

CONTRASTIVE POSITIVE UNLABELED LEARNING

Anonymous authors

Paper under double-blind review

ABSTRACT

Positive Unlabeled(PU) learning refers to the task of learning a binary classifier given a few labeled positive samples, and a set of unlabeled samples (which could be positive or negative). Majority of the existing approaches rely on additional knowledge of the class prior, which is unavailable in practice. Furthermore, these methods tend to perform poorly in low-data regimes, especially when very few positive examples are labeled. In this paper, we propose a novel PU learning framework that overcomes these limitations. We start by learning a feature space through pretext-invariant representation learning and then apply pseudo-labeling to the unlabeled examples, leveraging the cluster-preserving property of the representation space. Overall, our proposed PU learning framework handily outperforms state-of-the-art PU learning methods across several standard PU benchmark datasets, while not requiring a-priori knowledge or estimate of class prior. Remarkably, our method remains effective even when labeled data is scant, where previous PU learning algorithms falter. We also provide simple theoretical analysis motivating our proposed algorithms.

1 INTRODUCTION

This paper studies classical Positive Unlabeled (PU) learning – the *weakly supervised task of learning a binary (positive vs negative) classifier in the absence of any explicitly labeled negative examples, i.e., using an incomplete set of positives and a set of unlabeled samples*.

This setting is frequently encountered in several real-world applications, especially where obtaining negative samples is either resource-intensive or impractical. For instance, consider *personalized recommendation systems*, the training data typically consists of recorded user interactions, such as the items shown to the user and the items they clicked on. While the clicked items are considered positive preferences, the items not clicked on cannot be assumed to be negatives and should be treated as unlabeled, along with items that were not shown to the user (Naumov et al., 2019; Chen et al., 2021; Kelly & Teevan, 2003). Similarly, PU learning has also found applications in diverse domains such as drug, gene, and protein identification (Yang et al., 2012; Elkan & Noto, 2008), anomaly detection (Blanchard et al., 2010), fake news detection (Ren et al., 2014), matrix completion (Hsieh et al., 2015), data imputation (Denis, 1998), named entity recognition (NER) (Peng et al., 2019) and face recognition (Kato et al., 2018) among others.

PU Learning can also be viewed as a particular instance of *learning with class dependent label noise*. However, due to the unavailability of negative examples, *statistically consistent unbiased risk estimation is generally infeasible*, without imposing strong structural assumptions on $p(x)$ (Blanchard et al., 2010; Lopuhaa et al., 1991; Natarajan et al., 2013). In fact, we show that no robust ERM estimator can solve the equivalent class-dependent label noise problem reliably unless certain dataset conditions are met (see Appendix A.3.2, Lemma 1). The milestone is (Elkan & Noto, 2008), which additionally assumes a-priori knowledge of **class prior** $\pi_p = p(y=1|x)$ and treats the unlabeled examples as a mixture of positives and negatives: $p(x) = \pi_p p(x|y=1) + (1 - \pi_p)p(x|y=0)$. (Blanchard et al., 2010; Du Plessis et al., 2014) build on this idea, and develop *statistically consistent and unbiased risk estimators* to perform cost-sensitive learning which has become the backbone of modern large scale PU learning algorithms (Chen et al., 2020d; Garg et al., 2021).

Unfortunately, these approaches suffer from two major issues (Also see Appendix A.3.1):

- **Knowledge of class prior (π_p):** Firstly, in practice, the true π_p is often not available and needs to be estimated $\hat{\pi}_p$ from the data. Further, these estimators are highly sensitive to the $\hat{\pi}_p$ - resulting in

significantly inferior performance when the estimate is inaccurate (Kiryo et al., 2017; Chen et al., 2020a).¹ This can be attributed to the fact that the estimation error $\|\pi_p - \hat{\pi}_p\| \leq \epsilon$ introduces bias $\sim \mathcal{O}(\epsilon)$ in the risk estimation. Thus, π_p is often estimated via a separate Mixture Proportion Estimation (MPE) (Ramaswamy et al., 2016; Ivanov, 2020; Yao et al., 2021) sub-routine, adding significant computational overhead.

- **Limited supervision:** Moreover, even when the oracle class prior is available, if the supervision is limited, i.e., only a handful of n_P positive examples are labeled, existing PU learning approaches can suffer from significant drop in performance or even complete collapse (Chen et al., 2020a). This is primarily due to the increased variance in risk estimation, which scales as $\sim \mathcal{O}(1/n_P)$.

To this end, the primary aim of this work is to *develop a parameter-free approach that facilitates Positive Unlabeled (PU) learning, even in scenarios where the availability of labeled examples is limited, without requiring any additional side information, such as class prior.*

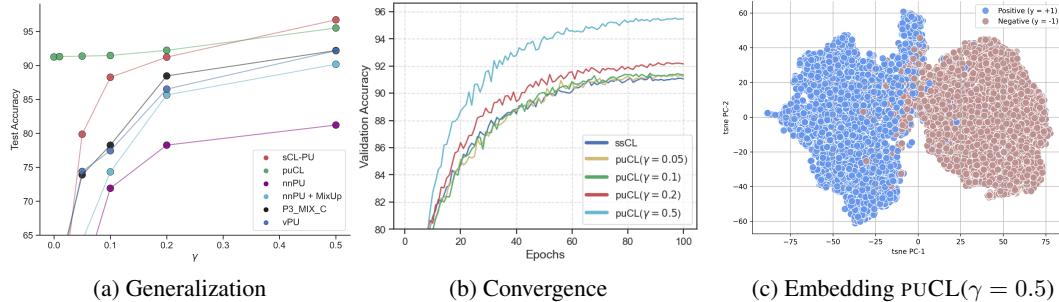


Figure 1: Incorporating supervision: In this experiment we train a ResNet-18 on **ImageWoof** vs **ImageNette** - two subsets of ImageNet-1k widely used in noisy label learning research <https://github.com/fastai/imagenette>. Amount of supervision is measured with fraction of positive to unlabeled samples $\gamma = \frac{n_P}{n_U}$. We keep the total number of samples $N = n_P + n_U$ fixed, while varying n_P . Observe that: **(a) Generalization:** PUCL remains robust across various levels of supervision, consistently outperforming its unsupervised counterpart, ssCL, and competing effectively with SCL-PU even in high supervision regimes. In contrast, SCL-PU experiences significant performance degradation, particularly in low-supervision regimes, where PUCL continues to demonstrate unique effectiveness. **(b) Convergence:** Incorporating additional positive also improves the convergence of PUCL over ssCL (Theorem 4). **(c) Embedding quality:** Finally, models that are trained with more PU supervision yields representation manifolds exhibiting better linear separability.

Orthogonal to the existing approaches, we introduce a novel framework that deftly leverages *semantic similarity* among samples, along with the available weak supervision to learn a representation space that exhibit linear separability, and develop that into an end-to-end method that remains effective even in low-data regime while obviating the need for a-priori knowledge or estimate of class prior.

Our proposed PU Learning framework, as outlined in Algorithm 1, involves two key steps:

- *Learning a cluster preserving representation manifold i.e. a feature space that preserves the underlying clusters by mapping semantically similar examples close to each other.* In particular, we adopt a simple PU specific modification of the standard self-supervised contrastive objective to take into account the available weak supervision in form of labeled positives resulting in significantly improved representations compared to the self-supervised counterpart.
- *Assign pseudo-labels to the unlabeled examples by exploiting the geometry of the representation manifold learnt in the previous step.* These pseudo-labels are then used to train the downstream linear classifier using ordinary supervised objective e.g. cross-entropy (CE).

Contributions. Our proposed approach can be summarized into the following key contributions:

¹For example, consider $\pi_p \neq \hat{\pi}_p = 1$ which leads to generate solution i.e. all the examples wrongly being predicted as positives (Chen et al., 2020a).

- In standard self-supervised contrastive learning approaches (Gutmann & Hyvärinen, 2010; Chen et al., 2020b), the loss function encourages similarity between samples and their augmentations, and dissimilarity between pairs of samples. We adopt a simple modification of this idea for the PU setting by including an additional similarity term for pairs of samples that are both labeled; we call our method PUCL and formally describe it in Algorithm 1(A). To the best of our knowledge, this paper represents the first work tailoring contrastive learning specifically to the PU setting.
- We compare PUCL to two natural baselines²: self-supervised contrastive learning SSCL which ignores the positive labels, and an adaptation of supervised contrastive learning SCL where all the unlabeled samples are treated as negatives. We show that the relative performances of these methods depend on the number of labeled positives, and that PUCL significantly outperforms the baselines, especially in settings where the number of labeled positives is small.
- We *theoretically ground* our empirical findings by providing a bias-variance justification, which provides more insight into the behavior of different contrastive objectives under various PU learning scenarios; see Section 2.2 for more details.
- Next, we develop a clever pseudo-labeling mechanism PUPL; that operates on the representation space learnt via PUCL. The key idea is to perform *clustering on the representation manifold*, where we additionally leverage the representations of the labeled positive examples to guide the cluster assignments as outlined in Algorithm 1(B). Theoretically, our algorithm enjoys $\mathcal{O}(1)$ multiplicative error compared to optimal clustering under mild assumption. It is worth noting that, due to judicious initialization, PUPL yields improved constant factor compared to kMeans++ (Theorem 3).
- Since, even when available labeled positives are limited, representation learning is possible via proposed PUCL, PUPL is able to produce high quality pseudo-labels. Thus, our overall approach of *contrastive pretraining followed by pseudo-labeling* enables PU learning even when only a handful of labeled examples are available, a realistic setting where existing approaches often fail.
- Extensive experiments across several standard PU learning benchmark data sets reveal that our approach results in significant improvement in generalization performance compared to existing PU learning methods with $\sim 2\%$ improvement over current SOTA averaged over six benchmark data sets demonstrating the value of our approach.

2 CONTRASTIVE APPROACH TO PU LEARNING

2.1 PROBLEM SETUP

Let $x \in \mathbb{R}^d$ and $y \in Y = \{0, 1\}$ be the underlying input (i.e., feature) and output (label) random variables respectively and let $p(x, y)$ denote the true underlying joint density of (x, y) . Then, a PU training dataset is composed of a set \mathcal{X}_P of n_P positively labeled samples and a set \mathcal{X}_U of n_U unlabeled samples (a mixture of both positives and negatives)³ i.e.

$$\mathcal{X}_{PU} = \mathcal{X}_P \cup \mathcal{X}_U, \quad \mathcal{X}_P = \{\mathbf{x}_i^P\}_{i=1}^{n_P} \stackrel{\text{i.i.d.}}{\sim} p(x|y=1), \quad \mathcal{X}_U = \{\mathbf{x}_i^U\}_{i=1}^{n_U} \stackrel{\text{i.i.d.}}{\sim} p(x) \quad (1)$$

Without the loss of generality, throughout the paper we assume that the overall classifier $f(\mathbf{x}) : \mathbb{R}^d \rightarrow \mathbb{R}^{|Y|}$ is parameterized in terms of (a) an encoder $g_B(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^k$ – a mapping function from the feature space to a lower dimensional manifold referred to as the *representation manifold* hereafter; and (b) a linear layer $v_v(\cdot) : \mathbb{R}^k \rightarrow \mathbb{R}^{|Y|}$ i.e.

$$f_{v,B}(\mathbf{x}) = v_v \circ g_B(\mathbf{x}) = \mathbf{v}^T g_B(\mathbf{x}), \quad \forall \mathbf{x} \in \mathbb{R}^d. \quad (2)$$

The goal in PU learning is thus to train a binary classifier $f_{v,B}(\mathbf{x})$ from \mathcal{X}_{PU} (1).

As discussed before, our proposed PU learning framework involves two key steps - (a) learning a mapping function $g_B(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^k$ to a cluster-preserving representation space via contrastive learning and (b) exploit the geometry of the feature space to pseudo-label the representations, used to train the subsequent linear layer $v_v(\cdot) : \mathbb{R}^k \rightarrow \mathbb{R}^{|Y|}$. In the rest of this section, we discuss these two ideas in more detail, explore different design choices and systematically develop the framework.

²Additionally, in Appendix A.5.4 we discuss some recent weakly supervised contrastive approaches.

³This particular setup of how PU learning dataset is generated is referred to as the case-control setting (Bekker et al., 2019; Blanchard et al., 2010). However, in Appendix A.3.1 we also experiment with different contrastive objectives in the single-dataset setting (Bekker & Davis, 2020).

2.2 REPRESENTATION LEARNING FROM POSITIVE UNLABELED DATA

Central to our approach is the construction of a representation space that fosters the proximity of semantically related instances while enforcing the separation of dissimilar ones. One way to obtain such a representation space via pretext-invariant representation learning where the representations $\mathbf{z}_i = g_{\mathbf{B}}(\mathbf{x}_i) \in \mathbb{R}^k$ are trained to be invariant to label-preserving distortions (Wu et al., 2018; Misra & Maaten, 2020)⁴. To prevent trivial solutions (Tian et al., 2021), a popular trick is to apply additional repulsive force between the embeddings of semantically dissimilar images, known as contrastive learning (Chopra et al., 2005; Schroff et al., 2015; Sohn, 2016). In particular, we study variants of InfoNCE family of losses (Oord et al., 2018) – a popular contrastive objective based on the idea of *Noise Contrastive Estimation* (NCE), a method of estimating the likelihood of a model by comparing it to a set of noise samples (Gutmann & Hyvärinen, 2010):

$$\mathcal{L}_{\text{CL}}^* = \mathbb{E}_{(\mathbf{x}_i, y_i) \sim p(\mathbf{x}, \mathbf{y})} \left[\mathbb{E}_{\substack{\mathbf{x}_j \sim p(\mathbf{x}|y_j=y_i) \\ \{\mathbf{x}_k\}_{k=1}^N \sim p(\mathbf{x}|y_k \neq y_i)}} \left[\mathbf{z}_i \cdot \mathbf{z}_j - \log \left(\exp(\mathbf{z}_i \cdot \mathbf{z}_j) + \sum_{k=1}^N \exp(\mathbf{z}_i \cdot \mathbf{z}_k) \right) \right] \right], \quad (3)$$

Where, the operator \cdot is defined as: $\mathbf{z}_i \cdot \mathbf{z}_j = \frac{1}{\tau} \frac{\mathbf{z}_i^T \mathbf{z}_j}{\|\mathbf{z}_i\| \|\mathbf{z}_j\|}$. Intuitively, the loss projects the representation vectors onto hypersphere $\mathcal{S}_1^{k-1} = \{\mathbf{z} \in \mathbb{R}^k : \|\mathbf{z}\| = \frac{1}{\tau}\}$ and aims to minimize the angular distance between similar samples while maximizing the angular distance between dissimilar ones. $\tau \in \mathbb{R}^+$ is a hyper-parameter that balances the spread of the representations on the hypersphere (Wang & Isola, 2020). The objective aims to minimize the angular distance between similar samples while maximizing the angular distance between dissimilar ones.

While, several frameworks have been proposed to realize the infoNCE family of losses in the finite sample setting (Caron et al., 2020; Grill et al., 2020; He et al., 2020; Zbontar et al., 2021), in this paper we adopt the SimCLR framework (Chen et al., 2020d).

Self Supervised Contrastive Learning (ssCL): In the unsupervised setting, since identifying similar and dissimilar example pairs from the appropriate class conditionals is intractable; different augmentations of the same image are treated as similar, while rest are considered as dissimilar pairs. In particular, for any random batch of samples $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^b$, corresponding *multi-viewed batch* is constructed by obtaining two augmentation (correlated views) of each sample: $\tilde{\mathcal{D}} = \{t(\mathbf{x}_i), t'(\mathbf{x}_i)\}_{i=1}^b$ where $t(\cdot), t'(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^d$ are stochastic label preserving transformations, such as color distortion, cropping, flipping etc. To facilitate the subsequent discussion, let we introduce $\mathbb{I} \equiv \{1, \dots, 2b\}$ corresponding to the elements of the multi-viewed batch. For augmentation indexed $i \in \mathbb{I}$, other augmentation originating from the same source sample is indexed as $a(i)$. Then, ssCL minimizes the following objective (Chen et al., 2020b) :

$$\mathcal{L}_{\text{ssCL}} = -\frac{1}{|\mathbb{I}|} \sum_{i \in \mathbb{I}} \left[\mathbf{z}_i \cdot \mathbf{z}_{a(i)} - \log Z(\mathbf{z}_i) \right] \quad (4)$$

Where, $Z(\mathbf{z}_i) = \sum_{j \in \mathbb{I}} \mathbf{1}(j \neq i) \exp(\mathbf{z}_i \cdot \mathbf{z}_j)$ is the *finite-sample approximation of the partition function within the batch*. In practice, rather than computing the loss over the encoder outputs i.e. $\mathbf{z}_i = g_{\mathbf{B}}(\mathbf{x}_i)$; it is beneficial to feed it through a small *nonlinear projection network* $h_{\Gamma}(\cdot) : \mathbb{R}^k \rightarrow \mathbb{R}^p$ to obtain a lower dimensional representation $\mathbf{z}_i = h_{\Gamma} \circ g_{\mathbf{B}}(\mathbf{x}_i) \in \mathbb{R}^p$ (Chen et al., 2020b; Schroff et al., 2015). Note that *the $h_{\Gamma}(\cdot)$ is only used during training and discarded during inference*.

2.2.1 INCORPORATING PU SUPERVISION

Despite its ability to learn robust representations, ssCL is entirely agnostic to semantic annotations, hindering its ability to benefit from additional supervision, especially when such supervision is reliable. This lack of semantic guidance often leads to inferior visual representations compared to fully supervised approaches (He et al., 2020; Kolesnikov et al., 2019).

⁴Consider dataset $\mathbf{x} \in \mathcal{X}, y \in Y$ with underlying ground-truth labeling mechanism $y = \mathcal{Y}(\mathbf{x}) \in Y$. Parameterized representation function $f_{\mathbf{W}}(\cdot)$ is said to be invariant under transformation $t : \mathcal{X} \rightarrow \mathcal{X}$ that do not change the ground truth label i.e. $\mathcal{Y}(t(\mathbf{x})) = \mathcal{Y}(\mathbf{x})$ if $f_{\mathbf{W}}(t(\mathbf{x})) \approx f_{\mathbf{W}}(\mathbf{x})$.

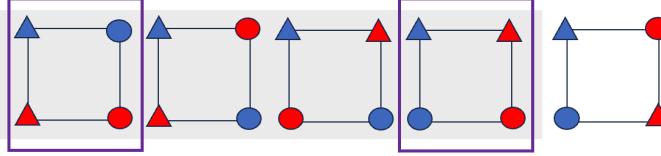


Figure 2: Geometric intuition of incorporating contrastive (PU)-supervision: Consider 1D feature space $\mathbf{x} \in \mathbb{R}$, e.g., $\mathbf{x}_i = 1$ if shape: triangle ($\blacktriangle, \triangle$), $\mathbf{x}_i = 0$ if shape: circle (\bullet, \circ). However, the labels are $y_i = 1$ if color: blue (\blacktriangle, \bullet) and $y_i = 1$ if color: red (\triangle, \circ). We show all possible configurations of arranging these points on the vertices of unit hypercube $\mathcal{H} \in \mathbb{R}^2$ when \triangle is fixed at $(0, 1)$. All the four shaded point configurations are favored by ssCL (4), since $\mathbf{x}_i = \mathbf{x}_j$ are placed neighboring vertices. However, the minimum loss configurations of PUCL (marked in rectangle) favors configurations, that additionally also preserve annotation consistency.

