Joint Link Prediction and Network Alignment via Cross-graph Embedding

Xingbo Du[1,3], **Junchi Yan**[2,3*] and **Hongyuan Zha**[1,3]
[1]School of Computer Science and Software Engineering, East China Normal University
[2]Department of Computer Science and Engineering & MoE Key Lab of Artificial Intelligence, Shanghai Jiao Tong University
[3]KLATASDS-MOE, East China Normal University
dxb630@126.com, yanjunchi@sjtu.edu.cn, zha@sei.ecnu.edu.cn

Abstract

Link prediction and network alignment are two important problems in social network analysis and other network related applications. Considerable efforts have been devoted to these two problems while often in an independent way to each other. In this paper, we argue that these two tasks are relevant and present a joint link prediction and network alignment framework, whereby a novel cross-graph node embedding technique is devised to allow for information propagation. Our approach can either work with a few initial vertex correspondences as seeds or from scratch. By extensive experiments on public benchmarks, we show that link prediction and network alignment can benefit each other especially for improving the recall for both tasks.

1 Introduction

Network alignment refers to finding vertex correspondences between two networks, based on both the (optional) node-wise features and edge (i.e. link) structures around vertices. In particular, node embedding models have recently received intensive attention for scalable treatment on large-scale networks with applications for node classification, etc..

Another related and widely studied problem is link prediction [Backstrom and Leskovec, 2011; Zhang and Philip, 2015], as proposed in [Liben-Nowell and Kleinberg, 2003], which aims to infer missing links in the network based on the observed links. It has a range of applications such as recommendation, knowledge management, relation mining, etc..

We argue in the paper that link prediction refers to the problem of network structure discovery and inherently relates to network alignment. Despite such a potential connection, to one's surprise, little work [Zhang and Philip, 2015] has explored the possibility for jointly solving these two tasks in a unified model. We make an initial effort in this direction and develop a cross-graph embedding model, whereby the two tasks are alternatively performed. As the structure information becomes richer by newly added links, new possible node correspondences can be identified and vice versa. As such, more links within each graph, together with the node correspondences between graphs, can be identified, leading to an improvement on recall for both tasks.

In a nutshell, the main contributions of this work are:

1) We present a joint link prediction and network alignment framework, in contrast to the majority of literature focusing on either link prediction or network alignment alone.

2) We develop a cross-graph embedding method based on random walks, whereby a new random walk formula is devised to facilitate network alignment. Also, a new cross-graph embedding based link prediction method is devised.

3) We perform extensive experiments on public benchmarks to show that alternatively performing the two tasks can benefit each other. Especially the recall can be improved for both link prediction and node correspondence establishment. Furthermore, we show how the distribution of the network's degrees relates to the effectiveness of our model.

2 The Proposed Method

We present our joint link prediction and network alignment approach, based on node embedding across networks. The two tasks can alternatively benefit to each other, as such the recall can be improved over the bootstrapping procedure.

2.1 Notations and Preliminaries

In this paper, we only consider unweighted and undirected networks and the directed version is left for future work. Also, we focus on the setting involving two networks for notational simplicity in line with the majority of literature.

Link pattern in two networks for alignment may differ in many aspects, including density, distributions, etc.. While a basic assumption is that they share corresponding vertices and some links in each network are missing due to different reasons. If we re-link all these lost edges, the aligned networks will be the same or highly similar, thus the aligning process will be easier. On the other hand, after aligning the networks, several pairs of aligned vertices will be found and will improve the link prediction in turn, as depicted in Fig. 1. Hence, we propose a cross-graph embedding-based method that alternately performs link prediction and network alignment to improve the performance of both, in a bootstrapping way.

We first give basic notations to facilitate the later discussion.

*Junchi Yan is corresponding author. The work was supported by the Open Research Fund of Key Lab of Advanced Theory and Application in Statistics and Data Science (East China Normal University), Ministry of Education, and NSFC 61602176 & 61672231.

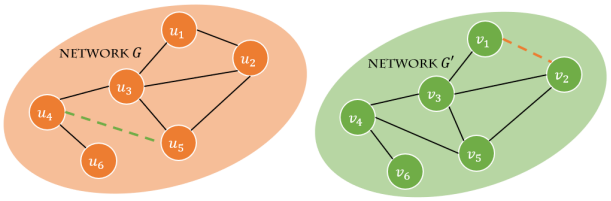


Figure 1: Illustration for how joint modeling of link prediction and network alignment can benefit to each other. Given the correspondences $v_1 = \pi(u_1), v_2 = \pi(u_2)$, it is convincing that (v_1, v_2) should be linked since (u_1, u_2) is linked. Conversely, if (u_4, u_5) and (v_4, v_5) are linked, u_4 is more likely to align with v_4 (similarly u_5 corresponds to v_5), as they have similar local structures.

