Duplicate Multi-modal Entities Detection with Graph Contrastive Self-training Network

Shuyun Gu, Xiao Wang, and Chuan Shi✉

Beijing University of Posts and Telecommunications
gsy793048702@163.com
{xiaowang,shichuan}@bupt.edu.cn

Abstract. Duplicate multi-modal entities detection aims to find highly similar entities from massive entities with multi-modal information, which is a basic task in many applications and becoming more important and urgent with the development of Internet and e-commerce platforms. Traditional methods employ machine learning or deep learning on feature embedding extracted from multi-modal information, which ignores the correlation among entities and modals. Inspired by the popular Graph Neural Networks (GNNs), we can analyze the multi-relation graph of entities constructed from their multi-modal information with GNN. However, this solution still faces the extreme label sparsity challenge, particularly in industrial applications. In this work, we propose a novel graph contrastive self-training network model, named **CT-GNN**, for duplicate multi-modal entities detection with extreme label sparsity. With the multi-relation graph of entities constructed from multi-modal features of entities with KNN, we first learn the preliminary node embeddings with existing GNN, e.g., GCNs. To alleviate the problem of extremely sparse labels, we design a layer contrastive module to effectively exploit implicit label information, as well as a pseudo labels extension module to determine label boundary. In addition, graph structure learning is introduced to refine the structure of the multi-relation graph. A uniform optimization framework is designed to seamlessly integrate these three components. Sufficient experiments on real datasets, in comparison with SOTA baselines, well demonstrate the effectiveness of our proposed method.

Keywords: Duplicate enetites · Graph learning · Self-supervised learning · Self-training learning.

1 Introduction

Duplicate entities detection, finding all two highly similar or even identical entities from massive data, has become a common and important problem, e.g., face matching [23] and user alignment [37]. In e-commerce scenarios, this problem is more challenging, because entities often have multi-modal features, e.g., texts, images and even videos. Duplicate multi-modal entities detection has urgent and realistic needs, which provides the basic function in many applications. For instance, as shown in Figure 1 (e.g., similar images and descriptions), some store managers defraud illegal subsidies from e-commerce platforms through registering inveracious duplicate stores. Thus, detecting these duplicate multi-modal entities is an important task for combating fake

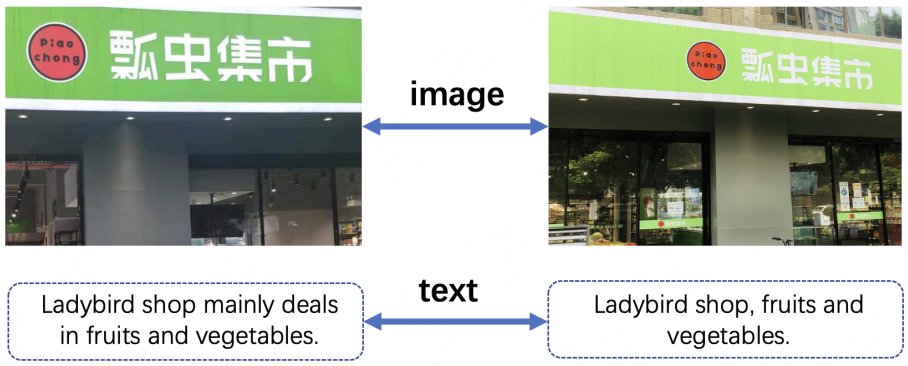


Fig. 1: An example of duplicate multi-modal stores. The images and texts are the outlines and description information of the stores respectively. Because of their highly similar features, they are determined to be duplicate entities.

information and saving cost. In order to solve the above problems, traditional industrial solutions are more inclined to machine learning or deep learning methods [1], making independent decisions for each pair of duplicate entities based on feature engineering, which measures the similarity of entities by feature extraction and feature combination. Nevertheless, these methods have two disadvantages. (1) They do not depict the correlation between entities explicitly. The correlation between entities is important prior knowledge, and thus ignoring them may result in performance degradation. (2) The associations between multiple modals are not considered. Different modals depict entities from distinct perspectives, and thus considering associations between multiple modals may benefit for characterizing entities more accurately.

For solving above disadvantages, a direct solution is to construct a multi-relation graph through employing K-Nearest Neighbors (KNN) [4] on multi-modal features, and then the popular GNN [35,10] can be applied to exploit the structure relations among entities and modals. However, because of the semi-supervised paradigm, this solution faces the extreme labels (i.e., known duplicate entity pairs) sparsity challenge, especially on industrial applications. For instance, an e-commerce platform has tens of millions of offline stores, the proportion of duplicate stores is relatively small. The labels we can obtain also rely on manual annotation. Due to very high labor costs, the known label data may only account for 5% of the duplicate stores, and thus the label data may be no more than 0.05% of all stores.

How to deal with the extreme labels sparsity challenge? A common way is self-supervised learning [34,32] or self-training learning [12,13]. However, these strategies are not easily applied to our problem settings and a single strategy is also not sufficient to solve the above challenge. For self-supervised learning, the conventional process is to generate two augmented views based on a graph, and emphasize the consistency between different views of the same node (i.e., entity) [26,36]. However, in our duplicate multi-modal entities detection scenario, this method of emphasizing the consistency of the same node does not have much benefit to mine whether two nodes is duplicate.

