Graph Alignment with Noisy Supervision

Shichao Pei
King Abdullah University of Science and Technology
Thuwal, Saudi Arabia
shichao.pei@kaust.edu.sa

Lu Yu
Ant Group
Hangzhou, China
bruceyu.yl@alibaba-inc.com

Guoxian Yu
Shandong University
Jinan, China
guoxian85@gmail.com

Xiangliang Zhang*
University of Notre Dame
Notre Dame, USA
xzhang33@nd.edu

ABSTRACT

Recent years have witnessed increasing attention on the application of graph alignment to on-Web tasks, such as knowledge graph integration and social network linking. Despite achieving remarkable performance, prevailing graph alignment models still suffer from noisy supervision, yet how to mitigate the impact of noise in labeled data is still under-explored. The negative sampling based noise discrimination model has been a feasible solution to detect the noisy data and filter them out. However, due to its sensitivity to the sampling distribution, the negative sampling based noise discrimination model would lead to an inaccurate decision boundary. Furthermore, it is difficult to find an abiding threshold to separate the potential positive (benign) and negative (noisy) data in the whole training process. To address these important issues, in this paper, we design a non-sampling discrimination model resorting to the unbiased risk estimation of positive-unlabeled learning to circumvent the harmful impact of negative sampling. We also propose to select the appropriate potential positive data at different training stages by an adaptive filtration threshold enabled by curriculum learning, for maximally improving the performance of alignment model and non-sampling discrimination model. Extensive experiments conducted on several real-world datasets validate the effectiveness of our proposed method.

CCS CONCEPTS

• **Information systems** → **Information integration**; • **Computing methodologies** → **Machine learning approaches**.

KEYWORDS

Graph Alignment, Robustness, Positive-Unlabeled Learning, Curriculum Learning

*Corresponding author. Dr. Xiangliang Zhang is secondly affiliated with King Abdullah University of Science and Technology, Saudi Arabia.

WWW '22, April 25–29, 2022, Virtual Event, Lyon, France
© 2022 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9096-5/22/04.
https://doi.org/10.1145/3485447.3512089

ACM Reference Format:
Shichao Pei, Lu Yu, Guoxian Yu, and Xiangliang Zhang. 2022. Graph Alignment with Noisy Supervision. In *Proceedings of the ACM Web Conference 2022 (WWW '22), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3485447.3512089

1 INTRODUCTION

Graph alignment is one of the most crucial research problems in the graph domain, which attempts to associate the same nodes across graphs [13, 69]. It has been widely employed to alleviate the challenging data sparsity issue in the fields of semantic web and social network by knowledge graph integration [11], and social network linking [16]. A number of supervised [8, 13, 57] and semi-supervised [30, 41] approaches proposed for the problem depend severely on a given set of *clean* labeled node pairs (training data) to learn the association between graphs. However, the involvement of noise is a common and unavoidable problem in the real-world data annotation process. The existence of *noisy pairs* in the training data has harmful effects on alignment results [42], like the issues caused to the general label-dependent learning [36, 50].

Despite a few of unsupervised methods [24, 31, 60, 71] have endeavoured to extricate from the demand of labeled data, they extremely rely on the highly discriminative attributes that are usually privacy sensitive, noise polluted, and hard to collect. Thus exploring how to ease the effect of noisy supervision deserves more attention. A recent model [42] for the first time attempts to train a discrimination model to identify the potential noise in the given labeled node pairs, following the principle of minimax game [56]. Nevertheless, there are key issues that remain unsolved.

The negative sampling issue. In order to train a *noise discrimination model*, negative samples are drawn to mimic the noise data, following the negative sampling strategy [45, 56]. In spite of effectiveness, it is reasonable to argue that sampling is not robust due to its sensitivity to the sampling distribution and the number of negative samples to draw [9, 23]. On the one hand, the sampling distribution that well fulfills the purpose of combating against noise always gives higher probabilities to the negative node pairs that are more difficult to be distinguished from the trusted labeled (positive) node pairs. As the toy example in Figure 1(a) shown, pair P3 and P6 in gray circles can be sampled as negative samples for P1 and P2, respectively. These negative pairs are likely close to the decision region and can alter the decision boundary such that it encroaches further into the territory dominated by positive cases, as shown in Figure 1(b). Several true positive (benign) pairs would

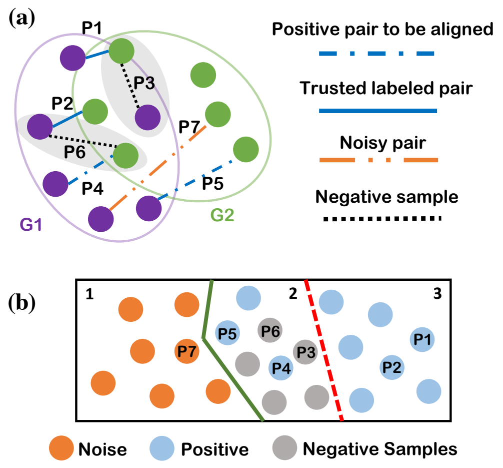

Figure 1: A toy example demonstrating the impact of negative sampling on the discriminator in robust graph alignment across two graphs. (a) Nodes in different colors indicate the embedding of nodes in different graphs. The pair P1 and P2 are trusted positive labeled node pairs and have been aligned. The pair P3 and P6 in gray circles are sampled negative samples for P1 and P2, respectively. The pair P4, P5, and P7 are in the set of untrustful labeled pairs. The pair P4 and P5 are true positive (benign) pairs that need to be aligned. The pair P7 is a noisy node pair that should be identified and not be aligned. (b) The discrimination of positive and noise samples in the embedding space of node pairs. The ground-truth decision boundary (green line) should separate noisy pair P7 from other positive pairs, but the boundary is alerted by the negative samples P3 and P6. The resulted decision boundary (red dotted line) encroaches into the space dominated by positive samples (including P1 and P2). Then, the true positive pair P4 and P5 would be misclassified as noise.

be misclassified as noise. On the other hand, the sampled small set of negative samples may not well reflect the true noise distribution. The estimated distribution based on them would be inaccurate and further cause difficulties on converging to the optimal discriminator regardless of how many update steps have been taken [61].

The positive data selection issue. In order to make the maximum usage of labeled data, positive (benign) node pairs are iteratively identified and utilized to enhance the alignment model to relate more nodes. For the discrimination model, this iterative process is a circular learning process, as the newly identified positive data can promote the generalization of the discrimination model to select more positive data. It is thus essential to ensure the cleanness of selected positive node pairs from the given labeled data. In learning with noisy label [42, 46], it is a simple and widely used strategy to apply a pre-defined threshold on the loss value of the discrimination model to select the usable positive samples. However, the setting of the fixed threshold is a dilemma. For example, a too small threshold would inevitably introduce noise to the alignment model at the early stage of training, not only hindering the capacity of alignment, but also degrading the accuracy of distinguishing true noise from benign data. On the contrary, a too large threshold disables the model to obtain sufficient usable positive data to improve the alignment performance.

Present work. To address the above challenging issues, in this paper, we propose a novel non-sampling and curriculum learning based method, named CPUGA (Curriculum Unbiased Positive-Unlabeled Graph Alignment), to mitigate the impact of noise data and improve the robustness of graph alignment model. Since the negative sampling can cause multiple above-discussed issues, we design a non-sampling discrimination model resorting to the unbiased risk estimation [14, 28] of positive-unlabeled (PU) learning and rewrite the ordinary classification risk to an equivalent form, which depends only on trusted positive labeled node pairs and unlabeled node pairs. The decision boundary learned by this discrimination model inherently gets rid of the harmful impact of negative sampling. Moreover, in order to learn more accurate representation for nodes to improve the performance of alignment, alignment model should be fed with a small set of selected positive pairs with the strongest confidence at initial, then gradually be fed with lower-confidence pairs, like human beings learning from easy samples to complex ones. Thus, we propose a curriculum learning based positive data selection strategy with adaptive threshold adjustment, to progressively select the potentially usable positive data with the training going on. The curriculum learning based selection method naturally considers the status of model in training and dynamically assigns confidence scores for positive pair selection at each iteration. In summary, our contributions of this work are as follows:

- We propose a novel robust graph alignment model designed with non-sampling learning to distinguish noise from benign data in the given labeled data. The proposed model is advanced in avoiding the issues caused by negative sampling.
- We design a curriculum learning based positive data selection strategy with adaptive threshold adjustment, such that the appropriate data at the different stages of training are used to boost the performance of both alignment model and non-sampling discrimination model.
- We conduct extensive experiments on real-world datasets, and validate the effectiveness of our proposed method with comparison to strong baselines.

2 RELATED WORK

2.1 Graph Alignment

Graph alignment is a task to associate the same nodes across graphs. It provides an effective way to facilitate many real-world applications [11, 21, 48]. In general, the existing methods can be mainly cataloged into two categories [70], i.e., (1) topological structure based methods [11, 13, 35, 37, 40, 41, 43, 52, 62, 68], which only depend on the structural information of graphs to associate nodes; (2) structure and attribute based methods, which leverage the structure and highly discriminative attribute features for aligning graphs, including the attribute of nodes [10, 17, 24, 30, 49, 51, 57, 64, 67] and attribute of edges [58, 59, 66]. Most of them still need a set of clean labeled node pairs except for several unsupervised methods [24, 31, 60, 71], which yet require the privacy-sensitive and noise-involved attribute on nodes or have to make an assumption of structure identity. However, the assumption does not hold when graphs come from different domains, because aligned nodes usually have different structures. REA [42] is the first work that considers the noisy node pairs in the training data. It follows the principle of adversarial training [18] to sample negative node pairs from unlabeled data as noise, and to train a classifier to distinguish the sampled noise data and the trusted positive data. Although REA is effective for improving the robustness of alignment model, it suffers from the harmful impact of negative sampling and the difficulty on selecting appropriate positive (benign) samples.

