Partial label learning (PLL) is a typical weakly supervised learning problem, where each training example is associated with a set of candidate labels among which only one is true. Most existing PLL approaches assume that the incorrect labels in each training example are randomly picked as the candidate labels. However, this assumption is not realistic since the candidate labels are always instance-dependent. In this paper, we consider instance-dependent PLL and assume that each example is associated with a latent label distribution constituted by the real number of each label, representing the degree to each label describing the feature. The incorrect label with a high degree is more likely to be annotated as the candidate label. Therefore, the latent label distribution is the essential labeling information in partially labeled examples and worth being leveraged for predictive model training. Motivated by this consideration, we propose a novel PLL method that recovers the label distribution as a label enhancement (LE) process and trains the predictive model iteratively in every epoch. Specifically, we assume the true posterior density of the latent label distribution takes on the variational approximate Dirichlet density parameterized by an inference model. Then the evidence lower bound is deduced for optimizing the inference model and the label distributions generated from the variational posterior are utilized for training the predictive model. Experiments on benchmark and real-world datasets validate the effectiveness of the proposed method. Source code is available at https://github.com/palm-ml/valen.

1 Introduction

Partial label learning (PLL) deals with the problem where each training example is associated with a set of candidate labels, among which only one label is valid [7, 5, 37]. Due to the difficulty in collecting exactly labeled data in many real-world scenarios, PLL leverages inexact supervision instead of exact labels. The need to learn from the inexact supervision leads to a wide range of applications for PLL techniques, such as web mining [25], multimedia content analysis [38, 4], ecoinformatics [24, 31], etc.

To accomplish the task of learning from partial label data, many approaches have been proposed. Identification-based PLL approaches [17, 23, 24, 5, 37] regard the ground-truth label as a latent variable and try to identify it. Average-based approaches [15, 7, 40] treat all the candidate labels equally and average the modeling outputs as the prediction. For confidence-based approaches [8, 34, 42], the confidence of each label is estimated instead of identifying the ground-truth label. These approaches always adopt the randomly picked candidate labels to corrupt benchmark data into partially labeled version despite having no explicit generation process of candidate label sets. To depict the instance-independent generation process of candidate label sets, Feng [9] proposes a statistical model and deduces a risk-consistent method and a classifier-consistent method. Under the
Figure 1: The examples about the latent label distributions for partial label learning. The candidate labels are in the box and the red one is valid.

same generation process, another classifier-consistent risk estimator is proposed for deep model and stochastic optimizers [26].

The previous methods assume that the candidate labels are randomly sampled with the uniform generating procedure [26, 9], which is commonly adopted to corrupt benchmark datasets into partially labeled versions in their experiments. However, the candidate labels are always instance-dependent (feature-dependent) in practice as the incorrect labels related to the feature are more likely to be picked as candidate label set for each instance. These methods usually do not perform as well as expected due to the unrealistic assumption on the generating procedure of candidate label sets.

In this paper, we consider instance-dependent PLL and assume that each instance in PLL is associated with a latent label distribution [33, 35, 11] constituted by the real number of each label, representing the degree to each label describing the feature. Then, the incorrect label with a high degree in the latent label distribution is more likely to be annotated as the candidate label. For example, the candidate label set of the handwritten digits in Figure 1(a) contains “1”, “3” and “5”, where “1” and “3” are not ground-truth but selected as candidate labels due to their high degrees in the latent label distribution of the instance. The object in Figure 1(b) is annotated with “bird” and “airplane” as the degrees of these two labels are much higher than others in the label distribution. The intrinsical ambiguity increases the difficulty of annotating, which leads to the result that annotators pick the candidate labels with high degrees in the latent label distribution of each instance instead of annotating the ground-truth label directly in PLL. Therefore, the latent label distribution is the essential labeling information in partially labeled examples and worth being leveraged for predictive model training.

Motivated by the above consideration, we deal with the PLL problem from two aspects. First, we enhance the labeling information by recovering the latent label distribution for each training example as a label enhancement process [33, 35]. Second, we run label enhancement and train the predictive model with recovered label distributions iteratively. The proposed method named VALEN, i.e., Variational Label Enhancement for instance-dependent partial label learning, uses the candidate labels to initialize the predictive model in the warm-up training stage, then recovers the latent label distributions via inferring the variational posterior density parameterized by an inference model with the deduced evidence lower bound, and trains the predictive model with a risk estimator by leveraging the candidate labels as well as the label distributions. Our contributions can be summarized as follows:

- We for the first time consider the instance-dependent PLL and assume that each partially labeled example is associated with a latent label distribution, which is the essential labeling information and worth being recovered for predictive model training.
- We infer the posterior density of the latent label distribution via taking on the approximate Dirichlet density parameterized by an inference model and deduce the evidence lower bound for optimization, in which the topological information and the features extracted from the predictive model are leveraged.
- We train predictive model with a proposed empirical risk estimator by leveraging the candidate labels as well as the label distributions. We iteratively recover the latent label distributions and train the predictive model in every epoch. After the network has been fully trained, the predictive model can perform predictions for future test examples alone.