As for self-training learning, its main idea is to generate some pseudo labels and train jointly with real labels. In the scenario of duplicate entities detection, we can generate pseudo labels by similarity measurement. However, pseudo labels with different similarities have different influences on the model, so it is difficult to determine a rational threshold for pseudo labels. More importantly, individual self-supervised or self-training learning may be not sufficient to solve the extreme label sparsity, especially in real industry scenorio, which motivates us to effectively integrate more strategies to solve this challenge.

In this paper, we propose a novel graph contrastive self-training network model **CT-GNN**, which solves the challenge of extreme label sparsity by means of seamlessly integrating self-supervised learning and self-training learning. Specifically, after multi-modal feature extraction through pre-training model [24,5], we build a multi-relation graph with KNN method, and learn the node embeddings with GCNs [10] model. In order to exploit implicit label information among graph structure, we propose a novel layer-contrastive module by using the strategy of multiple random walks [16,18,11] to find the others with the most frequency as positive nodes in the topology. At the same time, in order to more fully utilize the feature information of entities, we design a self-training module with a delicate boundary distance, which distinguishes the optimization intensity of labels with different similarities. In addition, the graph structure learning process is introduced to automatically adjust graph structure for iterative representation learning. Finally, a joint optimization function is designed to seamlessly optimize above three components.

We summarize the contributions of this work as below:

- To our best knowledge, we are the first to study the problem of duplicate multi-modal entities detection with extreme label sparsity, which is a basic task in many applications and becoming more important and urgent with the development of Internet and e-commerce platforms.
- We propose a novel graph model named CT-GNN, which seamlessly integrates self-supervised learning, self-training learning and graph structure learning in a uniform optimization framework. In particular, some delicate designs in CT-GNN make it suitable for extreme label sparsity challenge, i.e., layer contrastive module with multiple random walk and self-training module with boundary distance.
- We evaluate CT-GNN by designing both various offline and online experiments. Compared to state-of-the-art alternatives, the improvements of CT-GNN are obvious up to 9.88% in Recall and 7.13% in Precision.

2 Related Work

2.1 Graph Neural Networks

In recent years, Graph Neural Networks (GNNs) has become an extremely important field, e.g., recommendation systems [30,8], fraud detection [15], which learn the node embedding by aggregating the features of neighborhoods [28]. GCNs [10] implements layer-wise propagation to learn the node embedding, and GAT [28] learns different attention scores for neighbors when aggregating neighborhood information. Meanwhile,

Some recent models [21] are proposed to deal with heterogeneous graphs which are more practical in reality. RGCN [21] propose to learn node embedding based on multi-relation neighborhoods. Additionally, HAN [31] leverages the attention mechanism under node-level and semantic-level in heterogeneous graphs.

2.2 Self-supervised Learning for GNNs

Recently, motivated by profound success in natural language processing [5] and computer vision [7], self-supervised contrastive learning based graph representation learning attracts considerable attention. Deep Graph Infomax (DGI) [29] learns unsupervised representations for nodes in attributed graphs by the mutual informaton-based learning from Deep InfoMax [9]. GMI [19] is proposed to contrast between center node and its local patch from node features and topological structure. Another line of graph contrastive learning approaches called global-global contrast [20] directly study the relationships between the global context representations of different samples as what metric learning does. In heterogeneous domain,DMGI [17] and HeCo [32] employs network schema and meta-path as two views to capture both of local and high-order structures, and performs the contrastive learning across them.

2.3 Self-training Learning for GNNs

Due to the pressure of sparse supervised signals, some researchers propose that GNNs are not completely suitable for graph semi-supervised learning tasks [13]. Self-training [12] strategy is to first train GNNs with the existing training sets, then selects the samples with high predicting probability as pseudo labels and add them to the training sets, and then continue to train GNNs. The samples selected by self-training should have similar features with the label samples, so that the robustness will be improved after expanding the training sets [38]. These methods have uniform constraint strength for all generated labels, but samples with different prediction probabilities should have different influence on the model.

3 THE PROPOSED MODEL

3.1 Notations & Definitions

Given entity set $U = \{u_1, u_2, ..., u_n\}$ and feature set $E = \{e_1, e_2, ..., e_n\}$, each entity $u_i \in U$ has a feature $e_i \in E$, in which n is the number of entities and e_i includes multi-modal feature $e_i = \{e_i^{(1)}, e_i^{(2)}, ..., e_i^{(v)}\}$, and v is the number of modals. Without lost of generality, we consider image feature E_{img} and text feature E_{text} in this paper. For these multi-modal features, we can obtain their initial feature (i.e., E_{img} and E_{img}). Through multi-relation graph construction and GNN method, we fully learn node embedding (i.e., $\boldsymbol{X^0}$). Our goal is to mine all suspicious duplicate entities through the similarity calculation of node embedding.