2.2 Learning with Noisy Supervision

Preventing deep networks from overfitting to noisy labels has gained increasing attention. In general, related methods can be grouped into three categories, i.e., 1) noise transition matrix based methods [54], which try to estimate the probability of a class with clean label flipping into other classes; 2) objective-based methods [6, 65], which adopt regularization and reweighting techniques; 3) optimization policy based methods [22, 26, 39], which rely on memorization effects of deep neural network [2] to improve the robustness by self-training or co-training. However, they all focus on the general deep networks, and cannot be directly applied on the graph alignment problem, for two main reasons. First, the loss (distance between nodes in a node pair) for positive or noisy pairs is directly minimized according to the popular loss functions [11, 57] of graph alignment. There is no distinction between the loss for positive and noisy pairs, and the memorization effects based "large-small loss" trick cannot be applied. Second, no class information in graph alignment can be used for noise transition matrix estimation. Therefore, a distinctive approach for graph alignment with noisy supervision is expected.

2.3 Curriculum Learning

How to select training samples to learn a good model is an essential research problem in the field of machine learning. With the development of deep learning and its important role as a powerful learning paradigm in many applications, curriculum learning [5] that controls the order by which training examples are fed to neural networks during training is gaining increased attention [19]. Curriculum learning relies on the prior knowledge to construct a ranking function to assign learning priorities to different data samples. Previous studies have shown that curriculum learning can improve the generalization ability and the convergence of various models [20, 44]. It provides a feasible paradigm for the desire to achieve the adaptive threshold adjustment in our scenario. Thus, to progressively pick out plausible positive pairs, we design an adaptive positive data selection strategy based on curriculum learning, to improve the performance of both alignment model and non-sampling discrimination model.

2.4 Positive-Unlabeled Learning

There are many scenarios in real-world where only positive (P) data can be collected, together with a large amount of unlabeled (U) data. The task of learning with such data is called Positive-Unlabeled (PU) learning [4]. Early PU works [32, 34] are based on the negative (N) sample selection, which selects reliable N data from U data by various selection strategies then conducts ordinary PN learning.

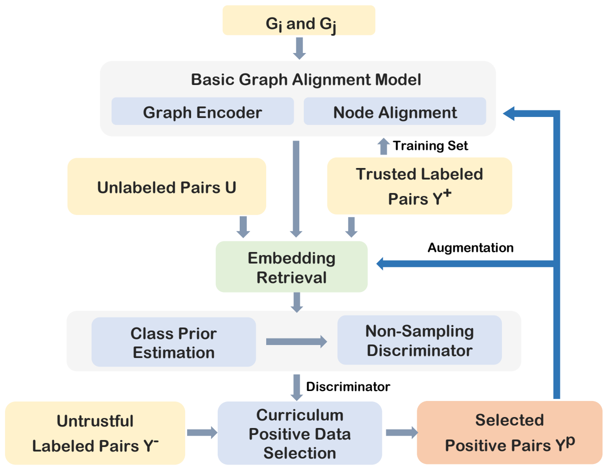

Figure 2: An overview of CPUGA model in training stage. The yellow boxes denote the data available for training. The green box works for retrieving node embedding from graph encoder. The blue modules are the key components of CPUGA, and trained jointly for delivering a graph alignment model to the testing stage.

After that, importance reweighing strategy based methods [15, 33] that regard U data as N data but with reduced importance weights show its superiority on performance. Recently, unbiased risk estimation based approaches [14, 28] that rewrite the classification risk to an equivalent form only depending on PU data achieve better performance. In our work, we apply the unbiased risk estimation to design a non-sampling method to avoid the issues caused by negative sampling in building noise discrimination model.

3 THE PROPOSED APPROACH

We first formulate the problem of graph alignment with noisy supervision. Then we introduce the proposed model in details, whose architecture is illustrated in Figure 2.

3.1 Problem Formulation

Let $\mathcal{G}_i = \{\mathcal{E}_i, \mathcal{R}_i\}$ and $\mathcal{G}_j = \{\mathcal{E}_j, \mathcal{R}_j\}$ be two distinct graphs that need to be aligned. In both graphs, $\mathcal{E}$ refers to a set of nodes, $\mathcal{R}$ denotes a set of edges. We use $n = |\mathcal{E}_i|$ and $m = |\mathcal{E}_j|$ to denote the number of nodes in $\mathcal{G}_i$ and $\mathcal{G}_j$, respectively. The node association matrix between graph $\mathcal{G}_i$ and $\mathcal{G}_j$ can be noted by $\mathbf{A} \in \mathbb{R}^{n \times m}$. Following the previous work in learning with noisy label, without loss of generality, we assume there exist a set of *trusted* positive labeled node pairs $\mathcal{Y}^+$ with $\mathbf{A}_{ij} = 1$, i.e., $\mathcal{Y}^+ = \{(e_i, e_j)|\mathbf{A}_{ij} = 1\}$ including no noise. There also exist a set of *untrustful* labeled node pairs $\mathcal{Y}^-$ that are labeled as positive, however, likely being a true positive (correctly labeled) or a false positive (wrongly labeled). Here the false positive refers to noise. These untrustful labeled node pairs are denoted by $\mathcal{Y}^- = \{(e_i, e_j)|\mathbf{A}_{ij} = 2\}$. For all other unlabeled pairs, they form the set $\mathcal{U} = \{(e_i, e_j)|\mathbf{A}_{ij} = 0\}$, indicating that the node pair has not been labeled but can be either aligned (positive) or not aligned (negative). As we focus on 1-to-1 node alignment rather than 1-to-many alignment in this paper, we set

Table 1: Notations

Symbol	Description
$\mathcal{G}$	A graph
$\mathbf{A}$	Node association matrix
$\mathcal{Y}^+$	Trusted positive labeled node pairs
$\mathcal{Y}^-$	Untrustful labeled node pairs
$\mathcal{U}$	Unlabeled node pairs
$\mathcal{Y}^p$	Selected positive (benign) node pairs
$\mathcal{Y}^T = \mathcal{Y}^+ + \mathcal{Y}^p$	Augmented positive node pairs

$||\mathbf{A}_{i,:}||_0 = 1$ and $||\mathbf{A}_{:,j}||_0 = 1$ as constraints for **any labeled node pair** (e_i, e_j). The constraints mean that we implicitly set $||\mathbf{A}_{i,:}||_0 = 0$ and $||\mathbf{A}_{:,j}||_0 = 0$ as constraints for any unaligned node e_i in $\mathcal{G}_i$ and e_j in $\mathcal{G}_j$.

Formally, given graph $\mathcal{G}_i$ and $\mathcal{G}_j$, and a node association matrix $\mathbf{A}$ with labeled pairs, graph alignment is to identify a set of unlabeled node pairs (e_i, e_j) with constraints $||\mathbf{A}_{i,:}||_0 = 0$ and $||\mathbf{A}_{:,j}||_0 = 0$ and change the associate indication from $\mathbf{A}_{ij} = 0$ to $\mathbf{A}_{ij} = 1$. Specifically, e_i and e_j in the identified node pairs (e_i, e_j) are the same nodes across graphs. With the objective of improving the robustness of graph alignment, the alignment model is designed to learn from a set of augmented positive node pairs $\mathcal{Y}^T = \mathcal{Y}^p \cup \mathcal{Y}^+$, where $\mathcal{Y}^p$ is the selected benign node pairs from $\mathcal{Y}^-$ with high confidence scores. The selection of $\mathcal{Y}^p$ is operated by a noise discrimination model that can distinguish $\mathcal{Y}^p$ from noisy pairs in $\mathcal{Y}^-$ by using trusted labeled node pairs $\mathcal{Y}^+$ and unlabeled node pairs $\mathcal{U}$. The discrimination model is expected to recognize and select as many true positive node pairs as possible in $\mathcal{Y}^-$.

3.2 Basic Graph Alignment Model

First, we introduce a graph encoder for learning the representation of nodes using the structural information of given graphs, and define the loss function for graph alignment. Inspired by the progress of graph neural networks (GNNs) [27] and its application on graph alignment [8, 57], we follow the previous work to adopt GNNs as the graph encoder. We define the encoder as:

$$\mathbf{H}^i, \mathbf{H}^j = \Phi(\mathcal{G}_i, \mathcal{G}_j) \tag{1}$$

where $\Phi(\cdot)$ is the graph encoder. Its input is graph $\mathcal{G}_i$ and $\mathcal{G}_j$, which are encoded jointly using the same encoder $\Phi(\cdot)$ to embed both graphs into the same embedding space, and its output is the learned embedding $\mathbf{H}^i = \left\{\mathbf{h}^i_1, \mathbf{h}^i_2, ..., \mathbf{h}^i_n\right\}$ of nodes in graph $\mathcal{G}_i$, and $\mathbf{H}^j = \left\{\mathbf{h}^j_1, \mathbf{h}^j_2, ..., \mathbf{h}^j_m\right\}$ of nodes in graph $\mathcal{G}_j$. We use $\mathbf{h}_z$ to denote the embedding of node e_z with ignorance of i and j for succinct presentation. GNNs used in the encoder work by:

$$\mathbf{h}^l_{\mathcal{N}_{e_i}} \leftarrow \text{Aggregate}(\left\{\mathbf{h}^l_k, \forall e_k \in \{e_i\} \cup \mathcal{N}_{e_i}\right\}) \tag{2}$$

