Binary Classification from Multiple Unlabeled Datasets via Surrogate Set Classification

Nan Lu [*1 2] **Shida Lei** [*1] **Gang Niu** [2] **Issei Sato** [1 2] **Masashi Sugiyama** [2 1]

Abstract

To cope with high annotation costs, training a classifier only from *weakly supervised data* has attracted a great deal of attention these days. Among various approaches, strengthening supervision from completely unsupervised classification is a promising direction, which typically employs *class priors* as the only supervision and trains a binary classifier from *unlabeled* (U) datasets. While existing *risk-consistent* methods are theoretically grounded with high flexibility, they can learn only from *two* U sets. In this paper, we propose a new approach for binary classification from m U sets for $m \geq 2$. Our key idea is to consider an auxiliary classification task called *surrogate set classification* (SSC), which is aimed at predicting from which U set each observed sample is drawn. SSC can be solved by a standard (multi-class) classification method, and we use the SSC solution to obtain the final binary classifier through a certain linear-fractional transformation. We built our method in a flexible and efficient *end-to-end* deep learning framework and prove it to be *classifier-consistent*. Through experiments, we demonstrate the superiority of our proposed method over state-of-the-art methods.

1. Introduction

Deep learning with large-scale supervised training data has shown great success on various tasks (Goodfellow et al., 2016). However, in practice, obtaining *strong supervision*, e.g., the complete ground-truth labels, for big data is very costly due to the expensive and time-consuming manual annotations (Zhou, 2018). Thus, it is desirable for machine learning techniques to work with *weaker forms of supervision*, such as noisy labels (Natarajan et al., 2013; Patrini et al., 2017; Van Rooyen & Williamson, 2018; Han et al., 2018; 2020; Fang et al., 2020; Xia et al., 2020), partial labels (Cour et al., 2011; Ishida et al., 2017; 2019; Feng et al., 2020; Lv et al., 2020), and pairwise comparison information (Bao et al., 2018; Xu et al., 2019; Feng et al., 2021).

This paper focuses on a challenging setting which we call U^m *classification*: the goal is to learn a binary classifier from m ($m \geq 2$) sets of U data with different *class priors*, i.e., the proportion of positives in each U set. Such a learning scheme can be conceivable in many real-world scenarios. For example, U sets with different class priors can be naturally collected from spatial or temporal differences. Considering morbidity rates, they can be potential patient data collected from different areas (Croft et al., 2018). Likewise, considering approval rates, they can be unlabeled voter data collected in different years (Newman, 2003). In such cases, individual labels are often not available due to privacy reasons, but the corresponding class priors of U sets, i.e., the morbidity rates or approval rates in the aforementioned examples, can be obtained from related medical reports or pre-existing census (Quadrianto et al., 2009; Ardehaly & Culotta, 2017; Tokunaga et al., 2020), and is the unique weak supervision that will be leveraged in this work.

Breakthroughs in U^m classification research were brought by Menon et al. (2015) and Lu et al. (2019) in proposing the *risk-consistent* methods given two U sets. Recently, Scott & Zhang (2020) extended them to incorporate multiple U sets by two steps: firstly, pair all the U sets so that they are sufficiently different in each pair; secondly, linearly combine the unbiased balanced risk estimators obtained from each pair. Although this method is advantageous since it is compatible with any model and stochastic optimizer, and is statistically consistent, there are several issues that may limit its potential for practical use: first, the computational complexity for the optimal pairing strategy is $O(m^3)$ for m U sets (Edmonds & Karp, 1972), which cannot work efficiently with a large number of U sets; second, the optimal combination weights are proved with strong model assumptions and thus remaining difficult to be tuned in practice.

Now, a natural question arises: can we propose a computationally efficient method for U^m classification with both *flexibility* on the choice of models and optimizers and *theo-*

[*]Equal contribution [1]The University of Tokyo, Tokyo, Japan [2]RIKEN, Tokyo, Japan. Correspondence to: Nan Lu <lu@ms.k.u-tokyo.ac.jp>, Shida Lei <leishida@is.s.u-tokyo.ac.jp>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

retical guarantees? The answer is affirmative.

In this paper, we provide a new approach for U^m classification by solving a Surrogate Set Classification task (U^m-SSC). More specifically, we regard the index of each U set as a *surrogate-set label* and consider the supervised multi-class classification task of predicting the surrogate-set labels given observations. The difficulty is how to link our desired binary classifier with the learned surrogate multi-class classifier. To solve it, we theoretically bridge the original and surrogate class-posterior probabilities with a linear-fractional transformation, and then implement it by adding a transition layer to the neural network so that the trained model is guaranteed to be a good approximation of the original class-posterior probability. Our proposed U^m-SSC scheme is built within an end-to-end framework, which is computationally efficient, compatible with any model architecture and stochastic optimization, and naturally incorporates multiple U sets. Our contributions can be summarized as follows:

- Theoretically, we prove that the proposed U^m-SSC method is *classifier-consistent* (Patrini et al., 2017; Lv et al., 2020), i.e., the classifier learned by solving the surrogate set classification task from multiple sets of U data converges to the optimal classifier learned from fully supervised data under mild conditions. Then we establish an *estimation error bound* of our method.
- Practically, we propose an easy-to-implement, flexible, and computationally efficient method for U^m classification, which is shown to outperform the state-of-the-art methods in experiments. We also verify the robustness of the proposed method by simulating U^m classification in the wild, e.g., on varied set sizes, set numbers, noisy class priors, and the results are promising.

Our method provides new perspectives of solving the U^m classification problem, and is more suitable to be applied in practice given its theoretical and practical advantages.

2. Problem Setup and Related Work

In this section, we introduce some notations, formulate the U^m classification problem, and review the related work.

2.1. Learning from Fully Labeled Data

Let $\mathcal{X}$ be the input feature space and $\mathcal{Y} = \{+1, -1\}$ be a binary label space, $\boldsymbol{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$ be the input and output random variables following an underlying joint distribution $\mathcal{D}$. Let $f : \mathcal{X} \to \mathbb{R}$ be an arbitrary binary classifier, and $\ell_{\mathrm{b}}(t, y) : \mathbb{R} \times \mathcal{Y} \to \mathbb{R}_+$ be a *loss function* such that the value $\ell_{\mathrm{b}}(t, y)$ means the loss by predicting t when the ground-truth is y. The goal of binary classification is to train a classifier f that minimizes the *risk* defined as

$$R(f) = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}[\ell_{\mathrm{b}}(f(\boldsymbol{x}), y)] \tag{1}$$

where $\mathbb{E}$ denotes the expectation. For evaluation, ℓ_{b} is often chosen as $\ell_{01}(t, y) = (1 - \mathrm{sign}((t - \frac{1}{2}) \cdot y))/2$ and then the risk R becomes the standard performance measure for classification, a.k.a. the *classification error*. For training, ℓ_{01} is replaced by a *surrogate loss*,[1] e.g., the logistic loss $\ell_{\log}(t, y) = \ln(1 + \exp(-t \cdot y))$, since ℓ_{01} is discontinuous and therefore difficult to optimize (Ben-David et al., 2003).

In most cases, R cannot be calculated directly because the joint distribution $\mathcal{D}$ is unknown to the learner. Given the labeled training set $\mathcal{X} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n \overset{\text{i.i.d.}}{\sim} \mathcal{D}$ with n samples, *empirical risk minimization* (ERM) (Vapnik, 1998) is a common practice that computes an approximation of R by

$$\widehat{R}(f) = \frac{1}{n}\sum\nolimits_{i=1}^{n} \ell_{\mathrm{b}}(f(\boldsymbol{x}_i), y_i). \tag{2}$$

2.2. Learning from Multiple Sets of U Data

Next, we consider U^m classification. We are given $m (m \geq 2)$ sets of unlabeled samples drawn from m marginal densities $\{p_{\mathrm{tr}}^j(\boldsymbol{x})\}_{j=1}^m$, where

$$p_{\mathrm{tr}}^j(\boldsymbol{x}) = \pi_j p_{\mathrm{p}}(\boldsymbol{x}) + (1 - \pi_j) p_{\mathrm{n}}(\boldsymbol{x}), \tag{3}$$

each $p_{\mathrm{tr}}^j(\boldsymbol{x})$ is seen as a mixture of the positive and negative class-conditional densities $(p_{\mathrm{p}}(\boldsymbol{x}), p_{\mathrm{n}}(\boldsymbol{x})) = (p(\boldsymbol{x}|y = +1), p(\boldsymbol{x}|y = -1))$, and $\pi_j = p_{\mathrm{tr}}^j(y = +1)$ denotes the class prior of the j-th U set. Note that given only U data, it is *theoretically impossible* to learn the class priors without any assumptions (Menon et al., 2015), so we assume all necessary class priors are given, which are the only *weak supervision* we will leverage.[2] To make the problem mathematically solvable, among the m sets of U data, we also assume that at least two of them are different, i.e., $\exists j, j' \in \{1, \ldots, m\}$ such that $j \neq j'$ and $\pi_j \neq \pi_{j'}$.

In contrast to supervised classification where we have a fully labeled training set $\mathcal{X}$ directly drawn from $\mathcal{D}$, now we only have access to m sets of U data $\mathcal{X}_{\mathrm{tr}} = \{\mathcal{X}_{\mathrm{tr}}^j\}_{j=1}^m$, where

$$\mathcal{X}_{\mathrm{tr}}^j = \{\boldsymbol{x}_1^j, \ldots, \boldsymbol{x}_{n_j}^j\} \overset{\text{i.i.d.}}{\sim} p_{\mathrm{tr}}^j(\boldsymbol{x}), \tag{4}$$

and n_j denotes the sample size of the j-th U set. But our goal is still the same as supervised classification: to obtain a binary classifier that generalizes well with respect to $\mathcal{D}$, despite the fact that it is unobserved.

2.3. Related Work

Here, we review some related works for U^m classification.

[1] The surrogate loss ℓ_{s} should be *classification-calibrated* so that the predictions can be the same for classifiers learned by using ℓ_{s} and ℓ_{01} (Bartlett et al., 2006).

[2] By introducing the *mutually irreducible assumption* (Scott et al., 2013), the class priors become identifiable and can be estimated in some cases, see Menon et al. (2015), Liu & Tao (2016), Jain et al. (2016), and Yao et al. (2020) for details.

Table 1. Comparisons of the proposed method with previous works in the U^m classification setting.

Methods	Deal with 2+ sets	Theoretical guarantee	No negative training risk	Pre-computing complexity	Risk Measure
$\widehat{R}_{\mathrm{U}^2}(f)$ (Lu et al., 2019)	×	✓	×	$O(1)$	Classification risk (1)
$\widehat{R}_{\mathrm{U}^2\text{-b}}(f)$ (Menon et al., 2015)	×	✓	×	$O(1)$	Balanced risk (7)
$\widehat{R}_{\mathrm{U}^2\text{-c}}(f)$ (Lu et al., 2020)	×	✓	✓	$O(1)$	Classification risk (1)
$\widehat{R}_{\mathrm{U}^\mathrm{m}}(f)$ (Scott & Zhang, 2020)	✓	✓	×	$O(m^3)$	Balanced risk (7)
$\widehat{R}_{\mathrm{prop\text{-}c}}(f)$ (Tsai & Lin, 2020)	✓	×	✓	$O(1)$	Proportion risk (5)
Proposed	✓	✓	✓	$O(1)$	Classification risk (1)

Clustering methods Learning from only U data is previously regarded as *discriminative clustering* (Xu et al., 2004; Gomes et al., 2010). However, these methods are often suboptimal since they rely on a critical assumption that *one cluster exactly corresponds to one class*, and hence even perfect clustering may still result in poor classification. As a consequence, we prefer ERM to clustering.

Proportion risk methods The U^m classification setting is also related to *learning with label proportions* (LLP), with a subtle difference in the experimental design.[3] However, most LLP methods are not ERM-based, but based on the following *empirical proportion risk* (EPR) (Yu et al., 2014):

$$\widehat{R}_{\mathrm{prop}}(f) = \sum\nolimits_{j=1}^{m} d_{\mathrm{prop}}(\pi_j, \hat{\pi}_j), \tag{5}$$

where π_j and $\hat{\pi}_j = \frac{1}{n_j}\sum_{i=1}^{n_j}(1+\mathrm{sign}(f(\boldsymbol{x}_i^j)-1/2))/2$ are the true and predicted label proportions for the j-th U set $\mathcal{X}_{\mathrm{tr}}^j$, and d_{prop} is a distance function. State-of-the-art method in this line combined EPR with consistency regularization and proposed the following learning objective:

$$\widehat{R}_{\mathrm{prop\text{-}c}}(f) = \widehat{R}_{\mathrm{prop}}(f) + \alpha\ell_{\mathrm{cons}}(f), \tag{6}$$

where $\ell_{\mathrm{cons}}(f) = d_{\mathrm{cons}}(f(\boldsymbol{x}), f(\hat{\boldsymbol{x}}))$ is the consistency loss given a distance function d_{cons} and $\hat{\boldsymbol{x}}$ is a perturbed input from the original one $\boldsymbol{x}$ (Tsai & Lin, 2020).

