SimPro: A Simple Probabilistic Framework Towards Realistic Long-Tailed Semi-Supervised Learning

Chaoqun Du [*1] **Yizeng Han** [*1] **Gao Huang** [1]

Abstract

Recent advancements in semi-supervised learning have focused on a more realistic yet challenging task: addressing imbalances in labeled data while the class distribution of unlabeled data remains both *unknown* and potentially *mismatched*. Current approaches in this sphere often presuppose rigid assumptions regarding the class distribution of unlabeled data, thereby limiting the adaptability of models to only certain distribution ranges. In this study, we propose a novel approach, introducing a highly adaptable framework, designated as **SimPro**, which does not rely on any predefined assumptions about the distribution of unlabeled data. Our framework, grounded in a probabilistic model, innovatively refines the expectation-maximization (EM) algorithm by *explicitly decoupling* the modeling of conditional and marginal class distributions. This separation facilitates a closed-form solution for class distribution estimation during the maximization phase, leading to the formulation of a Bayes classifier. The Bayes classifier, in turn, enhances the quality of pseudo-labels in the expectation phase. Remarkably, the SimPro framework not only comes with theoretical guarantees but also is straightforward to implement. Moreover, we introduce two novel class distributions broadening the scope of the evaluation. Our method showcases consistent state-of-the-art performance across diverse benchmarks and data distribution scenarios. Our code is available at https://github.com/LeapLabTHU/SimPro.

[*]Equal contribution [1]Department of Automation, BNRist, Tsinghua University, Beijing, China. Correspondence to: Gao Huang <gaohuang@tsinghua.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

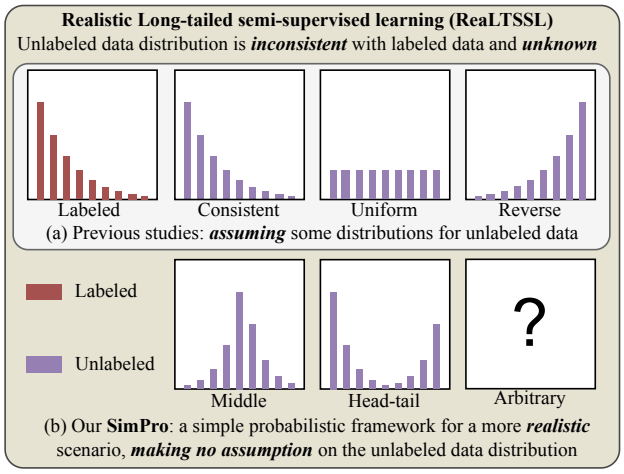

Figure 1. The general idea of SimPro addressing the ReaLTSSL problem. (a) Current methods typically rely on predefined or assumed class distribution patterns for unlabeled data, limiting their applicability. (b) In contrast, our SimPro embraces a more realistic scenario by introducing a simple and elegant framework that operates effectively without making any assumptions about the distribution of unlabeled data. This paradigm shift allows for greater flexibility and applicability in diverse ReaLTSSL scenarios.

1. Introduction

Semi-supervised learning (SSL) offers a viable solution to the scarcity of labeled data by leveraging unlabeled data (Tarvainen & Valpola, 2017; Berthelot et al., 2019b; Miyato et al., 2018; Sohn et al., 2020). Common SSL algorithms typically generate pseudo-labels for unlabeled data to facilitate model training (Lee et al., 2013). However, real-world data often adheres to a long-tailed distribution, leading to a predominant focus on majority classes and resulting in imbalanced pseudo-labels (Liu et al., 2019; Kang et al., 2020; Du et al., 2024). This phenomenon, known as long-tailed semi-supervised learning (LTSSL), presents significant challenges in the field. Traditional LTSSL methods (Lai et al., 2022; Lee et al., 2021; Wei et al., 2022; 2021; Kim et al., 2020) assume consistency in class distributions between labeled and unlabeled data, an often unrealistic premise. In practice, class distributions can be *inconsistent* and *unknown*, especially as new data are continuously collected or from different tasks. This ongoing integration process can lead to significant shifts in class distributions.

In response to these challenges, the concept of *realistic long-tailed semi-supervised learning* (ReaLTSSL), which aims at addressing the *unknown and mismatched* class distributions, has garnered significant attention (Kim et al., 2020; Wei et al., 2021; Oh et al., 2022; Wei & Gan, 2023). Notably, recent works ACR (Wei & Gan, 2023) and CPE (Ma et al., 2024) pre-define anchor distributions for unlabeled data (Fig. 1 (a)). The ACR estimates the distributional distance to adapt consistency regularization, while CPE involves training multiple classifiers, each tailored to a specific class distribution. However, this approach presupposes certain knowledge about the unlabeled data distribution, preventing its applications in real-world scenarios where anchor distributions may not represent all possible distributions. Furthermore, the prevailing techniques often employ multi-branch frameworks and introduce additional loss functions, adding complexity and limiting their generality.

To address these limitations, we propose a *Simple Probabilistic* (**SimPro**) framework for ReaLTSSL. We revisit pseudo-label-based SSL techniques through the lens of the Expectation-Maximization (EM) algorithm. The EM algorithm, a well-known iterative method in statistical modeling, is particularly relevant in SSL for handling unobserved latent variables, such as pseudo-labels of unlabeled data. The E-step entails generating pseudo-labels with the model, while the M-step involves model training using both labeled and unlabeled data. In the context of *unknown* and *mismatched* class distributions, the E-step may produce biased pseudo-labels, diminishing the algorithm's effectiveness. Our SimPro *avoids fixed assumptions about the unlabeled data distribution*, instead of innovatively extending the EM algorithm for ReaLTSSL. Specifically, we *explicitly decouple* the modeling of conditional and marginal distributions. Such separation enables a closed-form solution for the marginal distribution in the M step. Subsequently, this solution is employed to train a Bayes classifier. This Bayes classifier, in turn, improves the quality of pseudo-labels generated in the E-step. Not only does SimPro offer high effectiveness, but it is also easy to implement, requiring *minimal* code modifications.

Moreover, we expand upon existing evaluation methods (Oh et al., 2022), which primarily focus on three known class distributions (consistent, uniform, and reversed), by introducing two novel realistic scenarios: middle and head-tail distributions (Fig. 1 (b)). The middle distribution represents a concentration of classes in the middle range of labeled data's classes, whereas the head-tail distribution indicates a concentration at both extremes. Notably, our method is theoretically general enough to handle any other distribution patterns, since no prior assumptions are required.

We summarize our contributions as follows:

1. We present SimPro, a simple probabilistic framework tailored for realistic long-tailed semi-supervised learning. This framework does not presuppose any knowledge about the class distribution of unlabeled data. It hinges on the explicit estimation and utilization of class distributions within the EM algorithm. SimPro effectively mitigates the challenges posed by unknown and mismatched class distributions, stepping towards a more realistic LTSSL scenario.

2. We introduce two novel class distribution patterns for unlabeled data, complementing the existing three standard ones. This expansion facilitates a more comprehensive and realistic evaluation of ReaLTSSL algorithms, bridging the gap between theoretical models and practical applications.

3. Comprehensive experiments on five commonly used benchmarks (CIFAR10/100-LT, STL10-LT, and ImageNet-127/1k) and five distinct class distributions validate that our SimPro consistently achieves SOTA performance.

2. Related Work

Semi-supervised learning (SSL) has gained prominence through a subset of algorithms that use unlabeled data to enhance model performance. This enhancement primarily occurs through the generation of pseudo-labels, effectively forming a self-training loop (Miyato et al., 2018; Berthelot et al., 2019a;b; Huang & Du, 2022; Wang et al., 2023). Modern SSL methodologies, such as those presented in (Berthelot et al., 2019a; Sohn et al., 2020), integrate pseudo-labeling with consistency regularization. This integration fosters uniform predictions across varying representations of a single image, thereby bolstering the robustness of deep networks. A notable example, FixMatch (Sohn et al., 2020), has demonstrated exceptional results in image recognition tasks, outperforming competing SSL approaches.

The efficacy of SSL algorithms heavily relies on the quality of the pseudo-labels they generate. However, both labeled and unlabeled data follow a long-tailed class distribution in the LTSSL scenario. Conventional SSL methods are prone to produce biased pseudo-labels, which significantly downgrade their effectiveness.

Long-tailed semi-supervised learning has garnered considerable interest due to its relevance in numerous real-world applications. In this domain, DARP (Kim et al., 2020) and CReST (Wei et al., 2021) aim to mitigate the issue of biased pseudo-labels by aligning them with the class distribution of labeled data. Another notable approach (Lee et al., 2021) employs an auxiliary balanced classifier, which is trained through the down-sampling of majority classes, to enhance generalization capabilities. These algorithms have markedly improved performance but operate under the assumption of identical class distributions for labeled and unlabeled data.

In addressing Realistic LTSSL challenges, DASO (Oh et al.,

2022) innovates by adapting the proportion of linear and semantic pseudo-labels to the unknown class distribution of unlabeled data. Its success largely depends on the discriminative quality of the representations, a factor that becomes less reliable in long-tailed distributions. ACR (Wei & Gan, 2023), on the other hand, attempts to refine consistency regularization by pre-defining distribution anchors and achieves promising results. CPE (Ma et al., 2024) trains multiple anchor experts where each is tasked to model one distribution. However, such anchor distribution-based approaches might not encompass all potential class distribution scenarios, and their complexity could hinder the broader application.

3. Method

In this section, we first introduce the problem formulation of ReaLTSSL (Sec. 3.1), setting the stage for our method. Subsequently, we delve into the proposed simple and probabilistic framework, SimPro (Sec. 3.2). We provide implementation details in Sec. 3.3 to elucidate SimPro further.

3.1. Preliminaries

Problem formulation. We begin by outlining the formulation for the realistic long-tailed semi-supervised learning (ReaLTSSL) problem, laying the groundwork for our approach. The setup involves a labeled dataset $\mathcal{D}_l = \{(x_i, y_i)\}_{i=1}^N$ and an unlabeled dataset $\mathcal{D}_u = \{x_i\}_{i=1}^M$, where $x_i \in \mathbb{R}^d$ represents the i-th data sample and $y_i \in \{0,1\}^K$ is the corresponding one-hot label, with K denoting the number of classes. The objective of ReaLTSSL is to train a classifier $F_{\boldsymbol{\theta}} : \mathbb{R}^d \mapsto \{0,1\}^K$, parameterized by $\boldsymbol{\theta}$.

Assumption 1. We assume a realistic scenario where labeled, unlabeled, and test data share the same conditional distribution $P(x|y)$, yet may exhibit distinct marginal distributions $P(y)$. Crucially, the marginal distribution $P(y)$ of the unlabeled data remains *unknown*.