Experiments on the corrupted benchmark datasets and real-world PLL datasets validate the effectiveness of the proposed method.
2 Proposed Method

First of all, we briefly introduce some necessary notations. Let $\mathcal{X} = \mathbb{R}^q$ be the $q$-dimensional instance space and $\mathcal{Y} = \{y_1, y_2, \ldots, y_c\}$ be the label space with $c$ class labels. Given the PLL training set $\mathcal{D} = \{(x_i, S_i) | 1 \leq i \leq n\}$ where $x_i$ denotes the $q$-dimensional instance and $S_i \subseteq \mathcal{Y}$ denotes the candidate label set associated with $x_i$. Note that $S_i$ contains the correct label of $x_i$ and the task of PLL is to induce a multi-class classifier $f : \mathcal{X} \mapsto \mathcal{Y}$ from $\mathcal{D}$. For each PLL training example $(x_i, S_i)$, we use the logical label vector $l_i = [l_i^0, l_i^1, \ldots, l_i^c]^\top \in \{0, 1\}^c$ to represent whether $y_j$ is the candidate label, i.e., $l_i^j = 1$ if $y_j \in S_i$, otherwise $l_i^j = 0$. The label distribution of $x_i$ is denoted by $d_i = [d_i^0, d_i^1, \ldots, d_i^c]^\top \in [0, 1]^c$ where $\sum_{j=1}^c d_i^j = 1$. Then $L = [l_1, l_2, \ldots, l_n]$ and $D = [d_1, d_2, \ldots, d_n]$ represent the logical label matrix and label distribution matrix, respectively.

2.1 Overview

To deal with PLL problem, we iteratively recover the latent label distribution for each example $x$ and train the predictive model by leveraging the recovered label distribution. We start with a warm-up period, in which we train the predictive model with the PLL minimal loss [26]. This allows us to attain a reasonable predictive model before it starts fitting incorrect labels. After the warm-up period, the features extracted from the predictive model can help for recovering the latent label distribution. Benefited from the essential labeling information in the recovered label distribution, the performance of the predictive model could be further improved.

VALEN implements label enhancement and classifier training iteratively in every epoch. In label enhancement, we assume the true posterior density of the latent label distribution takes on the variational approximate Dirichlet density parameterized by an inference model. Then the evidence lower bound is deduced for optimizing the inference model and the label distributions can be generated from the variational posterior. In classifier training, the predictive model is trained by leveraging the recovered label distributions and candidate labels with an empirical risk estimator. After the models has been fully trained, the predictive model can perform prediction for future test instances alone.

2.2 Warm-up Training

The predictive model $\theta$ is trained on partially labeled examples by minimizing the following PLL minimal loss function [26]:

$$\mathcal{L}_{\text{min}} = \sum_{i=1}^n \min_{y_j \in S_i} \ell(f(x_i), e^{y_j}),$$

where $\ell$ is cross-entropy loss and $e^{y_j} = \{e^{y_j} : y_j \in \mathcal{Y}\}$ denotes the standard canonical vector in $\mathbb{R}^c$, i.e., the $j$-element in $e^{y_j}$ equals 1 and others equal 0. Similar to [26], the $\min$ operator in Eq. (1) is replaced by using the current predictions for slightly weighting on the possible labels in warm-up training. Then we could extract the feature $\phi$ of each $x$ via using the predictive model.

2.3 Label Enhancement

We assume that the prior density $p(d)$ is a Dirichlet with $\hat{\alpha}$, i.e., $p(d) = \text{Dir}(d | \hat{\alpha})$ where $\hat{\alpha} = [\varepsilon, \varepsilon, \ldots, \varepsilon]^\top$ is a $c$-dimensional vector with a minor value $\varepsilon$. Then we let the prior density $p(D)$ be the product of each Dirichlet

$$p(D) = \prod_{i=1}^n \text{Dir}(d_i | \hat{\alpha}).$$

We consider the topological information of the feature space, which is represented by the affinity graph $G = (V, E, A)$. Here, the feature vector $\phi_i$ of each example could be extracted from the predictive model $\theta$ in current epoch, $V = \{\phi_i | 1 \leq i \leq n\}$ corresponds to the vertex set consisting of feature vectors, $E = \{\{\phi_i, \phi_j\} | 1 \leq i \neq j \leq n\}$ corresponds to the edge set, and a sparse adjacency matrix $A = [a_{ij}]_{n \times n}$ can be obtained by

$$a_{ij} = \begin{cases} 1 & \text{if } \phi_i \in \mathcal{N}(\phi_j) \\ 0 & \text{otherwise} \end{cases},$$
where $\mathcal{N}(\phi_j)$ is the set for $k$-nearest neighbors of $\phi_j$ and the diagonal elements of $A$ are set to 1.

