot \left[- \frac{\pi_P}{1 + \pi_P} log(D(x_P)) - (1 - \frac{\pi_P}{1 + \pi_P}) log(1 - D(x_P)) \right] \\
=& (1 + \pi_P) \cdot H\big(D(x_P), \frac{\pi_P}{1 + \pi_P}\big) \\
=& (1 + \pi_P) \cdot H(D(x_P), \delta),
\end{aligned} \tag{6}$$

with $\delta = \pi_P / (1 + \pi_P)$. Consequently, the PU risk becomes:

$$R_{PU}(D) = \mathbb{E}_{x_N \sim p_N}[(1 - \pi_P) H(D(x_N), 1)] + \mathbb{E}_{x_P \sim p_P}[(1 + \pi_P) H(D(x_P), \delta)]. \tag{7}$$

Such a PU risk has been previously called biased or constrained in the literature [Liu et al., 2002]. The identity between Equations 4 and 7 makes it possible to estimate the restricted interval of possible values for δ without using prior such that if $\pi_P \in (0, 1)$ then:

$$0 < \pi_P < 1 \Leftrightarrow 0 < \delta < \frac{1}{1 + 1}. \tag{8}$$

In other words, $\delta \in (0, 1/2)$. This confirms that for any π_P value between 0 and 1, labeled and unlabeled positive examples are associated with a label value δ comprised between 0 and 1/2. Therefore, when training D with the risk R_{PU}, the D prediction related to the unlabeled positive

examples is shifted away from the label value 1. From D prediction output point of view, this risk makes the positive distribution p_P *diverging* from the negative distribution p_N. **Thus, D is trained to predict the label value 1 exclusively for the counter-examples.**

In this way, we have demonstrated in this section that we can consider the discriminator PU training loss function R_{PU} as a PN training loss function, referred to as R_{PN}, by replacing the two opposite labels 0 and 1 associated to positive samples distribution p_P by an intermediate label value δ depending on π_P, such that we obtain:

$$R_{PU}(D) = R_{PN}(D), \tag{9}$$

with:

$$\begin{cases} R_{PU}(D) = & \mathbb{E}_{x_U \sim \boldsymbol{p_U}}[H(D(x_U), \mathbf{1})] + \mathbb{E}_{x_P \sim \boldsymbol{p_P}}[H(D(x_P), \mathbf{0})], \\ R_{PN}(D) = & \mathbb{E}_{x_N \sim \boldsymbol{p_N}}[(1-\pi_P)H(D(x_N), \mathbf{1})] + \mathbb{E}_{x_P \sim \boldsymbol{p_P}}[(1+\pi_P)H(D(x_P), \boldsymbol{\delta})]. \end{cases} \tag{10}$$

3.3. Proposed generative model

The insight in the proposed D-GAN model can be expressed as follows: D addresses to G the riddle: *Show me what IS unlabeled AND NOT positive.* It turns out that negative examples included in the unlabeled dataset are both unlabeled and not positive. Consequently, G addresses this riddle by learning to show the negative samples distribution to D.

GAN background: We first give a short recall of the original GAN discriminator. It is trained to distinguish real unlabeled samples distribution p_U from generated samples distribution p_G with the loss function $L_{D_{GAN}}$ defined as:

$$L_{D_{GAN}}(G, D) = \mathbb{E}_{x_U \sim p_U}[-logD(x_U)] + \mathbb{E}_{z \sim p_z}[-log(1 - D(G(z)))], \tag{11}$$

where z stands for the input random vector of the generative model G such that $G(z)$ is a generated sample. z follows a uniform or normal distribution. It turns out that the binary cross-entropy formulation (Eq. 5) implies $H(D(X), 1) = -log(D(X))$ and $H(D(X), 0) = -log(1 - D(X))$.

Consequently, $L_{D_{GAN}}$ can be expressed as follows:

$$L_{D_{GAN}}(G, D) = \mathbb{E}_{x_U \sim p_U}[H(D(x_U), 1)] + \mathbb{E}_{z \sim p_z}[H(D(G(z)), 0)]. \tag{12}$$

Towards a GAN biased discriminator loss function: The proposed approach aims at training G to learn the negative samples distribution p_N instead of learning the distribution p_U. This replaces the standard GAN task *"Show me what is unlabeled "*, by the task *"Show me what is both unlabeled and not positive"*. We now propose to incorporate the benefits of a biased PU risk (Eq. 2) into the original GAN discriminator loss function (Eq. 11). To this end, we define the D-GAN discriminator loss function L_D by adding the term $\mathbb{E}_{x_P \sim p_P}[H(D(x_P), 0)]$ to $L_{D_{GAN}}$. Consequently, in the proposed D-GAN framework, the training discriminator loss function L_D of D becomes:

$$L_D(G, D) = L_{D_{GAN}}(G, D) + \mathbb{E}_{x_P \sim p_P}[H(D(x_P), 0)]. \tag{13}$$

If we develop the term $L_{D_{GAN}}$, we then obtain:

$$\begin{aligned} L_D(G, D) =& \mathbb{E}_{x_U \sim p_U}[H(D(x_U), 1)] + \mathbb{E}_{z \sim p_z}[H(D(G(z)), 0)] + \mathbb{E}_{x_P \sim p_P}[H(D(x_P), 0)] \\ =& R_{PU}(D) + \mathbb{E}_{z \sim p_z}[H(D(G(z)), 0)]. \end{aligned} \tag{14}$$

In other words, the R_{PU} risk (Eq. 2) is incorporated inside the D-GAN discriminator loss function. To this extent, D can be trained to only consider the counter-examples as the most real examples by exclusively associating to them the label value 1. This can be considered as applying a constrained optimization.

The generator generates the counter-examples distribution: In contrast, the role of G during the adversarial training is to generate samples considered by D as 1. As suggested by [Goodfellow et al., 2014], the training loss function L_G of G is such that:

$$\begin{aligned} L_G(G, D) &= \mathbb{E}_{z \sim p_z}[-log(D(G(z)))] \\ &= \mathbb{E}_{z \sim p_z}[H(D(G(z)), 1)]. \end{aligned} \tag{15}$$

Algorithm 1 Minibatch SGD training of the D-GAN

GAN training (1^{st} step)

for number of training iterations **do**

Sample minibatch of m noise samples $\{z^{(1)}, ..., z^{(m)}\}$ from noise prior p_z.

Sample minibatch of m unlabeled examples $\{x_U^{(1)}, ..., x_U^{(m)}\}$ from data distribution p_U.

Sample minibatch of m positive labeled examples $\{x_P^{(1)}, ..., x_P^{(m)}\}$ from data distribution p_P.

Update D by descending its stochastic gradient:

$$\nabla_{\theta_D} \frac{1}{m} \sum_{i=0}^{m} \Big[-logD(x_U^{(i)}) - log\big[1 - D(G(z^{(i)}))\big] - log\big[1 - D(x_p^{(i)})\big] \Big]$$

Sample minibatch of m noise samples $\{z^{(1)}, ..., z^{(m)}\}$ from noise prior p_z.

Update G by descending its stochastic gradient:

$$\nabla_{\theta_G} \frac{1}{m} \sum_{i=0}^{m} -log\big[D(G(z^{(i)}))\big]$$

end for

Classifier training (2^{nd} step):

for number of training iterations **do**

Sample minibatch of m positive labeled examples $\{x_P^{(1)}, ..., x_P^{(m)}\}$ from data distribution p_P.

Sample minibatch of m noise samples $\{z^{(1)}, ..., z^{(m)}\}$ from data distribution p_z.

Update C by descending its stochastic gradient:

$$\nabla_{\theta_C} \frac{1}{2 \cdot m} \sum_{i=1}^{m} \Big[l(C(x_P^{(i)}), 1) + l(C(G(z^{(i)})), 0) \Big]$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We use Adam in our experiments.

As previously discussed, we recall that D exclusively considers the negative examples as 1 thanks to the R_{PU} risk. Thus, if D trainable weights are fixed in the proposed framework, then we propose to reinterpret in L_G the label value 1 as $D(x_N)$, as follows:

$$\begin{aligned} L_G(G, D) &= \mathbb{E}_{z \sim p_z, x_N \sim p_N}[H(D(G(z)), D(x_N))] \\ &= \mathbb{E}_{z \sim p_z, x_N \sim p_N}[-D(x_N) log(D(G(z)))], \end{aligned} \tag{16}$$

such that the distance between the generated samples distribution and p_N is minimized. Consequently, this justifies the convergence of G in the proposed D-GAN framework towards the negative samples distribution p_N, for any $\pi_P \in (0, 1)$.