Classification risk methods A breakthrough of the ERM-based method for U^m classification is Lu et al. (2019) which assumed $m=2$ and $\pi_1 > \pi_2$, and proposed an equivalent expression of the classification risk (1):

$$R_{\mathrm{U}^2}(f) = \underbrace{\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{tr}}^1} c_1^+ \ell_{\mathrm{b}}(f(\boldsymbol{x}),+1) - \mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{tr}}^2} c_2^+ \ell_{\mathrm{b}}(f(\boldsymbol{x}),+1)}_{R_{\mathrm{U}^2\text{-p}}(f)}$$
$$\underbrace{-\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{tr}}^1} c_1^- \ell_{\mathrm{b}}(f(\boldsymbol{x}),-1) + \mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{tr}}^2} c_2^- \ell_{\mathrm{b}}(f(\boldsymbol{x}),-1)}_{R_{\mathrm{U}^2\text{-n}}(f)},$$

where $c_1^+ = \frac{(1-\pi_2)\pi_{\mathcal{D}}}{\pi_1-\pi_2}$, $c_1^- = \frac{\pi_2(1-\pi_{\mathcal{D}})}{\pi_1-\pi_2}$, $c_2^+ = \frac{(1-\pi_1)\pi_{\mathcal{D}}}{\pi_1-\pi_2}$, $c_2^- = \frac{\pi_1(1-\pi_{\mathcal{D}})}{\pi_1-\pi_2}$, and $\pi_{\mathcal{D}}$ denotes the class prior of the test set. If $\pi_{\mathcal{D}}$ is assumed to be $\frac{1}{2}$ in $R_{\mathrm{U}^2}(f)$, the obtained $R_{\mathrm{U}^2\text{-b}}(f)$ (Menon et al., 2015) corresponds to the balanced risk, a.k.a. the *balanced error* (Brodersen et al., 2010):

$$R_{\mathrm{b}}(f) = \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{p}}}[\ell_{\mathrm{b}}(f(\boldsymbol{x}),+1)] + \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{n}}}[\ell_{\mathrm{b}}(f(\boldsymbol{x}),-1)], \tag{7}$$

where ℓ_{b} is ℓ_{01}. Note that $R_{\mathrm{b}}(f) = R(f)$ for any f if and only if $\pi_{\mathcal{D}} = \frac{1}{2}$, which means that it definitely biases learning when $\pi_{\mathcal{D}} \approx \frac{1}{2}$ is not the case. Given $\mathcal{X}_{\mathrm{tr}}^1$ and $\mathcal{X}_{\mathrm{tr}}^2$, $R_{\mathrm{U}^2}(f)$ and $R_{\mathrm{U}^2\text{-b}}(f)$ can be approximated by their empirical counterparts $\widehat{R}_{\mathrm{U}^2}(f)$ and $\widehat{R}_{\mathrm{U}^2\text{-b}}(f)$.

It is shown in Lu et al. (2020) that the empirical training risk $\widehat{R}_{\mathrm{U}^2}(f)$ can take negative values which causes overfitting, so they proposed a corrected learning objective that wraps the empirical risks of the positive class $\widehat{R}_{\mathrm{U}^2\text{-p}}(f)$ and the negative class $\widehat{R}_{\mathrm{U}^2\text{-n}}(f)$ into some non-negative correction function f_{c}, such that $f_{\mathrm{c}}(x) = x$ for all $x \geq 0$ and $f_{\mathrm{c}}(x) > 0$ for all $x < 0$: $\widehat{R}_{\mathrm{U}^2\text{-c}}(f) = f_{\mathrm{c}}(\widehat{R}_{\mathrm{U}^2\text{-p}}(f)) + f_{\mathrm{c}}(\widehat{R}_{\mathrm{U}^2\text{-n}}(f))$. Note that $\widehat{R}_{\mathrm{U}^2\text{-c}}$ is biased with finite samples, but Lu et al. (2020) showed its risk-consistency, i.e., it converges to the original risk R in (1) if $n_1, n_2 \to \infty$.

Although these risk-consistent methods are advantageous in terms of flexibility and theoretical guarantees, they are limited to 2 U sets. Recently, Scott & Zhang (2020) extended the previous method for the general $m(m \geq 2)$ setting. More specifically, they assumed the number of sets $m = 2k$ and proposed a pre-processing step that finds k pairs of the U sets by solving a maximum weighted matching problem (Edmonds, 1965). Then they linearly combine the unbiased balanced risk estimator of each pair.[4] The resulted weighted learning objective is given by

$$\widehat{R}_{\mathrm{U}^\mathrm{m}}(f) = \sum\nolimits_{j=1}^{k} \omega_j \widehat{R}_{\mathrm{U}^2\text{-b}}(f). \tag{8}$$

This method is promising but has some practical issues:

[3] The majority of LLP papers use uniform sampling for bag generation, which may result in the same label proportion for all the U sets and make the LLP problem computationally intractable (Scott & Zhang, 2020). Our simulation in Sec. 4 avoids the issue.

[4] In Scott & Zhang (2020), it is assumed that $\pi_1 \neq \pi_2$ in each pair, which is a stronger assumption than ours.

the pairing step is computationally very inefficient and the weights are hard to tune in practice.

A comparison of the previous works with our proposed method that will be introduced in Sec. 3 is given in Table 1.

3. U^m classification via Surrogate Set Classification

In this section, we propose a new ERM-based method for learning from multiple U sets via a surrogate set classification task and analyze it theoretically. All the proofs are given in Appendix A.

3.1. Surrogate Set Classification Task

The main challenge in the U^m classification problem is that we have *no* access to the ground-truth labels of the training examples so that the empirical risk (2) in supervised binary classification cannot be computed directly. Our idea is to consider a surrogate set classification task that could be tackled easily from the given U sets. It serves as a proxy and gives us a classifier-consistent solution to the original binary classification problem.

Specifically, denote by $\bar{y} \in \{1, 2, \ldots, m\}$ the index of the U set, i.e., the index of the corresponding marginal density. By treating $\bar{y}$ as a *surrogate label*, we formulate the surrogate set classification task as the standard multi-class classification. Let $\bar{\mathcal{D}}$ be the joint distribution for the random variables $\boldsymbol{x} \in \mathcal{X}$ and $\bar{y} \in \bar{\mathcal{Y}} = \{1, 2, \ldots, m\}$. Any $\bar{\mathcal{D}}$ can be identified via the class priors $\{\rho_j = p(\bar{y} = j)\}_{j=1}^m$ and the class-conditional densities $\{p(\boldsymbol{x} \mid \bar{y} = j) = p_{\mathrm{tr}}^j(\boldsymbol{x})\}_{j=1}^m$, where ρ_j can be estimated by $\rho_j = \frac{n_j}{\sum_{j=1}^m n_j}$.

The goal of surrogate set classification is to train a classifier $\boldsymbol{g}(\boldsymbol{x}) : \mathcal{X} \to \mathbb{R}^m$ that minimizes the following risk:

$$R_{\mathrm{surr}}(\boldsymbol{g}) = \mathbb{E}_{(\boldsymbol{x},\bar{y})\sim\bar{\mathcal{D}}}[\ell(\boldsymbol{g}(\boldsymbol{x}), \bar{y})], \tag{9}$$

where $\ell(\boldsymbol{g}(\boldsymbol{x}), \bar{y}) : \mathbb{R}^m \times \bar{\mathcal{Y}} \to \mathbb{R}_+$ is a proper loss for m-class classification, e.g., the *cross-entropy loss*:

$$\ell_{\mathrm{ce}}(\boldsymbol{g}(\boldsymbol{x}), \bar{y}) = -\sum_{j=1}^{m} \mathbf{1}(\bar{y} = j) \log(g_j(\boldsymbol{x})) = -\log(g_{\bar{y}}(\boldsymbol{x})),$$

where $\mathbf{1}(\cdot)$ is the indicator function, $g_j(\boldsymbol{x})$ is the j-th element of $\boldsymbol{g}(\boldsymbol{x})$, and is a score function that estimates the true class-posterior probability $\bar{\eta}_j(\boldsymbol{x}) = p(\bar{y} = j \mid \boldsymbol{x})$. Typically, the predicted label $\bar{y}_{\mathrm{pred}}$ takes the form $\bar{y}_{\mathrm{pred}} = \mathrm{argmax}_{j\in[m]}\, g_j(\boldsymbol{x})$.

Now the unlabeled training sets given by (4) for the binary classification can be seen as a labeled training set $\mathcal{X}_{\mathrm{tr}} = \{(\boldsymbol{x}_i, \bar{y}_i)\}_{i=1}^{n_{\mathrm{tr}}} \overset{\text{i.i.d.}}{\sim} \bar{\mathcal{D}}$ for the m-class classification, where $n_{\mathrm{tr}} = \sum_{j=1}^m n_j$ is the total number of U data. We can use $\mathcal{X}_{\mathrm{tr}}$ to approximate the risk R_{surr} by

$$\widehat{R}_{\mathrm{surr}}(\boldsymbol{g}) = \frac{1}{n_{\mathrm{tr}}} \sum\nolimits_{i=1}^{n_{\mathrm{tr}}} \ell(\boldsymbol{g}(\boldsymbol{x}_i), \bar{y}_i). \tag{10}$$

3.2. Bridge Two Posterior Probabilities

Let $\eta(\boldsymbol{x}) = p(y = +1 \mid \boldsymbol{x})$ be the class-posterior probability for class $+1$ in the original binary classification problem, and $\bar{\eta}_j(\boldsymbol{x}) = p(\bar{y} = j \mid \boldsymbol{x})$ be the class-posterior probability for class j in the surrogate set classification problem. We theoretically bridge them by the following theorem.

Theorem 1. *By the definitions of $\mathcal{D}$, $\eta(\boldsymbol{x})$, $\bar{\mathcal{D}}$, and $\bar{\eta}_j(\boldsymbol{x})$, we have*

$$\bar{\eta}_j(\boldsymbol{x}) = T_j(\eta(\boldsymbol{x})), \quad \forall j = 1, \ldots, m, \tag{11}$$

where

$$T_j(\eta(\boldsymbol{x})) = \frac{a_j \cdot \eta(\boldsymbol{x}) + b_j}{c \cdot \eta(\boldsymbol{x}) + d},$$

$a_j = \rho_j(\pi_j - \pi_{\mathcal{D}})$, $b_j = \rho_j \pi_{\mathcal{D}}(1 - \pi_j)$, $c = \sum_{j=1}^m \rho_j(\pi_j - \pi_{\mathcal{D}})$, and $d = \sum_{j=1}^m \rho_j \pi_{\mathcal{D}}(1 - \pi_j)$.

Such a relationship has been previously studied by Menon et al. (2015) in the context of *corrupted label learning* for a specific 2×2 case, i.e., 2 clean classes are transformed to 2 corrupted classes, and they used $T_j(\cdot)$ to post-process the threshold of the score function learned from corrupted data. Our proposal can be regarded as its extension to a general $2 \times m$ case and $T_j(\cdot)$ is used to connect the original binary classifier with the surrogate multi-set-class classifier.

Let $\boldsymbol{T}(\cdot) : \mathbb{R} \to \mathbb{R}^m$ be a vector form of the transition function $\boldsymbol{T}(\cdot) = [T_1(\cdot), \ldots, T_m(\cdot)]^\top$. Note that the coefficients in $T_j(\cdot)$ are all constants and $\boldsymbol{T}(\cdot)$ is deterministic. Next, we study properties of the transition function $\boldsymbol{T}$ in the following lemma, which implies the feasibility of approaching $\eta(\boldsymbol{x})$ by means of estimating $\bar{\eta}_j(\boldsymbol{x})$.

Lemma 2. *The transition function $\boldsymbol{T}(\cdot)$ is an injective function in the domain $[0, 1]$.*

3.3. Classifier-consistent Algorithm

Given the transition function $\boldsymbol{T}$, we have two choices to obtain $\eta(\boldsymbol{x})$ from $\bar{\eta}_j(\boldsymbol{x})$. First, one can estimate $\bar{\eta}_j(\boldsymbol{x})$, then calculate $\eta(\boldsymbol{x})$ via the inverse function $T_j^{-1}(\bar{\eta}_j(\boldsymbol{x}))$. Second, one can encode $\eta(\boldsymbol{x})$ as a latent variable into the computation of $\bar{\eta}_j(\boldsymbol{x})$ and obtain both of them simultaneously. We prefer the latter for three reasons.

- Computational efficiency: the latter is a one-step solution and avoids additional computations of the inverse functions, which provides computational efficiency and easiness for implementation.
- Robustness: since the coefficients of $T_j(\cdot)$ may be perturbed by some noise in practice, its inversion in

Algorithm 1 U^m-SSC based on stochastic optimization

Input: Model f, m sets of unlabeled data $\mathcal{X}_{\mathrm{tr}}$, class priors $\{\pi_j\}_{j=1}^m$ and $\pi_{\mathcal{D}}$

Compute a_j, b_j, c and d of $\boldsymbol{T}(\cdot) = [T_1(\cdot), \ldots, T_m(\cdot)]^\top$ in Theorem 1 using $\{\pi_j\}_{j=1}^m$ and $\pi_{\mathcal{D}}$.
Let $\boldsymbol{g} = \boldsymbol{T}(f)$ and $\mathcal{A}$ be a SGD-like optimizer working on $\boldsymbol{g}$.
for $t = 1, 2, \ldots$,number_of_epochs **do**
  Shuffle $\mathcal{X}_{\mathrm{tr}}$
  for $i = 1, 2, \ldots$,number_of_mini-batches **do**
    Fetch mini-batch $\bar{\mathcal{X}}_{\mathrm{tr}}$ from $\mathcal{X}_{\mathrm{tr}}$
    Forward $\bar{\mathcal{X}}_{\mathrm{tr}}$ and get $f(\bar{\mathcal{X}}_{\mathrm{tr}})$
    Compute $\boldsymbol{g}\left(\bar{\mathcal{X}}_{\mathrm{tr}}\right) = \boldsymbol{T}\left(f(\bar{\mathcal{X}}_{\mathrm{tr}})\right)$
    Compute loss by (10) using $\boldsymbol{g}(\bar{\mathcal{X}}_{\mathrm{tr}})$
    Update $\boldsymbol{g}$ by $\mathcal{A}$, which induces an update on f
  end for
end for

Output: f

the former method may enlarge the noise by orders of magnitude, making the learning process less robust.