Let features matrix $\Phi = [\phi_1, \phi_2, \ldots, \phi_n]$, adjacency matrix $A$ and logical labels $L$ be observed matrix. VALEN aims to infer the posterior density $p(D|L, \Phi, A)$. As the computation of the exact posterior density $p(D|L, \Phi, A)$ is intractable, a fixed-form density $\eta(D|L, \Phi, A)$ is employed to approximate the true posterior. We let the approximate posterior be the product of each Dirichlet parameterized by a vector $\alpha_i = [\alpha_i^1, \alpha_i^2, \ldots, \alpha_i^n]$:

$$q_w(D|L, \Phi, A) = \prod_{i=1}^{n} Dir(d_i|\alpha_i).$$

Here, the parameters $\Delta = [\alpha_1, \alpha_2, \ldots, \alpha_n]$ are outputs of the inference model parameterized by $w$, which is defined as a two-layer GCN [20] by GCN($L, \Phi, A$) = $\hat{A}$ ReLU ($\hat{A}ZW_0$) $W_1$, with $Z = [\Phi; L]$ and weight $W_0, W_1$. Here $\hat{A} = \hat{A}^{-\frac{1}{2}}A\hat{A}^{-\frac{1}{2}}$ is the symmetrically normalized weight matrix where $\hat{A}$ is the degree matrix of $A$.

By following the Variational Bayes techniques, a lower bound on the marginal likelihood of the model is derived which ensures that $q_w(D|L, \Phi, A)$ is as close as possible to $p(D|L, \Phi, A)$. For logical label matrix $L$, feature matrix $\Phi$, and the corresponding $A$, the log marginal probability can be decomposed as follows:

$$\log p(L, \Phi, A) = \mathcal{L}_{ELBO} + KL[q_w(D|L, \Phi, A)||p(D|L, \Phi, A)].$$

where

$$\mathcal{L}_{ELBO} = \mathbb{E}_{q_w(D|L, \Phi, A)}[\log p(L, \Phi, A|D)] - KL[q_w(D|L, \Phi, A)||p(D)].$$

Due to the non-negative property of KL divergence, the first term $\mathcal{L}_{ELBO}$ constitutes a lower bound of $\log p(L, \Phi, A)$, which is often called as evidence lower bound (ELBO), i.e., $\log p(L, \Phi, A) \geq \mathcal{L}_{ELBO}$.

According to Eq. (2) and Eq. (4), the KL divergence in Eq. (6) can be analytically calculated as follows:

$$KL(q_w(D|L, \Phi, A)||p(D)) = \sum_{i=1}^{n} \left( \log \Gamma \left( \sum_{j=1}^{c} \alpha_i^j \right) - \sum_{j=1}^{c} \log \Gamma \left( \alpha_i^j \right) \right)
- \log \Gamma (c \cdot \varepsilon) + c \log \Gamma (\varepsilon) + \sum_{j=1}^{c} \left( \alpha_i^j - \varepsilon \right) \left( \psi \left( \alpha_i^j \right) - \psi \left( \sum_{j=1}^{c} \alpha_i^j \right) \right),$$

where $\Gamma(\cdot)$ and $\psi(\cdot)$ are Gamma function and Digamma function, respectively.

As the first part of Eq. (6) is intractable, we employ the implicit reparameterization trick [10] to approximate it by Monte Carlo (MC) estimation. Inspired by [20], we simply drop the dependence on $\Phi$:

$$p(L|A, D) = \prod_{i=1}^{n} p(l_i|A, D),$$

$$p(A|D) = \prod_{i=1}^{n} \prod_{j=1}^{n} p(a_{ij}|d_i, d_j), \text{ with } p(a_{ij} = 1|d_i, d_j) = s(d_i^T d_j).$$

Here, $s(\cdot)$ is the logistic sigmoid function. We further assume that $p(l_i|A, D)$ is a multivariate Bernoulli with probabilities $\tau_i$. In order to simplify the observation model, $T^{(m)} = [T^{(m)}_1, T^{(m)}_2, \ldots, T^{(m)}_n]$ is computed from $m$-th sampling $D^{(m)}$ with a three-layer MLP parameterized by $\eta$. Then the first part of Eq. (6) can be tractable:

$$\mathbb{E}_{q_w(D|L, \Phi, A)}[\log p_\eta(L, \Phi, A|D)] = \frac{1}{M} \sum_{m=1}^{M} \left( \text{tr} \left( (I - L)^T \log (I - T^{(m)}) \right) + \text{tr} \left( L^T \log T^{(m)} \right) - \|A - S(D^{(m)}D^{(m)^T})\|_F^2 \right).$$

More detailed calculations can be seen in Appendix A.1.
Algorithm 1 VALEN Algorithm

Input: The PLL training set \( D = \{(x_i, S_i)\}_{i=1}^n \), epoch \( T \) and iteration \( I \):
1: Initialize the predictive model \( \theta \) by warm-up training, the reference model \( w \) and observation model \( \eta \);
2: Extract the features \( \Phi \) from predictive model \( \theta \) and calculate the adjacency matrix \( A \);
3: for \( t = 1, \ldots, T \) do
4: Shuffle training set \( D = \{(x_i, S_i)\}_{i=1}^n \) into \( I \) mini-batches;
5: for \( k = 1, \ldots, I \) do
6: Obtain label distribution \( d_i \), for each example \( x_i \) by Eq. (4);
7: Update \( \theta, w \) and \( \eta \) by forward computation and back-propagation by fusing Eq. (12) and Eq. (13);
8: end for
9: end for

Output: The predictive model \( \theta \).

