Partial Label Learning via Generative Adversarial Nets

Yabin Zhang\textsuperscript{1,3} and Guang Yang\textsuperscript{1} and Suyun Zhao\textsuperscript{1,2} and Peng Ni\textsuperscript{1} and Hairong Lian\textsuperscript{3} and Hong Chen\textsuperscript{1} and Cuiping Li\textsuperscript{1}

Abstract. Partial label learning (PLL) is a weakly supervised learning framework, in which each sample is provided with multiple candidate labels while only one of them is correct. Most of the existing methods are designed based on some conventional machine learning techniques, from kNN to SVM and logistic regression. Till now, it is still unclear whether we can use adversarial networks to solve partial label problems. This paper gives a positive answer to this question for the first time. We are the first to solve partial label learning with the network structure of CGAN combine SSGAN. In partial label learning with adversarial networks, it is interesting to find that some fake samples close to real sample distribution are generated, and then all these samples gradually promote discriminator to disambiguate the candidate labels of real samples. We give theoretical justifications of PL-GAN on challenging partial label data classification. Numerical experiments on artificial and real-world partial label datasets show that our approach significantly outperforms state-of-the-art counterparts.

1 INTRODUCTION

Partial label (PL) learning, also called superset label learning or ambiguous label learning [14, 5, 8], deals with the problem that each sample is provided with a set of candidate labels, only one of which is the ground-truth label. For example, in Figure 1, annotators may roughly assign a set of candidate labels for the picture, but only one is the ground-truth label. In recent years, partial label learning techniques have been used in many real-world scenarios, such as web mining [16], ecoinformatics [15, 22], automatic face naming [27, 15, 22].

The set of candidate labels

\begin{itemize}
  \item lion \checkmark
  \item tiger \xmark
  \item leopard \xmark
\end{itemize}

Figure 1: An example of partial label learning. The image is partially labeled by noisy annotator. Among the candidate labels, lion is ground-truth label while tiger and leopard are invalid labels.

Formally speaking, let $X = \mathbb{R}^d$ be the $d$-dimensional feature space and $Y = \{1, 2, \ldots, l\}$ be the label space including $l$ labels. Suppose the PL dataset is denoted by $D = \{(x^{(i)}, S^{(i)})|1 \leq i \leq m\}$ where $x^{(i)} \in X$ is an $d$-dimensional feature vector and $S^{(i)} \in Y$ is the corresponding candidate label set where the ground-truth label $y_i$ must be in this candidate label set, i.e., $y_i \in S^{(i)}$. Given such data, the goal of partial label learning is to train a multi-class classification model $f: X \mapsto Y$ in training samples and tries to correctly predict the label of a test samples.

Due to the semantic ambiguities conveyed by the label space, the key of partial label learning is to disambiguate the candidate label set, thereby targeting the ground-truth label [8]. To achieve this, most of the existing disambiguation-based approaches normally follow two typical strategies: the average-based strategy [13, 5] and identification-based strategy [14, 15, 28, 30, 29, 9]. The existing disambiguation-based approaches mainly attach emphasis on the construction of regularization terms according to some relationship between feature space and label space, which has been prevailing recently. Essentially, however, the main approach of most current algorithms is to fit the distribution of partial label and then to disambiguate the candidate labels [8, 25]. The effect such approach achieved covers two processes, partial label fitting and labels disambiguation, where there is a progressive relationship. When it comes to exact algorithms, the partial label fitting process is achieved by some conventional machine learning methods such as logistic regression. Whereas label disambiguation process resorts to some artificially constructed regularization terms, expected to improve the result.

Although these approaches improve the result to an extent, most of regularization terms estimate the confidence values using iterative label propagation and choose the candidate labels with high confidence values as credible labels, which are then used to induce a multi-label predictive model [26, 8, 25]. These works however suffer from the cumulative errors induced in propagation, which may impact the estimation of the credible labels and consequently severely impair the predictive model.

To narrow this kind of gap, in this paper we propose a novel method PL-GAN which adopts the adversarial learning model to replace the regularization terms. Our main motivation could be demonstrated in Figure 2. As shown in the left of Figure 2, many pictures share a kind of common candidate labels. This shows that it is possible to construct a generative network to generate a new sample which is similar to the pictures with the given candidate labels. As shown in the right of Figure 2, the pictures with the same ground truth label are usually similar, but these similar pictures may have different candidate labels. As a result, it is feasible to construct a discriminative network which could disambiguate noisy labels by the adversarial process.