- Identifiability: calculating $T_j^{-1}(\bar{\eta}_j(\boldsymbol{x}))$ in the former method for all $j = \{1, \ldots, m\}$ induces m estimates of $\eta(\boldsymbol{x})$, and they are usually non-identical due to the estimation error of $\bar{\eta}_j(\boldsymbol{x})$ from finite samples or noisy $T_j(\cdot)$, causing a new non-identifiable problem.

Therefore, we choose to embed the estimation of $\eta(\boldsymbol{x})$ into the estimation of $\bar{\eta}_j(\boldsymbol{x})$. More specifically, let $f(\boldsymbol{x})$ be the model output that estimates $\eta(\boldsymbol{x})$, then we make use of the transition function $T_j(\cdot)$ and model $g_j(\boldsymbol{x}) = T_j\left(f(\boldsymbol{x})\right)$. Based on it, we propose to learn with the following modified loss function:

$$\ell(\boldsymbol{g}(\boldsymbol{x}), \bar{y}) = \ell(\boldsymbol{T}(f(\boldsymbol{x})), \bar{y}), \tag{12}$$

where $\boldsymbol{T}(f(\boldsymbol{x})) = [T_1(f(\boldsymbol{x})), \ldots, T_m(f(\boldsymbol{x}))]^\top$. Then the corresponding risk for the surrogate task can be written as

$$\begin{aligned} R_{\mathrm{surr}}(f) &= \mathbb{E}_{(\boldsymbol{x},\bar{y})\sim\bar{\mathcal{D}}}[\ell(\boldsymbol{T}(f(\boldsymbol{x})), \bar{y})] \\ &= \mathbb{E}_{(\boldsymbol{x},\bar{y})\sim\bar{\mathcal{D}}}[\ell(\boldsymbol{g}(\boldsymbol{x}), \bar{y})] = R_{\mathrm{surr}}(\boldsymbol{g}), \end{aligned} \tag{13}$$

and an equivalent expression of the empirical risk (10) is given by

$$\widehat{R}_{\mathrm{surr}}(f) = \frac{1}{n_{\mathrm{tr}}} \sum\nolimits_{i=1}^{n_{\mathrm{tr}}} \ell(\boldsymbol{T}(f(\boldsymbol{x}_i)), \bar{y}_i). \tag{14}$$

In order to prove that this method is classifier-consistent, we introduce the following lemma.

Lemma 3. *Let $\bar{\boldsymbol{\eta}}(\boldsymbol{x}) = [\bar{\eta}_1(\boldsymbol{x}), \ldots, \bar{\eta}_m(\boldsymbol{x})]^\top$ and $\boldsymbol{g}^\star(\boldsymbol{x}) = \operatorname{argmin}_{\boldsymbol{g}} R_{\mathrm{surr}}(\boldsymbol{g}; \ell)$ be the optimal classifier of (9). Provided that a proper loss function, e.g., the cross-entropy loss or mean squared error, is chosen for ℓ, we have $\boldsymbol{g}^\star(\boldsymbol{x}) = \bar{\boldsymbol{\eta}}(\boldsymbol{x})$.*

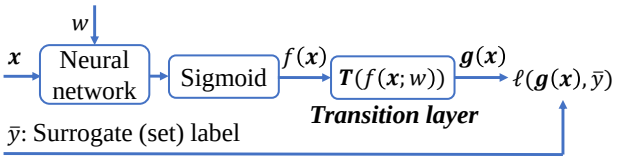

Figure 1. Implementation diagram of U^m-SSC.

Since $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{T}(f(\boldsymbol{x}))$ and $\boldsymbol{T}(\cdot)$ is deterministic, when considering minimizing $R_{\mathrm{surr}}(f)$ that takes f as the argument, we can prove the following classifier-consistency.

Theorem 4 (Identification of the optimal binary classifier)**.** *Assume that the cross-entropy loss or mean squared error is used for ℓ and ℓ_{b}, and the model $\mathcal{G}$ used for learning $\boldsymbol{g}$ is very flexible, e.g., deep neural networks, so that $\boldsymbol{g}^\star \in \mathcal{G}$. Let $f^\star_{\mathrm{surr}}$ be the U^m-SSC optimal classifier induced by $\boldsymbol{g}^\star$, and $f^\star = \operatorname{argmin}_f R(f; \ell_{\mathrm{b}})$ be the optimal classifier of (1), we have $f^\star_{\mathrm{surr}} = f^\star$.*

So far, we have proved that the optimal classifier for the original binary classification task can be identified by the U^m-SSC learning scheme. Its algorithm is described in Algorithm 1 and its implementation is illustrated in Figure 1.

We implement $\boldsymbol{T}(\cdot)$ by adding a transition layer following the sigmoid function of the neural network (NN). At the training phase, a sample $(\boldsymbol{x}_{\mathrm{tr}}, \bar{y}_{\mathrm{tr}})$ is fetched to the network. A sigmoid function $f_{\mathrm{sig}}(x) = \frac{1}{1+e^{-x}}$ is used to map the output of NN to the range $[0, 1]$ such that the output $f(\boldsymbol{x})$ is an estimate of $\eta(\boldsymbol{x})$. Then $f(\boldsymbol{x})$ is forwarded to the transition layer and a vector output $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{T}(f(\boldsymbol{x}))$ is obtained. The loss computed on the output $\boldsymbol{g}(\boldsymbol{x})$ and the surrogate label $\bar{y}_{\mathrm{tr}}$ by (10) is then used for updating the NN weights w. Note that the transition layer is fixed and only the weights in the base network are learnable. At the test phase, for any test sample $\boldsymbol{x}_{\mathrm{te}}$, we compute $f(\boldsymbol{x}_{\mathrm{te}})$ using only the trained base network and sigmoid function. The test sample is classified by using the sign function, i.e., $\operatorname{sign}(f(\boldsymbol{x}_{\mathrm{te}}) - \frac{1}{2})$. Our proposed method is model-agnostic and can be easily trained with a stochastic optimization algorithm, which ensures its scalability to large-scale datasets.

3.4. Theoretical Analysis

In what follows, we upper-bound the estimation error of our proposed method. Let $\hat{f}_{\mathrm{surr}} = \operatorname{argmin}_{f\in\mathcal{F}} \widehat{R}_{\mathrm{surr}}(f)$ be our empirical classifier, where $\mathcal{F} = \{f : \mathcal{X} \to \mathbb{R}\}$ is a class of measurable functions, and $f^\star_{\mathrm{surr}} = \operatorname{argmin}_{f\in\mathcal{F}} R_{\mathrm{surr}}(f)$ be the optimal classifier, the estimation error is defined as the gap between the risk of $\hat{f}_{\mathrm{surr}}$ and that of $f^\star_{\mathrm{surr}}$, i.e., $R_{\mathrm{surr}}(\hat{f}_{\mathrm{surr}}) - R_{\mathrm{surr}}(f^\star_{\mathrm{surr}})$. To derive the estimation error bound, we firstly investigate the Lipschitz continuity of the transition function $\boldsymbol{T}(f(\boldsymbol{x}))$.

Table 2. Specification of datasets and corresponding models.

Dataset	# Train	# Test	# Features	$\pi_{\mathcal{D}}$	Model
MNIST (LeCun et al., 1998)	60,000	10,000	784	0.49	5-layer MLP
Fashion-MNIST (Xiao et al., 2017)	60,000	10,000	784	0.8	5-layer MLP
Kuzushiji-MNIST (Clanuwat et al., 2018)	60,000	10,000	784	0.3	5-layer MLP
CIFAR-10 (Krizhevsky, 2009)	50,000	10,000	3,072	0.7	ResNet-32

Lemma 5. *Assume that among the m sets of U data, at least two of them are different, i.e., $\exists j, j' \in \{1, \dots, m\}$ such that $j \neq j'$ and $\pi_j \neq \pi_{j'}$, and $0 \leq f(\boldsymbol{x}) \leq 1, \forall x \in \mathcal{X}$, e.g., $f(\boldsymbol{x})$ is mapped to $[0, 1]$ by the sigmoid function. Then, $\forall j = 1, \dots, m$, the function $T_j(f(\boldsymbol{x}))$ is Lipschitz continuous w.r.t. $f(\boldsymbol{x})$ with a Lipschitz constant $2/\alpha^2$, where*

$$\alpha = \min\left(\sum_{j=1}^{m} \rho_j \pi_j (1 - \pi_{\mathcal{D}}), \sum_{j=1}^{m} \rho_j \pi_{\mathcal{D}} (1 - \pi_j)\right).$$

Then we analyze the estimation error as follows.

Theorem 6 (Estimation error bound)**.** *Assume that the loss $\ell(\boldsymbol{T}(f), \bar{y})$ is upper-bounded by M_ℓ and is $\mathcal{L}_\ell$-Lipschitz continuous w.r.t. $\boldsymbol{T}(f)$. Let $\mathfrak{R}_{n_{\mathrm{tr}}}(\mathcal{F})$ be the* Rademacher complexity *of $\mathcal{F}$ (Mohri et al., 2012; Shalev-Shwartz & Ben-David, 2014). Then, for any $\delta > 0$, we have with probability at least $1 - \delta$,*

$$R_{\mathrm{surr}}(\hat{f}_{\mathrm{surr}}) - R_{\mathrm{surr}}(f^{\star}_{\mathrm{surr}}) \leq \frac{8\sqrt{2}m\mathcal{L}_\ell}{\alpha^2}\mathfrak{R}_{n_{\mathrm{tr}}}(\mathcal{F}) + 2M_\ell\sqrt{\frac{\ln(2/\delta)}{2n_{\mathrm{tr}}}}. \tag{15}$$

Theorem 6 demonstrates that as the number of training samples goes to infinity, the risk of $\hat{f}_{\mathrm{surr}}$ converges to the risk of $f^{\star}_{\mathrm{surr}}$, since $\mathfrak{R}_{n_{\mathrm{tr}}}(\mathcal{F}) \to 0$ for all parametric models with a bounded norm. Moreover, the coefficient α implies that a tighter error bound could be obtained when the class priors π_j are close to 0 or 1. This conclusion agrees with our intuition that purer U sets (containing almost only positive/ negative examples) lead to better performance.

4. Experiments

In this section, we experimentally analyze the proposed method and compare it with state-of-the-art methods in the U^m classification setting.[5]

Datasets We train on widely adopted benchmarks MNIST, Fashion-MNIST, Kuzushiji-MNIST, and CIFAR-10. Table 2 briefly summarizes the benchmark datasets. Since the four datasets contain 10 classes originally, we manually corrupt them into binary classification datasets. More details about the datasets are in Appendix B.1.

In the experiments, unless otherwise specified, the number of training data contained in all U sets are the same and fixed as $n_j = n_{\mathrm{tr}}/m$ for all benchmark datasets; and the class priors $\{\pi_j\}_{j=1}^m$ of all U sets are randomly sampled from the range $[0.1, 0.9]$ under the constraint that the sampled class priors are not all identical, ensuring that the problem is mathematically solvable. Given $\{n_j\}_{j=1}^m$ and $\{\pi_j\}_{j=1}^m$, we generate m sets of U training data following (4). Note that in most LLP papers, each U set is uniformly sampled from the shuffled U training data, therefore the label proportions of all the U sets are the same in expectation. As the set size increases, all the proportions converge to the same class prior, making the LLP problem computationally intractable (Scott & Zhang, 2020). As shown above, our experimental scheme avoids this issue by determining valid class priors before sampling each U set.

Models The models are optimizers used are also described in Table 2, where MLP refers to *multi-layer perceptron*, ResNet refers to *residual networks* (He et al., 2016), and their detailed architectures are in Appendix B.2. As a common practice, we use Adam (Kingma & Ba, 2015) with the cross-entropy loss for optimization. We train 300 epochs for all the experiments, and the classification error rates at the test phase are reported. All the experiments are repeated 3 times and the mean values with standard deviations are recorded for each method.

Baselines We compare the proposed method with state-of-the-art methods based on the classification risk (Scott & Zhang, 2020) and the empirical proportion risk (Tsai & Lin, 2020) for the U^m classification problem. Recall that the proposed learning objective $\widehat{R}_{\mathrm{U^m}}(f)$ in Scott & Zhang (2020) is a combination of the unbiased balanced risk estimators $\widehat{R}_{\mathrm{U^2\text{-}b}}(f)$, which are shown to underperform the unbiased risk estimator $\widehat{R}_{\mathrm{U^2}}(f)$ in Lu et al. (2019). So we improve the baseline method of Scott & Zhang (2020) by combining $\widehat{R}_{\mathrm{U^2}}(f)$ instead of $\widehat{R}_{\mathrm{U^2\text{-}b}}(f)$. As shown in Lu et al. (2020), the empirical risks $\widehat{R}_{\mathrm{U^2}}(f)$ can go negative during training which may cause overfitting, so we further improve the baseline by combining the corrected non-negative risk estimators $\widehat{R}_{\mathrm{U^2\text{-}c}}(f)$. The baselines are summarized as follows:

[5] Our implementation of U^m-SSC is available at https://github.com/leishida/Um-Classification.