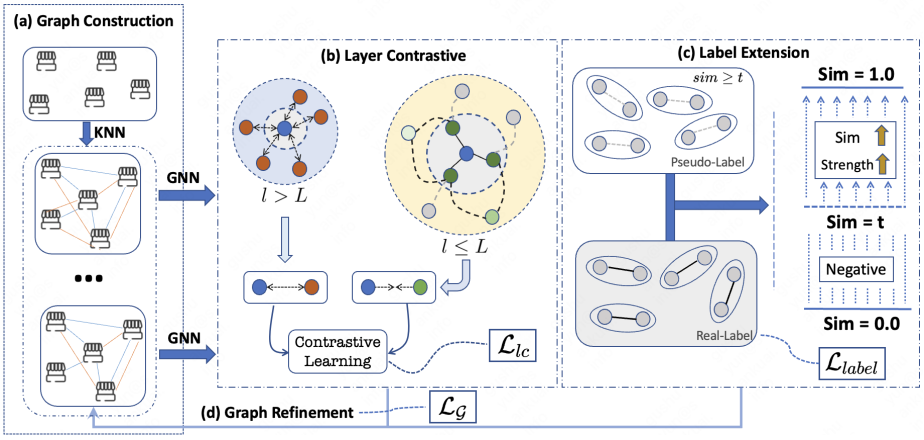


Fig. 2: The architecture of CT-GNN. (a) is the constructing process of multi-relation graph. (b) is the layer contrastive module through multiple random walks. (c) is the self-training learning considering boundary distance. (d) is the graph refinement process.

3.2 Overall Framework

Figure 2 shows the overall framework of our CT-GNN model. After constructing multi-relation graph by multi-modal features, we design a graph contrastive self-training network to solve the extreme label sparsity challenge, including layer contrastive learning, self-training with boundary distance and graph structure learning, in a uniform optimization framework. Concretely, after learning graph embedding with GCNs, we firstly design a layer contrastive module to sufficiently utilize self-supervised information, which uses multiple random walks to find the important neighbors of the central node as positive samples of contrastive learning. Then we introduce a self-training module to flexibly extend label information. Based on the similarity between entities, we smartly generate pseudo labels through a delicate boundary distance loss distinguishing the optimization intensity of labels with different similarities. Meanwhile, in order to refine the graph structure, we further design a graph structure refinement process during iteration.

3.3 Multi-relation Graph Construction

Firstly, we construct a multi-relation graph of entities under the multi-modal features with the KNN method. Without lost of generality, we consider image and text feature. ***Image Feature:*** we employ the pre-training image model VGG [24,22] to acquire the image feature vectors $\boldsymbol{X_{img}} \in \boldsymbol{R}^{n*d}$ and d represents feature dimension. ***Text Feature:*** we use the pre-training language model BERT [5,25] and convert all of the text information for each entity feature vector $\boldsymbol{X_{text}} \in \boldsymbol{R}^{n*d}$.

Based on $\boldsymbol{X_{img}}$ and $\boldsymbol{X_{text}}$, we can construct the K-Nearest Neighbor graph $\mathcal{G}_{img} = (\boldsymbol{A_{img}}, \boldsymbol{X_{img}})$ and $\mathcal{G}_{text} = (\boldsymbol{A_{text}}, \boldsymbol{X_{text}})$, where $\boldsymbol{A_{img}}$ and $\boldsymbol{A_{text}}$ are the adjacency matrix of KNN graphs under images and texts respectively. Specifically, under

$\boldsymbol{X_{img}}$ or $\boldsymbol{X_{text}}$, for each sample, we first find its top-K similar neighbors and set edges to connect it with its neighbors. There are many methods to calculate the similarity matrix $\boldsymbol{S} \in \boldsymbol{R}^{n*n}$ of samples. Here we list two common methods for building KNN graph,

– **Cosine Similarity:** It uses the cosine value of the angle between two vectors to measure the similarity:

$$\boldsymbol{S}_{ij} = \frac{\boldsymbol{x_i} \cdot \boldsymbol{x_j}}{|\boldsymbol{x_i}||\boldsymbol{x_j}|}. \tag{1}$$

– **Heat Kernel:** The similarity is calculated by Eq. (2) where t is the time parameter in heat conduction equation.

$$\boldsymbol{S}_{ij} = e^{-\frac{||\boldsymbol{x_i}-\boldsymbol{x_j}||^2}{t}}. \tag{2}$$

Here we uniformly choose the Cosine Similarity. By this way, we can obtain two graphs: $\mathcal{G}_{img}$ and $\mathcal{G}_{text}$, and then combine them to get a multi-relation graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$, in which $\mathcal{V}$ and $\mathcal{E}$ represent the node set (all entities) and edge set respectively, and $\mathcal{R} = \{image, text\}$ represent all relations between two nodes, i.e., the association between nodes in the image and text dimension.

