Self-PU: Self Boosted and Calibrated Positive-Unlabeled Training

Aeiau Zzzz [*1] **Bauiu C. Yyyy** [*1 2] **Cieua Vvvvv** [2] **Iaesut Saoeu** [3] **Fiuea Rrrr** [1] **Tateu H. Yasehe** [3 1 2]
Aaoeu Iasoh [2] **Buiui Eueu** [3] **Aeuia Zzzz** [3] **Bieea C. Yyyy** [1 2] **Teoau Xxxx** [3] **Eee Pppp** [3]

Abstract

Many real-world applications have to tackle the Positive-Unlabeled (PU) learning problem, *i.e.*, learning binary classifiers from a large amount of unlabeled data and a few labeled positive examples. While current state-of-the-art methods employ importance reweighting to design various risk estimators, they completely ignored the learning capability of the model itself, which could provide reliable supervision. This motivates us to propose a novel **Self-PU** learning framework, which seamlessly integrates PU learning and self-training. Self-PU highlights three "self"-oriented building blocks: a *self-paced* training algorithm that adaptively discovers and augments confident positive/negative examples as the training proceeds; a *self-calibrated* instance-aware loss; and a *self-distillation* scheme that introduces teacher-students learning as an effective regularization for PU learning. We demonstrate the state-of-the-art performance of Self-PU on common PU learning benchmarks (MNIST and CIFAR10), which compare favorably against the latest competitors. Moreover, we study a real-world application of PU learning, *i.e.*, classifying brain images of Alzheimer's Disease. Self-PU obtains significantly improved results on the renowned Alzheimer's Disease Neuroimaging Initiative (ADNI) database over existing methods.

1. Introduction

For standard supervised learning of binary classifiers, both positive and negative classes need to be collected for training purposes. However, this is not always a realistic setting in many applications, where one certain class of data could be difficult to be collected or annotated. For example, in chronic disease diagnosis, while we might safely consider a diagnosed patient to be "positive", the much larger population of "undiagnosed" individuals are practically mixed with both "positive" (patient) and "negative" (healthy) examples, since people might be undergoing the disease's incubation period (Armenian & Lilienfeld, 1974) or might just have not seen doctors. Roughly labeling the "undiagnosed" examples all as negative will hence lead to biased classifiers that inevitably underestimate the risk of chronic disease.

Given those practical demands, Positive-Unlabeled (PU) Learning has been increasingly studied in recent years, where a binary classifier is to be learned from a part of positive examples, plus an unlabeled sample pool of mixed and unspecified positive and negative examples. Because of this weak supervision, PU learning is more challenging than standard supervised or semi-supervised classification problems. Early works tried to identify reliable negative examples from the unlabeled data by hand-crafted heuristics or standard semi-supervised learning methods (Liu et al., 2002; Li & Liu, 2003). Recently, importance reweighting methods like unbiased PU (uPU) (Du Plessis et al., 2014; 2015) treat unlabeled data as negative ones associated with reduced weights. non-negative PU (nnPU) (Kiryo et al., 2017) incorporated both the reweighting and overfitting avoidance. Despite these success, self-supervision via auxiliary or surrogate tasks are never considered, which could potentially provide reliable supervision.

This motivates us to explore and leverage the learning capability of the model itself. The key insight is that more reliable supervision could be attained for PU learning. Our proposed **Self-PU** learning framework exploits three aspects of such "self-boosts": (a) we design a self-paced training strategy to progressively select unlabeled examples and update the "trust" set of confident examples; (b) we explore a fine-grained calibration of the functions for unconfident examples in a meta-learning fashion; and (c) we construct a collaborative self-supervision between teacher and student models, and enforce their consistency as a new regularization, against the weak supervision in PU learning. Our main contributions are outlined as follows:

[*]Equal contribution [1]Department of Computation, University of Torontoland, Torontoland, Canada [2]Googol ShallowMind, New London, Michigan, USA [3]School of Computation, University of Edenborrow, Edenborrow, United Kingdom. Correspondence to: Cieua Vvvvv <c.vvvvv@googol.com>, Eee Pppp <ep@eden.co.uk>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

- A novel **self-paced** learning pipeline is first introduced to adaptively mine confident examples from unlabeled data and to label them into trusted positive/negative classes. A hybrid loss is applied on both the augmented "labeled" examples and the remaining unlabeled data for supervision. The procedure is repeated progressively, with more unlabeled examples selected.
- A **self-calibration** strategy is leveraged to further explore fine-grained treatment of loss functions over unconfident examples, in a meta-learning fashion.
- A **self-distillation** scheme is designed via the collaborative training between several teacher networks and student networks, providing a consistency regularization as another fold of self-supervision.
- In addition to standard testbeds (MNIST, CIFAR10), a new real-world application of PU learning, *i.e.*, Alzheimer's Disease neuroimage classification, is evaluated for the first time. On the Alzheimer's Disease Neuroimaging Initiative (ADNI) database, Self-PU achieves superior results over existing solutions.

2. Related Work

2.1. PU Learning

Let $X \in \mathbb{R}^d$ and $Y \in \{+1, -1\}$ ($d \in \mathbb{N}$) be the input and output random variables. In PU learning, the training dataset D is composed of a positive set D_P and an unlabeled set D_U, where we have $D = D_P \cup D_U$. D_P contains n_p positive examples $\boldsymbol{x}_p$ sampled from $P(\boldsymbol{x}|Y=+1)$ and D_U contains n_u unlabeled examples $\boldsymbol{x}_u$ sampled from $P(\boldsymbol{x})$. Denote the class prior probability $\pi_p = P(Y=+1)$ and $\pi_n = P(Y=-1)$, where we follow the convention (Kiryo et al., 2017) to assume π_p as known throughout the paper. Denote the binary classifier g and the loss function L. The classifier g needs to be trained from D_P and D_U.