The main contributions are summarized as follows:

\textsuperscript{1}Key Lab of Data Engineering and Knowledge Engineering of MOE and School of Information, Renmin University of China, PR, China
\textsuperscript{2}corresponding author, email: zhaosuyan@ruc.edu.cn
\textsuperscript{3}School of Science, China University of Geosciences (Beijing), China
We propose a framework to adopt a novel Generative Adversarial Nets to solve partial label problems by combining CGAN\cite{17} and SSGAN\cite{19}.

We give theoretical justifications of PL-GAN on challenging partial label data classification to verify that the prediction distribution of discriminator is identical to truth distribution of label space.

Experiments on various datasets validate the effectiveness of the proposed approach. It is further proved that adversarial network generally outperforms conventional machine learning methods.

The rest of this paper is organized as follows. First, the related work of partial label learning and applications of GANs are briefly reviewed. Second, the proposed approach and theoretical analyses are introduced, respectively. Third, the results of comparative experiments are reported. Finally, we draw a conclusion of this paper.

2 RELATED WORK

2.1 Partial label learning

Most of the existing disambiguation-based approaches normally follow two typical strategies, i.e., the average-based strategy \cite{13,5} and identification-based strategy \cite{14,15,28,30,29,9}. The former treats each candidate label equally, and makes the final prediction by averaging the modeling outputs of candidate labels. The latter aims to handle the candidate labels with discrimination, and usually employs an iterative process to gradually update the confidence of each candidate label.

To summarize, the implement of most approaches mentioned can be subdivided into two processes, that is to fit and to disambiguate. The first process, label fitting, is to fit the existing partial label of samples, closing to its distribution roughly. The next process, label disambiguation is to eliminate probably existing ambiguous information further, through extracting the relations between feature space and label space, to get a promotion which is usually slight but significant based on the result of fitting, just like a kind of fine trimming. Correspondingly, by state-of-the-art literature \cite{25,8} review, we find that the loss function of partial label learning composed of two parts, which are basic loss and regularization terms, to realize the above two processes respectively.

Now there exists one work, i.e., Adversarial Partial Multi-Label Learning \cite{26}, which uses the encoder-decoder framework to tackle the partial multi-label learning problem. However, in Adversarial Partial Multi-Label Learning, the mapping from the labels to the input features is hard to learn since the label space does not contain the complete information of input features. As a result, it is still promising to propose an adversarial network to conduct PLL.

2.2 Applications of GANs

Generative Adversarial Nets(GANs) \cite{10} framework is one of the most popular approaches to generative modeling. The goal of GANs is to train a generator network $G(z; \theta_g)$ that produces samples from the data distribution $p_{data}(x)$, by transforming vectors of noise $z$ as $x = G(z; \theta_g)$. The training signal for $G$ is provided by a discriminator network $D(x)$ that is trained to distinguish fake samples subject to the generator distribution $p_{model}(x)$ from real data. The generator network $G$ in turn is then trained to fool the discriminator into accepting its outputs as being real. $D$ and $G$ play the following two-player minimax game with value function $V(G, D)$:

$$
\min_G \max_D V(D, G) = E_{x \sim p_{data}(x)}[\log D(x)] + E_{z \sim p(z)}[\log(1 - D(G(z)))]
$$

Generative adversarial training has been a large range of applications in recent years, such as semi-supervised learning\cite{6,21}, unsupervised representation learning\cite{3}, imitation learning\cite{12}, Few-Shot learning\cite{31}. Unlike those supervised/semi-supervised studies, our model employs GANs in a weak-supervised task. In this paper we combine CGAN and SSGAN to solve the partial label problems.

Conditional GAN(CGAN)\cite{17}, without labels, the standard GAN generates fake samples at random, which sometimes causes some inconvenience. CGAN introduces the conditional version of generative adversarial nets, which can be constructed by simply feeding the data and the label, wishing to condition on to both the generator and discriminator.

Now, many improvements of CGAN have been proposed to do semi-supervised learning, such as SSGAN \cite{19} and Triple-GAN \cite{4}.

Semi-supervised GANs(SSGAN) \cite{19} extends CGAN to the semi-supervised context by forcing the discriminator network to output class labels. SSGAN trains a generative model $G$ and a discriminator $D$ on a dataset of $L$ classes, with $D$ made to predict $L + 1$ classes, where an extra class is added to correspond to the outputs of $G$. It is showed that this method can be used to create a more data-efficient classifier.