3.4 Graph Embedding Learning

Now we have built the graph $\mathcal{G}$. The initial node feature are image and text feature, i.e., $\boldsymbol{X_{img}}$ and $\boldsymbol{X_{text}}$. We first learn from the idea of GNNs (e.g., GCNs [10]) to get the embedding of each type of feature, and then integrate them together. The generation method of embedding under image feature is as follows,

$$\boldsymbol{x_{i_{img}}}^{l+1} = \sum_{j \in \mathcal{N}_i^{img}} \frac{1}{\sqrt{|\mathcal{N}_i^{img}||\mathcal{N}_j^{img}|}} \boldsymbol{x_{j_{img}}^{l}}, \tag{3}$$

where $\boldsymbol{x_{i_{img}}}^{l+1}$ represents the embedding of node i at the $(l+1)^{th}$ layer under image feature. $\mathcal{N}_i^{img}$ represents the neighbor set of i (including the central node i) under the image feature. The embedding learning method under the text feature is the same as the image feature.

In order to obtain a comprehensive node embedding for subsequent graph learning, we fuse two types of embeddings by a function f, as follows,

$$\boldsymbol{x_i}^{l+1} = f(\boldsymbol{x_{i_{img}}}^{l+1}, \boldsymbol{x_{i_{text}}}^{l+1}), \tag{4}$$

where $\boldsymbol{x_i}^{l+1}$ is the embedding of node i after fusion. The common designs of f are concatenation, weighted sum and softmax [14], and we choose concatenation operation in this paper.

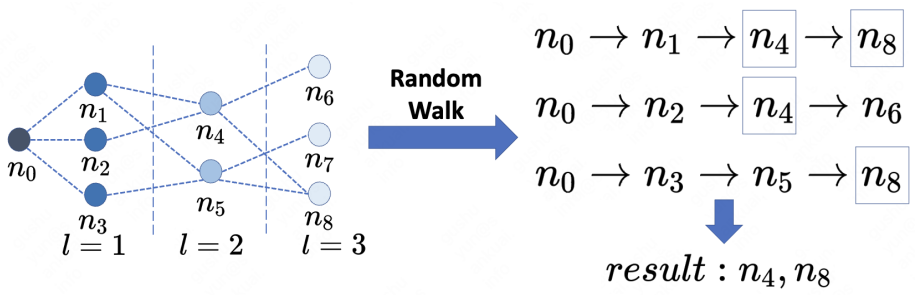


Fig. 3: Positive example selection in layer contrastive module. Three random walks are performed from the central node n_0, and $L = 3$ is limited, and the n_4 and n_8 appear the most times, i.e., n_4 and n_8 are taken as positive samples of n_0.

3.5 Self-supervised Learning with Layer Contrastive

In this section, we introduce a layer contrastive module to alleviate the problem of extreme sparse supervised signals. Traditional graph contrastive learning is to generate two augmented views based on a graph, and emphasize the consistency between different views of the same node, which does not match our goal of predicting whether two entities are duplicate. Because the duplicate entities usually meet the graph structure proximity principle, i.e., the two nodes that meet the duplication condition should be topologically highly related in the graph structure. Therefore, we design a layer contrastive learning module, which implements N random walks within L layer and finds the first top m nodes that appear more times. In this way, we can capture the neighbor (which can be multi-hop) nodes that are more important in the topology of the central node, and take them as positive samples in contrastive learning. As is shown in Figure 3, starting from node n_0, we generate three random walk paths ($L = 3$). Among all paths, n_4 and n_8 appear the most frequently. We think n_4 and n_8 have the strongest correlation with the central node n_0 in the topology, that is, they are most likely to be duplicate with n_0. Therefore, n_4 and n_8 are used as positive samples of n_0 in contrastive learning. As for the negative samples, we randomly select from outside the L^{th} layer.

Obviously, the strength of entity repeatability is inversely proportional to the layer and directly proportional to the similarity. Therefore, we hope to use features and layers to constrain embedded learning. After finding the positive and negative examples, we adopt the contrastive loss, InfoNCE [6], to maximize the agreement of positive pairs and minimize that of negative pairs. On this basis, we consider the different effects of layers and similarities on the optimization strength, and update the loss function, as below,

$$\mathcal{L}_{lc} = -\sum_{v\in\mathcal{V}}\sum_{l=1}^{L}\sum_{u^+\in nebor^l}\frac{sim(\boldsymbol{x_v},\boldsymbol{x_{u^+}})}{l^\gamma} log\frac{exp\{sim(\boldsymbol{x_v},\boldsymbol{x_{u^+}})/\tau\}}{\sum_{u^-\in\mathcal{N}}exp\{sim(\boldsymbol{x_v},\boldsymbol{x_{u^-}})/\tau\}}, \tag{5}$$

where $(\boldsymbol{x_v}, \boldsymbol{x_{u^+}})$ is the the positive pair and $(\boldsymbol{x_v}, \boldsymbol{x_{u^-}})$ is the negative pair. τ is a hyper-parameter, known as the $temperature$ coefficient in softmax. We can see that the optimization proportion of positive samples is proportional to the similarity and inversely proportional to the layer l (i.e., the distance between nodes), and the influence intensity of the l is controlled by an intensity coefficient γ.