Note that we can use only one MC sample in Eq. (9) during the training process as suggested in [19, 35].

In addition, VALEN improves the label enhancement by employing the compatibility loss, which enforces that the recovered label distributions should not be completely different from the confidence \( \zeta(x_i) \) [9, 26] estimated by current prediction \( f(x_i; \theta) \):

\[
\mathcal{L}_\zeta = -\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{c} \zeta_j(x_i) \log d_i^{y_j}
\]  

where

\[
\zeta_j(x_i) = \begin{cases} 
  f_j(x; \theta) / \sum_{j=1}^{m} f_k(x; \theta) & \text{if } y_j \in S_i \\
  0 & \text{otherwise} 
\end{cases}
\]  

Now we can easily get the objective of label enhancement \( \mathcal{L}_{LE} \) as follows:

\[
\mathcal{L}_{LE} = \lambda \mathcal{L}_\zeta - \mathcal{L}_{ELBO}
\]  

where \( \lambda \) is a hyper-parameter. The label distribution matrix \( D \) is sampled from \( q(D|L, \Phi, A) \), i.e., \( d_i \sim \text{Dir}(\alpha_i) \). Note that the implicit reparameterization gradient [10] is employed, which avoids the inversion of the standardization function, which makes the gradients can be computed analytically in backward pass.

2.4 Classifier Training

To train the predictive model, we minimize the following empirical risk estimator by levering the recovered label distributions:

\[
\hat{R}_\psi(f) = \frac{1}{n} \sum_{i=1}^{n} \left( \frac{1}{\sum_{y_j \in S_i} d_i^{y_j}} \sum_{y_j \in S_i} d_i^{y_j} \ell(f(x_i), e^{y_j}) \right).
\]  

Here we adopt the average value of \( d_i \) sampled by \( d_i \sim \text{Dir}(\alpha_i) \). We can use any deep neural network as the predictive model, and then equip it with the VALEN framework to deal with PLL. Note that we could train the predictive model and update the label distributions in a principled end-to-end manner by fusing the objective Eq. (12) and Eq. (13). The algorithmic description of the VALEN is shown in Algorithm 1.

Let \( \bar{f}_\psi = \min_{f \in \mathcal{F}} \hat{R}_\psi(f) \) be the empirical risk minimizer and \( f^* = \min_{f \in \mathcal{F}} R_\psi(f) \) be the optimal risk minimizer where \( R_\psi(f) \) is the risk estimator. Besides, we define the function space \( \mathcal{H}_{y_j} \) for the label \( y_j \in Y \) as \( \{ h : x \mapsto f_{y_j}(x) \mid f \in \mathcal{F} \} \). Let \( R_n(\mathcal{H}_{y_j}) \) be the expected Rademacher complexity [2] of \( \mathcal{H}_{y_j} \) with sample size \( n \), then we have the following theorem.

Theorem 1 Assume the loss function \( \ell(f(x), e^{y_j}) \) is \( L \)-Lipschitz with respect to \( f(x) \) (\( 0 < L < \infty \)) for all \( y_j \in Y \) and upper-bounded by \( M \), i.e., \( M = \sup_{x \in X, f \in \mathcal{F}, y_j \in Y} \ell(f(x), e^{y_j}) \). Then, for any
\( \delta > 0, \text{ with probability at least } 1 - \delta, \)

\[
R \left( \hat{f}_V \right) - R (f^*) \leq 4\sqrt{2L} \sum_{j=1}^{c} \mathfrak{R}_n \left( \mathcal{H}_{y_j} \right) + M \sqrt{\log \frac{2}{\delta} \frac{2}{2n}}
\]

The proof of Theorem 1 is provided in Appendix A.2. Theorem 1 shows that the empirical risk minimizer \( f_V \) converges to the optimal risk minimizer \( f^* \) as \( n \to \infty \) and \( \mathfrak{R}_n \left( \mathcal{H}_{y_j} \right) \to 0 \) for all parametric models with a bounded norm.

