Deep Graph Matching for Partial Label Learning

Gengyu Lyu[*] , **Yanan Wu**[*] and **Songhe Feng**[†]
School of Computer and Information Technology, Beijing Jiaotong University
{18112030, 19112034, shfeng}@bjtu.edu.cn

Abstract

Partial Label Learning (PLL) aims to learn from training data where each instance is associated with a set of candidate labels, among which only one is correct. In this paper, we formulate the task of PLL problem as an "instance-label" matching selection problem, and propose a DeepGNN-based graph matching PLL approach to solve it. Specifically, we first construct all instances and labels as graph nodes into two different graphs respectively, and then integrate them into a unified matching graph by connecting each instance to its candidate labels. Afterwards, the graph attention mechanism is adopted to aggregate and update all nodes state on the instance graph to form structural representations for each instance. Finally, each candidate label is embedded into its corresponding instance and derives a matching affinity score for each instance-label correspondence with a progressive cross-entropy loss. Extensive experiments on various data sets have demonstrated the superiority of our proposed method.

1 Introduction

In Partial Label Learning (PLL), each instance is associated with a set of candidate labels, among which only one is the ground-truth label [Wang *et al.*, 2019a; Wang *et al.*, 2022; Feng and An, 2019; Lyu *et al.*, 2020a]. This learning problems is motivated in domains where a large number of ambiguously labeled examples are available while it is expensive to acquire complete and explicit labels, such as web mining [Luo and Orabona, 2010; Wang *et al.*, 2019b], image classification [Cour *et al.*, 2009; Zeng *et al.*, 2013; Chen *et al.*, 2017], crowdsourcing [Liu and Dietterich, 2012; Liu *et al.*, 2018; Wang *et al.*, 2020], etc.

Formally speaking, let $\mathcal{X} = \mathbb{R}^d$ denote the d-dimensional input space, and $\mathcal{Y} = \{1, 2, \ldots, q\}$ denote the output space with q class labels. The task of PLL is to induce a multi-class classifier $\boldsymbol{f} : \mathcal{X} \mapsto \mathcal{Y}$ from the training data $\mathcal{D} = \{(\boldsymbol{x}_i, \mathcal{S}_i)\}(1 \leq i \leq n)$ with n examples, where the instance $\boldsymbol{x}_i \in \mathcal{X}$ is described as a d-dimensional feature vector, the candidate label set $\mathcal{S}_i = \{y_{i_1}, y_{i_2}, \ldots, y_{i_{|\mathcal{S}_i|}}\} \subseteq \mathcal{Y}$ is associated with the instance $\boldsymbol{x}_i$ and $|\mathcal{S}_i|$ represents the number of candidate labels for instance $\boldsymbol{x}_i$.

Obviously, a successful PLL model rely heavily on its superior capability of disambiguating ambiguous labels. Therefore, most of existing PLL approaches [Lyu *et al.*, 2020b] focus on how to effectively disambiguate the candidate label set and obtain the correct "instance-label" matching $\{(\boldsymbol{x}_i, y_i)|1 \leq i \leq n\}$ for model induction. Among these methods, [Hullermeier and Beringer, 2005] and [Zhang and Yu, 2015] adopt an *averaging disambiguation* strategy, which assumes each candidate label contributes equally to the learning model and assigns label for unseen instance by (weighted) averaging the outputs from all candidate labels. [Yu and Zhang, 2016] and [Feng and An, 2018] employ another disambiguation strategy named *identification disambiguation*, which views the ground-truth label as a latent variable first, and then refines it iteratively during the model training phase. Intuitively, the above two kinds of strategies serve for the model design from different perspective, where the former focuses on exploiting the similarity of k-nearest neighbor instances while the latter pays more attention to leveraging the distinctions of varying candidate labels.

Recently, [Lyu *et al.*, 2021] integrates the two kinds of relationships, formulated as *"instance-label" matching consistent* by a predefined instance-label affinity matrix $\boldsymbol{K}$, and propose a graph matching PLL approach, formulated as:

$$\begin{aligned} &\arg\max_{\boldsymbol{p}} \boldsymbol{p}^\top \boldsymbol{K} \boldsymbol{p} \\ &s.t. \quad \boldsymbol{p} \in \{0,1\}^{u\times 1} \quad \boldsymbol{P}\underline{\boldsymbol{1}} = \underline{\boldsymbol{1}}, \end{aligned} \tag{1}$$

where, $\boldsymbol{p} \in \mathbb{R}^{\sum_{i=1}^{n}|S_i|\times 1}$ is the row-wise vectorized replica of label confidence matrix $\boldsymbol{P} \in \mathbb{R}^{n\times q}$ and the non-candidate labels have been removed when generating $\boldsymbol{p}$. $\boldsymbol{K} \in \mathbb{R}^{\sum_{i=1}^{n}|S_i|\times\sum_{i=1}^{n}|S_i|}$ is the instance-label affinity matrix that encodes "instance-label" matching consistent.

Although this approach has achieved competitive performance on solving PLL problem, it still suffers from some shortcomings: (**i**) The integrated relationship in $\boldsymbol{K}$ is *manually predetermined* from original features instead of *learning* from structural representations, which inevitably decreases the performance of the learned model. (**ii**) The predefined affinity matrix $\boldsymbol{K}$ characterizes the co-occurrence confidence

[*]indicates equal contributions.
[†]is the corresponding author.

of different "instance-label" matchings, formulated with a large scale of $(\sum_n |\mathcal{S}_i|)^2$, which significantly increases the computational cost of learning model and reduces its learning efficiency on large scale data sets.