Implementation: The corresponding implementation algorithm 1 of the proposed first-stage D-GAN approach enables to adversarially train D and G to respectively minimize loss functions L_D and L_G.

Second-stage: Positive-Generative learning. Once the D-GAN training is completed, the second step can be carried out. It consists of training a classifier C to distinguish fake generated examples $x_{FN} = G(z)$, which are ideally equivalent to the real negative samples, from real positive labeled samples as illustrated in Figure 3.

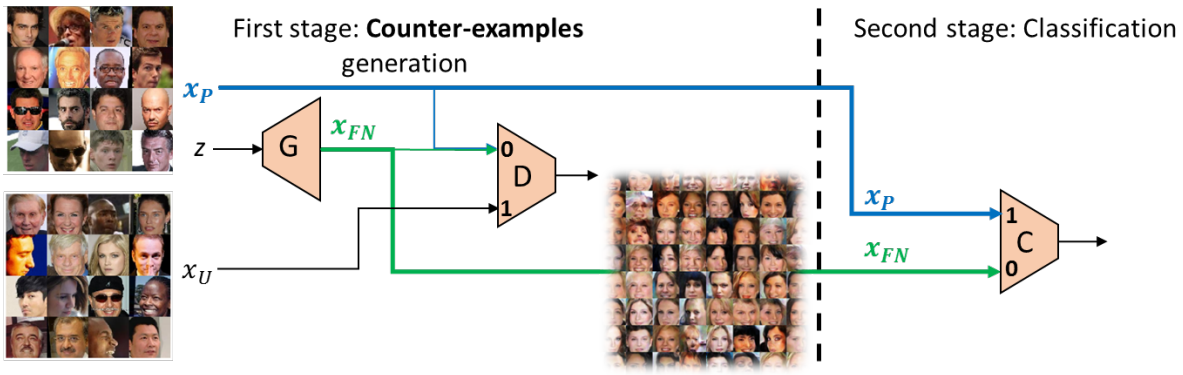


Figure 3: Proposed GAN-based PU approach, x_{FN} represents the generated samples which are similar to real negative samples x_N, G is the generative model, D is the discriminator, C is the classifier used to perform the binary Positive-Negative (PN) classification.

In practice, the worst-case scenario is when D overfits the positive examples during the adversarial training. Another pitfall is when D cannot encode the complexity of the boundary between positive and negative examples included in the unlabeled dataset. In such cases, D will consider some unlabeled positive examples as negative ones. As a consequence, this implies that G will also generate some examples following a subset of the positive samples distribution. Thus, the D-GAN will tend to behave as the PGAN [Chiaroni et al., 2018], which seems to be the best solution in this situation.

The next section presents experimental results demonstrating the effectiveness of the proposed approach.

4. Experimental Results

In this section, we assess the performance of the proposed approach. We first experimentally validate the expected discriminator prediction behaviour when it is applied on a positive unlabeled dataset (Sec. 4.2.1), and study the impact of regularization (Sec. 4.2.2). Then, we show the ability of the generator to generate counter-examples for different types of PU datasets, including two-dimensional points and natural RGB images (Sec. 4.2.3). Finally, we evaluate the proposed

model prediction robustness and compare it with state-of-the-art PU learning methods in terms of prior noise (Sec. 4.3.1) and first-stage overfitting (Sec. 4.3.2).

4.1. Settings

We detail in this section the settings of the experiments. We have adapted the first-stage discriminator and generator architectures of the proposed GAN based PU framework depending on the dataset on which they are applied, as follows:

- **2D point dataset**: In order to deal with 2D point datasets, we have implemented the original GAN [Goodfellow et al., 2014] architecture composed of fully connected layers (FullyConnected). The generator and discriminator architectures are summarized in Figure 4.

- **MNIST** [LeCun et al., 1998]: In order to deal with grayscale images of dimension 28*28 pixels from the MNIST dataset, we use a deep convolutional GAN architecture (DCGAN) [4] such that the generator contains transposed convolutional (DeConv2D) top layers, and the discriminator contains convolutional (Conv2D) bottom layers as illustrated in Figures 5 (a) and (b). The second-stage classifier[5] presented in Figure 5 (c) also contains convolutional bottom layers.

- **CIFAR-10** [Krizhevsky and Hinton, 2009]: In order to deal with RGB images of size 32*32 pixels from the CIFAR-10 dataset, we use the same DCGAN and classifier architectures presented in Figures 5 (a), (b) and (c). We only adapt the feature maps size depending on the width (w), the height (h), and the number of channels (ch) of input RGB images.

- **celebA** [Liu et al., 2015]: In order to deal with RGB images of size 64*64 from the celebA dataset, we use a deeper convolutional GAN architecture[6] presented in Figure 6.

Concerning the PU dataset initialization from a standard PN dataset, in all the experiments, except the ones in Sec. 4.3.1, we use the methodology proposed by [Chiaroni et al., 2018]. More

[4]The architecture is based on this code: `https://github.com/hwalsuklee/tensorflow-generative-model-collections`.

[5]The architecture is based on this code: `https://github.com/tensorflow/tensorflow/blob/master/tensorflow/examples/tutorials/mnist/mnist_softmax.py`.

[6]The architecture is based on this code: `https://github.com/guojunq/lsgan`.

Input: $z \in \mathbb{R}^8 \sim \mathcal{N}(0,1)$
FullyConnected (128) ELU
FullyConnected (2) Sigmoid
Output: 2D point $x \in \mathbb{R}^2$

(a) Generator

Input: Image $x \in \mathbb{R}^2$
FullyConnected (128) ELU
FullyConnected (1) Sigmoid
Output: Scalar $y \in (0,1)$

(b) Discriminator

Figure 4: Fully connected GAN model architecture used for two dimensional points datasets. Minibatch size 64, optimizer Adam. We trained the model during 100 epochs on 2D point datasets.

Input: $z \in \mathbb{R}^{100} \sim \mathcal{N}(0,1)$
FullyConnected (1024) BN ReLU
FullyConnected ($128 \times \frac{h}{4} \times \frac{w}{4}$) BN ReLU
DeConv2D (64 filters 4×4) BN ReLU
DeConv2D (ch filters 4×4) Sigmoid
Output: Image $x \in \mathbb{R}^{h \times w \times ch}$

(a) Generator

Input: Image $x \in \mathbb{R}^{h \times w \times ch}$
Conv2D (64 filters 4×4) SN LeakyReLU
Conv2D (128 filters 4×4) SN LeakyReLU
FullyConnected (1024) SN LeakyReLU
Dropout (0.5) FullyConnected (1) Sigmoid
Output: Scalar $y \in (0,1)$

(b) Discriminator

Input: Image $x \in \mathbb{R}^{h \times w \times ch}$
Conv2D (32 filters 5×5) ReLU Maxpooling (2×2)
Conv2D (64 filters 5×5) ReLU Maxpooling (2×2)
FullyConnected (1024) ReLU
Dropout (0.5) FullyConnected (2) Softmax
Output: One hot vector $y \in (0,1)^2$

(b) Classifier

Figure 5: Convolutional GAN model architecture used for 28*28 grayscale MNIST and 32*32 RGB CIFAR-10 image datasets. For MNIST we set h=28, w=28, ch=1. For CIFAR-10 we set h=32, w=32, ch=3. Minibatch size: 64, optimizer: Adam, strides of 2×2 for the generator Deconv2D and the discriminator Conv2D layers, strides of 1×1 for the classifier Conv2D layers. We trained the model during 40 epochs and 1000 epochs respectively on MNIST and CIFAR-10 datasets.

specifically, we set $\rho = 0.5$ which is the fraction of positive labeled examples of the initial PN dataset that we unlabel such that they are included into the unlabeled dataset. Then, we set π_P which is the fraction that represents these unlabeled positive examples among the unlabeled dataset. This method is interesting for testing an approach depending on π_P, independently of the selected fraction $1 - \rho$ of positive labeled samples.

4.2. Qualitative analysis

We start by studying qualitatively whether the discriminator behaves as expected in practice. More precisely, we need to verify whether it exclusively associates the counter-examples distribution with the label value 1, and the positive samples distribution with an intermediate label value between 0 and 1/2.

