Progressive Identification of True Labels for Partial-Label Learning

Jiaqi Lv, Miao Xu, Lei Feng, Gang Niu, Xin Geng, Masashi Sugiyama

Abstract

Partial-label learning (PLL) is a typical weakly supervised learning problem, where each training instance is equipped with a set of candidate labels among which only one is the true label. Most existing methods elaborately designed learning objectives as constrained optimizations that must be solved in specific manners, making their computational complexity a bottleneck for scaling up to big data. The goal of this paper is to propose a novel framework of PLL with flexibility on the model and optimization algorithm. More specifically, we propose a novel estimator of the classification risk, theoretically analyze the classifier-consistency, and establish an estimation error bound. Then we propose a progressive identification algorithm for approximately minimizing the proposed risk estimator, where the update of the model and identification of true labels are conducted in a seamless manner. The resulting algorithm is model-independent and loss-independent, and compatible with stochastic optimization. Thorough experiments demonstrate it sets the new state of the art.

1. Introduction

The increasing demand for massive data used in training modern machine learning models, such as deep neural networks (DNNs), makes it inevitable that not all examples are equipped with strong supervision information. Weak supervision may result in a detrimental effect on model training and has been widely studied (Zhou, 2017; Patrini et al., 2017; Kamnitsas et al., 2018; Lu et al., 2019; Gong et al., 2019). We focus on an important weakly supervised learning problem called partial-label learning (PLL) (Cour et al., 2011; Wang et al., 2019; Lyu et al., 2019), where each training instance is associated with a set of candidate labels among which exactly one is true, reducing the overhead of finding exact label from ambiguous candidates. This problem arises in many real-world tasks such as automatic image annotation (Chen et al., 2018), web mining (Luo & Orabona, 2010), etc.

Related research on PLL was pioneered by a multiple-label learning (MLL) method (Jin & Ghahramani, 2003). Despite the same form of supervision information, i.e., a set of candidate labels, that each training instance is assigned with, a vital difference between MLL and PLL is that the goal of PLL is identifying the only one true label among candidate labels whereas for MLL identifying an arbitrary label in the candidate label set is acceptable, i.e., treating all candidate labels equally. In practice, Jin & Ghahramani (2003) formulated an unconstrained learning objective to minimize the KL divergence between the class prior and the model-based conditional distribution, and solved it by EM algorithm, resulting in a procedure iterating between estimating the prior distribution and training the model.

Jin & Ghahramani (2003) used the class prior as the fitting target of the model, in which the label with maximum prior probability can be naturally regarded as the “true label”. Coincidentally, the true label of training instance in PLL is obscured by candidate labels that hinder the learning. The key to success is also identifying the true label. Therefore, the foundational work enlightened the successors to identify the true label reasonably. Along this line, great efforts have been made in designing learning objectives with some elaborate constraints for making assumptions on the model (Chen et al., 2013; 2014) or better leveraging the prior knowledge (Nguyen & Caruana, 2008; Zhang & Yu, 2015; Yu & Zhang, 2017; Gong et al., 2017; Feng & An, 2019a) so as to recover the true labeling information.

However, the constrained learning objectives of the existing methods are coupled to some specific optimization algorithms. A non-linear time complexity in the total volume of data becomes a major limiting factor when these methods envision large amounts of training data. Furthermore, Bottou & Bousquet (2007) proved stochastic optimization,
which few of the existing methods are compatible with, is the best choice for large-scale learning problems considering the estimation-optimization tradeoff. Therefore, the PLL problem is still practically challenging in scaling up to big data. In order to magnify PLL’s potential on big data, we rethink critically whether the original unconstrained method in Jin & Ghahramani (2003) should be discarded, or is worth further studying in the deep learning era?

Our answer is in two folds. First, the decade-old method restricted to a simple linear model fed by the handcrafted features, which may be incompetent to represent and discriminate. Moreover, few PLL work hitherto can be generalized to deep network architectures. We would like to advance the previous work to enjoy the leading-edge models and optimizers from deep learning communities (Goodfellow et al., 2016). Second, the essence of our investigation is an algorithm agnostic in classification models. Thus we focus on a method that does not benefit purely from the network architecture, but also an innovative algorithm design.