The risk of classifier g, $\hat{R}_{PU}(g)$ can be approximated by

$$\begin{aligned}\hat{R}_{PU}(g) = \pi_p \mathop{\mathbb{E}}_{X|Y=+1}[L(g(X)]+ \\ (\mathop{\mathbb{E}}_{X}[L(-g(X))] - \pi_p \mathop{\mathbb{E}}_{X|Y=+1}[L(-g(X))])\end{aligned} \tag{1}$$

which has been known as the unbiased risk estimator for uPU (Elkan & Noto, 2008; Du Plessis et al., 2014; 2015). It was later pointed out that the second line in Eq. (1) would become negative due to the overfitting of complex models (Kiryo et al., 2017). A non-negative version (nnPU) of Eq. (1) was therefore suggested by the authors:

$$\begin{aligned}\hat{R}_{PU}(g) =& \pi_p \mathop{\mathbb{E}}_{X|Y=+1}[L(g(X)] + \max(0, \mathop{\mathbb{E}}_{X}[L(-g(X))]- \\ & \pi_p \mathop{\mathbb{E}}_{X|Y=+1}[L(-g(X))])\end{aligned} \tag{2}$$

Importance reweighting methods (e.g. uPU, nnPU) achieve the state of the arts, although treating unlabeled data as "weighted" negative examples still brings in unreliable supervision. Generative adversarial networks were introduced by Hou et al. (2018), where the conditional generator produced both negative and positive examples resembling the unlabeled real data. DAN (Liu et al., 2019) tried to recover the positive and negative distributions from the unlabeled data without requiring the class prior.

2.2. Self-Paced Learning

Self-paced learning (Kumar et al., 2010) was presented as a special case of curriculum learning (Bengio et al., 2009), where the feed of training examples was dynamically generated by the model based on its learning history, aiming to simulate the learning principle of starting by learning easy instances and then gradually taking more challenging cases (Khan et al., 2011). Early PU works designed heuristics for sample selection. In recent "aaPU" work (Xu et al., 2019), positive instead of negative examples are permanently selected by analyzing the distribution of sample loss. Unlike previous PU learning works which rely on crafted sample selection heuristics, we are the first to leverage the data-driven self-paced learning to progressively turn unlabeled data into labeled ones.

2.3. Self-Supervised Learning

In many supervision-starved fields (*e.g.* medical data), it is generally hard to obtain accurate annotations, despite the vast number of unlabeled data available for training. Self-supervised learning aims to form supervision for learning informative/discriminative features, from the data itself. Models are required to predict on "proxy" tasks that are relevant to the main goal. For example, the Γ model (Laine & Aila, 2016) augmented each unlabeled example with random noises, and forced consistency between the two predictions. In (Tarvainen & Valpola, 2017), two identical models were used during training: the student learned as usual while the teacher model generated labels and updated its weights through a moving average of the student, forcing consistency between two models. (Zhang et al., 2018) further suggested that, instead of exchanging examples , mutual feature distillation between peer networks can form another strong source of supervision, and can enable the collaborative learning of an ensemble of students. To our best knowledge, we are the first to consider such self-supervision to improve PU learning.

3. The Self-PU Framework

Our proposed Self-PU framework exploits the learning capability of the model itself via self-training (Figure 1). We first design a self-paced learning pipeline to progressively select and label confident examples from unlabeled data for supervised learning. On top of that, we calibrate the loss functions over the unconfident examples via meta-learning. Moreover, consistency loss is introduced between peer networks with

different learning paces, which collaboratively teach each other. We further extend our consistency from peer networks to their moving-averages (Tarvainen & Valpola, 2017; Laine & Aila, 2016), as another form of supervision.

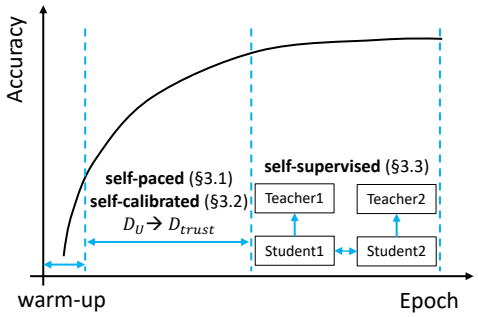

Figure 1. Illustration of the proposed "Self-PU" framework. After a short warm-up period, the classifier is first trained with self-paced learning, where confident examples in D_U are progressively selected and labeled (positive/negative) into a trusted D_{trust} subset for supervised learning, with the loss functions over unconfident examples carefully calibrated. After collecting enough confident examples, we start the self-supervised learning via distillation between two collaborative students and their teacher networks.

3.1. Self-Paced PU Learning

Despite the success of unbiased PU risk estimators, they still rely on the estimated class prior and reduced weights on unlabeled data. As shown in (Arpit et al., 2017), during gradient descent, deep neural networks tend not to memorize all training data at the same time but tend to memorize frequent or easy patterns first and later irregular patterns. If we could first find out easy examples and label them with confidence, and then augmenting this labeled pool for the training progress, then we can enjoy "progressively increased" confident full supervision along the training, in addition to the weak supervision from the PU risk estimators.

Given the model g, an input example $\boldsymbol{x}$ and the corresponding label y, we may compute the output $g(\boldsymbol{x})$ and then calculate the probability of $\boldsymbol{x}$ being positive as $p(\boldsymbol{x}) = P(Y = +1|\boldsymbol{x}) = f(g(\boldsymbol{x}))$, where f is a monotonic function of mapping $\mathbb{R}^n \rightarrow [0, 1]$ (*e.g.* sigmoid function). A greater $p(\boldsymbol{x})$ suggests higher confidence that $\boldsymbol{x}$ belongs to positive class as predicted by g, and vice versa. By descending sort of $p(\boldsymbol{x})$ each time, we can select n most confident positive and n most confident negative examples from the current unlabeled data pool D_U. They will be removed from D_U and added to our trusted subset D_{trust}, considered as labeled training examples hereinafter.