### 3 Related Work

As shown in Section 1, supervision information conveyed by partially labeled training examples is implicit as the ground-truth label is hidden within the candidate label set. Therefore, partial label learning can be regarded as a weak supervision learning framework \([18]\) with implicit labeling information. Intuitively, the basic strategy for handling partial label learning is disambiguation, i.e., trying to identify the ground-truth label from the candidate label set associated with each training example, where existing strategies include disambiguation by identification or disambiguation by averaging. For identification-based disambiguation, the ground-truth label is regarded as a latent variable and identified \([17, 28, 24, 5, 37]\). For averaging-based disambiguation, all the candidate labels are treated equally and the prediction is made by averaging their modeling outputs \([15, 7, 40]\).

Most existing algorithms aim to fulfill the learning task by fitting widely-used learning techniques to partial label data. For maximum likelihood techniques, the likelihood of observing each partially labeled training example is defined over its candidate label set instead of the unknown ground-truth label \([17, 24]\). \( K \)-nearest neighbor techniques determine the class label of unseen instances via voting among the candidate labels of its neighboring examples \([13, 40]\). For maximum margin techniques, the classification margins over the partially labeled training examples are defined by discriminating modeling outputs from candidate labels and non-candidate labels \([28, 37]\). For boosting techniques, the weight over each partially labeled training example and the confidence over the candidate labels are updated in each boosting round \([31]\). For disambiguation-free strategies, the generalized description degree is estimated by using a graph Laplacian and induce a multi-output regression \([34]\). The confidence of each candidate label is estimated by using the manifold structure of feature space \([42]\). However, these methods just estimate the soft labeling information and train the predictive models in separate stages without considering the feedback of the predictive models.

The above-mentioned works were solved in specific low-efficiency manners and incompatible with high-efficient stochastic optimization. To handle large-scale datasets, the deep networks are employed with an entropy-based regularizer to maximize the margin between the potentially correct label and the unlikely ones \([36, 26]\) proposes a classifier-consistent risk estimator and a progressive identification, which is compatible with deep models and stochastic optimizers. \([9]\) proposes a statistical model to depict the generation process of candidate label sets, which deduces a risk-consistent method and a classifier-consistent method.

The previous methods assume that the candidate labels are randomly sampled with the uniform generating procedure. However, the candidate labels are always instance-dependent (feature-dependent) in practice as the incorrect labels related to the feature are more likely to be picked as candidate label set for each instance. In this paper, we consider instance-dependent PLL and assume that each instance in PLL is associated with a latent label distribution \([33, 35, 11]\) constituted by the real number of each label, representing the degree to each label describing the feature. Label enhancement (LE) \([12, 33, 35]\) recovers the latent label distribution from the observed logical labels, in which the recovered label distribution is a kind of pseudo label \([29, 23]\), actually. Note that the generation process of soft labels in label smoothing \([30, 27]\) and distillation \([14, 89]\) could also be regarded as a label enhancement process.
We run 5 trials on the four benchmark datasets and perform five-fold cross-validation on UCI datasets. We adopt four widely used benchmark datasets including MNIST, A progressive identification partial label learning approach which approximately minimizes a risk estimator and identifies the true labels in a seamless manner; 2) RC [9]: A risk-consistent partial label learning approach which employs the importance reweighting strategy to converges the true risk.

### Table 1: Classification accuracy (mean±std) of each comparing approach on benchmark datasets corrupted by the instance-dependent generating procedure.

<table>
<thead>
<tr>
<th>Method</th>
<th>MNIST</th>
<th>Kuzushiji-MNIST</th>
<th>Fashion-MNIST</th>
<th>CIFAR-10</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALEN</td>
<td>97.85±0.05%</td>
<td>86.19±0.14%</td>
<td>86.17±0.19%</td>
<td>80.38±0.52%</td>
</tr>
<tr>
<td>PRODEN</td>
<td>97.69±0.04%</td>
<td>85.71±0.12%</td>
<td>85.54±0.09%</td>
<td>79.80±0.28%</td>
</tr>
<tr>
<td>RC</td>
<td>97.60±0.05%</td>
<td>84.86±0.11%</td>
<td>85.51±0.10%</td>
<td>79.46±0.25%</td>
</tr>
<tr>
<td>CC</td>
<td>97.44±0.03%</td>
<td>82.67±1.82%</td>
<td>85.19±0.04%</td>
<td>78.98±0.60%</td>
</tr>
<tr>
<td>D2CNN</td>
<td>94.63±0.16%</td>
<td>83.03±0.78%</td>
<td>82.42±0.21%</td>
<td>73.11±0.11%</td>
</tr>
<tr>
<td>GA</td>
<td>95.25±0.07%</td>
<td>82.45±0.63%</td>
<td>80.41±0.24%</td>
<td>77.57±0.76%</td>
</tr>
</tbody>
</table>

### Table 2: Classification accuracy (mean±std) of each comparing approach on benchmark datasets corrupted by the uniform generating procedure.