$$\mathbf{h}^{l+1}_i \leftarrow \sigma(\mathbf{W}^l \cdot \mathbf{h}^l_{\mathcal{N}_{e_i}}) \tag{3}$$

where $\mathcal{N}_{e_i}$ denotes the set of neighboring nodes around e_i, $\mathbf{h}^l_k$ is the embedding of e_k generated from the l-th aggregation, $\mathbf{h}^l_{\mathcal{N}_{e_i}}$ is the intermediate embedding for node e_i and its neighbors, and $\mathbf{W}^l$ is a trainable parameter of layer l. An aggregation operator, such as normalized mean pooling [27] and attentional weighted summation [55], is applied in Eq. (2) to aggregate neighboring nodes. The activation function $\sigma(\cdot)$ in Eq. (3) can be set as LeakyReLU. After obtaining the embedding of nodes, we can define the loss function following the popular setting [57] for graph alignment as:

$$\mathcal{L}_{\text{GA}} = \sum_{(e_i,e_j)\in\mathcal{Y}^T} \sum_{(e'_i,e'_j)\in N(e_i,e_j)} [f_e(\mathbf{h}_i, \mathbf{h}_j) - f_e(\mathbf{h}'_i, \mathbf{h}'_j) + \gamma]_+ \tag{4}$$

where $\mathcal{Y}^T$ is the augmented positive node pairs, which can be initialized using $\mathcal{Y}^+$, then be complemented by the selected node pairs $\mathcal{Y}^p$ from $\mathcal{Y}^-$. We will discuss how to obtain $\mathcal{Y}^p$ later. γ is a predefined margin parameter which is greater than 0, and $[x]_+ = \max\{0, x\}$. The function $f_e(\cdot)$ can be defined as: $f_e(\mathbf{h}_i, \mathbf{h}_j) = ||\mathbf{h}_i - \mathbf{h}_j||_1$. To optimize the margin-based ranking loss in Eq. (4), we require a set of negative node pairs $N(e_i, e_j)$ for each pair (e_i, e_j): $N(e_i, e_j) = \{(e'_i, e_j)|e'_i \in \mathcal{E}_i\} \cup \{(e_i, e'_j)|e'_j \in \mathcal{E}_j\}$, where we replace e_i and e_j by the Bernoulli negative [10] sampled nodes e'_i and e'_j. Here the negative sampling in $\mathcal{L}_{\text{GA}}$ following the popular loss function [8, 57] does not lead to the above-discussed negative sampling issues because the aim of pairwise ranking based loss $\mathcal{L}_{\text{GA}}$ is to learn an optimal ordering of node pairs rather than to learn an accurate decision boundary for distinguishing noise like the noise discrimination model. We focus on circumventing the impact of negative sampling on *noise discrimination model* in this work.

3.3 Non-sampling Discrimination Model

To avoid applying negative sampling in the noise discrimination model, we resort to the unlabeled node pairs $\mathcal{U}$ to design a non-sampling discrimination model because $\mathcal{U}$ includes positive data and abundant negative data. Training a discrimination model using unlabeled data $\mathcal{U}$ under the guidance of trusted positive labeled data $\mathcal{Y}^+$ is in fact a Positive-Unlabeled (PU) classification problem [14] where only a part of the positive data are labeled. Suppose that the unlabeled data $\mathcal{U}$ is composed of positive pairs $\mathcal{U}^+$ and negative (noisy) pairs $\mathcal{U}^-$. The marginal density of $\mathcal{U}$ is $p_{\mathcal{U}}(x)$ where x is a random variable referring to node pairs. $p_{\mathcal{U}^+}(x) = p_{\mathcal{U}}(x|1)$ and $p_{\mathcal{U}^-}(x) = p_{\mathcal{U}}(x|-1)$ are the class conditional densities of $\mathcal{U}^+$ and $\mathcal{U}^-$, respectively. The marginal density $p_{\mathcal{U}}(x)$ can be written as:

$$p_{\mathcal{U}}(x) = \pi p_{\mathcal{U}^+}(x) + (1-\pi) p_{\mathcal{U}^-}(x) \tag{5}$$

where π is the positive class prior. In positive-unlabeled learning, π is unknown and has to be estimated from a validation set of a fully labeled data set [15] or from the PU data [12]. To classify $\mathcal{U}^+$ and $\mathcal{U}^-$ in $\mathcal{U}$ with a binary classifier D that learns the distribution of $\mathcal{U}$ from $p_{\mathcal{U}}$, we need to minimize its expected miss-classification rate $\mathcal{L}_{\text{R}}(D)$, defined as:

$$\begin{aligned}\mathcal{L}_{\text{R}}(D) = \pi&\mathbb{E}_{x\sim p_{\mathcal{U}^+}(x)}[l(D(x), 1)] \\ &+ (1-\pi)\mathbb{E}_{x\sim p_{\mathcal{U}^-}(x)}[l(D(x), -1)]\end{aligned} \tag{6}$$

where $l(D(x), z)$ is a loss function measuring the loss of prediction with ground true label z, e.g., $l(D(x), z) = 1/(1+\exp(z\cdot D(x)))$. Since the separation of $\mathcal{U}^+$ and $\mathcal{U}^-$ is unknown, and $p_{\mathcal{U}^+}$ conforms to the distribution of $p_{\mathcal{Y}^+}$ under the assumption of Selected Completely at Random [3, 15], we utilize $p_{\mathcal{Y}^+}$ to replace $p_{\mathcal{U}^+}$ in Eq. (5). So $p_{\mathcal{U}}(x)$ can be defined as:

$$p_{\mathcal{U}}(x) = \pi p_{\mathcal{Y}^+}(x) + (1-\pi) p_{\mathcal{U}^-}(x) \tag{7}$$

Since $\mathcal{U}^-$ in unknown as well, we simply rewrite Eq. (7) as $(1-\pi)p_{\mathcal{U}^-}(x) = p_{\mathcal{U}}(x) - \pi p_{\mathcal{Y}^+}(x)$, and take the expectation of $l(D(x),-1)$ on both sides to have:

$$(1-\pi)\mathbb{E}_{x\sim p_{\mathcal{U}^-}(x)}[l(D(x),-1)] = \mathbb{E}_{x\sim p_{\mathcal{U}}(x)}[l(D(x),-1)] - \pi\mathbb{E}_{x\sim p_{\mathcal{Y}^+}(x)}[l(D(x),-1)] \tag{8}$$

By combining Eq. (6) and Eq. (8), we can obtain the updated definition of loss function $\mathcal{L}_{\mathrm{R}}(D)$ as:

$$\mathcal{L}_{\mathrm{R}}(D) = \pi\mathbb{E}_{x\sim p_{\mathcal{Y}^+}(x)}[l(D(x),1)] + \mathbb{E}_{x\sim p_{\mathcal{U}}(x)}[l(D(x),-1)] - \pi\mathbb{E}_{x\sim p_{\mathcal{Y}^+}(x)}[l(D(x),-1)] \tag{9}$$

By optimizing Eq. (9), the discrimination model D is exempted from requiring the noise (negative) data for the traditional positive-negative (PN) classification. It learns to distinguish the positive and negative by adopting the unbiased risk estimation on given positive and unlabeled data.

Note that the model may suffer from overfitting because of the flexibility of model design and the negative loss [28] caused by $\mathbb{E}_{x\sim p_{\mathcal{U}}(x)}[l(D(x),-1)] - \pi\mathbb{E}_{x\sim p_{\mathcal{Y}^+}(x)}[l(D(x),-1)]$. We take the non-negative risk estimator following the recent work [28] to handle the overfitting issue:

$$\mathcal{L}_{\mathrm{R}}(D) = \pi\mathbb{E}_{x\sim p_{\mathcal{Y}^+}(x)}[l(D(x),1)] + \max\left\{0, \mathbb{E}_{x\sim p_{\mathcal{U}}(x)}[l(D(x),-1)] - \pi\mathbb{E}_{x\sim p_{\mathcal{Y}^+}(x)}[l(D(x),-1)]\right\} \tag{10}$$

The discriminator D is implemented by a two-layer neural network with ReLU and the representation of a node pair as the input (e.g., $||\mathbf{h}_i - \mathbf{h}_j||_1$ for node pair (e_i, e_j)). The remaining key factor for addressing Eq. (10) is the estimation of positive class prior π.

3.4 Class Prior Estimation

To optimize the loss function $\mathcal{L}_{\mathrm{R}}$, we have to assign a value to π in Eq. (10), but the unknown $\mathcal{U}^-$ hinders the direct estimation of class prior. With the desire to jointly estimate the class prior with the optimization of discrimination model by backpropagation, we aim to find a variational approximation to the posterior distribution on Ψ that is a set of parameters describing the distribution of representation of node pairs from both $\mathcal{Y}^+$ and $\mathcal{U}$. This is inspired by the recent work about stochastic variational inference [25], Bayes by Backprop [7], and its application on PU learning [1]. Specifically, we suppose that the representation of node pairs conforms to a Gaussian mixture distribution which includes two components, one of the positive node pairs from $\mathcal{Y}^+$ and $\mathcal{U}^+$, and the other of the negative node pairs from $\mathcal{U}^-$. The density function of the representation of node pairs can be defined as $p(\mathbf{X}|\Psi) = \phi_1\mathcal{N}(\mathbf{X}|\mu_1,\Sigma_1) + \phi_2\mathcal{N}(\mathbf{X}|\mu_2,\Sigma_2)$, where $\mathbf{X}$ denotes the representation of node pairs, μ_1 and μ_2 are mean, Σ_1 and Σ_2 are covariance matrices, ϕ_1 and ϕ_2 are non-negative mixture weights, and $\phi_1 + \phi_2 = 1$. The distribution $p(\mathbf{X})$ is determined by these parameters, jointly denoted as Ψ for simplicity in later description.