Letting L_{XE} be the cross-entropy loss and L_{nnPU} be the nnPU loss, and together with the given positive subset D_P, our hybrid loss for self-paced learning becomes:

$$L_{\text{SP}} = \sum_{\boldsymbol{x} \in D_{\text{trust}}} L_{\text{XE}}(\boldsymbol{x}) + \sum_{\boldsymbol{x} \in D_P \cup (D_U - D_{\text{trust}})} L_{\text{nnPU}}(\boldsymbol{x}) \quad (3)$$

It is worth noting that previous works select either only confident positive examples (Xu et al., 2019) or negative examples (Li & Liu, 2003), while our self-paced learning selects both. Since the cross entropy is used as our supervised loss, the advantage is that the class distribution in each sampling step can be well balanced, where equal number (*i.e.* n) of trusted positive and negative examples are selected, in order to avoid the potential pitfall of extreme class imbalance caused by incrementally sampling only one class.

Except for that, previous sample selection approaches (Xu et al., 2019) stick to a fixed, rigid learning schedule. In contrast, we unleash more flexibility for the model to automatically and adaptively adjust its own learning pace, via the following techniques. Later on, we will experimentally verify their effectiveness via a step-by-step ablation study.

3.1.1. Dynamic Rate Sampling

As the learning progresses, training examples with easy/frequent patterns and those with harder/irregular patterns are memorized in different training stages (Arpit et al., 2017). It is important to make our self-paced learning compatible with the memorizing process of the model. A small number of easy examples should be selected first, and then intermediate to hard examples can be labeled after the model is well-trained. Instead of fixing the number of selected confident examples, we propose to dynamically choose the number of confident examples during the self-paced learning. Specifically, as the self-paced learning proceeds, we linearly increase the size of D_{trust} from 0 to $r|D_U|$, where the sampling ratio r could range from 10% to 40% in our experiments (see section 4.3.2). Empirically, we first "warm-up" the model by training 10 epochs before starting the self-paced learning, in order to keep the selected confident examples as accurate as possible.

3.1.2. "In-and-Out" Trusted Set

In previous sample selection approaches, once selected, a trusted example will never be deprived of its label during the subsequent training. In contrast, we allow our training to "regret" on earlier selections in D_{trust}. Especially at the early training stage, the intermediate model might not be well-trained enough and not always reliable for predicting labels, which could mislead training if continuing acting as supervision. To this end, we adaptively update D_{trust} by also re-examining its current examples each time when we augment new confident ones. The previously selected examples will be removed from D_{trust} if their predictions by the current model become of low-confidence, and will be treated as unlabeled again.

3.1.3. Soft Labels

Instead of giving the selected confident examples hard labels, we directly use the prediction $f(g(\boldsymbol{x}))$ as soft labels: $[1 - f(g(\boldsymbol{x})), f(g(\boldsymbol{x}))]$ as $[P(Y = -1|\boldsymbol{x}), P(Y = +1|\boldsymbol{x})]$,

as the practice of label smoothing (Szegedy et al., 2016) appears to benefit learning robustness against label noise.

3.2. Self-Calibrated Loss Reweighting

Only leveraging nnPU loss on $D_U - D_{\text{trust}}$ may not be optimal, as some examples in this set can still provide meaningful supervision. To exploit more supervision from this noisy sets, we introduce a *learning-to-reweight* paradigm (Ren et al., 2018) to the PU learning field for the first time. Specifically, we hope to adaptively combine L_{XE} and L_{nnPU} for each training example i in $D_U - D_{\text{trust}}$, that can best minimize the validation loss at each training step, namely:

$$l_i(\boldsymbol{x}_i) = w_{i,1} \cdot L_{\text{XE}}(\boldsymbol{x}_i) + w_{i,2} \cdot L_{\text{nnPU}}(\boldsymbol{x}_i). \quad (4)$$

Note that we also adopt the soft label strategy (section 3.1.3) in L_{XE} here. To learn the optimal $\boldsymbol{w}_i = [w_1, w_2]$ together in training, we take a single gradient descent step on a mini-batch of validation examples w.r.t. $\boldsymbol{w}_i$, and then rectify the output to be non-negative:

$$\boldsymbol{w}_i^* = \arg\min_{w_{1,i}, w_{2,i}} \frac{1}{M} \sum_{j=1}^{M} L_{\text{XE}}(\boldsymbol{x}_j^v) \quad (5)$$

$$\boldsymbol{u}_i = -\frac{\partial}{\partial \boldsymbol{w}_i} \frac{1}{M} \sum_{j=1}^{M} L_{\text{XE}}(\boldsymbol{x}_j^v)|_{\boldsymbol{w}_i = \boldsymbol{0}} \quad (6)$$

$$\tilde{\boldsymbol{w}}_i = \max(\boldsymbol{u}_i, \boldsymbol{0}) \quad (7)$$

$$w_{1,i} = \frac{\tilde{w}_{1,i}}{\sum_i \tilde{w}_{1,i}}, w_{2,i} = \frac{\tilde{w}_{2,i}}{\sum_i \tilde{w}_{2,i}} \quad (8)$$

where M denotes the mini-batch size on the validation set which contains clean positive and negative examples, and $\boldsymbol{x}_j^v (j \in [1, M])$ is an example from the validation set which has the accurate ground truth.

Meanwhile, on $D_U - D_{trust}$, weighting the cross-entropy loss too much might not be beneficial to the classifier, especially when the soft labels are not accurate enough. Therefore, we restrict the total weights of the cross-entropy loss via a balancing factor γ:

$$T = \sup\{k : \sum_{i=1}^{k} w_{2,i} < \gamma M\} \quad (9)$$

$$w_{1,i}^* = w_{1,i} 1_{\{i<T\}} + 1_{\{i \geq T\}} \quad (10)$$

$$w_{2,i}^* = w_{2,i} 1_{\{i<T\}} \quad (11)$$

The corresponding hybrid loss becomes:

$$L_{\text{SP+Reweight}} = \sum_{\boldsymbol{x} \in D_{\text{trust}}} L_{\text{XE}}(\boldsymbol{x}) + \sum_{\boldsymbol{x} \in D_P \cup (D_U - D_{\text{trust}})} l(\boldsymbol{x}), \quad (12)$$

where $l(\boldsymbol{x}) = \sum_{i=1}^{M} l_i(\boldsymbol{x}_i)/M$.