<table>
<thead>
<tr>
<th>Method</th>
<th>MNIST</th>
<th>Kuzushiji-MNIST</th>
<th>Fashion-MNIST</th>
<th>CIFAR-10</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALEN</td>
<td>97.93±0.05%</td>
<td>88.76±0.26%</td>
<td>88.98±0.16%</td>
<td>81.93±1.01%</td>
</tr>
<tr>
<td>PRODEN</td>
<td>97.97±0.03%</td>
<td>88.55±0.10%</td>
<td>88.94±0.12%</td>
<td>81.53±0.53%</td>
</tr>
<tr>
<td>RC</td>
<td>97.86±0.03%</td>
<td>86.65±0.10%</td>
<td>88.59±0.08%</td>
<td>81.30±1.30%</td>
</tr>
<tr>
<td>CC</td>
<td>97.73±0.02%</td>
<td>87.99±0.03%</td>
<td>88.93±0.06%</td>
<td>80.17±1.09%</td>
</tr>
<tr>
<td>D2CNN</td>
<td>95.12±0.16%</td>
<td>84.03±0.78%</td>
<td>80.42±0.21%</td>
<td>75.11±0.11%</td>
</tr>
<tr>
<td>GA</td>
<td>96.29±0.19%</td>
<td>82.36±0.98%</td>
<td>81.81±0.99%</td>
<td>60.14±1.35%</td>
</tr>
</tbody>
</table>

### 4 Experiments

#### 4.1 Datasets

We adopt four widely used benchmark datasets including MNIST [22], Fashion-MNIST [32], Kuzushiji-MNIST [6], and CIFAR-10 [21], and five datasets from the UCI Machine Learning Repository [1], including Yeast, Texture, Dermatology, Synthetic Control, and 20Newgroups.

We manually corrupt these datasets into partially labeled versions by using a flipping probability \( \xi_j(x) = P(\tilde{y}_j = 1 | \tilde{y}_j = 0, x_i) \), where \( \tilde{y}_j \) is the original clean label. To synthesize the instance-dependent candidate labels, we set the flipping probability of each incorrect label corresponding to an example \( x_i \) by using the confidence prediction of a clean neural network \( \hat{\theta} \) (trained with the original clean labels) [43] with \( \xi_j(x_i) = \max_{\hat{y}_j, \hat{y}_j(x_i; \hat{\theta})} f_j(x_i; \hat{\theta}) \), where \( \hat{y}_j \) is the incorrect label set of \( x_i \). The uniform corrupted version adopts the uniform generating procedure [26, 9] to flip the incorrect label into candidate label, where \( \xi_l(x_i) = \frac{1}{2} \).

In addition, five real-world PLL datasets are adopted, which are collected from several application domains including Lost [7], Soccer Player [38] and Yahoo! News [13] for automatic face naming from images or videos, MSRCv2 [24] for object classification, and BirdSong [3] for bird song classification. The detailed descriptions of these datasets are provided in Appendix A.3.

We run 5 trials on the four benchmark datasets and perform five-fold cross-validation on UCI datasets and real-world PLL datasets. The mean accuracy as well as standard deviation are recorded for all comparing approaches.

#### 4.2 Baselines

The performance of VALEN is compared against five DNN based approaches: 1) PRODEN [26]: A progressive identification partial label learning approach which approximately minimizes a risk estimator and identifies the true labels in a seamless manner; 2) RC [9]: A risk-consistent partial label learning approach which employs the importance reweighting strategy to converges the true risk.

---

[2] The datasets corrupted by the instance-dependent generating procedure are available at [https://drive.google.com/drive/folders/1J_68EqGrN6tA56RcyJr1komJ83J1?usp=sharing](https://drive.google.com/drive/folders/1J_68EqGrN6tA56RcyJr1komJ83J1?usp=sharing)
Table 3: Classification accuracy (mean±std) of each comparing approach on UCI datasets corrupted by the instance-dependent generating procedure.