3.6 Self-training Learning with Boundary Distance

In this section, we introduce a self-training module to further relieve the extreme lacking labels. Conventional methods calculate the similarity between entities, and judge whether are pseudo labels by a boundary (i.e. threshold). However, a popular fixed boundary is not retional, since the pseudo labels with different distances from the threshold have different effects in the optimization process. And thus we design a smart threshold with boundary distance.

We use R to represent the set of real labels and generate pseudo labels with the similarity of nodes. Specifically, we set a super parameter t, and then calculate the cosine similarity between nodes. For two nodes whose similarity is greater than or equal to t, we think they are likely to be duplicated, so we regard them as a pair of pseudo labels. We use P to represent the set of pseudo labels. The generation process of P is formalized as follows,

$$(u, v) \in P, sim(\boldsymbol{x_u}, \boldsymbol{x_v}) \geq t. \tag{6}$$

Obviously, this threshold t can be considered as the boundary between duplicate and non-duplicate entities, and the node similarities in the real labels are far away from t. In order to distinguish the optimization strength of pseudo labels with different distances from t, we design a semi-supervised normal form based on boundary distance, as follows,

$$\begin{aligned} \mathcal{L}_{label} = -\frac{1}{N} \sum_{\substack{(u,u^+)\in P\cup R \\ (v,v^-)\in Neg}} & (\frac{sim(\boldsymbol{x_u}, \boldsymbol{x_{u+}}) - t}{1-t})^{\alpha} \\ & log\{\sigma(sim(\boldsymbol{x_u}, \boldsymbol{x_{u^+}}) - sim(\boldsymbol{x_v}, \boldsymbol{x_{v^-}}))\}, \end{aligned} \tag{7}$$

in which Neg is the negative sample set. Negative samples are randomly selected from the set whose similarity is less than t. α and σ are the strength-control coefficient and a non-linear activation function, respectively. As shown in Eq. (7), the optimization strength of all pseudo labels (i.e. positive samples) is different, and it decreases with the increase of the distance from the boundary t.

3.7 Graph Structure Learning

In order to ensure the credibility of the graph structure in the model iteration process, we choose to refine the graph structure in each epoch. The method of reconstructing the graph is shown in Section 3.3.

Let the graph adjacency matrix of the last epoch be A_{pre}, and that of the current epoch be A_{new}. In order to constrain the stability of nodes embeddings during the

training process, we need to constrain the graph structure changes between the two epochs, which is as shown below,

$$\mathcal{L}_{\mathcal{G}} = ||A_{new} - A_{pre}||_2. \tag{8}$$

3.8 Joint Optimization

In order to combine the above modules, we jointly optimize the model, which is as below,

$$\mathcal{L} = \mathcal{L}_{lc} + \lambda_1 \mathcal{L}_{label} + \lambda_2 \mathcal{L}_{\mathcal{G}}, \tag{9}$$

in which λ_1 and λ_2 are hyperparameters to control the proportion of label extension module and graph refinement module, respectively.

4 EXPERIMENTS

4.1 Datasets

Three datasets are utilized in our evaluation, and can be described as follows:

- **M Stores**: It is an offline stores datasets of Mplatform. It includes multi-modal information (e.g., store outline images and store names). We extracted partial data, including 111,635 entities (stores).
- **M commodities**: M platform maintains a large number of commodities online. When they are released, sellers need to upload multi-modal information of the commodities, such as the appearance images and the commodity names. We extract 125,320 entities (commodities) for experiment.
- **T commodities**: T has a large number of commodities for users to choose. We can obtain multi-modal information of commodities including the image, name and attribute, etc. We obtain 69,911 entities (commodities).

4.2 Experimental Settings

Baseline. We compare CT-GNN with several state-of-the-art methods. The baseline can be divided into three categories: traditional industrial models, graph models and multi-modal models. The traditional industrial models include: XGBoost [2] and MLP [27]. The graph models include GCN [10], GAT [28], RGCN [21] and HAN [31]. The multi-modal models include ITA [33] and HVPNet [3].

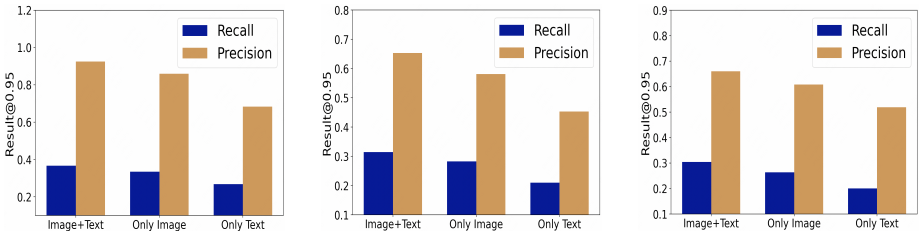
Table 1: The **R** (Recall) and **P** (Precision) results under different threshold (**t**) on three datasets.