Motivated by these observations, *our goal is to design a contrastive loss that can leverage the available weak supervision in PU learning (in the form of labeled positive examples) in an efficient manner; to learn more discriminative representations compared to $\mathcal{L}_{\text{ssCL}}$.*

Positive Unlabeled Supervised Contrastive Learning (SCL-PU): In the fully supervised setting, Supervised Contrastive Learning (SCL) (Khosla et al., 2020) addresses this issue by utilizing the semantic annotations to guide the choice of similar and dissimilar pairs, often resulting in significant empirical gains. However, unfortunately in PU learning, *since negative examples are not available, it is non-trivial to extend SCL in this setting*. Consider the naive *disambiguation-free* (Li et al., 2022) adaptation, we refer as SCL-PU – wherein, the unlabeled training examples are treated as *pseudo-negative* instances⁵ i.e.

$$\mathcal{L}_{\text{SCL-PU}} = -\frac{1}{|\mathbb{U}|} \sum_{i \in \mathbb{U}} \left[\left(\mathbf{1}(i \in \mathbb{P}) \frac{1}{|\mathbb{P} \setminus i|} \sum_{j \in \mathbb{P} \setminus i} \mathbf{z}_i \cdot \mathbf{z}_j + \mathbf{1}(i \in \mathbb{U}) \frac{1}{|\mathbb{U} \setminus i|} \sum_{j \in \mathbb{U} \setminus i} \mathbf{z}_i \cdot \mathbf{z}_j \right) - \log Z(\mathbf{z}_i) \right] \quad (5)$$

Here, \mathbb{P} and \mathbb{U} denote the subset of indices in $\tilde{\mathcal{D}}$ that are labeled and unlabeled respectively i.e. $\mathbb{P} = \{i \in \mathbb{I} : \mathbf{x}_i = \mathcal{X}_P\}$, $\mathbb{U} = \{i \in \mathbb{I} : \mathbf{x}_i \in \mathcal{X}_U\}$. It is easy to follow that this naive adaptation suffers from statistical bias to estimate $\mathcal{L}_{\text{CL}}^*$ which becomes increasingly pronounced as the level of available supervision decreases as characterized in Theorem 1.

Theorem 1. $\mathcal{L}_{\text{SCL-PU}}$ (5) is a biased estimator of $\mathcal{L}_{\text{CL}}^*$ characterized as follows:

$$\mathbb{E}_{\mathcal{X}_{\text{PU}}} \left[\mathcal{L}_{\text{SCL-PU}} \right] - \mathcal{L}_{\text{CL}}^* = \frac{\pi_p(1 - \pi_p)}{1 + \gamma} \left[2\tilde{\mu}_{\text{PN}} - (\mu_p^* + \mu_n^*) \right]$$

Here, $\mu_p^* = \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j \sim p(\mathbf{x}|y=1)} (\mathbf{z}_i \cdot \mathbf{z}_j)$ and $\mu_n^* = \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j \sim p(\mathbf{x}|y=0)} (\mathbf{z}_i \cdot \mathbf{z}_j)$ capture the proximity between samples from same class marginals and $\tilde{\mu}_{\text{PN}} = \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j \sim p(\mathbf{x}|y_i \neq y_j)} (\mathbf{z}_i \cdot \mathbf{z}_j)$ captures the proximity between dissimilar samples. $\gamma = \frac{n_p}{n_u}$ captures the proportion of cardinality of labeled to unlabeled training subset.

Consistent with the theoretical observation, our experiments (Figure 4) also reveal that, while in the low-supervision regime, $\mathcal{L}_{\text{SCL-PU}}$ might suffer from significant drop in generalization performance, it still results in significant improvements over the unsupervised $\mathcal{L}_{\text{ssCL}}$ when sufficient labeled positives are available. *indicating a bias-variance trade-off that can be further exploited to arrive at an improved loss.*

Positive Unlabeled Contrastive Learning (PUCL): In response, we consider a simple modification to the standard contrastive objective for the PU setting that is able to incorporate the available (weak) supervision judiciously. In particular, the modified objective dubbed PUCL leverages the available supervision as follows – each labeled positive anchor is attracted closer to all other labeled

⁵Note that, this is a reduction of PU Learning to learning with class-dependent noisy label with noise rates $E(\xi_P) = \frac{\pi_p}{\gamma + \pi_p}$ and $\xi_N = 0$. Refer to Appendix A.3.2 for more details.

positive samples in the batch, whereas an unlabeled anchor is only attracted to its own augmentation.

$$\mathcal{L}_{\text{PUCL}} = -\frac{1}{|\mathbb{I}|} \sum_{i \in \mathbb{I}} \left[\mathbf{1}(i \in \mathbb{U}) \left(\mathbf{z}_i \cdot \mathbf{z}_{a(i)} \right) + \mathbf{1}(i \in \mathbb{P}) \frac{1}{|\mathbb{P} \setminus i|} \sum_{j \in \mathbb{P} \setminus i} \mathbf{z}_i \cdot \mathbf{z}_j - \log Z(\mathbf{z}_i) \right] \quad (6)$$

In essence, the unsupervised part in PUCL enforces consistency between representations learned via label-preserving augmentations i.e. between \mathbf{z}_i and $\mathbf{z}_{a(i)}$ $\forall i \in \mathbb{I}$, whereas the supervised component injects structural knowledge derived from labeled positives (see Figure 2, 7, Appendix A.5.2).

It is not hard to see that, $\mathcal{L}_{\text{PUCL}}$ is an unbiased estimator of $\mathcal{L}_{\text{CL}}^*$. Additionally, by using the available labeled positives it enjoys a lower variance compared to $\mathcal{L}_{\text{SSCL}}$ and the gap is a monotonically increasing function of $\gamma = \frac{n_p}{n_u}$ as summarized in Theorem 2.

Theorem 2. Assume that $\mathbf{x}_i, \mathbf{x}_{a(i)}$ are i.i.d draws from the same class marginal (Saunshi et al., 2019; Tosh et al., 2021), then it follows that the objective functions $\mathcal{L}_{\text{SSCL}}$ (4) and $\mathcal{L}_{\text{PUCL}}$ (6) are unbiased estimators of $\mathcal{L}_{\text{CL}}^*$ (3). Additionally, it holds that:

$$\Delta_\sigma(\gamma) \geq 0 \quad \forall \gamma \geq 0 ; \quad \Delta_\sigma(\gamma_1) \geq \Delta_\sigma(\gamma_2) \quad \forall \gamma_1 \geq \gamma_2 \geq 0$$

where, $\Delta_\sigma(\gamma) = \text{Var}(\mathcal{L}_{\text{SSCL}}) - \text{Var}(\mathcal{L}_{\text{PUCL}})$.

Empirical Evidence: Our ablation experiments (see Figure 4, Table 2 and Appendix A.5.4) also suggest that indeed $\mathcal{L}_{\text{PUCL}}$ consistently produces representations that have improved linear separability over $\mathcal{L}_{\text{SSCL}}$ indicated by its improved downstream classification performance. These improvements are particularly pronounced when a sufficient number of labeled instances are available. Further, we observe that PUCL also significantly improves over SCL-PU objective especially in the practical settings where γ is usually small while staying competitive for settings where a large fraction of the data is labeled as discussed in more detail in Section 3.

2.3 POSITIVE UNLABELED PSEUDO LABELING

While so far we have only discussed about learning a representation function, mapping the input features to a contrastive representation manifold, where semantically dissimilar samples are likely to be easily separable – performing inference on this manifold is not entirely obvious. Under the standard semi-supervised setting, the linear layer $v_v(\cdot)$ can be trained using CE loss over the representations of the labeled data (Assran et al., 2020) to perform downstream inference. However, in the PU learning setting since we do not have any negative examples, a naive disambiguation-free approach would fit the bias in semantic annotations resulting in decision boundary deviation (Li et al., 2022) even for completely separable feature space (Figure 12). One natural approach would be to train the linear classifier using specialized cost-sensitive PU learning algorithm such as nnPU (Kiryo et al., 2017) – the de-facto approach to solve PU problems in practical settings and at the core of most modern PU learning algorithms. However, they suffer from the issues mentioned in Section 1.

In response, we ask the question: *Can we develop a scheme to train a downstream PU classifier over the representations learnt via contrastive pretraining, that remains effective even in extreme low-supervision regime (i.e. when only a handful of positive examples are labeled) while not requiring the knowledge (or estimate) of dataset properties such as class prior?*

To this end, we propose a clever pseudo-labeling mechanism that *obviates the need of class prior knowledge* and significantly simplifies the downstream inference problem which can now be solved using standard CE loss over the pseudo-labels – even when only a handful of labeled examples are available. Our algorithm relies on the fact that the contrastive representation manifold resulting from Algorithm 1(A) fosters proximity of semantically similar examples i.e. it is likely to preserve underlying clustering structure (Parulekar et al., 2023). In particular, we seek to find centers $C^* = \{\mu_P, \mu_N\}$ on the representation space, such that it approximately solves the NP-hard k -means problem (Mahajan et al., 2012) i.e. minimize the following potential function:

$$\phi^* = \phi(\mathcal{Z}_{\text{PU}}, C^*) = \sum_{\mathbf{x} \in \mathcal{Z}_{\text{PU}}} \min_{\mu \in C^*} \|\mathbf{x} - \mu\|^2, \quad \mathcal{Z}_{\text{PU}} = \{g_B(\mathbf{x}_i) \in \mathbb{R}^k : \mathbf{x}_i \in \mathcal{X}_{\text{PU}}\} \quad (7)$$

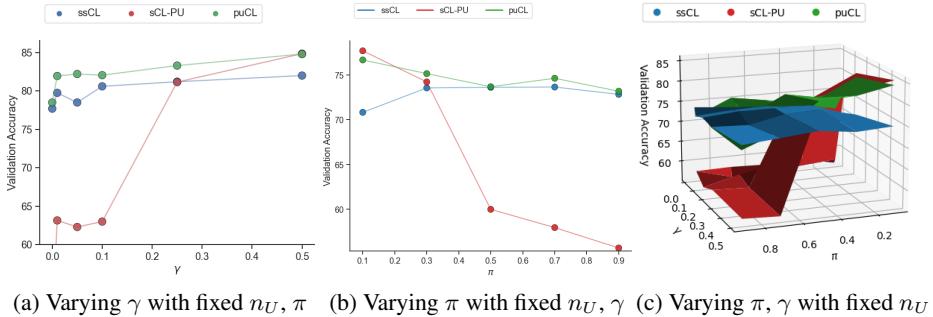


Figure 4: Ablation of Contrastive Representation Learning under different PU dataset settings: To better understand the bias-variance trade-off in PU representation learning, we experiment with different PU learning settings: class prior π_p and amount of labeled data captured by $\gamma = \frac{n_p}{n_U}$ (Theorem 4). Experiments train ResNet-34 on ImageNet Dogs vs Non-Dogs. Embedding evaluation was performed using fully supervised kNN classification. They flesh out several interesting aspects of contrastive learning over PU data and supplement our theoretical findings. Please refer to Section 3 for a detailed discussion.

are aimed at understanding this bias-variance trade-off and gain insight about the behavior of contrastive objectives in the PU setting. Our observations are summarized as follows:

- **Role of γ :** In these experiments, we fix the amount of unlabeled data n_U and π_p , and vary the number of labeled positive examples n_p , resulting in different values of $\gamma = \frac{n_p}{n_U}$. We consistently observe that PUCL outperforms ssCL across all settings. The improvements are particularly significant when γ is larger. While, for larger values of γ , SCL-PU shows accuracy gains, it suffers from performance degradation, especially for smaller values of γ - aligning with Theorem 1
- **Role of π_p :** The bias characterization of SCL-PU indicate a dependence on π_p . To understand this in isolation, we fix γ and n_U while using different π_p to create the unlabeled set i.e. $\pi_p n_U$ positives and $(1 - \pi_p) n_U$ negatives are mixed. Indeed as π_p approaches 1/2 we observe that the SCL-PU loss starts to degrade and more interestingly, as $\pi_p \rightarrow 1$ SCL-PU completely collapses. This is possibly because a large value of π_p implies that the training algorithm was presented with less amount of negative (unlabeled) examples. While the other two objectives remain fairly robust; gains of PUCL over ssCL by using available supervision is diminished.
- **Convergence:** We Find that PUCL not only has better generalization, it also enjoys uperior convergence compared to ssCL since it suffers a lower bias compared to the ideal fully supervised loss. Due to space constraint we discuss this in Figure 9, Theorem 4 in Appendix A.5.

4 CONCLUSION, LIMITATIONS AND BROADER IMPACT

In summary, we present a novel, simple and practical PU learning solution with superior empirical performance, without needing additional knowledge like class prior. Our approach uniquely stays effective even with extremely limited labels, unlike prior PU methods. Overall, by pioneering PU learning with semantic similarity through contrastive learning and pseudo-labeling, we provide a theoretically-grounded technique that opens a valuable new research direction for PU learning.

One potential limitation of our method is that it depends on contrastive learning to find cluster-preserving embedding space and might fail when it can't do so. Further, it depends on contrasting augmentations, which might be challenging to extend in some domains e.g. time series.

REFERENCES

- Anish Acharya, Abolfazl Hashemi, Prateek Jain, Sujay Sanghavi, Inderjit S Dhillon, and Ufuk Topcu. Robust training in high dimensions via block coordinate geometric median descent. In *International Conference on Artificial Intelligence and Statistics*, pp. 11145–11168. PMLR, 2022.
- David Arthur and Sergei Vassilvitskii. K-means++ the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 1027–1035, 2007.
- Yuki Markus Asano, Christian Rupprecht, and Andrea Vedaldi. Self-labelling via simultaneous clustering and representation learning. *arXiv preprint arXiv:1911.05371*, 2019.
- Mahmoud Assran, Nicolas Ballas, Lluis Castrejon, and Michael Rabbat. Supervision accelerates pre-training in contrastive semi-supervised learning of visual representations. *arXiv preprint arXiv:2006.10803*, 2020.
- Francis Bach and Zaïd Harchaoui. Diffrac: a discriminative and flexible framework for clustering. *Advances in Neural information processing systems*, 20, 2007.
- Jessa Bekker and Jesse Davis. Learning from positive and unlabeled data: A survey. *Machine Learning*, 109(4):719–760, 2020.
- Jessa Bekker, Pieter Robberechts, and Jesse Davis. Beyond the selected completely at random assumption for learning from positive and unlabeled data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 71–85. Springer, 2019.
- Gilles Blanchard, Gyemin Lee, and Clayton Scott. Semi-supervised novelty detection. *The Journal of Machine Learning Research*, 11:2973–3009, 2010.
- Matko Bošnjak, Pierre H Richemond, Nenad Tomasev, Florian Strub, Jacob C Walker, Felix Hill, Lars Holger Buesing, Razvan Pascanu, Charles Blundell, and Jovana Mitrovic. Semppl: Predicting pseudo-labels for better contrastive representations. *arXiv preprint arXiv:2301.05158*, 2023.
- Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 132–149, 2018.
- Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33:9912–9924, 2020.
- Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé J’egou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9630–9640, 2021.
- Hui Chen, Fangqing Liu, Yin Wang, Liyue Zhao, and Hao Wu. A variational approach for learning from positive and unlabeled data. *Advances in Neural Information Processing Systems*, 33: 14844–14854, 2020a.
- Jia-Lue Chen, Jia-Jia Cai, Yuan Jiang, and Sheng-Jun Huang. Pu active learning for recommender systems. *Neural Processing Letters*, 53(5):3639–3652, 2021.
- Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020b.
- Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255, 2020c.
- Xuxi Chen, Wuyang Chen, Tianlong Chen, Ye Yuan, Chen Gong, Kewei Chen, and Zhangyang Wang. Self-pu: Self boosted and calibrated positive-unlabeled training. In *International Conference on Machine Learning*, pp. 1510–1519. PMLR, 2020d.

- Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pp. 539–546. IEEE, 2005.
- Ching-Yao Chuang, Joshua Robinson, Lin Yen-Chen, Antonio Torralba, and Stefanie Jegelka. Debiased contrastive learning. *arXiv preprint arXiv:2007.00224*, 2020.
- Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 215–223. JMLR Workshop and Conference Proceedings, 2011.
- Michael B Cohen, Yin Tat Lee, Gary Miller, Jakub Pachocki, and Aaron Sidford. Geometric median in nearly linear time. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pp. 9–21, 2016.
- Jingyi Cui, Weiran Huang, Yifei Wang, and Yisen Wang. Rethinking weak supervision in helping contrastive learning. *arXiv preprint arXiv:2306.04160*, 2023.
- François Denis. Pac learning from positive statistical queries. In *International Conference on Algorithmic Learning Theory*, pp. 112–126. Springer, 1998.
- David L Donoho and Peter J Huber. The notion of breakdown point. *A festschrift for Erich L Lehmann*, 157184, 1983.
- Jun Du and Zhihua Cai. Modelling class noise with symmetric and asymmetric distributions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.
- Marthinus C Du Plessis, Gang Niu, and Masashi Sugiyama. Analysis of learning from positive and unlabeled data. *Advances in neural information processing systems*, 27:703–711, 2014.
- Charles Elkan and Keith Noto. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 213–220, 2008.
- Logan Engstrom, Andrew Ilyas, Hadi Salman, Shibani Santurkar, and Dimitris Tsipras. Robustness (python library), 2019. URL <https://github.com/MadryLab/robustness>.
- Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. *ArXiv*, abs/2104.08821, 2021.
- Saurabh Garg, Yifan Wu, Alexander J Smola, Sivaraman Balakrishnan, and Zachary Lipton. Mixture proportion estimation and pu learning: A modern approach. *Advances in Neural Information Processing Systems*, 34, 2021.
- Aritra Ghosh and Andrew Lan. Contrastive learning improves model robustness under label noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2703–2708, 2021.
- Aritra Ghosh, Naresh Manwani, and PS Sastry. Making risk minimization tolerant to label noise. *Neurocomputing*, 160:93–107, 2015.
- Aritra Ghosh, Himanshu Kumar, and P Shanti Sastry. Robust loss functions under label noise for deep neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.
- Jean-Bastien Grill, Florian Strub, Florent Altch'e, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. *ArXiv*, abs/2006.07733, 2020.
- Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 297–304. JMLR Workshop and Conference Proceedings, 2010.

- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738, 2020.
- Cho-Jui Hsieh, Nagarajan Natarajan, and Inderjit Dhillon. Pu learning for matrix completion. In *International conference on machine learning*, pp. 2445–2453. PMLR, 2015.
- Wenpeng Hu, Ran Le, Bing Liu, Feng Ji, Jinwen Ma, Dongyan Zhao, and Rui Yan. Predictive adversarial learning from positive and unlabeled data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 7806–7814, 2021.
- Tianyu Hua, Wenxiao Wang, Zihui Xue, Sucheng Ren, Yue Wang, and Hang Zhao. On feature decorrelation in self-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9598–9608, 2021.
- Peter J Huber. *Robust statistical procedures*. SIAM, 1996.
- Dmitry Ivanov. Dedpul: Difference-of-estimated-densities-based positive-unlabeled learning. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 782–790. IEEE, 2020.
- Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.
- Armand Joulin and Francis Bach. A convex relaxation for weakly supervised classifiers. *arXiv preprint arXiv:1206.6413*, 2012.
- Masahiro Kato, Takeshi Teshima, and Junya Honda. Learning from positive and unlabeled data with a selection bias. In *International conference on learning representations*, 2018.
- Diane Kelly and Jaime Teevan. Implicit feedback for inferring user preference: a bibliography. In *Acm Sigir Forum*, volume 37, pp. 18–28. ACM New York, NY, USA, 2003.
- Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020.
- Ryuichi Kiryo, Gang Niu, Marthinus C Du Plessis, and Masashi Sugiyama. Positive-unlabeled learning with non-negative risk estimator. *Advances in neural information processing systems*, 30, 2017.
- Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1920–1929, 2019.
- Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.
- Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- Wee Sun Lee and Bing Liu. Learning with positive and unlabeled examples using weighted logistic regression. In *ICML*, volume 3, pp. 448–455, 2003.
- Changchun Li, Ximing Li, Lei Feng, and Jihong Ouyang. Who is your right mixup partner in positive and unlabeled learning. In *International Conference on Learning Representations*, 2022.
- Bing Liu, Wee Sun Lee, Philip S Yu, and Xiaoli Li. Partially supervised classification of text documents. In *ICML*, volume 2, pp. 387–394. Sydney, NSW, 2002.

- Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip S Yu. Building text classifiers using positive and unlabeled examples. In *Third IEEE International Conference on Data Mining*, pp. 179–186. IEEE, 2003.
- Qinchao Liu, Bangzuo Zhang, Haichao Sun, Yu Guan, and Lei Zhao. A novel k-means clustering algorithm based on positive examples and careful seeding. In *2010 International Conference on Computational and Information Sciences*, pp. 767–770. IEEE, 2010.
- Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.
- Hendrik P Lopuhaa, Peter J Rousseeuw, et al. Breakdown points of affine equivariant estimators of multivariate location and covariance matrices. *The Annals of Statistics*, 19(1):229–248, 1991.
- Chuan Luo, Pu Zhao, Chen Chen, Bo Qiao, Chao Du, Hongyu Zhang, Wei Wu, Shaowei Cai, Bing He, Saravanakumar Rajmohan, et al. Pulns: Positive-unlabeled learning with effective negative sample selector. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 8784–8792, 2021.
- Meena Mahajan, Prajakta Nimborkar, and Kasturi Varadarajan. The planar k-means problem is np-hard. *Theoretical Computer Science*, 442:13–21, 2012.
- Stanislav Minsker et al. Geometric median and robust estimation in banach spaces. *Bernoulli*, 21(4):2308–2335, 2015.
- Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6707–6717, 2020.
- Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. *Advances in neural information processing systems*, 26, 2013.
- Maxim Naumov, Dheevatsa Mudigere, Hao-Jun Michael Shi, Jianyu Huang, Narayanan Sundaraman, Jongsoo Park, Xiaodong Wang, Udit Gupta, Carole-Jean Wu, Alisson G Azzolini, et al. Deep learning recommendation model for personalization and recommendation systems. *arXiv preprint arXiv:1906.00091*, 2019.
- Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.
- Advait Parulekar, Liam Collins, Karthikeyan Shanmugam, Aryan Mokhtari, and Sanjay Shakkottai. Infonce loss provably learns cluster-preserving representations. *arXiv preprint arXiv:2302.07920*, 2023.
- Minlong Peng, Xiaoyu Xing, Qi Zhang, Jinlan Fu, and Xuanjing Huang. Distantly supervised named entity recognition using positive-unlabeled learning. *arXiv preprint arXiv:1906.01378*, 2019.
- Qi Qian. Stable cluster discrimination for deep clustering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16645–16654, 2023.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.
- Harish Ramaswamy, Clayton Scott, and Ambuj Tewari. Mixture proportion estimation via kernel embeddings of distributions. In *International conference on machine learning*, pp. 2052–2060. PMLR, 2016.
- Yafeng Ren, Donghong Ji, and Hongbin Zhang. Positive unlabeled learning for deceptive reviews detection. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 488–498, 2014.
- Nikunj Saunshi, Orestis Pavlakidis, Sanjeev Arora, Mikhail Khodak, and Hrishikesh Khandeparkar. A theoretical analysis of contrastive unsupervised representation learning. In *International Conference on Machine Learning*, pp. 5628–5637. PMLR, 2019.

- Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 815–823, 2015.
- Yuming Shen, Ziyi Shen, Menghan Wang, Jie Qin, Philip Torr, and Ling Shao. You never cluster alone. *Advances in Neural Information Processing Systems*, 34:27734–27746, 2021.
- Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems*, pp. 1857–1865, 2016.
- Daiki Tanaka, Daiki Ikami, and Kiyoharu Aizawa. A novel perspective for positive-unlabeled learning via noisy labels. *arXiv preprint arXiv:2103.04685*, 2021.
- Yuandong Tian, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning dynamics without contrastive pairs. In *International Conference on Machine Learning*, pp. 10268–10278. PMLR, 2021.
- Christopher Tosh, Akshay Krishnamurthy, and Daniel Hsu. Contrastive estimation reveals topic posterior information to linear models. *The Journal of Machine Learning Research*, 22(1):12883–12913, 2021.
- Wouter Van Gansbeke, Simon Vandenbende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Scan: Learning to classify images without labels. In *European conference on computer vision*, pp. 268–285. Springer, 2020.
- Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pp. 9929–9939. PMLR, 2020.
- Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 322–330, 2019.
- Tong Wei, Feng Shi, Hai Wang, Wei-Wei Tu Li, et al. Mixpul: consistency-based augmentation for positive and unlabeled learning. *arXiv preprint arXiv:2004.09388*, 2020.
- Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3733–3742, 2018.
- Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.
- Linli Xu, James Neufeld, Bryce Larson, and Dale Schuurmans. Maximum margin clustering. *Advances in neural information processing systems*, 17, 2004.
- Peng Yang, Xiao-Li Li, Jian-Ping Mei, Chee-Keong Kwok, and See-Kiong Ng. Positive-unlabeled learning for disease gene identification. *Bioinformatics*, 28(20):2640–2647, 2012.
- Yu Yao, Tongliang Liu, Bo Han, Mingming Gong, Gang Niu, Masashi Sugiyama, and Dacheng Tao. Rethinking class-prior estimation for positive-unlabeled learning. In *International Conference on Learning Representations*, 2021.
- Jordan Yoder and Carey E Priebe. Semi-supervised k-means++. *Journal of Statistical Computation and Simulation*, 87(13):2597–2608, 2017.
- Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962*, 2019.
- Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *ICML*, 2021.
- Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Zhi-Hua Zhou. A brief introduction to weakly supervised learning. *National science review*, 5(1): 44–53, 2018.

CONTENTS

1	Introduction	1
2	Contrastive Approach to PU Learning	3
2.1	Problem Setup	3
2.2	Representation Learning from Positive Unlabeled Data	4
2.2.1	Incorporating PU Supervision	4
2.3	Positive Unlabeled Pseudo Labeling	6
3	Empirical Evidence	8
4	Conclusion, Limitations and Broader Impact	9
A	Appendix	17
A.1	Notations and Abbreviations	17
A.2	Extended Related Work	18
A.3	Background	19
A.3.1	Discussion on different PU Learning problem settings	19
A.3.2	Connection to Learning under Class Dependent Label Noise	19
A.3.3	Cost Sensitive PU Learning	20
A.4	Full Algorithm: Parameter Free Contrastive PU Learning	22
A.5	PUCL: POSITIVE UNLABELED REPRESENTATION LEARNING	23
A.5.1	Generalization Benefits of Incorporating Additional Positives	23
A.5.2	Grouping semantically different objects together :	23
A.5.3	Convergence Benefits of Incorporating Additional Positives	26
A.5.4	Comparison with Parametric Contrastive Learning Objectives:	29
A.5.5	Bias Variance Tradeoff	30
A.6	PUPL: POSITIVE UNLABELED PSEUDO LABELING	34
A.6.1	NECESSARY DEFINITIONS AND INTERMEDIATE LEMMAS	34
A.6.2	PROOF OF Theorem 3.	37
A.6.3	PROOF OF LEMMA 2	38
A.6.4	PROOF OF LEMMA 3	39
A.7	Additional Reproducibility Details	40
A.7.1	Positive Unlabeled Benchmark Datasets	40
A.7.2	Positive Unlabeled Baselines	40
A.7.3	Image Augmentations for Contrastive Training	41
A.7.4	PyTorch Style Pseudo Codes	41

A APPENDIX

A.1 NOTATIONS AND ABBREVIATIONS

ssCL	Self Supervised Contrastive Learning
sCL-PU	Naive PU adaptation of Supervised Contrastive Learning
PUCL	Positive Unlabeled Contrastive Learning
PUPL	Positive Unlabeled Pseudo Labeling
a	A scalar (integer or real)
\mathbf{a}	A vector
\mathbf{A}	A matrix
a	A scalar random variable
\mathbf{a}	A vector-valued random variable
\mathbb{A}	A set
$\{0, 1\}$	The set containing 0 and 1
$\{0, 1, \dots, n\}$	The set of all integers between 0 and n
$[a, b]$	The real interval including a and b
$\mathbb{A} \setminus \mathbb{B}$	Set subtraction, i.e., the set containing the elements of \mathbb{A} that are not in \mathbb{B}
\mathbf{a}_i	Element i of the random vector \mathbf{a}
$P(\mathbf{a})$	A probability distribution over a discrete variable
$p(\mathbf{a})$	A probability distribution over a continuous variable, or over a variable whose type has not been specified
$f : \mathbb{A} \rightarrow \mathbb{B}$	The function f with domain \mathbb{A} and range \mathbb{B}
$f \circ g$	Composition of the functions f and g
$f(\mathbf{x}; \theta)$	A function of \mathbf{x} parametrized by θ . (Sometimes we write $f(\mathbf{x})$ and omit the argument θ to lighten notation)
$\ \mathbf{x}\ _p$	L^p norm of \mathbf{x}
$\mathbf{1}(condition)$	is 1 if the condition is true, 0 otherwise

A.2 EXTENDED RELATED WORK

Positive Unlabeled (PU) Learning : Existing PU learning algorithms primarily differ in the way they handle the semantic annotations of unlabeled examples.

One set of approaches rely on heuristic based *sample selection* where the idea is to identify potential negatives, positives or both samples in the unlabeled set; followed by performing traditional supervised learning using these pseudo-labeled instances in conjunction with available labeled positive data (Liu et al., 2002; Bekker & Davis, 2020; Luo et al., 2021; Wei et al., 2020).

A second set of approaches adopt a *re-weighting* strategy, where the unlabeled samples are treated as down-weighted negative examples (Liu et al., 2003; Lee & Liu, 2003). However, both of these approaches can be difficult to scale, as identifying reliable negatives or finding appropriate weights can be challenging or expensive to tune, especially in deep learning scenarios (Garg et al., 2021).

Recent works, further use the cost-sensitive classifiers in conjunction with other techniques. For example (Chen et al., 2021) use self distillation to improve the initial nnPU model; (Wei et al., 2020) uses mixup data augmentation to create augmented unlabeled samples with soft labels. Moreover, PU learning is also closely related to other robustness and weakly supervised settings, including learning under distribution shift (Garg et al., 2021), asymmetric label noise (Tanaka et al., 2021; Du & Cai, 2015) and semi-supervised learning (Chen et al., 2020c; Assran et al., 2020; Zhou, 2018).

Contrastive Representation Learning: Self-supervised learning has demonstrated superior performances over supervised methods on various benchmarks. Joint-embedding methods (Chen et al., 2020b; Grill et al., 2020; Zbontar et al., 2021; Caron et al., 2021) are one the most promising approach for self-supervised representation learning where the embeddings are trained to be invariant to distortions. To prevent trivial solutions, a popular method is to apply pulsive force between embeddings from different images, known as contrastive learning. Contrastive loss is shown to be useful in various domains, including natural language processing (Gao et al., 2021), multimodal learning (Radford et al., 2021). Contrastive loss can also benefit supervised learning (Khosla et al., 2020).

Clustering based Pseudo Labeling: Our approach is also closely related simultaneous clustering and pseudo-labeling approaches like DeepCluster (Caron et al., 2020) **Clustering Based Representation Learning:** Simultaneous clustering and representation learning has gained popularity recently. **DeepCluster** (Caron et al., 2018) uses off-the-shelf clustering method e.g. kMeans to assign pseudo labels based on cluster membership and subsequently learns the representation using standard CE loss over the pseudo-labels. However, this standard simultaneous clustering and representation learning framework is often susceptible to degenerate solutions (e.g. trivially assigning all the samples to a single label) even for linear models (Xu et al., 2004; Joulin & Bach, 2012; Bach & Harchaoui, 2007). **SeLA** (Asano et al., 2019) alleviate this by adding the constraint that the label assignments must partition the data in equally-sized subsets. **Twin contrastive clustering (TCC)** (Shen et al., 2021), **SCAN** (Van Gansbeke et al., 2020), (Qian, 2023), **SwAV** (Caron et al., 2020; Bošnjak et al., 2023) combines ideas from contrastive learning and clustering based representation learning methods to perform simultaneous clusters the data while enforcing consistency between cluster assignments produced for different augmentations of the same image in an online fashion.

A.3 BACKGROUND

A.3.1 DISCUSSION ON DIFFERENT PU LEARNING PROBLEM SETTINGS

Case Control Setting: Recall, the PU setting we have studied in the paper. Let $\mathbf{x} \in \mathbb{R}^d$ and $y \in Y = \{0, 1\}$ be the underlying input (i.e., feature) and output (label) random variables respectively and let $p(\mathbf{x}, y)$ denote the true underlying joint density of (\mathbf{x}, y) . Then, a PU training dataset is composed of a set \mathcal{X}_P of n_P positively labeled samples and a set \mathcal{X}_U of n_U unlabeled samples (a mixture of both positives and negatives) i.e.

$$\mathcal{X}_{PU} = \mathcal{X}_P \cup \mathcal{X}_U, \quad \mathcal{X}_P = \{\mathbf{x}_i^P\}_{i=1}^{n_P} \stackrel{i.i.d.}{\sim} p(\mathbf{x}|y=1), \quad \mathcal{X}_U = \{\mathbf{x}_i^U\}_{i=1}^{n_U} \stackrel{i.i.d.}{\sim} p(\mathbf{x}) \quad (8)$$

This particular setup of how PU learning dataset is generated is referred to as the case-control setting (Bekker et al., 2019; Blanchard et al., 2010) and possibly widely used.

Single Dataset Setting: Now, we consider another setting referred to as the Single Dataset setting where there is only one dataset. The positive samples are randomly labeled from the dataset as opposed to being independent samples from the positive marginal. Thus the unlabeled set is no longer truly representative of the mixture. However our experiments ?? reveal that contrastive learning is still able learn representations following similar trends as case-control settings possibly because it is agnostic to how the data is generated unlike unbiased PU Learning methods.

A.3.2 CONNECTION TO LEARNING UNDER CLASS DEPENDENT LABEL NOISE

PU Learning is also closely related to the popular learning under label noise problem where the goal is to robustly train a classifier when a fraction of the training examples are mislabeled. This problem is extensively studied under both generative and discriminative settings and is an active area of research (Ghosh et al., 2015; 2017; Ghosh & Lan, 2021; Wang et al., 2019; Zhang et al., 2017).

Consider the following instance of *learning a binary classifier under class dependent label noise* i.e. the class conditioned probability of being mislabeled is ξ_P and ξ_N respectively for the positive and negative samples. Formally, let \mathcal{X}_{PN} be the underlying clean binary dataset.

$$\mathcal{X}_{PN} = \mathcal{X}_P \cup \mathcal{X}_N, \quad \mathcal{X}_P = \{\mathbf{x}_i^P\}_{i=1}^{n_P} \stackrel{i.i.d.}{\sim} p(\mathbf{x}|y=+1), \quad \mathcal{X}_N = \{\mathbf{x}_i^N\}_{i=n_P+1}^{n_N} \stackrel{i.i.d.}{\sim} p(\mathbf{x}|y=-1) \quad (9)$$

Instead of \mathcal{X}_{PN} , a binary classifier needs to be trained from a noisy dataset $\tilde{\mathcal{X}}_{PN}$ with class dependent noise rates ξ_P and ξ_N i.e.

$$\tilde{\mathcal{X}}_{PN} = \{(\mathbf{x}_i, \tilde{y}_i)\}_{i=1}^{n_P+n_N}, \quad \xi_P = p(\tilde{y}_i \neq y_i | y_i = +1), \quad \xi_N = p(\tilde{y}_i \neq y_i | y_i = -1) \quad (10)$$

REDUCTION OF PU LEARNING 2.1 TO LEARNING WITH LABEL NOISE : Recall from Section 2.2 the naive disambiguation-free approach (Li et al., 2022), where the idea is to pseudo label the PU dataset as follows: Treat the unlabeled examples as negative and train an ordinary binary classifier over the pseudo labeled dataset. Clearly, since the unlabeled samples (a mixture of positives and negatives) are being pseudo labeled as negative, this is an instance of learning with class dependent label noise:

$$\tilde{\mathcal{X}}_{PN} = \mathcal{X}_P \cup \tilde{\mathcal{X}}_N, \quad \mathcal{X}_P = \{\mathbf{x}_i^P\}_{i=1}^{n_P} \stackrel{i.i.d.}{\sim} p(\mathbf{x}|y=1), \quad \tilde{\mathcal{X}}_N = \{\mathbf{x}_i^U\}_{i=1}^{n_U} \stackrel{i.i.d.}{\sim} p(\mathbf{x}) \quad (11)$$

It is easy to show that noise rates are:

$$E(\xi_P) = \frac{\pi_P}{\gamma + \pi_P} \text{ and } \xi_N = 0 \quad (12)$$

Where $E(\gamma) = \frac{n_P}{n_U}$ and $\pi_P = p(y=1|\mathbf{x})$ are training distribution dependent parameters.

Under the standard robust Empirical Risk Minimization (ERM) framework, the goal is to robustly estimate the true risk i.e. for some loss we want the estimated risk (from the noisy data) to be close to the true risk (from the clean data) i.e. with high probability:

$$\Delta = \left\| \hat{\mathcal{R}}(\boldsymbol{\theta}) - \mathcal{R}(\boldsymbol{\theta}^*) \right\|_2 = \mathbb{E} \left\| \ell\left(f_{\boldsymbol{\theta}}(\mathbf{x}), \tilde{y}\right) - \ell\left(f_{\boldsymbol{\theta}^*}(\mathbf{x}), y\right) \right\|_2 \leq \epsilon$$

A popular way to measure the resilience of an estimator against corruption is via breakdown point analysis (Donoho & Huber, 1983; Huber, 1996; Lopuhaa et al., 1991; Acharya et al., 2022).

Definition 1 (Breakdown point). *Breakdown point ψ of an estimator is simply defined as the smallest fraction of corruption that must be introduced to cause an estimator to break implying Δ (risk estimation error) can become unbounded i.e. the estimator can produce arbitrarily wrong estimates.*

It is easy to show the following result:

Lemma 1. *Consider the problem of learning a binary classifier (P vs N) in presence of class-dependent label noise with noise rates $E(\xi_P) = \frac{\pi_p}{\gamma + \pi_p}$, $\xi_N = 0$. Without additional distributional assumption, no robust estimator can guarantee bounded risk estimate $\left\| \hat{\mathcal{R}}(\boldsymbol{\theta}) - \mathcal{R}(\boldsymbol{\theta}^*) \right\|_2 \leq \epsilon$ if $\gamma \leq 2\pi_p - 1$ where $\gamma = \frac{n_p}{n_u}$ and $\pi_p = p(y=1|\mathbf{x})$ denotes the underlying class prior.*

Proof. This result follows from using the fact that for any estimator $0 \leq \psi < \frac{1}{2}$ (Lopuhaa et al., 1991; Minsker et al., 2015; Cohen et al., 2016; Acharya et al., 2022) i.e. for robust estimation to be possible, the corruption fraction $\alpha = \frac{\pi_p}{\gamma+1} < \frac{1}{2}$. ■

This result suggests that PU Learning cannot be solved by off-the-shelf label noise robust algorithms and specialized algorithms need to be designed.

A.3.3 COST SENSITIVE PU LEARNING

Consider training linear classifier $v_{\mathbf{v}}(\cdot) : \mathbb{R}^k \rightarrow \mathbb{R}^{|Y|}$ where $k \in \mathbb{R}^+$ is the dimension of the features. In the (fully) supervised setting (PN) labeled examples from both class marginals are available and the linear classifier can be trained using standard supervised classification loss e.g. CE. However, in the PU learning setup since no labeled negative examples are provided it is non-trivial to train.