Further, we consider five diverse distributions for the unlabeled data (Fig. 1), reflecting various real-world situations.

The EM algorithm in semi-supervised learning. In SSL, pseudo-labeling is a key technique for leveraging unlabeled data. This involves creating pseudo-labels for the unlabeled data using the model and then training the model with both the pseudo-labeled and ground-truth data. This aligns with the Expectation-Maximization (EM) algorithm (Dempster et al., 1977), where the E-step generates pseudo-labels, and the M-step updates the parameters using the pseudo-labels, maximizing the likelihood function.

Our method builds on a popular algorithm FixMatch (Sohn et al., 2020), which integrates consistency regularization in the standard SSL setting. Pseudo-labels are created via *weakly-augmented* unlabeled data and applied to train *strongly-augmented* samples based on a confidence threshold. The loss for unlabeled data is

$$\mathcal{L}_u(x_i) = \mathbb{I}(\max(q_\omega) \geq t) \cdot \mathcal{H}(\arg\max(q_\omega), q_\Omega), \quad (1)$$

where q_ω and q_Ω represent the prediction logits for weakly and strongly augmented samples, respectively, $\mathcal{H}$ denotes the cross-entropy loss, and t is the confidence threshold.

Long-tailed learning. In typical SSL scenarios, the assumption of identical distributions for labeled, unlabeled, and test data often prevails. However, long-tailed learning tasks usually involve imbalanced training sets and balanced test sets, leading to discrepancies in the prior distribution of $P(y)$ between training and testing data. Some studies (Ren et al., 2020; Menon et al., 2021; Hong et al., 2021) tackle this via Bayesian inference, introducing a prior distribution over class labels:

$$\begin{aligned}\mathcal{L}_l(x) &= -\log P(y|x;\boldsymbol{\theta},\boldsymbol{\pi}) \\ &= -\log \frac{P(y;\boldsymbol{\pi})P(x|y;\boldsymbol{\theta})}{P(x)} \\ &= -\log \frac{\phi_y \exp(f_{\boldsymbol{\theta}}(x,y))}{\sum_{y'} \phi_{y'} \exp(f_{\boldsymbol{\theta}}(x,y'))},\end{aligned} \quad (2)$$

where ϕ_y denotes the class frequency in the training or test set, $\boldsymbol{\pi}$ is the class distribution parameter and $\boldsymbol{\theta}$ is the parameter of $P(x|y)/P(x)$. Here we omit the parameter of $P(x)$ for simplicity. The detailed mathematical derivation is provided in App. A.

While supervised learning allows for a known distribution parameter $\boldsymbol{\pi}$, enabling a direct application to model $P(y)$ and explicit decoupling from $\boldsymbol{\theta}$, ReaLTSSL poses a greater challenge as the prior $\boldsymbol{\pi}$ for unlabeled data is *unknown*. This necessitates innovative approaches to adapt to the imbalanced data while maintaining model efficacy.

3.2. SimPro Framework

Framework overview. In the realistic semi-supervised learning (ReaLTSSL) context, the conventional assumption of independent and identically distributed (i.i.d.) labeled and unlabeled data is no longer valid. Moreover, the marginal (class) distribution $P(y)$ of the unlabeled data may be *inconsistent* with that of the labeled data and remains *unknown*, which challenges the traditional SSL frameworks.

To overcome this, we introduce SimPro, an elegant and effective probabilistic framework adapted for the unique ReaLTSSL setting. Illustrated in Fig. 2, SimPro distinctively decouples $\boldsymbol{\pi}$ and $\boldsymbol{\theta}$, unlike traditional SSL methods (Sohn et al., 2020). In the E-step, we generate pseudo-labels using the parameters $\boldsymbol{\pi}$ and $\boldsymbol{\theta}$ obtained from the previous M-step. The M-step then models the conditional distribution $P(x|y)$ using network parameters $\boldsymbol{\theta}$, which are optimized through

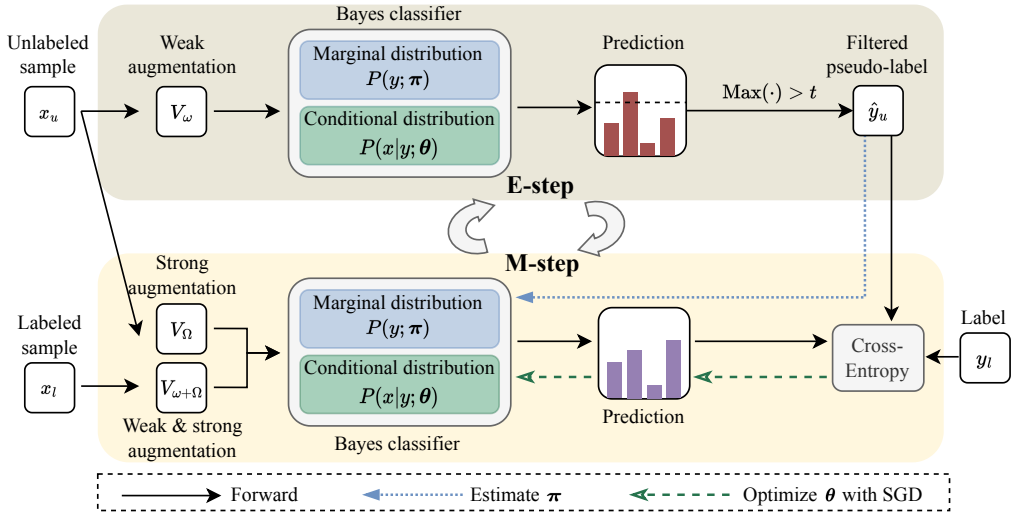

Figure 2. The SimPro Framework Overview. This framework distinctively separates the conditional and marginal (class) distributions. In the E-step (top), pseudo-labels are generated using the current parameters $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$. In the subsequent M-step (bottom), these pseudo-labels, along with the ground-truth labels, are utilized to compute the Cross-Entropy loss (refer to Eq. (13)), facilitating the optimization of network parameters $\boldsymbol{\theta}$ via gradient descent. Concurrently, the marginal distribution parameter $\boldsymbol{\pi}$ is recalculated using a closed-form solution based on the generated pseudo-labels (as detailed in Eq. (7)).

gradient descent. Simultaneously, we derive a closed-form solution for the class distribution $P(y)$, represented by $\boldsymbol{\pi}$.

It is worth noting that the treatment of the $\boldsymbol{\pi}$ in our framework is not heuristic. It is firmly rooted in probabilistic modeling and the principles of the EM algorithm, providing theoretical soundness as substantiated in Props. 1 and 2.

Probabilistic model. In addressing the ReaLTSSL challenge, we adopt an Expectation-Maximization (EM) approach, underpinned by a robust probabilistic model. The model is governed by the fundamental principles of conditional probability, as shown in:

$$P(\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{\theta}, \boldsymbol{\pi}) = P(\boldsymbol{y}|\boldsymbol{x}; \boldsymbol{\theta}, \boldsymbol{\pi}) P(\boldsymbol{x}). \tag{3}$$

Here, we do not explicitly parameterize $P(x)$, as per the independence of parameters through conditional parameterization (Koller & Friedman, 2009). Thus, when x is not a condition, the parameters of the relevant notions omit the parameters of $P(x)$, such as $P(x)$, $P(x|y)$, $P(x, y)$, etc. According to Eq. (2), this may lead to a potential misunderstanding, as the equation $P(x) = \sum_y P(x|y; \boldsymbol{\theta}) P(y; \boldsymbol{\pi})$ seems to suggest that $P(x)$ is parameterized by $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$, which is not the case. The detailed mathematical derivation is provided in App. A.

We focus on estimating the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$, pivotal for learning a discriminative model. Consequently, we concentrate on those terms dependent on $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$, sidelining those independent of these parameters.

The complete data log-likelihood is thus articulated as:

$$P(\boldsymbol{y}|\boldsymbol{x}; \boldsymbol{\theta}, \boldsymbol{\pi}) = \prod_{i=1}^{N} P(y_i|x_i; \boldsymbol{\theta}, \boldsymbol{\pi}_l) \prod_{j=1}^{M} P(y_j|x_j; \boldsymbol{\theta}, \boldsymbol{\pi}_u), \tag{4}$$

where $\boldsymbol{\pi} = \{\boldsymbol{\pi}_l, \boldsymbol{\pi}_u\}$ signifies the class distributions for labeled and unlabeled data, respectively, with N and M representing the number of labeled/unlabeled samples.

E-step (generating pseudo-labels). By Eq. (4), the expected complete data log-likelihood $\mathcal{Q}$ function is derived from the preceding iteration's parameters, $\boldsymbol{\theta}'$ and $\boldsymbol{\pi}'$:

$$\begin{aligned} \mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\pi}; \boldsymbol{\theta}', \boldsymbol{\pi}') &= \mathbb{E}_{\boldsymbol{y}|\boldsymbol{x}; \boldsymbol{\theta}', \boldsymbol{\pi}'} \left[\log P(\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{\theta}, \boldsymbol{\pi})\right] \\ &= \sum_i \log P(y_i|x_i; \boldsymbol{\theta}, \boldsymbol{\pi}_l) \\ &\quad + \sum_{j,y} P(y|x_j; \boldsymbol{\theta}', \boldsymbol{\pi}') \log P(y|x_j; \boldsymbol{\theta}, \boldsymbol{\pi}_u). \end{aligned} \tag{5}$$

The E-step involves generating soft pseudo-labels $P(y|x_j; \boldsymbol{\theta}', \boldsymbol{\pi}')$ under the current $\boldsymbol{\theta}'$ and $\boldsymbol{\pi}'$. These soft pseudo-labels are specifically defined by Eq. (10), which is detailed in Prop. 2. In the subsequent M-step, these pseudo-labels are used alongside the one-hot labels of the labeled data to compute the cross-entropy loss.

M-step (optimizing $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$). The M-step focuses on optimizing the expected complete data log-likelihood $\mathcal{Q}$-function concerning the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$.