With the desire to estimate parameters Ψ to assign a value for π, we employ variational inference to approximate the posterior distribution $p(\Psi|\mathbf{X})$ due to the intractability of this posterior distribution. Variational learning finds parameters θ of a variational distribution on Ψ that minimizes the Kullback-Leibler (KL) divergence with posterior distribution. We denote the variational distribution as $q(\Psi|\theta)$ and define the KL divergence as:

$$\begin{aligned}\mathrm{KL}[q(\Psi|\theta)||p(\Psi|\mathbf{X})] &= \int q(\Psi|\theta)\log\frac{q(\Psi|\theta)p(\mathbf{X})}{p(\Psi)p(\mathbf{X}|\Psi)}d\Psi \\ &= \mathrm{KL}[q(\Psi|\theta)||p(\Psi)] - \mathbb{E}_{q(\Psi|\theta)}[\log p(\mathbf{X}|\Psi)] + \log p(\mathbf{X})\end{aligned} \tag{11}$$

where $p(\Psi)$ refers to the prior distribution. We adopt Dirichlet distribution as the prior for ϕ, and use Gaussian distribution as the prior for μ, then Gamma distribution as the prior for Σ. $\mathbb{E}_{q(\Psi|\theta)}[\log p(\mathbf{X}|\Psi)]$ is the log-likelihood of $\mathbf{X}$ fitting to the Gaussian mixture distribution with parameters Ψ. As $\log p(\mathbf{X})$ is a constant, we do not consider this term in the optimization process. The loss function can be defined as:

$$\mathcal{L}_{\text{prior}} = \mathrm{KL}[q(\Psi|\theta)||p(\Psi)] - \mathbb{E}_{q(\Psi|\theta)}[\log p(\mathbf{X}|\Psi)] \tag{12}$$

The loss function $\mathcal{L}_{\text{prior}}$ is called as evidence lower bound (ELBO). We can approximate this exact loss through a Monte Carlo sampling procedure as follows:

$$\mathcal{L}_{\text{prior}} \approx \sum_{i=1}^{n_s}\log q(\Psi^{(i)}|\theta) - \log p(\Psi^{(i)}) - \log p(\mathbf{X}|\Psi^{(i)}) \tag{13}$$

where $\Psi^{(i)}$ denotes the i-th Monte Carlo sample drawn from the variational posterior distribution, n_s is the number of samples. Then the loss function $\mathcal{L}_{\text{prior}}$ can be optimized by stochastic gradient descent to find the θ^*. Then we obtain the optimal Ψ by sampling from $q(\Psi|\theta^*)$. Last, we can estimate the class prior π as follows:

$$\pi = q(\phi_i|\theta^*), i = \arg\max_{k=1,2}|\#_k| \tag{14}$$

$$\#_1 = \left\{x|x\in\mathcal{Y}^+, p(x|\phi_1,\mu_1,\Sigma_1) > p(x|\phi_2,\mu_2,\Sigma_2)\right\} \tag{15}$$

$$\#_2 = \left\{x|x\in\mathcal{Y}^+, p(x|\phi_2,\mu_2,\Sigma_2) > p(x|\phi_1,\mu_1,\Sigma_1)\right\} \tag{16}$$

where $\#_1$ and $\#_2$ denote two sets of node pairs from trusted positive labeled pairs $\mathcal{Y}^+$, and $|\#_k|$ counts the number of trusted labeled pairs in $\#_k$. We denote that the component i of Gaussian mixture distribution contains more positive samples. Since the component i with more positive samples can be regarded as $p_{\mathcal{U}^+}(x)$, it is natural to select ϕ_i as the class prior π.

3.5 Curriculum Positive Data Selection

The need of more positive data for improving the alignment performance motivates us to select the potential positive node pairs $\mathcal{Y}^p$ from untrustful labeled node pairs $\mathcal{Y}^-$ to augment the trusted positive labeled node pairs $\mathcal{Y}^+$. The augmented positive data can be used for training not only the graph alignment but also the non-sampling discrimination model. It is always easy to pick out the plausible positive samples in the beginning and then it gets harder and harder to pick out more plausible positive samples. Therefore, we design a curriculum positive data selection strategy to progressively augment $\mathcal{Y}^+$ with the training going on, since curriculum learning can improve the generalization and the convergence of learning models by starting with easy samples and then gradually increasing the difficulty.

3.5.1 *Curriculum Learning based Data Selection Strategy.*

The main idea of curriculum learning is to feed easy samples to a model at the early stage of training and gradually provide difficult samples with the training process going on. In our scenario, we aim to feed a small set of selected positive node pairs $\mathcal{Y}^p$ with the strongest confidence to the alignment model and the non-sampling discrimination model at initial, then gradually enlarge $\mathcal{Y}^p$ with lower-confidence node pairs from $\mathcal{Y}^-$. Specifically, we introduce a parameter $\mathbf{v}$ to weigh each node pair (e_i, e_j) in $\mathcal{Y}^-$ to indicate if a node pair should be selected as a potential positive data. Therefore, our goal is to learn the latent weight variable $\mathbf{v} = [v_1, v_2, ..., v_{|\mathcal{Y}^-|}]^T$ in each iteration of training. To learn the optimal $\mathbf{v}$ in iteration λ, we first define a loss function for $\mathcal{Y}^-$ with weight parameters $\mathbf{v}$ as:

$$\min_{\mathbf{v}\in[0,1]^{|\mathcal{Y}^-|}} \mathcal{L}_{sp}(\mathbf{v};\lambda) = -\sum_{i=1}^{|\mathcal{Y}^-|} v_i l_{sp}(D(x_i), -1) + f_r(\lambda)\sum_{i=1}^{|\mathcal{Y}^-|} v_i \tag{17}$$

where x_i denotes a node pair in $\mathcal{Y}^-$, v_i is the weight parameter for x_i, D is the discriminator defined in Eq. (10), and $l_{sp}(\cdot)$ is a *scoring function* to measure the confidence of node pairs (we will discuss and define it in later description). We tentatively set the label of all pairs in $\mathcal{Y}^-$ as negative (i.e., -1). Then the node pairs with large loss would be the potential positive samples, while the node pairs with small loss can be regarded as noisy samples. Besides, $||\mathbf{v}||_1 = \sum_{i=1}^{|\mathcal{Y}^-|} v_i$ is an l_1-norm regularizer. Since D is optimized in Eq. (10), D is fixed when minimizing $\mathcal{L}_{sp}$. The global optimal $\mathbf{v}^* = [v_1^*, v_2^*, ..., v_{|\mathcal{Y}^-|}^*]$ in iteration λ can be reached with analytical solution for the loss $\mathcal{L}_{sp}$:

$$v_i^* = \begin{cases} 1 & l_{sp}(D(x_i), -1) > f_r(\lambda) \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

where $f_r(\lambda)$ is a *pacing function* parameterized by iteration number λ for controlling the learning pace. Node pair x_i would be selected in iteration λ if its score $l_{sp}(D(x_i), -1)$ is larger than $f_r(\lambda)$. We define $f_r(\lambda) = a + b \cdot \exp(-\frac{\lambda}{c})$ where a, b and c are pre-defined parameters to control the learning pace. $f_r(\lambda)$ would decrease with λ increasing. It means the number of node pairs selected in each iteration increases with the training going on, first few high-confidence node pairs, then more lower confidence node pairs.

Next, we define the scoring function l_{sp} used in Eq. (17). Though Sigmoid loss has been widely used in classification tasks, it has a drawback in our scenario, i.e., its loss value is always constrained in the range of (0, 1), making the loss of potential positive node pairs upper-bounded. We leverage logistic loss as the alternative that is not upper bounded, but is Lipschitz continuous and differentiable everywhere. We define the scoring function as $l_{sp}(D(x), z) = \ln(1 + \exp(-z \cdot D(x)))$, where z denotes the label of input data x.

With the scoring function $l_{sp}(\cdot)$ and pacing function $f_r(\cdot)$ in curriculum learning, we can dynamically select potential positive data with the training going on, and the confidence score of each node pair in $\mathcal{Y}^-$ can be updated with the optimization of the score function $l_{sp}(\cdot)$.

3.5.2 *Utilization of Selected Positive Data.*

The selected node pairs $\mathcal{Y}^p$ can be used to append $\mathcal{Y}^+$, as shown in Eq. (4), to enlarge the training set. Furthermore, $\mathcal{Y}^p$ can be used for improving the performance of non-sampling discrimination model because they enlarge the positive data. However, node pairs $\mathcal{Y}^p$ have a large tendency to be biased [63], as the selected pairs usually have large loss and may not cover all the positive node pair space. Then, the term $\pi\mathbb{E}_{x\sim p_{\mathcal{Y}^+\cup\mathcal{Y}^p}(x)}[l(D(x), -1)]$ in Eq. (10) could be very large if we directly combine $\mathcal{Y}^+$ and $\mathcal{Y}^p$. The empirical miss-classification rate $\mathcal{L}_\mathrm{R}(D)$ would be underestimated if subtracting the term in Eq. (10). To avoid the impact of bias in the selected data $\mathcal{Y}^p$, we modify the empirical miss-classification rate as follow:

$$\begin{aligned} \mathcal{L}'_\mathrm{R}(D) &= \pi\mathbb{E}_{x\sim p_{\mathcal{Y}^T}(x)}[l(D(x), 1)] \\ &+ \max\left\{0, \mathbb{E}_{x\sim p_{\mathcal{U}}(x)}[l(D(x), -1)] - \pi\mathbb{E}_{x\sim p_{\mathcal{Y}^+}(x)}[l(D(x), -1)]\right\} \end{aligned} \tag{19}$$

It differs from Eq. (10) in the first term, by replacing $x \sim p_{\mathcal{Y}^+}(x)$ with $x \sim p_{\mathcal{Y}^T}(x)$. We keep the latter terms the same as Eq. (10) so that the impact of bias can be avoided.