As discussed in Appendix A.3.2, without additional assumptions the equivalent class dependent label noise learning problem cannot be solved when $\gamma \leq 2\pi_p - 1$. However, note that in PU Learning, we additionally know that a subset of the dataset is correctly labeled i.e.

$$p(\tilde{y}_i = y_i = 1 | \mathbf{x}_i \in \mathbb{P}) = 1$$

Can we use this additional information to enable PU Learning even when $\gamma \leq 2\pi_p - 1$?

Remarkably, SOTA cost-sensitive PU learning algorithms tackle this by forming an unbiased estimate of the true risk from PU data (Blanchard et al., 2010) by assuming additional knowledge of the true class prior $\pi_p = p(y=1|\mathbf{x})$. The unbiased estimator dubbed uPU (Blanchard et al., 2010; Du Plessis et al., 2014) of the true risk $R_{PN}(v)$ from PU data is given as:

$$\hat{R}_{pu}(v) = \pi_p \hat{R}_p^+(v) + \left[\hat{R}_u^-(v) - \pi_p \hat{R}_p^-(v) \right]$$

where we denote the empirical estimates computed over PU dataset (1) as:

$$\hat{R}_p^+(v) = \frac{1}{n_p} \sum_{i=1}^{n_p} \ell(v(\mathbf{x}_i^P), 1), \quad \hat{R}_p^-(v) = \frac{1}{n_p} \sum_{i=1}^{n_p} \ell(v(\mathbf{x}_i^P), 0), \quad \hat{R}_u^-(v) = \frac{1}{n_u} \sum_{i=1}^{n_u} \ell(v(\mathbf{x}_i^U), 0)$$

and $\ell(\cdot, \cdot) : Y \times Y \rightarrow \mathbb{R}$ is the classification loss e.g. CE.

In practice, clipping the estimated negative risk results in a further improvement (Kiryo et al., 2017).

$$\hat{R}_{pu}(v) = \pi_p \hat{R}_p^+(v) + \max \left\{ 0, \hat{R}_u^-(v) - \pi_p \hat{R}_p^-(v) \right\} \quad (13)$$

This clipped loss dubbed NNPU is the de-facto approach to solve PU problems in practical settings and we use this as a powerful baseline for training the downstream PU classifier.

As discussed before, we identify two main issues related to these cost-sensitive estimators:

- **Class Prior Estimate :** The success of these estimators hinges upon the knowledge of the oracle class prior π_p^* for their success. It is immediate to see that an error in class prior estimate $\|\hat{\pi}_p - \pi_p^*\|_2 \leq \xi$ results in an estimation bias $\sim \mathcal{O}(\xi)$ that can result in poor generalization, slower convergence or both.

Our experiments (Figure 15) suggest that even small approximation error in estimating the class prior can lead to notable degradation in the overall performance of the estimators.

Unfortunately however in practical settings (e.g. large-scale recommendation) the class prior is not available and often estimating it with high accuracy using some MPE (Garg et al., 2021; Ivanov, 2020; Ramaswamy et al., 2016) algorithm can be quite costly.

- **Low Supervision Regime :** While these estimators are significantly more robust than the vanilla supervised approach, our experiments (Figure 14) suggest that they might produce decision boundaries that are not closely aligned with the true decision boundary especially as γ becomes smaller (Kiryo et al., 2017; Du Plessis et al., 2014). Note that, when available supervision is limited i.e. when γ is small, the estimates \hat{R}_p^+ and \hat{R}_p^- suffer from increased variance resulting in increased variance of the overall estimator $\sim \mathcal{O}(\frac{1}{n_p})$. For sufficiently small γ these estimators are likely to result in poor performance due to large variance.

A.4 FULL ALGORITHM: PARAMETER FREE CONTRASTIVE PU LEARNING

Algorithm 1 Contrastive Positive Unlabeled Learning

initialize: PU training data \mathcal{X}_{PU} ; batch size b; temperature parameter $\tau > 0$; randomly initialized encoder $g_{\mathbf{B}}(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^k$, projection network: $h_{\mathbf{\Gamma}}(\cdot) : \mathbb{R}^k \rightarrow \mathbb{R}^p$, and linear classifier $v_{\mathbf{v}}(\cdot) : \mathbb{R}^k \rightarrow \mathbb{R}^{|\mathcal{Y}|}$; family of stochastic augmentations \mathcal{T} .

A. PUCL : Positive Unlabeled Contrastive Representation Learning

for epochs $e = 1, 2, \dots$, until convergence **do**

$\text{select mini-batch: } \mathcal{D} = \{\mathbf{x}_i\}_{i=1}^b \sim \mathcal{X}_{\text{PU}}$ $\text{and sample augmentations: } t(\cdot) \sim \mathcal{T}, t'(\cdot) \sim \mathcal{T}$ $\text{create multi-viewed batch: } \tilde{\mathcal{D}} = \{\tilde{\mathbf{x}}_i = t(\mathbf{x}_i), \tilde{\mathbf{x}}_{a(i)} = t'(\mathbf{x}_i)\}_{i=1}^b$ $\mathbb{I} = \{1, 2, \dots, 2b\} \text{ is the index set of } \tilde{\mathcal{D}}$ $\text{and } \mathbb{P} = \{i \in \mathbb{I} : \mathbf{x}_i \in \mathcal{X}_{\text{P}}\}, \mathbb{U} = \{j \in \mathbb{I} : \mathbf{x}_j \in \mathcal{X}_{\text{U}}\}$ $\text{obtain representations: } \{\mathbf{z}_j\}_{j \in \mathbb{I}} = \{\mathbf{z}_i = h_{\mathbf{\Gamma}} \circ g_{\mathbf{B}}(\tilde{\mathbf{x}}_i), \mathbf{z}_{a(i)} = h_{\mathbf{\Gamma}} \circ g_{\mathbf{B}}(\tilde{\mathbf{x}}_{a(i)})\}_{i=1}^b$ $\text{compute pairwise similarity: } \mathbf{z}_i \cdot \mathbf{z}_j = \frac{1}{\tau} \frac{\mathbf{z}_i^T \mathbf{z}_j}{\ \mathbf{z}_i\ \ \mathbf{z}_j\ }, P_{i,j} = \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_j)}{\sum_{k \in \mathbb{I}} \mathbf{1}(k \neq i) \exp(\mathbf{z}_i \cdot \mathbf{z}_k)}, \forall i, j \in \mathbb{I}$ $\text{compute loss: } \mathcal{L}_{\text{PUCL}} = -\frac{1}{ \mathbb{I} } \sum_{i \in \mathbb{I}} \left[\mathbf{1}(i \in \mathbb{P}) \frac{1}{ \mathbb{P} \setminus i } \sum_{j \in \mathbb{P}} \mathbf{1}(j \neq i) \log P_{i,j} + \mathbf{1}(i \in \mathbb{U}) \log P_{i,a(i)} \right]$ $\text{update network parameters } \mathbf{B}, \mathbf{\Gamma} \text{ to minimize } \mathcal{L}_{\text{PUCL}}$	
--	--

end

return: encoder $g_{\mathbf{B}}(\cdot)$ and throw away $h_{\mathbf{\Gamma}}(\cdot)$.

B. PUPPL: Positive Unlabeled Pseudo Labeling

obtain representations: $\mathcal{Z}_{\text{P}} = \{\mathbf{r}_i = g_{\mathbf{B}}(\mathbf{x}_i) : \forall \mathbf{x}_i \in \mathcal{X}_{\text{P}}\}$, $\mathcal{Z}_{\text{U}} = \{\mathbf{r}_j = g_{\mathbf{B}}(\mathbf{x}_j) : \forall \mathbf{x}_j \in \mathcal{X}_{\text{U}}\}$
initialize pseudo labels: $\tilde{y}_i = y_i = 1 : \forall \mathbf{r}_i \in \mathcal{Z}_{\text{P}}$ and $\tilde{y}_j = 0 : \forall \mathbf{r}_j \in \mathcal{Z}_{\text{U}}$

$\text{initialize cluster centers: } \mu_{\text{P}} = \frac{1}{ \mathcal{Z}_{\text{P}} } \sum_{\mathbf{r}_i \in \mathcal{Z}_{\text{P}}} \mathbf{r}_i, \mu_{\text{N}} \stackrel{D(\mathbf{x}')}{\sim} \mathcal{Z}_{\text{U}}$ $\text{where } D(\mathbf{x}') = \frac{\ \mathbf{x}' - \mu_{\text{P}}\ ^2}{\sum_{\mathbf{x}} \ \mathbf{x} - \mu_{\text{P}}\ ^2}$	
--	--

while not converged **do**

$\text{pseudo-label: } \forall \mathbf{r}_i \in \mathcal{Z}_{\text{U}} : \tilde{y}_i = 1 \text{ if } \mu_{\text{P}} = \arg \min_{\mu \in \{\mu_{\text{P}}, \mu_{\text{N}}\}} \ \mathbf{r}_i - \mu\ ^2 \text{ else } \tilde{y}_i = 0$ $\tilde{\mathcal{Z}}_{\text{P}} = \mathcal{Z}_{\text{P}} \cup \{\mathbf{r}_i \in \mathcal{Z}_{\text{U}} : \tilde{y}_i = 1\}, \tilde{\mathcal{Z}}_{\text{N}} = \{\mathbf{z}_i \in \mathcal{Z}_{\text{U}} : \tilde{y}_i = 0\}$ $\text{update cluster centers: } \mu_{\text{P}} = \frac{1}{ \tilde{\mathcal{Z}}_{\text{P}} } \sum_{\mathbf{z}_i \in \tilde{\mathcal{Z}}_{\text{P}}} \mathbf{z}_i, \mu_{\text{N}} = \frac{1}{ \tilde{\mathcal{Z}}_{\text{N}} } \sum_{\mathbf{z}_i \in \tilde{\mathcal{Z}}_{\text{N}}} \mathbf{z}_i$	
--	--

end

return: $\tilde{\mathcal{X}}_{\text{PU}} = \{(\mathbf{x}_i, \tilde{y}_i) : \forall \mathbf{x}_i \in \mathcal{X}_{\text{PU}}\}$

C. Train Binary Classifier

update network parameters \mathbf{v} to minimize cross-entropy loss $\mathcal{L}_{\text{CE}}(\mathbf{v}^T g_{\mathbf{B}}(\mathbf{x}_i), \tilde{y}_i)$
return: Positive Unlabeled classifier : $f_{\mathbf{v}, \mathbf{B}} = v_{\mathbf{v}} \circ g_{\mathbf{B}}(\cdot)$

A.5 PUCL: POSITIVE UNLABELED REPRESENTATION LEARNING

Summary: One way to obtain a representation manifold where the embeddings (features) exhibit linear separability is via contrastive learning (Parulekar et al., 2023).

- However, standard self-supervised contrastive loss ssCL (4) is unable to leverage the available supervision in the form of labeled positives.
- (Theorem 1) On the other hand, naive adaptation of the supervised contrastive loss SCL-PU (5) suffers from statistical bias in the PU setting that can result in significantly poor representations especially in the low supervision regime i.e. when only a handful labeled positive examples are available.
- (Theorem 2) To this end, the proposed objective PUCL leverages the available supervision judiciously to form an unbiased risk estimator of the ideal objective. Further, we show that it is provably more efficient than the self-supervised counterpart.

A.5.1 GENERALIZATION BENEFITS OF INCORPORATING ADDITIONAL POSITIVES

As previously discussed, the main observation we make is that judiciously incorporating available PU supervision is crucial for the success of contrastive learning over PU Learning. The unsupervised ssCL (4) objective is completely agnostic of the labels, resulting in representation that are while robust to noisy label, has poor generalization performance on downstream PU classification. On the other hand, SCL-PU: naive adaptation of SCL while performs well in low (high) noise (supervision) settings, it suffers from major performance degradation in the high (low) noise (supervision) regime. PUCL interpolates nicely between the robustness and generalization trade-off by judiciously incorporating the labeled positive to form an unbiased version of SCL. In Figure 5, we present a few more results affirming this observation over multiple datasets, encompassing both Single Dataset and Case-Control PU learning settings.

A.5.2 GROUPING SEMANTICALLY DIFFERENT OBJECTS TOGETHER :

An important underlying assumption unsupervised learning is that the features contains information about the underlying label. Indeed, if $p(x)$ has no information about $p(y|x)$, no unsupervised representation learning method e.g. ssCL can hope to learn cluster-preserving representations.

However, in **fully supervised setting**, since semantic annotations are available, it is possible to find a representation space where semantically dissimilar objects are grouped together based on labels i.e. clustered based on semantic annotations via supervised objectives e.g. CE. While, it is important to note that such models would be prone to over-fitting and might generalize poorly to unseen data. Since supervised contrastive learning objectives SCL (14) (Khosla et al., 2020) use semantic annotations to guide the contrastive training, it can also be effective in such scenarios.

In particular, in SCL, in addition to self-augmentations, each anchor is attracted to all the other augmentations in the batch that share the same class label. For a fully supervised binary setting it takes the following form:

$$\mathcal{L}_{\text{SCL}} = -\frac{1}{|\mathbb{P}|} \sum_{i \in \mathbb{P}} \left[\mathbf{1}(i \in \mathbb{P}) \frac{1}{|\mathbb{P} \setminus i|} \sum_{j \in \mathbb{P} \setminus i} \mathbf{z}_i \cdot \mathbf{z}_j + \mathbf{1}(i \in \mathbb{N}) \frac{1}{|\mathbb{N} \setminus i|} \sum_{j \in \mathbb{N} \setminus i} \mathbf{z}_i \cdot \mathbf{z}_j - \log Z(\mathbf{z}_i) \right] \quad (14)$$

where \mathbb{P} and \mathbb{N} denote the subset of indices in the augmented batch $\tilde{\mathcal{D}}$ that are labeled positive and negative respectively i.e. $\mathbb{P} = \{i \in \mathbb{I} : \mathbf{y}_i = 1\}$, $\mathbb{N} = \{i \in \mathbb{I} : \mathbf{y}_i = 0\}$. Clearly, \mathcal{L}_{SCL} (14) is a **consistent estimator** of the ideal objective $\mathcal{L}_{\text{CL}}^*$ (3). Since the expected similarity of positive pairs is computed over all the available samples from the same class marginal as anchor, this loss enjoys a lower variance compared to its self-supervised counterpart $\mathcal{L}_{\text{ssCL}}$ (4).

In the **PU learning setting**, PUCL (6) behaves in a similar way. It incorporates both semantic similarity (via pulling self augmentations together) and semantic annotation (via pulling together labeled positives together). Intuitively, by interpolating between supervised and unsupervised contrastive objectives, PUCL favors representations where both semantically similar (feature) examples are grouped together along with all the labeled positives (annotations) are grouped together.

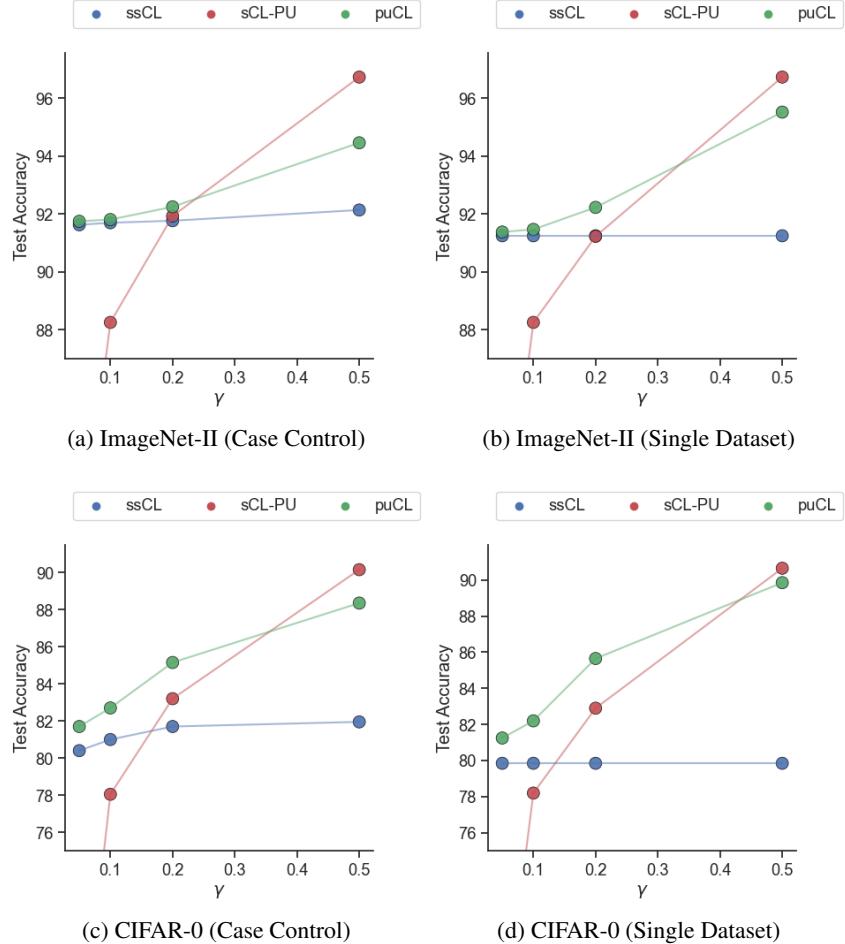


Figure 5: Generalization with varying supervision: In this experiment we train a ResNet-18 on CIFAR-0 (Subset of Dogs and Cats) and ImageNet-II (ImageWoof vs ImageNette) under both case-control and single-dataset PU Learning setting. For case control setting, number of unlabeled samples n_U is kept fixed while we vary the number of labeled positives n_P . On the other hand for the Single Dataset setting we keep the total number of samples fixed $N = n_P + n_U$ while varying n_P . In both settings, we find PUCL to remain quite robust across different levels of supervision while consistently outperforming its unsupervised counterpart SSCL and being competitive with SCL-PU even in high supervision regimes. While., SCL-PU suffers from large degradation especially in the low-supervision regime.

ARRANGING POINTS ON UNIT HYPERCUBE:

To further understand the behavior of interpolating between semantic annotation (labels) and semantic similarity (feature) - Consider 1D feature space $\mathbf{x} \in \mathbb{R}$, e.g., $\mathbf{x}_i = 1$ if shape: triangle (\blacktriangle , \blacktriangle), $\mathbf{x}_i = 0$ if shape: circle (\bullet , \bullet). However, the labels are $y_i = 1$ if color: blue (\blacktriangle , \bullet) and $y_i = 1$ if color: red (\blacktriangle , \bullet) i.e $p(\mathbf{x})$ contains no information about $p(y|\mathbf{x})$. Figure 7 shows several representative configurations (note that, other configurations are similar) of arranging these points on the vertices of unit hypercube $\mathcal{H} \in \mathbb{R}^2$ when \blacktriangle is fixed at $(0, 1)$.

- **Unsupervised objectives** e.g. ssCL (4) only rely on semantic similarity (feature) to learn embeddings, implying they attain minimum loss configuration when semantically similar objects $\mathbf{x}_i = \mathbf{x}_j$ are placed close to each other (neighboring vertices on \mathcal{H}^2) since this minimizes the inner product between representations of similar examples(Figure 7(a)).