In this paper, we aim at proposing a method of PLL with flexibility on the model and stochastic optimization. Towards this goal, we first derive a classifier-consistent risk estimator, and then propose a novel algorithm that is compatible with arbitrary multi-class classifier (ranging from linear to deep model) and stochastic optimizers. Experiments verify its superiority on synthetic and real-world partial-label datasets. Our contributions can be summarized as follows:

- Theoretically, we propose a classifier-consistent risk estimator for PLL, i.e., the classifier learned from partially labeled data converges to the optimal one learned from ordinarily labeled data under mild conditions. Then, we establish an estimation error bound for it.
- Practically, we propose a progressive identification method. The proposed method operates in a mini-batched training manner where the update of the model and the identification of true labels are accomplished seamlessly. This method is model-independent and loss-independent, as well as viable for any stochastic optimization (e.g., Robbins & Monro, 1951; Duchi et al., 2011).

2. Background

In this section, we formalize ordinary multi-class classification, partial-label learning, and complementary-label learning, and briefly review the related work.

2.1. Ordinary Multi-Class Classification

In ordinary multi-class classification, let \( X \subseteq \mathbb{R}^d \) be the instance space and \( Y = [c] \) be the label space, where \( d \) is the feature space dimension, \([c] := \{1, 2, \ldots, c\}\) and \( c > 2 \) is the number of classes. Let \( p(x,y) \) be the underlying joint density of random variables \((X,Y) \in X \times Y\). The target is to learn a classifier \( g : X \rightarrow \mathbb{R}^c \) that minimizes the estimator of the classification risk:

\[
\mathcal{R}(g) = \mathbb{E}_{(X,Y) \sim p(x,y)}[\ell(g(X), e^Y)],
\]

in which \( \ell : \mathbb{R}^c \times e^Y \rightarrow \mathbb{R} \) is a proper loss, i.e., it is continuous non-negative and \( \ell(\hat{y}, y) = 0 \) only when \( \hat{y} = y \). \( e^Y = \{e^i : i \in Y\} \) denotes the standard canonical vector in \( \mathbb{R}^c \), i.e., the \( i \)-element in \( e^i \) equals 1 and others equal 0. Typically, the predicted label \( \hat{Y} \) is assumed to take the following form:

\[
\hat{Y} = \arg \max_{i \in Y} g_i(X),
\]

where \( g_k(\cdot) \) is the \( k \)-th element of \( g(\cdot) \) that is interpreted as an estimate of the true label probability, i.e., \( g_k(X) = p(Y = k|X) \). The hypothesis in \( g \) with minimal error \( g^* = \arg \min_{g \in G} \mathcal{R}(g) \) is the optimal classifier. We say a method is classifier-consistent if the learned classifier by the method converges to \( g^* \) by increasing the example size (Patrini et al., 2017; Xia et al., 2019).

Since \( p(x,y) \) is usually unknown, the expectation in Eq. (1) is typically approximated by the average over the training examples \( \{(x_i, y_i)\}_{i=1}^n \sim p(x,y) : \mathcal{R}(g) = \frac{1}{n} \sum_{i=1}^n [\ell(g(x_i), e^{y_i})] \), and \( \hat{g} = \arg \min_{g \in G} \mathcal{R}(g) \) is returned by the empirical risk minimization (ERM) principle (Vapnik, 1998).

2.2. Partial-Label Learning

Different from ordinary multi-class classification, in PLL, the labelling information is weakened into candidate label set \( S \subseteq \mathcal{Y} \), where \( \mathcal{S} = \{p(Y)/\emptyset/\mathcal{Y}\} \) is the power set of \( \mathcal{Y} \) except for the empty set and the whole label set. Hence we need to train a classifier with access only to the partially labeled examples \((X,S)\) drawn from the distribution with probability density \( p(x,s) \), which is a marginal density of the complete density \( p(x,y,s) \). Note that \( p(x,y,s) \) can be decomposed into \( p(x,s) \) and the class-probability density \( p(y|x,s) \), or equivalently into the marginal density \( p(x,y) \) and the class-conditional density \( p(s|x,y) \). Then the basic definition for PLL that the true label \( Y \) of an instance \( X \) must be in its candidate label set \( S \) can be formulated as

\[
\Pr_{(X,Y) \sim p(x,y), S \sim p(s|x,y)}(Y \in S) = 1.
\]

The PLL risk estimator is defined over \( p(x,s) \):

\[
\mathcal{R}_{PLL}(g) = \mathbb{E}_{(X,S) \sim p(x,s)}[\ell_{PLL}(g(X), S)],
\]

where \( \ell_{PLL} : \mathbb{R}^c \times \mathcal{P}(\mathcal{Y}) \rightarrow \mathbb{R} \). The goal of PLL is still inducing a multi-class classifier to assign the true labels for the unseen instances.

To estimate the risk in partially labeled data, defining \( \ell_{PLL} \) matters. Jin & Ghahramani (2003) defined
\textbf{3. Learning with Partial Labels}

In this section, we define a classifier-consistent risk estimator, and theoretically establish an estimation error bound.