3.6 Optimization

To jointly optimize the loss of graph alignment $\mathcal{L}_\mathrm{GA}$, the loss of non-sampling discrimination model $\mathcal{L}'_\mathrm{R}(D)$, and the loss of class prior estimation $\mathcal{L}_\mathrm{prior}$, as well as the loss for curriculum data selection $\mathcal{L}_{sp}$, we adopt the iterative optimization strategy to train the proposed CPUGA. First, we initialize the weight parameters $\mathbf{v}$ as $\mathbf{0}$, and optimize the loss of graph alignment $\mathcal{L}_\mathrm{GA}$ to obtain the embedding of nodes. Then we optimize the loss $\mathcal{L}_\mathrm{prior}$ to estimate the class prior π, then train the non-sampling discrimination model using node pairs $\mathcal{Y}^+$ and $\mathcal{U}$ with class prior π. With optimized discriminator D, we can optimize loss $\mathcal{L}_{sp}$ and calculate the optimal $\mathbf{v}$ in current iteration, then augmented positive node pairs $\mathcal{Y}^p$ can be collected to further optimize the loss of graph alignment $\mathcal{L}_\mathrm{GA}$ and the loss of discrimination model $\mathcal{L}'_\mathrm{R}(D)$. We repeat the above process with enough iterations until the whole model converges. The detailed training procedure can be found in the Appendix B.3.

4 EXPERIMENTS

In this section, we describe the conducted evaluation experiments for verifying the effectiveness of the proposed method CPUGA on knowledge graph alignment problem [11], which is a canonical graph alignment task and has attracted increasing attention in the recent years.

Datasets. Following the previous studies [52, 57], we use two widely used public datasets DBP15K [51] and DWY100K [52]. We randomly split 30% of labeled node (i.e., entity in knowledge graphs) pairs for training and use the rest 70% of them for testing. Since the given labeled node pairs in above datasets are clean, we generate some noise data to replace a part of the clean data, in order to simulate the real raw labeled node pairs collected from annotation platforms or generated by some current alignment models. Following the recent work [42], we randomly corrupt 40% of the training set as the noisy pairs, and keep the rest 60% as the positive pairs. Then we randomly select 50% of positive pairs as the trusted positive labeled node pairs $\mathcal{Y}^+$, and mix other 50% of positive pairs with the noisy pairs as the untrustful labeled node pairs $\mathcal{Y}^-$. To construct the node association matrix $\mathbf{A}$, we first fill the node pairs in $\mathcal{Y}^+$ into the matrix $\mathbf{A}$ as 1, then fill the other 50% of positive node pairs in $\mathcal{Y}^-$ as 2. To corrupt the clean

Table 2: Graph alignment performance comparison with noisy labeled pairs in DBP15K **and** DWY100K. **The best results are in bold, and the strongest baseline is underlined.**

Approaches	$\text{DBP15K}_{\text{ZH-EN}}$			$\text{DBP15K}_{\text{JA-EN}}$			$\text{DBP15K}_{\text{FR-EN}}$			$\text{DWY100K}_{\text{WD}}$			$\text{DWY100K}_{\text{YG}}$		
	Hits@1	Hits@5	MRR	Hits@1	Hits@5	MRR	Hits@1	Hits@5	MRR	Hits@1	Hits@5	MRR	Hits@1	Hits@5	MRR
MTransE [11]	0.169	0.362	0.216	0.148	0.345	0.198	0.143	0.338	0.192	0.154	0.325	0.203	0.137	0.318	0.186
ITransE [72]	0.185	0.394	0.258	0.174	0.386	0.242	0.181	0.402	0.269	0.193	0.414	0.296	0.158	0.346	0.223
GCN-Align [57]	0.223	0.424	0.316	0.223	0.439	0.321	0.231	0.462	0.337	0.293	0.461	0.372	0.354	0.505	0.428
AlignEA [52]	0.263	0.457	0.342	0.254	0.451	0.338	0.278	0.471	0.357	0.331	0.487	0.392	0.376	0.525	0.448
MuGNN [8]	0.274	0.471	0.361	0.279	0.481	0.368	0.284	0.485	0.372	0.348	0.503	0.417	0.401	0.554	0.475
AliNet [53]	0.286	0.468	0.365	0.295	0.470	0.379	0.298	0.486	0.384	0.372	0.514	0.437	0.420	0.563	0.490
REA-KE [42]	0.235	0.437	0.319	0.236	0.451	0.334	0.229	0.456	0.332	0.312	0.468	0.379	0.352	0.513	0.432
REA [42]	0.289	0.486	0.380	0.293	0.498	0.388	0.304	0.539	0.403	0.368	0.547	0.444	0.426	0.577	0.494
CPUGA-KE	0.228	0.426	0.316	0.230	0.446	0.323	0.228	0.457	0.334	0.298	0.462	0.375	0.356	0.509	0.427
CPUGA	**0.306**	**0.506**	**0.397**	**0.312**	**0.521**	**0.406**	**0.321**	**0.556**	**0.424**	**0.390**	**0.568**	**0.467**	**0.449**	**0.603**	**0.524**

node pair (e_i, e_j), we replace (e_i, e_j) with $(e_{i'}, e_{j'})$ sampled from $\{(e_{i'}, e_j)|\mathbf{A}_{i'j} = 0, ||\mathbf{A}_{i',:}||_0 = 0\} \cup \{(e_i, e_{j'})|\mathbf{A}_{ij'} = 0, ||\mathbf{A}_{:,j'}||_0 = 0\}$. The corrupted pair has only one node difference from the clean pair. This is analogous to a careless mistake made in the real-world alignment annotation process. Then we can fill the sampled noisy node pairs into $\mathbf{A}$ as 2. The detailed information of the datasets can be found in Appendix C.

Baseline methods. To validate the effectiveness of our proposed method, we compare it with several embedding-based methods without the consideration of noise in labeled data: MTransE [11], ITransE [72], AlignEA [52], GCN-Align [57], MuGCN [8], and AliNet [53]. We also compare the proposed method with a negative sampling based robust knowledge graph alignment model: REA [42]. Note that some graph alignment models were proposed to design more powerful graph encoders to encode more comprehensive information, such as attribute information of nodes and edges. As our model solely relies on structural information and focuses on the robustness of alignment model, we do not take these models into the comparison. In addition, CPUGA-KE denotes a variant of CPUGA that only contains a graph encoder without discrimination model and positive data selection.

Performance of graph alignment with noise data. We present the alignment results of all evaluated models in Table 2. We can see that CPUGA outperforms the state-of-the-art structured based embedding methods for graph alignment by Hits@1, Hits@5 and MRR on different datasets. Although these embedding based approaches, such as MuGNN and AliNet, develop different knowledge graph encoders using advanced techniques, they still suffer from the noise in the given labeled node pairs. Since these approaches do not have any mechanism to combat against the noise data, we can notice that the influence of noisy labels on their alignment result is significant. REA leverages a negative sampling based method to detect the noise data. However, the improvement of REA on graph alignment is suppressed due to the utilization of negative sampling. Our CPUGA consistently outperforms REA, as we adopt a non-sampling discrimination model to learn a more reasonable decision boundary to distinguish positive data from noise, and a curriculum learning based positive data selection strategy to avoid the involvement of noise in the early stage of training. Hence, the harmful influence of negative sampling has been avoided in CPUGA. We also notice that CPUGA-KE has the similar performance with REA-KE, showing that the *improvement of CPUGA mainly comes from the design of non-sampling and curriculum learning strategies*, rather than the design of graph encoder.

Table 3: Results on distinguishing potential positive node pairs from different proportions of noise.

%Noise		ZH-EN		JA-EN		FR-EN	
		REA	CPUGA	REA	CPUGA	REA	CPUGA
20%	Pre.	0.935	0.935	0.929	**0.933**	0.930	**0.931**
	Rec.	0.731	**0.773**	0.714	**0.757**	0.750	**0.791**
	F1	0.820	**0.846**	0.808	**0.835**	0.831	**0.855**
40%	Pre.	0.871	**0.874**	0.874	**0.878**	0.867	**0.875**
	Rec.	0.760	**0.817**	0.738	**0.807**	0.778	**0.820**
	F1	0.812	**0.844**	0.800	**0.841**	0.820	**0.847**

Identification of benign and noisy node pairs. We also investigate how many positive (benign) and noisy pairs in the set of untrustful pairs $\mathcal{Y}^-$ can be correctly recognized with different proportions of noise. Specifically, we regard the predicted label of all selected pairs as 1, and the rest of $\mathcal{Y}^-$ as 0. Then, the predicted label can be compared with their ground truth label. As the results in Table 3 shown, compared with REA, CPUGA achieves higher precision, recall and F1 score on all datasets and at different levels of noise setting. This confirms again the effectiveness of CPUGA on recognizing more true positive pairs in $\mathcal{Y}^-$ correctly.