Input: $z \in \mathbb{R}^{100} \sim \mathcal{N}(0,1)$
FullyConnected (512) BN ReLU
DeConv2D (256 filters 5×5) BN ReLU
DeConv2D (128 filters 5×5) BN ReLU
DeConv2D (64 filters 5×5) BN ReLU
DeConv2D (3 filters 5×5) tanh
Output: Image $x \in \mathbb{R}^{64 \times 64 \times 3}$

(a) Generator

Input: Image $x \in \mathbb{R}^{64 \times 64 \times 3}$
Conv2D (64 filters 5×5) SN LeakyReLU
Conv2D (128 filters 5×5) SN LeakyReLU
Conv2D (256 filters 5×5) SN LeakyReLU
Conv2D (512 filters 5×5) SN LeakyReLU
Dropout (0.5) FullyConnected (1) LeakyReLU
Output: Scalar $y \in (0,1)$

(b) Discriminator

Figure 6: Convolutional GAN model architecture used for 64*64 RGB images of celebA dataset. Minibatch size: 64, optimizer: Adam, 2D stride of 2×2. We trained the model during 100 epochs on the celebA dataset.

In Sec. 4.2.1, we start by showing the relation between the PU loss function and the proposed equivalent PN loss function including a biased label for positive examples, as mentioned in Sec. 3.2. Then, in Sec. 4.2.2, we investigate which regularization techniques enable to preserve the same behaviour on an image dataset such that the discriminator does not suffer from overfitting during the epoch training iterations.

4.2.1. Empirical Positive Unlabeled risk analysis

We have previously demonstrated (Eq. 6 and Eq. 7) that we can reformulate the discriminator PU training loss function R_{PU} (Eq. 2) into a PN training loss function R_{PN} (Eq. 10), by replacing the two opposite labels 0 and 1 associated to positive samples distribution p_P by an intermediate label value δ depending on π_P.

It turns out that we can verify the same relation empirically. As illustrated in Figure 7 with 2D point samples following gaussian distributions, if we train the discriminator D, with the multilayer perceptron structure presented in Figure 4 (b), using the PU loss function R_{PU}, then its output predictions for an unlabeled batch sample are partitioned in the vicinity of two different labels. The predictions for positive examples are centered around an intermediate label value corresponding to δ. Conversely, D output predictions for the negative examples are centered around the label value 1. In addition, we have also computed the approximated PN risk $\hat{R}_{PN}$ using labeled negative

and labeled positive samples, instead of labeled positive and unlabeled samples, for several δ values between 0 and 1/2. We can observe that, the global minimum of the PN approximated risk $\hat{R}_{PN}$ depending on δ, graphically corresponds to the global maximum of the density function corresponding to D output predictions for a positive set. This coincides with the equality presented in Equation 9.

To sum up, this experimentally illustrates that if D is trained with the R_{PU} loss function, then it should predict the label value 1 exclusively for the negative samples, which is the necessary condition to guide the generator during the adversarial training to learn exclusively the counter-examples distribution.

However, this behaviour is only possible if D does not overfit labeled and unlabeled positive samples. In other words, D should be able to discriminate unlabeled positive examples from the unlabeled negative ones. Therefore, in order to generalize the proposed GAN framework to image datasets, we compare in the next section some state-of-the-art regularization techniques commonly used in deep learning models, in order to select the most appropriate one.

4.2.2. Impact of regularizations on the discriminator

Nowadays, Batch Normalization (BN) [Ioffe and Szegedy, 2015] is considered as a one of the most relevant regularization techniques commonly used in deep neural networks architectures. Its utility for GANs training has been highlighted by [Radford et al., 2015] for the DCGAN architecture in order to stabilize the adversarial training. Other variants like the Wasserstein-GAN [Arjovsky et al., 2017] or the Loss-Sensitive GAN [Qi, 2017] confirmed its interest. As developed in [Ioffe and Szegedy, 2015], BN addresses issues like vanishing or exploding gradient problems, as well as the risk of getting stuck in a poor local minima, by reducing the internal covariate shift problem of the learning model. A higher learning rate can be used and it can significantly improve the training speed.

Multiple minibatch manipulation incompatibility. BN regularizes the model, in such a way that a training example (i.e. single instance) from a given minibatch sample is considered in conjunction with other examples of this minibatch sample. This is the consequence of estimating the mean and variance normalization parameters one time per minibatch, and then applying them

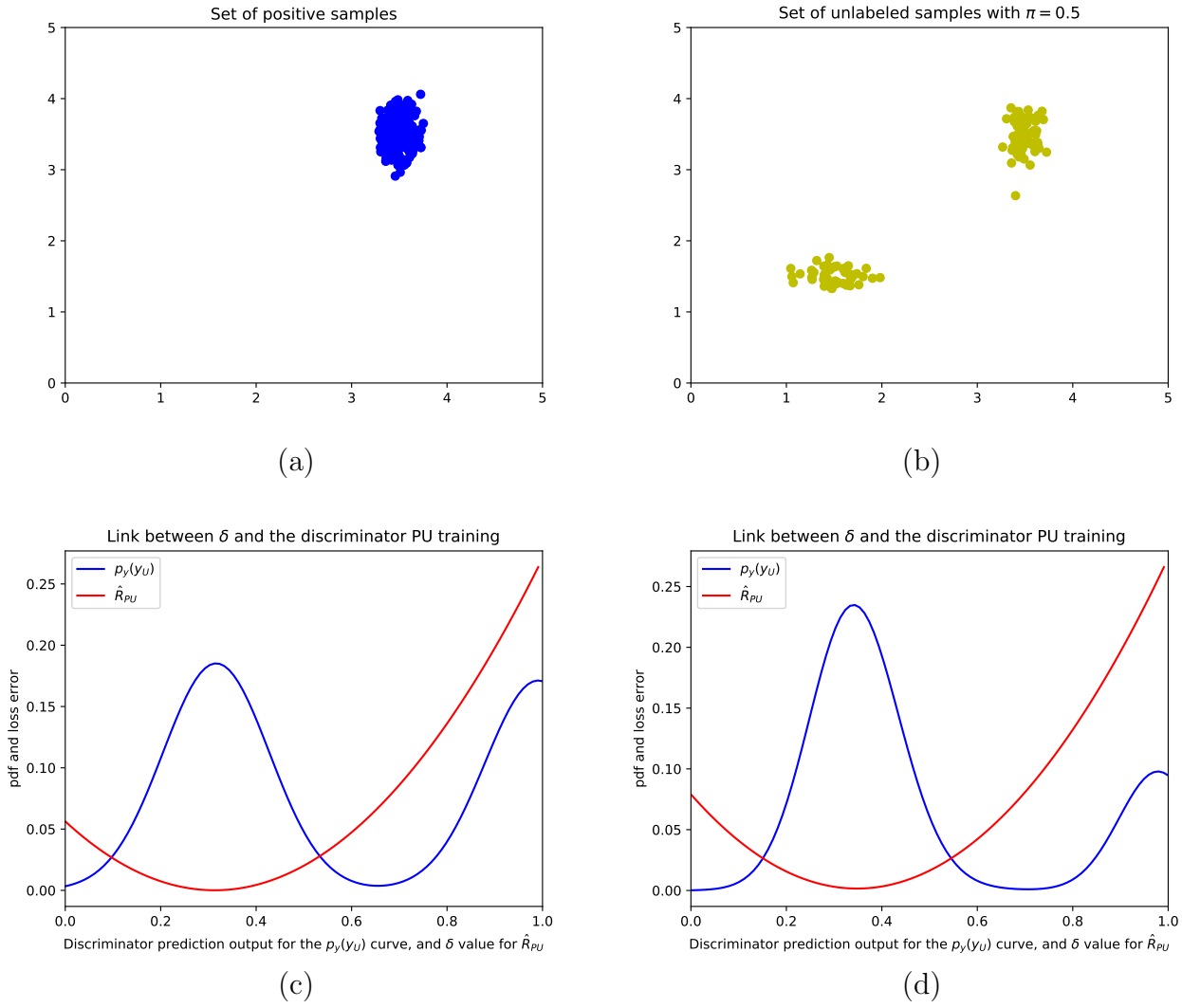


Figure 7: Link between the suggested PN loss function (Eq. 9) and the distribution of the discriminator output predictions for an input PU training minibatch. For this experiment, D is a multi-layer perceptron. D has been trained to distinguish a 2D gaussian distribution from another one by using the risk R_{PU} on a PU dataset. (a) Shows a set of 2D points considered as positive examples. (b) Shows a set of 2D points considered as unlabeled examples. Both curves in (c) and (d) have been normalized to get a better visualization. For (c), $p_Y(y_U)$ (in blue), with $y_U = D(x_U)$, represents the probability distribution of D predicted outputs for a minibatch of unlabeled samples, with $\pi_P = 0.5$. $\hat{R}_{PU}$ (in red) represents the PN risk computed depending on δ with the R_{PN} proposed in Eq. 10 on a minibatch of positive and negative labeled samples, once D is trained with R_{PU} risk (Eq. 2). (d) shows the same curves as in (c) but by giving in input a concatenation of an unlabeled minibatch with a positive labeled minibatch. Unlabeled positive and labeled positive samples provide a unified prediction output distribution.