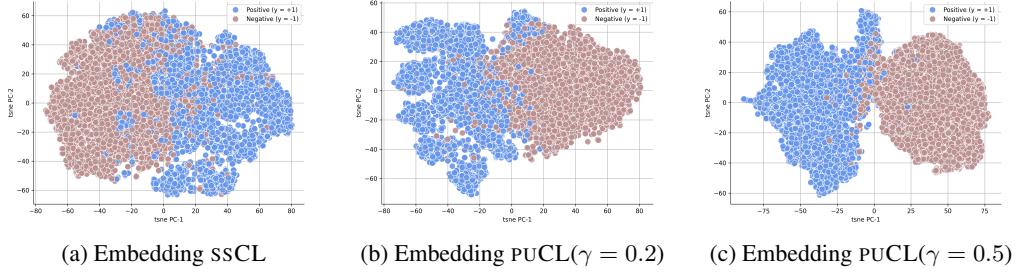


Figure 6: Embedding Quality with varying supervision: In this experiment we train a ResNet-18 on **ImageNet-II: ImageWoof vs ImageNette** - two subsets of ImageNet-1k widely used in noisy label learning research <https://github.com/fastai/imagenette>. Amount of supervision is measured with the ratio of labeled to unlabeled data $\gamma = \frac{n_p}{n_u}$. We keep the total number of samples $N = n_p + n_u$ fixed, while varying n_p . We observe that the embeddings obtained via PUCL exhibit significantly improved separability than that of the unsupervised baseline ssCL especially with increasing supervision.

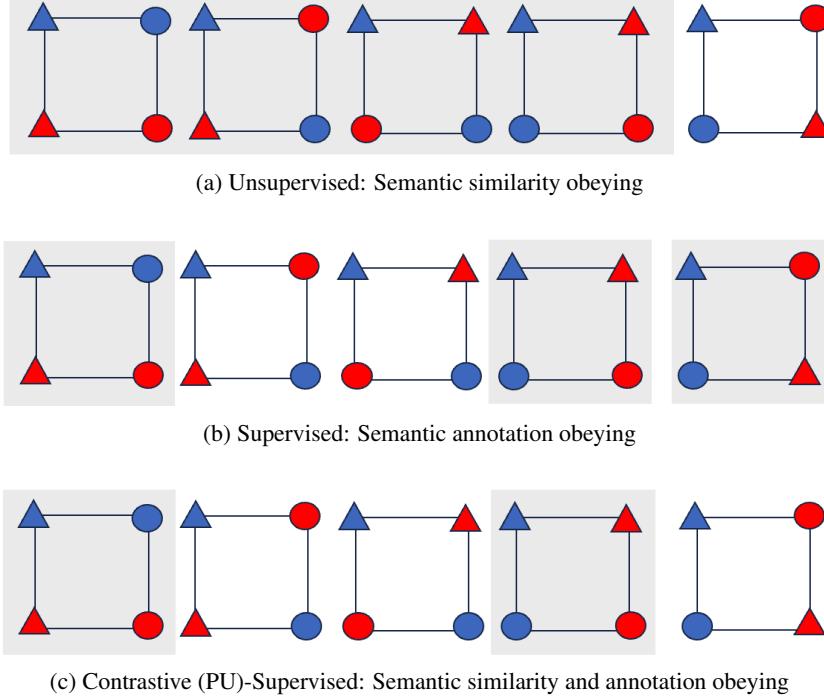


Figure 7: Geometric intuition of incorporating supervision: Consider 1D feature space $x \in \mathbb{R}$, e.g., $x_i = 1$ if shape: triangle ($\blacktriangle, \triangle$), $x_i = 0$ if shape: circle (\bullet, \circ). However, the labels are $y_i = 1$ if color: blue (\blacktriangle, \bullet) and $y_i = 1$ if color: red (\triangle, \circ). We show possible configurations (other configurations are similar) of arranging these points on the vertices of unit hypercube $\mathcal{H} \in \mathbb{R}^2$ when \blacktriangle is fixed at $(0, 1)$. (a) Unsupervised objectives e.g. ssCL only rely on semantic similarity (feature) to learn embeddings, implying they attain minimum loss configuration when semantically similar objects are placed close to each other (neighboring vertices on \mathcal{H}^2). (b) Supervised objectives on the other hand All the four shaded point configurations are favored by ssCL (4), since $x_i = x_j$ are placed neighboring vertices. However, the minimum loss configurations of PUCL (marked in rectangle) additionally also preserves annotation consistency.

- **Supervised objectives** e.g. CE on the other hand, updates the parameters such that the logits match the label. Thus purely supervised objectives attain minimum loss when objects sharing same annotation are placed next to each other (Figure 7(b)).

chain rule we have,

$$\frac{\partial \ell_i}{\partial \mathbf{w}} = \frac{\partial \ell_i}{\partial \mathbf{z}_i} \cdot \frac{\partial \mathbf{z}_i}{\partial \mathbf{w}} \quad (16)$$

Since, the second term depends on the encoder and fixed across the losses, the first term is sufficient to compare the gradients resulting from different losses. Thus, taking the differential of (15) w.r.t representation \mathbf{z}_i we get:

$$\begin{aligned} \frac{\partial \ell_i}{\partial \mathbf{z}_i} &= -\frac{1}{\tau} \left[\mathbf{z}_{a(i)} - \frac{\sum_{j \in \mathbb{I} \setminus \{i\}} \mathbf{z}_j \exp(\mathbf{z}_i \cdot \mathbf{z}_j / \tau)}{Z(\mathbf{z}_i)} \right] \\ &= -\frac{1}{\tau} \left[\mathbf{z}_{a(i)} - \frac{\mathbf{z}_{a(i)} \exp(\mathbf{z}_i \cdot \mathbf{z}_{a(i)} / \tau) + \sum_{j \in \mathbb{I} \setminus \{i, a(i)\}} \mathbf{z}_j \exp(\mathbf{z}_i \cdot \mathbf{z}_j / \tau)}{Z(\mathbf{z}_i)} \right] \\ &= -\frac{1}{\tau} \left[\mathbf{z}_{a(i)} \left(1 - \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_{a(i)} / \tau)}{Z(\mathbf{z}_i)} \right) - \sum_{j \in \mathbb{I} \setminus \{i, a(i)\}} \mathbf{z}_j \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_j / \tau)}{Z(\mathbf{z}_i)} \right] \\ &= -\frac{1}{\tau} \left[\mathbf{z}_{a(i)} \left(1 - \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_{a(i)} / \tau)}{Z(\mathbf{z}_i)} \right) - \sum_{j \in \mathbb{I} \setminus \{i, a(i)\}} \mathbf{z}_j \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_j / \tau)}{Z(\mathbf{z}_i)} \right] \\ &= -\frac{1}{\tau} \left[\mathbf{z}_{a(i)} (1 - P_{i,a(i)}) - \sum_{j \in \mathbb{I} \setminus \{i, a(i)\}} \mathbf{z}_j P_{i,j} \right] \end{aligned} \quad (17)$$

Where, the functions $P_{i,j}$ are defined as:

$$P_{i,j} = \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_j / \tau)}{Z(\mathbf{z}_i)} \quad (18)$$

Gradient derivation of PUCL Recall that, given a randomly sampled mini-batch \mathcal{D} , PUCL (6) takes the following form for any sample $i \in \mathbb{I}$ where \mathbb{I} is the corresponding multi-viewed batch. Let, $\mathbb{P}(i) = \mathbb{P} \setminus i$ i.e. all the other positive labeled examples in the batch w/o the anchor.

$$\begin{aligned} \ell_i &= -\frac{1}{|\mathbb{P}(i)|} \sum_{q \in \mathbb{P}(i)} \log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_q / \tau)}{Z(\mathbf{z}_i)} ; \forall i \in \mathbb{I} \\ &= -\frac{1}{|\mathbb{P}(i)|} \sum_{q \in \mathbb{P}(i)} \left[\frac{\mathbf{z}_i \cdot \mathbf{z}_q}{\tau} - \log Z(\mathbf{z}_i) \right] \end{aligned} \quad (19)$$

where $Z(\mathbf{z}_i)$ is defined as before. Then, we can compute the gradient w.r.t representation \mathbf{z}_i as:

$$\begin{aligned}
\frac{\partial \ell_i}{\partial \mathbf{z}_i} &= -\frac{1}{|\mathbb{P}(i)|} \sum_{q \in \mathbb{P}(i)} \left[\frac{\mathbf{z}_q}{\tau} - \frac{\partial Z(\mathbf{z}_i)}{Z(\mathbf{z}_i)} \right] \\
&= -\frac{1}{\tau |\mathbb{P}(i)|} \sum_{q \in \mathbb{P}(i)} \left[\mathbf{z}_q - \frac{\sum_{j \in \mathbb{I} \setminus \{i\}} \mathbf{z}_j \exp(\mathbf{z}_i \cdot \mathbf{z}_j / \tau)}{Z(\mathbf{z}_i)} \right] \\
&= -\frac{1}{\tau |\mathbb{P}(i)|} \sum_{q \in \mathbb{P}(i)} \left[\mathbf{z}_q - \sum_{q' \in \mathbb{P}(i)} \mathbf{z}_{q'} P_{i,q'} - \sum_{j \in \mathbb{U}(i)} \mathbf{z}_j P_{i,j} \right] \\
&= -\frac{1}{\tau |\mathbb{P}(i)|} \left[\sum_{q \in \mathbb{P}(i)} \mathbf{z}_q - \sum_{q \in \mathbb{P}(i)} \sum_{q' \in \mathbb{P}(i)} \mathbf{z}_{q'} P_{i,q'} - \sum_{q \in \mathbb{P}(i)} \sum_{j \in \mathbb{U}(i)} \mathbf{z}_j P_{i,j} \right] \\
&= -\frac{1}{\tau |\mathbb{P}(i)|} \left[\sum_{q \in \mathbb{P}(i)} \mathbf{z}_q - \sum_{q' \in \mathbb{P}(i)} |\mathbb{P}(i)| \mathbf{z}_{q'} P_{i,q'} - \sum_{j \in \mathbb{U}(i)} |\mathbb{P}(i)| \mathbf{z}_j P_{i,j} \right] \\
&= -\frac{1}{\tau} \left[\frac{1}{|\mathbb{P}(i)|} \sum_{q \in \mathbb{P}(i)} \mathbf{z}_q - \sum_{q \in \mathbb{P}(i)} \mathbf{z}_q P_{i,q} - \sum_{j \in \mathbb{U}(i)} \mathbf{z}_j P_{i,j} \right] \\
&= -\frac{1}{\tau} \left[\sum_{q \in \mathbb{P}(i)} \mathbf{z}_q \left(\frac{1}{|\mathbb{P}(i)|} - P_{i,q} \right) - \sum_{j \in \mathbb{U}(i)} \mathbf{z}_j P_{i,j} \right]
\end{aligned} \tag{20}$$

where we have defined $\mathbb{U}(i) = \mathbb{I} \setminus \{i, \mathbb{P}(i)\}$ i.e. $\mathbb{U}(i)$ is the set of all samples in the batch that are unlabeled.

In case of fully supervised setting we would similarly get:

$$\frac{\partial \ell_i}{\partial \mathbf{z}_i} = -\frac{1}{\tau} \left[\sum_{q \in \mathbb{P}(i)} \mathbf{z}_q \left(\frac{1}{|\mathbb{P}(i)|} - P_{i,q} \right) - \sum_{j \in \mathbb{N}(i)} \mathbf{z}_j P_{i,j} \right] \tag{21}$$

Since, in the fully supervised setting $\mathbb{I} - \mathbb{P}(i) = \mathbb{N}(i)$. Thus, by comparing the last term of the three gradient expressions, it is clear that PUCL enjoys lower bias compared to ssCL with respect to fully supervised counterpart. ■

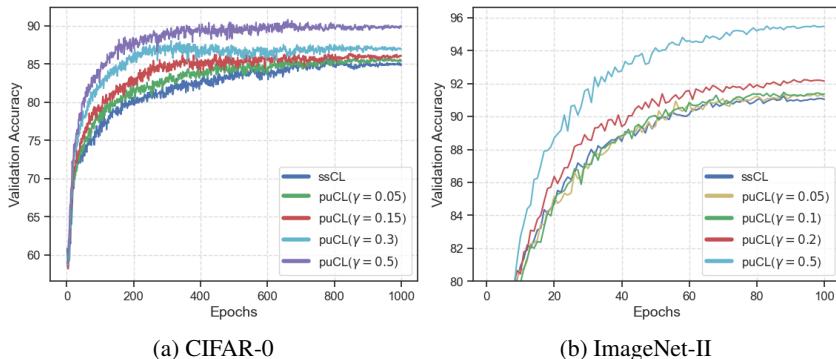


Figure 9: Convergence benefits from incorporating labeled positives: Training ResNet-18 on (a) **CIFAR-0:** Dogs vs Cats subsets from CIFAR10. (b) **ImageNet-II:** ImageWoof vs ImageNette subsets <https://github.com/fastai/imagenette> from ImageNet-1k. Observe that, by judiciously incorporating available labeled positives into the contrastive loss not only improves generalization it also improves convergence of contrastive representation learning from PU data as explained in Theorem 4

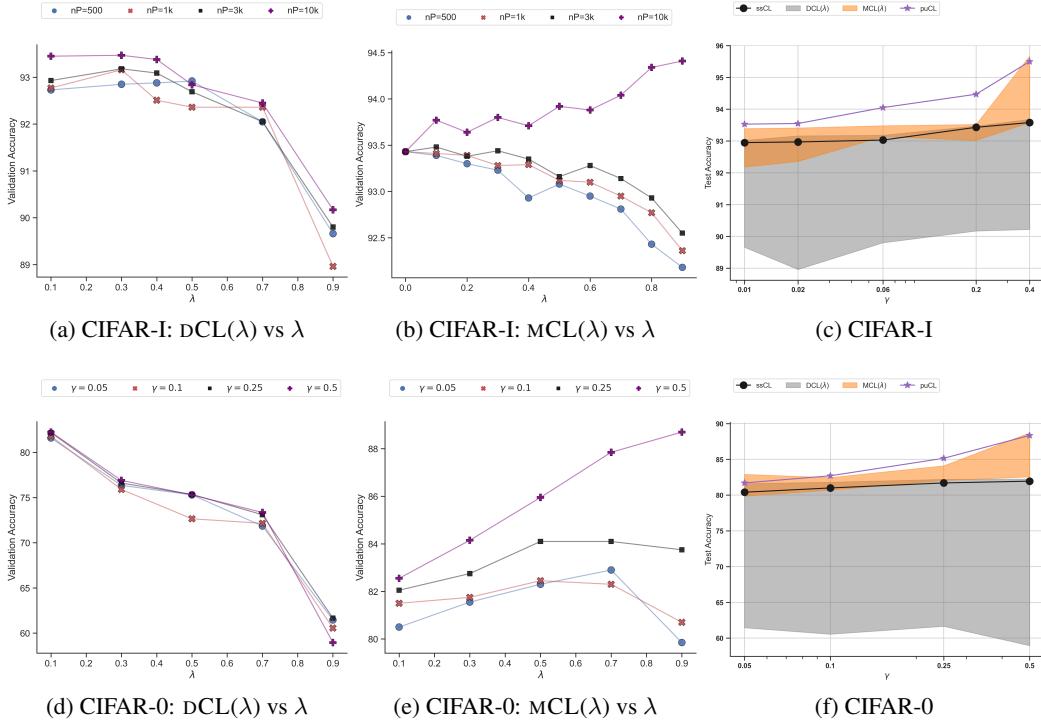


Figure 10: Parametric Weakly/Un-supervised Contrastive Objectives: The parametric objective MCL and DCL are both sensitive to the choice of the hyperparameter. Further, we note that the gains from unsupervised (with bias correction) DCL over ssCL is not as significant as the gains PUCL enjoys over ssCL, thanks to incorporating available positives. MCL however, for suitable choice of λ provide competitive performance as PUCL and in certain cases even does better. However, we note that the right choice of hyperparameter depends on the problem and data making it a less attractive choice to adopt in practice.

A.5.4 COMPARISON WITH PARAMETRIC CONTRASTIVE LEARNING OBJECTIVES:

Mixed Contrastive Learning (MCL): At a high level, the main idea of PUCL is to interpolate between the supervised and unsupervised objectives judiciously. By doing so, it is able to exploit the bias-variance trade-off as discussed in Section 2.2.

Another intuitive approach to interpolate between the robustness of ssCL and generalization benefits of SCL-PU could be to simply optimize over a convex combination of the two objectives. Such a hybrid objective has been proven effective in settings with label noise (i.e. when the labels are flipped with some constant probability) (Cui et al., 2023) and thus warrants investigation in the PU learning setting. We refer to this loss as Mixed Contrastive Loss (MCL) defined as follows:

$$\mathcal{L}_{\text{MCL}}(\lambda) = \lambda \mathcal{L}_{\text{SCL-PU}} + (1 - \lambda) \mathcal{L}_{\text{ssCL}}, \quad 0 \leq \lambda \leq 1 \quad (22)$$

Similar to PUCL; MCL combines two key components: the unsupervised part in MCL enforces consistency between representations learned via label-preserving augmentations (i.e. between \mathbf{z}_i and $\mathbf{z}_{a(i)} \forall i \in \mathbb{I}$), whereas the supervised component injects structural knowledge derived from available semantic annotation (labeled positives).

It is worth noting that, PUCL can be viewed as a special case of MCL where loss on unlabeled samples is equivalent to $\mathcal{L}_{\text{MCL}}(\lambda = 0)$ and on the labeled samples $\mathcal{L}_{\text{MCL}}(\lambda = 1)$ i.e.

$$\mathcal{L}_{\text{PUCL}} = \frac{1}{n} \sum_{i=1}^n \mathbf{1}(\mathbf{x}_i \in \mathbb{P}) \ell_{\text{MCL}}^i(1) + \mathbf{1}(\mathbf{x}_i \notin \mathbb{P}) \ell_{\text{MCL}}^i(0)$$

In the PU setting, since the structural knowledge of classes perceived by the disambiguation-free objective $\mathcal{L}_{\text{SCL-PU}}$ is noisy, the generalization performance of MCL is sensitive to the choice of

hyper-parameter λ . This can be attributed to a similar bias-variance trade-off argument as discussed before. We validate this intuition by extensive ablation experiments across various choices of λ (Figure 10) under different PU learning scenarios i.e. under varying levels of supervision (varying γ).

Our experiments suggest that, when available supervision is limited i.e. for small values of γ a smaller value of λ (i.e. less reliance on supervised part of the loss) is preferred. Conversely, for larger values of γ larger contribution from the supervised counterpart is necessary.

Since, the success of MCL is sensitive to the appropriate choice of λ , tuning which can be quite challenging and depends on the dataset and amount of available supervision, making MCL often less practical in the real world PU learning scenario. Thus, overall, PUCL is a more practical method as it alleviates the need for hyper-parameter tuning and works across various PU Learning scenarios while not suffering from performance degradation.

Debiased Contrastive Loss (DCL): Another popular approach to incorporate latent weak supervision in the unsupervised setting is via appropriately compensating for the sampling bias referred to as debiased contrastive learning (DCL) (Chuang et al., 2020).