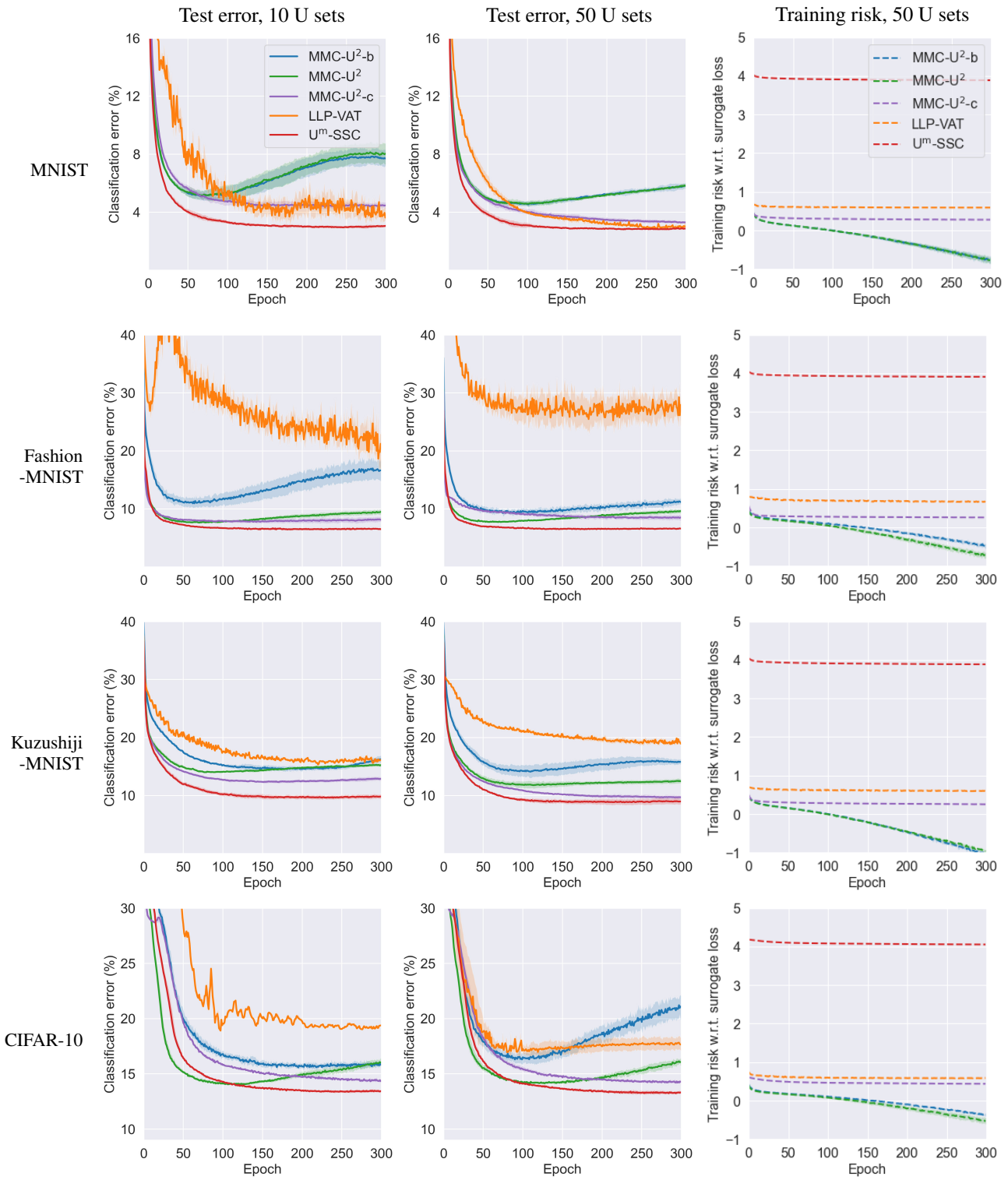

Figure 2. Experimental results of learning from 10 and 50 sets of U data. Solid curves are the test errors (in percentage) and dashed curves are the empirical training risks. Dark colors show the mean errors (risks) of 3 trials and light colors show the standard deviations.

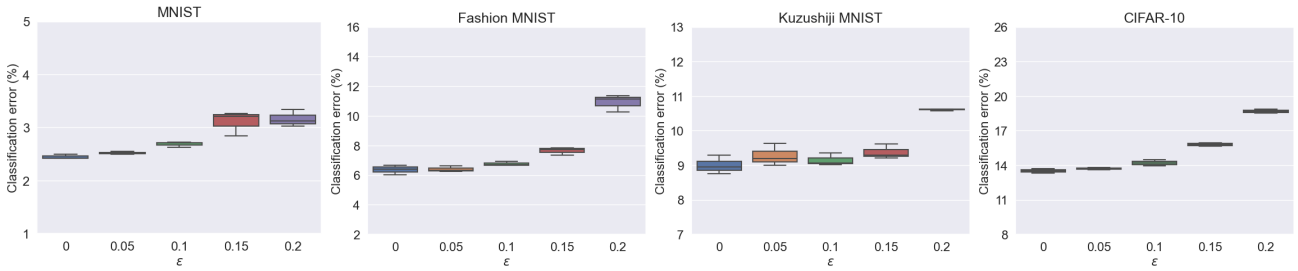

Figure 3. Box plot of the classification errors for the proposed U^m-SSC method tested on learning from 50 U sets with inaccurate class priors ($\epsilon = 0$ means *true*; larger ϵ, larger noise).

Table 3. Mean errors (standard deviations) over 3 trials in percentage for the proposed U^m-SSC method tested on different set sizes. The uniform set size n_j is shifted to $\tau \cdot n_j$ (smaller τ, larger shift). Random means uniformly sample a set size from range $[0, T]$.

Dataset	Sets	n_j	$\tau = 0.8$	$\tau = 0.6$	$\tau = 0.4$	$\tau = 0.2$	Random
MNIST	10	6000	2.83 (0.18)	2.91 (0.04)	3.2 (0.2)	3.19 (0.35)	2.66 (0.08)
	50	1200	2.46 (0.1)	2.58 (0.08)	2.76 (0.13)	2.97 (0.11)	3.0 (0.12)
Fashion-MNIST	10	6000	7.88 (0.21)	7.68 (0.36)	7.84(0.17)	7.89 (0.24)	7.04 (0.13)
	50	1200	8.29 (0.19)	8.91 (0.29)	7.61 (0.55)	8.8 (0.31)	8.62 (0.24)
Kuzushiji-MNIST	10	6000	8.98 (0.52)	9.83 (0.57)	9.43 (0.41)	10.03 (0.81)	8.38 (0.31)
	50	1200	9.35 (0.33)	9.53 (0.64)	9.89 (0.72)	11.08 (0.61)	10.34 (0.73)
CIFAR-10	10	5000	12.55 (0.61)	12.25 (0.8)	12.41 (0.35)	12.49 (0.98)	11.65 (0.44)
	50	1000	12.16 (0.23)	12.19 (0.75)	12.88 (0.37)	13.66 (0.54)	12.09 (0.42)

- MMC-U^2-b (Scott & Zhang, 2020): the classification risk based method in the *multiple mutual contamination* (MMC) framework, i.e., (8);
- MMC-U^2: the method that improves MMC-U^2-b with unbiased risk estimators $\widehat{R}_{\mathrm{U}^2}(f)$;
- MMC-U^2-c: the method that improves MMC-U^2 with non-negative risk correction $\widehat{R}_{\mathrm{U}^2\text{-c}}(f)$;
- LLP-VAT (Tsai & Lin, 2020): the empirical proportion risk based method, i.e., (6).

More details about the implementation of baselines can be found in Appendix B.3.

4.1. Comparison with State-of-the-art Methods

We first compare our proposed method with state-of-the-art methods for the U^m classification problem. The experimental results of learning from 10 and 50 U sets are reported in Figure 2 and a table of the final errors is in Appendix C.1.

We can see that the *classification risk* based methods, i.e., MMC-U^2-b, MMC-U^2, MMC-U^2-c, and our proposed U^m-SSC method generally outperform the *empirical proportion risk* based method, i.e., LLP-VAT, with lower classification error and more stability, which demonstrates the superiority of the *consistent* methods.

Within the classification risk based methods, our observations are as follows. First, the proposed U^m-SSC method outperforms others in most cases. We believe that the advantage comes from the surrogate set classification mechanism in U^m-SSC, which implies the *classifier-consistent* methods perform better than the *risk-consistent* methods. Second, compared to MMC-U^2-b, we can see our advantage becomes bigger when $\pi_{\mathcal{D}} \approx \frac{1}{2}$ is not the case, e.g., Fashion-MNIST and CIFAR-10. Moreover, the performance of the improved MMC-U^2 method (combination of unbiased risk estimators) is better than MMC-U^2-b (combination of balanced risk estimators) in all cases. These empirical findings corroborate our analysis that the balanced classification risk (7) can be biased in such cases. Third, we confirm that the training risks of MMC-U^2-b and MMC-U^2 go negative as training proceeds, which incurs overfitting. Other methods do not have this negative empirical training risk issue. And we can see that the improved MMC-U^2-c method effectively mitigates this overfitting but its performance is still inferior to our proposed method. These results are consistent with the observations in Lu et al. (2020). We also notice that the empirical training risks of the proposed U^m-SSC method are obviously higher than other baseline methods. This is due to the fact that the added transition layer rescales the range of model output. We provide a detailed analysis on this point in Appendix C.1. A notable effect is that a relatively small learning rate is more suitable for our method.

4.2. On the Variation of Set Size

In practice, the size of the U sets may vary from a large range depends on different tasks. However, as the set size varies, given the data generation process in (3), the marginal density of our training data $p_{\mathrm{tr}}(\boldsymbol{x})$ shifts from that of the test one,

Table 4. Mean test errors (standard deviations) over 3 trials in percentage for the U^m-SSC method tested with different set numbers.

Dataset	2 sets	100 sets	500 sets	1000 sets
MNIST	2.36 (0.27)	2.59 (0.11)	3.07 (0.14)	2.84 (0.16)
Fashion-MNIST	7.95 (0.85)	8.61 (0.43)	8.50 (0.44)	8.64 (0.26)
Kuzushiji-MNIST	9.68 (0.96)	10.20 (0.50)	11.64 (0.54)	10.68 (0.40)
CIFAR-10	13.21 (1.26)	13.30 (0.65)	12.98 (0.37)	13.16 (0.70)

which may cause severe *covariate shift* (Shimodaira, 2000; Zhang et al., 2020). To verify the robustness of our proposed method against covariate shift, we conducted experiments on the variation of set size. Recall that in other experiments, we use uniform set size, i.e., all sets contain n_{tr}/m U data. In this subsection, we investigate two kinds of set size shift:

- Randomly select $\lceil m/2 \rceil$ U sets and change their set sizes to $\tau \cdot n_{\mathrm{tr}}/m$ where $\tau \in [0, 1]$;
- Randomly sample each set size n_j from range $[0, n_{\mathrm{tr}}]$ such that $\sum_{j=1}^{m} n_j = n_{\mathrm{tr}}$.

As shown in Table 3, the proposed method is reasonably robust as τ moves towards 0 in the first shift setting. The slight performance degradation may come from the decreased total number of training samples n_{tr} as τ decreases. We also find that our method reaches the best performance in 3 out of 4 benchmark datasets in the second shift setting. Since it is a more natural way for generating set sizes, the robustness of the proposed method on varied set sizes can be verified.

4.3. On the Variation of Set Numbers

Another main factor that may affect the performance is the number of available U sets. As the U data can be easily collected from multiple sources, the learning algorithm is expected to be able to handle the variation of set numbers well. The experimental results of learning from 10 and 50 U sets have been shown in Section 4.1. In this subsection, we test the proposed U^m-SSC method on extremely small set numbers e.g., m = 2, and large set numbers, e.g., m=1000. The experimental results of learning form 2, 100, 500, and 1000 U sets are reported in Table 4.

From the results, we can see that the performance of the proposed method is reasonably well on different set numbers. In particular, a lower classification error can be observed for $m = 2$ across all 4 benchmark datasets. The better performance may come from the larger number of U data contained in a single set, i.e., $n_{\mathrm{tr}}/2$ in this case. Since our method uses class priors as the only weak supervision, an increasing number of the sampled data within each U set guarantees a better approximation of them. These experimental results demonstrate the effectiveness of the proposed method on the variation of set numbers. We note that it is a clear advantage over the LLP methods, whose performance drops significantly when the set number becomes small and set size becomes large, because label proportions converge to the same class prior in their setup, making the LLP problem computationally intractable.

4.4. Robustness against Inaccurate Class Priors

Hitherto, we have assumed that the values of class priors $\{\pi_j\}_{j=1}^{m}$ are accessible and accurately used in the construction of our method, which may not be true in practice. In order to simulate U^m classification in the wild, where we may suffer some errors from estimating the class priors, we design experiments that add noise to the true class priors. More specifically, we test the U^m-SSC method by replacing π_j with the noisy $\pi'_j = \pi_j + \gamma \cdot \epsilon$, where γ uniform randomly take values in $\{+1, -1\}$ and $\epsilon \in \{0, 0.05, 0.1, 0.15, 0.2\}$, so that the method would treat noisy π'_j as the true π_j during the whole learning process. The experimental setup is exactly same as before except the replacement of π_j. Note that we tailor the noisy π_j to $[0, 1]$ if it surpasses the range.

The results on learning from 50 U sets with inaccurate class priors are reported in Figure 3 and a table of the final test errors is in Appendix C.2, where $\epsilon = 0$ means true class priors. We can see that our method works reasonably well using noisy π'_j, though the classification error slightly increases for higher noise level ϵ which is as expected.

5. Conclusions

In this work, we focused on learning from multiple sets of U data and proposed a new method based on a surrogate set classification task. We bridged the original and surrogate class-posterior probabilities via a linear-fractional transformation, and then studied its properties. Based on them, we proposed the U^m-SSC algorithm and implemented it by adding a transition layer to the neural network. We also proved that the U^m-SSC method is classifier-consistent and established an estimation error bound for it. Extensive experiments demonstrated that the proposed method could successfully train binary classifiers from multiple U sets, and it compared favorably with state-of-the-art methods.

Acknowledgments

The authors would like to thank Tianyi Zhang, Wenkai Xu, Nontawat Charoenphakdee, and Yuting Tang for helpful discussions. NL was supported by the MEXT scholarship No. 171536 and MSRA D-CORE Program. GN and MS were supported by JST AIP Acceleration Research Grant Number JPMJCR20U3, Japan. MS was also supported by the Institute for AI and Beyond, UTokyo.