3 THE PROPOSED APPROACH

3.1 Model architecture of PL-GAN

The task of partial label learning is to induce a multi-class classifier $f: \mathcal{X} \rightarrow \mathcal{Y}$ from partial label training set $D = \{(x^{(i)}, y^{(i)})|1 \leq i \leq n\}$. Specifically, we denote the feature matrix and the label matrix given in the partial label training set by $\mathcal{X} = [x^{(1)}, x^{(2)}, \cdots, x^{(m)}] \in R^{n \times d}$ and $\mathcal{Y} = [y^{(1)}_1, y^{(2)}_2, \cdots, y^{(m)}_m] \in (0, 1)^m \times L$, respectively. Here, $y^{(i)}_j = 1$ means that the j-th label is among the candidate label set of the sample $x^{(i)}$ (i.e., $j \in S^{(i)}$). $y^{(i)}_j = 0$ means that the j-th label is a non-candidate label of $x^{(i)}$.

In this paper, we use regularized $\mathcal{Y}$. That is, suppose $y^{(1)}_j = [1, 0, 0]$ and $y^{(2)}_j = [0, 1, 0, 1]$, we have $y^{(1)}_j = [0.5, 0.5, 0, 0]$ and $y^{(2)}_j = [0, 0.5, 0, 0.5]$.

In this paper, we propose a novel method PL-GAN to solve partial label learning problem. The overall training framework is presented in Fig. 3. PL-GAN comprises two component networks: 1) Generator adopts the idea of CGAN by given a kind of candidate labels to generate the samples which are similar to the real samples with the given
candidate labels; 2) Discriminator adopts the idea of SSGAN which not only distinguishes the real samples and the generated samples, but also predicts the ground-truth labels.

The total loss function of PL-GAN plays the following two-player minmax game with the value function \( V(D, G) \):

\[
\max_D \min_G V(D, G) = E_{x \sim p_{\text{data}}(x)}[D(x)] - E_{z \sim p_z}(G(z)) - y_c \]  
\[
+ E_{z \sim p(z)}[D(G(z)) - y_c] \sum_{i=1}^{n} (D(G(z^{(i)})) - y^{(i)}_f)^2 \tag{2}
\]

Where \( p_{\text{data}} \) is the distribution of real training data. \( z \) is a noise vector, which is subject to Standard Normal Distribution. \( y^{(i)}_p \) is the partial label vector of real training data, \( y^{(i)}_c \) and \( y^{(i)}_f \) are the reconstructed labels of the real samples and the generated samples respectively.

To reconstruct labels, we set a \( L + 1 \) dimensional vector following [19]. That is, \( y^{(i)}_c = [y^{(i)}_p, 0], y^{(i)}_f = [0, 1] \). For example, suppose there is a real sample with the partial label \([0.5, 0.5, 0, 0, 0]\) and a generated sample. We reconstruct the label of the real sample as \([0.5, 0.5, 0, 0, 0]\) and the label of the generated sample as \([0, 0, 0, 0, 1]\).

### 3.2 Loss of Generator

Fixing the discriminator \( D \), the generator \( G \) maximizes the labeling error of the generated samples:

\[
\max_G V(D, G) = E_{z \sim p(z)}[D(G(z))] - y_c \]  
\[
+ E_{z \sim p(z)}[D(G(z)) - y_f] \sum_{i=1}^{n} (D(G(z^{(i)})) - y^{(i)}_f)^2 \tag{3}
\]

The generator \( G \) uses a neural network with one or more hidden layers, realizing the mapping from noise with partial label to feature space. To make \( G \) generate samples with the assigned partial labels, we need to input the noise \( z \) and the partial labels \( y^{(i)}_p \) into the generator \( G \) simultaneously, where \( z \) is noise vector of 100 dimensions, subject to Standard Normal Distribution. That is, the noise vector \( z \) and the partial label \( y^{(i)}_p \) are concatenated as the input of the generator \( G \). Thus, we add the partial label information into the hidden layers and then enhance the guidance effect of partial label. As a result, the generated samples are verified to be similar to the real samples with the partial label vector \( y^{(i)}_p \). It is necessary to point out that the output of the generator has the same number of dimensions as feature space with the real sample. In accordance with the scaled feature space, the activation of output layer is set to \( \tanh \), whose range is \([-1, 1]\).

### 3.3 Loss of Discriminator

Fixing the generator \( G \), the discriminator \( D \) minimizes the sum of labeling error of the real samples and the generated samples:

\[
\min_D V(D, G) = E_{z \sim p(z)}[D(G(z))] - y_c \]  
\[
+ E_{z \sim p(z)}[D(G(z)) - y_f] \sum_{i=1}^{n} (D(G(z^{(i)})) - y^{(i)}_f)^2 \tag{4}
\]

A standard classifier takes \( x \) as input and outputs a \( L \)-dimensional vector of logits \( l_1, \ldots, l_L \), that can be turned into class probabilities by applying the softmax: \( p_{\text{model}}(y = j|x) = \frac{\exp(l_j)}{\sum_{i=1}^{L} \exp(l_i)} \) [21].