(a) *Optimization of* $\boldsymbol{\pi}$: The closed-form solution for $\boldsymbol{\pi}$ can be derived directly from the $\mathcal{Q}$-function (Eq. (5)). Specifically, the terms involving $\boldsymbol{\pi}$ in $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\pi}; \boldsymbol{\theta}', \boldsymbol{\pi}')$ are given by

$$\sum_i \log P(y_i; \boldsymbol{\pi}_l) + \sum_{j,y} P(y|x_j; \boldsymbol{\theta}', \boldsymbol{\pi}') \log P(y; \boldsymbol{\pi}_u). \quad (6)$$

Proposition 1 (Closed-form Solution for $\boldsymbol{\pi}$)**.** *The optimal* $\hat{\boldsymbol{\pi}}$ *that maximizes* $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\pi}; \boldsymbol{\theta}', \boldsymbol{\pi}')$ *is*

$$\hat{\boldsymbol{\pi}}_l = \frac{1}{N}\sum_{i=1}^{N} y_i, \quad \hat{\boldsymbol{\pi}}_u = \frac{1}{M}\sum_{j=1}^{M} P(y|x_j; \boldsymbol{\theta}', \boldsymbol{\pi}'). \quad (7)$$

(b) *Optimization of* $\boldsymbol{\theta}$: The network parameters $\boldsymbol{\theta}$, unlike $\boldsymbol{\pi}$ which have a closed-form solution, are optimized using standard stochastic gradient descent (SGD). Combining with Eq. (2), the terms involving $\boldsymbol{\theta}$ in $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\pi}; \boldsymbol{\theta}', \boldsymbol{\pi}')$ are

$$(\sum_i + \sum_{j,y} P(y|x_j; \boldsymbol{\theta}', \boldsymbol{\pi}')) \log \frac{P(x|y; \boldsymbol{\theta})}{P(x)} \quad (8)$$

$$=(\sum_i + \sum_{j,y} P(y|x_j; \boldsymbol{\theta}', \boldsymbol{\pi}')) \log \frac{\exp(f_{\boldsymbol{\theta}}(x, y))}{\sum_{y'} \phi_{y'} \exp(f_{\boldsymbol{\theta}}(x, y'))},$$

which simplifies to the supervised scenario in Eq. (2) by treating $P(y|x_j; \boldsymbol{\theta}', \boldsymbol{\pi}')$ as soft labels. Maximizing Eq. (8) corresponds to minimizing the cross-entropy loss. Here, $\phi_{y'}$ is interpreted as the estimated overall frequency of class y'. The optimization of model parameters $\boldsymbol{\theta}$ using the overall frequency vector $\boldsymbol{\phi}$ is crucial for learning a Bayes classifier.

Proposition 2 (Bayes Classifier)**.** *In conjunction with the high-confidence filtering (Eq.* (1)*), the optimal* $\hat{\boldsymbol{\phi}}$ *for learning a Bayes classifier is mathematically derived as:*

$$\begin{aligned}\hat{\boldsymbol{\phi}} &= [\hat{\phi}_1, \hat{\phi}_2, \cdots, \hat{\phi}_K] \\ &= \frac{1}{N+M}(\sum_i y_i + \sum_j P(y|x_j; \boldsymbol{\theta}', \boldsymbol{\pi}')). \quad (9)\end{aligned}$$

Subsequently, with the model parameter $\boldsymbol{\theta}$ *which is optimized using the* $\hat{\boldsymbol{\phi}}$*, the corresponding Bayes classifier for unlabeled or test dataset with estimated or uniform class distribution is defined by the equation:*

$$P(y|x; \boldsymbol{\theta}; \hat{\boldsymbol{\pi}}) = \frac{P(y; \hat{\boldsymbol{\pi}}) \exp(f_{\boldsymbol{\theta}}(x, y))}{\sum_{y'} P(y'; \hat{\boldsymbol{\pi}}) \exp(f_{\boldsymbol{\theta}}(x, y'))}, \quad (10)$$

$$\textit{or} \quad P(y|x; \boldsymbol{\theta}) = \frac{\exp(f_{\boldsymbol{\theta}}(x, y))}{\sum_{y'} \exp(f_{\boldsymbol{\theta}}(x, y'))}. \quad (11)$$

Building upon Prop. 2, it is crucial to acknowledge that the parameter vector $\boldsymbol{\phi}$ is vital for learning Bayes classifiers. Consequently, to delve deeper into the theoretical foundations, we evaluate the impact of $\boldsymbol{\phi}$ on the model's performance. In line with the principles of online decision theory, we establish a regret bound for the decision error rate on the test set, denoted as $P(e; \boldsymbol{\phi})$. Our analysis is simplified by concentrating on a binary classification scenario, where the labels y belong to $\{-1, +1\}$.

Algorithm 1 Pseudocode of SimPro in a PyTorch-like style.

```
# N_e: (K,), where N_e[k] denotes the number of
    labeled data in class k in one epoch
# pi_u: (K,), the class distribution parameters of
    unlabeled samples
# phi: (K,), the overall class frequency
# f: deep network parameterized by theta
# alpha, tau, m: hyper-parameters
# CE: CrossEntropyLoss
# aug_w, aug_s: weak and strong augmentation
pi_u.init(uniform)
phi.init(consistent)
for epoch in range(epochs):
   pi_e = zeros(K) # temporary estimation of pi_u
   # load labeled and unlabeled samples
   for (x_l, y_l), x_u in zip(loader_l, loader_u):

      # E step: generating pseudo labels
      lgt_l = f.forward(aug(x_l))
      lgt_w = f.forward(aug_w(x_u)).detach()
      lgt_s = f.forward(aug_s(x_u))
      # Bayes classifer
      psd_lbs = softmax(lgt_w + tau*log(pi_u), dim=-1)
      # filter out pseudo labels with high confidence
      mask = max(psd_lbs, dim=-1)[0].ge(t)

      # M step: solving pi and phi, optimizing theta
      # solve pi_u with Eq. (7)
      pi_e += sum(psd_lbs[mask], dim=0)
      # optimize f (theta) with Eq. (11)
      loss_l = CE(lgt_l + tau * log(phi), y_l)
      loss_u = mean(CE(lgt_s + tau * log(phi),
            psd_lbs, reduction='none') * mask)
      loss = alpha * loss_l + loss_u
      loss.backward()
      update(theta)

   # update pi_u and phi
   phi_e = (pi_e + N_e) / sum(pi_e + N_e) # Eq. (9)
   # moving average
   phi = m * phi + (1 - m) * phi_e
   pi_u = m * pi_u + (1 - m) * pi_e / sum(pi_e)
```

Proposition 3 (Regret Bound)**.** *Let* $\boldsymbol{\phi}^*$ *denote the vector* $\boldsymbol{\phi}$ *obtained in Eq.* (9) *when pseudo-labels are replaced by ground-truth labels. For the decision error rate* $P(e; \boldsymbol{\phi})$ *on the test set, the regret bound is expressed as:*

$$P(e; \hat{\boldsymbol{\phi}}) - \inf_{\boldsymbol{\phi}} P(e; \boldsymbol{\phi}) \leq \frac{1}{2\phi^*_{+1}\phi^*_{-1}} |\hat{\phi} - \phi^*|, \quad (12)$$

where $|\hat{\phi} - \phi^*| = |\hat{\phi}_{+1} - \phi^*_{+1}| = |\hat{\phi}_{-1} - \phi^*_{-1}|$.

Prop. 3 illustrates that the regret bound is primarily governed by the first-order term of the estimation deviation. Additionally, it is inversely proportional to the ground truth $\boldsymbol{\phi}^*$, highlighting the learning challenges associated with imbalanced training data from a regret-bound perspective.

3.3. Implementation Details

Training objective for optimizing $\boldsymbol{\theta}$**.** In SimPro, the E-step primarily involves generating pseudo-labels using parameters $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$. Consequently, in the M-step, we first focus on optimizing the network parameter $\boldsymbol{\theta}$ guided by Eq. (8) via Stochastic Gradient Descent (SGD). Building on the FixMatch algorithm (Sohn et al., 2020), the overall training objective is formulated as:

$$\mathcal{L} = \alpha \mathcal{L}_l + \mathcal{L}_u, \quad (13)$$

Table 1. Top-1 accuracy (%) on CIFAR10-LT ($N_1 = 500$, $M_1 = 4000$) with different class imbalance ratios γ_l and γ_u under five different unlabeled class distributions. † indicates we reproduce ACR without anchor distributions for a fair comparison.

	consistent		uniform		reversed		middle		head-tail	
	$\gamma_l = 150$	$\gamma_l = 100$	$\gamma_l = 150$	$\gamma_l = 100$	$\gamma_l = 150$	$\gamma_l = 100$	$\gamma_l = 150$	$\gamma_l = 100$	$\gamma_l = 150$	$\gamma_l = 100$
	$\gamma_u = 150$	$\gamma_u = 100$	$\gamma_u = 1$	$\gamma_u = 1$	$\gamma_u = 1/150$	$\gamma_u = 1/100$	$\gamma_u = 150$	$\gamma_u = 100$	$\gamma_u = 150$	$\gamma_u = 100$
FixMatch (Sohn et al., 2020)	62.9 ±0.36	67.8 ±1.13	67.6 ±2.56	73.0 ±3.81	59.9 ±0.82	62.5 ±0.94	64.3 ±0.63	71.7 ±0.46	58.3 ±1.46	66.6 ±0.87
w/ CReST+ (Wei et al., 2021)	67.5 ±0.45	76.3 ±0.86	74.9 ±0.80	82.2 ±1.53	62.0 ±1.18	62.9 ±1.39	58.5 ±0.68	71.4 ±0.60	59.3 ±0.72	67.2 ±0.48
w/ DASO (Oh et al., 2022)	70.1 ±1.81	76.0 ±0.37	83.1 ±0.47	86.6 ±0.84	64.0 ±0.11	71.0 ±0.95	69.0 ±0.31	73.1 ±0.68	70.5 ±0.59	71.1 ±0.32
w/ ACR† (Wei & Gan, 2023)	70.9 ±0.37	76.1 ±0.42	91.9 ±0.02	92.5 ±0.19	83.2 ±0.39	85.2 ±0.12	73.8 ±0.83	79.3 ±0.30	77.6 ±0.20	79.3 ±0.48
w/ SimPro	**74.2** ±0.90	**80.7** ±0.30	**93.6** ±0.08	**93.8** ±0.10	**83.5** ±0.95	**85.8** ±0.48	**82.6** ±0.38	**84.8** ±0.54	**81.0** ±0.27	**83.0** ±0.36

Table 2. Top-1 accuracy (%) on CIFAR100-LT and STL10-LT with different class imbalance ratios γ_l and γ_u. Due to the unknown ground-truth labels of the unlabeled data for STL10, we conduct the experiments by controlling the imbalance ratio of the labeled data. † indicates we reproduce the results of ACR without anchor distributions for fair comparison.