Ablation study. To gain deeper insight into the contributions of different components involved in our approach, we conduct ablation studies by considering the following variants: (1) CPUGA-SNS that adopts the simplest negative sampling based discriminator, which randomly samples negative pairs for each trusted positive pair; (2) CPUGA-NS that leverages the negative sampling strategy employed in [42], which always selects the most difficult negative pairs. Note that both CPUGA-SNS and CPUGA-NS do not implement the curriculum learning based positive data selection. (3) CPUGA-w/o-CL that only has the non-sampling discriminator and does not contain the curriculum learning based positive data

Table 4: Results of ablation study.

Method	ZH-EN		JA-EN		FR-EN	
	Hits@1	F1	Hits@1	F1	Hits@1	F1
CPUGA-SNS	0.266	0.735	0.268	0.738	0.282	0.751
CPUGA-NS	0.286	0.807	0.291	0.796	0.298	0.816
CPUGA-w/o-CL	0.298	0.838	0.305	0.835	0.315	0.836
CPUGA	**0.306**	**0.844**	**0.312**	**0.841**	**0.321**	**0.847**

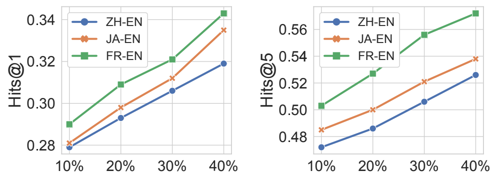

Figure 3: Results on DBP15K **when varying the proportion of trusted labeled pairs $\mathcal{Y}^+$ (x-axis).**

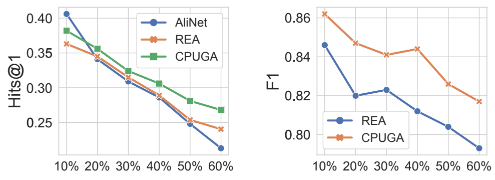

Figure 4: Results on DBP15K$_{\mathrm{ZH-EN}}$ **when the intensity of noise varies on x-axis.**

selection. The results are summarized in Table 4. Hits@1 indicates the alignment performance, and F1 denotes the performance of potential positive (benign) pair identification. We see that the performance of CPUGA-w/o-CL is inferior to that of CPUGA, because the alignment model without an appropriate threshold learns to fit to the noise at the whole stage of training. And CPUGA-w/o-CL outperforms CPUGA-NS, due to the adopted non-sampling discriminator which extricates from the impact of negative samples. Besides, it is not surprising that CPUGA-w/o-CL achieves a better performance than CPUGA-SNS, because the random negative sampling is incapable of selecting sufficient and effective negative samples for the discriminator.

Impact of the proportion of trusted positive labeled pairs $\mathcal{Y}^+$. To understand how the trusted positive labeled node pairs influence the performance of CPUGA model, we test the model with 10%, 20%, 30%, and 40% of node pairs from DBP15K as the sets of trusted positive labeled pairs. Figure 3 shows Hits@1 and Hits@5 results on three datasets with different proportions of $\mathcal{Y}^+$. As expected, the performance of CPUGA is improved as the proportion of $\mathcal{Y}^+$ increases. The reason is that more trusted positive labeled pairs not only enhance the performance of graph alignment model, but also improve the learned non-sampling discrimination model.

Impact of the intensity of noise. To explore the impact of noise intensity on the performance of alignment models, we randomly replace 10% to 60% of labeled pairs with noisy node pairs. Figure 4 shows the alignment performance using Hits@1 (left) and the performance of potential positive (benign) pair identification using F1 score (right) with different intensity of noise. We find that our proposed CPUGA consistently outperforms REA on F1 score and Hits@1, showing our method can distinguish more positive pairs from noisy pairs, and improve the robustness of alignment model. Moreover, AliNet achieves better performance than CPUGA and REA when the intensity of noise is 10%, because the harmful impact of noise is mild and AliNet applies a more advanced KG encoder, while CPUGA and REA use a basic GCN-based encoder.

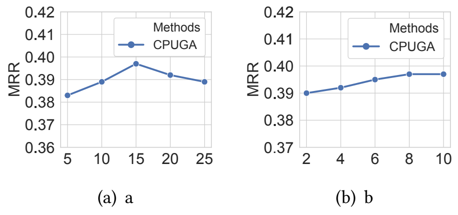

Figure 5: Parameter analysis on DBP15K$_{\mathrm{ZH-EN}}$.

Sensitivity to hyper-parameter settings. Figure 5 shows how different hyper-parameters, i.e., parameter a and b in pacing function $f_r(\cdot)$, influence the performance of CPUGA on dataset DBP15K$_{\mathrm{ZH-EN}}$. Figure 5 (a) shows that a small a degrades the performance of CPUGA because a controls the lower-bound of pacing function, a smaller a would introduce more noise, but a large a may mistakenly recognize some positive samples as noise. Figure 5 (b) shows that a small b decreases MRR score because b controls the threshold at the early stage of training. A small b recognizes more noisy pairs as positive at the beginning of training so that the alignment model is trained with more noisy node pairs. Note that the learning difficulty parameterized by a pacing function $f_r(\lambda)$ in curriculum learning does not bring the proposed method into another dilemma. Here λ denotes the iteration number. $f_r(\lambda)$ would be decreasing with λ increasing. Therefore, $f_r(\lambda)$ is an adaptive filtration threshold with the training going on. We only need to decide the lower-bound (i.e., a) and upper-bound (i.e., $a + b$) of pacing function $f_r(\lambda)$, which can be easily given by roughly estimating the score of given trusted positive node pairs using the trained non-sampling discriminator with mildly manual adjustment to find the best configuration.

5 CONCLUSION

In this paper, we proposed a robust graph alignment model to combat the noise in the given labeled data. Specifically, we designed a non-sampling discrimination model to learn more accurate decision boundary without the impact of negative sampling. We also proposed to progressively select the appropriate potential positive data at different training stages by adaptive filtration threshold enabled by curriculum learning. Extensive experimental results demonstrate the rationality and effectiveness of our proposed method. In future work, we will focus on designing an alignment model without the demand for the set of trusted labeled data because the positive samples still require human efforts to some extent. And we will attempt to devise a strategy for learning the hyper-parameters in curriculum learning to avoid manual settings.

ACKNOWLEDGMENTS

The research was supported by King Abdullah University of Science and Technology (KAUST), under award number BAS/1/1635-01-01.

REFERENCES

[1] Uchenna Akujuobi, Jun Chen, Mohamed Elhoseiny, Michael Spranger, and Xiangliang Zhang. 2020. Temporal Positive-unlabeled Learning for Biomedical Hypothesis Generation via Risk Estimation. In *NeurIPS*. 1–10.

[2] Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. 2017. A closer look at memorization in deep networks. In *ICML*. 233–242.

[3] Jessa Bekker and Jesse Davis. 2018. Estimating the class prior in positive and unlabeled data through decision tree induction. In *AAAI*. 2712–2719.

[4] Jessa Bekker and Jesse Davis. 2020. Learning from positive and unlabeled data: a survey. *Machine Learning* 109, 4 (2020), 719–760.

[5] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *ICML*. 41–48.

[6] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. 2019. Mixmatch: A holistic approach to semi-supervised learning. In *NeurIPS*. 1–10.

[7] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. 2015. Weight uncertainty in neural network. In *ICML*. 1613–1622.

[8] Yixin Cao, Zhiyuan Liu, Chengjiang Li, Juanzi Li, and Tat-Seng Chua. 2019. Multi-Channel Graph Neural Network for Entity Alignment. In *ACL*. 1452–1461.

[9] Chong Chen, Min Zhang, Chenyang Wang, Weizhi Ma, Minming Li, Yiqun Liu, and Shaoping Ma. 2019. An efficient adaptive transfer neural network for social-aware recommendation. In *SIGIR*. 225–234.

[10] Muhao Chen, Yingtao Tian, Kai-Wei Chang, Steven Skiena, and Carlo Zaniolo. 2018. Co-training Embeddings of Knowledge Graphs and Entity Descriptions for Cross-lingual Entity Alignment. In *IJCAI*. 3998–4004.

[11] Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. 2017. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In *IJCAI*. 1511–1517.

[12] Marthinus Christoffel, Gang Niu, and Masashi Sugiyama. 2016. Class-prior Estimation for Learning from Positive and Unlabeled Data. In *ACML*. 221–236.

[13] Xiaokai Chu, Xinxin Fan, Di Yao, Zhihua Zhu, Jianhui Huang, and Jingping Bi. 2019. Cross-network embedding for multi-network alignment. In *Web Conf.* 273–284.

[14] Marthinus C Du Plessis, Gang Niu, and Masashi Sugiyama. 2014. Analysis of learning from positive and unlabeled data. In *NeurIPS*. 703–711.

[15] Charles Elkan and Keith Noto. 2008. Learning classifiers from only positive and unlabeled data. In *KDD*. 213–220.

[16] Jie Feng, Mingyang Zhang, Huandong Wang, Zeyu Yang, Chao Zhang, Yong Li, and Depeng Jin. 2019. Dplink: User identity linkage via deep neural network from heterogeneous mobility data. In *Web Conf.* 459–469.

[17] Matthias Fey, Jan E Lenssen, Christopher Morris, Jonathan Masci, and Nils M Kriege. 2019. Deep Graph Matching Consensus. In *ICLR*.

[18] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *NeurIPS*. 2672–2680.

[19] Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. 2017. Automated curriculum learning for neural networks. In *ICML*. 1311–1320.

[20] Sheng Guo, Weilin Huang, Haozhi Zhang, Chenfan Zhuang, Dengke Dong, Matthew R Scott, and Dinglong Huang. 2018. Curriculumnet: Weakly supervised learning from large-scale web images. In *ECCV*. 135–150.