\textbf{3.1. Classifier-Consistent Risk Estimator}

To make Eq. (3) estimable, an intuitive way is through a surrogate loss that treats all the candidate labels equally (Jin & Ghahramani, 2003): $\ell_{PLL}(g(X), S) = 1/|S| \sum_{i \in S} \ell(g(X), e^i)$. Nonetheless, the true label may be overwhelmed by the destructive outputs of the multiple false positive labels. Therefore, we consider that only the true label contributes to retrieving the classifier. Capturing this idea we define the PLL loss as the \textit{minimal loss} over the candidate label set:

$$\ell_{PLL}(g(X), S) = \min_{i \in S} \ell(g(X), e^i),$$

which immediately leads to a new risk estimator:

$$R_{PLL}(g) = \mathbb{E}_{(X,S) \sim p(x,s)} \min_{i \in S} \ell(g(X), e^i).$$

For a detailed description of Eq. (4), we can disassemble it by defining a function in terms of $Y$ as a classifier $g$’s best guess at the true label:

$$Y^g = \arg \min_{i \in S} \ell(g(X), e^i).$$

Then correspondingly the PLL loss can be rewritten into an equivalent expression, i.e., $g$ only needs to fit the best guess $Y^g$ among $S$:

$$\ell_{PLL}(g(X), S) = \min_{i \in S} \ell(g(X), e^i) = \ell(g(X), e^{Y^g}).$$

From now on, we explain under which conditions Eq. (5) is classifier-consistent. We start investigating this question from the following two lemmas.

\textbf{Lemma 1.} (Summarized from Sec.3.1 of Liu & Dietterich, 2014) The ambiguity degree is defined as

$$\gamma = \sup_{(X,Y) \sim p(x,y), \bar{Y} \in Y, S \sim p(s|x,y)} \mathbb{P}(\bar{Y} \in S).$$

\textbf{If} $\gamma < 1$, i.e. under the small ambiguity degree condition, \textit{the PLL problem is ERM learnability}.

\gamma is the maximum probability of a negative label $\bar{Y}$ co-occurs with the true label $Y$. The small ambiguity degree condition implies that except for the true label, no other labels will be one hundred percent included in the candidate label set, which guarantees a classification error made on any instance will be detected with probability at least $1 - \gamma$.

Then we provide a condition of the identifiability of the ordinary optimal classifier.

\textbf{Lemma 2.} (Yu et al., 2018) \textit{If} $\ell$ is the cross-entropy loss or mean squared error loss, the optimal classifier $g^*_X$ of Eq. (1) satisfies $g^*_X(X) = p(Y = i | X)$.

With the above lemmas, we can prove Eq. (5) possess classifier-consistency under a reasonable assumption, that is, learning is conducted under the deterministic case.
Theorem 1. Under the deterministic scenario, if the small ambiguity degree condition is satisfied, and cross-entropy or mean squared error loss is used, then, the PLL optimal classifier \( g_{PLL}^* \) of Eq. (5) is equivalent to the ordinary optimal classifier \( g^* \) of Eq. (1), i.e., \( g_{PLL}^* = g^* \).

It is natural to assume a deterministic learning scenario, i.e., the true label \( Y_X \) of an instance is uniquely determined by the measurable function \( g^{**} : p(Y = i|X) = g^{**}_i(X), Y_X = \arg \max_{i \in Y} g^{**}_i(X), \) for ensuring the basic definition for PLL, i.e., Eq. (2) always holds. In the light of Lemma 2, \( g^* = g^{**} \). In this case, the optimal classifier \( g^* \) can be recovered by minimizing Eq. (5). The proof is provided in Appendix.

3.2. Estimation Error Bound

Let \( \hat{R}_{PLL} \) be the empirical counterpart of \( R_{PLL} \), and \( \hat{g}_{PLL} = \arg \min_{g \in \mathcal{G}} \hat{R}_{PLL}(g) \) be the empirical risk classifier. Suppose \( \mathcal{G}_y \) be a class of real functions, and \( \mathcal{G} = \oplus_{y \in [Y]} \mathcal{G}_y \) be a c-valued function class. Assume there exists \( C_{\mathcal{G}} > 0 \) such that \( \sup_{g \in \mathcal{G}} ||g||_{\infty} \leq C_{\mathcal{G}} \), and the loss function \( \ell(g(X), Y) \) is Lipschitz continuous for all \( |g| \leq C_{\mathcal{G}} \) with a Lipschitz constant \( L_{\ell} \) upper-bounded by \( M \), i.e., \( M = \sup_{X \in X, |g| \leq C_{\mathcal{G}}, Y \in Y} \ell(g(X), Y) \). The Rademacher complexity of \( \mathcal{G} \) over \( p(x) \) with sample size \( n \) as defined is \( R_n(\mathcal{G}) \) (Bartlett & Mendelson, 2002). Then we have the following estimation error bound.