To overcome the above shortcomings, in this paper, we propose a **D**eepGNN-based **G**raph m**A**tching **P**LL (**D-GAP**) approach, where the Deep Graph Neural Network (Deep-GNN) and Graph Matching (GM) are simultaneously incorporated owing to their excellent performance of excavating and exploiting the structural information of training data. Specifically, we first construct all instances and labels into two different graphs (instance graph and label graph) respectively, and then integrate them into a unified matching graph by connecting each instance to its candidate labels. Afterwards, the graph attention mechanism is adopted to aggregate and update all nodes state on the instance graph to form structural representations for each instance. Owing to the inherent characteristics of PLL problem that the prior pairwise-label relationship is always missing (i.e., the labels have no significant correlations), we remove the edges in the label graph and discard the updates to all label nodes. Finally, we conduct *affinity* estimation for each "instance-label" matching on the unified matching graph, and design a progressive cross-entropy loss to maximize the affinity scores between each instance and its ground-truth label.

Notably, compared with previous PLL methods, our proposed D-GAP has the following superiorities:

- Each instance node is updated via aggregating its k-nearest neighbors to obtain its intrinsic structural representation, which well filters potential outliers and builds more reliable "instance-label" relationship.
- Our employed "instance-label" matching affinity relationships are *learned* from the above structural data representations, instead of *manually predetermined and remaining fixed* during the training phase, which guides the proposed approach to gain higher performance.
- Such "instance-label" matching *affinity* relationship replaces previous "instance-label" matching *consistency* relationship, which significantly reduces the computational cost from $O\left((\sum_n |\mathcal{S}_i|)^2\right)$ to $O(\sum_n |\mathcal{S}_i|)$ and improve its efficiency of learning from large-scale data.

2 Related Work

2.1 Averaging Disambiguation Strategy

Averaging disambiguation-based PLL approaches usually assume that each candidate label has equal contribution to the learning model and they make prediction for unseen instances by averaging the outputs from all candidate labels. [Cour *et al.*, 2011] proposes a convex learning approach, which identifies the ground-truth label from the ambiguous label set by averaging the outputs of all candidate labels. [Hullermeier and Beringer, 2005] adopts an instance-based model, which assigns predictive labels by the voting results of its k-nearest neighbors' candidate labels. Considering the different contributions of k-nearest-neighbor instances, [Zhang and Yu, 2015] employs minimum error reconstruction criterion and they improve the prediction accuracy via k-nearest neighbors' weighted-voting results. Obviously, the above approaches mainly focus on leveraging instance relationship while ignoring the latent label relationship, which leads to a shortcoming that the output of the ground-truth label is overwhelmed by that of other false positive labels.

2.2 Identification Disambiguation Strategy

Identification disambiguation-based PLL approaches usually pay attention to the differences of candidate labels and directly identify the ground-truth label from the ambiguous candidate label set. Existing PLL approaches following such strategy often regard the ground-truth label as a latent variable first, and then refine the latent variable and the model parameter iteratively by utilizing Expectation-Maximization procedure [Jin and Ghahramani, 2003]. Among these approaches, some of them usually incorporate the maximum likelihood criterion and obtain the optimal label via maximizing the outputs of candidate labels [Liu and Dietterich, 2012]. Others often utilize the maximum margin criterion and identify the ground-truth label according to maximizing the margin between the outputs of candidate labels and that of the non-candidate labels [Yu and Zhang, 2016]. The above approaches focus on identifying the unique ground-truth label, which avoid its outputs overwhelmed by other false positive labels and lead a better label disambiguation performance.

2.3 Disambiguation Free Strategy

Disambiguation free-based approaches usually attempt to learn from partial label data by fitting the data to some off-the-shelf learning techniques, where they can directly predict labels for unseen instances without conducting disambiguation operation. [Zhang *et al.*, 2017] proposes an *Error-Correcting Output Codes* based PLL algorithm named PL-ECOC, which transfers the PLL problem into binary learning problem. [Wu and Zhang, 2018] proposes another disambiguation-free algorithm called *PALOC*, which enables binary decomposition for partial label data without relying on extra manipulations. Despite extensive experiments have empirically demonstrated that the above approaches can achieve competitive performance with the other disambiguation based PLL approaches, such strategy has still not been widely adopted in constructing other PLL model, since it relies heavily on the fitting degrees between the partial label data and the employed off-the-shelf learning techniques.

3 The Proposed Method

D-GAP is a novel PLL framework based on deep graph matching scheme, which aims to excavate the structural information of training data and establish an accurate assignment relationship between instance space $\mathcal{X}$ and label space $\mathcal{Y}$. To better introduce our motivation and make our subsequent descriptions easily understanding, we illustrate the D-GAP approach as a GM structure in Figure 1 in advance.