	CIFAR100-LT ($\gamma_l = 20$, $N_1 = 50$, $M_1 = 400$)						STL10-LT ($\gamma_u = N/A$)	
	$\gamma_u = 20$	$\gamma_u = 1$	$\gamma_u = 1/20$	$\gamma_u = 20$	$\gamma_u = 20$		$N_1 = 450$,	$M = 1\times10^5$
	consistent	uniform	reversed	middle	head-tail		$\gamma_l = 10$	$\gamma_l = 20$
FixMatch (Sohn et al., 2020)	40.0 ±0.96	39.6 ±1.16	36.2 ±0.63	39.7 ±0.61	38.2 ±0.82	FixMatch (Sohn et al., 2020)	72.4 ±0.71	64.0 ±2.27
w/ CReST+ (Wei et al., 2021)	40.1 ±1.28	37.6 ±0.88	32.4 ±0.08	36.9 ±0.57	35.1 ±1.10	w/ CReST+ (Wei et al., 2021)	71.5 ±0.96	68.5 ±1.88
w/ DASO (Oh et al., 2022)	43.0 ±0.15	49.4 ±0.93	44.1 ±0.25	43.1 ±1.20	43.8 ±0.43	w/ DASO (Oh et al., 2022)	78.4 ±0.80	75.3 ±0.44
w/ ACR† (Wei & Gan, 2023)	40.7 ±0.57	50.2 ±0.82	44.1 ±0.14	42.4 ±0.47	41.1 ±0.09	w/ ACR (Wei & Gan, 2023)	83.0 ±0.32	81.5 ±0.25
w/ SimPro	**43.1** ±0.40	**52.2** ±0.16	**45.5** ±0.34	**43.6** ±0.35	**44.8** ±0.56	w/ SimPro	**84.5** ±0.39	**82.5** ±0.25

where $\mathcal{L}_l$ and $\mathcal{L}_u$ represent the losses on labeled and unlabeled data, respectively. The hyper-parameter α acts as a scaling factor, the specifics of which are elucidated later.

For $\mathcal{L}_l$, we modify it (originally the standard cross-entropy loss) following Eq. (8):

$$\mathcal{L}_l = -\frac{1}{B}\sum_{i=1}^{B}\log\frac{\exp(f_{\boldsymbol{\theta}}(x_i, y_i))}{\sum_{y'}\phi_{y'}^{\tau}\exp(f_{\boldsymbol{\theta}}(x_i, y'))}, \tag{14}$$

where τ is a hyper-parameter for enhancing adaptability to long-tail distributions (Menon et al., 2021). B is batch size.

For $\mathcal{L}_u$, we implement the standard SSL format (Eq. (1)) and adapt it for ReaLTSSL:

$$\mathcal{L}_u = -\frac{1}{\mu B}\sum_{j=1}^{\mu B}\mathbb{I}(\max_y(q_y)) \geq t)\sum_y q_y \log p_y, \tag{15}$$

where μ controls the number of unlabeled samples, and t is a confidence threshold. The pseudo-label of weak augmentation ω from the Bayes classifier (Eq. (10)) is denoted by

$$q_y = P(y|\omega(x_j); \boldsymbol{\theta}, \hat{\boldsymbol{\pi}}), \tag{16}$$

and the actual prediction p_y is obtained using strong augmentation Ω and calibrated with $\boldsymbol{\phi}$ as shown in Eq. (8):

$$p_y = \frac{\exp(f_{\boldsymbol{\theta}}(\Omega(x_j), y))}{\sum_{y'}\phi_{y'}^{\tau}\exp(f_{\boldsymbol{\theta}}(\Omega(x_j), y'))}. \tag{17}$$

Moreover, in practical situations, the size of the unlabeled dataset M is generally larger than that of the labeled dataset N. To ensure a balanced sample size in each iteration, we usually set $\mu = M/N$ in Eq. (15). In specific scenarios, we further adjust the balance factor in Eq. (13) to $\alpha = \mu \cdot N/M < 1$. This methodology effectively mitigates overfitting to the labeled data (see Tabs. 6 and 7).

Closed-form solution of $\boldsymbol{\pi}$. As discussed in Sec. 3.2, the parameter $\boldsymbol{\pi}$ of marginal distribution $P(y)$, has a closed-form solution in the M-step. Therefore, unlike $\boldsymbol{\theta}$, which requires SGD optimization, $\boldsymbol{\pi}$ (Eq. (7)) and $\boldsymbol{\phi}$ (Eq. (9)) are computed and updated via a moving average during training.

Extended EM algorithm and pseudo-code. Based on the previous analysis, Our SimPro framework can be summarized as an extended EM algorithm, which includes:

· *E-step* (Eq. (10)): Generating pseudo-labels using model parameters $\boldsymbol{\theta}$ and estimated distribution parameters $\boldsymbol{\pi}$;

· *M-step*: Optimizing network parameters $\boldsymbol{\theta}$ via SGD using Eq. (8) (actually using Eq. (13)), and solving distribution parameters $\boldsymbol{\pi}$ and hyper-parameters $\boldsymbol{\phi}$ by Eq. (7) and Eq. (9).

For further clarity, the pseudo-code of SimPro is provided in Alg. 1. The modifications we made to the core training code, in comparison to FixMatch, are highlighted in bold. In the SimPro, we incorporate just a *single additional line* of code in the M-step to compute the closed-form solution of π (Prop. 1). Furthermore, only *four lines* of code need to be modified to construct a Bayes classifier (Prop. 2) and to balance the loss between labeled and unlabeled data (denoted as α). These minor yet crucial adjustments demonstrate that our SimPro framework is not only grounded in *rigorous theoretical derivation* but is also straightforward to implement in practice, exemplifying both *simplicity and elegance*.

Table 3. Top-1 accuracy (%) on ImageNet-127 ($\gamma_l = \gamma_u \approx 286$, $N_1 \approx 28$k, and $M_1 \approx 250$k) and ImageNet-1k ($\gamma_l = \gamma_u = 256$, $N_1 = 256$, $M_1 = 1024$) with different test class imbalance ratios γ_t and image resolutions. † indicates we reproduce ACR without anchor distributions for a fair comparison. The results of $\gamma_t \approx 286$ are sourced from ACR (Wei & Gan, 2023).

	$\gamma_t \approx 286$	
ImageNet-127	32×32	64×64
FixMatch (Sohn et al., 2020)	29.7	42.3
w/ DARP (Kim et al., 2020)	30.5	42.5
w/ CReST+ (Wei et al., 2021)	32.5	44.7
w/ CoSSL (Fan et al., 2022)	43.7	53.9
w/ ACR (Wei & Gan, 2023)	57.2	63.6
w/ SimPro	**59.1**	**67.0**
ImageNet-127	$\gamma_t = 1$	
FixMatch (Sohn et al., 2020)	38.7	46.7
w/ ACR† (Wei & Gan, 2023)	49.5	56.1
w/ ACR (Wei & Gan, 2023)	50.6	57.3
w/ SimPro	**55.7**	**63.8**
ImageNet-1k	$\gamma_t = 1$	
FixMatch (Sohn et al., 2020)	–	–
w/ ACR† (Wei & Gan, 2023)	13.2	23.4
w/ ACR (Wei & Gan, 2023)	13.8	23.3
w/ SimPro	**19.7**	**25.0**

4. Experiments

In this section, we first present the main results on various ReaLTSSL benchmarks in Sec. 4.1. More analysis, including the ablation studies and the visualization results, is presented in Sec. 4.2 to further evaluate the effectiveness of our SimPro. For detailed information regarding the experimental setup, please refer to App. B.

4.1. Results

We first conduct experiments on the four representative benchmark datasets with different class imbalance ratios. We denote the class imbalance ratio of labeled, unlabeled, and test data as γ_l, γ_u, and γ_t, respectively. Our method is compared with five competitive baseline approaches, i.e., FixMatch (Sohn et al., 2020), CReST+ (Wei et al., 2021), DASO (Oh et al., 2022), ACR (Wei & Gan, 2023), and CPE (Ma et al., 2024). Note that for a fair comparison, we first compare with ACR in the ReaLTSSL setting, where the unlabeled class distribution is unknown and inaccessible. Specifically, we compare our vanilla SimPro with ACR's variant that removes its pre-defined anchor distributions, denoted as ACR†. Then we implement SimPro⋆ by also alleviating the anchor distributions in our SimPro framework, comparing SimPro⋆ with the original ACR and CPE.

Main results and comparison with SOTA baselines. The results are presented in Tab. 1 (for CIFAR10-LT), Tab. 2 (for CIFAR100-LT and STL10-LT), and Tab. 3 (for ImageNet-127/1k). It can be concluded that our method consistently outperforms the competitors across all distributions of unlabeled data and achieves SOTA performance. Notably, SimPro exhibits significant performance improvements on our two newly introduced distributions of unlabeled data: middle and head-tail. This substantiates the robust generalization capabilities of SimPro across various distributions that could potentially appear in real-world scenarios.

Table 4. The impact of the predefined anchor distribution in ACR and CPE (Ma et al., 2024) on CIFAR10-LT with $\gamma_l = 150$, $N_1 = 500$, and $M_1 = 4000$. $\star$ denotes that we use the predefined anchor distributions to estimate $P(y|\boldsymbol{\pi})$ in our SimPro. See more analysis in the main text and more results in App. C.

	$\gamma_u = 150$	$\gamma_u = 1$	$\gamma_u = 1/150$	$\gamma_u = 150$	$\gamma_u = 150$
	consistent	uniform	reversed	middle	head-tail
CPE	76.8	81.0	80.8	–	–
ACR	77.0	91.3	81.8	77.9	79.0
SimPro	74.2	93.6	83.5	**82.6**	**81.0**
SimPro⋆	**80.0**	**94.1**	**85.0**	–	–

It is worth noting that compared to CIFAR10/100-LT, STL10-LT is a more challenging dataset that mirrors the real-world data distribution scenarios: an *unknown* distribution for the unlabeled data. The results in Tab. 2 demonstrate the significant improvements of SimPro over baseline methods.

Moreover, we also conduct experiments on ImageNet-127, whose test dataset is imbalanced and consistent with the labeled data and unlabeled data. However, this is not suitable as a benchmark for long-tail learning, as biased classifiers tend to perform well in such scenarios, which is precisely what we aim to avoid. Therefore, we resample it to achieve a uniform test distribution ($\gamma_t = 1$). The results highlight that our SimPro achieves substantial performance enhancements when evaluated against this balanced test dataset. Beyond this, we further conduct experiments on ImageNet-1k to validate the performance of our method across a broader range of classes. The results in Tab. 3 demonstrate that our SimPro achieves state-of-the-art performance on ImageNet-1k.