Theorem 2. For any \( \delta > 0 \), we have with probability at least \( 1 - \delta \),

\[
\hat{R}_{PLL}(\hat{g}_{PLL}) - R_{PLL}(g_{PLL}^*) \leq 4\sqrt{2cL_{\ell}} \sum_{y=1}^{c} R_n(\mathcal{G}_y) + 2M \sqrt{\frac{\log(2/\delta)}{2n}}.
\]

The proof is given in Appendix. As \( n \to \infty, R_n(\mathcal{G}_y) \to 0 \) for all parametric models with a bounded norm. Hence, \( \hat{R}_{PLL}(\hat{g}_{PLL}) \to R_{PLL}(g_{PLL}^*) \) as the number of training data approaches infinity.

4. Benchmark Solution

In the previous section, based on a novel PLL loss Eq. (4) we provided a risk estimator Eq. (5) with intriguing theoretical properties. Although the estimator is without any assumption on the model used, the min operator makes optimization difficult, because if a wrong label \( i \) in Eq. (5) is selected in the beginning, the optimization will focus on the wrong label till the end. Thus we design a benchmark solution to approximately solve Eq. (5) according to the idea behind Eq. (4) that only one (true) label should be taken into account, and consequently we can easily implement the algorithm over arbitrary multi-class classifier and powerful stochastic optimization.

Algorithm 1 PRODEN Algorithm

input \( D \): the partial-label training set \( \{(x_i, s_i)\}_{i=1}^{n} \),
\( T \): number of epochs;
output \( \Theta \): model parameter for \( g(x; \Theta) \)

1: Let \( \mathcal{A} \) be an stochastic optimization algorithm;
2: Initialize uniform weights \( w^0 \) according to Eq. (9);
3: for \( t = 1 \) to \( T \) do
4: Shuffle training set \( D \) into \( B \) mini-batches;
5: for \( k = 1 \) to \( B \) do
6: Compute \( L \) according to Eq. (8);
7: Set gradient \( -\nabla_{\Theta} L \);
8: Update \( w \) according to Eq. (10);
9: Update \( \Theta \) by \( \mathcal{A} \);
10: end for
11: end for

In particular, we relax the min operator in Eq. (4) by the dynamic weights. In the first place, we require that \( \ell \) can be decomposed into each label, i.e.,

\[
\ell(g(X), e^Y) = \sum_{i=1}^{c} \ell(g_i(X), e_i^Y),
\]

where \( e_i^Y \) is the \( i \)-th element of \( e^Y \). Many commonly used multi-class loss functions are decomposable, for example, \( \ell \) can be the cross-entropy loss: \( \ell_{CE}(g(X), e^Y) = -\sum_{i=1}^{c} e_i^Y \log(g_i(X)) \) or mean squared error loss: \( \ell_{MSE}(g(X), e^Y) = \sum_{i=1}^{c} (g_i(X) - e_i^Y)^2 \). In this case, the empirical risk estimator is rewritten as follows:

\[
\hat{R}_{PLL} = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{c} w_{ij} \ell(g_j(x_i), e_j^s),
\]

where \( e_j^s \) is the \( j \)-th coordinate of \( e^s \) and \( e_j^s = \sum_{k \in s_i} e_k \). \( w_i \in \Delta^{c-1} \) is the \( c \)-dimensional simplex and the \( j \)-th element of it, i.e., \( w_{ij} \), is the corresponding label weight, i.e., the confidence of the \( j \)-th label being the true label of the \( i \)-th instance. With appropriate weights, i.e., \( w_{ij} = 1 \) if \( \arg \min_{k \in s_i} \ell(g(x_i), e_k) = j \) and 0 otherwise, which is in consonance with Eq. (6), Eq. (8) is equivalent to the empirical counterpart of Eq. (5). Ideally, the label with weight 1 is exactly the true label, i.e., \( j \) is the true label of \( x_i \) if \( w_{ij} = 1 \), which means we have identified the true label successfully. To eventually achieve such an ideal situation, we propose a effective algorithm named \textit{PRODEN} (PROgressive IDENTification).