Recall the infoNCE family of losses (3):

$$\mathcal{L}_{\text{CL}}^* = \mathbb{E}_{(\mathbf{x}_i, y_i) \sim p(\mathbf{x}, y)} \mathbb{E}_{\substack{\mathbf{x}_j \sim p(\mathbf{x} | y_j = y_i) \\ \{\mathbf{x}_k\}_{k=1}^N \sim p(\mathbf{x} | y_k \neq y_i)}} \left[\mathbf{z}_i \cdot \mathbf{z}_j - \log \left(\exp(\mathbf{z}_i \cdot \mathbf{z}_j) + \sum_{k=1}^N \exp(\mathbf{z}_i \cdot \mathbf{z}_k) \right) \right],$$

Further, recall that in the fully unsupervised setting, since no supervision is available the negatives are chosen as all the samples in the batch (Chen et al., 2020b) (4).

$$\begin{aligned} \mathcal{L}_{\text{SSCL}} &= -\frac{1}{|\mathbb{I}|} \sum_{i \in \mathbb{I}} \left[\mathbf{z}_i \cdot \mathbf{z}_{a(i)} - \log \sum_{j \in \mathbb{I}} \mathbf{1}(j \neq i) \exp(\mathbf{z}_i \cdot \mathbf{z}_j) \right] \\ &= -\frac{1}{|\mathbb{I}|} \sum_{i \in \mathbb{I}} \left[\mathbf{z}_i \cdot \mathbf{z}_{a(i)} - \underbrace{\log \left(\mathbf{z}_i \cdot \mathbf{z}_{a(i)} + \sum_{j \in \mathbb{I} \setminus \{i, a(i)\}} \exp(\mathbf{z}_i \cdot \mathbf{z}_j) \right)}_{R_N: \text{Negative pairs sum}} \right] \end{aligned}$$

Compared to $\mathcal{L}_{\text{CL}}^*$ this finite sample objective is biased since some of the samples treated as negative might belong to the same latent class as the anchor. (Chuang et al., 2020) refers to this phenomenon as sampling bias and propose a modified objective Debiased Contrastive Learning (DCL) to alleviate this issue. In particular, they follow (13) to form an estimate of the negative sum as:

$$R_n^- = \frac{1}{1-\lambda} \left[\hat{R}_u^-(v) - \lambda \hat{R}_p^-(v) \right]$$

λ is a hyper-parameter that needs to be tuned. Our experiments Figure 10 suggest that we note that the gains from unsupervised (with bias correction) DCL over SSCL is not as significant as the gains PUCL enjoys over SSCL, thanks to incorporating available positives. Moreover, we see that DCL is quite sensitive to the choice of hyperparameter making it hard to adopt in real world.

A.5.5 BIAS VARIANCE TRADEOFF

PROOF OF THEOREM 1.

We restate Theorem 1 for convenience -

Theorem 1. $\mathcal{L}_{\text{SCL-PU}}$ (5) is a biased estimator of $\mathcal{L}_{\text{CL}}^*$ characterized as follows:

$$\mathbb{E}_{\mathcal{X}_{\text{PU}}} \left[\mathcal{L}_{\text{SCL-PU}} \right] - \mathcal{L}_{\text{CL}}^* = \frac{\pi_p(1-\pi_p)}{1+\gamma} \left[2\tilde{\mu}_{\text{PN}} - (\mu_{\text{P}}^* + \mu_{\text{N}}^*) \right]$$

Here, $\mu_{\text{P}}^* = \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j \sim p(\mathbf{x} | y=1)} (\mathbf{z}_i \cdot \mathbf{z}_j)$ and $\mu_{\text{N}}^* = \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j \sim p(\mathbf{x} | y=0)} (\mathbf{z}_i \cdot \mathbf{z}_j)$ capture the proximity between samples from same class marginals and $\tilde{\mu}_{\text{PN}} = \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j \sim p(\mathbf{x} | y_i \neq y_j)} (\mathbf{z}_i \cdot \mathbf{z}_j)$ captures the

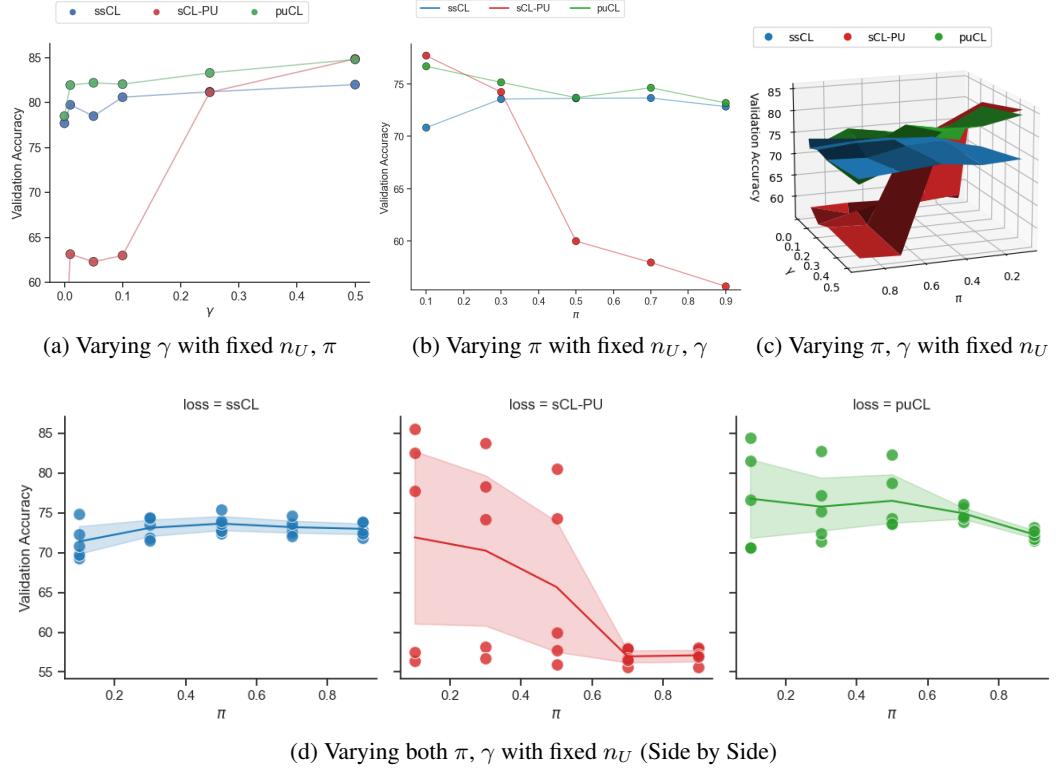


Figure 11: Ablation of Non-Parametric Contrastive Losses under different PU settings: To better understand the bias-variance trade-off we experiment with different PU learning settings: class prior π_p and amount of labeled data captured by $\gamma = \frac{n_p}{n_U}$. Experiments train ResNet-34 on ImageNet Dogs vs Non-Dogs. Embedding evaluation was performed using fully supervised kNN classification. Overall these experiments indicate that unlike SCL, PUCL and ssCL remain robust across various PU learning settings wherein, PUCL enjoys superior generalization performance on downstream classification. Further, they flesh out several interesting aspects of contrastive learning over PU data and supplement our theoretical findings. Please refer to Section 3 for a detailed discussion.

proximity between dissimilar samples. $\gamma = \frac{n_p}{n_U}$ captures the proportion of cardinality of labeled to unlabeled training subset.

Proof. Suppose, \mathcal{X}_{PU} is generated from the underlying supervised dataset $\mathcal{X}_{PN} = \mathcal{X}_P \cup \mathcal{X}_N$ i.e. labeled positives \mathcal{X}_{P_L} is a subset of n_{P_L} elements chosen uniformly at random from all subsets of \mathcal{X}_P of size $n_L : \mathcal{X}_{P_L} \subset \mathcal{X}_P = \{\mathbf{x}_i \in \mathbb{R}^d \sim p(\mathbf{x}|y=1)\}_{i=1}^{n_p}$. Further, denote the set positive and negative examples that are unlabeled as \mathcal{X}_{P_U} and \mathcal{X}_{N_U} .

$$\mathcal{X}_{PU} = \mathcal{X}_{P_L} \cup \mathcal{X}_{P_U} \cup \mathcal{X}_{N_U}, \quad \mathcal{X}_P = \mathcal{X}_{P_L} \cup \mathcal{X}_{P_U} \text{ and } \mathcal{X}_U = \mathcal{X}_{P_U} \cup \mathcal{X}_{N_U} \quad (23)$$

Now, we can establish the result by carefully analyzing the bias of \mathcal{L}_{SCL-PU} (5) in estimating the ideal contrastive loss (3) over each of these subsets.

For the labeled positive subset \mathcal{X}_{P_L} the bias can be computed as:

$$\mathcal{B}_{\mathcal{L}_{SCL-PU}}(\mathbf{x}_i \in \mathcal{X}_{P_L}) = \mathbb{E}_{\mathbf{x}_i \in \mathcal{X}_{P_L}} \left[\frac{1}{n_{P_L}} \sum_{\mathbf{x}_j \in \mathcal{X}_{P_L}} \mathbf{z}_i \cdot \mathbf{z}_j \right] - \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j \sim p(\mathbf{x}|y=1)} \left[\mathbf{z}_i \cdot \mathbf{z}_j \right] \quad (24)$$

Clearly under the PU setting (1), since the labeled positives are i.i.d samples from the positive marginal the first expectation the two expectations align implying the zero bias. However obviously the variance scales inversely with the amount of labeled positives $\mathcal{O}(\frac{1}{|\mathcal{X}_{P_L}|})$.

For the unlabeled positive subset \mathcal{X}_{P_U} the bias can be computed as:

$$\begin{aligned}
\mathcal{B}_{L_{SCL-PU}}(\mathbf{x}_i \in \mathcal{X}_{P_U}) &= \mathbb{E}_{\mathbf{x}_i \in \mathcal{X}_{P_U}} \left[\frac{1}{n_U} \sum_{\mathbf{x}_j \in \mathcal{X}_U} \mathbf{z}_i \cdot \mathbf{z}_j \right] - \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j \sim p(x|y=1)} \left[\mathbf{z}_i \cdot \mathbf{z}_j \right] \\
&= \mathbb{E}_{\mathbf{x}_i \in \mathcal{X}_{P_U}} \left[\pi_p \mathbb{E}_{\mathbf{x}_j \in \mathcal{X}_{P_U}} \left(\mathbf{z}_i \cdot \mathbf{z}_j \right) + (1 - \pi_p) \mathbb{E}_{\mathbf{x}_j \in \mathcal{X}_{N_U}} \left(\mathbf{z}_i \cdot \mathbf{z}_j \right) \right] - \mu_P^* \\
&= \pi_p \mu_P^* + (1 - \pi_p) \mathbb{E}_{\mathbf{x}_i \in \mathcal{X}_{P_U}} \left[\mathbb{E}_{\mathbf{x}_j \in \mathcal{X}_{N_U}} \left(\mathbf{z}_i \cdot \mathbf{z}_j \right) \right] - \mu_P^* \\
&= (1 - \pi_p) \mathbb{E}_{\mathbf{x}_i \sim p(x|y=1)} \left[\mathbb{E}_{\mathbf{x}_j \sim p(x|y=0)} \left(\mathbf{z}_i \cdot \mathbf{z}_j \right) \right] - (1 - \pi_p) \mu_P^* \\
&= (1 - \pi_p) \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j \sim p(x|y_i \neq y_j)} \left(\mathbf{z}_i \cdot \mathbf{z}_j \right) - (1 - \pi_p) \mu_P^* \\
&= (1 - \pi_p) \tilde{\mu}_{PN} - (1 - \pi_p) \mu_P^*
\end{aligned}$$

Finally, for the negative unlabeled set:

$$\begin{aligned}
\mathcal{B}_{L_{SCL-PU}}(\mathbf{x}_i \in \mathcal{X}_{N_U}) &= \mathbb{E}_{\mathbf{x}_i \in \mathcal{X}_{N_U}} \left[\frac{1}{n_U} \sum_{\mathbf{x}_j \in \mathcal{X}_U} \mathbf{z}_i \cdot \mathbf{z}_j \right] - \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j \sim p(x|y=0)} \left[\mathbf{z}_i \cdot \mathbf{z}_j \right] \\
&= \mathbb{E}_{\mathbf{x}_i \in \mathcal{X}_{N_U}} \left[\pi_p \mathbb{E}_{\mathbf{x}_j \in \mathcal{X}_{P_U}} \left(\mathbf{z}_i \cdot \mathbf{z}_j \right) + (1 - \pi_p) \mathbb{E}_{\mathbf{x}_j \in \mathcal{X}_{N_U}} \left(\mathbf{z}_i \cdot \mathbf{z}_j \right) \right] - \mu_N^* \\
&= \pi_p \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j \sim p(x|y_i \neq y_j)} \left(\mathbf{z}_i \cdot \mathbf{z}_j \right) + (1 - \pi_p) \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j \sim p(x|y=0)} \left(\mathbf{z}_i \cdot \mathbf{z}_j \right) - \mu_N^* \\
&= \pi_p \tilde{\mu}_{PN} - \pi_p \mu_N^*
\end{aligned}$$

Now, using the fact that the unlabeled examples are sampled uniformly at random from the mixture distribution with positive mixture weight π_p we can compute the total bias as follows:

$$\mathcal{B}_{L_{SCL-PU}}(\mathbf{x}_i \in \mathcal{X}_{P_U}) = \frac{\pi_p}{1 + \gamma} \mathcal{B}_{L_{SCL-PU}}(\mathbf{x}_i \in \mathcal{X}_{P_U}) + \frac{1 - \pi_p}{1 + \gamma} \mathcal{B}_{L_{SCL-PU}}(\mathbf{x}_i \in \mathcal{X}_{N_U}) \text{ where } \gamma = \frac{|\mathcal{X}_{P_U}|}{|\mathcal{X}_U|}$$

We get the desired result by plugging in the bias of the subsets and simplifying. \blacksquare

PROOF OF THEOREM 2.

We restate Theorem 1, 2 for convenience -

Theorem 2. Assume that $\mathbf{x}_i, \mathbf{x}_{a(i)}$ are i.i.d draws from the same class marginal (Saunshi et al., 2019; Tosh et al., 2021), then it follows that the objective functions \mathcal{L}_{SSCL} (4) and \mathcal{L}_{PUCL} (6) are unbiased estimators of \mathcal{L}_{CL}^* (3). Additionally, it holds that:

$$\Delta_\sigma(\gamma) \geq 0 \quad \forall \gamma \geq 0 ; \quad \Delta_\sigma(\gamma_1) \geq \Delta_\sigma(\gamma_2) \quad \forall \gamma_1 \geq \gamma_2 \geq 0$$

where, $\Delta_\sigma(\gamma) = \text{Var}(\mathcal{L}_{SSCL}) - \text{Var}(\mathcal{L}_{PUCL})$.

Proof. We first prove that both \mathcal{L}_{SSCL} (4) and \mathcal{L}_{PUCL} (6) are unbiased estimators of \mathcal{L}_{CL}^* (3) using a similar analysis as the previous proof.

For the labeled positive subset \mathcal{X}_{P_L} the bias can be computed as:

$$\mathcal{B}_{L_{PUCL}}(\mathbf{x}_i \in \mathcal{X}_{P_L}) = \mathbb{E}_{\mathbf{x}_i \in \mathcal{X}_{P_L}} \left[\frac{1}{n_{P_L}} \sum_{\mathbf{x}_j \in \mathcal{X}_{P_L}} \mathbf{z}_i \cdot \mathbf{z}_j \right] - \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j \sim p(x|y=1)} \left[\mathbf{z}_i \cdot \mathbf{z}_j \right] = 0$$

Here we have used the fact that labeled positives are drawn i.i.d from the positive marginal. For the unlabeled samples

$$\begin{aligned}\mathcal{B}_{\mathcal{L}_{\text{PUCL}}}(\mathbf{x}_i \in \mathcal{X}_{\text{U}}) &= \mathbb{E}_{\mathbf{x}_i \in \mathcal{X}_{\text{U}}} \left[\mathbf{z}_i \cdot \mathbf{z}_{a(i)} \right] - \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j \sim p(\mathbf{x} | y_i = y_j)} \left[\mathbf{z}_i \cdot \mathbf{z}_j \right] \\ &= \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j \sim p(\mathbf{x} | y_i = y_j)} \left[\mathbf{z}_i \cdot \mathbf{z}_j \right] - \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j \sim p(\mathbf{x} | y_i = y_j)} \left[\mathbf{z}_i \cdot \mathbf{z}_j \right] = 0\end{aligned}$$

Thus $\mathcal{L}_{\text{PUCL}}$ is an unbiased estimator of $\mathcal{L}_{\text{CL}}^*$. Clearly, the i.i.d assumption similarly implies $\mathcal{L}_{\text{SSCL}}$ is also an unbiased estimator.

Next we can do a similar decomposition of the variances for both the objectives. Then the difference of variance under the PU dataset -

$$\begin{aligned}\Delta_{\sigma}(\mathcal{X}_{\text{PU}}) &= \text{Var}_{\mathcal{L}_{\text{SSCL}}}(\mathcal{X}_{\text{PU}}) - \text{Var}_{\mathcal{L}_{\text{PUCL}}}(\mathcal{X}_{\text{PU}}) \\ &= \Delta_{\sigma}(\mathcal{X}_{\text{P}_L}) + \Delta_{\sigma}(\mathcal{X}_{\text{U}}) \\ &= \Delta_{\sigma}(\mathcal{X}_{\text{P}_L}) \\ &= \left(1 - \frac{1}{n_{\text{P}_L}}\right) \text{Var}\left(\mathbf{z}_i \cdot \mathbf{z}_j : \mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}_{\text{P}_L}\right) \\ &= \left(1 - \frac{1}{\gamma |\mathcal{X}_{\text{U}}|}\right) \text{Var}\left(\mathbf{z}_i \cdot \mathbf{z}_j : \mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}_{\text{P}_L}\right)\end{aligned}$$

Clearly, since variance is non-negative we have $\forall \gamma > 0 : \Delta_{\sigma}(\mathcal{X}_{\text{PU}}) \geq 0$

Now consider two settings where we have different amounts of labeled positives defined by ratios γ_1 and γ_2 and denote the two resulting datasets $\mathcal{X}_{\text{PU}}^{\gamma_1}$ and $\mathcal{X}_{\text{PU}}^{\gamma_2}$ then

$$\begin{aligned}\Delta_{\sigma}(\mathcal{X}_{\text{PU}}^{\gamma_1}) - \Delta_{\sigma}(\mathcal{X}_{\text{PU}}^{\gamma_2}) &= \Delta_{\sigma}(\mathcal{X}_{\text{P}_L}^{\gamma_1}) - \Delta_{\sigma}(\mathcal{X}_{\text{P}_L}^{\gamma_2}) \\ &= \frac{1}{|\mathcal{X}_{\text{U}}|} \left(\frac{1}{\gamma_2} - \frac{1}{\gamma_1} \right) \text{Var}\left(\mathbf{z}_i \cdot \mathbf{z}_j : \mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}_{\text{P}_L}\right) \\ &\geq 0\end{aligned}$$

The last inequality holds since $\gamma_1 \geq \gamma_2$. This concludes the proof. ■

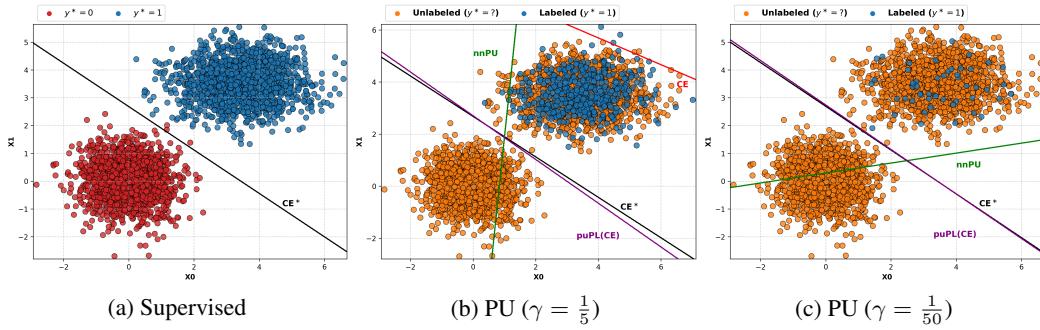


Figure 12: **Geometric Intuition of PUPL (separable):** We train logistic regression over (almost) separable 2D Gaussian Mixture. **CE*** denote the supervised classifier for comparison with the decision boundaries obtained by **CE**, **NNPU** (trained with π_p^*) and **PUPL(CE)**. It is clear that, as $\gamma = \frac{n_p}{n_U}$ is decreased **CE** soon diverges; **NNPU** suffers from significant decision boundary deviation. On the other hand, **PUPL(CE)** almost surely remains close to the true decision boundary as long as the feature space displays inherent clustering structure.