3.1 Assignment Graph Construction

As depicted in Figure 1, both instance space and label space are formulated as two distinct graphs $\mathbb{G}^{(i)} = (\mathbb{V}^{(i)}, \mathbb{E}^{(i)})$ of size $m^{(i)}$, where $i \in \{1, 2\}$, and $m^{(1)} = n$, $m^{(2)} = q$. The

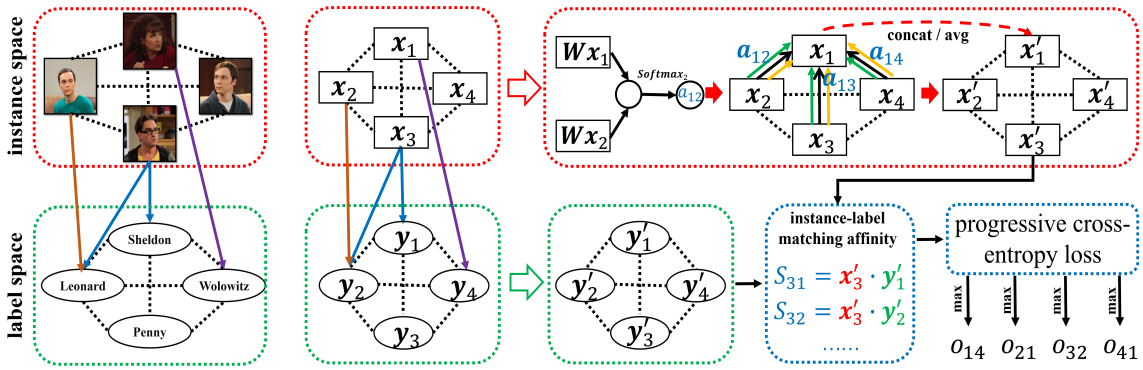

Figure 1: The overview architecture of our proposed D-GAP consists of three components: *Assignment Graph Construction*, *"Instance-Label" Matching Affinity Modeling* and *"Instance-Label" Matching Selection*. In assignment graph construction module, we construct all instances and labels into different graphs (i.e., instance spatial graph and label semantic graph) respectively, and then connect each instance and its candidate labels as candidate "instance-label" matching edges. In "instance-label" matching affinity module, graph attention mechanism and linear transformation are separately applied to form structural data representations on instance graph and label graph, then the "instance-label" matching affinity is characterized by measuring the similarity of structural representation of each instance and its candidate labels. In "instance-label" matching selection module, a progressive cross-entropy loss is designed to estimate the above "instance-label" matching affinity degree, where the matching affinity between each instance and its ground-truth label is progressively increasing.

nodes $\mathbb{V}^{(i)}$ in the two graphs represent the instances and labels respectively, while the edges $\mathbb{E}^{(i)}$ encode their similarities. Specifically, in the instance graph $\mathbb{G}^{(1)}$, we described each instance node $v_r^{(1)} \in \mathbb{V}^{(1)}$ $(1 \leq r \leq n)$ by a $d^{(1)}$-dimensional vector $\boldsymbol{x}_r$, and then the edges $e^{(1)} \in \mathbb{E}^{(1)}$ between each pair of instance nodes can be produced as:

$$e_{rs}^{(1)} = \begin{cases} 1, & \text{where } v_s^{(1)} \in \mathcal{N}(v_r^{(1)}), S_r \cap S_t \neq \emptyset \\ 0, & \text{otherwise} \end{cases}, \quad (2)$$

where $\mathcal{N}(v_r^{(1)})$ denotes the k-nearest neighbors (measured by Euclidean distance) of $v_r^{(1)}$, and $e_{rs}^{(1)} = 1$ indicates a directed edge from $v_s^{(1)}$ to $v_r^{(1)}$, $e_{rs}^{(1)} = 0$ otherwise. Note that, when constructing instance graph, we drop the edges that connect two instances with completely different candidate labels (i.e., $S_r \cap S_t = \emptyset$), which avoids the interference of false positive instances.

In the label graph $\mathbb{G}^{(2)}$, each label node $v_{r'}^{(2)} \in \mathbb{V}^{(2)}$ $(1 \leq r' \leq q)$ is represented by a $d^{(2)}$-dimensional vector

$$\boldsymbol{y}_{r'} = \frac{1}{|\mathcal{T}|} \sum_i \boldsymbol{x}_i, \quad \text{where } \mathcal{T} = \{\boldsymbol{x}_i | y_{r'} \in S_i\} \quad (3)$$

and the edges $e^{(2)} \in \mathbb{E}^{(2)}$ are generated following:

$$e_{r's'}^{(2)} = \begin{cases} 1, & \text{where } r' = s' \\ 0, & \text{where } r' \neq s' \end{cases}, \quad (4)$$

where the edges between different labels are dropped owing to the inherent characteristic of PLL problem that the prior pairwise-label relationship does not hold. Note that, once the label relationship can be obtained, the proposed D-GAP can still be easily extended to satisfy the problem.

After obtaining the above two graphs, we incorporate them into a unified matching graph by connecting each instance to its candidate labels, where the edges encode the *"instance-label" matching affinity* relationship. Thereafter, the task of PLL is transferred to a matching (edge) selection problem between each instance and its candidate labels. And the goal of D-GAP is to obtain the accurate "instance-label" matching (edge) between $\mathbb{G}^{(1)}$ and $\mathbb{G}^{(2)}$.

3.2 Modeling "Instance-Label" Matching Affinity

In order to achieve accurate "instance-label" matching between $\mathbb{G}^{(1)}$ and $\mathbb{G}^{(2)}$, we apply Deep-GNN to the unified matching graph to excavate structural representation of training data, and then incorporate GM scheme to model the "instance-label" matching affinity relationship by utilizing the structural representation data.