3.3. Self-Supervised Consistency via Distillation

To explore additional sources of supervision, we encourage *two forms of self-supervised consistency*: among different learning paces of the model, and along the model's own moving averaged trajectory. The two goals are altogether achieved by an innovative distillation scheme, with a pair of collaborative student models and their teacher models.

3.3.1. Consistency for Different Learning Paces: Making A Pair of Students

The consistency between two self-paced models trained with different paces (*i.e.* sampling ratio in self-paced learning) makes the trained model more resilient to perturbations caused by training stochasticity. To form this self-supervision, we simultaneously train two networks that share the identical architecture, with the same D_P and D_U to start on. They are however set with different confidence thresholds and select different amounts from D_{U} to D_{trust} each time, making their learning paces un-synchronized and resulting in two different "trusted sets" D_{trust1} and D_{trust2}. Since each network starts from a different initial condition, their estimation of the probabilities of classes vary and therefore provide extra information in distillation. We then add a Kullback-Leibler (KL) Divergence regularization on the two models' predictions on $D_U - D_{\text{trust1}}$ and $D_U - D_{\text{trust2}}$, for encouraging their consistency, even though one is a "more aggressive learner" than the other.

Denote two networks g_1 and g_2, the KL divergence from g_1 to g_2 is defined over $D_U - D_{\text{trust1}}$:

$$D_{KL}(g_2||g_1) = \sum_{\boldsymbol{x} \in D_U - D_{\text{trust1}}} g_2(\boldsymbol{x}) \log \frac{g_2(\boldsymbol{x})}{g_1(\boldsymbol{x})}. \quad (13)$$

The KL divergence from g_2 to g_1 is defined over $D_U - D_{\text{trust2}}$:

$$D_{KL}(g_1||g_2) = \sum_{\boldsymbol{x} \in D_U - D_{\text{trust2}}} g_1(\boldsymbol{x}) \log \frac{g_1(\boldsymbol{x})}{g_2(\boldsymbol{x})}. \quad (14)$$

A *hard sample mining* strategy is further adopted on top, where we only calculate KL divergence over those "challenging" unlabeled examples who show large nnPU loss (Eq. (2)). Small nnPU loss indicates easy examples where we do not apply the distillation loss.

Later on, the pair of networks will become two student models for distillation. We therefore have our *first part* of self-supervised consistency loss as:

$$L_{\text{students}} = \sum_{\boldsymbol{x} \in D_U - D_{\text{trust1}}} \begin{cases} g_2(\boldsymbol{x}) \log \frac{g_2(\boldsymbol{x})}{g_1(\boldsymbol{x})}, & L_{\text{nnPU}}(\boldsymbol{x}) > \alpha g_2(\boldsymbol{x}) \log \frac{g_2(\boldsymbol{x})}{g_1(\boldsymbol{x})} \\ 0, & L_{\text{nnPU}}(\boldsymbol{x}) \leq \alpha g_2(\boldsymbol{x}) \log \frac{g_2(\boldsymbol{x})}{g_1(\boldsymbol{x})} \end{cases} + \sum_{\boldsymbol{x} \in D_U - D_{\text{trust2}}} \begin{cases} g_1(\boldsymbol{x}) \log \frac{g_1(\boldsymbol{x})}{g_2(\boldsymbol{x})}, & L_{\text{nnPU}}(\boldsymbol{x}) > \alpha g_1(\boldsymbol{x}) \log \frac{g_1(\boldsymbol{x})}{g_2(\boldsymbol{x})} \\ 0, & L_{\text{nnPU}}(\boldsymbol{x}) \leq \alpha g_1(\boldsymbol{x}) \log \frac{g_1(\boldsymbol{x})}{g_2(\boldsymbol{x})} \end{cases} \quad (15)$$

where we study the effect of choosing α in section 4.3.3. The KL Divergence between the two students is only calculated on $D_U - D_{\text{trust1}}$ and $D_U - D_{\text{trust2}}$. One reason

why we choose such design is that the accuracy on D_{trust1} and D_{trust2} discovered by the self-paced learning is much higher than the accuracy on D_U. In addition, on the D_{trust1} and D_{trust2} the prediction entropy is 0.005, while on the unlabeled set it is 0.074, which indicates much lower confidence.

3.3.2. Consistency for Moving Averaged Weights: Adding Teachers to Distill

Inspired by (Tarvainen & Valpola, 2017), in addition to the consistency between two students, we also encourage them to be consistent with their moving averaged trajectory of weights. Assume that g_1 and g_2 are parameterized by θ_1 and θ_2. For each student we introduce a new teacher model, G_1 and G_2, parameterized by Θ_1 and Θ_2 with the same structure as g_1 and g_2. The weights of G_1, G_2 are updated via the following moving average:

$$\begin{aligned}\Theta_{1,t} &= \beta\theta_{1,t-1} + (1-\beta)\theta_{1,t} \\ \Theta_{2,t} &= \beta\theta_{2,t-1} + (1-\beta)\theta_{2,t}\end{aligned} \tag{16}$$

where $\theta_{1,t}$ denotes the instance of θ_1 at time t, and similarly for others. We study the effect of β in section 4.3.4.

An mean-square-error (MSE) loss is next enforced for G_1 and G_2 to distill from g_1 and g_2, namely:

$$\begin{aligned}L_{\text{teachers}} = &\sum_{\boldsymbol{x}\in D_{\text{U}}-D_{\text{trust1}}} ||G_1(\boldsymbol{x}) - g_1(\boldsymbol{x})||^2 \\ &+ \sum_{\boldsymbol{x}\in D_{\text{U}}-D_{\text{trust2}}} ||G_2(\boldsymbol{x}) - g_2(\boldsymbol{x})||^2\end{aligned} \tag{17}$$

The above constitutes the *second part* of our self-supervised consistency cost.