Because the weights are latent, the minimizer of Eq. (8) cannot be solved directly. Inspiring by the EM algorithm,
we start by training an initial model with uniform weights:

\[ w_{ij} = \begin{cases} 
1/|s_i| & \text{if } j \in s_i, \\
0 & \text{otherwise.}
\end{cases} \tag{9} \]

Thanks to the memorization effects (Arpit et al., 2017; Han et al., 2018), the networks will remember “frequent patterns” in the first few iterations. If the small ambiguity degree condition is satisfied, they tend to remember the true labels in the initial epochs, which guides us towards a discriminative classifier giving relatively low losses for more possible true labels. Next, the problem becomes how to retain the informative predictions and gradually filter out the false positive labels. Instead of estimating the weights in a separate E-step, we tackle it simply using the current predictions for slightly putting more weights on more possible labels:

\[ w_{ij} = \begin{cases} 
g_j(x_i)/\sum_{k \in s_i} g_k(x_i) & \text{if } j \in s_i, \\
0 & \text{otherwise.} \tag{10} \end{cases} \]

In this fashion, the true labels are progressively identified, and the refined labels in turn help to improve the classifier. The overall procedure is summarized in Algorithm 1, whose identification step (Step 8) and optimization step (Step 9) are conducted in a seamless manner. Next we provide more detailed arguments about the superiority and generalization of the proposed method.

Firstly, PRODEN gets rid of the overfitting issues of EM methods. The previous iterative EM methods (Jin & Ghahramani, 2003; Tang & Zhang, 2017; Feng & An, 2019a) trained the model until convergence in the M-step, but overemphasizing the convergence may result in redundant computation and overfitting issues, as the model will eventually fit the initial inexact prior knowledge and make a less informative estimate in the E-step on which the subsequent learning is based. To mitigate these problems, our method advances the procedure by merging the E-step and M-step. Since the learning process has no clear separation of the E-step and M-step, the weights can be updated at any epoch such that the local convergence within each epoch is not necessary in our training process.

Secondly, PRODEN has great flexibility for models and loss functions. In the deep learning era, loss functions with different theoretical results are one of the key factors that affect the performance of DNNs (Ghosh et al., 2017). In this way, models are welcomed to be loss-independent that allows flexibility with any loss function (Ishida et al., 2019). However, existing PLL methods were restricted to some specific loss functions for optimizable, e.g., Jin & Ghahramani (2003) limited the loss function to the KL divergence; the loss function of Yao et al. (2020b) was a combination of multiple cross-entropy losses. Instead, our proposal is flexible enough to be compatible with a large group of decomposable losses. Moreover, we will show in Appendix that the proposed method is provably a generalization of Jin & Ghahramani (2003).

5. Experiments

In this section, we experimentally analyze the proposed method PRODEN, and compare with state-of-the-art PLL methods. The implementation is based on PyTorch (Paszke et al., 2019) and experiments were carried out with NVIDIA Tesla V100 GPU; it is available at https://github.com/lvcrezia77/PRODEN.

Datasets We use four widely used benchmark datasets including MNIST (LeCun et al., 1998), Fashion-MNIST (H. Xiao & Vollgraf, 2017), Kuzushiji-MNIST (Clanuwat et al., 2018), and CIFAR-10 (Krizhevsky & Hinton, 2009), and five datasets from the UCI Machine Learning Repository (Krizhevsky & Hinton, 2009), including Yeast, Texture, Dermatology, Synthetic Control, and 20Newsgroups. Similar to the corrupted strategy in label noise (Patrini et al., 2017; Han et al., 2018), we manually corrupt these datasets into partially labeled versions by a flipping probability \( q \) where \( q = \Pr(\tilde{y} = 1|y = 0) \) gives the probability that a false positive label \( \tilde{y} \) is flipped from negative label \( y \). We adopt a binomial flipping strategy: \( c = 1 \) independent experiments are conducted on all training examples, each determining whether a negative label is flipped with probability \( q \). Then for the examples that none of the negative labels is flipped, we additionally flip a random negative label to the candidate label set for ensuring all the training examples are partially labeled. In this paper, we conduct the experiments under both less-partial circumstances \( q = 0.1 \) and strong-partial circumstances \( q = 0.7 \). In addition, five real-world partial-label datasets are adopted, including Lost (Cour et al., 2011), Birdsong (Briggs et al., 2012), MSRCv2 (Liu & Dietterich, 2012), Soccer Player (Zeng et al., 2013), and Yahoo! News (Guillaumin et al., 2010).