Instance Space Given the original instance node feature $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\} \in \mathbb{R}^{d^{(1)} \times n}$, we employ an attention-based DeepGNN architecture (GAT) [Veličković *et al.*, 2018] to compute the hidden representation of each instance node $\{\boldsymbol{x}'_1, \boldsymbol{x}'_2, \ldots, \boldsymbol{x}'_n\} \in \mathbb{R}^{d^{(1)'} \times n}$ by attending over its k-nearest neighbors, where GAT is incorporated owing to its excellent performance on measuring the different contributions of k-nearest neighbors when updating graph nodes. During the process of model construction, we first transform the input features $\boldsymbol{x}_r$ into higher-level features $\boldsymbol{W}\boldsymbol{x}_r$ to obtain sufficient expressive power, where the shared linear transformation is parameterized by a weight matrix $\boldsymbol{W} \in \mathbb{R}^{d^{(1)'} \times d^{(1)}}$. Then, the attention coefficient a_{rs} that encodes the importance of node v_s's features to node v_r can be computed as:

$$\begin{aligned} a_{rs} &= \text{softmax}_s\left(h(\boldsymbol{W}\boldsymbol{x}_r || \boldsymbol{W}\boldsymbol{x}_s)\right) \\ &= \frac{\exp\left(h(\boldsymbol{W}\boldsymbol{x}_r || \boldsymbol{W}\boldsymbol{x}_s)\right)}{\sum_{\boldsymbol{x}_t \in \mathcal{N}(\boldsymbol{x}_s)} \exp\left(h(\boldsymbol{W}\boldsymbol{x}_r || \boldsymbol{W}\boldsymbol{x}_t)\right)}, \end{aligned} \quad (5)$$

where $h(\cdot) = \text{LeakyReLU}(\boldsymbol{h}^\top \cdot)$ is a single-layer feedforward neural network, parameterized by a weight vector $\boldsymbol{h} \in$

$\mathbb{R}^{2d^{(1)'}}$, and applying with the LeakyReLU nonlinearity. $\cdot^\top$ represents transposition and $||$ is the concatenation operation. After obtaining the attention coefficients a_{rs}, we further employ *multi-head attention* mechanism [Vaswani *et al.*, 2017] to stabilize the learning process, and L independent attention mechanisms execute the transformation of $\boldsymbol{x}_r \rightarrow \boldsymbol{x}'_r|_1^L$, then their features are averaged, resulting the final output feature representation:

$$\boldsymbol{x}'_r = \frac{1}{L}\sum_{l=1}^{L}\sum_{\boldsymbol{x}_s \in \mathcal{N}(\boldsymbol{x}_r)} a_{rs}^l \boldsymbol{W}^l \boldsymbol{x}_s, \tag{6}$$

where a_{rs}^l is the attention coefficient computed by the l-th attention mechanism (h^l), and $\boldsymbol{W}^l$ is the corresponding input linear transformation's weight matrix.

Label Space Due to the lack of pairwise-label relationship in PLL, each node in label graph $\mathbb{G}^{(2)}$ can only be updated by itself. Therefore, when given the original label node feature $\{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_q\}$, to obtain sufficient expressive power, we obtain their structural representations by directly transferring the input $\boldsymbol{y}_{r'}$ to the corresponding higher-level

$$\boldsymbol{y}'_{r'} = \boldsymbol{H}\boldsymbol{y}_{r'}, \tag{7}$$

where $\boldsymbol{H} \in \mathbb{R}^{d^{(2)} \times d^{(2)'}}$ is the linear transform matrix. Here, we emphasize again that once the label relationship can be obtained, the label graph can be easily extended to be updated to obtain more expressively structural label representations.

"Instance-Label" Matching Affinity After obtaining the structural representations from both feature and label space, we incorporate the GM scheme to model the"instance-label" matching affinity relationship, where the affinities between each instance and its candidate labels are calculated by

$$o_{rs} = \mathrm{softmax}_{s'}\left(\boldsymbol{x}'_r \cdot \boldsymbol{y}'_{s'}\right), \tag{8}$$

where $y_{s'} \in S_r$ and $d^{(1)'} = d^{(2)'}$. Furthermore, we dig out the superiority of our designed "instance-label" matching affinity relationship. On one hand, compared with previous instance-label relationship that directly constructed from original training data, our "instance-label" matching affinity relationship is designed from the learned structural training data, which can provide more robust "instance-label" relationship for model induction. On the other hand, compared with the "instance-label" matching consistency relationship in [Lyu *et al.*, 2021], which describes the co-occurrence confidence of two different "instance-label" matchings, our "instance-label" matching affinity relationship directly characterizes the matching confidence between each instance and its candidate labels, which significantly reduces the computational complexity by an order of magnitude (from $O\left((\sum_n |\mathcal{S}_i|)^2\right)$ to $O(\sum_n |\mathcal{S}_i|)$). Therefore, our proposed D-GAP is not only effective but also efficient.