References

Allen, D. M. Mean square error of prediction as a criterion for selecting variables. *Technometrics*, 13(3):469–475, 1971.

Ardehaly, E. M. and Culotta, A. Mining the demographics of political sentiment from twitter using learning from label proportions. In *ICDM*, 2017.

Bao, H., Niu, G., and Sugiyama, M. Classification from pairwise similarity and unlabeled data. In *ICML*, 2018.

Bartlett, P. L., Jordan, M. I., and McAuliffe, J. D. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

Ben-David, S., Eiron, N., and Long, P. On the difficulty of approximately maximizing agreements. *Journal of Computer and System Sciences*, 66(3):496–514, 2003.

Bertsekas, D. P. Nonlinear programming. *Journal of the Operational Research Society*, 48(3):334–334, 1997.

Brodersen, K. H., Ong, C. S., Stephan, K. E., and Buhmann, J. M. The balanced accuracy and its posterior distribution. In *ICPR*, 2010.

Clanuwat, T., Bober-Irizar, M., Kitamoto, A., Lamb, A., Yamamoto, K., and Ha, D. Deep learning for classical japanese literature. *arXiv preprint arXiv:1812.01718*, 2018.

Cour, T., Sapp, B., and Taskar, B. Learning from partial labels. *Journal of Machine Learning Research*, 12:1501–1536, 2011.

Croft, J. B., Wheaton, A. G., Liu, Y., Xu, F., Lu, H., Matthews, K. A., Cunningham, T. J., Wang, Y., and Holt, J. B. Urban-rural county and state differences in chronic obstructive pulmonary disease—united states, 2015. *Morbidity and Mortality Weekly Report*, 67(7):205, 2018.

De Boer, P.-T., Kroese, D. P., Mannor, S., and Rubinstein, R. Y. A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67, 2005.

Edmonds, J. Maximum matching and a polyhedron with 0, 1-vertices. *Journal of research of the National Bureau of Standards B*, 69(125-130):55–56, 1965.

Edmonds, J. and Karp, R. M. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM)*, 19(2):248–264, 1972.

Fang, T., Lu, N., Niu, G., and Sugiyama, M. Rethinking importance weighting for deep learning under distribution shift. In *NeurIPS*, 2020.

Feng, L., Lv, J., Han, B., Xu, M., Niu, G., Geng, X., An, B., and Sugiyama, M. Provably consistent partial-label learning. In *NeurIPS*, 2020.

Feng, L., Shu, S., Lu, N., Han, B., Xu, M., Niu, G., An, B., and Sugiyama, M. Pointwise binary classification with pairwise confidence comparisons. In *ICML*, 2021.

Gomes, R., Krause, A., and Perona, P. Discriminative clustering by regularized information maximization. In *NeurIPS*, 2010.

Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016.

Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I., and Sugiyama, M. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NeurIPS*, 2018.

Han, B., Niu, G., Yu, X., Yao, Q., Xu, M., Tsang, I., and Sugiyama, M. Sigua: Forgetting may make learning with noisy labels more robust. In *ICML*, 2020.

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

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Ishida, T., Niu, G., Hu, W., and Sugiyama, M. Learning from complementary labels. In *NeurIPS*, 2017.

Ishida, T., Niu, G., Menon, A. K., and Sugiyama, M. Complementary-label learning for arbitrary losses and models. In *ICML*, 2019.

Jain, S., White, M., and Radivojac, P. Estimating the class prior and posterior from noisy positives and unlabeled data. In *NeurIPS*, pp. 2693–2701, 2016.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kingma, D. P. and Ba, J. L. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Krizhevsky, A. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Liu, T. and Tao, D. Classification with noisy labels by importance reweighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(3):447–461, 2016.

Lu, N., Niu, G., Menon, A. K., and Sugiyama, M. On the minimal supervision for training any binary classifier from only unlabeled data. In *ICLR*, 2019.

Lu, N., Zhang, T., Niu, G., and Sugiyama, M. Mitigating overfitting in supervised classification from two unlabeled datasets: A consistent risk correction approach. In *AISTATS*, 2020.

Lv, J., Xu, M., Feng, L., Niu, G., Geng, X., and Sugiyama, M. Progressive identification of true labels for partial-label learning. In *International Conference on Machine Learning*, 2020.

Maurer, A. A vector-contraction inequality for rademacher complexities. In *ALT*, pp. 3–17. Springer, 2016.

McDiarmid, C. On the method of bounded differences. *Surveys in combinatorics*, 141(1):148–188, 1989.

Menon, A. K., van Rooyen, B., Ong, C. S., and Williamson, R. C. Learning from corrupted binary labels via class-probability estimation. In *ICML*, 2015.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. *Foundations of Machine Learning*. MIT Press, 2012.

Nair, V. and Hinton, G. E. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

Natarajan, N., Dhillon, I. S., Ravikumar, P., and Tewari, A. Learning with noisy labels. In *NeurIPS*, 2013.

Newman, B. Integrity and presidential approval, 1980–2000. *Public Opinion Quarterly*, 67(3):335–367, 2003.

Patrini, G., Rozza, A., Menon, A. K., Nock, R., and Qu, L. Making deep neural networks robust to label noise: A loss correction approach. In *CVPR*, 2017.

Quadrianto, N., Smola, A. J., Caetano, T. S., and Le, Q. V. Estimating labels from label proportions. *Journal of Machine Learning Research*, 10:2349–2374, 2009.

Scott, C. and Zhang, J. Learning from label proportions: A mutual contamination framework. In *NeurIPS*, 2020.

Scott, C., Blanchard, G., and Handy, G. Classification with asymmetric label noise: Consistency and maximal denoising. In *COLT*, 2013.

Shalev-Shwartz, S. and Ben-David, S. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

Shimodaira, H. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958, 2014.

Tokunaga, H., Iwana, B. K., Teramoto, Y., Yoshizawa, A., and Bise, R. Negative pseudo labeling using class proportion for semantic segmentation in pathology. In *ECCV*, 2020.

Tsai, K.-H. and Lin, H.-T. Learning from label proportions with consistency regularization. In *ACML*, 2020.

Van Rooyen, B. and Williamson, R. C. A theory of learning with corrupted labels. *Journal of Machine Learning Research*, 18(1):8501–8550, 2018.

Vapnik, V. N. *Statistical Learning Theory*. John Wiley & Sons, 1998.

Xia, X., Liu, T., Han, B., Wang, N., Gong, M., Liu, H., Niu, G., Tao, D., and Sugiyama, M. Part-dependent label noise: Towards instance-dependent label noise. In *NeurIPS*, 2020.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Xu, L., Neufeld, J., Larson, B., and Schuurmans, D. Maximum margin clustering. In *NeurIPS*, 2004.

Xu, L., Honda, J., Niu, G., and Sugiyama, M. Uncoupled regression from pairwise comparison data. In *NeurIPS*, 2019.

Yao, Y., Liu, T., Han, B., Gong, M., Deng, J., Niu, G., and Sugiyama, M. Dual t: Reducing estimation error for transition matrix in label-noise learning. In *NeurIPS*, 2020.

Yu, F. X., Choromanski, K., Kumar, S., Jebara, T., and Chang, S.-F. On learning from label proportions. *arXiv preprint arXiv:1402.5902*, 2014.

Zhang, T., Yamane, I., Lu, N., and Sugiyama, M. A one-step approach to covariate shift adaptation. In *ACML*, 2020.

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

Supplementary Material

A. Proofs

In this appendix, we prove all theorems.

A.1. Proof of Theorem 1

On one hand, $\forall j \in [1, \ldots, m]$ we have

$$\begin{aligned}\bar{\eta}_j(\boldsymbol{x}) &= \frac{p(\boldsymbol{x}, \bar{y} = j)}{\bar{p}(\boldsymbol{x})} \\ &= \frac{p(\boldsymbol{x} \mid \bar{y} = j) \cdot p(\bar{y} = j)}{\bar{p}(\boldsymbol{x})} \\ &= \frac{\rho_j \cdot [\pi_j \cdot p_p(\boldsymbol{x}) + (1 - \pi_j) \cdot p_n(\boldsymbol{x})]}{\sum_{j=1}^{m} \rho_j \cdot [\pi_j \cdot p_p(\boldsymbol{x}) + (1 - \pi_j) \cdot p_n(\boldsymbol{x})]}. \end{aligned} \tag{16}$$

In the third equality, we substitute $p(\boldsymbol{x} \mid \bar{y} = j)$ with p_{tr} that is defined in (3). On the other hand, by Bayes' rule we have

$$p_p(\boldsymbol{x}) = p(\boldsymbol{x} \mid y = +1) = \frac{p(y = +1 \mid \boldsymbol{x}) \cdot p(\boldsymbol{x})}{p(y = +1)} = \frac{\eta(\boldsymbol{x}) \cdot p(\boldsymbol{x})}{\pi_{\mathcal{D}}}, \tag{17}$$

$$p_n(\boldsymbol{x}) = p(\boldsymbol{x} \mid y = -1) = \frac{p(y = -1 \mid \boldsymbol{x}) \cdot p(\boldsymbol{x})}{p(y = -1)} = \frac{(1 - \eta(\boldsymbol{x})) \cdot p(\boldsymbol{x})}{1 - \pi_{\mathcal{D}}}. \tag{18}$$

Then, we plug (17) and (18) into (16) and obtain

$$\begin{aligned}\bar{\eta}_j(\boldsymbol{x}) &= \frac{\rho_j \cdot [\pi_j \eta(\boldsymbol{x}) \cdot (1 - \pi_{\mathcal{D}}) + (1 - \pi_j) \cdot (1 - \eta(\boldsymbol{x})) \cdot \pi_{\mathcal{D}}]}{\sum_{j=1}^{m} \rho_j \cdot [\pi_j \eta(\boldsymbol{x}) \cdot (1 - \pi_{\mathcal{D}}) + (1 - \pi_j) \cdot (1 - \eta(\boldsymbol{x})) \cdot \pi_{\mathcal{D}}]} \\ &= \frac{\rho_j \cdot (\pi_j - \pi_{\mathcal{D}}) \cdot \eta(\boldsymbol{x}) + \rho_j \cdot (1 - \pi_j) \cdot \pi_{\mathcal{D}}}{\sum_{j=1}^{m} \rho_j \cdot (\pi_j - \pi_{\mathcal{D}}) \cdot \eta(\boldsymbol{x}) + \sum_{j=1}^{m} \rho_j \cdot (1 - \pi_j) \cdot \pi_{\mathcal{D}}}. \end{aligned}$$

By setting the coefficients a_j, b_j, c, d accordingly we conclude the proof. □

A.2. Proof of Lemma 2

We proceed the proof by firstly showing that the denominator of each function $T_j(t)$, $j = 1, \ldots, m$, is strictly greater than zero for all $t \in [0, 1]$, and then showing that $\boldsymbol{T}(t_1) = \boldsymbol{T}(t_2)$ if and only if $t_1 = t_2$.

For all $j = 1, \ldots, m$, the denominators of $T_j(t)$ are the same, i.e., $c \cdot t + d$, where $c = \sum_{j=1}^{m} \rho_j(\pi_j - \pi_{\mathcal{D}})$ and $d = \sum_{j=1}^{m} \rho_j \pi_{\mathcal{D}}(1 - \pi_j)$. We know that d is positive because $\rho_j > 0$, $\pi_{\mathcal{D}} > 0$, and there exists $j \in 1, \ldots, m$ such that $\pi_j < 1$. Given that $t \in [0, 1]$, we discuss the sign of c as follows:

1. if $c \geq 0$, the minimum value of $c \cdot t + d$ is $c \cdot 0 + d = d > 0$;
2. if $c < 0$, the minimum value of $c \cdot t + d$ is $c \cdot 1 + d = \sum_{j=1}^{m} \rho_j(\pi_j - \pi_{\mathcal{D}}) + \sum_{j=1}^{m} \rho_j \pi_{\mathcal{D}}(1 - \pi_j) = \sum_{j=1}^{m} \rho_j \pi_j (1 - \pi_{\mathcal{D}}) > 0$, where the last inequality is due to the existence of $j \in 1, \ldots, m$ such that $\pi_j > 0$.

Hitherto, we have shown that the denominator $c \cdot t + d > 0$. Next, we prove the one-to-one mapping property by contradiction. Assume that there exist $t_1, t_2 \in [0, 1]$ such that $t_1 \neq t_2$ but $\boldsymbol{T}(t_1) = \boldsymbol{T}(t_2)$, which indicates that $T_j(t_1) = T_j(t_2), \forall j = 1, \ldots, m$. For all j, we have

$$\begin{aligned}T_j(t_1) - T_j(t_2) &= \frac{a_j \cdot t_1 + b_j}{c \cdot t_1 + d} - \frac{a_j \cdot t_2 + b_j}{c \cdot t_2 + d} \\ &= \frac{(a_j \cdot t_1 + b_j)((c \cdot t_2 + d)) - (a_j \cdot t_2 + b_j)((c \cdot t_1 + d))}{(c \cdot t_1 + d)(c \cdot t_2 + d)} \\ &= \frac{(t_1 - t_2)(a_j \cdot d - b_j \cdot c)}{(c \cdot t_1 + d)(c \cdot t_2 + d)} \\ &= 0, \end{aligned} \tag{19}$$

where $a_j = \rho_j \cdot (\pi_j - \pi_{\mathcal{D}})$ and $b_j = \rho_j \cdot (1 - \pi_j) \cdot \pi_{\mathcal{D}}$. As shown previously, the denominator of (19) is non-zero for all j. Next we show that there exists $j \in 1, \ldots, m$ such that $a_j \cdot d - b_j c \neq 0$. Since c and d are constants and irrelevant to i, we have

$$\begin{aligned} a_j \cdot d - b_j \cdot c &= (\rho_j \cdot (\pi_j - \pi_{\mathcal{D}})) \cdot d - (\rho_j \cdot (1 - \pi_j) \cdot \pi_{\mathcal{D}}) \cdot c \\ &= \rho_j \cdot (\pi_j \cdot d - \pi_{\mathcal{D}} \cdot d - c + \pi_j \cdot c) \\ &= \rho_j \cdot (\pi_j \cdot (c + d) - \pi_{\mathcal{D}} \cdot d - c). \end{aligned}$$

This equation equals to zero if and only if $\pi_j = \frac{c + \pi_{\mathcal{D}} \cdot d}{c + d}$. According to our assumption that at least two of the U sets are different, $\exists j' \in 1, \ldots, m$ such that $\pi_{j'} \neq \frac{c + \pi_{\mathcal{D}} \cdot d}{c + d}$. For such j', $T_{j'}(t_1) = T_{j'}(t_2)$ if and only if $t_1 = t_2$, which leads to a contradiction since $t_1 \neq t_2$. So we conclude the proof that $\boldsymbol{T}(t_1) = \boldsymbol{T}(t_2)$ if and only if $t_1 = t_2$. □

A.3. Proof of Lemma 3

We provide a proof of the cross-entropy loss and mean squared error, which are commonly used losses because of their numerical stability and good convergence rate (De Boer et al., 2005; Allen, 1971).