on each example in the minibatch. When positive examples x_P and unlabeled examples x_U are not in the same training minibatch, as this is the case in our discriminator loss function, this does not enable to link labeled positive examples with the unlabeled positive ones. Consequently, this cannot produce a distance between positive and negative examples predictions. To counter

this problem, we could imagine to apply BN on a unified minibatch which contains a fraction of each distribution x_P, x_U and x_F. But the BN effect is greatly influenced by the content of the minibatch on which it is applied. Therefore, the fraction π_P of positive examples included in x_U will negatively impact the BN outcome.

Compatible normalization techniques: However, BN benefits in a more traditional training are not negligible. Hence, we propose to use an alternative technique in order to replace the BN role in the proposed GAN-based PU framework. **Spectral Normalization** (SN) [Miyato et al., 2018] is a recent competing technique for GANs training which can stabilize the training of D against input perturbations [Farnia et al., 2019] by performing a weight normalization. In this way, a training manipulating multiple types of minibatch distributions preserves SN effectiveness. For these reasons, we propose to apply SN instead of BN inside our discriminative model structure.

Dropout alleviates the positive overfitting problem: As mentioned in the previous section, we can only deduce Equation 4 if we consider that the positive samples distribution is the same for both labeled and unlabeled ones. In practice, this assumption holds in the case of a large dataset, such that this overfitting problem concerning the positive examples disappears. While some model averaging strategies such as bootstrap aggregating techniques have been previously combined with Support-Vector Machines (SVMs) in order to deal with PU learning [Mordelet and Vert, 2014], the dropout [Srivastava et al., 2014] generalization technique is also a solution concerning the deep neural networks. Consequently, in the context of the proposed D-GAN training, we propose to introduce dropout in the top fully connected layer of D. We enable it during D training steps, and conversely disable it during G training steps. This improves the evaluation of generated samples which is transmitted from D to G by back-propagation. Hence, dropout alleviates the positive examples overfitting during long D-GAN trainings. This insures to exclusively generate counter-examples.

We compare in Figure 8 the ability of D to distinguish positive from negative samples distributions included inside the unlabeled training dataset when D is trained on a PU image dataset without normalization and with BN and SN normalizations. We also consider the cases when they are combined with the dropout regularization. In this experiment, D is trained alone such that it

is not adversarially trained with G. This enables to better observe and anticipate the adversarial behaviour of D, and consequently the behaviour of G during the adversarial training.

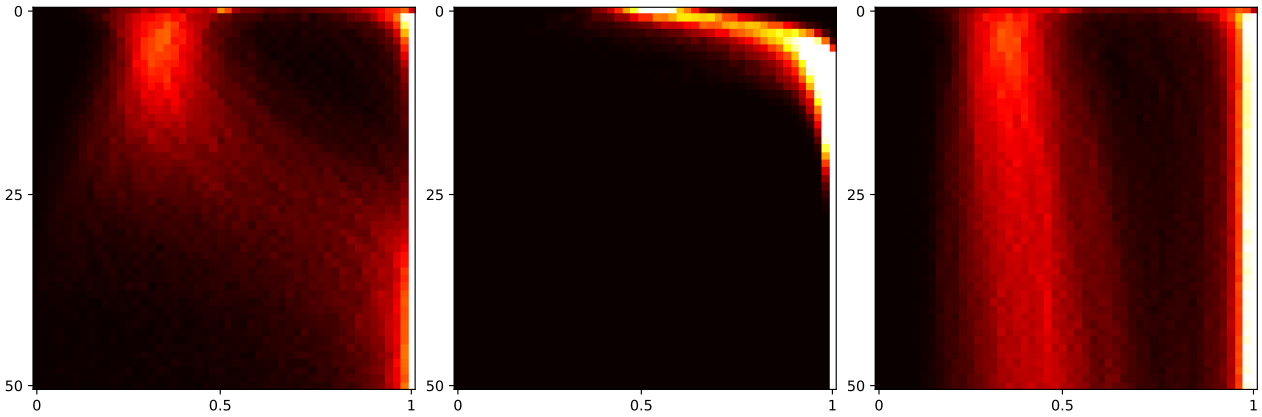
We show in Figure 8 the histograms of D predictions concerning the unabeled training examples. As previously explained in the Section 3.2, if D associates exclusively the label 1 with the distribution p_N, then we can observe a mixture of two distributions in the corresponding histograms. The one on the right corresponds to D predictions for unlabeled negative examples. The second one on the left corresponds to D predictions for unlabeled positive examples. It is shifted away from the label 1 and centered around δ. Both distributions cannot be observed with BN. In contrast, SN considerably decreases this overfitting problem. Moreover, the addition of the dropout further helps, such that the dispersion of D predictions is attenuated. This confirms that BN is not compatible with the proposed framework. We conclude that the combination $SN + Dropout$ is the best solution to preserve the distinction between p_P and p_N for long trainings. This is consistent with the previously discussed arguments.

Now that we have validated the discriminator ability to separate positive and negative distributions from a positive unlabeled dataset, we select the most appropriate regularization techniques SN and dropout to train adversarially the discriminator and the generator hereafter. The proposed GAN based PU model ability to generate relevant counter-examples is assessed in the next section.

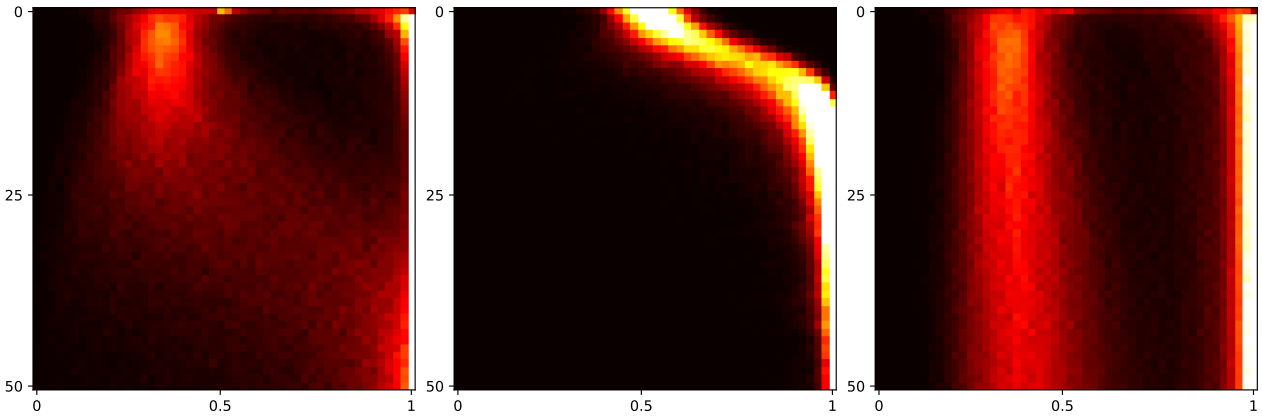
4.2.3. Generating counter-examples

From a qualitative point of view, and contrary to the PGAN model, the proposed D-GAN paradigm generates items which only follow the counter-examples distribution for diverse data types. This is illustrated in Figure 9 for 2D point datasets and in Figure 10 for image datasets.