3.3 Progressive Cross-Entropy Loss

Due to the existence of noisy labels in candidate label set, if we directly employ the standard cross-entropy loss to induce a model, it will naturally overfit these noisy labels, thereby achieving poor learning performance. To address this issue, motivated by [Lv *et al.*, 2020], we design a *progressive cross-entropy loss* to guide the learning process,

$$\mathcal{L} = \sum_{r=1}^{n}\sum_{s=1}^{q} z_{rs}\ell\left(o_{rs}, u_s^{\mathcal{S}_r}\right), \tag{9}$$

where ℓ is the standard cross-entropy loss, z_{rs} is a weight value that encodes the confidence of the s-th label being the ground-truth label of the r-th instance, $u_s^{\mathcal{S}_r}$ is the s-th element of $\boldsymbol{u}^{\mathcal{S}_r}$ and $\boldsymbol{u}^{\mathcal{S}_r}$ is a vector that $\{i\}_{y_i \in \mathcal{S}_r}$-th elements in $\boldsymbol{u}^{\mathcal{S}_r}$ equals 1 and others equal 0. During the training process, we optimize the matching affinity o_{rs} and the weight z_{rs} in an iteration manner. In order to put more weights on possible labels, the weight z_{rs} is directly estimated by using current "instance-label" matching affinity, i.e.,

$$z_{rs} = \begin{cases} o_{rs} / \sum_{s=1}^{|\mathcal{S}_r|} o_{rs}, & \text{if } y_s \in \mathcal{S}_r \\ 0, & \text{otherwise.} \end{cases}, \tag{10}$$

In this fashion, the correct "instance-label" matchings are aggregately and progressively identified. And under the ideal situation, the label with weight 1 (i.e., $z_{rs} = 1$) indicates we have identified the ground-truth label successfully.

Data Sets	EXP*	FEA*	CL*	AVG-CL*
Vehicle	846	18	4	-
Abalone	4177	7	29	-
Satimage	6345	36	7	-
Nuswide	79216	500	81	-
Lost	1122	108	16	2.33
MSRCv2	1758	48	23	3.16
FG-NET	1002	262	99	7.48
BirdSong	4998	38	13	2.18
SoccerPlayer	17472	279	171	2.09
YahooNews	22991	163	219	1.91

Table 1: Characteristics of the synthetic and real-world data sets

4 Experiments

4.1 Experimental Settings

To evaluate the performance of the proposed D-GAP method, we implement experiments on four UCI data sets and six real-world data sets: **(1) Synthetic data sets**. Under different configurations of two controlling parameters (i.e. p and r), four UCI data sets generate 84 $(7 \times 3 \times 4)$ synthetic data sets [Cour *et al.*, 2011], where $p \in \{0.1, 0.2, \ldots, 0.7\}$ is the proportion of partial-label examples and $r \in \{1, 2, 3\}$ is the number of false candidate labels. **(2) Real-world data sets**. These data sets are collected from four task domains: (A) *Facial Age Estimation* [FG-NET]; (B) *Object Classification* [MSRCv2]; (C) *Bird Sound Classification* [BirdSong]; (D) *Automatic Face Naming* [Lost] [SoccerPlayer] [YahooNews]. Table 1 summarize the characteristics of these data sets, including the number of examples (**EXP***), the number of features (**FEA***), and the whole/average number of class labels (**CL*/AVG-CL***).

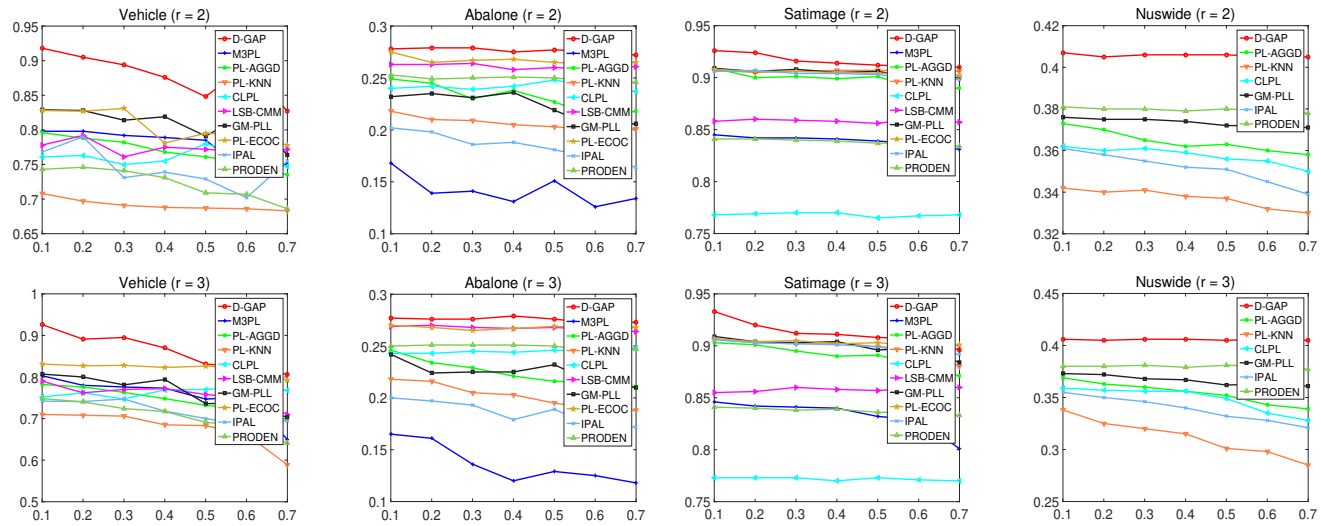

Figure 2: The classification accuracy of each comparing approach on synthetic data sets under different configurations ($r = \{2, 3\}$ and $p \in \{0.1, 0.2, \ldots, 0.7\}$). Due to page limit, experimental results under $r = 1$ are presented in Technical Appendix. The experimental results of some approaches on *Nuswide* data set are not illustrated since their time consumptions are over one day.