In summary, the benefits of self-supervised learning for PU learning come from two folds: 1) the enlarged labeled examples (D_{trust}) introduces stronger supervision into PU learning and brings high accuracy; 2) the consistency cost between diverse student and teacher models introduces the learning stability (low variance). Eventually, our overall loss function[1] of Self-PU is:

$$L = L_{\text{SP+Reweight}} + L_{\text{students}} + L_{\text{teachers}}. \tag{18}$$

In all experiments and as shown in Figure 1, we first apply self-paced learning and self-calibrated loss reweighting in the first 50 epochs, followed by a self-distillation period from 50th to 200th epoch. That allows for the models to learn sufficient meaningful information before being distilled. After training, we compare the validation accuracies of two teacher models and select the better performer to be applied on the testing set[2].

[1] Since here we have two students of different learning paces, our $L_{\text{SP+Reweight}}$ is also extended to both D_{trust1} and D_{trust2}.

[2] Note that, here we only select one and discard the other, only for simplicity purpose. Other approaches, such as average or weighted-fusion of the two teachers models, are applicable too.

4. Experiments

4.1. Datasets

In order to evaluate our proposed "Self-PU" learning framework, we conducted experiments on two common testbeds for PU learning: MNIST, and CIFAR10; plus **a new real-world benchmark**, *i.e.* ADNI (Jack Jr et al., 2008), for the application of Alzheimer's Disease diagnosis.

4.1.1. Introduction to the ADNI database

The Alzheimer's Disease Neuroimaging Initiative (ADNI) database[3] was constructed to test whether brain scans, *e.g.* magnetic resonance imaging (MRI) and other biological markers, can be utilized to predict the early-stage Alzheimer's disease (AD), in order for more timely prevention and treatment. The dataset, especially its MRI image collection, has been widely adopted and studied for the classification of Alzheimer's disease (Khvostikov et al., 2018; Li et al., 2015). Fig. 2 shows visual examples.

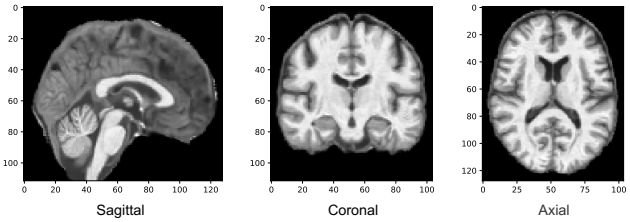

Figure 2. Cross-sectional imaging of a 104×128×112 MRI example from the ADNI dataset. Images are from the 52^{nd}, 64^{th}, 56^{th} slice of sagittal, coronal, and axial plane, respectively. An MRI image is of gray scale with a value from 0 to 255 for each voxel, and was processed by the intensity inhomogeneity correction, skull-stripping and cerebellum removing.

Traditionally, the machine learning community considers the AD diagnosis task as a binary, fully supervised classification task, between the *patient* and the *healthy* classes. It has **never** been connected to PU learning. Yet, we advocate that this task could become a new **suitable, realistic** and **challenging** application benchmark for PU learning.

The early-stage AD prediction/diagnosis is highly nontrivial for multi-fold, field-specific reasons. First, many nuance factors can heavily affect the feature effectiveness, ranging from individual patient variability to (mechanical/optical) equipment functional fluctuations, to manual operation and sensor/environment noise. Second, within the whole-brain scans, only some (not fully-specified) local brain regions are found to be indicative of AD symptoms. Third and most importantly, in contrast to the diagnosed patients, the remaining population, who are not yet clinically diagnosed with AD, *cannot* be simply treated as all *healthy*: on one hand, the above challenges of AD diagnosis

[3] http://adni.loni.usc.edu

Table 1. Specification of benchmark datasets and models.

Dataset	#Train	#Test	Input Size	π_p	Positive/Negative	Model
MNIST	60,000	10,000	28×28	0.49	Odd/Even	6-layer MLP
CIFAR10	50,000	10,000	3×32×32	0.40	Traffic tools/Animals	13-layer CNN
ADNI	822	113	104×128×112	0.43	AD Positive/Negative	3-branch 2-layer CNN

inevitably lead to incorrectly missed patient cases; on the other hand, and **more notably**, the AD patients go through a stage called *mild cognitive impairment* (MCI) (Larson et al., 2004; Duyckaerts & Hauw, 1997), a critical transition period between the expected cognitive decline of normal aging, and the severe decline of true dementia. During the MCI stage, those people were "clinically" considered as AD patients already (if diagnosed with more intrusive biochemical means); however, no symptom is known to be observable in current MRI images or other bio-markers.

In other words, the MCI examples have definitely been included in the currently "healthy"-labeled samples in ADNI, while they should have belonged to the "patient" class. In training, we label patients as **positive** class, the "healthy"-labeled examples can then be considered as **unlabeled** class, which mixes the true healthy people (*i.e.*, from the actual **negative** class) and the MCI people (*i.e.*, from the **positive** class). We communicated with several seasoned medical doctors practicing in AD fields, and **they unanimously agreed** that AD diagnosis should be described as a PU learning problem rather than (the traditional treatment as) a binary classification problem. In this paper, we study the specific setting of MRI image classification task on the ADNI dataset, while other bio-marker classification can be similarly studied in PU settings.

4.1.2. Dataset Setting

We report our dataset protocols towards PU learning. More metadata are summarized in Table 1.

- **MNIST**: odd numbers 1, 3, 5, 7, 9 form the positive class while even numbers 0, 2, 4, 6, 8 form the negative.
- **CIFAR10**: four vehicles classes ("airplane", "automobile", "ship", "truck") constitute the positive class, and six animal classes ("bird", "cat", "deer", "dog", "frog" "horse") constitute the negative.
- **ADNI**: We utilized the public ADNI data set as (Li et al., 2015; Yuan et al., 2018) suggested: The T1-weighted MRI images were processed by first correcting the intensity inhomogeneity, followed by skull-stripping and cerebellum removing. We consider the subjects as positive class if they: 1) have positive clinical diagnosis records on file; or 2) have their standardized uptake value ratio (SUVR) values [4] no less than 1.08 (Villeneuve et al., 2015; Ott et al., 2017; Yuan et al., 2018). There are no missing labels in all data we used.