Dataset	t	M	XGB	MLP	GCN	RGCN	ITA	HVPNet	CT-GNN	Improv.
M Stores	**0.85**	**R**	0.3633	0.3152	0.3578	0.3756	0.3312	0.3621	**0.4122**	9.74%
		P	0.7429	0.7093	0.8033	0.8154	0.7645	0.8392	**0.8567**	2.09%
	0.90	**R**	0.3422	0.2978	0.3420	0.3693	0.3247	0.3529	**0.3923**	6.23%
		P	0.7803	0.7432	0.8358	0.8492	0.7850	0.8415	**0.8625**	1.57%
	0.95	**R**	0.3137	0.2765	0.3197	0.3128	0.2933	0.3367	**0.3670**	5.52%
		P	0.8232	0.7938	0.8552	0.8737	0.8022	0.8639	**0.9245**	5.81%
M commodities	**0.85**	**R**	0.3278	0.3024	0.3538	0.3793	0.3488	0.3725	**0.3928**	3.56%
		P	0.5281	0.4933	0.5324	0.5633	0.5384	0.5528	**0.5933**	5.33%
	0.90	**R**	0.3055	0.2933	0.3228	0.3387	0.3176	0.3495	**0.3582**	2.50%
		P	0.5468	0.5162	0.5533	0.5966	0.5539	0.5932	**0.6124**	2.65%
	0.95	**R**	0.2873	0.2734	0.2956	0.3034	0.2753	0.3008	**0.3143**	3.59%
		P	0.5884	0.5478	0.6055	0.6374	0.5976	0.6123	**0.6534**	2.51%
T commodities	**0.85**	**R**	0.3256	0.2833	0.3277	0.3328	0.3165	0.3245	**0.3547**	6.58%
		P	0.5218	0.4908	0.5329	0.5587	0.5180	0.5267	**0.5853**	4.76%
	0.90	**R**	0.2945	0.2678	0.3002	0.3086	0.2858	0.3224	**0.3290**	2.05%
		P	0.5468	0.5100	0.5591	0.5933	0.5265	0.5371	**0.6356**	7.13%
	0.95	**R**	0.2449	0.2208	0.2675	0.2773	0.2508	0.2729	**0.3047**	9.88%
		P	0.5934	0.5736	0.6093	0.6533	0.5732	0.6262	**0.6603**	1.07%

4.3 Performance Evaluation

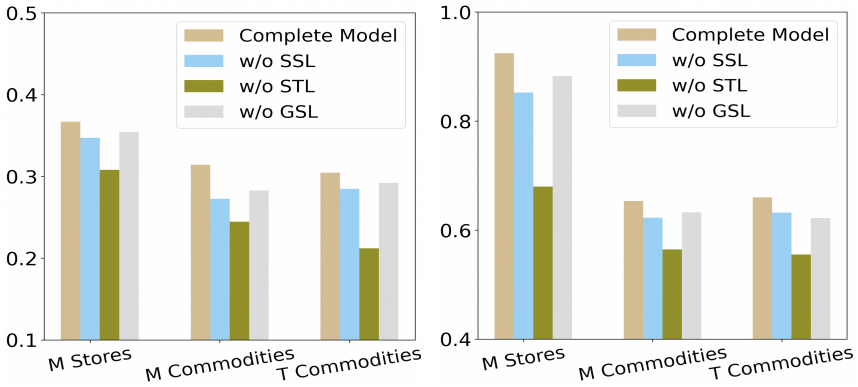
In this section, we empirically compare CT-GNN with several state-of-art alternatives and analyze the experimental results. In order to fully evaluate the results, we take t = 0.85, 0.90 and 0.95 respectively. As shown in Table 1, the following major observations can be made.

Obviously, CT-GNN achieves the best performance in the duplicate entities detection task on all datasets. Compared with the second best result, the improvement is up to 9.88% in Recall and 7.13% in Precision. This phenomenon is reasonable. Compared with non-GNN-based methods (i.e., XGBoost and MLP), GNN-based methods can mine high-order information through graph structure. Compared with other GNN-based models (i.e., GCN and RGCN), our model alleviates extreme label sparsity challenge with self-supervised module and self-training module. Note that, compared with the multi-modal based methods (i.e., ITA and HVPNet), our model achieve significant performance improvement because of capturing the correlation among entities and modals.



(a) M Stores (b) M commodities (c) T commodities

Fig. 4: The result comparison of removing different information, i.e., removing images or texts. (we set $t = 0.95$).



(a) Recall (b) Precision

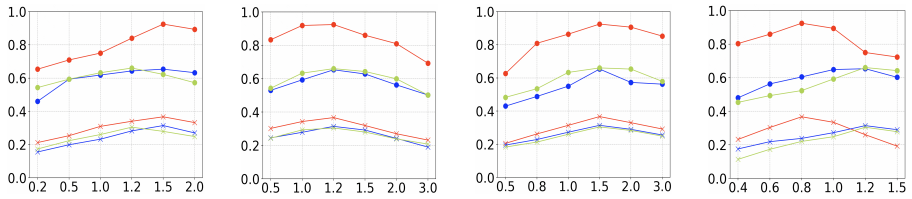
Fig. 5: The comparison result of removing different modules. SSL is Self-supervised learning. STL is Self-training learning. GSL is Graph structure learning (we set $t = 0.95$).

4.4 Ablation Analysis

In order to extensively validate our model, we conduct the following ablation experiments.