<table>
<thead>
<tr>
<th></th>
<th>Yeast</th>
<th>Texture</th>
<th>Synthetic Control</th>
<th>Dermatology</th>
<th>20Newsgroup</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VALEN</strong></td>
<td><strong>57.57±1.08%</strong></td>
<td><strong>94.76±0.93%</strong></td>
<td><strong>82.86±0.76%</strong></td>
<td><strong>89.86±1.31%</strong></td>
<td><strong>81.88±0.47%</strong></td>
</tr>
<tr>
<td><strong>PRODEN</strong></td>
<td>54.78±1.28%</td>
<td>89.87±2.14%</td>
<td>71.16±6.19%</td>
<td>88.53±3.87%</td>
<td>78.06±0.74%</td>
</tr>
<tr>
<td><strong>RC</strong></td>
<td>54.77±1.27%</td>
<td>89.57±2.37%</td>
<td>65.99±2.72%</td>
<td>88.53±3.49%</td>
<td>78.02±0.79%</td>
</tr>
<tr>
<td><strong>CC</strong></td>
<td>54.98±0.91%</td>
<td>88.92±5.56%</td>
<td>66.99±2.47%</td>
<td>88.26±4.11%</td>
<td>77.88±0.39%</td>
</tr>
<tr>
<td><strong>D2CNN</strong></td>
<td>44.94±1.87%</td>
<td>69.52±6.39%</td>
<td>62.66±8.92%</td>
<td>81.95±6.18%</td>
<td>73.55±0.92%</td>
</tr>
<tr>
<td><strong>GA</strong></td>
<td>25.86±3.17%</td>
<td>74.84±2.87%</td>
<td>56.43±1.29%</td>
<td>84.85±1.43%</td>
<td>49.49±3.42%</td>
</tr>
<tr>
<td><strong>CLPL</strong></td>
<td>54.92±2.38%</td>
<td>81.27±9.09%</td>
<td>66.33±3.25%</td>
<td>92.07±3.42%</td>
<td>77.62±0.23%</td>
</tr>
<tr>
<td><strong>PL-SVM</strong></td>
<td>41.85±5.92%</td>
<td>39.03±4.35%</td>
<td>50.33±5.73%</td>
<td>84.98±4.56%</td>
<td>72.89±0.41%</td>
</tr>
<tr>
<td><strong>PL-KNN</strong></td>
<td>47.44±2.69%</td>
<td>70.05±0.70%</td>
<td>80.50±1.26%</td>
<td>83.61±3.15%</td>
<td>33.28±1.09%</td>
</tr>
<tr>
<td><strong>IPAL</strong></td>
<td>56.40±2.07%</td>
<td>93.49±0.89%</td>
<td>77.66±3.60%</td>
<td>78.94±3.84%</td>
<td>67.38±0.95%</td>
</tr>
<tr>
<td><strong>PPL</strong></td>
<td>55.53±1.74%</td>
<td>84.45±1.07%</td>
<td>66.16±7.96%</td>
<td>93.16±2.58%</td>
<td>75.54±0.66%</td>
</tr>
</tbody>
</table>

Table 4: Classification accuracy (mean±std) of each comparing approach on UCI datasets corrupted by the uniform generating procedure.

<table>
<thead>
<tr>
<th></th>
<th>Yeast</th>
<th>Texture</th>
<th>Synthetic Control</th>
<th>Dermatology</th>
<th>20Newsgroup</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VALEN</strong></td>
<td><strong>58.18±1.46%</strong></td>
<td><strong>97.30±5.57%</strong></td>
<td><strong>97.17±0.47%</strong></td>
<td><strong>97.07±0.41%</strong></td>
<td><strong>71.75±0.32%</strong></td>
</tr>
<tr>
<td><strong>PRODEN</strong></td>
<td>56.32±1.98%</td>
<td>97.75±0.53%</td>
<td>95.83±1.95%</td>
<td>95.07±1.84%</td>
<td>68.28±0.91%</td>
</tr>
<tr>
<td><strong>RC</strong></td>
<td>56.39±1.85%</td>
<td>97.77±0.55%</td>
<td>95.99±1.80%</td>
<td>95.62±1.51%</td>
<td>68.44±1.09%</td>
</tr>
<tr>
<td><strong>CC</strong></td>
<td>56.25±1.89%</td>
<td>97.79±0.57%</td>
<td>96.33±1.39%</td>
<td>95.90±1.69%</td>
<td>67.95±0.95%</td>
</tr>
<tr>
<td><strong>D2CNN</strong></td>
<td>54.04±1.90%</td>
<td>97.23±0.72%</td>
<td>81.16±8.11%</td>
<td>90.43±2.38%</td>
<td>65.88±2.56%</td>
</tr>
<tr>
<td><strong>GA</strong></td>
<td>22.98±2.57%</td>
<td>95.09±1.07%</td>
<td>56.87±1.53%</td>
<td>51.95±3.89%</td>
<td>58.29±1.74%</td>
</tr>
<tr>
<td><strong>CLPL</strong></td>
<td>56.54±3.35%</td>
<td>98.14±0.59%</td>
<td>94.66±6.41%</td>
<td>96.72±0.76%</td>
<td>70.45±0.91%</td>
</tr>
<tr>
<td><strong>PL-SVM</strong></td>
<td>46.23±7.21%</td>
<td>39.74±2.11%</td>
<td>76.50±5.31%</td>
<td>92.37±5.08%</td>
<td>70.44±0.37%</td>
</tr>
<tr>
<td><strong>PL-KNN</strong></td>
<td>44.40±2.50%</td>
<td>95.31±0.85%</td>
<td>95.33±2.98%</td>
<td>92.91±2.92%</td>
<td>27.10±0.49%</td>
</tr>
<tr>
<td><strong>IPAL</strong></td>
<td>43.86±3.39%</td>
<td><strong>98.71±0.37%</strong></td>
<td>96.83±1.90%</td>
<td>95.35±2.08%</td>
<td>65.39±1.21%</td>
</tr>
<tr>
<td><strong>PPL</strong></td>
<td>53.58±2.86%</td>
<td>98.40±0.40%</td>
<td>89.66±1.91%</td>
<td>90.98±1.85%</td>
<td>53.88±0.59%</td>
</tr>
</tbody>
</table>