Baselines We first analyze PRODEN by comparing it with seven baseline methods based on DNNs, including three variants of PRODEN, two supervised methods, and two state-of-the-art PLL methods:

- **PRODEN-itera** means updating the label weights in the iterative EM manner instead of a seamless manner proposed in Section 4.
- **PRODEN-sudden** means performing a sudden identification, i.e., updating the weights \( w_{ik} = 1 \) if \( \arg\max_{j \in s_i} g_j(x_i) = k \) and \( w_{ij} = 0, \forall j \neq k \) in every iteration step.
- **PRODEN-naive** means never updating the uniform weights.
- **PN-oracle** means supervised learning from ordinary labels. It should achieve the best performance, and is used merely for a proof of concept.
We employ multiple base models, including linear model, 5-layer perceptron (MLP), 12-layer ConvNet (Laine & Aila, 2017) and 32-layer ResNet (He et al., 2016) to show that our proposal is compatible with a wide family of learning models. The detailed descriptions of the datasets with their corresponding models are provided in Appendix. The optimizer is stochastic gradient descent (SGD) (Robbins & Monro, 1951) with momentum 0.9. We train each model 500 epochs with softmax function and cross-entropy loss. And PRODEN-itera updates label weights every 100 epochs.

We further compare PRODEN with six classical PLL methods that can hardly be implemented by DNNs on small-scale datasets.

Figure 1. Test accuracy for various models and datasets. Dark colors show the mean accuracy of 5 trials and light colors show standard deviation. Fashion is short for Fashion-MNIST, Kuzushiji is short of Kuzushiji-MNIST, CIFAR is short of CIFAR-10.
datasets (UCI and real-world datasets). They are four parametric methods: SURE (Feng & An, 2019a), CLPL (Cour et al., 2011), ECOC (Zhang et al., 2017), PL SVM (Nguyen & Caruana, 2008), and two non-parametric methods: PLkNN (Hullermeier & Beringer, 2006) and IPAL (Zhang & Yu, 2015). Their hyper-parameters are selected according to the suggested parameter settings in original papers. Detailed information is provided in Appendix. For a fair comparison, PRODEN and all the parametric methods employ a linear model. We average the classification accuracy of PRODEN over the last 10 epochs as the results to prove that PRODEN is always stable and will not cause performance degradation due to overfitting when the number of epochs increases.

Results on benchmark datasets We record inductive results which indicate the classification accuracy in the test set (test accuracy). The means and standard deviations of test accuracy based on 5 random samplings in the binomial case are shown in Figure 1. The transductive accuracy which reflects the ability in identifying the true labels in the training set can be found in Appendix.

We first observe the performance when $q = 0.1$ (the left two columns). PRODEN is always the best method and comparable to PN-oracle with all the models. PRODEN-itera is comparable to PRODEN with a linear model, but its performance deteriorates drastically with complex models because of the overfitting issues. This phenomenon is consistent with the discussions in Section 4. In PRODEN-naive, the weights are always equally distributed, while in PRODEN-sudden, the sudden identification concentrates all the weights to the winners, resulting in their poorer performance.

Next, we compare these methods when $q = 0.7$ (the right two columns). This is significantly harder than $q = 0.1$. But when the task difficulty is relatively low (e.g. MNIST, Fashion-MNIST), PRODEN is still comparable to PN-oracle with such a large flipping probability, while the performance of the baselines is greatly reduced. The superiority always stands out for PRODEN compared with two latest deep methods GA and D²CNN.

Analysis on the ambiguity degree It is intuitive that the higher the ambiguity degree is, the more difficult it is to find the true labels. We investigate the influence of ambiguity degree $γ$ through a pair flipping strategy: only the classes that are similar to the true class will be put into the candidate label set with probability $q$. We gradually move $q$ from 0.5 to 0.9 to simulate $γ$ ($γ \rightarrow q$ as $n \rightarrow \infty$), and the experimental results on Kuzushiji-MNIST are reported in Figure 2, and phenomena on other datasets are similar and reported in Appendix. We can see that even when the ambiguity degree is high, PRODEN is still highly competitive. PRODEN tends to be less affected with increased ambiguity, while the baselines are affected severely.

Results on small-scale datasets We perform five-fold cross-validation, and use paired $t$-test at 5% significance level. Table 1 and Table 2 report the mean test accuracy with standard deviation on UCI datasets and the real-world datasets, respectively. */○ represents whether PRODEN is significantly better/worse than the comparing methods. Clearly, PRODEN is overall the best performing, confirming the advantage of PRODEN afforded not only by the network architecture, but also the progressive identification process.
Table 1. Test accuracy (mean±std) on the UCI datasets in the binomial case.