Firstly, we explore the impact of multi-modal information, i.e., images and texts. In order to explore the importance of each kind of information in the model, we remove images and texts respectively, and then test the experimental results. As shown in Figure 4, without either images or texts, the experimental results are significantly lower than complete CT-GNN model. Through the analysis of experimental results, we found that the two kinds of information, image and text, have different influences in the duplicate entities detection. For example, as shown in Figure 4(a), the prediction accuracy without image information is far lower than that without text information. It indicates that image information has a greater impact on the detection results.

Secondly, we explore the importance of three modules (i.e., self-supervised learning, self-training learning and graph structure learning) in our model. We remove them



(a) γ (b) α (c) λ_1 (d) λ_2

Fig. 6: Impact of γ, α, λ_1 and λ_2 to the result under three datasets. The red line, blue line and green line respectively represent M stores, M commodities and T commodities. $\times$ is Recall and $\bullet$ is Precision (we set $t = 0.95$).

respectively, and then compare the experimental results. As shown in Figure 5, we find that after removing the self-training module, the decline of experimental results is the most significant, followed by removing the self-supervised and graph structure learning module. It indicates that all three modules play important roles in the result of detection, and the self-training module has the greatest influence.

4.5 Hyperparameter Analysis

In this section, we analyze several important parameters in the model. First, we analyze the two strength control factors γ and α set in layer contrastive module and label extension module respectively. As our model jointly optimizes the three modules with hyperparameter λ_1 and λ_2 in Eq. (9), we then explore the effect of them on the final performance, and their change trends are shown in Figure 6.

Firstly, we observe two strength coefficients, i.e., γ in Eq. (5) and α in Eq. (7). we tune γ in $\{0.2, 0.5, 1.0, 1.2, 1.5, 2.0\}$ and view the corresponding results. As shown in Figure 6(a), the index of the experiment increases gradually with γ from 0.2 to 1.5, and reaches the peak at $\gamma = 1.5$. Then it shows a downward trend. Then we tune α in $\{0.5, 1.0, 1.2, 1.5, 2.0, 3.0\}$ and observe the results. As shown in Figure 6(b), the experimental results can get the maximum value at $\alpha = 1.2$, and decrease at both sides of 1.2. The change trend of this result also verifies the rationality of setting strength parameters γ in Eq. (5) and α in Eq. (7).

Next, we evaluate the impact of λ_1 and λ_2. We perform experiments on three datasets and tune λ_1 in $\{0.5, 0.8, 1.0, 1.5, 2.0, 3.0\}$ and λ_2 in $\{0.4, 0.6, 0.8, 1.0, 1.2, 1.5\}$. As we can see in Figure 6(c) that when $\lambda_1 = 1.5$, the best experimental results can be obtained on three datasets, and the two sides of $\lambda_1 = 1.5$ show a downward trend, in which we can infer that the label extension module is more important than the layer contrastive module. As for λ_2 shown in Figure 6(d), the best performance is achieved when $\lambda_2 = 0.8$ on M Stores dataset. While on the other two datasets, the best performance is achieved at $\lambda_2 = 1.2$. It can be seen that the influence of the scale parameter λ_2 of graph refinement on different datasets is different. We speculate that in M Stores dataset, the features of entities are relatively dispersed and the initial graph is relatively accurate. However, in other datasets, the features of entities are relatively dense and the reliability of the graph is low, so the influence of λ_2 is greater.

5 CONCLUSION

In this paper, we study the multi-modal entities detection under extreme label sparsity. We propose a novel model named CT-GNN, which can alleviate the extreme labels sparsity challenge by two module, i.e., self-supervised learning with layer contrastive and self-training learning with boundary distance. Meanwhile, graph structure learning is introduced to stabilize learning performance. We carry out comprehensive experiments, and the results demonstrate that CT-GNN has significant performance improvement on three datasets compared with SOTA models.

6 ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their constructive feedback. This work is supported in part by the National Natural Science Foundation of China (No. 62192784, U1936104, U20B2045, 62172052, 62002029).

References

1. Belgiu, M., Drăguţ, L.: Random forest in remote sensing: A review of applications and future directions. ISPRS journal of photogrammetry and remote sensing **114**, 24–31 (2016)
2. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. pp. 785–794 (2016)
3. Chen, X., Zhang, N., Li, L., Yao, Y., Deng, S., Tan, C., Huang, F., Si, L., Chen, H.: Good visual guidance makes a better extractor: Hierarchical visual prefix for multimodal entity and relation extraction. arXiv preprint arXiv:2205.03521 (2022)
4. Cunningham, P., Delany, S.J.: K-nearest neighbour classifiers-a tutorial. ACM Computing Surveys (CSUR) **54**(6), 1–25 (2021)
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
6. Gutmann, M., Hyvärinen, A.: Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics. pp. 297–304. JMLR Workshop and Conference Proceedings (2010)
7. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9729–9738 (2020)
8. He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., Wang, M.: Lightgcn: Simplifying and powering graph convolution network for recommendation. In: Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval. pp. 639–648 (2020)
9. Hjelm, R.D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Bachman, P., Trischler, A., Bengio, Y.: Learning deep representations by mutual information estimation and maximization. arXiv preprint arXiv:1808.06670 (2018)
10. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)