Cross-entropy loss Since the cross-entropy loss is non-negative by its definition, minimizing $R_{\mathrm{surr}}(\boldsymbol{g})$ can be obtained by minimizing the conditional risk $\mathbb{E}_{p(\bar{y} \mid \boldsymbol{x})}[\ell(\boldsymbol{g}(\boldsymbol{x}), \bar{y}) \mid \boldsymbol{x}]$ for every $\boldsymbol{x} \in \mathcal{X}$. So we are now optimizing

$$\phi(\boldsymbol{g}) = -\sum_{j=1}^{m} p(\bar{y} = j \mid \boldsymbol{x}) \cdot \log(g_j(\boldsymbol{x})), \quad \text{s.t.} \sum_{j=1}^{m} g_j(\boldsymbol{x}) = 1.$$

By using the Lagrange multiplier method (Bertsekas, 1997), we have

$$\mathcal{L} = -\sum_{j=1}^{m} p(\bar{y} = j \mid \boldsymbol{x}) \cdot \log(g_j(\boldsymbol{x})) - \lambda \cdot (\sum_{j=1}^{m} g_j(\boldsymbol{x}) - 1).$$

The derivative of $\mathcal{L}$ with respect to $\boldsymbol{g}$ is

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{g}} = [-\frac{p(\bar{y} = 1 \mid \boldsymbol{x})}{g_1(\boldsymbol{x})} - \lambda, \cdots, -\frac{p(\bar{y} = m \mid \boldsymbol{x})}{g_m(\boldsymbol{x})} - \lambda]^{\top}.$$

By setting this derivative to 0 we obtain

$$g_j(\boldsymbol{x}) = -\frac{1}{\lambda} \cdot p(\bar{y} = j \mid \boldsymbol{x}), \quad \forall j = 1, \ldots, m \text{ and } \forall \boldsymbol{x} \in \mathcal{X}.$$

Since $\boldsymbol{g} \in \Delta^{m-1}$ is the m-dimensional simplex, we have $\sum_{j=1}^{m} g_j^{\star}(\boldsymbol{x}) = 1$ and $\sum_{j=1}^{m} p(\bar{y} = j \mid \boldsymbol{x}) = 1$. Then

$$\sum_{j=1}^{m} g_j^{\star}(\boldsymbol{x}) = -\frac{1}{\lambda} \cdot \sum_{j=1}^{m} p(\bar{y} = j \mid \boldsymbol{x}) = 1.$$

Therefore we obtain $\lambda = -1$ and $g_j^{\star}(\boldsymbol{x}) = p(\bar{y} = j \mid \boldsymbol{x}) = \bar{\eta}_j(\boldsymbol{x}), \forall j = 1, \ldots, m$ and $\forall \boldsymbol{x} \in \mathcal{X}$, which is equivalent to $\boldsymbol{g}^{\star} = \bar{\boldsymbol{\eta}}$. Note that when $m = 2$, the softmax is reduce to sigmoid function and the cross-entropy is reduced to logistic loss $\ell_{\log}(z) = \ln(1 + \exp(-z))$.

Mean squared error Similarly to the cross-entropy loss, we transform the risk minimization problem to the following constrained optimization problem

$$\phi(\boldsymbol{g}) = \sum_{j=1}^{m} (p(\bar{y} = j \mid \boldsymbol{x}) - g_j(\boldsymbol{x}))^2, \quad \text{s.t.} \sum_{j=1}^{m} g_j(\boldsymbol{x}) = 1.$$

By using the Lagrange multiplier method, we obtain

$$\mathcal{L} = \sum_{j=1}^{m} (p(\bar{y} = j \mid \boldsymbol{x}) - g_j(\boldsymbol{x}))^2 - \lambda \cdot (\sum_{j=1}^{m} g_j(\boldsymbol{x}) - 1).$$

The derivative of $\mathcal{L}$ with respect to $\boldsymbol{g}$ is

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{g}} = [2g_1(\boldsymbol{x}) - 2p(\bar{y} = 1 \mid \boldsymbol{x}) - \lambda, \cdots, 2g_m(\boldsymbol{x}) - 2p(\bar{y} = m \mid \boldsymbol{x}) - \lambda]^\top.$$

By setting this derivative to 0 we obtain

$$g_j(\boldsymbol{x}) = p(\bar{y} = j \mid \boldsymbol{x}) + \frac{\lambda}{2}.$$

Since $\sum_{j=1}^m g_j^\star(\boldsymbol{x}) = 1$ and $\sum_{j=1}^m p(\bar{y} = j \mid \boldsymbol{x}) = 1$, we have

$$\sum_{j=1}^m g_j^\star(\boldsymbol{x}) = \sum_{j=1}^m p(\bar{y} = j \mid \boldsymbol{x}) + \frac{\lambda \cdot m}{2},$$
$$\frac{\lambda \cdot m}{2} = 0.$$

Since $m \geq 2$, we can obtain $\lambda = 0$. Consequently, $g_j^\star(\boldsymbol{x}) = p(\bar{y} = j \mid \boldsymbol{x}) = \bar{\eta}_j(\boldsymbol{x})$, which leads to $\boldsymbol{g}^\star = \bar{\boldsymbol{\eta}}$. We conclude the proof. □

A.4. Proof of Theorem 4

According to Lemma 3, when a cross-entropy loss or mean squared error is used for ℓ, the mapping $\boldsymbol{g}^\star(\boldsymbol{x}) = \bar{\boldsymbol{\eta}}(\boldsymbol{x})$ is the unique minimizer of $R_{\mathrm{surr}}(\boldsymbol{g}; \ell)$. Let $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{T}(f(\boldsymbol{x}))$, since $\boldsymbol{g}^\star \in \mathcal{G}$, $R_{\mathrm{surr}}(\boldsymbol{g}(\boldsymbol{x})) = \mathbb{E}_{(\boldsymbol{x},\bar{y})\sim\bar{\mathcal{D}}}[\ell(\boldsymbol{g}(\boldsymbol{x}), \bar{y})]$ achieves its minimum if and only if $\boldsymbol{g}(\boldsymbol{x}) = \bar{\boldsymbol{\eta}}(\boldsymbol{x}) = \boldsymbol{g}^\star(\boldsymbol{x})$. Combining this result with Theorem 1 and Lemma 2, we then obtain that $\boldsymbol{g}(\boldsymbol{x}) = \bar{\boldsymbol{\eta}}(\boldsymbol{x})$ if and only if $f(\boldsymbol{x}) = \eta(\boldsymbol{x})$. Since

$$\begin{aligned} R_{\mathrm{surr}}(f) &= \mathbb{E}_{(\boldsymbol{x},\bar{y})\sim\bar{\mathcal{D}}}[\ell(\boldsymbol{T}(f(\boldsymbol{x})), \bar{y})] \\ &= \mathbb{E}_{(\boldsymbol{x},\bar{y})\sim\bar{\mathcal{D}}}[\ell(\boldsymbol{g}(\boldsymbol{x}), \bar{y})] = R_{\mathrm{surr}}(\boldsymbol{g}), \end{aligned}$$

$f_{\mathrm{surr}}^\star$ is induced by $\boldsymbol{g}^\star = \operatorname{argmin}_{\boldsymbol{g}} R_{\mathrm{surr}}(\boldsymbol{g})$. So we have $f_{\mathrm{surr}}^\star(\boldsymbol{x}) = \operatorname{argmin}_f R_{\mathrm{surr}}(f) = \eta(\boldsymbol{x})$.

On the other hand, when ℓ_{b} is a cross-entropy loss, i.e., the logistic loss in the binary case, or mean squared error, the mapping $f^\star$ is the unique minimizer of $R(f; \ell_{\mathrm{b}})$. We skip the proof since it is similar to the proof of Lemma 3. So we obtain that $f^\star(\boldsymbol{x}) = \eta(\boldsymbol{x}) = f_{\mathrm{surr}}^\star(\boldsymbol{x})$, which concludes the proof. □

A.5. Proof of Lemma 5

$\forall j \in 1, \ldots, m$, by taking derivative of $T_j(t)$ with respect to t, we obtain

$$\left|\frac{\partial T_j}{\partial t}\right| = \frac{|a_j d - b_j c|}{(c \cdot t + d)^2}, \tag{20}$$

where

$$a_j = \rho_j(\pi_j - \pi_{\mathcal{D}}), \quad b_j = \rho_j \pi_{\mathcal{D}}(1 - \pi_j), \quad c = \sum_{j=1}^m \rho_j(\pi_j - \pi_{\mathcal{D}}), \quad d = \sum_{j=1}^m \rho_j \pi_{\mathcal{D}}(1 - \pi_j).$$

Since for all class priors we have $0 \leq \pi_j \leq 1$, $0 < \pi_D < 1$, $0 < \rho_j < 1$, $\sum_{j=1}^m \rho_j = 1$, and $\exists j, j' \in \{1, \ldots, m\}$ such that $j \neq j'$ and $\pi_j \neq \pi_{j'}$, obviously we can obtain

$$-1 \leq a_j \leq 1, \quad 0 \leq b_j \leq 1, \quad -1 \leq c \leq 1, \quad \text{and } 0 < d \leq 1.$$

Therefore, the numerator of (20) satisfies

$$|a_j d - b_j c| \leq 2. \tag{21}$$

On the other hand, since $d > 0$ and $0 \leq t \leq 1$, by substituting $t = 0$ and $t = 1$ respectively, we can obtain

$$c \cdot t + d \geq c + d = \sum_{j=1}^m \rho_j \pi_j (1 - \pi_{\mathcal{D}}) > 0, \quad \text{if } c < 0;$$
$$c \cdot t + d \geq d > 0, \quad \text{if } c \geq 0.$$

Next we lower bound this term by $c \cdot t + d \geq \min(c+d, d) > 0$. As a result, the denominator of (20) satisfies

$$\begin{aligned}(c \cdot t + d)^2 &\geq (\min(c+d, d))^2 \\ &= \left(\min \left(\sum_{j=1}^{m} \rho_j \pi_j (1 - \pi_{\mathcal{D}}), \sum_{j=1}^{m} \rho_j \pi_{\mathcal{D}} (1 - \pi_j) \right) \right)^2 \\ &= \alpha^2.\end{aligned} \tag{22}$$

Then, by combing (21) and (22), we have

$$\left| \frac{\partial T_j}{\partial t} \right| \leq \frac{2}{\alpha^2}.$$

This bound illustrates that $T_j(f(\boldsymbol{x}))$ is Lipschitz-continuous with respect to $f(\boldsymbol{x})$ with a Lipschitz constant $2/\alpha^2$ and we complete the proof. □

A.6. Proof of Theorem 6

We first introduce the following lemmas which are useful to derive the estimation error bound.