<table>
<thead>
<tr>
<th></th>
<th>q</th>
<th>Yeast</th>
<th>Texture</th>
<th>Dermatology</th>
<th>Synthetic Control</th>
<th>20Newsgroups</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRODEN</td>
<td>0.1</td>
<td>59.05±4.66%</td>
<td>99.64±0.16%</td>
<td>96.16±3.26%</td>
<td>98.33±1.32%</td>
<td>77.28±0.68%</td>
</tr>
<tr>
<td></td>
<td>0.7</td>
<td>55.15±3.87%</td>
<td>99.33±0.32%</td>
<td>95.34±3.15%</td>
<td>95.00±4.75%</td>
<td>64.74±0.90%</td>
</tr>
<tr>
<td>PRODEN-itera</td>
<td>0.1</td>
<td>57.55±3.96%</td>
<td>99.73±0.13%</td>
<td>95.56±2.90%</td>
<td>82.38±4.18%</td>
<td>77.04±0.77%</td>
</tr>
<tr>
<td></td>
<td>0.7</td>
<td>53.62±5.54%</td>
<td>94.79±1.63%</td>
<td>76.22±4.48%</td>
<td>48.33±7.96%</td>
<td>52.43±0.75%</td>
</tr>
<tr>
<td>GA</td>
<td>0.1</td>
<td>25.43±3.81%</td>
<td>96.77±0.51%</td>
<td>73.53±7.73%</td>
<td>64.70±0.42%</td>
<td>66.53±2.60%</td>
</tr>
<tr>
<td></td>
<td>0.7</td>
<td>22.59±2.51%</td>
<td>95.76±0.62%</td>
<td>52.00±12.32%</td>
<td>49.72±6.38%</td>
<td>56.47±0.59%</td>
</tr>
<tr>
<td>D²CNN</td>
<td>0.1</td>
<td>59.05±4.66%</td>
<td>98.80±0.31%</td>
<td>95.34±2.29%</td>
<td>84.50±11.28%</td>
<td>73.20±0.46%</td>
</tr>
<tr>
<td></td>
<td>0.7</td>
<td>55.15±3.87%</td>
<td>97.23±0.71%</td>
<td>91.59±1.08%</td>
<td>71.00±13.02%</td>
<td>52.13±0.41%</td>
</tr>
<tr>
<td>SURE</td>
<td>0.1</td>
<td>55.52±4.92%</td>
<td>97.96±0.32%</td>
<td>95.16±2.25%</td>
<td>76.67±2.83%</td>
<td>69.82±0.26%</td>
</tr>
<tr>
<td></td>
<td>0.7</td>
<td>49.32±3.95%</td>
<td>94.75±3.68%</td>
<td>90.96±2.02%</td>
<td>54.67±7.67%</td>
<td>62.66±1.21%</td>
</tr>
<tr>
<td>CLPL</td>
<td>0.1</td>
<td>57.90±4.35%</td>
<td>98.78±0.22%</td>
<td>94.79±3.27%</td>
<td>78.00±3.10%</td>
<td>76.26±0.77%</td>
</tr>
<tr>
<td></td>
<td>0.7</td>
<td>54.88±7.78%</td>
<td>95.09±1.01%</td>
<td>92.33±2.67%</td>
<td>63.33±7.02%</td>
<td>50.37±0.57%</td>
</tr>
<tr>
<td>ECOC</td>
<td>0.1</td>
<td>59.01±3.72%</td>
<td>99.15±0.27%</td>
<td>94.71±2.08%</td>
<td>96.67±2.08%</td>
<td>77.67±1.11%</td>
</tr>
<tr>
<td></td>
<td>0.7</td>
<td>53.37±2.37%</td>
<td>97.69±0.82%</td>
<td>91.47±3.02%</td>
<td>91.67±2.08%</td>
<td>61.32±2.45%</td>
</tr>
<tr>
<td>PLSVM</td>
<td>0.1</td>
<td>54.73±2.89%</td>
<td>93.75±1.99%</td>
<td>92.88±2.25%</td>
<td>92.50±2.12%</td>
<td>76.25±1.22%</td>
</tr>
<tr>
<td></td>
<td>0.7</td>
<td>42.34±1.92%</td>
<td>50.69±5.62%</td>
<td>89.32±4.98%</td>
<td>85.50±2.61%</td>
<td>59.12±0.41%</td>
</tr>
<tr>
<td>PL&amp;NN</td>
<td>0.1</td>
<td>54.28±3.64%</td>
<td>97.20±0.28%</td>
<td>94.52±2.74%</td>
<td>94.83±2.08%</td>
<td>42.39±0.97%</td>
</tr>
<tr>
<td></td>
<td>0.7</td>
<td>28.72±1.01%</td>
<td>96.78±3.18%</td>
<td>85.48±3.70%</td>
<td>85.50±6.36%</td>
<td>18.01±0.34%</td>
</tr>
<tr>
<td>IPAL</td>
<td>0.1</td>
<td>50.56±2.83%</td>
<td>99.33±0.26%</td>
<td>95.34±2.29%</td>
<td>98.33±0.83%</td>
<td>75.01±0.69%</td>
</tr>
<tr>
<td></td>
<td>0.7</td>
<td>41.10±3.71%</td>
<td>97.84±0.57%</td>
<td>95.02±2.03%</td>
<td>94.83±6.55%</td>
<td>56.93±0.50%</td>
</tr>
</tbody>
</table>