A.6 PUPL: POSITIVE UNLABELED PSEUDO LABELING

Summary: Consider performing PU learning over a (almost) linearly separable feature space (i.e., where the true positives and true negatives form separate clusters).

- Standard supervised classification loss, e.g., CE, suffers from decision boundary deviation when the number of labeled examples is limited.
- Cost-sensitive PU learning approaches address this issue by forming an unbiased risk estimator by leveraging the unlabeled and labeled positives (see Appendix A.3.3). However, we observe that when only a handful of positives are labeled, even these approaches are unable to recover the ideal decision boundary as the unbiased estimate suffers from large variance.
- Our proposed approach PUPL (Algorithm 1(B)) on the other hand, is able to identify the correct pseudo-labels (i.e. cluster assignments) almost surely (within constant multiplicative approximation error).
- Consequently training using the pseudo-labels with standard classification loss e.g. CE loss often achieves decision boundaries that closely align with the true boundaries.

Consider the naive disambiguation-free setup where the unlabeled samples are treated as pseudo-negatives and the classifier is trained using standard CE loss. As demonstrated in Figure 14, the bias induced via the pseudo-labeling results in the decision boundary to deviate further from the true (fully supervised) decision boundary when only a limited amount of labeled examples are available.

A.6.1 NECESSARY DEFINITIONS AND INTERMEDIATE LEMMAS

Before proceeding with the proofs we would need to make some definitions more formal and state some intermediate results.

Definition 2 (Supervised Dataset). Let \mathcal{X}_{PN} denote the true underlying fully supervised (PN) dataset

$$\mathcal{X}_{\text{PN}} = \{\mathbf{x}_i \sim p(\mathbf{x})\}_{i=1}^n = \mathcal{X}_P \cup \mathcal{X}_N, \quad \mathcal{X}_P = \{\mathbf{x}_i^P\}_{i=1}^{n_P} \stackrel{i.i.d.}{\sim} p_p(\mathbf{x}), \quad \mathcal{X}_N = \{\mathbf{x}_i^N\}_{i=1}^{n_N} \stackrel{i.i.d.}{\sim} p_n(\mathbf{x})$$

where, $p(\mathbf{x})$ denotes the underlying true mixture distribution; $p_p(\mathbf{x}) = p(\mathbf{x}|y=1)$ and $p_n(\mathbf{x}) = p(\mathbf{x}|y=0)$ denote the underlying positive and negative class marginal respectively.

Definition 3 (Class Prior). The mixture component weights of $p(\mathbf{x})$ are π_p and $\pi_n = 1 - \pi_p$.

$$p(\mathbf{x}) = \pi_p p_p(\mathbf{x}) + (1 - \pi_p) p_n(\mathbf{x}) \quad \text{where, } \pi_p = p(y=1|\mathbf{x})$$

Definition 4 (Positive Unlabeled Dataset). Let $p(\mathbf{x})$ denotes the underlying true mixture distribution of positive and negative class with class prior $\pi_p = p(y=1|\mathbf{x})$. Further let, $p_p(\mathbf{x}) = p(\mathbf{x}|y=1)$ and $p_n(\mathbf{x}) = p(\mathbf{x}|y=0)$ denote the true underlying positive and negative class marginal respectively.

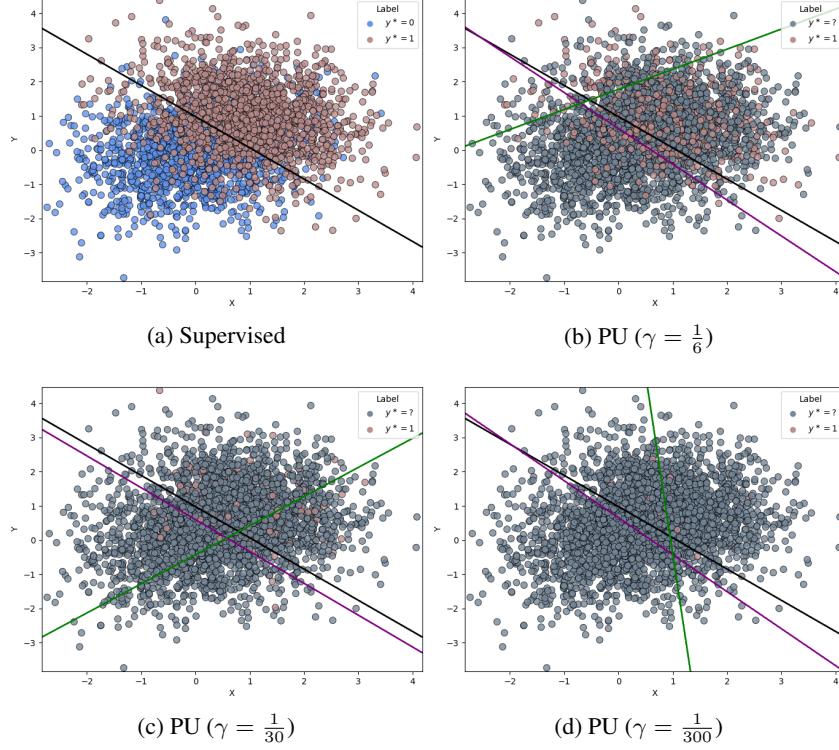


Figure 13: **Geometric Intuition of PUPL (overlapping):** Here we aim to learn a linear classifier over overlapping gaussians. We note that, puPL matches the bounday of supervised CE even in this setting.

Then the PU dataset is generated as:

$$\mathcal{X}_{\text{PU}} = \mathcal{X}_{\text{P}_L} \cup \mathcal{X}_{\text{U}}, \quad \mathcal{X}_{\text{P}_L} = \{\mathbf{x}_i^{\text{P}}\}_{i=1}^{n_{\text{P}_L}} \stackrel{i.i.d.}{\sim} p_p(\mathbf{x}), \quad \mathcal{X}_{\text{U}} = \{\mathbf{x}_i^{\text{U}}\}_{i=1}^{n_u} \stackrel{i.i.d.}{\sim} p(\mathbf{x})$$

Definition 5 (Clustering). A clustering refers to a set of centroids $C = \{\mu_{\text{P}}, \mu_{\text{N}}\}$ that defines the following pseudo-labels to the unlabeled instances:

$$\forall \mathbf{z}_i \in \mathcal{Z}_{\text{U}} : \tilde{y}_i = \begin{cases} 1, & \text{if } \mu_{\text{P}} = \arg \min_{\mu \in C} \|\mathbf{z}_i - \mu\|^2 \\ 0, & \text{otherwise} \end{cases}$$

Definition 6 (Potential Function). Given a clustering C the potential function over the dataset is:

$$\phi(\mathcal{Z}_{\text{PU}}, C) = \sum_{\mathbf{z}_i \in \mathcal{Z}_{\text{PU}}} \min_{\mu \in C} \|\mathbf{z}_i - \mu\|^2, \quad \mathcal{Z}_{\text{PU}} = \{\mathbf{z}_i = g_{\mathbf{B}}(\mathbf{x}_i) \in \mathbb{R}^k : \mathbf{x}_i \in \mathcal{X}_{\text{PU}}\}$$

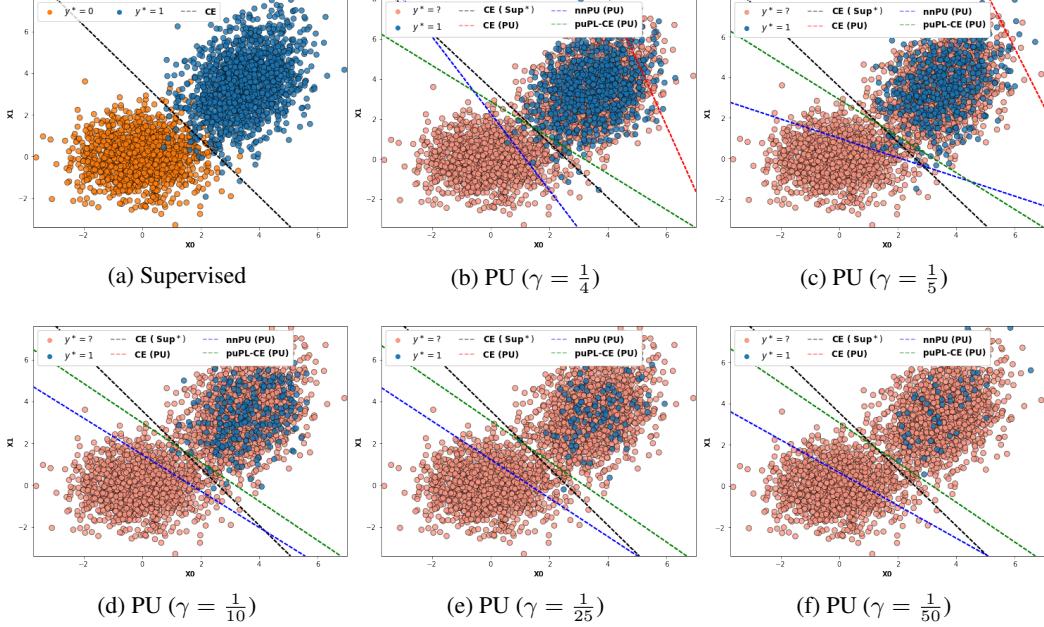
Definition 7 (Optimal Clustering). Refers to the optimal clustering $C^* = \{\mu_{\text{P}}^*, \mu_{\text{N}}^*\}$ that solves the k-means problem i.e. attains the minimum potential function:

$$\phi^*(\mathcal{Z}_{\text{PU}}, C^*) = \sum_{\mathbf{z}_i \in \mathcal{Z}_{\text{PU}}} \min_{\mu \in C^*} \|\mathbf{z}_i - \mu\|^2, \quad \mathcal{Z}_{\text{PU}} = \{\mathbf{z}_i = g_{\mathbf{B}}(\mathbf{x}_i) \in \mathbb{R}^k : \mathbf{x}_i \in \mathcal{X}_{\text{PU}}\}$$

Definition 8 (D^2). Given clustering C the $D^2(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^+$ score is:

$$D^2(\mathbf{x}) = \phi(\{\mathbf{x}\}, C)$$

Central to the analysis is the following two lemmas:

Figure 14: **Decision Boundary Deviation** : Training a linear classifier over 2D Gaussian Mixture.

(0) CE* - Denotes the ideal supervised classification boundary.

(1) CE - We consider the standard disambiguation-free CE loss - wherein the unlabeled samples are simply treated as pseudo-negatives and a binary classifier is trained to separate the labeled positives from these pseudo negatives (unlabeled) examples. CE loss is unable to recover the true decision boundary (i.e. the decision boundary learnt in the fully supervised setting) in this setting. In fact as amount of labeled positives decrease (i.e. for smaller γ) clearly the decision boundary deviates dramatically from the true boundary due to the biased supervision and eventually diverges.

(2) nnPU - Models trained with nnPU objective (Kiryo et al., 2017). We note that nnPU is significantly more robust than the naive disambiguation-free approach. Especially when sufficient labeled positives are provided the decision boundary learnt by nnPU is closely aligned with the true decision boundary. However, when only a handful of positives are labeled we observe that nnPU might also result in significant generation gap possibly because the variance of the estimator is high in this case. Note that, *All nnPU experiments here are run with oracle knowledge of class prior information $\pi_p^* = \frac{1}{2}$* .

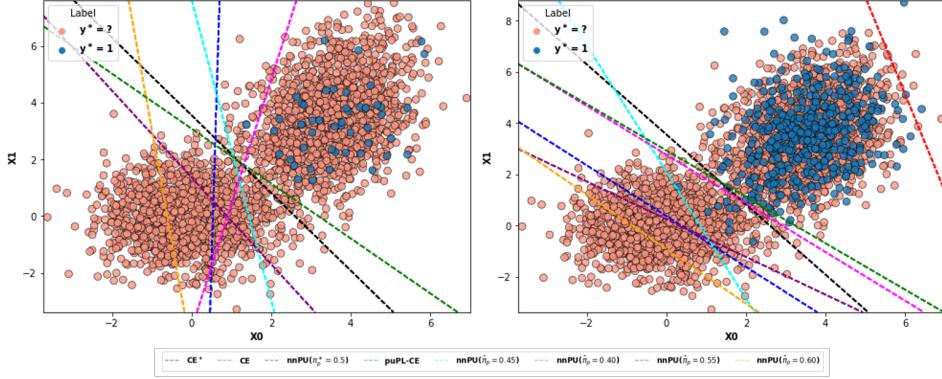
(3) puPL + CE - On the other hand our clustering based pseudo-labeling approach almost surely recovers the true underlying labels even when only a few positive examples are available resulting in consistent improvement over existing SOTA cost-sensitive approaches. Further our approach obviates the need of class prior knowledge unlike nnPU.

Lemma 2 (Positive Centroid Estimation). Suppose, \mathcal{Z}_{P_L} is a subset of n_L elements chosen uniformly at random from all subsets of \mathcal{Z}_P of size $n_L : \mathcal{Z}_{\text{P}_L} \subset \mathcal{Z}_P = \{\mathbf{z}_i = g_{\mathbf{B}}(\mathbf{x}_i) \in \mathbb{R}^k : \mathbf{x}_i \in \mathbb{R}^d \sim p(\mathbf{x}|y=1)\}_{i=1}^{n_P}$ implying that the labeled positives are generated according to (1). Let, μ denote the centroid of \mathcal{Z}_{P_L} i.e. $\mu = \frac{1}{n_{\text{P}_L}} \sum_{\mathbf{z}_i \in \mathcal{Z}_{\text{P}_L}} \mathbf{z}_i$ and μ^* denote the optimal centroid of \mathcal{Z}_P i.e. $\phi^*(\mathcal{Z}_P, \mu^*) = \sum_{\mathbf{z}_i \in \mathcal{Z}_{\text{P}_U}} \|\mathbf{z}_i - \mu^*\|^2$ then we can establish the following result:

$$\mathbb{E} \left[\phi(\mathcal{Z}_P, \mu) \right] = \left(1 + \frac{n_P - n_{\text{P}_L}}{n_{\text{P}_L}(n_P - 1)} \right) \phi^*(\mathcal{Z}_P, \mu^*)$$

Lemma 3 (k-means++ Seeding). Given initial cluster center $\mu_P = \frac{1}{n_{\text{P}_L}} \sum_{\mathbf{z}_i \in \mathcal{Z}_{\text{P}_L}} \mathbf{z}_i$, if the second centroid μ_N is chosen according to the distribution $D(\mathbf{z}) = \frac{\phi(\{\mathbf{z}\}, \{\mu_P\})}{\sum_{\mathbf{z} \in \mathcal{Z}_U} \phi(\{\mathbf{z}\}, \{\mu_P\})} \forall \mathbf{z} \in \mathcal{Z}_U$, then:

$$\mathbb{E} \left[\phi(\mathcal{Z}_{\text{P}_U}, \{\mu_P, \mu_N\}) \right] \leq 2\phi(\mathcal{Z}_{\text{P}_L}, \{\mu_P\}) + 16\phi^*(\mathcal{Z}_U, C^*)$$



A.6.2 PROOF OF THEOREM 3.

We restate Theorem 3 for convenience -

Theorem 3. Suppose, PU data is generated as (1), then running Algorithm I(B) on \mathcal{Z}_{PU} yields: $\mathbb{E}[\phi(\mathcal{Z}_{\text{PU}}, C_{\text{PUPL}})] \leq 16\phi^*(\mathcal{Z}_{\text{PU}}, C^*)$. In comparison, running k-means++ on \mathcal{Z}_{PU} we get, $\mathbb{E}[\phi(\mathcal{Z}_{\text{PU}}, C_{\text{k-means++}})] \leq 21.55\phi^*(\mathcal{Z}_{\text{PU}}, C^*)$.

We will closely follows the proof techniques from (Arthur & Vassilvitskii, 2007) *mutatis mutandis* to prove this theorem.

Proof. Recall that we choose our first center from supervision i.e. $\mu_P = \frac{1}{n_{P_L}} \sum_{\mathbf{z}_i \in \mathcal{Z}_{P_L}} \mathbf{z}_i$ and then choose the next center from the unlabeled samples according to probability $D(\mathbf{z}) = \frac{\phi(\{\mathbf{z}\}, \{\mu_P\})}{\sum_{\mathbf{z} \in \mathcal{Z}_U} \phi(\{\mathbf{z}\}, \{\mu_P\})} \forall \mathbf{z} \in \mathcal{Z}_U$. Then, from Lemma 3:

$$\begin{aligned} \mathbb{E} \left[\phi(\mathcal{Z}_{\text{PU}}, \{\mu_P, \mu_N\}) \right] &\leq 2\phi(\mathcal{Z}_{P_L}, \{\mu_P\}) + 16\phi^*(\mathcal{Z}_U, C^*) \\ &= 2\phi(\mathcal{Z}_{P_L}, \{\mu_P\}) + 16 \left(\phi^*(\mathcal{Z}_{\text{PU}}, C^*) - \phi^*(\mathcal{Z}_{P_L}, C^*) \right) \\ &= 2 \left(\phi(\mathcal{Z}_{P_L}, \{\mu_P\}) - 8\phi^*(\mathcal{Z}_{P_L}, \{\mu_P^*\}) \right) + 16\phi^*(\mathcal{Z}_{\text{PU}}, C^*) \end{aligned}$$

Now we use Lemma 2 to bound the first term -

$$\begin{aligned} \mathbb{E} \left[\phi(\mathcal{Z}_{\text{PU}}, \{\mu_P, \mu_N\}) \right] &\leq 2 \left[\left(1 + \frac{n_P - n_{P_L}}{n_{P_L}(n_P - 1)} \right) - 8 \right] \phi^*(\mathcal{Z}_{P_L}, \{\mu_P^*\}) + 16\phi^*(\mathcal{Z}_{\text{PU}}, C^*) \\ &\leq 2 \left[\frac{n_P - n_{P_L}}{n_{P_L}(n_P - 1)} - 7 \right] \phi^*(\mathcal{Z}_{P_L}, \{\mu_P^*\}) + 16\phi^*(\mathcal{Z}_{\text{PU}}, C^*) \\ &\leq 16\phi^*(\mathcal{Z}_{\text{PU}}, C^*) \end{aligned}$$

Note that this bound is much tighter in practice when a large amount of labeled examples are available i.e. for larger values of n_{P_L} . Additionally our guarantee holds only after the initial cluster assignments are found. Subsequent standard k-means iterations can only further decrease the potential.