Lemma 7 (Uniform deviation bound)**.** *Let $\boldsymbol{g} \in \mathcal{G}$, where $\mathcal{G} = \{\boldsymbol{x} \mapsto \boldsymbol{T}(f(\boldsymbol{x})) \mid f \in \mathcal{F}\}$ is a class of measurable functions, $\mathcal{X}_{\mathrm{tr}} = \{(\boldsymbol{x}_i, \bar{y}_i)\}_{i=1}^{n_{\mathrm{tr}}} \overset{\text{i.i.d.}}{\sim} \bar{\mathcal{D}}$ be a fixed sample of size n_{tr} i.i.d. drawn from $\bar{\mathcal{D}}$, and $\{\sigma_1, \ldots, \sigma_{n_{\mathrm{tr}}}\}$ be the Rademacher variables, i.e., independent uniform random variables taking values in $\{-1, 1\}$. Let $\mathfrak{R}_{n_{\mathrm{tr}}}(\ell \circ \mathcal{G})$ be the Rademacher complexity of $\ell \circ \mathcal{G}$ which is defined as*

$$\mathfrak{R}_{n_{\mathrm{tr}}}(\ell \circ \mathcal{G}) = \mathbb{E}\left[\sup_{\boldsymbol{g} \in \mathcal{G}} \frac{1}{n_{\mathrm{tr}}} \sum_{i=1}^{n_{\mathrm{tr}}} \sigma_i \ell(\boldsymbol{g}(\boldsymbol{x}_i), \bar{y}_i) \right].$$

Under the assumptions of Theorem 6, $\ell(\boldsymbol{g}(\boldsymbol{x}), \bar{y})$ is upper-bounded by M_ℓ. Then, for any $\delta > 0$, we have with probability at least $1 - \delta$,

$$\sup_{\boldsymbol{g} \in \mathcal{G}} |\widehat{R}_{\mathrm{surr}}(\boldsymbol{g}) - R_{\mathrm{surr}}(\boldsymbol{g})| \leq 2\mathfrak{R}_{n_{\mathrm{tr}}}(\ell \circ \mathcal{G}) + M_\ell \sqrt{\frac{\ln(2/\delta)}{2n_{\mathrm{tr}}}}.$$

Proof. We consider the one-side uniform deviation $\sup_{\boldsymbol{g} \in \mathcal{G}} \widehat{R}_{\mathrm{surr}}(\boldsymbol{g}) - R_{\mathrm{surr}}(\boldsymbol{g})$. Suppose that a sample $(\boldsymbol{x}_i, \bar{y}_i)$ is replaced by another arbitrary sample $(\boldsymbol{x}_j, \bar{y}_j)$, the change of $\sup_{\boldsymbol{g} \in \mathcal{G}} \widehat{R}_{\mathrm{surr}}(\boldsymbol{g}) - R_{\mathrm{surr}}(\boldsymbol{g})$ is no more than M_ℓ / n_{tr}, since the loss $\ell(\cdot)$ is bounded by M_ℓ. By applying the McDiarmid's inequality (McDiarmid, 1989), for all $\epsilon' > 0$ we have

$$\Pr\{\sup_{\boldsymbol{g} \in \mathcal{G}} \widehat{R}_{\mathrm{surr}}(\boldsymbol{g}) - R_{\mathrm{surr}}(\boldsymbol{g}) - \mathbb{E}[\sup_{\boldsymbol{g} \in \mathcal{G}} \widehat{R}_{\mathrm{surr}}(\boldsymbol{g}) - R_{\mathrm{surr}}(\boldsymbol{g})] \geq \epsilon'\} \leq \exp\left(\frac{-2n_{\mathrm{tr}}\epsilon'^2}{M_\ell^2} \right).$$

Equivalently, for any $\delta > 0$, with probability at least $1 - \delta/2$,

$$\sup_{\boldsymbol{g} \in \mathcal{G}} \widehat{R}_{\mathrm{surr}}(\boldsymbol{g}) - R_{\mathrm{surr}}(\boldsymbol{g}) \leq \mathbb{E}\left[\sup_{\boldsymbol{g} \in \mathcal{G}} \widehat{R}_{\mathrm{surr}}(\boldsymbol{g}) - R_{\mathrm{surr}}(\boldsymbol{g}) \right] + M_\ell \sqrt{\frac{\ln(2/\delta)}{2n_{\mathrm{tr}}}}.$$

By *symmetrization* (Vapnik, 1998), it is a routine work to show that

$$\mathbb{E}\left[\sup_{\boldsymbol{g} \in \mathcal{G}} \widehat{R}_{\mathrm{surr}}(\boldsymbol{g}) - R_{\mathrm{surr}}(\boldsymbol{g}) \right] \leq 2\mathfrak{R}_{n_{\mathrm{tr}}}(\ell \circ \mathcal{G}).$$

The other side uniform deviation $\sup_{\boldsymbol{g} \in \mathcal{G}} R_{\mathrm{surr}}(\boldsymbol{g}) - \widehat{R}_{\mathrm{surr}}(\boldsymbol{g})$ can be bounded similarly. By combining the two sides' inequalities, we complete the proof. □

Lemma 8. *Let $f \in \mathcal{F}$, where $\mathcal{F} = \{f : \mathcal{X} \to \mathbb{R}\}$ is a class of measurable functions, $\{\boldsymbol{x}_i\}_{i=1}^{n_{\mathrm{tr}}} \overset{\text{i.i.d.}}{\sim} p_{\mathrm{tr}}(\boldsymbol{x})$ be a fixed sample of size n_{tr} i.i.d. drawn from the marginal density $p_{\mathrm{tr}}(\boldsymbol{x})$, and $\{\sigma_1, \ldots, \sigma_{n_{\mathrm{tr}}}\}$ be the Rademacher variables. Let $\mathfrak{R}_{n_{\mathrm{tr}}}(\mathcal{F})$ be the Rademacher complexity of $\mathcal{F}$ which is defined as*

$$\mathfrak{R}_{n_{\mathrm{tr}}}(\mathcal{F}) = \mathbb{E}\left[\sup_{f \in \mathcal{F}} \frac{1}{n_{\mathrm{tr}}} \sum_{i=1}^{n_{\mathrm{tr}}} \sigma_i f(\boldsymbol{x}_i)\right].$$

Then we have

$$\mathfrak{R}_{n_{\mathrm{tr}}}(\ell \circ \mathcal{G}) \leq \frac{2\sqrt{2}m\mathcal{L}_\ell}{\alpha^2} \mathfrak{R}_{n_{\mathrm{tr}}}(\mathcal{F}).$$

Proof. In what follows, we upper-bound $\mathfrak{R}_{n_{\mathrm{tr}}}(\ell \circ \mathcal{G})$. Since $\ell(\boldsymbol{g}(\boldsymbol{x}), \bar{y})$ is $\mathcal{L}_\ell$-Lipschitz continuous w.r.t $\boldsymbol{g}$, according to the Rademacher vector contraction inequality (Maurer, 2016), we have

$$\begin{aligned}
\mathfrak{R}_{n_{\mathrm{tr}}}(\ell \circ \mathcal{G}) &= \mathbb{E}\left[\sup_{\boldsymbol{g} \in \mathcal{G}} \frac{1}{n_{\mathrm{tr}}} \sum_{i=1}^{n_{\mathrm{tr}}} \sigma_i \ell(\boldsymbol{g}(\boldsymbol{x}_i), \bar{y}_i)\right] \\
&\leq \frac{\sqrt{2}\mathcal{L}_\ell}{n_{\mathrm{tr}}} \cdot \mathbb{E}\left[\sup_{\boldsymbol{g} \in \mathcal{G}} \sum_{i=1}^{n_{\mathrm{tr}}} \sum_{j=1}^{m} \sigma_{ij} g_j(\boldsymbol{x}_i)\right] \\
&\leq \frac{\sqrt{2}\mathcal{L}_\ell}{n_{\mathrm{tr}}} \cdot \sum_{j=1}^{m} \mathbb{E}\left[\sup_{\boldsymbol{g} \in \mathcal{G}} \sum_{i=1}^{n_{\mathrm{tr}}} \sigma_{ij} g_j(\boldsymbol{x}_i)\right], \qquad (23)
\end{aligned}$$

where $g_j(\boldsymbol{x}_i)$ is the j-th component of $\boldsymbol{g}(\boldsymbol{x}_i)$, and σ_{ij} are an $n_{\mathrm{tr}} \times m$ matrix of independent Rademacher variables. As shown in Lemma 5, $g_j(\boldsymbol{x}) = T_j(f(\boldsymbol{x}))$ and $T_j(f)$ is Lipschitz continuous w.r.t f with a Lipschitz constant $2/\alpha^2$. Then we apply the Talagrand's contraction lemma (Shalev-Shwartz & Ben-David, 2014) and obtain

$$\begin{aligned}
\sum_{j=1}^{m} \mathbb{E}\left[\sup_{\boldsymbol{g} \in \mathcal{G}} \sum_{i=1}^{n_{\mathrm{tr}}} \sigma_{ij} g_j(\boldsymbol{x}_i)\right] &= \sum_{j=1}^{m} \mathbb{E}\left[\sup_{f \in \mathcal{F}} \sum_{i=1}^{n_{\mathrm{tr}}} \sigma_{ij} T_j(f(\boldsymbol{x}_i))\right] \\
&\leq \frac{2}{\alpha^2} \sum_{j=1}^{m} \mathbb{E}\left[\sup_{f \in \mathcal{F}} \sum_{i=1}^{n_{\mathrm{tr}}} \sigma_{ij} f(\boldsymbol{x}_i)\right] \\
&= \frac{2mn_{\mathrm{tr}}}{\alpha^2} \mathfrak{R}_{n_{\mathrm{tr}}}(\mathcal{F}).
\end{aligned}$$

By substituting it into (23), we complete the proof. □

Based on Lemma 7 and Lemma 8, the estimation error bound is proven through

$$\begin{aligned}
&R_{\mathrm{surr}}(\hat{f}_{\mathrm{surr}}) - R_{\mathrm{surr}}(f^\star_{\mathrm{surr}}) \\
&= \left(\widehat{R}_{\mathrm{surr}}(\hat{f}_{\mathrm{surr}}) - \widehat{R}_{\mathrm{surr}}(f^\star_{\mathrm{surr}})\right) + \left(R_{\mathrm{surr}}(\hat{f}_{\mathrm{surr}}) - \widehat{R}_{\mathrm{surr}}(\hat{f}_{\mathrm{surr}})\right) + \left(\widehat{R}_{\mathrm{surr}}(f^\star_{\mathrm{surr}}) - R_{\mathrm{surr}}(f^\star_{\mathrm{surr}})\right) \\
&\leq \left(R_{\mathrm{surr}}(\hat{f}_{\mathrm{surr}}) - \widehat{R}_{\mathrm{surr}}(\hat{f}_{\mathrm{surr}})\right) + \left(\widehat{R}_{\mathrm{surr}}(f^\star_{\mathrm{surr}}) - R_{\mathrm{surr}}(f^\star_{\mathrm{surr}})\right) \\
&\leq 2 \sup_{f \in \mathcal{F}} |\widehat{R}_{\mathrm{surr}}(f) - R_{\mathrm{surr}}(f)| \\
&= 2 \sup_{\boldsymbol{g} \in \mathcal{G}} |\widehat{R}_{\mathrm{surr}}(\boldsymbol{g}) - R_{\mathrm{surr}}(\boldsymbol{g})| \\
&\leq 4\mathfrak{R}_{n_{\mathrm{tr}}}(\ell \circ \mathcal{G}) + 2M_\ell \sqrt{\frac{\ln(2/\delta)}{2n_{\mathrm{tr}}}} \\
&\leq \frac{8\sqrt{2}m\mathcal{L}_\ell}{\alpha^2} \mathfrak{R}_{n_{\mathrm{tr}}}(\mathcal{F}) + 2M_\ell \sqrt{\frac{\ln(2/\delta)}{2n_{\mathrm{tr}}}},
\end{aligned}$$

where the second equality is due to that $\mathcal{G} = \{\boldsymbol{x} \mapsto \boldsymbol{T}(f(\boldsymbol{x})) \mid f \in \mathcal{F}\}$ and $\boldsymbol{T}(\cdot)$ is deterministic. □

B. Supplementary Information on the Experiments

In this appendix, we provide supplementary information on the experiments.

B.1. Datasets

We describe details of the datasets as follows.

MNIST This is a dataset of normalized grayscale images containing handwritten digits from 0 to 9. All the images are fitted into a 28×28 pixels. The total number of training images and test images is 60,000 and 10,000 respectively. We use the even digits as the positive class and odd digits as the negative class.

Fashion-MNIST This is a dataset of grayscale images of different types of modern clothes. All the images are of the size 28×28 pixels. Similar to MNIST, this dataset has 60,000 training images and 10,000 test images. We convert this 10-class dataset into a binary dataset as follows:

- The classes 'Pullover', 'Dress', 'T-shirt', 'Trouser', 'Shirt', 'Bag', 'Ankle boot' and 'Sneaker' are denoted as the positive class;
- The classes 'Coat' and 'Sandal' are denoted as the negative class.

Kuzushiji-MNIST This is a dataset of grayscale images of cursive Japanese (Kuzushiji) characters. This dataset also has all images of size 28×28. And the total number of training images and test images is 60,000 and 10,000 respectively. We convert this 10-class dataset into a binary dataset as follows:

- The classes 'ki', 're', and 'wo' are denoted as the positive class;
- The classes 'o', 'su', 'tsu', 'na', 'ha', 'ma', and 'ya' are denoted as the negative class.

CIFAR-10 This dataset is made up of color images of ten types of objects and animals. The size of all images in this dataset is 32×32. There are 5,000 training images and 1,000 test images for each class, so 50,000 training and 10,000 test images in total. We convert this 10-class dataset into a binary dataset as follows:

- The positive class consists of 'airplane', 'bird', 'deer', 'dog', 'frog', 'cat', and 'horse';
- The negative class consists of 'automobile', 'ship', and 'truck'.

The generation of each U set is the same for all four benchmark datasets. More specifically, given the number of U sets m, class priors $\{\pi_j\}_{j=1}^m$, and the set sizes $\{n_j\}_{j=1}^m$, for j-th U set, we go through the following process:

1. Randomly shuffle the benchmark dataset;
2. Randomly select $n_j^p = n_j \times \pi_j$ samples of positive class;
3. Randomly select $n_j^n = n_j - n_j^p$ samples of negative class;
4. Combine them and we obtain the j-th U set.

B.2. Models

We describe details of the model architecture and optimization algorithm as follows.

MLP It is a 5-layer fully connected perceptron with ReLU (Nair & Hinton, 2010) as the activation function. The model architecture was $d - 300 - 300 - 300 - 1$, where d is the dimension of the input. Batch normalization (Ioffe & Szegedy, 2015) was applied before each hidden layer and ℓ_2-regularization was added. Dropout (Srivastava et al., 2014) with rate 0.2 was also added before each hidden layer. The optimizer was Adam (Kingma & Ba, 2014) with the default momentum parameters ($\beta_1 = 0.9$ and $\beta_2 = 0.999$).