Table 2. Test accuracy (mean±std) on the real-world datasets.

<table>
<thead>
<tr>
<th></th>
<th>Lost</th>
<th>Birdsong</th>
<th>MSRCv2</th>
<th>Soccer Player</th>
<th>Yahoo! News</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRODEN</td>
<td>76.57±1.47%</td>
<td>72.01±0.44%</td>
<td>45.27±1.73%</td>
<td>55.99±0.58%</td>
<td>67.40±0.55%</td>
</tr>
<tr>
<td>PRODEN-itera</td>
<td>68.09±3.78%</td>
<td>68.02±0.76%</td>
<td>42.79±2.80%</td>
<td>53.50±0.94%</td>
<td>67.04±0.67%</td>
</tr>
<tr>
<td>D²CNN</td>
<td>69.61±5.48%</td>
<td>66.58±1.49%</td>
<td>40.17±1.99%</td>
<td>49.06±0.15%</td>
<td>56.39±0.89%</td>
</tr>
<tr>
<td>GA</td>
<td>48.21±4.44%</td>
<td>29.77±1.43%</td>
<td>22.30±2.71%</td>
<td>51.88±0.44%</td>
<td>34.32±0.95%</td>
</tr>
<tr>
<td>SURE</td>
<td>71.61±3.44%</td>
<td>58.04±1.22%</td>
<td>31.57±2.48%</td>
<td>49.16±0.20%</td>
<td>45.73±0.90%</td>
</tr>
<tr>
<td>CLPL</td>
<td>76.17±1.81%</td>
<td>67.56±1.12%</td>
<td>43.64±0.24%</td>
<td>49.88±4.29%</td>
<td>53.74±0.95%</td>
</tr>
<tr>
<td>ECOC</td>
<td>63.93±5.45%</td>
<td>71.47±1.24%</td>
<td>46.78±2.84%</td>
<td>55.51±0.54%</td>
<td>64.78±0.78%</td>
</tr>
<tr>
<td>PLSVM</td>
<td>72.86±5.45%</td>
<td>60.46±1.99%</td>
<td>38.97±4.62%</td>
<td>46.15±1.00%</td>
<td>60.46±1.48%</td>
</tr>
<tr>
<td>PL&amp;NN</td>
<td>35.00±4.71%</td>
<td>64.22±1.14%</td>
<td>41.60±2.30%</td>
<td>49.18±0.26%</td>
<td>40.30±0.90%</td>
</tr>
<tr>
<td>IPAL</td>
<td>71.25±1.40%</td>
<td>71.19±1.54%</td>
<td>52.36±2.87%</td>
<td>54.41±0.56%</td>
<td>66.22±0.80%</td>
</tr>
</tbody>
</table>

6. Conclusion

In this paper, we focused on proposing a novel method for PLL which is compatible with flexible multi-class classifiers and stochastic optimization. We first proposed a classifier-consistent risk estimator based on the minimal loss incurred by candidate labels. The estimator has good theoretical properties, while the min operator makes it difficult to be solved by the epoch-wise optimization of neural networks. Therefore, we proposed a progressive identification method named PRODEN for approximately minimizing the proposed risk estimator, whose idea is relaxing min to a weighted combination. Learning the weights and the classifier are then conducted in a seamless manner for mitigating the overfitting problem. At last, experiments demonstrated that the proposed method could successfully train various models, and it compared favorably with state-of-the-art methods.
Acknowledgements

JL and XG were supported by the National Key Research & Development Plan of China (No. 2017YFB1002801), the National Science Foundation of China (61622203), the Collaborative Innovation Center of Novel Software Technology and Industrialization, and the Collaborative Innovation Center of Wireless Communications Technology. MX was supported by the Open Project Program of the State Key Lab. for Novel Software Technology, Nanjing University, Nanjing, 210093, PR.China. LF was supported by MOE, NRF, and NTU, Singapore. GN and MS were supported by JST AIP Acceleration Research Grant Number JPMJCR20U3, Japan.

References