The results of SimPro⋆ using anchor distributions. To investigate the impact of the anchor distributions in ACR and CPE (Ma et al., 2024), we also incorporate them into our approach, referred to as SimPro⋆. Instead of calculating the distribution distance and adjusting the consistency regularization as in ACR or employing multiple anchor experts as in CPE, our usage of these anchors is more straightforward: after training for five epochs, we calculate the distance between our estimated distribution $P(y|\boldsymbol{\pi})$ and the three anchor distributions (i.e. consistent, uniform, and reversed). This calculation helps us predict the actual distribution and construct the Bayes classifier. Then we fix the marginal

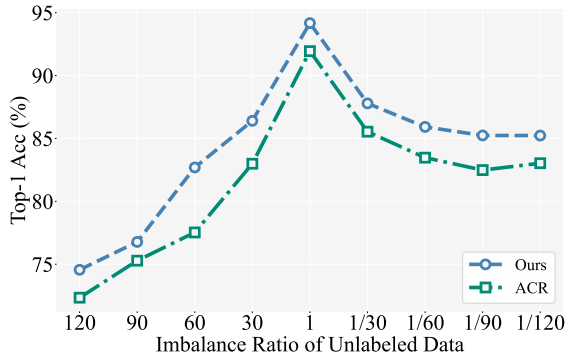

Figure 3. Test the performance under more imbalance ratios on CIFAR10-LT with $\gamma_l = 150$, $N_1 = 500$, and $M_1 = 4000$.

Table 5. Ablation study of estimating the marginal distribution $P(y|\boldsymbol{\pi})$ in M-step (Eq. (7)) and using it for constructing Bayes classifier in E-step (Eq. (5)). We conduct the experiments on CIFAR10-LT with $\gamma_l = 150$, $N_1 = 500$, and $M_1 = 4000$.

Distribution Estimation		$\gamma_u = 150$	$\gamma_u = 1$	$\gamma_u = 1/150$	$\gamma_u = 150$	$\gamma_u = 150$
E-step	M step	consistent	uniform	reversed	middle	head-tail
✗	✗	40.7	35.3	43.2	27.1	47.7
✗	✓	64.1	92.6	78.6	64.9	74.8
✓	✓	**74.2**	**93.6**	**83.5**	**82.6**	**81.0**

Table 6. Ablation study of $\mu = M/N$ (Eq. (15)) on CIFAR10-LT with $\gamma_l = 150$, $N_1 = 500$, and $M_1 = 4000$. For the baseline methods without our Bayes classifier, the performance drops significantly when we set $\mu = M/N = 8$. This large μ leads to an imbalance between labeled and unlabeled samples in each mini-batch. In contrast, our SimPro is not affected by such imbalance thanks to the Bayes classifier (Prop. 2). Moreover, we effectively leverage the large number of unlabeled data for a more accurate estimate of the marginal distribution parameters $\boldsymbol{\pi}$.

		$\gamma_u = 150$	$\gamma_u = 1/150$	$\gamma_u = 150$	$\gamma_u = 150$
	$\mu = M/N$	consistent	reversed	middle	head-tail
FixMatch	✗	62.9	59.9	64.3	58.3
	✓	40.7	43.2	27.1	47.7
w/ ACR	✗	70.9	83.2	73.8	77.6
	✓	68.7	58.9	69.7	72.4
w/ SimPro	✗	52.7	78.8	58.8	71.5
	✓	**75.2**	**83.5**	**82.6**	**81.0**

Table 7. Abalation study of α for balancing loss (Eq. (13)). The results indicate that α substantially improves the model's performance and prevents the model from overfitting to labeled data.

	CIFAR10-LT ($\gamma_u = 1$)		CIFAR100-LT ($\gamma_u = 1$)	STL10-LT ($\gamma_u =$ N/A)	
	$N_1 = 500, M_1 = 4000$		$N_1 = 50, M_1 = 400$	$N_1 = 450, M = 1\times10^5$	
α	$\gamma_l = 150$	$\gamma_l = 100$	$\gamma_l = 20$	$\gamma_l = 20$	$\gamma_l = 10$
✗	92.1	91.2	49.4	76.4	80.0
✓	**93.6**	**93.8**	**52.2**	**83.0**	**85.2**

distribution parameters $\boldsymbol{\pi}$ in the remainder of the training.

The results in Tab. 4 indicate that (1) the usage of anchor distributions can significantly enhance the performance of SimPro*, consistently outperforming the original ACR and CPE; (2) our estimation for $\boldsymbol{\pi}$ is accurate (Fig. 5 further validates the accurate estimation of $\boldsymbol{\pi}$); (3) when the pre-defined anchors fail to cover the evaluated distributions (middle and head-tail), SimPro outperforms ACR by a large margin; (4) even compared to the original ACR, SimPro exhibits enhanced performance across all scenarios except the consistent distribution. This demonstrates the superior *robustness* and *generalization ability* of our method. We believe these advantages guarantee a better *adaptable* nature for SimPro in real applications and are more valuable than the accuracy improvements when using the anchor distributions.

Evaluation under more imbalance ratios. Fig. 3 reports the performance of SimPro under more imbalance ratios of unlabeled data. The results indicate that our method consistently outperforms ACR across all imbalance ratios, further substantiating the robustness of our method.

4.2. Analysis

Ablation Study. We conduct a series of ablation studies to validate the effectiveness of different components in SimPro.

(a) *Marginal distribution estimation*. We first investigate the impact of the estimation for $P(y|\boldsymbol{\pi})$ (M-step, Prop. 1) and its usage in building the Bayes classifier for pseudo-label generation (E-step, Prop. 2). The results in Tab. 5 substantiate the high effectiveness and the necessity of this estimation in driving the success of our approach, thereby validating our methodology both theoretically and practically.

(b) *More unlabeled samples (larger μ) in each iteration.* As mentioned in the experimental setup, we manually set the ratio between unlabeled and labeled samples in each training iteration as $\mu = M/N$ (Eq. (15)). In ACR or Fixmatch, this ratio is set as 2. To investigate the impact of this adjustment, we adopt our setting of $\mu = M/N = 8$ for ACR and Fixmatch. The results in Tab. 6 demonstrate that our method can effectively leverage the unlabeled data for a more accurate estimation of the marginal distribution parameters $\boldsymbol{\pi}$. However, the baseline methods suffer from an imbalanced number of labeled/unlabeled samples, because of the absence of our Bayes classifier derived in Prop. 2.

(b) *Scaling factor α*. As elucidated in Sec. 3.3, we introduce a scaling factor $\alpha = \mu \cdot N/M$ (Eq. (13)) to mitigate the risk of overfitting. This measure is primarily due to memory and training environment limitations, which restrict the feasible batch size and μ, when $M \gg N$. In the test configurations detailed in Tab. 7, the ratio M/N is about 30 for CIFAR and 67 for STL10-LT. An insufficient μ results in a disproportionately high number of labeled data within a mini-batch, potentially leading to overfitting. Hence, we incorporate the α to balance the losses between labeled and unlabeled data. Our empirical findings demonstrate that the use of this

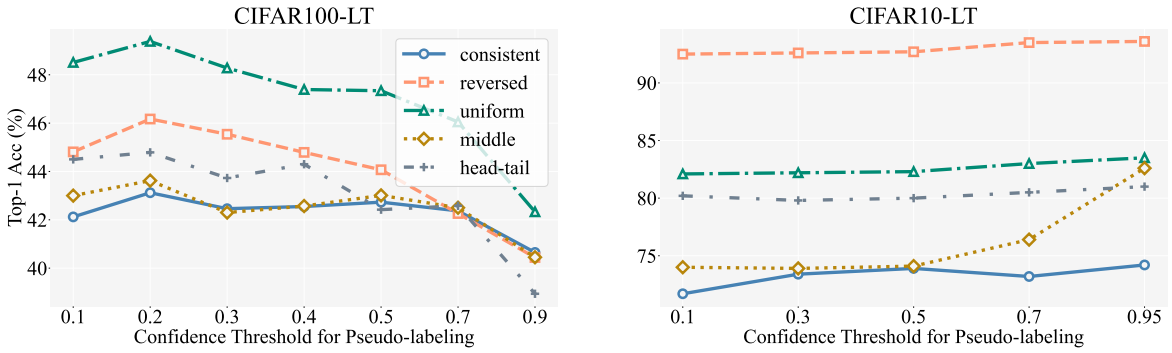

Figure 4. Sensitive analysis of the confidence threshold t on CIFAR100-LT with $\gamma_l = 20$, $N_1 = 50$, and $M_1 = 400$ and CIFAR10-LT with $\gamma_l = 150$, $N_1 = 500$, and $M_1 = 4000$. The optimal performance is consistently achieved across different settings when the threshold is set at $t = 0.2$ and 0.95, respectively.

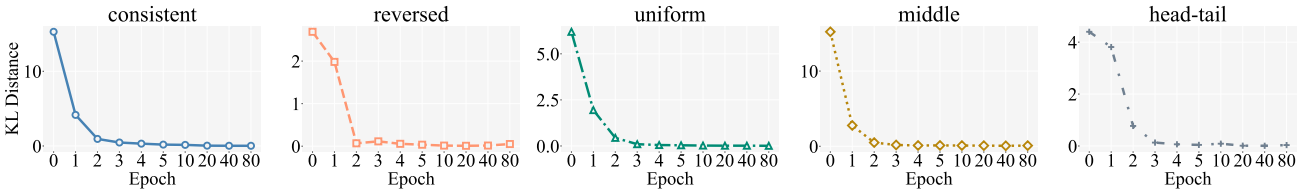

Figure 5. Visualization of the quality of the estimated distribution on CIFAR10-LT with $\gamma_l = 150$, $N_1 = 500$, and $M_1 = 4000$. The KL distances reduce to near-zero values after very few epochs.

simple parameter α can significantly enhance model performance, particularly STL10-LT, where there is a substantial disparity between the sizes of labeled and unlabeled data.

Hyperparameter Sensitivity. As outlined in App. B, we discover that reducing the threshold value improves performance for the CIFAR100-LT dataset. The rationale behind adjusting the confidence threshold is based on the observation that an increase in the number of classes typically results in a corresponding decrease in the confidence of the predictions. Consequently, it becomes necessary to lower the threshold to accommodate this change in confidence levels. Sensitivity analysis regarding the threshold value is presented in Fig. 4. It is consistently observed across different settings that the optimal performance is achieved when the threshold is set at $t = 0.2$ and $t = 0.95$ for the CIFAR100-LT and CIFAR10-LT, respectively. Moreover, to compare with ACR, we also conduct a sensitivity analysis of the confidence threshold t for ACR. The results in Fig. 6 of App. C demonstrate that lowering the threshold does not improve performance for ACR.

Visualization of Estimation Quality. Our study includes a visualization of the estimated distribution quality in Fig. 5. The vertical axis quantifies the Kullback-Leibler (KL) divergence, which measures the deviation of the estimated distribution from the ground truth. The results indicate a significant improvement in estimation accuracy after only a few training epochs. This empirical evidence validates the effectiveness of the theoretically derived estimation method of distribution, as outlined in Prop. 1.