ResNet-32 It is a 32-layer residual network (He et al., 2016) and the architecture was as follows:
0th (input) layer: $(32 * 32 * 3)-$
1st to 11th layers: $C(3 * 3, 16) - [C(3 * 3, 16), C(3 * 3, 16)] * 5-$
12th to 21st layers: $[C(3 * 3, 32), C(3 * 3, 32)] * 5-$
22nd to 31st layers: $[C(3 * 3, 64), C(3 * 3, 64)] * 5-$

32nd layer: Global Average Pooling-1,
where $C(3*3,96)$ represents a 96-channel of $3*3$ convolutions followed by a ReLU activation function, $[\cdot]$*2 represents a repeat of twice of such layer, $C(3*3,96,2)$ represents a similar layer but with stride 2, and $[\cdot,\cdot]$ represents a building block. Batch normalization was applied for each hidden layers and ℓ_2-regularization was also added. The optimizer was Adam with the default momentum parameters ($\beta_1 = 0.9$ and $\beta_2 = 0.999$).

The MLP model was used for the MNIST, Fashion-MNIST, Kuzushiji-MNIST dataset, and the ResNet-32 model was used for the CIFAR-10 dataset.

B.3. Other Details

We implemented all the methods by Keras and conducted all the experiments on an NVIDIA Tesla P100 GPU. The batch size was 256 for all the methods. For MNIST, Fashion-MNIST, and Kuzushiji MNIST dataset, the initial learning-rate was 1e-5 for U^m-SSC and 1e-4 for the MMC based methods and LLP-VAT. For CIFAR-10 dataset, the initial learning-rate was 5e-6 for U^m-SSC and 1e-5 for the MMC based methods and LLP-VAT. In addition, the learning rate was decreased by $1/(1+\text{decay}\cdot\text{epoch})$, where the decay parameter was 1e-4. This is the built-in learning rate scheduler of Keras.

We describe details of the hyper-parameters for the baseline methods as follows.

- MMC-U^2-b (Scott & Zhang, 2020): by assuming that the number of sets $m = 2k$, this baseline method firstly pairs all the U sets and then linearly combines the unbiased balanced risk estimator of each pair, The learning objective is

$$\widehat{R}_{\text{MMC-U}^2\text{-b}}(f) = \sum\nolimits_{j=1}^{k} \omega_j \widehat{R}_{\text{U}^2\text{-b}}(f),$$

  where

$$\widehat{R}_{\text{U}^2\text{-b}}(f) = \frac{c_{b1}^{+}}{n}\sum_{i=1}^{n_1}\ell_b(f(\boldsymbol{x}_i^1),+1) - \frac{c_{b2}^{+}}{n}\sum_{j=1}^{n_2}\ell_b(f(\boldsymbol{x}_j^2),+1) - \frac{c_{b1}^{-}}{n}\sum_{i=1}^{n_1}\ell_b(f(\boldsymbol{x}_i^1),-1) + \frac{c_{b2}^{-}}{n}\sum_{j=1}^{n_2}\ell_b(f(\boldsymbol{x}_j^2),-1),$$

  $c_{b1}^{+} = \frac{1-\pi_2}{2(\pi_1-\pi_2)}$, $c_{b1}^{-} = \frac{\pi_2}{2(\pi_1-\pi_2)}$, $c_{b2}^{+} = \frac{1-\pi_1}{2(\pi_1-\pi_2)}$, and $c_{b2}^{-} = \frac{\pi_1}{2(\pi_1-\pi_2)}$. For the pairing process, since we use the uniform set sizes, i.e., the set size of each U set is the same as n_{tr}/m ($n_{\text{tr}} = 60,000$ in MNIST, Fashion-MNIST, and Kuzushiji-MNIST, $n_{\text{tr}} = 50,000$ in CIFAR-10), we pair all the U sets following Proposition 9 in Appendix S6 of Scott & Zhang (2020), i.e., match the U set with the largest class prior π_j with the smallest, the U set with the second largest class prior π_j with the second smallest, and so on. For the combination weights, we set them following Theorem 5 in Section 2.2 of Scott & Zhang (2020). More specifically, for the j-th pair of U sets: $\mathcal{X}_{\text{tr}}^1$ and $\mathcal{X}_{\text{tr}}^2$, assume $\pi_1 > \pi_2$, since we use uniform set sizes, the optimal weights $\omega_j \propto (\pi_1-\pi_2)^2$. So we set the weight ω_j as $(\pi_1-\pi_2)^2$ and then normalize all of them to sum to 1, i.e., $\sum_{j=1}^{k}\omega_j = 1$.
- MMC-U^2: this method improves the MMC-U^2-b baseline by replacing the unbiased balanced risk estimator $\widehat{R}_{\text{U}^2\text{-b}}(f)$ with the unbiased risk estimators $\widehat{R}_{\text{U}^2}(f)$ (Lu et al., 2019). The learning objective is

$$\widehat{R}_{\text{MMC-U}^2}(f) = \sum\nolimits_{j=1}^{k} \omega_j \widehat{R}_{\text{U}^2}(f),$$

  where

$$\widehat{R}_{\text{U}^2}(f) = \underbrace{\frac{c_1^{+}}{n}\sum_{i=1}^{n_1}\ell_b(f(\boldsymbol{x}_i^1),+1) - \frac{c_2^{+}}{n}\sum_{j=1}^{n_2}\ell_b(f(\boldsymbol{x}_j^2),+1)}_{\widehat{R}_{\text{U}^2\text{-p}}(f)} \underbrace{- \frac{c_1^{-}}{n}\sum_{i=1}^{n_1}\ell_b(f(\boldsymbol{x}_i^1),-1) + \frac{c_2^{-}}{n}\sum_{j=1}^{n_2}\ell_b(f(\boldsymbol{x}_j^2),-1)}_{\widehat{R}_{\text{U}^2\text{-n}}(f)},$$

$c_1^+ = \frac{(1-\pi_2)\pi_{\mathcal{D}}}{\pi_1-\pi_2}$, $c_1^- = \frac{\pi_2(1-\pi_{\mathcal{D}})}{\pi_1-\pi_2}$, $c_2^+ = \frac{(1-\pi_1)\pi_{\mathcal{D}}}{\pi_1-\pi_2}$, and $c_2^- = \frac{\pi_1(1-\pi_{\mathcal{D}})}{\pi_1-\pi_2}$. The pairing process and the combination weights setup follow those of MMC-U^2-b.

- MMC-U^2-c: this method improves the MMC-U^2 baseline by replacing the unbiased risk estimators $\widehat{R}_{\mathrm{U}^2}(f)$ with the non-negative risk estimators $\widehat{R}_{\mathrm{U}^2\text{-c}}(f)$ (Lu et al., 2020). The learning objective is

$$\widehat{R}_{\text{MMC-U}^2\text{-c}}(f) = \sum\nolimits_{j=1}^{k} \omega_j \widehat{R}_{\mathrm{U}^2\text{-c}}(f),$$

where

$$\widehat{R}_{\mathrm{U}^2\text{-c}}(f) = f_{\mathrm{c}}(\widehat{R}_{\mathrm{U}^2\text{-p}}(f)) + f_{\mathrm{c}}(\widehat{R}_{\mathrm{U}^2\text{-n}}(f)).$$

According to Lu et al. (2020), the *generalized leaky ReLU* function, i.e.,

$$f_{\mathrm{c}}(r) = \begin{cases} r & (r \geq 0), \\ -\kappa r & (r < 0), \end{cases}$$

for $\kappa \geq 0$, works well as the correction function f_{c}, so we choose it for implementing this baseline method. The hyper-parameter κ was chosen based on a validation dataset, and the pairing process and the combination weights setup follow those of MMC-U^2-b.

- LLP-VAT (Tsai & Lin, 2020): this baseline method is based on empirical proportion risk minimization. The learning objective is

$$\widehat{R}_{\text{prop-c}}(f) = \widehat{R}_{\text{prop}}(f) + \alpha \ell_{\text{cons}}(f),$$

where

$$\widehat{R}_{\text{prop}}(f) = \sum\nolimits_{j=1}^{m} d_{\text{prop}}(\pi_j, \hat{\pi}_j)$$

is the proportion risk, π_j and

$$\hat{\pi}_j = \frac{1}{n_j} \sum_{i=1}^{n_j} \frac{1 + \mathrm{sign}(f(\boldsymbol{x}_i^j) - 1/2)}{2}$$

are the true and predicted label proportions for the j-th U set $\mathcal{X}_{\mathrm{tr}}^j$, d_{prop} is a distance function, and

$$\ell_{\text{cons}}(f) = d_{\text{cons}}(f(\boldsymbol{x}), f(\hat{\boldsymbol{x}}))$$

is the consistency loss, d_{cons} is a distance function, $\hat{\boldsymbol{x}}$ is a perturbed input from the original one $\boldsymbol{x}$. We set the hyper-parameters $\alpha = 0.05$ and the perturbation weight $\mu = 6.0$ for LLP-VAT following the default implementation in their paper (Tsai & Lin, 2020).

C. Supplementary Experimental Results

In this appendix, we provide supplementary experimental results.

C.1. Comparison with State-of-the-art Methods

Please find Table 5 the final classification errors of comparing our proposed method with state-of-the-art methods on learning from 10, 25, and 50 U sets (corresponds to Figure 2).

In the experiments, we also find that the empirical training risk of the proposed U^m-SSC is obviously higher than all other baseline methods. This is due to the added transition layer and the rescales the output range. We provide a detailed explanation as follows.

By using the monotonicity of the transition function $T_j(\cdot)$ (Menon et al., 2015), we can compute the range of the model output. Since $g(\boldsymbol{x}) \in [0, 1]$, by plugging in $g(\boldsymbol{x}) = 0$ and $g(\boldsymbol{x}) = 1$ respectively we obtain

$$T_j(0) = \frac{b_j}{d} = \frac{\rho_j \pi_{\mathcal{D}}(1-\pi_j)}{\sum_{j=1}^m \rho_j \pi_{\mathcal{D}}(1-\pi_j)},$$
$$T_j(1) = \frac{a_j + b_j}{c_j + d} = \frac{\rho_j \pi_j(1-\pi_{\mathcal{D}})}{\sum_{j=1}^m \rho_j \pi_j(1-\pi_{\mathcal{D}})}.$$

Table 5. Means (standard deviations) of the classification error over three trials in percentage of each method on learning from 10, 25 and 50 U sets. Best and comparable methods (paired *t*-test at significance level 5%) are highlighted in boldface.

Dataset	Sets	MMC-U^2-b	MMC-U^2	MMC-U^2-c	LLP-VAT	U^m-SSC
MNIST	10	7.7(0.55)	8.03(0.74)	4.46(0.23)	3.62(0.38)	**3.05(0.08)**
	25	5.35(0.22)	5.32(0.28)	3.69(0.11)	3.28(0.35)	**2.51(0.02)**
	50	5.81(0.22)	5.82(0.12)	3.29(0.09)	3.02(0.22)	**2.86(0.04)**
Fashion-MNIST	10	16.63(1.38)	9.49(0.37)	8.12(0.51)	21.23(3.52)	**6.5(0.21)**
	25	11.1(0.45)	9.12(0.1)	7.45(0.1)	26.66(0.4)	**6.14(0.02)**
	50	11.18(0.53)	9.6(0.47)	8.52(0.48)	27.92(2.22)	**6.6(0.06)**
Kuzushiji-MNIST	10	16.25(0.61)	15.23(0.3)	12.88(0.35)	16.12(0.41)	**9.83(0.4)**
	25	15.93(0.71)	14.02(0.12)	10.18(0.33)	19.48(1.84)	**8.98(0.07)**
	50	15.8(0.37)	12.46(0.43)	9.69(0.37)	18.94(0.4)	**8.97(0.52)**
CIFAR-10	10	15.83(0.21)	16.01(0.32)	14.33(0.06)	19.38(0.05)	**13.43(0.14)**
	25	19.6(0.77)	16.18(0.27)	14.19(0.25)	16.89(0.15)	**13.31(0.13)**
	50	21.1(1.03)	16.08(0.38)	14.28(0.13)	17.66(0.57)	**13.32(0.19)**

Table 6. Means (standard deviations) of the classification error over three trials in percentage for the U^m-SSC method tested on inaccurate class priors.

Dataset	Sets	True	$\epsilon = 0.05$	$\epsilon = 0.1$	$\epsilon = 0.15$	$\epsilon = 0.2$
MNIST	10	2.54(0.02)	2.64(0.06)	3.31(0.18)	2.98(0.14)	3.84(0.25)
	50	2.45(0.04)	2.52(0.02)	2.69(0.04)	3.11(0.19)	3.16(0.13)
Fashion-MNIST	10	6.22(0.05)	6.31(0.03)	6.13(0.13)	6.61(0.04)	9.39(0.19)
	50	6.37(0.26)	6.39(0.17)	6.76(0.11)	7.64(0.22)	10.91(0.47)
Kuzushiji-MNIST	10	8.74(0.24)	8.97(0.23)	9.77(0.29)	11.31(0.21)	11.62(0.56)
	50	9.0(0.22)	9.27(0.26)	9.15(0.15)	9.38(0.18)	10.61(0.03)
CIFAR-10	10	13.54(0.23)	13.7(0.25)	14.43(0.22)	16.82(0.29)	19.7(0.54)
	50	13.55(0.18)	13.75(0.09)	14.19(0.22)	15.69(0.21)	18.84(0.26)

According to our generation processes of class priors and set size, $\pi_j \in [0.1, 0.9]$ and $\rho_j = 1/m$ for any $j = 1, \ldots, m$. The upper bound of the model output $\max(T_j(0), T_j(1))$ takes value between 0.01 and 0.1. As a result, the cross-entropy loss gives its value in range $[2.3, 4.6]$, which is relatively high than usual training loss. We note that this high training loss has an effect on hyper-parameters tuning, especially for the learning rate. We may need a relatively small learning rate for better performance of our method.

C.2. Robustness against Inaccurate Class Priors

Please find Table 6 the final classification errors of our method on learning from 50 U sets with inaccurate class priors (corresponds to Figure 3).