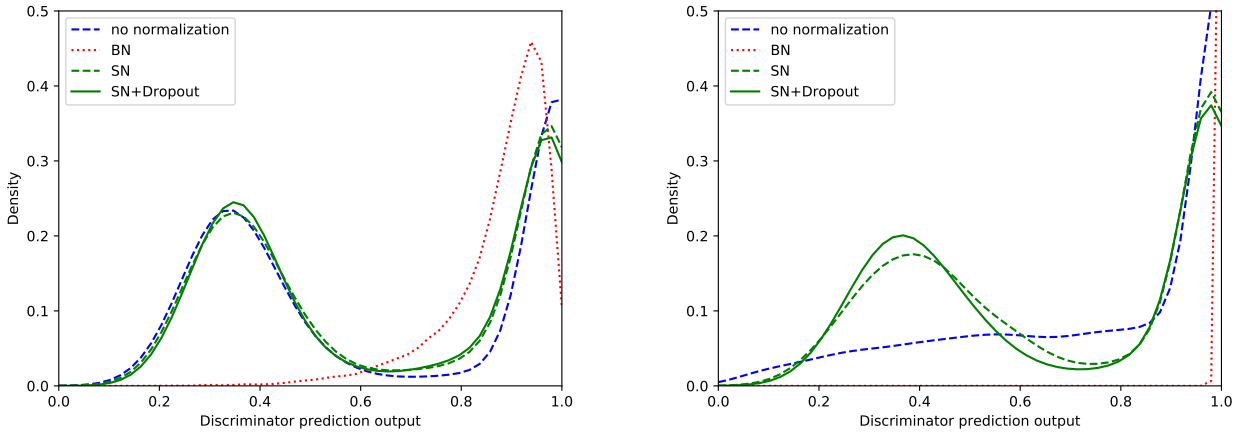
In Figure 9, we can observe on the top line that the generated samples exclusively follow the distribution of the counter-examples included in the unlabeled set (i.e. simultaneously not positive and unlabeled). On the bottom line, we can observe that the generator has learned the distribution of confident complements of the positive sample distribution over the uniform distribution of unlabeled sample. In addition, we can also observe that a small area around the positive sample distribution is not captured by the generator. This shows the ability of the proposed generative model to not overfit the positive sample distribution boundary.



(a) No norm (b) BN (c) SN



(d) No norm+Dropout (e) BN+Dropout (f) SN+Dropout



(g) Slice after 5 epochs (h) Slice after 25 epochs

Figure 8: D predictions on unlabeled training examples. (a), (b), (c), (d), (e), (f) images show the evolution of the histograms of predictions during the training of D. Each horizontal line of pixels represents the histogram of predictions, between 0 and 1 along the horizontal axis, of D on the entire unlabeled training dataset. Clear hot colors represent a high density of prediction. The vertical axis indicates the training iterations from 0 to 50 epochs. Figures (g) and (h) represent the corresponding histograms of predictions after 5 and 25 epochs. Settings are with positive class 8 and negative class 3 of MNIST dataset, with $\pi_P = 0.5$.

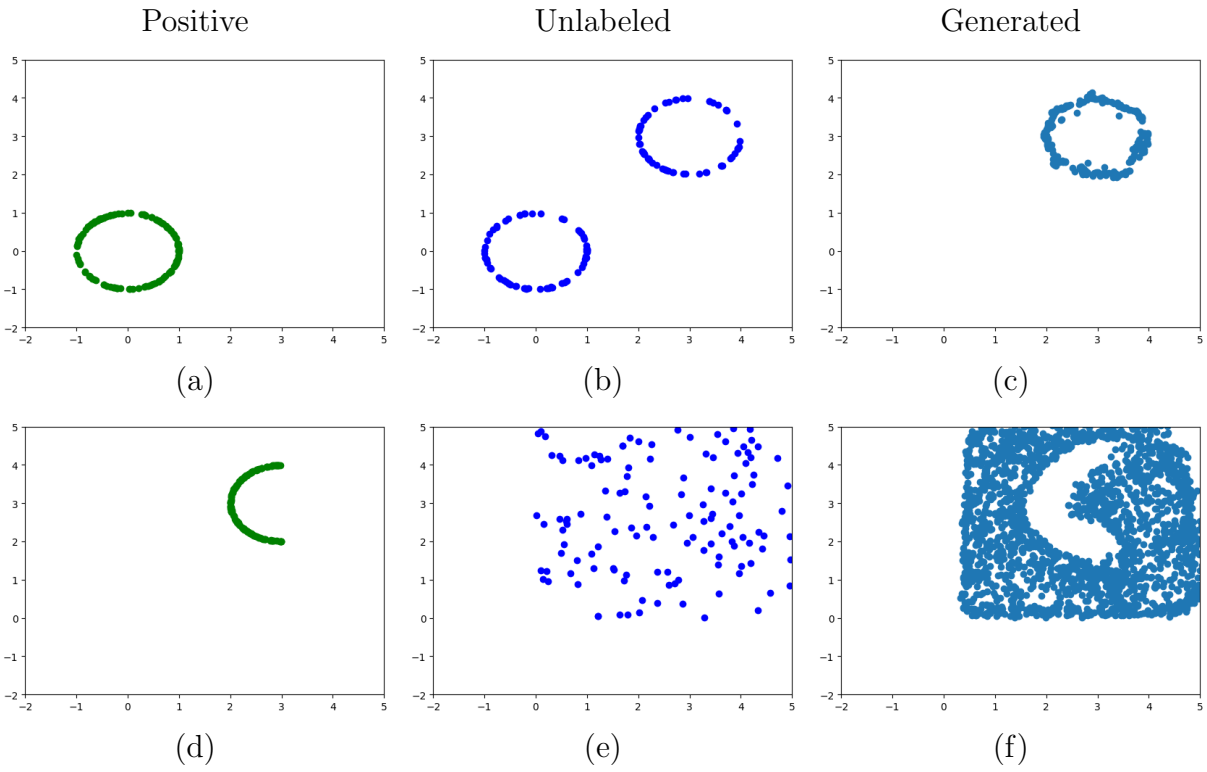


Figure 9: Proposed approach applied to two different clusters of 2D points. D and G have a multilayer-perceptron structure with respectively 128 hidden units. From left to right, figures are respectively labeled positive, unlabeled with $\pi_P = 0.5$, and generated samples. Figures (a), (b), (c) correspond to distributions following circle shapes. Figures (d), (e), (f) correspond to a half circle distribution for positive examples, and a uniform distribution over a defined interval for unlabeled examples.

In Figure 10, we can also observe that the generated examples systematically follow the counter-examples distribution on three image datasets: MNIST, CIFAR-10 and celebA.

Moreover, as mentioned previously, the regularization technique used in the discriminator has a direct impact on the samples generated by the generator. Figure 11 shows samples generated by G depending on the normalization technique used in D. We can observe that in the first column, with $\pi_P = 0.3$, we naturally obtain around thirty percent of men faces generated using any normalization techniques with the original GAN framework used in PGAN. The generated images quality seems visually equivalent between BN and SN. The D-GAN trained with $\pi_P = 0.3$ and BN naturally generates around thirty percents of men faces, as we recall that BN does not enable to capture the counter-examples distribution. Conversely, the D-GAN performs well with SN, as it exclusively generates women faces. Similar observations are made when $\pi_P = 0.5$. Those results are consistent with Sec. 4.2.2. This confirms that the generator behaviour is highly dependent