Following the convention of nnPU (Kiryo et al., 2017), we use $n_p = |D_P| = 1000$ in MNIST and CIFAR10.In ADNI, we end up with $n_p = 113$. $n_u = |D_U|$ equals the size of remaining training data on all three datasets.

4.2. Baselines and Implementations

Following nnPU (Kiryo et al., 2017), we use a 6-layer *multilayer perceptron* (MLP) with ReLU on MNIST. On CIFAR10, we use a 13-layer CNN with ReLU. We designed a multi-scale network for ADNI, which is used as the backbone for all compared baselines: please see supplementary materials for details. We use Adam optimizer with a cosine annealing learning rate scheduler for training. The batch size is 256 for MNIST and CIFAR10, and 64 for ADNI. The γ is set to $\frac{1}{16}$.

For fair comparison, all our experiments are each run five times, and the mean and standard deviations of accuracies are reported. All codes are implemented in PyTorch and **will be released upon paper acceptance**.

4.3. Ablation Study

In this section, we carry out a thorough ablation study, on the key components introduced during the self-paced learning stage (*i.e.*, the selection of trusted set) and the self-supervised distillation stage (*i.e.*, the diversity of students, the effect of hard sample mining when training students, and the effect of weight averaging further by teachers). All experiments are conducted on the **CIFAR10** dataset[5]. We will study the effect of γ in the supplementary materials.

4.3.1. Selection of "Trusted Set" D_{TRUST}

Since self-paced learning aims to mine more confident positive/negative examples, it is crucial to ensure the trustworthiness of the selected D_{trust}. We therefore calculate the accuracy of assigned labels for D_{trust} along the self-paced training, as an indicator of the sampling strategy reliability.

We compare three settings: 1) *"Fixed sampling size"*: each time, the model selects a fixed number of samples (*e.g.* 25% of examples in D_U), assigning soft labels and adding them into D_{trust}. Meanwhile, low-confidence samples in

[4] SUVR is a therapy monitoring or response, considered as an important indicator of Alzheimer's Disease.

[5] To conduct controlled experiments we disable the self-calibration strategy in Table 3, 4, 5

Table 2. Classification comparison on CIFAR10: we report both **means** and **standard deviations** (in parentheses) from five runs. L_{SP}: self-paced training. $L_{\text{SP + Reweighting}}$: self-paced training with self-calibrated loss reweighting in section 3.2. L_{students}: self-distillation from a pair of students. L_{teacher}: self-distillation from teacher networks. Self-PU: $L_{\text{SP + Reweighting}} + L_{\text{students}} + L_{\text{teacher}}$ + Soft Label

Method	CIFAR10 %
nnPU (baseline)	88.60 (0.40)
L_{SP} (fixed size)	88.05 (0.59)
L_{SP} (w.o. replacement)	88.27 (0.43)
L_{SP}	88.66 (0.40)
L_{SP} + Soft Label	88.75 (0.27)
$L_{\text{SP + Reweighting}}$	89.25 (0.42)
$L_{\text{SP + Reweighting}}$ + Soft Label	89.39 (0.36)
$L_{\text{SP}} + L_{\text{students}}$	88.84 (0.36)
$L_{\text{SP}} + L_{\text{students}}$ + Soft Label	88.93 (0.28)
$L_{\text{SP}} + L_{\text{students}} + L_{\text{teacher}}$	89.43 (0.42)
$L_{\text{SP}} + L_{\text{students}} + L_{\text{teacher}}$ + Soft Label	89.65 (0.33)
Self-PU	**89.68 (0.22)**

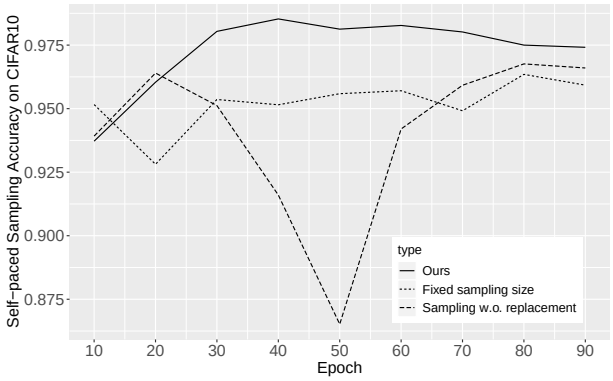

Figure 3. Accuracy of selected confident examples during self-paced learning. We compared self-paced learning with three different sampling settings: fixed sampling size (dot line), sampling without replacement (dash line), and our proposed dynamic "in-and-out" sampling (solid line). It is clear that self-paced learning with fixed sampling size or without replacement suffers from low sampling accuracy, and no-replacement is even jeopardized by the inaccurate examples remain in the D_{trust}.

D_{trust} will also be removed in next round of selection. 2) *"Sampling without replacement"*: each example selected by model will permanently reside in D_{trust}. Here the size of D_{trust} is linearly increased along the training progress. 3) Our *default approach* in Self-PU: both "Dynamic Rate Sampling" and "In-and-Out Trusted Set" are enabled. All three settings end up with $|D_{\text{trust}}| = 0.25|D_U|$.

From Figure 3, we clearly see that sampling either with a fixed size or without replacement results in a less reliable selection of D_{trust}, compared to our strategy. Moreover, the inaccurately selected examples in D_{trust} will further cause much unstable training (dash line). We demonstrate that both "Dynamic Rate Sampling" and "In-and-Out Trusted Set" are vital to achieving an accurate and stable self-paced learning (solid line). Table 2 shows the final test accuracies of three settings, where our proposed self-paced learning pipeline (L_{SP}) significantly outperforms the other two settings (L_{SP} of fixed sampling size and sampling without replacement). The better accuracy and lower variance show the advantage of our strategy.