5. Conclusion

In this paper, we introduce SimPro, a novel probabilistic framework for realistic long-tailed semi-supervised learning (ReaLTSSL). This framework represents a pioneering advancement in the field by innovatively enhancing the Expectation-Maximization (EM) algorithm. The key innovation lies in the explicit separation of the estimation process for conditional and marginal distributions. In the M-step, this separation allows for the derivation of a *closed-form* solution for the marginal distribution parameters. Additionally, SimPro optimizes the parameters of the conditional distribution via gradient descent, facilitating the learning of a *Bayes classifier*. In the E-step. the Bayes classifier, in turn, generates high-quality pseudo-labels. SimPro is characterized by its *elegant* theoretical underpinnings and its *ease* of implementation, which requires only minimal modifications to existing codebases. Furthermore, we incorporate two innovative class distributions specifically for unlabeled data. These distributions provide a more comprehensive and realistic evaluation of ReaLTSSL algorithms. Empirical evidence from various benchmarks demonstrates that SimPro consistently delivers state-of-the-art performance, highlighting its robustness and superior generalization capabilities.

Acknowledgements

This work was supported in part by the National Key R&D Program of China under Grant 2021ZD0140407, in part by the National Natural Science Foundation of China under Grants 62276150, 62321005 and 42327901.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Berthelot, D., Carlini, N., Cubuk, E. D., Kurakin, A., Sohn, K., Zhang, H., and Raffel, C. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. *arXiv preprint*, 2019a.

Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., and Raffel, C. A. Mixmatch: A holistic approach to semi-supervised learning. In *NeurIPS*, 2019b.

Coates, A., Ng, A., and Lee, H. An analysis of single-layer networks in unsupervised feature learning. In *AISTATS*, volume 15, pp. 215–223. PMLR, 2011.

Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society: series B (methodological)*, 39(1):1–22, 1977.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

Du, C., Wang, Y., Song, S., and Huang, G. Probabilistic contrastive learning for long-tailed visual recognition. *TPAMI*, 2024.

Fan, Y., Dai, D., Kukleva, A., and Schiele, B. Cossl: Co-learning of representation and classifier for imbalanced semi-supervised learning. In *CVPR*, 2022.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, 2016.

Hong, Y., Han, S., Choi, K., Seo, S., Kim, B., and Chang, B. Disentangling label distribution for long-tailed visual recognition. In *CVPR*, 2021.

Huang, G. and Du, C. The high separation probability assumption for semi-supervised learning. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 52 (12):7561–7573, 2022.

Kang, B., Xie, S., Rohrbach, M., Yan, Z., Gordo, A., Feng, J., and Kalantidis, Y. Decoupling representation and classifier for long-tailed recognition. In *ICLR*, 2020.

Kim, J., Hur, Y., Park, S., Yang, E., Hwang, S. J., and Shin, J. Distribution aligning refinery of pseudo-label for imbalanced semi-supervised learning. In *NeurIPS*, 2020.

Koller, D. and Friedman, N. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

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

Lai, Z., Wang, C., Gunawan, H., Cheung, S.-C. S., and Chuah, C.-N. Smoothed adaptive weighting for imbalanced semi-supervised learning: Improve reliability against unknown distribution data. In *ICML*, 2022.

Lee, D.-H. et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, pp. 896. Atlanta, 2013.

Lee, H., Shin, S., and Kim, H. Abc: Auxiliary balanced classifier for class-imbalanced semi-supervised learning. In *NeurIPS*, 2021.

Liu, Z., Miao, Z., Zhan, X., Wang, J., Gong, B., and Yu, S. X. Large-scale long-tailed recognition in an open world. In *CVPR*, 2019.

Loshchilov, I. and Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*, 2017.

Ma, C., Elezi, I., Deng, J., Dong, W., and Xu, C. Three heads are better than one: Complementary experts for long-tailed semi-supervised learning. In *AAAI*, 2024.

Menon, A. K., Jayasumana, S., Rawat, A. S., Jain, H., Veit, A., and Kumar, S. Long-tail learning via logit adjustment. In *ICLR*, 2021.

Miyato, T., Maeda, S.-i., Koyama, M., and Ishii, S. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *TPAMI*, 2018.

Oh, Y., Kim, D.-J., and Kweon, I. S. DASO: Distribution-aware semantics-oriented pseudo-label for imbalanced semi-supervised learning. In *CVPR*, 2022.

Ren, J., Yu, C., Ma, X., Zhao, H., and Yi, S. Balanced meta-softmax for long-tailed visual recognition. In *NeurIPS*, 2020.

Sohn, K., Berthelot, D., Carlini, N., Zhang, Z., Zhang, H., Raffel, C. A., Cubuk, E. D., Kurakin, A., and Li, C.-L. FixMatch: Simplifying semi-supervised learning with consistency and confidence. In *NeurIPS*, 2020.

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

Wang, Y., Guo, J., Wang, J., Wu, C., Song, S., and Huang, G. Erratum to meta-semi: A meta-learning approach for semi-supervised learning. *CAAI Artificial Intelligence Research*, 2023.

Wei, C., Sohn, K., Mellina, C., Yuille, A., and Yang, F. CReST: A class-rebalancing self-training framework for imbalanced semi-supervised learning. In *CVPR*, 2021.

Wei, T. and Gan, K. Towards realistic long-tailed semi-supervised learning: Consistency is all you need. In *CVPR*, 2023.

Wei, T., Liu, Q.-Y., Shi, J.-X., Tu, W.-W., and Guo, L.-Z. Transfer and share: semi-supervised learning from long-tailed data. *Machine Learning*, pp. 1–18, 2022.

Zagoruyko, S. and Komodakis, N. Wide residual networks. In *Procedings of the British Machine Vision Conference 2016*. British Machine Vision Association, 2016.

A. Details of the Probabilistic Model

We provide a detailed derivation and analysis to demonstrate that the probabilistic model is correctly defined with an explicit parameterization of $p(x;\xi)$. Given the independence of parameters through conditional parameterization (Koller & Friedman, 2009), we can decompose the joint probability distribution as follows:

$$p(x,y;\theta,\pi,\xi) = p(x;\xi)p(y|x;\theta,\pi) = p(y;\pi)p(x|y;\theta,\xi). \tag{18}$$

Applying Bayes' rule, we obtain:

$$\frac{p(y|x;\theta,\pi)}{p(y;\pi)} = \frac{p(x|y;\theta,\xi)}{p(x;\xi)}. \tag{19}$$

It is evident that π and ξ appear only on the left and right sides of the equation, respectively, indicating that the equation is neither a function of π nor ξ but is parameterized solely by θ. We define the above equation as $g(x,y;\theta)$, that is:

$$\frac{p(x|y;\theta,\xi)}{p(x;\xi)} = \frac{p(y|x;\theta,\pi)}{p(y;\pi)} = g(x,y;\theta). \tag{20}$$

Returning to the equation in the main paper, we have:

$$p(y|x;\theta,\pi) = \frac{p(y;\pi)p(x|y;\theta,\xi)}{p(x;\xi)} = p(y;\pi)g(x,y;\theta). \tag{21}$$

Although we explicitly parameterize $p(x;\xi)$, it is clear that $p(y|x;\theta,\pi)$ is parameterized solely by θ and π, and is independent of ξ. In fact, the fitting target of the network parameters θ is $g(x,y;\theta) = p(x|y;\theta,\xi)/p(x;\xi)$.

Since we did not explicitly parameterize $p(x)$ in our framework, when x is not a condition, the parameters of the relevant notions omit the parameters of $p(x)$, such as $p(x)$, $p(x|y)$, $p(x,y)$, etc. This may lead to a potential misunderstanding, as:

$$p(x) = \sum_y p(x|y;\theta)p(y;\pi) \tag{22}$$

suggests that $p(x)$ seems to be parameterized by θ and π. However, if we recover the omitted parameter of $p(x)$, we have:

$$p(x;\xi) = \sum_y p(x|y;\theta,\xi)p(y;\pi) = p(x;\xi)\sum_y g(x,y;\theta)p(y;\pi) = p(x;\xi)\sum_y p(y|x;\theta,\pi) = p(x;\xi), \tag{23}$$

which is consistent with the explicit parameterization of $p(x;\xi)$.

Therefore, the probabilistic model is correctly defined without the explicit parameterization of $p(x)$.

B. Experimental Setup

Datasets. To validate the effectiveness of SimPro, we employ five commonly used SSL datasets, CIFAR10/100 (Krizhevsky et al., 2009), STL10 (Coates et al., 2011), ImageNet-127 (Fan et al., 2022) and orginal ImageNet-1k (Deng et al., 2009). Following the methodology described in ACR (Wei & Gan, 2023), we denote the number of samples for each category in the labeled, unlabeled dataset as $N_1 \geq \cdots \geq N_K$, $M_1 \geq \cdots \geq M_K$, respectively, where $1, \cdots, K$ are the class indices. We define γ_l, γ_u, and γ_t as the class imbalance ratios for labeled, unlabeled, and test data, respectively. We specify 'LT' for those imbalanced variants. These ratios are calculated as $\gamma_l = N_1/N_K$ and $\gamma_u = M_1/M_K$. The sample number of the k-th class follows an exponential distribution, expressed as $N_k = N_1 \cdot \gamma_l^{-\frac{k-1}{K-1}}$ for labeled and $M_k = M_1 \cdot \gamma_u^{-\frac{k-1}{K-1}}$ for unlabeled data.

As illustrated in Fig. 1, for the CIFAR10/100 datasets, we constructed five class distributions to test the performance of different algorithms under more general settings. Regarding the STL10 dataset, due to the unknown ground-truth labels of the unlabeled data, we approach the experiments by controlling the imbalance ratio of the labeled data.

For ImageNet-127, we follow the original setting in Fan et al. (2022) ($\gamma_l = \gamma_u = \gamma_t \approx 286$). Nevertheless, this approach does not serve as an appropriate benchmark for long-tail learning. In these scenarios, biased classifiers often exhibit high performance, which is exactly the outcome we seek to prevent. Consequently, we also resample the test dataset to ensure a uniform distribution ($\gamma_t = 1$). Following Fan et al. (2022), we reduce the image resolution to 32×32 and 64×64 in response to resource constraints.

Training hyper-parameters. Our experimental setup mainly follows FixMatch (Sohn et al., 2020) and ACR (Wei & Gan, 2023). For example, we employ the Wide ResNet-28-2 (Zagoruyko & Komodakis, 2016) on CIFAR10/100 and STL10, and ResNet-50 (He et al., 2016) on ImageNet-127. All models are optimized with SGD. Several settings are slightly different from those in ACR: as outlined in Sec. 3.3, to achieve a balanced training, we set the ratio between unlabeled and labeled samples in each batch as $\mu = M/N$ (8 on CIFAR, 16 on STL10-LT, and 2 on ImageNet-127,), where M, N are the total sample numbers of unlabeled/labeled data. In contrast, μ is set as 2 for all datasets in ACR. The effectiveness of this adjustment is validated in Tab. 6.