	Lost	MSRCv2	YahooNews	BirdSong	SoccerPlayer	FG-NET
D-GAP	**0.900±0.011**	**0.770±0.010**	**0.883±0.015**	0.840±0.005	**0.775±0.008**	**0.194±0.017**
PL-KNN	0.615±0.036 •	0.616±0.006 •	0.692±0.010 •	0.772±0.021 •	0.492±0.015 •	0.173±0.017 •
CLPL	0.894±0.005 •	0.656±0.010 •	0.834±0.002 •	0.822±0.004 •	0.680±0.010 •	0.158±0.018 •
LSB-CMM	0.721±0.010 •	0.524±0.007 •	0.872±0.001 •	0.716±0.014 •	0.704±0.002 •	0.138±0.019 •
IPAL	0.840±0.041 •	0.714±0.015 •	0.823±0.008 •	0.833±0.030 •	0.673±0.014 •	0.158±0.024 •
M3PL	0.860±0.006 •	0.732±0.025 •	0.870±0.002 •	0.855±0.030 ○	0.761±0.010 •	0.127±0.013 •
PL-ECOC	0.851±0.013 •	0.555±0.030 •	0.862±0.007 •	**0.886±0.014** ○	0.671±0.003 •	0.132±0.019 •
PL-AGGD	0.875±0.041 •	0.743±0.028 •	0.867±0.009 •	0.873±0.015 ○	0.672±0.006 •	0.183±0.017 •
GM-PLL	0.881±0.005 •	0.770±0.013 •	0.705±0.612 •	0.834±0.010 •	0.668±0.003 •	0.186±0.021 •
PRODEN	0.840±0.008 •	0.459±0.026 •	0.553±0.012 •	0.687±0.016 •	0.503±0.012 •	0.156±0.024 •

Table 2: Training accuracy (mean ± std) of each comparing algorithm on real-world data sets, where the best performance is on bold. •/○ indicates that D-GAP is statistically superior / inferior to the comparing algorithm on each data set (pairwise t-test at 0.05 significant level).

	Lost	MSRCv2	YahooNews	BirdSong	SoccerPlayer	FG-NET
D-GAP	**0.756±0.038**	**0.573±0.044**	**0.676±0.012**	0.717±0.043	**0.573±0.011**	**0.078±0.015**
PL-KNN	0.424±0.030 •	0.448±0.012 •	0.457±0.009 •	0.614±0.024 •	0.497±0.004 •	0.039±0.008 •
CLPL	0.735±0.024 •	0.413±0.020 •	0.462±0.009 •	0.632±0.017 •	0.368±0.004 •	0.063±0.017 •
LSB-CMM	0.707±0.019 •	0.456±0.008 •	0.648±0.015 •	0.717±0.024 •	0.525±0.006 •	0.059±0.008 •
IPAL	0.726±0.041 •	0.523±0.025 •	0.667±0.014 •	0.708±0.014 •	0.547±0.014 •	0.057±0.023 •
M3PL	0.732±0.035 •	0.521±0.030 •	0.655±0.010 •	0.709±0.010 •	0.446±0.013 •	0.037±0.025 •
PL-ECOC	0.703±0.052 •	0.505±0.027 •	0.662±0.010 •	**0.740±0.016** ○	0.537±0.020 •	0.040±0.018 •
PL-AGGD	0.748±0.043 •	0.503±0.039 •	0.634±0.005 •	0.730±0.017 ○	0.535±0.014 •	0.070±0.023 •
GM-PLL	0.737±0.043 •	0.530±0.019 •	0.629±0.007 •	0.663±0.010 •	0.549±0.009 •	0.065±0.021 •
PRODEN	0.563±0.055 •	0.351±0.026 •	0.514±0.013 •	0.665±0.026 •	0.495±0.013 •	0.061±0.027 •

Table 3: Test accuracy (mean ± std) of each comparing algorithm on real-world data sets, where the best performance is on bold. •/○ indicates that D-GAP is statistically superior / inferior to the comparing algorithm on each data set (pairwise t-test at 0.05 significate level).

Meanwhile, we employ nine baselines for comparative studies: **PL-KNN** [Hullermeier and Beringer, 2005], **CLPL** [Cour *et al.*, 2011], **LSB-CMM** [Liu and Dietterich, 2012], **IPAL** [Zhang and Yu, 2015], **M3PL** [Yu and Zhang, 2016], **PL-ECOC** [Zhang *et al.*, 2017], **PRODEN** [Lv *et al.*, 2020] and **GM-PLL** [Lyu *et al.*, 2021], where the configured pa-