4.3.2. Effects of Student Diversity

Different learning paces enable the diversity of two students and thus make the collaborative teaching between two students effective. Therefore we study how the student diversity, *i.e.* combination of their different learning paces, can affect the final results. Table 3 considers three different pace pairs. For example, Pace1 "10%" means that the self-paced learning of the first student model will end up with $|D_{\text{trust}}| = 0.1|D_U|$, and all students will complete the sampling for self-paced learning within the same number of training epochs. Table 3 shows that, while student diversity helps ("20%" + "30%" > "25%" + "25%"), too large student pace discrepancy will hurt the learning too ("20%" + "30%" > "10%" + "40%"). Students with very different paces are harmful because a large gap in two learning paces results in a smaller intersection-over-union of D_{trust1} and D_{trust2}. It is difficult to keep consistency between two models trained with different amounts of labeled data. Therefore, it is important to keep diversity, while not too extreme.

4.3.3. Effects of Sample Mining Threshold

L_{students} takes the hard sample mining threshold α as an important hyperparameter: the smaller α is, the more examples are counted in computing the KL divergence loss, which implies stronger self-supervision consistency between two students. Table 4 shows that a moderate α = 10 leads to the best performance. Understandably, either "under-mining" (α = 20) or "over-mining" (α = 5) hurts the performance: the former is not sufficiently regularized, while the latter starts to eliminate the emphasis over hard examples.

4.3.4. Effects of Smoothing Coefficient β

The smoothing coefficient β controls how "conservative" we distill the teachers from the students: the larger β is, the more reluctant the teacher models get updated from the students. Table 5 investigate three different β values: similar to the previous experiment on α, β also favors a reasonably moderate value, while either overly large or small β, corresponding to "over-smoothening" and "under-smoothening" during the distillation from students to teachers respectively, degrades the final performance.

Table 3. Study of student diversity (learning paces) for two-student distillation on CIFAR10 dataset. Pace1/Pace2 denotes the final ratio of $|D_{\text{trust}}|$ over $|D_U|$.

Pace1	Pace2	Test Accuracy %
10%	40%	89.32 (0.36)
15%	35%	89.55 (0.46)
20%	**30%**	**89.65 (0.33)**
25%	25%	89.64 (0.47)

Table 4. Study of hard sample mining threshold α for two-student distillation on CIFAR10 dataset. Smaller α indicates stronger distillation (Eq. (15))

α	Test Accuracy %
5	89.59 (0.39)
10	**89.65 (0.33)**
20	89.38 (0.51)

Table 5. Study of smoothing coefficient β for teacher networks on CIFAR10 dataset. Greater β indicates slower updates of teachers from the students (Eq. (17))

β	Test Accuracy %
0.2	89.37 (0.39)
0.3	**89.65 (0.33)**
0.4	89.47 (0.41)

Table 6. Classification comparison on MNIST and CIFAR10. "*" indicates that 3,000 positive examples were initialized for training, while others used 1,000.

Method	MNIST %	CIFAR10 %
uPU (Du Plessis et al., 2014)	92.52 (0.39)	88.00 (0.62)
nnPU (Kiryo et al., 2017)	93.41 (0.20)	88.60 (0.40)
DAN* (Liu et al., 2019)	-	89.7 (0.40)
Self-PU	94.21 (0.54)	89.68 (0.22)
Self-PU*	**96.00 (0.29)**	**90.77 (0.21)**

Table 7. Classification accuracies of different methods on ADNI. "naive" means that we treat the entire unlabeled class as negative.

Method	ADNI %
naive	73.27 (1.45)
uPU	73.45 (1.77)
nnPU	75.96 (1.42)
Self-PU	**79.50 (1.80)**

4.3.5. Effect of Teacher and Students

We verify the effect of two types of distillation in our self-supervised learning: mutually between L_{students}, and by L_{teachers}. In Table 2, distillation from two students with different learning paces (L_{students}) improve the accuracy of nnPU baseline from 88.60% to 88.84% on CIFAR10. By adding two teachers for self-distillation, the performance is further boosted to 89.43%, which endorses the complementary power of two types of self-distillation.

4.4. Comparison to State-of-the-Art Methods

We compare the performance of the proposed Self-PU with several popular baselines: the unbiased PU learning (uPU) (Du Plessis et al., 2014); the non-negative PU learning (nnPU) (Kiryo et al., 2017)[6], and DAN (a latest GAN-based PU method) (Liu et al., 2019).

Table 6 summarizes the comparison results on MNIST and CIFAR-10. On MNIST, Self-PU outperforms uPU and nnPU by over 0.5%, setting new performance records. On CIFAR-10, Self-PU surpasses nnPU by over 1% (a considerable gap). More importantly, by only leveraging 1000 positive examples, Self-PU achieves comparable performance as DAN where 3000 positive samples were used. Training with 3000 positive examples further boosts our performance which outperforms DAN by 1%.

Our "Self-PU" achieved not only high accuracy, but more importantly a **much more stable PU learning process** (Figure 4). As noted in (Kiryo et al., 2017), uPU suffers from overfitting with complex models. We also empirically found a similar phenomenon in PU learning with nnPU risk estimator, where the validation accuracy remains unstable and even drops in the late training stage. However, the training process of our "Self-PU" is significantly more stable than uPU and nnPU. This training stability benefits from both accurately identified examples in self-paced training and prediction consistency forced by our self-supervised distillations.

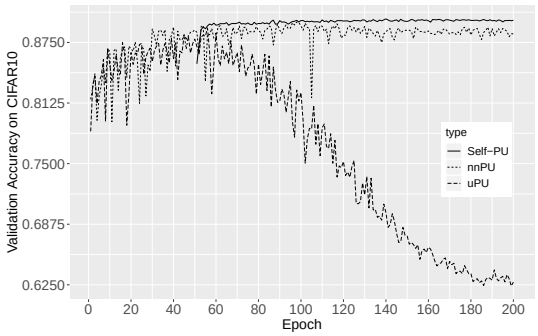

Figure 4. Validation accuracy during training on CIFAR10. Our "Self-PU" framework[8]achieved a more stable training compared with uPU and nnPU methods.