The batch size for labeled data is 64 on CIFAR10/100 and STL10-LT, and 32 on ImageNet-127. To ensure a fair comparison, the training epochs are reduced to 86 on CIFAR10/100 and STL10-LT, and 500 on ImageNet-127. The initial learning rate η is linearly scaled to 0.17 on CIFAR10/100 and STL10-LT, and 0.01 on ImageNet-127, which decays with a cosine schedule (Loshchilov & Hutter, 2017) as in ACR.

Regarding the hyperparameter τ used in Eq. (14), we follow the guidelines from Menon et al. (2021) and set $\tau = 2.0$ and 1.0 for CIFAR10-LT/STL10-LT and CIFAR100-LT/ImageNet-127, respectively.

For the confidence threshold t in Eq. (1)), we set $t = 0.95$ on CIFAR10-LT/STL10-LT following Sohn et al. (2020). We adjust it to 0.2 on CIFAR100-LT/Imagenet-127, as we observe that reducing the threshold enhances performance (Fig. 4).

Specifically, the settings on ImageNet-1k are identical to those on ImageNet-127.

C. More Experimental Results

Table 8. The impact of the predefined anchor distribution in ACR (Wei & Gan, 2023) on CIFAR10-LT with $\gamma_l = 100$, $N_1 = 500$, and $M_1 = 4000$. $\star$ denotes that we use the predefined anchor distributions to estimate $P(y|\boldsymbol{\pi})$ in our SimPro.

	$\gamma_u = 150$	$\gamma_u = 1$	$\gamma_u = 1/150$	$\gamma_u = 150$	$\gamma_u = 150$
	consistent	uniform	reversed	middle	head-tail
ACR	81.6	92.1	85.0	73.6	79.8
SimPro	80.7	93.8	85.8	**84.8**	**83.0**
SimPro*	**82.7**	**94.3**	**86.0**	–	–

Table 9. The impact of the predefined anchor distribution in ACR (Wei & Gan, 2023) on CIFAR100-LT with $\gamma_l = 20$, $N_1 = 50$, and $M_1 = 400$. $\star$ denotes that we use the predefined anchor distributions to estimate $P(y|\boldsymbol{\pi})$ in our SimPro.

	$\gamma_u = 150$	$\gamma_u = 1$	$\gamma_u = 1/150$	$\gamma_u = 150$	$\gamma_u = 150$
	consistent	uniform	reversed	middle	head-tail
ACR	44.9	52.2	42.3	42.6	42.6
SimPro	43.1	52.3	45.5	**43.6**	**44.8**
SimPro*	**45.9**	**53.8**	**46.0**	–	–

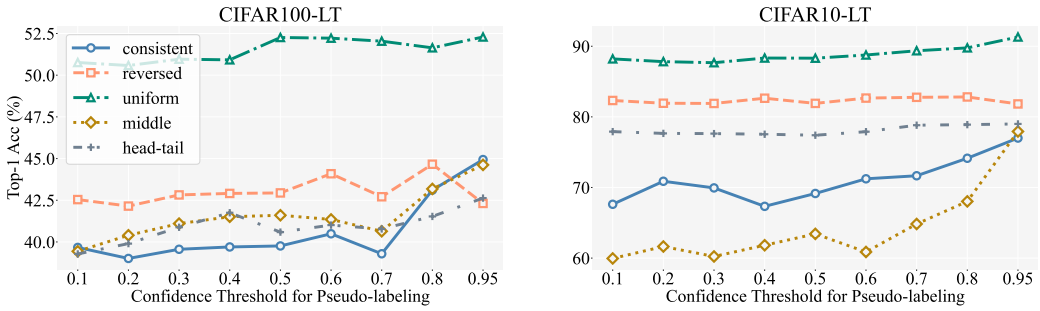

Figure 6. Sensitive analysis of the confidence threshold t for ACR (Wei & Gan, 2023) on CIFAR100-LT with $\gamma_l = 20$, $N_1 = 50$, and $M_1 = 400$ and CIFAR10-LT with $\gamma_l = 150$, $N_1 = 500$, and $M_1 = 4000$. The optimal performance is achieved across different settings when the threshold is set at 0.95.

D. Proof of Proposition 1

Proof. We employ the method of Lagrange multipliers to find the optimal values of $\hat{\boldsymbol{\pi}}_l$ and $\hat{\boldsymbol{\pi}}_u$ that maximize the $\mathcal{Q}$ function subject to the constraints of probability distributions (i.e., the elements of $\boldsymbol{\pi}_l$ and $\boldsymbol{\pi}_u$ must sum to 1). Let λ_l and λ_u be the Lagrange multipliers for these constraints. The Lagrangian $\mathcal{L}$ can be formulated as [1]:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\pi}_l, \boldsymbol{\pi}_u, \lambda_l, \lambda_u) = &\sum_i \log P(y_i; \boldsymbol{\pi}_l) + \sum_{j,y} P(y|x_j; \boldsymbol{\theta}', \boldsymbol{\pi}') \log P(y; \boldsymbol{\pi}_u) \\ &- \lambda_l \left(\sum_y \pi_{ly} - 1 \right) - \lambda_u \left(\sum_y \pi_{uy} - 1 \right). \end{aligned} \tag{24}$$

The partial derivatives of $\mathcal{L}$ with respect to $\boldsymbol{\pi}_l$ and $\boldsymbol{\pi}_u$ are calculated as follows:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\pi}_l} = \frac{\partial}{\partial \boldsymbol{\pi}_l} \left(\sum_i \log P(y_i; \boldsymbol{\pi}_l) \right) - \lambda_l \frac{\partial}{\partial \boldsymbol{\pi}_l} \left(\sum_y \pi_{ly} - 1 \right), \tag{25}$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\pi}_u} = \frac{\partial}{\partial \boldsymbol{\pi}_u} \left(\sum_{j,y} P(y|x_j; \boldsymbol{\theta}', \boldsymbol{\pi}') \log P(y; \boldsymbol{\pi}_u) \right) - \lambda_u \frac{\partial}{\partial \boldsymbol{\pi}_u} \left(\sum_y \pi_{uy} - 1 \right). \tag{26}$$

Due to fact that $P(y; \boldsymbol{\pi})$ is a categorical distribution, the derivatives are:

$$\frac{\partial \mathcal{L}}{\partial \pi_{ly}} = \sum_i \frac{\mathbf{1}_{\{y_i = y\}}}{\pi_{ly}} - \lambda_l, \tag{27}$$

$$\frac{\partial \mathcal{L}}{\partial \pi_{uy}} = \sum_j P(y|x_j; \boldsymbol{\theta}', \boldsymbol{\pi}') \frac{1}{\pi_{uy}} - \lambda_u. \tag{28}$$

Setting these partial derivatives to zero and solving for π_{ly} and π_{uy}:

$$\hat{\pi}_{ly} = \frac{\sum_i \mathbf{1}_{\{y_i = y\}}}{\lambda_l}, \tag{29}$$

$$\hat{\pi}_{uy} = \frac{\sum_j P(y|x_j; \boldsymbol{\theta}', \boldsymbol{\pi}')}{\lambda_u}. \tag{30}$$

Applying the constraint that the sum of probabilities equals 1, we get:

$$\sum_y \hat{\pi}_{ly} = 1 \Rightarrow \lambda_l = \sum_y \sum_i 1 = N, \tag{31}$$

$$\sum_y \hat{\pi}_{uy} = 1 \Rightarrow \lambda_u = \sum_y \sum_j P(y|x_j; \boldsymbol{\theta}', \boldsymbol{\pi}') = M. \tag{32}$$

Therefore, the optimal solutions are:

[1] y is the one-hot label, but when y is the subscript of a variable, it represents the y-th category, that is, $\phi_y = \phi_{\operatorname{argmax} y}$. we omit this difference without affecting the understanding.

$$\hat{\boldsymbol{\pi}}_l = \frac{1}{N}\sum_{i=1}^{N} y_i, \quad \hat{\boldsymbol{\pi}}_u = \frac{1}{M}\sum_{j=1}^{M} P(y|x_j;\boldsymbol{\theta}',\boldsymbol{\pi}'). \tag{33}$$

□

E. Proof of Proposition 2

Proof. We structure our proof in two parts: First, we validate the proposition when training exclusively with labeled data. Then, we extend our analysis to include scenarios incorporating unlabeled data. This approach stems from our threshold-based strategy for filtering low-confidence pseudo labels in the training process. Initially, only labeled data are used, gradually integrating unlabeled data as training progresses.

Case 1: Labeled Data Only

Our proof begins by revisiting the definition of $\mathcal{Q}(\boldsymbol{\theta},\boldsymbol{\pi};\boldsymbol{\theta}',\boldsymbol{\pi}')$ with respect to $\boldsymbol{\theta}$:

$$\begin{aligned}\mathcal{Q}(\boldsymbol{\theta}) &= \sum_i \log \frac{\exp(f_{\boldsymbol{\theta}}(x_i,y_i))}{\sum_{y'}\phi_{y'}\exp(f_{\boldsymbol{\theta}}(x,y'))} \\ &= \sum_i \left(\log \frac{\phi_{y_i}\exp(f_{\boldsymbol{\theta}}(x_i,y_i))}{\sum_{y'}\phi_{y'}\exp(f_{\boldsymbol{\theta}}(x,y'))} - \log\phi_y\right).\end{aligned} \tag{34}$$

Ignoring the constant term, maximizing $\mathcal{Q}(\boldsymbol{\theta})$ is equivalent to minimizing the empirical risk:

$$R_{\text{emp}}(\boldsymbol{\theta}) = -\frac{1}{N}\sum_i \log \frac{\phi_{y_i}\exp(f_{\boldsymbol{\theta}}(x_i,y_i))}{\sum_{y'}\phi_{y'}\exp(f_{\boldsymbol{\theta}}(x,y'))}. \tag{35}$$

The empirical risk serves as an approximation of the expected risk, with $\mathcal{D}_l$ denoting the distribution of labeled data:

$$\begin{aligned}R_{\text{exp}}(\boldsymbol{\theta}) &= -\mathbb{E}_{(x,y)\sim\mathcal{D}_l}\log Q(y|x) \\ &= -\int_x \sum_y P(y|x)\log Q(y|x)\ \mathrm{d}x \\ &= \int_x H(P(y|x)) + D_{\text{KL}}(P(y|x)||Q(y|x))\ \mathrm{d}x,\end{aligned} \tag{36}$$

where $Q(y|x)$ is defined as:

$$Q(y|x) = \frac{\phi_y\exp(f_{\boldsymbol{\theta}}(x,y))}{\sum_{y'}\phi_{y'}\exp(f_{\boldsymbol{\theta}}(x,y'))}. \tag{37}$$