[21] Pietro Hiram Guzzi and Tijana Milenković. 2018. Survey of local and global biological network alignment: the need to reconcile the two sides of the same coin. *Briefings in Bioinformatics* 19, 3 (2018), 472–481.

[22] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. 2018. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NeurIPS*. 8527–8537.

[23] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Web Conf.* 173–182.

[24] Mark Heimann, Haoming Shen, Tara Safavi, and Danai Koutra. 2018. Regal: Representation learning-based graph alignment. In *CIKM*. 117–126.

[25] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. 2013. Stochastic variational inference. *JMLR* 14, 5 (2013).

[26] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. 2018. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *ICML*. 2304–2313.

[27] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[28] Ryuichi Kiryo, Gang Niu, Marthinus C Du Plessis, and Masashi Sugiyama. 2017. Positive-unlabeled learning with non-negative risk estimator. In *NeurIPS*. 1675–1685.

[29] Chengjiang Li, Yixin Cao, Lei Hou, Jiaxin Shi, Juanzi Li, and Tat-Seng Chua. 2019. Semi-supervised Entity Alignment via Joint Knowledge Embedding Model and Cross-graph Model. In *EMNLP-IJCNLP*. 2723–2732.

[30] Chaozhuo Li, Senzhang Wang, Hao Wang, Yanbo Liang, Philip S Yu, Zhoujun Li, and Wei Wang. 2019. Partially shared adversarial learning for semi-supervised multi-platform user identity linkage. In *CIKM*. 249–258.

[31] Chaozhuo Li, Senzhang Wang, Philip S Yu, Lei Zheng, Xiaoming Zhang, Zhoujun Li, and Yanbo Liang. 2018. Distribution distance minimization for unsupervised user identity linkage. In *CIKM*. 447–456.

[32] Xiaoli Li and Bing Liu. 2003. Learning to classify texts using positive and unlabeled data. In *IJCAI*. 587–592.

[33] Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip S Yu. 2003. Building text classifiers using positive and unlabeled examples. In *ICDM*. 179–186.

[34] Bing Liu, Wee Sun Lee, Philip S Yu, and Xiaoli Li. 2002. Partially supervised classification of text documents. In *ICML*. 387–394.

[35] Li Liu, William K Cheung, Xin Li, and Lejian Liao. 2016. Aligning Users across Social Networks Using Network Embedding.. In *IJCAI*. 1774–1780.

[36] Tongliang Liu and Dacheng Tao. 2015. Classification with noisy labels by importance reweighting. *TPAMI* 38, 3 (2015), 447–461.

[37] Yangwei Liu, Hu Ding, Danyang Chen, and Jinhui Xu. 2017. Novel geometric approach for global alignment of PPI networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[38] Zhiyuan Liu, Yixin Cao, Liangming Pan, Juanzi Li, and Tat-Seng Chua. 2020. Exploring and Evaluating Attributes, Values, and Structure for Entity Alignment. In *EMNLP*. 6355–6364.

[39] Yueming Lyu and Ivor W Tsang. 2019. Curriculum Loss: Robust Learning and Generalization against Label Corruption. In *ICLR*.

[40] Huda Nassar, Nate Veldt, Shahin Mohammadi, Ananth Grama, and David F Gleich. 2018. Low rank spectral network alignment. In *Web Conf.* 619–628.

[41] Shichao Pei, Lu Yu, Robert Hoehndorf, and Xiangliang Zhang. 2019. Semi-supervised entity alignment via knowledge graph embedding with awareness of degree difference. In *Web Conf.* 3130–3136.

[42] Shichao Pei, Lu Yu, Guoxian Yu, and Xiangliang Zhang. 2020. Rea: Robust cross-lingual entity alignment between knowledge graphs. In *KDD*. 2175–2184.

[43] Shichao Pei, Lu Yu, and Xiangliang Zhang. 2019. Improving cross-lingual entity alignment via optimal transport. In *IJCAI*. 3231–3237.

[44] Anastasia Pentina, Viktoriia Sharmanska, and Christoph H Lampert. 2015. Curriculum learning of multiple tasks. In *CVPR*. 5492–5500.

[45] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*. 452–461.

[46] Yanyao Shen and Sujay Sanghavi. 2019. Learning with bad training data via iterative trimmed loss minimization. In *ICML*. 5739–5748.

[47] Xiaofei Shi and Yanghua Xiao. 2019. Modeling Multi-mapping Relations for Precise Cross-lingual Entity Alignment. In *EMNLP-IJCNLP*. 813–822.

[48] Kai Shu, Suhang Wang, Jiliang Tang, Reza Zafarani, and Huan Liu. 2017. User identity linkage across online social networks: A review. *ACM SIGKDD Explorations Newsletter* 18, 2 (2017), 5–17.

[49] Sen Su, Li Sun, Zhongbao Zhang, Gen Li, and Jielun Qu. 2018. MASTER: across Multiple social networks, integrate Attribute and STructure Embedding for Reconciliation.. In *IJCAI*. 3863–3869.

[50] Sainbayar Sukhbaatar, Joan Bruna Estrach, Manohar Paluri, Lubomir Bourdev, and Robert Fergus. 2015. Training convolutional networks with noisy labels. In *ICLR*.

[51] Zequn Sun, Wei Hu, and Chengkai Li. 2017. Cross-lingual entity alignment via joint attribute-preserving embedding. In *ISWC*. 628–644.

[52] Zequn Sun, Wei Hu, Qingheng Zhang, and Yuzhong Qu. 2018. Bootstrapping Entity Alignment with Knowledge Graph Embedding.. In *IJCAI*. 4396–4402.

[53] Zequn Sun, Chengming Wang, Wei Hu, Muhao Chen, Jian Dai, Wei Zhang, and Yuzhong Qu. 2020. Knowledge graph alignment network with gated multi-hop neighborhood aggregation. In *AAAI*. 222–229.

[54] Brendan Van Rooyen and Robert C Williamson. 2017. A Theory of Learning with Corrupted Labels. *JMLR* 18, 1 (2017), 8501–8550.

[55] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.

[56] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. Irgan: A minimax game for unifying generative and discriminative information retrieval models. In *SIGIR*. 515–524.

[57] Zhichun Wang, Qingsong Lv, Xiaohan Lan, and Yu Zhang. 2018. Cross-lingual Knowledge Graph Alignment via Graph Convolutional Networks.. In *EMNLP*. 349–357.

[58] Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, Rui Yan, and Dongyan Zhao. 2019. Relation-aware entity alignment for heterogeneous knowledge graphs. In *IJCAI*. 5278–5284.

[59] Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, and Dongyan Zhao. 2020. Neighborhood Matching Network for Entity Alignment. In *ACL*. 6477––6487.
[60] Wei Xie, Xin Mu, Roy Ka-Wei Lee, Feida Zhu, and Ee-Peng Lim. 2018. Unsupervised user identity linkage via factoid embedding. In *ICDM*. 1338–1343.
[61] Xin Xin, Fajie Yuan, Xiangnan He, and Joemon M Jose. 2018. Batch is not heavy: Learning word representations from all samples. In *ACL*. 1853–1862.
[62] Hongteng Xu, Dixin Luo, Hongyuan Zha, and Lawrence Carin Duke. 2019. Gromov-wasserstein learning for graph matching and node embedding. In *ICML*. 6932–6941.
[63] Miao Xu, Bingcong Li, Gang Niu, Bo Han, and Masashi Sugiyama. 2019. Revisiting sample selection approach to positive-unlabeled learning: Turning unlabeled data into positive rather than negative. *arXiv preprint arXiv:1901.10155* (2019).
[64] Abdurrahman Yasar and Ümit V Çatalyürek. 2018. An iterative global structure-assisted labeled network aligner. In *KDD*. 2614–2623.
[65] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2017. mixup: Beyond empirical risk minimization. In *ICLR*.
[66] Qingheng Zhang, Zequn Sun, Wei Hu, Muhao Chen, Lingbing Guo, and Yuzhong Qu. 2019. Multi-view knowledge graph embedding for entity alignment. In *IJCAI*. 5429–5435.
[67] Si Zhang and Hanghang Tong. 2016. Final: Fast attributed network alignment. In *KDD*. 1345–1354.
[68] Si Zhang, Hanghang Tong, Ross Maciejewski, and Tina Eliassi-Rad. 2019. Multilevel network alignment. In *Web Conf.* 2344–2354.
[69] Yutao Zhang, Jie Tang, Zhilin Yang, Jian Pei, and Philip S Yu. 2015. Cosnet: Connecting heterogeneous social networks with local and global consistency. In *KDD*. 1485–1494.
[70] Zijie Zhang, Zeru Zhang, Yang Zhou, Yelong Shen, Ruoming Jin, and Dejing Dou. 2020. Adversarial attacks on deep graph matching. *NeurIPS* 33 (2020).
[71] Zexuan Zhong, Yong Cao, Mu Guo, and Zaiqing Nie. 2018. Colink: An unsupervised framework for user identity linkage. In *AAAI*.
[72] Hao Zhu, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2017. Iterative entity alignment via joint knowledge embeddings. In *IJCAI*. 4258–4264.
[73] Qiannan Zhu, Xiaofei Zhou, Jia Wu, Jianlong Tan, and Li Guo. 2019. Neighborhood-aware attentional representation for multilingual knowledge graphs. In *IJCAI*. 10–16.