In PL-GAN, such a model is then trained by minimizing the Least Squares between the observed reconstruction labels and the predictive labels. The details of reconstructed label \( y^{(i)}_c \) and \( y^{(i)}_f \) are in Subsection 3.1. Identically, the discriminator \( D \) uses a neural network with one or more hidden layers (mainly depends on the data), realizing the mapping from feature space to label space. The objective of our discriminator consists in two aspects: to give the ground-truth label of the input sample and to give the source of the input sample. And then we increase the dimension of our classifier \( D \)'s output from \( L \) to \( L + 1 \).

**Algorithm 1** Minibatch stochastic gradient descent training of PL-GAN. The number of steps to apply to the discriminator, \( k=1 \), is a hyperparameter.

**input:** training set \( D \); test set \( T(x) \); partial label \( y^{(i)}_p \) of real samples. **output:** prediction accuracy \( \hat{y} \) of test set.

for number of training iterations do

- Sample minibatch of \( n \) samples \( \{z^{(1)}, \ldots, z^{(n)}\} \) from noise prior \( p_z(z) \).
- Sample \( n \) partial label vectors \( \{y^{(1)}_p, \ldots, y^{(n)}_p\} \) from training set \( D \).
- Update the generator by descending its stochastic gradient:

\[
\nabla_{\theta_g} \frac{1}{2n} \sum_{i=1}^{n} (D(G(z^{(i)})) - y^{(i)}_f)^2 \tag{5}
\]

for \( k \) steps do

- Sample minibatch of \( n \) samples \( \{x^{(1)} | y^{(1)}_p, \ldots, x^{(n)} | y^{(n)}_p\} \) from training set \( D \).
- Sample minibatch of \( n \) samples \( \{z^{(1)} | y^{(1)}_p, \ldots, z^{(n)} | y^{(n)}_p\} \) from training set \( p_z(z^{(i)}) \).
- Update the discriminator by descending its stochastic gradient:

\[
\nabla_{\theta_d} \frac{1}{2n} \sum_{i=1}^{n} [(D(x^{(i)} | y^{(i)}_p) - y^{(i)}_c)^2 + (D(G(z^{(i)} | y^{(i)}_p)) - y^{(i)}_f)^2] \tag{6}
\]

end for

end for

- Use the discriminator to predict the test sample \( T(x) \).
3.4 Prediction

In Proposition 1 of GAN [10], we showed that when D is fixed, the objective function of Generator Eq. (3) can also be transformed into the following:

\[
E_q(3) = \max _{G} V(D,G) \Leftrightarrow \min _{G} V(D,G) = \\
E_x(z^{(i)}|y^{(i)} \sim p_{data}(z^{(i)}|y^{(i)}) ||D(x^{(i)}|y^{(i)}) - y_c^{(i)}||^2 \\
+ E_z(y^{(i)} \sim p_{z}(y^{(i)}) ||D(G(z^{(i)}) - y_c^{(i)}||^2)
\]

Proposition 2. Assume D and H have sufficient capacity. Given the fixed D, the minimum of C(G) is the lower bound by 0, which can be achieved when

\[
p_{\theta}(x^{(i)}|y^{(i)}) \approx p_{data}(x^{(i)}|y^{(i)}) \quad \text{and} \quad F(y|G(z^{(i)})) = E_{x^{(i)} \sim p_{data}(x^{(i)}|y^{(i)})} \hat{p}(y^{(i)}|x^{(i)})
\]

Proof. Based on the solution for optimal discriminator \(D_G(x^{(i)}|y^{(i)})\) in Proposition 1 and Eq. (10), we have:

\[
\min C(G) = E_{x^{(i)}|y^{(i)}} \sim p_{data}(x^{(i)}|y^{(i)}) (||D_G(x^{(i)}|y^{(i)}) - y_c^{(i)}||^2 \\
+ E_z(y^{(i)} \sim p_{z}(y^{(i)}) ||D_G(z^{(i)}|y^{(i)}) - y_c^{(i)}||^2)
\]

Considering \(L + 1\) dimension of the output of D, we have \(D_G(x^{(i)}|y^{(i)})|L+1| = 0, y_c^{(i)}|L+1| = 1\). According to Eq. (8) and \(D_G(x^{(i)}|y^{(i)})|L+1| = 0, y_c^{(i)}|L+1| = 1, \) Eq. (12) can be rewritten as:

\[
\min C^{L+1}(G) = \\
\frac{1}{2} E_{x^{(i)}|y^{(i)}} \sim p_{data}(x^{(i)}|y^{(i)}) \quad \frac{p_{data}(x^{(i)}|y^{(i)})}{p_{data}(x^{(i)}|y^{(i)}) + p_{\theta}(x^{(i)}|y^{(i)})} \frac{p_{\theta}(x^{(i)}|y^{(i)})}{p_{\theta}(x^{(i)}|y^{(i)}) + p_{\theta}(x^{(i)}|y^{(i)})}^2
\]

where \(p_{data}\) and \(p_{\theta}\) denote the feature space distribution of real data and generated data respectively.
Figure 4: Classification performance on controlled UCI datasets with \( p \) ranging from 0.1 to 0.7 (\( r = 1 \)).