Finally, we compare on the more complex real-word ADNI data in Table 7. We first run a naive fully supervised classification baseline, by treating the entire **unlabeled** class as **negative**. Its accuracy is much inferior to our PU learning results, validating our PU formulation of the ADNI task. Next, Self-PU gains significantly over uPU and nnPU, showing highly promising performance on ADNI and setting new state-of-the-arts. Our sophisticated building blocks seem

[6]We reproduced the uPU and nnPU baselines using the official codebase from: https://github.com/kiryor/nnPUlearning

[8]Since in "Self-PU" we use the teacher model G for the final prediction, the solid line shows the accuracy of G starts from epoch 50 when the self-paced training ends.

to add robustness to handling the real-world data variations and challenges. Furthermore, it also reminds that conventional PU benchmarks like CIFAR-10 and MNIST may have been over-simplified and therefore saturated (as they already did in image classification): now is the time to pay more attention to more challenging PU learning benchmarks.

5. Conclusion

We proposed Self-PU, that bridges self-training strategy into PU learning for the first time. It leverages both the self-paced selected set of trusted samples and the consistency supervision via self-distillation and self-calibration. Experiments report state-of-the-art performance of Self-PU on two conventional (and potentially oversimplified) benchmarks, plus our newly introduced real-world PU testbed of ADNI classification. Our future work will explore more realistic PU learning setting, which we believe will motivate new algorithmic findings.

References

Armenian, H. and Lilienfeld, A. The distribution of incubation periods of neoplastic diseases. *American journal of epidemiology*, 1974.

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

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. Curriculum learning. In *ICML*, pp. 41–48, 2009.

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

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

Duyckaerts, C. and Hauw, J.-J. Diagnosis and staging of alzheimer disease. *Neurobiology of aging*, 18(4):S33–S42, 1997.

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

Hou, M., Chaib-draa, B., Li, C., and Zhao, Q. Generative adversarial positive-unlabelled learning. *IJCAI*, Jul 2018. doi: 10.24963/ijcai.2018/312.

Jack Jr, C., Bernstein, M., Fox, N., Thompson, P., Alexander, G., Harvey, D., Borowski, B., Britson, P., Whitwell, J., and Ward, C. The alzheimer's disease neuroimaging initiative (adni): Mri methods. *Journal of Magnetic Resonance Imaging*, pp. 685–691, 2008.

Khan, F., Mutlu, B., and Zhu, J. How do humans teach: On curriculum learning and teaching dimension. In *NeurIPS*, pp. 1449–1457, 2011.

Khvostikov, A., Aderghal, K., Benois-Pineau, J., Krylov, A., and Catheline, G. 3d cnn-based classification using smri and md-dti images for alzheimer disease studies. *arXiv preprint arXiv:1801.05968*, 2018.

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

Kumar, M. P., Packer, B., and Koller, D. Self-paced learning for latent variable models. In *NeurIPS*. 2010.

Laine, S. and Aila, T. Temporal ensembling for semi-supervised learning. *arXiv:1610.02242*, 2016.

Larson, E. B., Shadlen, M.-F., Wang, L., McCormick, W. C., Bowen, J. D., Teri, L., and Kukull, W. A. Survival after initial diagnosis of alzheimer disease. *Annals of internal medicine*, 140(7):501–509, 2004.

Li, F., Tran, L., Thung, K.-H., Ji, S., Shen, D., and Li, J. A robust deep model for improved classification of ad/mci patients. *Biomedical Health Informatics*, 2015.

Li, X. and Liu, B. Learning to classify texts using positive and unlabeled data. In *IJCAI*, 2003.

Liu, B., Lee, W. S., Yu, P. S., and Li, X. Partially supervised classification of text documents. In *ICML*, 2002.

Liu, F., Chen, H., and Wu, H. Discriminative adversarial networks for positive-unlabeled learning. *arXiv:1906.00642*, 2019.

Ott, B. R., Jones, R. N., Noto, R. B., Yoo, D. C., Snyder, P. J., Bernier, J. N., Carr, D. B., and Roe, C. M. Brain amyloid in preclinical alzheimer's disease is associated with increased driving risk. *Alzheimer's & Dementia: Diagnosis, Assessment & Disease Monitoring*, 6:136–142, 2017.

Ren, M., Zeng, W., Yang, B., and Urtasun, R. Learning to reweight examples for robust deep learning. *arXiv preprint arXiv:1803.09050*, 2018.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. *CVPR*, Jun 2016. doi: 10.1109/cvpr.2016.308.

Tarvainen, A. and Valpola, H. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NeurIPS*, 2017.

Villeneuve, S., Rabinovici, G. D., Cohn-Sheehy, B. I., Madison, C., Ayakta, N., Ghosh, P. M., La Joie, R., Arthur-Bentil, S. K., Vogel, J. W., Marks, S. M., et al. Existing pittsburgh compound-b positron emission tomography thresholds are too high: statistical and pathological evaluation. *Brain*, 138(7):2020–2033, 2015.

Xu, M., Li, B., Niu, G., Han, B., and Sugiyama, M. Revisiting sample selection approach to positive-unlabeled learning: Turning unlabeled data into positive rather than negative. 2019.

Yuan, Y., Wang, Z., Lee, W., Thiyyagura, P., Reiman, E. M., and Chen, K. Feasibility of quantifying amyloid burden using volumetric mri data: Preliminary findings based on the deep learning 3d convolutional neural network approach. *Alzheimer's & Dementia: The Journal of the Alzheimer's Association*, 14(7):P695, 2018.

Zhang, Y., Xiang, T., Hospedales, T. M., and Lu, H. Deep mutual learning. In *ICCV*, 2018.