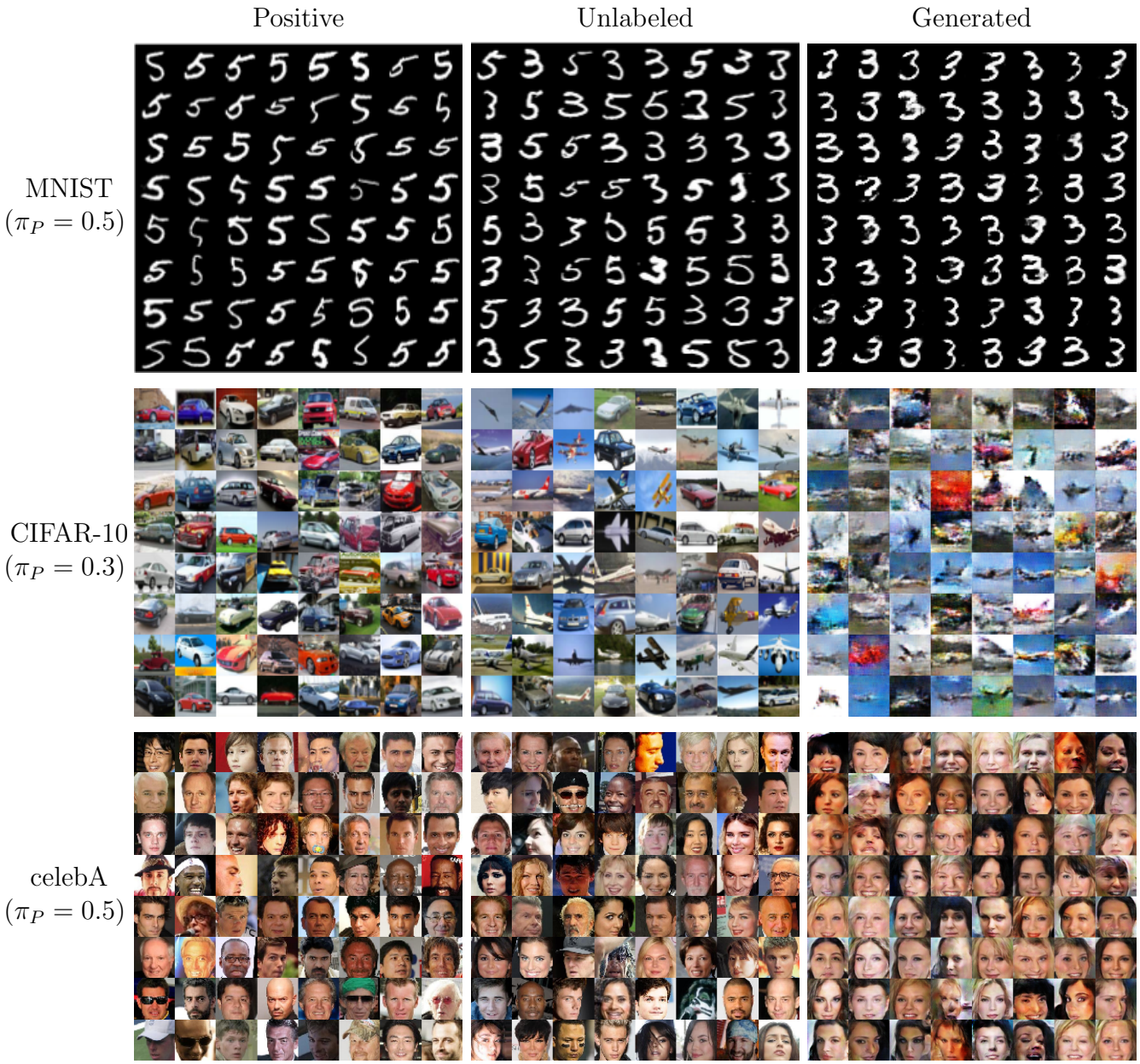


Figure 10: Counter-examples generation from Positive Unlabeled image datasets. The two left columns present input positive and unlabeled training samples x_P and x_U. The right column presents output generated minibatch samples x_G. The first row presents results for MNIST classification task 5-*vs*-3 when $\pi_P = 0.5$. The second row presents results for CIFAR-10 classification task *Car-vs-Airplane* when $\pi_P = 0.3$. The third row presents results for the arbitrary celebA classification task *Male-vs-Female* when $\pi_P = 0.5$. Visually, all generated samples follow the counter-examples distribution.

on the discriminator generalization ability, which in turn depends on normalization techniques used. This also confirms that the proposed D-GAN framework presents the interesting ability to exclusively hallucinate counter-examples on a real PU image dataset when it is combined with the appropriate discriminator regularization.

Figure 11: Discriminator regularizations impacts on the generated samples from a PU celebA image dataset after 100 training epochs iterations. The two rows respectively correspond to training experiments with BN and SN normalization techniques. The left column presents samples generated using the original LS-GAN discriminator loss function. The middle and right columns present the samples generated by integrating the proposed model discriminator loss function term $\mathbb{E}_{x_P \sim p_P}[MSE(D(x_P), 0)]$ in the original LS-GAN loss function, with MSE the mean squared error metric.

In order to enable reproducibility, a D-GAN implementation corresponding to Figure 11 is available[7] and is applied on the LS-GAN model [Qi, 2017]. Our code also includes the method proposed by [Chiaroni et al., 2018] to establish a PU training dataset from a fully labeled dataset with parameters ρ and π_P.

We have shown in this section, from a qualitative point of the view, the discriminator ability to separate positive and negative distributions from a positive unlabeled dataset, and the generator ability to learn the counter-examples distribution on various datasets during the first stage. Next, we propose in Sec. 4.3 to quantitatively evaluate the proposed model through an empirical study

[7]The code is available in supplementary material.

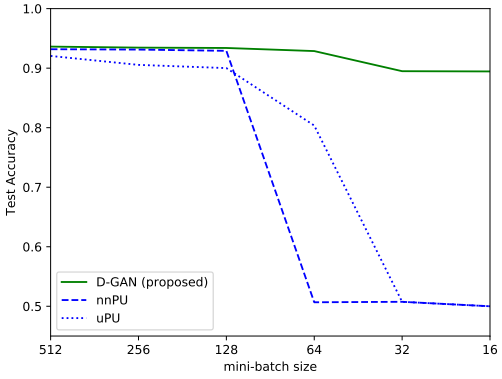
by focusing on the second-stage classifier C output predictions.

4.3. Divergent-GAN for Positive Unlabeled learning

In this section, we evaluate empirically our method on standard PU learning tasks in order to verify its ability to address respective issues of the state-of-the-art methods presented in Section 2. Concerning these comparative experiments, we use the DCGAN [Radford et al., 2015] architecture.

4.3.1. Robustness to prior noise

Nowadays, SGD remains a useful deep learning regularization technique for large-scale machine learning problems [Bottou, 2010]. SGD provides a regularizing effect by using minibatches [Wilson and Martinez, 2003]. However, a smaller batch size implies a higher prior noise per batch. Thus, in this section, we empirically study the proposed model robustness to prior noise using small batch sizes.



(a) Curves

Even-vs-Odd (MNIST)		**D-GAN**	nnPU	uPU
Without prior		✓	×	×
minibatch size	$std(\pi_P) \cdot 10^2$	Test Accuracy		
512	2.22	**0.936**	0.932	0.921
256	3.2	**0.935**	0.931	0.906
128	4.31	**0.934**	0.929	0.9
64	6.51	**0.929**	0.507	0.804
32	8.62	**0.895**	0.508	0.508
16	13.06	**0.907**	0.5	0.5

(b) Detailed scores

Figure 12: Prediction test Accuracy on MNIST for the *Even-vs-Odd* classification task, as a function of the minibatch size. We choose the prior value $\pi_P = 0.5$, as the standard deviation of the real prior per minibatch is the highest in this way (see Fig. 2). This eases to observe the prior sensitivity. We reproduce the experiment *exp-mnist* proposed by nnPU. The PU dataset contains one thousand positive labeled examples, which are *even digits*. The unlabeled set is composed of the entire initial dataset, thus including also the positive labeled ones. $std(\pi_P)$ is the standard deviation of the prior per minibatch. uPU and nnPU results have been obtained with the code provided by the authors of the nnPU work. (b) details the prediction scores used to plot the curves in (a).

We reproduce the *Even-vs-Odd* experiment proposed by [Kiryo et al., 2017] as a function of the batch training size. It consists of learning to discriminate *even* digits *0, 2, 4, 6, 8* from *odd* digits *1, 3, 5, 7, 9*. We use the same classifier architecture for nnPU, uPU and the D-GAN proposed

approach. It is the *multilayer perceptron* model provided by [Kiryo et al., 2017] [8]. We only replace the bottom fully connected layer of the classifier by a convolutional layer, similarly to the generator top layer and discriminator bottom layer in the DCGAN [Radford et al., 2015] structure that we use. This avoids compatibility problems between the generator top convolutional layer output and the bottom classifier layer input. Unwanted artifacts in output of GANs MLP structure are slightly different from unwanted artifacts observed in output of GANs convolutional structures. It turns out that PU approaches using prior such as uPU, nnPU and GenPU make the assumption that the global training dataset prior π_P is fixed and known. But in the same PU context, when the minibatch size decreases, the dispersion of π_P per minibatch consequently increases. Figure 12 (a) shows that using small batch training sizes causes critical prediction performances collapse issues for unbiased techniques like nnPU and uPU. On the other hand, our proposed approach without using prior knowledge is drastically less sensitive to this problem: While nnPU and uPU methods become ineffective in terms of test Accuracy (i.e. Accuracy score around 0.5), the D-GAN still provides a prediction test Accuracy of 0.907 for training minibatches of size 16 in D, G and C to address the *Even-vs-Odd* MNIST superclass classification task, as detailed in Figure 12 (b). We can conclude that the D-GAN outperforms nnPU and uPU in terms of prediction performances such that it can use minibatches to take advantage of SGD. This capacity is also interesting for incremental learning requirements where only small sample sizes may be managed at each new training iteration. Moreover, recent studies show that it is possible to continually train GANs models [Lesort et al., 2018].