<table>
<thead>
<tr>
<th></th>
<th>PL-GAN</th>
<th>PL-SVM</th>
<th>PL-KNN</th>
<th>LALO</th>
<th>PL-ECOC</th>
<th>PL-AGGD</th>
<th>SDIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>abalone</td>
<td>0.75</td>
<td>0.65</td>
<td>0.94</td>
<td>0.96</td>
<td>0.97</td>
<td>0.97</td>
<td>0.25</td>
</tr>
<tr>
<td>usps</td>
<td>0.75</td>
<td>0.65</td>
<td>0.94</td>
<td>0.96</td>
<td>0.97</td>
<td>0.97</td>
<td>0.35</td>
</tr>
<tr>
<td>letter</td>
<td>0.75</td>
<td>0.65</td>
<td>0.94</td>
<td>0.96</td>
<td>0.97</td>
<td>0.97</td>
<td>0.35</td>
</tr>
</tbody>
</table>

Figure 5: Classification performance on controlled UCI datasets with \( p \) ranging from 0.1 to 0.7 (\( r = 3 \)).

<table>
<thead>
<tr>
<th></th>
<th>PL-GAN</th>
<th>PL-SVM</th>
<th>PL-KNN</th>
<th>LALO</th>
<th>PL-ECOC</th>
<th>PL-AGGD</th>
<th>SDIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>abalone</td>
<td>0.75</td>
<td>0.65</td>
<td>0.94</td>
<td>0.96</td>
<td>0.97</td>
<td>0.97</td>
<td>0.25</td>
</tr>
<tr>
<td>usps</td>
<td>0.75</td>
<td>0.65</td>
<td>0.94</td>
<td>0.96</td>
<td>0.97</td>
<td>0.97</td>
<td>0.35</td>
</tr>
<tr>
<td>letter</td>
<td>0.75</td>
<td>0.65</td>
<td>0.94</td>
<td>0.96</td>
<td>0.97</td>
<td>0.97</td>
<td>0.35</td>
</tr>
</tbody>
</table>

\[
= -2\log(2) + KL(p_{data}(x^{(i)}|\gamma^{(i)}_p)) \bigg|\bigg| \frac{p_{data}(x^{(i)}|\gamma^{(i)}_p) + p_g(x^{(i)}|\gamma^{(i)}_p)}{2}
\]

\[
KL(p_g(x^{(i)}|\gamma^{(i)}_p)) \bigg|\bigg| \frac{p_{data}(x^{(i)}|\gamma^{(i)}_p) + p_g(x^{(i)}|\gamma^{(i)}_p)}{2}
\]

\[
= -2\log(2) + 2 JSD(p_{data}(x^{(i)}|\gamma^{(i)}_p)||p_g(x^{(i)}|\gamma^{(i)}_p))
\]

\[
\geq 0
\]

Where JSD is the Jensen–Shannon divergence. Thus minimizing Eq. (13) is equal to minimizing the divergence of \( p_{data}(x^{(i)}|\gamma^{(i)}_p) \) and \( p_g(x^{(i)}|\gamma^{(i)}_p) \). Thus, the minimum of \( C^{L+1}(G) \) is the lower bound by 0, i.e., the distributions of \( p_{data}(x^{(i)}|\gamma^{(i)}_p) \) and \( p_g(x^{(i)}|\gamma^{(i)}_p) \) are more and more similar but not exactly equal. That is

\[
JSD(p_{data}(x^{(i)}|\gamma^{(i)}_p)||p_g(x^{(i)}|\gamma^{(i)}_p)) = \log 2
\]

\[
\Leftrightarrow p_{data}(x^{(i)}|\gamma^{(i)}_p) \simeq p_g(x^{(i)}|\gamma^{(i)}_p)
\]