For all the DNN based approaches, we adopt the same predictive model for fair comparisons. Specifically, the 32-layer ResNet is trained on CIFAR-10 in which the learning rate, weight decay and mini-batch size are set to $0.05, 10^{-3}$ and 256, respectively. The three-layer MLP is trained on MNIST, Fashion-MNIST and Kuzushiji-MNIST where the learning rate, weight decay and mini-batch size are set to $10^{-2}, 10^{-4}$ and 256, respectively. The linear model is trained on UCI and real-world PLL datasets where the learning rate, weight decay and mini-batch size are set to $10^{-2}, 10^{-4}$ and 100, respectively. We implement the comparing methods with PyTorch. The number of epochs is set to 500, in which the first 10 epochs are warm-up training. We also want to use VALEN on MindSpore\(^3\) which is a new deep learning computing framework. These problems are left for future work.

In addition, we also compare with five classical partial label learning approaches, each configured with parameters suggested in respective literatures: 1) CLPL \(^7\): A convex partial label learning approach which uses averaging-based disambiguation; 2) PL-KNN \(^15\): An instance-based partial label learning approach which works by $k$-nearest neighbor weighted voting; 3) PL-SVM \(^28\): A maximum margin partial label learning approach which works by identification-based disambiguation; 4) IPAL \(^41\): A non-parametric method that applies the label propagation strategy to iteratively update the confidence of each candidate label; 5) PPL \(^34\): A two-stage partial label learning approach

\(^3\)https://www.mindspore.cn/
which estimates the generalized description degree of each class label values via graph Laplacian and induces a multi-label predictive model with the generalized description degree in separate stages.

4.3 Experimental Results

Table 1 reports the classification accuracy of each DNN-based method on benchmark datasets corrupted by the instance-dependent generating procedure and the uniform generating procedure, respectively. VALEN always achieves the best performance and significantly outperforms other compared methods in most cases. In addition, we also validate the effectiveness of our approach on uniform corrupted versions that is commonly adopted in previous works. From Table 2, we can observe that VALEN achieves superior or at least comparable performance to other approaches on uniform corrupted versions.

Table 3 and Table 4 report the classification accuracy of each method on UCI datasets corrupted by the instance-dependent generating procedure and the uniform generating procedure, respectively. VALEN always achieves the best performance and significantly outperforms other DNN-based methods in most cases on instance-dependent corrupted versions while achieves superior or at least comparable performance to other approaches on uniform corrupted versions. We further compare VALEN with five classical PLL methods that can hardly be implemented by DNNs on large-scale datasets. Despite the small scale of most UCI datasets, VALEN always achieve the best performance in most cases against the classical PLL methods as VALEN can deal with the high average number of candidate labels (can be seen in Appendix A.3) in the corrupted UCI datasets.

Table 5 reports the experimental results on real-world PLL datasets. We can find that VALEN achieves best performance against other DNN-based methods on the real-world PLL datasets. Note that VALEN achieves best performance against classical methods on all datasets except Lost and MSRCv2 as these datasets are small-scale and the average number of candidate labels in each dataset is low (can be seen in Appendix A.3), which leads to the result that DNN-based methods cannot take full advantage.

Figure 2(a) and Figure 2(b) illustrate the performance of VALEN on KMNIST corrupted by the instance-dependent generating procedure and the uniform generating procedure under different flipping probability, respectively. Besides, the performance of the ablation version that removes the label enhancement and trains the predictive model with PLL minimal loss (denoted by VALEN-NON) is recorded. These results clearly validate the usefulness of recovered label distributions for improving predictive performance. Figure 2(c) illustrates the recovered label distribution matrix over all training examples converges as the number of epoch (after warm-up training) on Kuzushiji-MNIST. We can see that the recovered label distributions converge fast with the increasing number of epoch.

5 Conclusion

In this paper, the problem of partial label learning is studied where a novel approach VALEN is proposed. We for the first time consider the instance-dependent PLL and assume that each
partially labeled example is associated with a latent label distribution, which is the essential labeling information and worth being recovered for predictive model training. VALEN recovers the latent label distribution via inferring the true posterior density of the latent label distribution by Dirichlet density parameterized with an inference model and deduce the evidence lower bound for optimization. In addition, VALEN iteratively recovers latent label distributions and trains the predictive model in every epoch. The effectiveness of the proposed approach is validated via comprehensive experiments on both synthesis datasets and real-world PLL datasets.

6 Acknowledgments

This research was supported by the National Key Research & Development Plan of China (No. 2018AAA0100104, No. 2018AAA0100100), the National Science Foundation of China (62125602, 62076063), Jiangsu Province Science Foundation for Youths (BK20210220).

References