Now that we have shown that the proposed model is robust to prior noise, we continue the comparative tests with the methods which do not use prior knowledge π_P in their training cost-functions to address the PU learning task.

4.3.2. One versus Rest challenge

We compare in this section the D-GAN proposed approach with PGAN and RP [9] methods that we consider as baselines for the PU learning task without prior knowledge. D-GAN and

[8] The code is available at: `https://github.com/kiryor/nnPUlearning`.
[9] RP code is available at: `https://github.com/cgnorthcutt/rankpruning`.

PGAN GAN-based approaches use the same generator and discriminator architectures presented in Figures 5 (a) and (b). Moreover, D-GAN, PGAN and RP approaches use the same classifier presented in Figure 5 (c). We evaluate them on the challenging One-vs-Rest task which consists of trying to distinguish a class from all the other ones. This task is interesting for binary image classification applications where the labeling effort may be exclusively done on the class of interest, the positive class. Another motivation is that One-vs-Rest binary classification brings the tools for multiclass classification [Shalev-Shwartz and Ben-David, 2014].

One-vs-Rest	AVG_{MNIST}					$AVG_{\text{CIFAR-10}}$				
π_P	PN	PNGAN	**D-GAN**	PGAN	RP	PN	PNGAN	**D-GAN**	PGAN	RP
0.1	0.993	0.988	**0.989** (0.01)	0.965 (0.01)	0.967 (0.02)	0.680	0.812	**0.815** (0.05)	0.745 (0.08)	0.622 (0.10)
0.3	0.993	0.988	**0.983** (0.01)	0.958 (0.01)	0.975 (0.02)	0.680	0.812	**0.792** (0.05)	0.760 (0.03)	0.730 (0.07)
0.5	0.993	0.988	**0.971** (0.01)	0.946 (0.02)	0.951 (0.04)	0.680	0.812	**0.751** (0.04)	0.748 (0.03)	0.716 (0.06)
0.7	0.993	0.988	**0.938** (0.02)	0.875 (0.05)	0.933 (0.07)	0.680	0.812	**0.721** (0.04)	0.702 (0.03)	0.684 (0.08)

Table 2: One-vs-Rest task with two-stage **PU methods without prior**, as proposed in PGAN [Chiaroni et al., 2018]: From a fully labeled PN dataset, we firstly select a fraction ρ of positive labeled examples that we put in the simulated unlabeled set. Then, we add negative labeled examples in the latter to obtain up to a fraction π_P of positive examples in this unlabeled set. Compared to nnPU simulation method, this simulation method has the advantage to simultaneously and independently control the number of positive labeled examples to keep, and the fraction π_P for the unlabeled set to simulate. *PNGAN* expression represents GAN-based methods reference for the ideal case where $\pi_P = 0$, such that we train during the first stage a GAN exclusively over all the initial cleanly labeled counter-examples set. For each dataset and depending on the fraction π_P, we have tested respectively the ten One-vs-Rest task possibilities and display the corresponding average test F1-score predictions. The standard deviation is indicated in parenthesis.

Table 2 shows average predictions for the One-vs-Rest task over MNIST and CIFAR-10 datasets. We use the F1-Score metric for its relevance in such information retrieval and binary classification tasks as highlighted by Liu et al. [2002]: the F1-score measures the positive examples retrieval. The PU datasets are simulated as proposed by PGAN such that we can evaluate the results as a function of several π_P fractions. Concerning the second-stage classifier in these experiments, we have used the convolutional architecture presented in Figure 5 (c). We can observe that the D-GAN globally outperforms PGAN and RP methods in terms of test F1-Score on both MNIST and CIFAR-10 datasets. Moreover, PNGAN results highlight the GAN-based methods data augmentation advantage on complex datasets. This justifies the superior scores obtained by our method compared to RP over the CIFAR-10 dataset.

Reducing the overfitting problem: In addition, we can observe that the proposed model also outperforms PGAN on MNIST by a significant margin. This is due to the fact that, compared

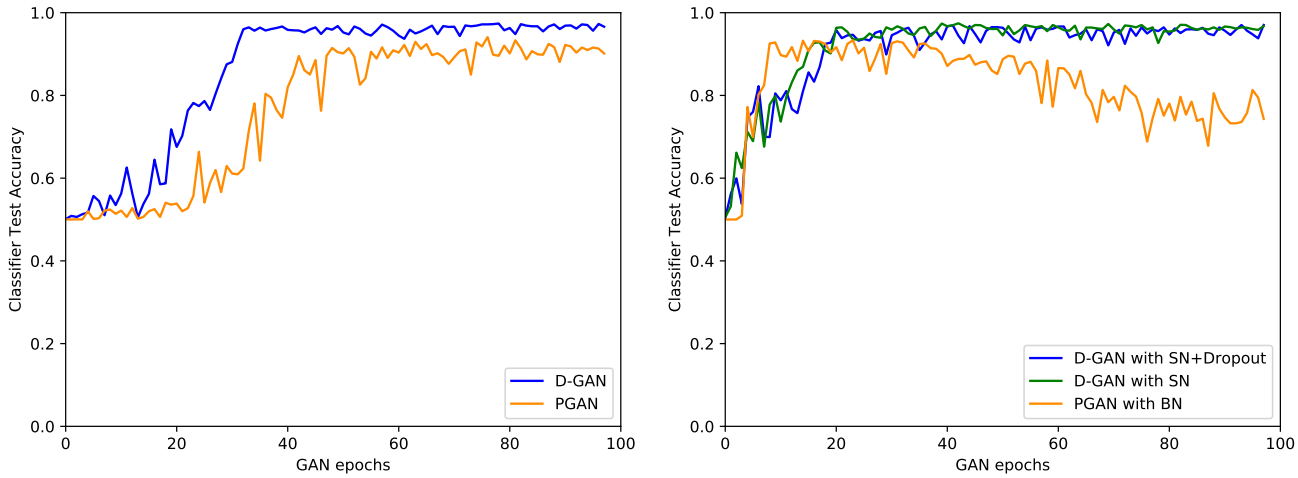


Figure 13: Second-stage Classifier (architecture presented in Figure 5 (c)) test Accuracy evolution as a function of the first-stage GAN epochs. 8-vs-Rest MNIST task, with $\rho = 0.5$ and $\pi_P = 0.5$. (a) D-GAN and PGAN are trained without normalization layers. (b) D-GAN and PGAN are respectively trained with SN, SN + dropout, and BN inside the discriminator.

to the PGAN which is trained to generate unlabeled examples, the proposed approach only generates counter-examples as previously shown in Figures 9 and 10. Consequently, the proposed first-stage generative model does not learn the positive samples distribution, and it avoids the PGAN first-stage overfitting issue on simple datasets like MNIST. Figure 13 illustrates this phenomenon. In Figure 13 (a), without normalization, the D-GAN method gets faster a better Accuracy than PGAN. In Figure 13 (b), the D-GAN with SN or SN+dropout follows the learning speed of the PGAN with BN, while demonstrating a steadier behaviour once the Accuracy progression is finished, as it overcomes the PGAN first-stage overfitting problem.

To sum up, in Sec. 4.2, we demonstrate that the proposed approach is effective at capturing and observing the counter-examples distribution of our class of interest from only positive and unlabeled data, without using the prior information π_P. In addition, comparative experiments in Sec. 4.3 have subsequently highlighted the proposed model ability to address state-of-the-art PU learning issues such as prior sensitivity and first-stage overfitting. It turns out that addressing simultaneously those issues fosters the proposed approach to outperform PU state-of-the-art methods in terms of prediction scores without using prior on both simple and complex image datasets.