Considering 1\(-L\)th dimension of the output of \( D \), \( D_G(x^{(i)}|\gamma^{(i)}_p)[1 : L] \sim p(\hat{y}|x^{(i)}) \in [0,1]^{1 \times L} \) holds and it is fixed. And \( y^{(i)}_C[1 : L] \sim p(y^{(i)}|x^{(i)}) \in [0,1]^{1 \times L} \). Then \( F(\hat{y}|G(z|\gamma^{(i)}_p)) = D_G(x^{(i)}|\gamma^{(i)}_p)[1 : L] = F(\hat{y}|x^{(i)}|\gamma^{(i)}_p)) \in [0,1]^{1 \times L} \). Thus Eq. (12) can be rewritten as:

\[
\min C^{L}(G) = E_{x^{(i)}|\gamma^{(i)}_p \sim p_{data}(x^{(i)}|\gamma^{(i)}_p)} ||D_G(x^{(i)}|\gamma^{(i)}_p)[1 : L] - y^{(i)}_C[1 : L]||^2
\]

\[
+ E_{x^{(i)}|\gamma^{(i)}_p \sim p_{z}(x|\gamma^{(i)}_p)} ||D_G(z|\gamma^{(i)}_p)[1 : L] - y^{(i)}_C[1 : L]||^2
\]

(20)

Adjusting \( G \) is only related to the loss of \( ||D_G(z|\gamma^{(i)}_p)[1 : L] -

\( y^{(i)}_C[1 : L] || \) in Eq. (20). So we have

\[
\min C^{L}(G) = E_{x^{(i)}|\gamma^{(i)}_p \sim p_{data}(x^{(i)}|\gamma^{(i)}_p)} ||D_G(x^{(i)}|\gamma^{(i)}_p)[1 : L] - y^{(i)}_C[1 : L]||^2
\]

(21)

The minimal \( C^{(L)}(G) = 0 \) can only be achieved when

\[
F(\hat{y}|G(z|\gamma^{(i)}_p)) = E_{x^{(i)}|\gamma^{(i)}_p \sim p_{data}(x^{(i)}|\gamma^{(i)}_p)} p(y^{(i)}|x^{(i)})
\]

(22)

To sum up, we find that the optimal \( \min C(G) \) is equal to the optimal \( C^{(L)}(G) \) and the optimal \( C^{(L+1)}(G) \). If \( G \) and \( D \) have enough capacity, then Algorithm 1 is convergent, just as subsection 4.2 in GANs [10]. It is obvious the optimal condition Eq. (12) = \( C^{(G)} \) is satisfied. Hence the proposition 2 is proved.

5 EXPERIMENTS

In this section, we conduct extensive experiments on artificial and Real-World datasets to demonstrate the effectiveness of our proposed approach. These datasets are public on the Internet and we only have handled datasets, whose feature dimension is already determined by handlers.

5.1 Comparing algorithms

We compare our approach with six state-of-the-art partial label learning approaches, each configured with suggested hyper-parameters in accordance with the respective literatures:

- PL-KNN [13] which makes predictions via \( k \)-nearest neighbor weighted voting [suggested configuration: \( k = 10 \)].
• PL-SVM [18] which learns from PL examples by optimizing margin based objective function [suggested configuration: $\lambda \in \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$].
• PL-ECOC [29] which transforms partial label learning problem into binary learning problem via ECOC coding matrix [suggested configuration: codeword length $L = \log_2(1 + p)$].
• LALO [7] which leverages the topological information in feature space to derive the confidence of each candidate label [suggested configuration: $k = 10$, $\lambda = 0.05$, $\mu = 0.005$].
• PL-AGGD [23] which preserves the manifold structure from feature space in the label space [suggested configuration: $\lambda \in \{0.001, 0.005, \ldots, 0.5\}$ and $\beta \in \{0.0001, 0.0005, \ldots, 0.1\}$].

For our approach PL-GAN, the learning rate of $G$ and $D$ are both set to 0.01 or 0.001. The dropout of $D$ is set to 0.5. The hidden layers activation function of $G$ and $D$ is Rectified linear Units (ReLU), and the activation function of the output layer of $G$ and $D$ are Tanh and Softmax, respectively. For all approaches, the training and testing accuracies are obtained based on ten-fold cross-validation. And the resulting mean prediction accuracies and the standard deviations are reported.

Table 1: Characteristics of the controlled UCI data sets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Examples</th>
<th>#Features</th>
<th>#Labels</th>
</tr>
</thead>
<tbody>
<tr>
<td>vehicle</td>
<td>846</td>
<td>18</td>
<td>4</td>
</tr>
<tr>
<td>abalone</td>
<td>4177</td>
<td>7</td>
<td>29</td>
</tr>
<tr>
<td>usps</td>
<td>9298</td>
<td>256</td>
<td>10</td>
</tr>
<tr>
<td>letter</td>
<td>200000</td>
<td>16</td>
<td>26</td>
</tr>
</tbody>
</table>

Configurations:

(I) $r = 1$, $p \in \{0.1, 0.2, \ldots, 0.7\}

(II) $r = 2$, $p \in \{0.1, 0.2, \ldots, 0.7\}

(III) $r = 3$, $p \in \{0.1, 0.2, \ldots, 0.7\}$

(IV) $p = 1$, $r = 1$, $\epsilon \in \{0.1, 0.2, \ldots, 0.7\}$

5.2 Controlled UCI data sets

Table 1 reports the characteristics of four UCI datasets [1] used in our experiments. Following the widely-used controlling protocol [28, 30, 7, 24], we use three controlling parameters $p$, $r$ and $\epsilon$ to generate the artificial partial label datasets. Here, $p$ controls the proportion of samples which have candidate labels (i.e., $|S^{(0)}| > 1$), $r$ controls the number of false positive labels, in other words, $|S^{(0)}| = r + 1$ and $\epsilon$ controls the co-occurring probability between one extra candidate label and the ground-truth label. As shown in Table 1, a total of 28 (4 x 7) parameter configurations are considered for each controlled UCI dataset. Fig. 4&5 report the classification accuracies of each approach as $p$ ranges from 0.1 to 0.7 with step size 0.1, when $r = 1$ and $r = 3$ respectively. Due to page limit, figures for the cases of $r = 2$ are not illustrated while similar results to Fig. 4&5 can be observed as well. Fig. 6 illustrates the classification accuracies of each comparing algorithm as $\epsilon$ increases from 0.1 to 0.7 with step-size 0.1 ($p = 1$, $r = 1$).

As shown in these figures, the results demonstrate that our method can achieve the best performance in more than 107 cases out of the 112 cases. That is to say, PL-GAN outperforms other comparing algorithms in most cases.

These experimental results on four UCI datasets demonstrate that PL-GAN is obviously superior to the state-of-the-art PLL methods.

5.3 Real-World data sets

Table 2 summarizes the characteristics of Real-World partial label data sets, which are collected from several application domains. The average number of candidate labels (avg. #CLs) for each real-world partial label data set is also recorded in Table 2.

Table 3 reports the mean and standard deviation of classification accuracy on each comparing algorithm. Pairwise-samples $t$-test at 0.05 significance level is conducted on all comparing algorithms except SDIM. Because the running time of SDIM is too long and then the results of SDIM is taken from [8]. The testing outcomes between PL-GAN and other algorithms are also recorded as specific marks in Table 3.

As shown in Table 3, it is impressive to observe that:

• It is worthy nothing that all the approaches achieve extremely poor performance on FG-NET, because its Avg. #CLs is very large. But PL-GAN has best promotion.
• On two comparatively large datasets, i.e., Soccer Player and Yahoo! News, PL-GAN achieves superior performance against all the comparing approaches.
• PL-GAN significantly outperforms the state-of-the-art algorithm SDIM on FG-NET, Soccer Player and Yahoo! News.
• PL-GAN significantly outperforms PL-AGGD, LALO, PLECOC, PL-SVM on most datasets.
• PL-GAN significantly outperforms PL-KNN on all datasets.

These experimental results on Real-World datasets demonstrate that PL-GAN is obviously superior to the state-of-the-art PLL methods.
Table 2: Characteristics of real-world partial label datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Examples</th>
<th>#Features</th>
<th>#Labels</th>
<th>Avg. #CLS</th>
<th>Task Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>FG-NET</td>
<td>1002</td>
<td>262</td>
<td>78</td>
<td>7.48</td>
<td>facial age estimation [20]</td>
</tr>
<tr>
<td>Lost</td>
<td>1122</td>
<td>108</td>
<td>16</td>
<td>2.23</td>
<td>automatic face naming [5]</td>
</tr>
<tr>
<td>MSRCv2</td>
<td>1758</td>
<td>48</td>
<td>23</td>
<td>3.16</td>
<td>object classification [15]</td>
</tr>
<tr>
<td>BirdSong</td>
<td>4988</td>
<td>38</td>
<td>13</td>
<td>2.18</td>
<td>bird song classification [2]</td>
</tr>
<tr>
<td>Soccer Player</td>
<td>17472</td>
<td>279</td>
<td>171</td>
<td>2.09</td>
<td>automatic face naming [11]</td>
</tr>
<tr>
<td>Yahoo! News</td>
<td>22991</td>
<td>163</td>
<td>219</td>
<td>1.91</td>
<td>automatic face naming [11]</td>
</tr>
</tbody>
</table>

Table 3: Classification accuracy of each algorithm on the real-world datasets. Furthermore, * / o indicates whether PL-GAN is statistically superior/inferior to the comparing algorithm (pairwise t-test at 0.05 significance level).