A FURTHER DISCUSSION ON RELATED WORK

A.1 Knowledge Graph Alignment

Knowledge graph alignment (i.e., entity alignment) is a task to associate entities which share the same semantic meaning cross knowledge graphs. It provides an effective way to facilitate the knowledge integration and knowledge graph connection. Recently, many methods have been proposed to solve entity alignment problem. The pioneering work of the embedding-based method is MTransE [11] which learns embedding translation matrices to map entities in a KG to their corresponding entities in another KG. Then several methods [52, 72] use bootstrapping algorithm to extend MTransE to its iterative versions. The unlabeled data also have been explored and leveraged [29, 41] to promote performance. GCN-Align [57] first tries to use GCNs to encode KGs into a unified feature space. MuGNN [8] and neighborhood matching models [53, 59] further improve the basic GCN-Align by employing self-attention mechanism and utilizing contextual information to encode KGs. Besides, several approaches [47, 58, 73] focus on the utilization of relation information to comprehensively encode KGs. Meanwhile, the attribute information of entities has been integrated into the alignment models [10, 38, 51, 66].

B FURTHER DISCUSSION ON PROPOSED METHOD

B.1 More Concept Explanations

Noise discrimination model. We roughly introduce the original noise discrimination model discussed in the introduction to help readers to understand the motivation of this paper. The noise discrimination model is designed as a binary classifier for distinguishing true positive node pairs from noisy node pairs. We do not know which node pair in training data is noise, negative sampling provides a feasible way to imitate the real noise data [42]. After obtaining the sampled negative (noise) data, the binary classifier can be trained using the noisy node pairs and trusted positive labeled node pairs. Since the noise discrimination model is a binary classifier, it suffers from the issues like the inaccurate decision boundary caused by negative sampling. In our work, we focus on designing a strategy to avoid negative sampling in the discrimination model.

1-to-1 node alignment. 1-to-1 node alignment in this paper means that a node in graph $\mathcal{G}_i$ can only be aligned to *at most* one node in graph $\mathcal{G}_j$. It is *not* guaranteed that every node in graph $\mathcal{G}_i$ can be aligned to one in $\mathcal{G}_j$. 1-to-1 alignment is a common constraint in recent works such as [42, 52], we follow the previous work for the problem setting.

B.2 Time Complexity

Curriculum learning. For the computational cost of curriculum learning, CPUGA has the same time complexity as CPUGA-w/o-CL (without curriculum learning) because (1) the global optimal of loss function $\mathcal{L}_{sp}$ in Eq.(17) can be reached with the analytical solution in Eq.(18) which has the same time complexity as CPUGA-w/o-CL, and (2) the time complexity of pacing function $f_r(\lambda)$ is $O(1)$. Therefore, the time complexity of the model with curriculum learning in each iteration is the same as that of the model without curriculum learning. In practice, we control the curriculum speed by hyperparameter c in the pacing function. Thus CPUGA needs more iterations to reach the convergence and has a little higher computational cost than CPUGA without curriculum learning.
The whole CPUGA model. The updating of GNNs in each iteration mainly involves the updating of node vectors and weight matrices, whose time complexity is $O(n_t \cdot d^2 + z \cdot d)$, where $n_t = n+m$ and z are the total number of nodes and the total number of edges in graph $\mathcal{G}_i$ and $\mathcal{G}_j$, respectively. d is the embedding dimensionality. Then the time complexity of updating the non-sampling discrimination model with class prior estimation in each iteration is $O(n_s \cdot d^2 + n_s)$ where n_s is the number of positive and unlabeled samples for training the discrimination model. And the time complexity of curriculum learning in each iteration is $O(n^- \cdot d^2)$ where n^- is the number of samples in $\mathcal{Y}^-$. Hence, the complexity of training our model per iteration is $O(n_t \cdot d^2 + z \cdot d + n_s \cdot d^2 + n_s + n^- \cdot d^2)$. Since $n_s \ll n_t$ and $n^- \ll n_t$, the complexity of our model is linear with $(n_t \cdot d^2 + z \cdot d)$ and depends on the number of nodes and edges, indicating the efficiency and scalability of our model.

B.3 Training Procedure

The training process of CPUGA is outlined in Algorithm 1.

Algorithm 1: Training Procedure of CPUGA

```
Require: Graph $\mathcal{G}_i$ and $\mathcal{G}_j$, trusted positive labeled node
  pairs $\mathcal{Y}^+$, untrustful labeled node pairs $\mathcal{Y}^-$, unlabeled
  node pairs $\mathcal{U}$;
Initialize embeddings $\mathbf{H}^i$, $\mathbf{H}^j$, parameters in discrimination
  model $D$ and graph encoder $\Phi(\cdot)$;
Initialize the weight parameter $\mathbf{v}$ for curriculum learning;
Initialize the parameter $\theta$ of variational distribution;
Initialize $\mathcal{Y}^T$ using $\mathcal{Y}^+$;
while not converge do
    foreach Iteration do
        Sample a batch of node pairs from $\mathcal{Y}^T$;
        Update $\mathbf{H}^i$, $\mathbf{H}^j$, and graph encoder $\Phi(\cdot)$ according to
          $\mathcal{L}_{\text{GA}}$ in Eq. (4).
    end
    foreach Iteration do
        Sample a batch of node pairs from $\mathcal{Y}^+$ and $\mathcal{U}$;
        Calculate $\mathcal{L}_{\text{prior}}$ by Monte Carlo Sampling in Eq.
          (13), and update parameter $\theta$;
        Obtain class prior according to Eq. (14) ;
    end
    foreach Iteration do
        Sample a batch of node pairs from $\mathcal{Y}^T$, $\mathcal{U}$, and $\mathcal{Y}^+$;
        Update discriminator $D$ according to $\mathcal{L}'_{\text{R}}(D)$ in Eq.
          (19) ;
    end
    foreach Batch do
        Sample a batch of node pairs from $\mathcal{Y}^-$;
        Update weight parameter $\mathbf{v}$ according to Eq. (17);
    end
    Update the pacing function $f_r(\cdot)$;
    Update the set of augmented positive node pair $\mathcal{Y}^T$
      using $\mathbf{v}$;
end
Return: Embedding $\mathbf{H}^i$ and $\mathbf{H}^j$;
```

Table 5: Statistics of Datasets.

Datasets		Entities	Relations	Triples
DBP15K$_{\text{ZH-EN}}$	Chinese	66,469	2,830	153,929
	English	98,125	2,317	237,674
DBP15K$_{\text{JA-EN}}$	Japanese	65,744	2,043	164,373
	English	95,680	2,096	233,319
DBP15K$_{\text{FR-EN}}$	French	66,858	1,379	192,191
	English	105,889	2,209	278,590
DWY100K$_{\text{WD}}$	DBpedia	100,000	330	463,294
	Wikidata	100,000	220	448,774
DWY100K$_{\text{YG}}$	DBpedia	100,000	302	428,952
	YAGO3	100,000	31	502,563

C DESCRIPTIONS AND STATISTICS OF DATASETS

We provide the detailed information about DBP15K and DWY100K dataset in Table 5.

- DBP15K [51] consists of three cross-lingual knowledge graphs extracted from DBpedia, including DBP15K$_{\text{ZH-EN}}$ (Chinese to English), DBP15K$_{\text{JA-EN}}$ (Japanese to English), and DBP15K$_{\text{FR-EN}}$ (French to English), each includes 15,000 labeled entity pairs.
- DWY100K [52] consists of two mono-lingual knowledge graphs extracted from DBpedia, Wikidata and YAGO3, which include DWY100K$_{\text{WD}}$ (DBpedia to Wikidata) and DWY100K$_{YD}$ (DBpedia to YAGO3), each includes 100,000 labeled entity pairs.

D IMPLEMENTATION DETAILS

We implement the proposed framework using Python library Tensorflow and conduct all the experiments on Intel Xeon CPU 2.50GHz and an NVIDIA GeForce RTX 2080Ti.

In our experiments, we employ a grid search to find the best hyper-parameter configuration. We set the margin parameter γ as 3.0, and use two-layer GCNs as the graph encoder. We also implement the discrimination model D as a two-layer MLP with ReLU as the activation function. The hidden units of each layer in D is set as 100 and 30. Then we set the hyper-parameters in pacing function $f_r(\cdot)$ as $a = 15$, $b = 8$ and $c = 20$. For the class prior estimation, we used Gaussian, square-root inverse Gamma, and Dirichlet distributions to model the mean, co-variance matrix and mixing coefficient variational posteriors, respectively. For Monte Carlo sampling, we set n_s as 1 following the common setting. To prepare the unlabeled data $\mathcal{U}$, we sample $N = 1\%$ of unlabeled node pairs $\left\{(e_i, e_j)|\mathbf{A}_{i,j} = 0\right\}$ as $\mathcal{U}$ to improve the efficiency of computation, owing to the enormous number of unlabeled node pairs. The dimension of node embedding is set as 200 for all methods, and the embedding of nodes and parameter matrices are initialized by drawing from a truncated normal distribution. Besides, we also find the optimal parameters or follow the original paper to achieve the best performance for baseline methods. For optimization, we apply Adam optimizer to optimize loss functions with the learning rate 0.005 for $\mathcal{L}'_{\text{R}(D)}$ and $\mathcal{L}_{\text{GA}}$, and 0.0005 for $\mathcal{L}_{\text{prior}}$. Following the previous studies, we report the Hits@1, Hits@5 and MRR (mean reciprocal rank) results to evaluate the performance of graph alignment, and report Precision, Recall and F1 score to evaluate the performance of benign pair detection. Higher Hits@1, Hits@5, MRR, Precision, Recall and F1 score indicate better performance. Each evaluation is repeated 5 times and averaged results are reported. The source code of CPUGA: https://github.com/scpei/CPUGA.