With the non-negativity and zero-equality conditions of KL divergence, minimizing the expected risk implies:

$$\frac{\phi_y\exp(f_{\boldsymbol{\theta}}(x,y))}{\sum_{y'}\phi_{y'}\exp(f_{\boldsymbol{\theta}}(x,y'))} = P(y|x) = \frac{\pi_{ly}P(x|y)}{P_l(x)}. \tag{38}$$

We aim to validate the Bayes classifier:

$$P(y|x;\boldsymbol{\theta},\boldsymbol{\pi}) = \frac{P(y;\boldsymbol{\pi})\exp(f_{\boldsymbol{\theta}}(x,y))}{\sum_{y'}P(y';\boldsymbol{\pi})\exp(f_{\boldsymbol{\theta}}(x,y'))}, \tag{39}$$

which leads to the formulation:

$$\frac{\pi_y P(x|y)}{\hat{P}(x)} = \frac{\pi_y\exp(f_{\boldsymbol{\theta}}(x,y))}{\sum_{y'}\pi_{y'}\exp(f_{\boldsymbol{\theta}}(x,y'))}. \tag{40}$$

Integrating Eq. (38) with Eq. (40) yields:

$$\frac{\phi_y}{\sum_{y'}\phi_{y'}\exp(f_{\boldsymbol{\theta}}(x,y'))} = \frac{\pi_{ly}\hat{P}(x)}{P_l(x)\sum_{y'}\pi_{y'}\exp(f_{\boldsymbol{\theta}}(x,y'))}. \tag{41}$$

Summing over y in Eq. (41) leads to:

$$\sum_y \phi_y = C = \frac{\hat{P}(x) \sum_{y'} \phi_{y'} \exp(f_{\boldsymbol{\theta}}(x, y'))}{P_l(x) \sum_{y'} \pi_{y'} \exp(f_{\boldsymbol{\theta}}(x, y'))]}. \tag{42}$$

Substituting Eq. (42) into Eq. (41) results in:

$$\phi_y = C\pi_{ly}. \tag{43}$$

As the constant C becomes irrelevant in the logarithmic term of $\mathcal{Q}(\boldsymbol{\theta})$, in light of Prop. 1, we deduce the optimal $\hat{\boldsymbol{\phi}}$:

$$\hat{\boldsymbol{\phi}} = \frac{1}{N} \sum_{i=1}^{N} y_i. \tag{44}$$

Case 2: Labeled and Unlabeled Data

Expanding our proof to include both labeled and unlabeled data, our optimization objective remains consistent: maximizing $\mathcal{Q}(\boldsymbol{\theta})$ by minimizing the empirical risk:

$$R_{\text{emp}}(\boldsymbol{\theta}) = -\frac{1}{N+M}(\sum_i + \sum_{j,y} P(y|x_j; \boldsymbol{\theta}', \boldsymbol{\pi}')) \log Q(y|x). \tag{45}$$

This risk approximates the expected risk over $\mathcal{D}$:

$$R_{\text{exp}}(\boldsymbol{\theta}) = -\mathbb{E}_{(x,y)\sim\mathcal{D}} \log Q(y|x), \tag{46}$$

where $\mathcal{D}$ represents the mixture distribution of both labeled and unlabeled data and its density is:

$$P_{\mathcal{D}}(x, y) = \frac{N}{M+N} P_l(x, y) + \frac{M}{M+N} P_u(x) P(y|x; \boldsymbol{\theta}', \boldsymbol{\pi}'). \tag{47}$$

Acknowledging that $P(y|x; \boldsymbol{\theta}', \boldsymbol{\pi}')$ is a Bayes classifier, we conclude:

$$P_{\mathcal{D}}(x, y) = \left(\frac{N}{M+N} \pi_{ly} + \frac{M}{M+N} \pi_{uy} \right) P(x|y), \tag{48}$$

which leads to the formulation:

$$P_{\mathcal{D}}(y) = \frac{N}{M+N} \pi_{ly} + \frac{M}{M+N} \pi_{uy}, \quad P_{\mathcal{D}}(x|y) = P(x|y). \tag{49}$$

Following the same logic as in the labeled data case:

$$\frac{\phi_y \exp(f_{\boldsymbol{\theta}}(x, y))}{\sum_{y'} \phi_{y'} \exp(f_{\boldsymbol{\theta}}(x, y'))} = P_{\mathcal{D}}(y|x) = \frac{P_{\mathcal{D}}(y) P(x|y)}{P_{\mathcal{D}}(x)}. \tag{50}$$

This results in:

$$\phi_y = C \cdot P_{\mathcal{D}}(y). \tag{51}$$

In accordance with Prop. 1, we determine the optimal $\hat{\boldsymbol{\phi}}$ as:

$$\begin{aligned} \hat{\boldsymbol{\phi}} &= \frac{N}{M+N} \hat{\boldsymbol{\pi}}_l + \frac{M}{M+N} \hat{\boldsymbol{\pi}}_u \\ &= \frac{1}{N+M}(\sum_i y_i + \sum_j P(y|x_j; \boldsymbol{\theta}', \boldsymbol{\pi}')). \end{aligned} \tag{52}$$

□

F. Proof of Proposition 3

Proof. We begin by examining $P(e;\hat{\boldsymbol{\phi}})$. The decision error rate is mathematically defined as follows:

$$P(e) = \int_x P(e|x)P(x)\ \mathrm{d}x, \quad P(e|x) = \begin{cases} P(-1|x) & \text{if decision is } +1; \\ P(+1|x) & \text{if decision is } -1. \end{cases} \tag{53}$$

This formulation quantifies the error in decision-making by integrating the conditional error rates across all possible outcomes.

Building upon the Bayes optimal classifier as outlined in Eq. (11), the posterior probability essential for decision-making on the test set is expressed as:

$$P_d(y|x) \propto \exp(f_{\boldsymbol{\theta}}(x,y)). \tag{54}$$

Here, $f_{\boldsymbol{\theta}}(x,y)$ represents the model's discriminative function, parameterized by $\boldsymbol{\theta}$, for decision outcome y given an input x.

The ground truth distribution, as approximated by parameters $\hat{\boldsymbol{\phi}}$ and $\boldsymbol{\theta}$, is denoted by $P(y;\boldsymbol{\phi}^*)$ and $P(x|y)$, respectively. The formal relationship between the estimated and true distributions is expressed as follows:

$$\frac{\hat{\phi}_y \exp(f_{\boldsymbol{\theta}}(x,y))}{\sum_{y'} \hat{\phi}_{y'} \exp(f_{\boldsymbol{\theta}}(x,y'))} = \frac{\phi^*_y P(x|y)}{\sum_{y'} \phi^*_{y'} P(x|y')}, \tag{55}$$

indicating a proportional relationship between the model's estimation and the true data distribution.

Integrating the formulation of posterior probability with the relationship between estimated and true distributions allows us to derive the following expression:

$$\begin{aligned} P_d(y|x) &\propto \exp(f_{\boldsymbol{\theta}}(x,y)) \\ &\propto \frac{\phi^*_y}{\hat{\phi}_y} P(x|y) \\ &\propto \frac{\phi^*_y}{\hat{\phi}_y} P(y|x), \end{aligned} \tag{56}$$

where the final step is justified by the fact that the class distribution in the test set is uniform.

The decision criterion is thus formulated as:

$$\text{decision} = \begin{cases} +1 & \text{if } l(x) \geq \lambda; \\ -1 & \text{if } l(x) \leq \lambda, \end{cases} \tag{57}$$

where

$$l(x) = \frac{P(+1|x)}{P(-1|x)}, \quad \lambda = \frac{\hat{\phi}_{+1}\phi^*_{-1}}{\hat{\phi}_{-1}\phi^*_{+1}}. \tag{58}$$

We assume $\lambda \leq 1$ without loss of generality, due to the symmetric nature of the decision problem.

The decision error rate is then expressed as:

$$P(e;\hat{\boldsymbol{\phi}}) = \int_x P(e|x;\hat{\boldsymbol{\phi}})P(x)\ \mathrm{d}x, \quad P(e|x;\hat{\boldsymbol{\phi}}) = \begin{cases} P(-1|x) & \text{if } l(x) \geq \lambda; \\ P(+1|x) & \text{if } l(x) \leq \lambda. \end{cases} \tag{59}$$

Minimizing the decision error rate is achieved when $\hat{\boldsymbol{\phi}} = \boldsymbol{\phi}^*$, leading to the Bayes decision error rate, which is the theoretical lower bound of error rates across all possible distributions.

$$\inf_{\boldsymbol{\phi}} P(e;\boldsymbol{\phi}) = P(e;\boldsymbol{\phi}^*) = \int_x P(e|x;\boldsymbol{\phi}^*)P(x)\ \mathrm{d}x, \quad P(e|x;\boldsymbol{\phi}^*) = \begin{cases} P(-1|x) & \text{if } l(x) \geq 1; \\ P(+1|x) & \text{if } l(x) \leq 1. \end{cases} \tag{60}$$

Finally, comparing $P(e;\hat{\boldsymbol{\phi}})$ with the optimal decision error rate, we explore the difference:

$$\begin{aligned}
P(e;\hat{\boldsymbol{\phi}}) - \inf_{\boldsymbol{\phi}} P(e;\boldsymbol{\phi}) &= \int_{\lambda \le l(x) \le 1} |P(-1|x) - P(+1|x)| P(x)\ \mathrm{d}x \\
&= \int_{\lambda \le l(x) \le 1} |1 - l(x)| P(-1|x) P(x)\ \mathrm{d}x \\
&\le (1-\lambda) \int_{\lambda \le l(x) \le 1} P(-1,x)\ \mathrm{d}x \\
&\le (1-\lambda) \int_{x} P(-1,x)\ \mathrm{d}x \\
&= (1-\lambda) P(-1) \\
&= \frac{1-\lambda}{2} \\
&= \frac{\hat{\phi}_{-1} - \phi^*_{-1}}{2\phi^*_{+1}\hat{\phi}_{-1}} \\
&\le \frac{1}{2\phi^*_{+1}\phi^*_{-1}} (\hat{\phi}_{-1} - \phi^*_{-1}) \\
&= \frac{1}{2\phi^*_{+1}\phi^*_{-1}} |\hat{\phi} - \phi^*|,
\end{aligned} \tag{61}$$

where the final inequality is justified by the condition that $\hat{\phi}_{-1} \ge \phi^*_{-1}$ due to $\lambda \le 1$.

□