<table>
<thead>
<tr>
<th></th>
<th>PL-GAN</th>
<th>SDIM</th>
<th>PLAGGD</th>
<th>LALO</th>
<th>PLECOC</th>
<th>PLSVM</th>
<th>PLKNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>FG-NET</td>
<td>0.158 ±0.047</td>
<td>0.076±0.019 *</td>
<td>0.079±0.029 *</td>
<td>0.076±0.036 *</td>
<td>0.037±0.025 o</td>
<td>0.054±0.025 o</td>
<td>0.042±0.016 o</td>
</tr>
<tr>
<td>Lost</td>
<td>0.676±0.028</td>
<td>0.801±0.031 o</td>
<td>0.776±0.033 o</td>
<td>0.742±0.041 o</td>
<td>0.673±0.054 o</td>
<td>0.713±0.056 o</td>
<td>0.351±0.036 o</td>
</tr>
<tr>
<td>MSRCv2</td>
<td>0.533±0.037</td>
<td>0.518±0.037 o</td>
<td>0.491±0.018 o</td>
<td>0.478±0.041 o</td>
<td>0.430±0.036 o</td>
<td>0.390±0.029 o</td>
<td>0.438±0.052 o</td>
</tr>
<tr>
<td>BirdSong</td>
<td>0.721±0.023</td>
<td>0.754±0.021 o</td>
<td>0.727±0.017 o</td>
<td>0.724±0.017 o</td>
<td>0.713±0.026 o</td>
<td>0.656±0.037 o</td>
<td>0.645±0.020 o</td>
</tr>
<tr>
<td>Soccer Player</td>
<td>0.696±0.008</td>
<td>0.557±0.016 o</td>
<td>0.543±0.009 o</td>
<td>0.540±0.010 o</td>
<td>0.562±0.011 o</td>
<td>0.469±0.010 o</td>
<td>0.494±0.008 o</td>
</tr>
<tr>
<td>Yahoo! News</td>
<td>0.721±0.013</td>
<td>0.663±0.013 o</td>
<td>0.648±0.012 o</td>
<td>0.636±0.012 o</td>
<td>0.657±0.010 o</td>
<td>0.597±0.014 o</td>
<td>0.409±0.009 o</td>
</tr>
</tbody>
</table>

Figure 7: left: the generated fake samples by given ground-truth labels (0, 1, · · · , 9); right: the generated fake samples by given candidate labels ([0, 1], [1, 2], · · · , [9, 0]).

5.4 Visualization of Generator

In this subsection, we visualize the results of the generator on MNIST. Setting r = 3, p = 7, MNIST is transformed into partial labeled data. Note that for each iteration, the input of D are fed in two times. First input (x(0), y0) to D, getting an output; then input (x(0), 0) to D, where 0 is to keep the same dimension, then getting another output. Concatenate the 1~Nth dimension of the second output and the N+1th dimension of the first output as the final output of D in this epoch. Doing this ensures the conditional effect of labels and avoid them influencing the classification process. The visual results of the generator are shown in Fig. 7. Comparing the two pictures, it is easy to find that the generated picture is a mixture of the features of the candidate labels. For example, the tenth line in the right picture is generated by given a partial label [9, 0], and the generated sample is a feature mixture of label 9 and 0.

5.5 Accuracy curve

This subsection displays the accuracy curves of PL-GAN. The curve in Fig. 8 demonstrates that the testing accuracy gradually increases to a desired and stable level with incremental iteration times. This shows that the optimal accuracy can be reached just in a proper number of iterations.

6 CONCLUSION

In this paper, we propose a novel approach, i.e., PL-GAN to solve partial label learning. In particular, we design the generator which adopts the idea of CGAN to generate samples similar to the real sample with the given candidate labels and the discriminator which adopts the idea of SSGAN to distinguish true/fake samples and to predict the ground-truth labels. By the game of the generator and the discriminator, the ground-truth label is found. Finally, we conduct extensive experiments on some UCI and Real-World datasets whose results show that PL-GAN outperforms all the comparison methods.

ACKNOWLEDGEMENTS

This work is supported by National Key Research & Develop Plan(2018YFB1004401, 2017YFB1400700, 2016YFB1000702), NSFC under the grant No.61702522, 61772536, 61772537, 61732006, 61532021 and NSSFC (No.12 & ZD220), National Basic Research Program of China (973) (No.2014CB340402), National High Technology Research and Development Program of China (863) (No.2014AAA015204). It was partially done when the authors worked in SA Center for Big Data Research in RUC. This Center is funded by a Chinese National 111 Project Attracting.

REFERENCES


Lei Feng and Bo An. Partial label learning by semantic difference maximization. Proceedings of the 28th International Joint Conference on Artificial Intelligence, 2019.


Gabriel Panis and Andreas Lanitis. An overview of research activities in facial age estimation using the fg-net aging database. 2015.