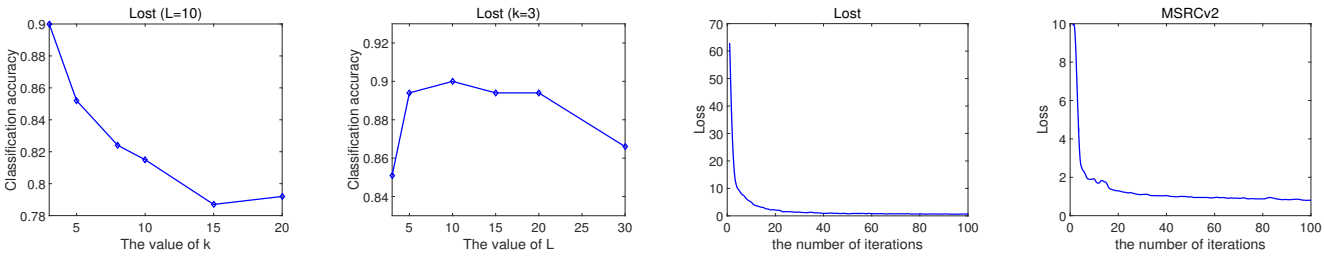

Figure 3: (a-b) The performance changes as k and L increase on *Lost* data set. (c-d) The convergence curve on *Lost* and *MSRCv2* data sets.

rameters are set via the suggestions in respective literatures.

In our experiments, we focus on how to select more accurate "instance-label" matchings during the training process, thus our experiments are biased on evaluating the disambiguation capability of learning model. With regard to the predictions for unseen examples, we just maximize the weighted voting results of their k-nearest neighbors from training examples like [Zhang and Yu, 2015]. We adopt ten-fold cross-validation to train the model and record the experimental results in Figure 2, Table 2 and 3.

4.2 Experimental Results

Synthetic Data Sets

Figure 2 illustrates the accuracy of each comparing method on 84 synthetic data sets as p increases from 0.1 to 0.7 with the step-size of 0.1. Out of 756 (9 approaches $\times$ 84 data sets) statistical comparisons show that:

- Among these comparing approaches, D-GAP achieves superior performance against LSB-CMM, CLPL, PL-AGGD and M3PL in all cases. And compared with PL-KNN, PL-ECOC, IPAL and GM-PLL, it also achieves superior performance in 97.4%, 93.2%, 99.2% and 96.7% cases, respectively.
- Among these synthetic data sets, D-GAP outperforms all comparing approaches on *Vehicle*, *Satimage* and *Nuswide* data sets. Especially for the large-scale *Nuswide* data set, D-GAP also shows superior performance in terms of both effectiveness and efficiency.
- Overall, among 756 statistical comparisons, D-GAP outperforms other approaches in 747 cases and achieves comparable performance in 5 cases (total: 99.4%).

Real-World Data Sets

The comparison results on real-world data sets are reported in Table 3-2. According to Table 3-2, it is clear to observe that:

- Compared with the approaches with different disambiguation strategies, D-GAP outperforms averaging-based approaches in 97.9% cases, identification-based approaches in 91.7% cases and disambiguation-free approaches in 83.3% cases, respectively.
- For different data sets, D-GAP achieves the best performance on most data sets, including *Lost*, *MSRCv2*, *SoccerPlayer*, *YahooNews* and *FG-NET* data sets.
- Among 108 statistical comparisons, D-GAP outperforms other comparing methods in 99 cases and achieves comparable performance in 6 cases (total: 97.2%).

4.3 Computational Complexity

Due to page limit, we illustrates the time complexity comparison between D-GAP and other comparing approaches in Technical Appendix. Our proposed D-GAP is **highly efficient**, and we attribute its efficiency to two aspects: **(i)** The GAT operation is efficient. The single attention operation can be parallelized across all edges and nodes, and multi-head attention can also be parallelized due to its fully independence. Thus, the time complexity can be directly expressed as $O(nd^{(1)}d^{(1)'} + nkd^{(1)'})$. **(ii)** The employed "instance-label" relationship is efficient. The previous *matching consistency relationship* with complexity of $O((\sum_n |\mathcal{S}_i|)^2)$ is replaced by *matching affinity relationship* with complexity of $O(\sum_n |\mathcal{S}_i|)$, which significantly reduces the computational cost and improves the efficiency of learning model.

4.4 Parameter and Convergence Analysis

We study the sensitivity of D-GAP with respect to its two parameters: k and L. Figure 3(a)-(b) shows the performance of D-GAP under different parameter configurations on *Lost* data set. From Figure 3, we can find that the values of k and L have great influence on the performance of the proposed framework. We set k among $\{3, 5, 8, 10, 15, 20\}$ and L among $\{3, 5, 10, 20, 30\}$ via cross-validation. Meanwhile, we conduct the convergence analysis of D-GAP on both *Lost* and *MSRCv2* data sets. Figure 3(c)-(d) illustrate the convergence curves of the optimization process of D-GAP. We can observe that the loss gradually decreases as t increases. Therefore, the convergence of D-GAP is empirically demonstrated.

5 Conclusion

In this paper, we proposed a DeepGNN-based graph matching PLL approach, where both DeepGNN and GM are employed to model a robust "instance-label" matching affinity relationship for model induction. Compared with previous PLL approaches, our employed matching affinity relationship is learned from structural data representations, which can guide the model to obtain more accurate "instance-label" matchings. Meanwhile, such matching affinity relationship alleviates the high computation consumption problem in previous graph matching PLL approaches, which extends the learning power to large-scale data sets. Extensive experiments have also demonstrated the effectiveness and efficiency of our proposed approach.