4.4. Limitations of the proposed approach

We recall that the counter-examples generator G consists of learning the distribution of examples associated by D to the label 1. For this reason, the proposed approach requires that D associates a mutual intermediate label δ to both labeled and unlabeled positive examples, such that D exclusively associates counter-examples to the label 1, as formalized in Sec. 3.2. These conditions are not met if:

- D lacks of generalization such that it overfits positive examples, by associating the labeled ones to the label 0 and the unlabeled ones to the label 1, instead of associating all positive examples to the mutual intermediate label δ. As empirically demonstrated in Sec. 4.2.2, an effective solution is to combine regularization techniques such as dropout with SN;

- D is not sufficiently flexible to separate p_P from p_N, thus considering these distributions as too similar. Consequently, a unique mutual intermediate label would be predicted by D for both p_P and p_N. It may be interesting to explore this failure case through distinct distributions sharing common patterns, such as face attributes of celebA dataset [Liu et al., 2015] images;

- The labeled positive set is partially corrupted such that it contains a fraction of negative examples. For reasons of expectation linearity, p_N would consequently be associated by D to an intermediate label between 0 and 1. A solution may be to extend the proposed framework to this noisy PU learning challenge by drawing on existing asymmetric noisy labeled learning techniques [Chiaroni et al., 2019]. More specifically, by modifying training loss functions, it is possible to enforce G to learn the distribution corresponding to this given intermediate label predicted by D for p_N.

5. Conclusion

In this work, we proposed to incorporate a constrained PU risk into the GAN discriminator loss function in order to deal with PU learning. In this way, the proposed model generates relevant counter-examples from a PU dataset. It outperforms state-of-the-art PU learning methods by

addressing their respective issues. Namely, it addresses the prior knowledge dependence of cost-sensitive PU methods and the lack of generalization of selective processes. Moreover, it reduces the PGAN first-stage overfitting, while keeping the minimalist standard GAN architecture, such that it is easily adaptable to recent GANs variants.

Nevertheless, although being competitive with the state-of-the-art, the proposed approach may present some failure cases under specific conditions, as outlined in Sec. 4.4. Future research work will aim at addressing these limitations. Furthermore, as adversarial and weakly supervised learning techniques are continuously evolving, we believe that the proposed approach stability, prediction performances, and computational cost still have the potential to be improved. For instance, recent promising GAN training approaches [Brock et al., 2019], not mandatorily using BN, may be suitable to extend the proposed approach for higher dimensional image datasets.

6. References

Arjovsky, M., Chintala, S., Bottou, L., 2017. Wasserstein generative adversarial networks, in: International Conference on Machine Learning, pp. 214–223.

Bottou, L., 2010. Large-scale machine learning with stochastic gradient descent, in: Proceedings of COMPSTAT'2010. Springer, pp. 177–186.

Brock, A., Donahue, J., Simonyan, K., 2019. Large scale GAN training for high fidelity natural image synthesis, in: International Conference on Learning Representations.

de Carvalho Pagliosa, L., de Mello, R.F., 2018. Semi-supervised time series classification on positive and unlabeled problems using cross-recurrence quantification analysis. Pattern Recognition 80, 53–63.

Chiaroni, F., Rahal, M.C., Hueber, N., Dufaux, F., 2018. Learning with a generative adversarial network from a positive unlabeled dataset for image classification, in: IEEE International Conference on Image Processing.

Chiaroni, F., Rahal, M.C., Hueber, N., Dufaux, F., 2019. Hallucinating a Cleanly Labeled

Augmented Dataset from a Noisy Labeled Dataset Using GANs, in: IEEE (Ed.), 26th IEEE International Conference on Image Processing (ICIP).

Christoffel, M., Niu, G., Sugiyama, M., 2016. Class-prior Estimation for Learning from Positive and Unlabeled Data, in: Asian Conference on Machine Learning, pp. 221–236.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B., 2016. The cityscapes dataset for semantic urban scene understanding, in: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Denis, F., 1998. PAC learning from positive statistical queries, in: International Conference on Algorithmic Learning Theory, Springer. pp. 112–126.

Du Plessis, M., Niu, G., Sugiyama, M., 2015. Convex formulation for learning from positive and unlabeled data, in: International Conference on Machine Learning, pp. 1386–1394.

Ekambaram, R., Fefilatyev, S., Shreve, M., Kramer, K., Hall, L.O., Goldgof, D.B., Kasturi, R., 2016. Active cleaning of label noise. Pattern Recognition 51, 463–480.

Elkan, C., Noto, K., 2008. Learning classifiers from only positive and unlabeled data, in: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM. pp. 213–220.

Farnia, F., Zhang, J., Tse, D., 2019. Generalizable adversarial training via spectral normalization, in: International Conference on Learning Representations.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2014. Generative adversarial nets, in: Advances in neural information processing systems, pp. 2672–2680.

Hou, M., Chaib-Draa, B., Li, C., Zhao, Q., 2018. Generative adversarial positive-unlabeled learning, in: Proceedings of the 27th International Joint Conference on Artificial Intelligence, AAAI Press. pp. 2255–2261.

Ioffe, S., Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: International Conference on Machine Learning, pp. 448–456.

Kiryo, R., Niu, G., du Plessis, M.C., Sugiyama, M., 2017. Positive-Unlabeled Learning with Non-Negative Risk Estimator, in: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 30. Curran Associates, Inc., pp. 1675–1685.

Krizhevsky, A., Hinton, G., 2009. Learning multiple layers of features from tiny images .

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., 1998. Gradient-based learning applied to document recognition. Proceedings of the IEEE 86, 2278–2324.

Lesort, T., Caselles-Dupré, H., Garcia-Ortiz, M., Goudou, J.F., Filliat, D., 2018. Generative Models from the perspective of Continual Learning, in: Workshop on Continual Learning, NeurIPS 2018 - Thirty-second Conference on Neural Information Processing Systems, Montréal, Canada.

Li, M., Pan, S., Zhang, Y., Cai, X., 2016. Classifying networked text data with positive and unlabeled examples. Pattern Recognition Letters 77, 1–7.

Liu, B., Lee, W.S., Yu, P.S., Li, X., 2002. Partially supervised classification of text documents, in: International Conference on Machine Learning, Citeseer. pp. 387–394.

Liu, Z., Luo, P., Wang, X., Tang, X., 2015. Deep learning face attributes in the wild, in: Proceedings of International Conference on Computer Vision (ICCV).

Mao, X., Li, Q., Xie, H., Lau, R.Y., Wang, Z., Paul Smolley, S., 2017. Least squares generative adversarial networks, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 2794–2802.

Miyato, T., Kataoka, T., Koyama, M., Yoshida, Y., 2018. Spectral normalization for generative adversarial networks, in: International Conference on Learning Representations.

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

Northcutt, C.G., Wu, T., Chuang, I.L., 2017. Learning with Confident Examples: Rank Pruning for Robust Classification with Noisy Labels, in: Uncertainty in Artificial Intelligence (UAI).

du Plessis, M.C., Niu, G., Sugiyama, M., 2014. Analysis of learning from positive and unlabeled data, in: Advances in neural information processing systems, pp. 703–711.

Qi, G.J., 2017. Loss-sensitive generative adversarial networks on lipschitz densities. arXiv preprint arXiv:1701.06264 .

Radford, A., Metz, L., Chintala, S., 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 .

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L., 2015. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) 115, 211–252.

Sansone, E., De Natale, F.G., Zhou, Z.H., 2018. Efficient training for positive unlabeled learning. IEEE Transactions on Pattern Analysis and Machine Intelligence .

Shalev-Shwartz, S., Ben-David, S., 2014. Understanding machine learning: From theory to algorithms. Cambridge university press.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research 15, 1929–1958.

Ward, G., Hastie, T., Barry, S., Elith, J., Leathwick, J.R., 2009. Presence-only data and the EM algorithm. Biometrics 65, 554–563.

Wilson, D.R., Martinez, T.R., 2003. The general inefficiency of batch training for gradient descent learning. Neural Networks 16, 1429–1451.

Zhu, X., Liu, Y., Li, J., Wan, T., Qin, Z., 2018. Emotion classification with data augmentation using generative adversarial networks, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer. pp. 349–360.