On the other hand for k -means++ strategy (Arthur & Vassilvitskii, 2007) the guarantee is:

$$\mathbb{E} \left[\phi(\mathcal{Z}_{\text{PU}}, C_{k-\text{means++}}) \right] \leq \left(2 + \ln 2 \right) 8\phi^*(\mathcal{Z}_{\text{PU}}, C^*) \approx 21.55\phi^*(\mathcal{Z}_{\text{PU}}, C^*)$$

This concludes the proof. ■

A.6.3 PROOF OF LEMMA 2

Proof.

$$\begin{aligned} \mathbb{E} \left[\phi(\mathcal{Z}_P, \mu) \right] &= \mathbb{E} \left[\sum_{\mathbf{z}_i \in \mathcal{Z}_P} \|\mathbf{z}_i - \mu\|^2 \right] \\ &= \mathbb{E} \left[\sum_{\mathbf{z}_i \in \mathcal{Z}_P} \|\mathbf{z}_i - \mu^*\|^2 + n_P \|\mu - \mu^*\|^2 \right] \\ &= \phi^*(\mathcal{Z}_P, \mu^*) + n_P \mathbb{E} \left[\|\mu - \mu^*\|^2 \right] \end{aligned}$$

Now we can compute the expectation as:

$$\begin{aligned} \mathbb{E} \left[\|\mu - \mu^*\|^2 \right] &= \mathbb{E} \left[\mu^T \mu \right] + \mu^{*T} \mu^* - 2\mu^{*T} \mathbb{E} \left[\frac{1}{n_{P_L}} \sum_{\mathbf{z}_i \in \mathcal{Z}_{P_L}} \mathbf{z}_i \right] \\ &= \mathbb{E} \left[\mu^T \mu \right] + \mu^{*T} \mu^* - 2\mu^{*T} \frac{1}{n_{P_L}} \mathbb{E} \left[\sum_{\mathbf{z}_i \in \mathcal{Z}_{P_L}} \mathbf{z}_i \right] \\ &= \mathbb{E} \left[\mu^T \mu \right] + \mu^{*T} \mu^* - 2\mu^{*T} \frac{1}{n_{P_L}} n_{P_L} \mathbb{E}_{\mathbf{z}_i \in \mathcal{Z}_P} \left[\mathbf{z}_i \right] \\ &= \mathbb{E} \left[\mu^T \mu \right] - \mu^{*T} \mu^* \end{aligned}$$

We can compute the first expectation as:

$$\begin{aligned} \mathbb{E} \left[\mu^T \mu \right] &= \frac{1}{n_{P_L}^2} \mathbb{E} \left[\left(\sum_{\mathbf{z}_i \in \mathcal{Z}_{P_L}} \mathbf{z}_i \right)^T \left(\sum_{\mathbf{z}_i \in \mathcal{Z}_{P_L}} \mathbf{z}_i \right) \right] \\ &= \frac{1}{n_{P_L}^2} \left[p(i \neq j) \sum_{\mathbf{z}_i, \mathbf{z}_j \in \mathcal{Z}_P, i \neq j} \mathbf{z}_i^T \mathbf{z}_j + p(i = j) \sum_{\mathbf{z}_i \in \mathcal{Z}_P} \mathbf{z}_i^T \mathbf{z}_i \right] \\ &= \frac{1}{n_{P_L}^2} \left[\frac{\binom{n_P - 2}{2}}{\binom{n_P}{2}} \sum_{\mathbf{z}_i, \mathbf{z}_j \in \mathcal{Z}_P, i \neq j} \mathbf{z}_i^T \mathbf{z}_j + \frac{\binom{n_P - 1}{1}}{\binom{n_P}{2}} \sum_{\mathbf{z}_i \in \mathcal{Z}_P} \mathbf{z}_i^T \mathbf{z}_i \right] \\ &= \frac{1}{n_{P_L}^2} \left[\frac{n_{P_L}(n_{P_L} - 1)}{n_P(n_P - 1)} \sum_{\mathbf{z}_i, \mathbf{z}_j \in \mathcal{Z}_P, i \neq j} \mathbf{z}_i^T \mathbf{z}_j + \frac{n_{P_L}}{n_P} \sum_{\mathbf{z}_i \in \mathcal{Z}_P} \mathbf{z}_i^T \mathbf{z}_i \right] \end{aligned}$$

Plugging this back we get:

$$\begin{aligned}
\mathbb{E} \left[\|\mu - \mu^*\|^2 \right] &= \frac{1}{n_{PL}^2} \left[\frac{n_{PL}(n_{PL} - 1)}{n_P(n_P - 1)} \sum_{\mathbf{z}_i, \mathbf{z}_j \in \mathcal{Z}_P, i \neq j} \mathbf{z}_i^T \mathbf{z}_j + \frac{n_{PL}}{n_P} \sum_{\mathbf{z}_i \in \mathcal{Z}_P} \mathbf{z}_i^T \mathbf{z}_i \right] - \mu^{*T} \mu^* \\
&= \frac{n_P - n_{PL}}{n_{PL}(n_P - 1)} \left[\frac{1}{n_P} \sum_{\mathbf{z}_i \in \mathcal{Z}_P} \mathbf{z}_i^T \mathbf{z}_i - \mu^{*T} \mu^* \right] \\
&= \left(1 + \frac{n_P - n_{PL}}{n_{PL}(n_P - 1)} \right) \phi^*(\mathcal{Z}_P, \mu^*)
\end{aligned}$$

This concludes the proof. \blacksquare

A.6.4 PROOF OF LEMMA 3

Proof. This result is a direct consequence of Lemma 3.3 from (Arthur & Vassilvitskii, 2007) and specializing to our case where we only have 1 uncovered cluster i.e. $t = u = 1$ and consequently the harmonic sum $H_t = 1$. \blacksquare

A.7 ADDITIONAL REPRODUCIBILITY DETAILS

In this section we present more details on our experimental setup.

For all the experiments in Table 2, contrastive training is done using LARS optimizer (You et al., 2019), cosine annealing schedule with linear warm-up, batch size 1024, initial learning rate 1.2. We use a 128 dimensional projection layer $h_{\Gamma}(\cdot)$ composed of two linear layers with relu activation and batch normalization. We leverage Faiss (Johnson et al., 2019) for efficient implementation of PUPL. To ensure reproducibility, all experiments are run with deterministic cuDNN back-end and repeated 5 times with different random seeds and the confidence intervals are noted.

A.7.1 POSITIVE UNLABELED BENCHMARK DATASETS

Consistent with recent literature on PU Learning (Li et al., 2022; Chen et al., 2020a) we conduct our experiments on six benchmark datasets: STL-I, STL-II, CIFAR-I, CIFAR-II, FMNIST-I, and FMNIST-II, obtained via modifying STL-10 (Coates et al., 2011), CIFAR-10 (Krizhevsky et al., 2009), and Fashion MNIST (Xiao et al., 2017), respectively. The specific definitions of labels (“positive” vs “negative”) are as follows:

- **FMNIST-I:** The labels “positive” correspond to the classes “1, 4, 7”, while the labels “negative” correspond to the classes “0, 2, 3, 5, 6, 8, 9”.
- **FMNIST-II:** The labels “positive” correspond to the classes “0, 2, 3, 5, 6, 8, 9”, while the labels “negative” correspond to the classes “1, 4, 7”.
- **CIFAR-I:** The labels “positive” correspond to the classes “0, 1, 8, 9”, while the labels “negative” correspond to the classes “2, 3, 4, 5, 6, 7”.
- **CIFAR-II:** The labels “positive” correspond to the classes “2, 3, 4, 5, 6, 7”, while the labels “negative” correspond to the classes “0, 1, 8, 9”.
- **STL-I:** The labels “positive” correspond to the classes “0, 2, 3, 8, 9”, while the labels “negative” correspond to the classes “1, 4, 5, 6, 7”.
- **STL-II:** The labels “positive” correspond to the classes “1, 4, 5, 6, 7”, while the labels “negative” correspond to the classes “0, 2, 3, 8, 9”.

A.7.2 POSITIVE UNLABELED BASELINES

Next, we describe the PU baselines used in Table 2:

- **Unbiased PU learning (uPU)** (Du Plessis et al., 2014): This method is based on unbiased risk estimation and incorporates cost-sensitivity.
- **Non-negative PU learning (nnPU)** (Kiryo et al., 2017): This approach utilizes non-negative risk estimation and incorporates cost-sensitivity. Suggested settings: $\beta = 0$ and $\gamma = 1.0$.
- **nnPU w Mixup** Zhang et al. (2017) : This cost-sensitive method combines the nnPU approach with the mixup technique. It performs separate mixing of positive instances and unlabeled ones.
- **SELF-PU** Chen et al. (2020d): This cost-sensitive method incorporates a self-supervision scheme. Suggested settings: $\alpha = 10.0$, $\beta = 0.3$, $\gamma = \frac{1}{16}$, Pace1 = 0.2, and Pace2 = 0.3.
- **Predictive Adversarial Networks (PAN)** (Hu et al., 2021): This method is based on GANs and specifically designed for PU learning. Suggested settings: $\lambda = 1e - 4$.
- **Variational PU learning (vPU)** (Chen et al., 2020a): This approach is based on the variational principle and is tailored for PU learning. The public code from net.9 was used for implementation. Suggested settings: $\alpha = 0.3$, $\beta \in \{1e - 4, 3e - 4, 1e - 3, \dots, 1, 3\}$.
- **MIXPUL** (Wei et al., 2020): This method combines consistency regularization with the mixup technique for PU learning. The implementation utilizes the public code from net.10. Suggested settings: $\alpha = 1.0$, $\beta = 1.0$, $\eta = 1.0$.
- **Positive-Unlabeled Learning with effective Negative sample Selector PULNS** (Luo et al., 2021): This approach incorporates reinforcement learning for sample selection. We implemented a custom Python code with a 3-layer MLP selector, as suggested by the paper. Suggested settings: $\alpha = 1.0$ and $\beta \in \{0.4, 0.6, 0.8, 1.0\}$.
- **P³MIX-C/E** (Li et al., 2022): Denotes the heuristic mixup based approach.

A.7.3 IMAGE AUGMENTATIONS FOR CONTRASTIVE TRAINING

We provide the details of transformations used to obtain the contrastive learning benchmarks in this paper for each datasets.

```

1 cifar_transform = transforms.Compose([
2     transforms.RandomResizedCrop(input_shape),
3     transforms.RandomHorizontalFlip(p=0.5),
4     transforms.RandomApply([GaussianBlur([0.1, 2.0])], p=0.5),
5     transforms.RandomApply(
6         [transforms.ColorJitter(0.4, 0.4, 0.4, 0.1)], p=0.8),
7     transforms.RandomGrayscale(p=0.2),
8     transforms.ToTensor(),
9     transforms.Normalize(mean=self.mean, std=self.std)])

```

```

1 fmnist_transform = transforms.Compose([
2     transforms.RandomResizedCrop(input_shape),
3     transforms.RandomApply(
4         [transforms.ColorJitter(0.4, 0.4, 0.2, 0.1)], p=0.8),
5     transforms.ToTensor(),
6     transforms.Normalize(mean=self.mean, std=self.std)])

```

```

1 stl_transform = transforms.Compose([
2     transforms.RandomHorizontalFlip(),
3     transforms.RandomResizedCrop(size=96),
4     transforms.RandomApply(
5         [transforms.ColorJitter(0.5, 0.5, 0.5, 0.1)], p=0.8),
6     transforms.RandomGrayscale(p=0.2),
7     transforms.GaussianBlur(kernel_size=9),
8     transforms.ToTensor(),
9     transforms.Normalize((0.5,), (0.5,)))

```

```

1 imagenet_transform = transforms.Compose([
2     transforms.RandomResizedCrop(224, interpolation=Image.BICUBIC),
3     transforms.RandomHorizontalFlip(p=0.5),
4     transforms.RandomApply(
5         [transforms.ColorJitter(0.4, 0.4, 0.2, 0.1)], p=0.8),
6     transforms.RandomGrayscale(p=0.2),
7     transforms.RandomApply([GaussianBlur([0.1, 2.0])], p=0.5),
8     Solarization(p=0.2),
9     transforms.ToTensor(),
10    transforms.Normalize(mean=self.mean, std=self.std)])

```

```

1 imagenet_transform = transforms.Compose([
2     transforms.RandomResizedCrop(224, interpolation=Image.BICUBIC),
3     transforms.RandomHorizontalFlip(p=0.5),
4     transforms.RandomApply(
5         [transforms.ColorJitter(0.4, 0.4, 0.2, 0.1)], p=0.8),
6     transforms.RandomGrayscale(p=0.2),
7     transforms.RandomApply([GaussianBlur([0.1, 2.0])], p=0.5),
8     Solarization(p=0.2),
9     transforms.ToTensor(),
10    transforms.Normalize(mean=self.mean, std=self.std)])

```

A.7.4 PYTORCH STYLE PSEUDO CODES

```

1 class SelfSupConLoss(nn.Module):
2     """
3     Self Supervised Contrastive Loss
4     """
5     def __init__(self, temperature: float = 0.5, reduction="mean"):
6         super(SelfSupConLoss, self).__init__()

```

```

7     self.temperature = temperature
8     self.cross_entropy = nn.CrossEntropyLoss(reduction=reduction)
9
10    def forward(self, z: torch.Tensor, z_aug: torch.Tensor, *kwargs) ->
11        torch.Tensor:
12            """
13                :param z: features
14                :param z_aug: augmentations
15                :return: loss value, scalar
16            """
17
18            batch_size, _ = z.shape
19            # project onto hypersphere
20            z = nn.functional.normalize(z, dim=1)
21            z_aug = nn.functional.normalize(z_aug, dim=1)
22
23            # calculate similarities block-wise
24            inner_pdt_00 = torch.einsum('nc,mc->nm', z, z) / self.temperature
25            inner_pdt_01 = torch.einsum('nc,mc->nm', z, z_aug) / self.temperature
26            inner_pdt_10 = torch.einsum("nc,mc->nm", z_aug, z) / self.temperature
27            inner_pdt_11 = torch.einsum('nc,mc->nm', z_aug, z_aug) / self.
28            temperature
29
30            # remove similarities between same views of the same image
31            diag_mask = torch.eye(batch_size, device=z.device, dtype=torch.bool)
32            inner_pdt_00 = inner_pdt_00[~diag_mask].view(batch_size, -1)
33            inner_pdt_11 = inner_pdt_11[~diag_mask].view(batch_size, -1)
34
35            # concatenate blocks
36            inner_pdt_0100 = torch.cat([inner_pdt_01, inner_pdt_00], dim=1)
37            inner_pdt_1011 = torch.cat([inner_pdt_10, inner_pdt_11], dim=1)
38            logits = torch.cat([inner_pdt_0100, inner_pdt_1011], dim=0)
39
40            labels = torch.arange(batch_size, device=z.device, dtype=torch.long)
41            labels = labels.repeat(2)
42            loss = self.cross_entropy(logits, labels)
43
44            return loss

```

```

1 class SupConLoss(nn.Module):
2     """
3         Supervised Contrastive Loss
4     """
5     def __init__(self, temperature: float = 0.5, reduction="mean"):
6         super(SupConLoss, self).__init__()
7         self.temperature = temperature
8         self.reduction = reduction
9
10    def forward(self, z: torch.Tensor, z_aug: torch.Tensor, labels: torch.
11        Tensor, *kwargs) -> torch.Tensor:
12            """
13                :param z: features => bs * shape
14                :param z_aug: augmentations => bs * shape
15                :param labels: ground truth labels of size => bs
16                :return: loss value => scalar
17            """
18
19            batch_size, _ = z.shape
20
21            # project onto hypersphere
22            z = nn.functional.normalize(z, dim=1)
23            z_aug = nn.functional.normalize(z_aug, dim=1)
24
25            # calculate similarities block-wise

```

```

25     inner_pdt_00 = torch.einsum('nc,mc->nm', z, z) / self.temperature
26     inner_pdt_01 = torch.einsum('nc,mc->nm', z, z_aug) / self.temperature
27     inner_pdt_10 = torch.einsum("nc,mc->nm", z_aug, z) / self.temperature
28     inner_pdt_11 = torch.einsum('nc,mc->nm', z_aug, z_aug) / self.
29     temperature
30
31     # concatenate blocks
32     inner_pdt_001 = torch.cat([inner_pdt_00, inner_pdt_01], dim=1)
33     inner_pdt_1011 = torch.cat([inner_pdt_10, inner_pdt_11], dim=1)
34     inner_pdt_mtx = torch.cat([inner_pdt_001, inner_pdt_1011], dim=0)
35
36     max_inner_pdt, _ = torch.max(inner_pdt_mtx, dim=1, keepdim=True)
37     inner_pdt_mtx = inner_pdt_mtx - max_inner_pdt.detach()  # for
38     numerical stability
39
40     # compute negative log-likelihoods
41     nll_mtx = torch.exp(inner_pdt_mtx)
42     # mask out self contrast
43     diag_mask = torch.ones_like(inner_pdt_mtx, device=z.device, dtype=
44     torch.bool).fill_diagonal_(0)
45     nll_mtx = nll_mtx * diag_mask
46     nll_mtx /= torch.sum(nll_mtx, dim=1, keepdim=True)
47     nll_mtx[nll_mtx != 0] = - torch.log(nll_mtx[nll_mtx != 0])
48
49     # mask out contributions from samples not from same class as i
50     mask_label = torch.unsqueeze(labels, dim=-1)
51     eq_mask = torch.eq(mask_label, torch.t(mask_label))
52     eq_mask = torch.tile(eq_mask, (2, 2))
53     similarity_scores = nll_mtx * eq_mask
54
55     # compute the loss -by averaging over multiple positives
56     loss = similarity_scores.sum(dim=1) / (eq_mask.sum(dim=1) - 1)
57     if self.reduction == 'mean':
58         loss = torch.mean(loss)
59     return loss

```

```

1 class PUConLoss(nn.Module):
2 """
3     Positive Unlabeled Contrastive Loss
4 """
5
6     def __init__(self, temperature: float = 0.5):
7         super(PUConLoss, self).__init__()
8         # per sample unsup and sup loss : since reduction is None
9         self.sscl = SelfSupConLoss(temperature=temperature, reduction='none')
10        self.scl = SupConLoss(temperature=temperature, reduction='none')
11
12    def forward(self, z: torch.Tensor, z_aug: torch.Tensor, labels: torch.
13    Tensor, *kwargs) -> torch.Tensor:
14        """
15            @param z: Anchor
16            @param z_aug: Mirror
17            @param labels: annotations
18        """
19        # get per sample sup and unsup loss
20        sup_loss = self.scl(z=z, z_aug=z_aug, labels=labels)
21        unsup_loss = self.sscl(z=z, z_aug=z_aug)
22
23        # label for M-viewed batch with M=2
24        labels = labels.repeat(2).to(z.device)
25
26        # get the indices of P and U samples in the multi-viewed batch
27        p_ix = torch.where(labels == 1)[0]
28        u_ix = torch.where(labels == 0)[0]

```

```

28
29     # if no positive labeled it is simply SelfSupConLoss
30     num_labeled = len(p_ix)
31     if num_labeled == 0:
32         return torch.mean(unsup_loss)
33
34     # compute expected similarity
35     # -----
36     risk_p = sup_loss[p_ix]
37     risk_u = unsup_loss[u_ix]
38
39     loss = torch.cat([risk_p, risk_u], dim=0)
40     return torch.mean(loss)

```

```

1 def puPL(x_PU, y_PU, num_clusters=2):
2 """
3 puPL: Positive Unlabeled Pseudo Labeling
4 """
5     p_ix = y_PU==1
6     u_ix = y_PU==0
7     x_P = x_PU[p_ix]
8     x_U = x_PU[u_ix]
9
10    ## Initialize Cluster Centers ##
11    # Compute the mean of x_P as the first centroid
12    centroid_P = np.mean(x_P, axis=0)
13    # Next, use K-means++ to choose the second center from x_U
14    kmeans_pp = KMeans(n_clusters=1, init=np.array([centroid_2]))
15    kmeans_pp.fit(x_U)
16    centroid_N = kmeans_pp.cluster_centers_[0]
17    # Initialize the centroids with the computed values
18    centroids = np.array([centroid_N, centroid_P])
19
20    ## Perform K-means iterations ##
21    kmeans = KMeans(n_clusters=num_clusters, init=centroids)
22    kmeans.fit(np.concatenate((x_U, x_P), axis=0))
23
24    labels = kmeans.labels_
25    data = np.concatenate((x_U, x_P), axis=0)
26
27    return labels, data

```