Acknowledgements

This work was supported in part by the Fundamental Research Funds for the Central Universities (No. 2020YJS036), the National Natural Science Foundation of China (No. 61872032, No. 62072027, No. 62076021), the Beijing Natural Science Foundation(No. 4202058, No. 4202057, No. 4202060), and in part by the National Key Research and Development Project (No. 2018AAA0100300).

References

[Chen *et al.*, 2017] C. Chen, V. Patel, and R. Chellappa. Learning from ambiguously labeled face images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(7):1653–1667, 2017.

[Cour *et al.*, 2009] T. Cour, B. Sapp, C. Jordan, and B. Taskar. Learning from ambiguously labeled images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 919–926, 2009.

[Cour *et al.*, 2011] T. Cour, B. Sapp, and B. Taskar. Learning from partial labels. *Journal of Machine Learning Research*, 12(5):1501–1536, 2011.

[Feng and An, 2018] L. Feng and B. An. Leveraging latent label distributions for partial label learning. In *International Joint Conference on Artificial Intelligence*, pages 2107–2113, 2018.

[Feng and An, 2019] L. Feng and B. An. Partial label learning by semantic difference maximization. In *International Joint Conference on Artificial Intelligence*, pages 2294–2300, 2019.

[Hullermeier and Beringer, 2005] E. Hullermeier and J. Beringer. Learning from ambiguously labeled examples. *International Symposium on Intelligent Data Analysis*, 10(5):168–179, 2005.

[Jin and Ghahramani, 2003] R. Jin and Z. Ghahramani. Learning with multiple labels. In *Advances in Neural Information Processing Systems*, pages 921–928, 2003.

[Liu and Dietterich, 2012] L. Liu and T. Dietterich. A conditional multinomial mixture model for superset label learning. In *Advances in Neural Information Processing Systems*, pages 548–556, 2012.

[Liu *et al.*, 2018] S. Liu, C. Chen, Y. Lu, F. Ouyang, and B. Wang. An interactive method to improve crowdsourced annotations. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):235–245, 2018.

[Luo and Orabona, 2010] J. Luo and F. Orabona. Learning from candidate labeling sets. In *Advances in Neural Information Processing Systems*, pages 1504–1512, 2010.

[Lv *et al.*, 2020] J. Lv, M. Xu, L. Feng, G. Niu, X. Geng, and M. Sugiyama. Progressive identification of true labels for partial-label learning. In *International Conference on Machine Learning*, pages 6500–6510, 2020.

[Lyu *et al.*, 2020a] G. Lyu, S. Feng, Y. Li, Y. Jin, G. Dai, and C. Lang. Hera: Partial label learning by combining heterogeneous loss with sparse and low-rank regularization. *ACM Transactions on Intelligent Systems and Technology*, 11(3), 2020.

[Lyu *et al.*, 2020b] G. Lyu, S. Feng, T. Wang, and C. Lang. A self-paced regularization framework for partial-label learning. *IEEE Transactions on Cybernetics*, pages 1–13, 2020.

[Lyu *et al.*, 2021] G. Lyu, S. Feng, T. Wang, C. Lang, and Y. Li. Gm-pll: graph matching based partial label learning. *IEEE Transactions on Knowledge and Data Engineering*, 33:521–535, 2021.

[Vaswani *et al.*, 2017] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[Veličković *et al.*, 2018] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *International Conference on Learning Representations*, 2018.

[Wang *et al.*, 2019a] D. Wang, L. Li, and M. Zhang. Adaptive graph guided disambiguation for partial label learning. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 83–91, 2019.

[Wang *et al.*, 2019b] Q. Wang, Y. Li, and Z. Zhou. Partial label learning with unlabeled data. In *International Joint Conference on Artificial Intelligence*, pages 3755–3761, 2019.

[Wang *et al.*, 2020] H. Wang, W. Liu, Y. Zhao, T. Hu, K. Chen, and G. Chen. Learning from multi-dimensional partial labels. In *International Joint Conference on Artificial Intelligence*, pages 2943–2949, 2020.

[Wang *et al.*, 2022] H. Wang, R. Xiao, S. Li, L. Feng, G. Niu, G.n Chen, and J. Zhao. PiCO: Contrastive label disambiguation for partial label learning. In *International Conference on Learning Representations*, 2022.

[Wu and Zhang, 2018] X. Wu and M. Zhang. Towards enabling binary decomposition for partial label learning. In *International Joint Conference on Artificial Intelligence*, pages 2868–2874, 2018.

[Yu and Zhang, 2016] F. Yu and M. Zhang. Maximum margin partial label learning. *Machine Learning*, 4(106):573–593, 2016.

[Zeng *et al.*, 2013] Z. Zeng, S. Xiao, K. Jia, T. Chan, S. Gao, D. Xu, and Y. Ma. Learning by associating ambiguously labeled images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 708–715, 2013.

[Zhang and Yu, 2015] M. Zhang and F. Yu. Solving the partial label learning problem: an instance-based approach. In *International Joint Conference on Artificial Intelligence*, pages 4048–4054, 2015.

[Zhang *et al.*, 2017] M. Zhang, F. Yu, and C. Tang. Disambiguation-free partial label learning. *IEEE Transactions on Knowledge and Data Engineering*, 29(10):2155–2167, 2017.