11. Lawler, G.F., Limic, V.: Random walk: a modern introduction, vol. 123. Cambridge University Press (2010)
12. Li, Q., Han, Z., Wu, X.M.: Deeper insights into graph convolutional networks for semi-supervised learning. In: Thirty-Second AAAI conference on artificial intelligence (2018)
13. Liu, H., Hu, B., Wang, X., Shi, C., Zhang, Z., Zhou, J.: Confidence may cheat: Self-training on graph neural networks under distribution shift. In: Proceedings of the ACM Web Conference 2022. pp. 1248–1258 (2022)
14. Liu, W., Wen, Y., Yu, Z., Yang, M.: Large-margin softmax loss for convolutional neural networks. arXiv preprint arXiv:1612.02295 (2016)
15. Liu, Z., Chen, C., Yang, X., Zhou, J., Li, X., Song, L.: Heterogeneous graph neural networks for malicious account detection. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management. pp. 2077–2085 (2018)
16. Nikolentzos, G., Vazirgiannis, M.: Random walk graph neural networks. Advances in Neural Information Processing Systems **33**, 16211–16222 (2020)
17. Park, C., Kim, D., Han, J., Yu, H.: Unsupervised attributed multiplex network embedding. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 5371–5378 (2020)
18. Pearson, K.: The problem of the random walk. Nature **72**(1865), 294–294 (1905)
19. Peng, Z., Huang, W., Luo, M., Zheng, Q., Rong, Y., Xu, T., Huang, J.: Graph representation learning via graphical mutual information maximization. In: Proceedings of The Web Conference 2020. pp. 259–270 (2020)
20. Qiu, J., Chen, Q., Dong, Y., Zhang, J., Yang, H., Ding, M., Wang, K., Tang, J.: Gcc: Graph contrastive coding for graph neural network pre-training. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 1150–1160 (2020)
21. Schlichtkrull, M., Kipf, T.N., Bloem, P., Berg, R.v.d., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: European semantic web conference. pp. 593–607. Springer (2018)
22. Sengupta, A., Ye, Y., Wang, R., Liu, C., Roy, K.: Going deeper in spiking neural networks: Vgg and residual architectures. Frontiers in neuroscience **13**, 95 (2019)
23. Shen, S., Li, W., Zhu, Z., Huang, G., Du, D., Lu, J., Zhou, J.: Structure-aware face clustering on a large-scale graph with 107 nodes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9085–9094 (2021)
24. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
25. Tenney, I., Das, D., Pavlick, E.: Bert rediscovers the classical nlp pipeline. arXiv preprint arXiv:1905.05950 (2019)
26. Tian, Y., Sun, C., Poole, B., Krishnan, D., Schmid, C., Isola, P.: What makes for good views for contrastive learning? Advances in Neural Information Processing Systems **33**, 6827–6839 (2020)
27. Tolstikhin, I.O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., et al.: Mlp-mixer: An all-mlp architecture for vision. Advances in Neural Information Processing Systems **34**, 24261–24272 (2021)
28. Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. stat **1050**, 20 (2017)
29. Veličković, P., Fedus, W., Hamilton, W.L., Liò, P., Bengio, Y., Hjelm, R.D.: Deep graph infomax. arXiv preprint arXiv:1809.10341 (2018)
30. Wang, X., He, X., Wang, M., Feng, F., Chua, T.S.: Neural graph collaborative filtering. In: Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval. pp. 165–174 (2019)

31. Wang, X., Ji, H., Shi, C., Wang, B., Ye, Y., Cui, P., Yu, P.S.: Heterogeneous graph attention network. In: The world wide web conference. pp. 2022–2032 (2019)
32. Wang, X., Liu, N., Han, H., Shi, C.: Self-supervised heterogeneous graph neural network with co-contrastive learning. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. pp. 1726–1736 (2021)
33. Wang, X., Gui, M., Jiang, Y., Jia, Z., Bach, N., Wang, T., Huang, Z., Huang, F., Tu, K.: Ita: Image-text alignments for multi-modal named entity recognition. arXiv preprint arXiv:2112.06482 (2021)
34. Wu, J., Wang, X., Feng, F., He, X., Chen, L., Lian, J., Xie, X.: Self-supervised graph learning for recommendation. In: Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval. pp. 726–735 (2021)
35. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems **32**(1), 4–24 (2020)
36. You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., Shen, Y.: Graph contrastive learning with augmentations. Advances in Neural Information Processing Systems **33**, 5812–5823 (2020)
37. Zheng, V.W., Sha, M., Li, Y., Yang, H., Fang, Y., Zhang, Z., Tan, K.L., Chang, K.C.C.: Heterogeneous embedding propagation for large-scale e-commerce user alignment. In: 2018 IEEE International Conference on Data Mining (ICDM). pp. 1434–1439. IEEE (2018)
38. Zou, Y., Yu, Z., Liu, X., Kumar, B., Wang, J.: Confidence regularized self-training. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5982–5991 (2019)