Network alignment. Given two networks for alignment (G, G', π) where $G = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ is network with $\mathcal{V}, \mathcal{E}, \mathcal{A}$ as its vertices, edges and attributes respectively. $\pi : \mathcal{V} \to \mathcal{V}'$ denotes correspondence between G and G': $u = \pi(v)$,$v = \pi^{-1}(u)$ and vertices $u \in \mathcal{V}, v \in \mathcal{V}'$ correspond to each other.

Seed vertices and seed set. For network alignment, we denote the set of vertices whose correspondences have been established, via certain means or prior given beforehand by $\mathcal{S}$. In this paper, we address the unsupervised setting and assume no seed vertices are available in the beginning.

Link prediction within a network. Given networks G, G' and their alignment π, link prediction is to predict the probability of each unobserved link, which is defined as $P_G(u_1, u_2)$, for u_1, u_2 being the two unlinked vertices in G.

The k-hop neighbors. Vertex u is called v's k-hop neighbor if the distance from u to v is exactly k. Denote $\mathcal{N}_{G,k}(u)$ as the set of all k-hop neighbors of vertex u in network G.

Topological and structural similarity. In this paper, we define the inverse of the shortest path distance between two vertices as their topological similarity. While their structural similarity is defined by Eq. 1 and they may not be connected via a series of links. It is useful for cross-graph modeling.

2.2 Node Embedding

Most of recent node embedding techniques on one graph, including DeepWalk [Perozzi *et al.*, 2014], node2vec [Grover and Leskovec, 2016], LINE [Tang *et al.*, 2015], etc., are not tailored to and cannot measure the similarity across separate networks (at least on the surface). For network alignment, even if the two networks are connected through seed vertices, the corresponding vertices in different networks may still be dissimilar to each other regarding with their embedding vectors (as will be seen in Fig. 7). Though some node embedding techniques like struc2vec [Figueiredo *et al.*, 2017], to some extent, can obtain the structural similarities and alleviate the above problems, the topological similarity is little considered. Therefore, we propose a novel cross-graph node embedding technique that can obtain both the structural and topological similarities for nodes across networks.

Given G, G' and their alignment π, we construct a weighted, undirected compound network. Given vertices $u \in \mathcal{V}$, $v \in \mathcal{V}'$, the structural distance between u and v is defined as $f(u, v)$ when considering their k-hop neighbors $(k = 0, 1, \ldots, K)$, where the 0-hop neighbor is defined as

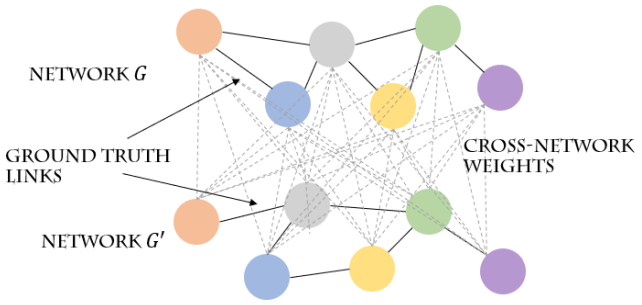


Figure 2: Weighted, undirected network $\tilde{G}$ consisting of two networks with cross-graph links among vertices in two networks.

the vertex itself and K is a hyperparameter that controls the depth of structural similarities. Specifically, we define the difference between the two vertices from two networks by

$$f(u, v) = \sum_{k=0}^{K} \text{dist}(s_k(u), s_k(v)), \tag{1}$$

where $s_k(u)$ is the rectified degree sequence of vertices of u's k-hop neighbors, i.e. $s_k(u) = [\tilde{d}_1, \tilde{d}_2, \ldots]$, in which the rectified degree is specified by $\tilde{d} = d\sqrt{\frac{|\mathcal{E}'|/|\mathcal{V}'|}{|\mathcal{E}|/|\mathcal{V}|}}$ for each degree d in order to maintain consistency of degree distributions in two networks. The $|\cdot|$ here is number of elements in a set. The $\text{dist}(s_k(u), s_k(v))$ in Eq. 1 is specified as:

$$\begin{aligned} \text{dist} = &\left| \min_{d \in s_k(u)} \log(d+1) - \min_{d \in s_k(v)} \log(d+1) \right| + \\ &\left| \max_{d \in s_k(u)} \log(d+1) - \max_{d \in s_k(v)} \log(d+1) \right|. \end{aligned} \tag{2}$$

Note u, v are vertices in different networks, thus $f(u, v)$ focuses on cross-graph structural similarity. Moreover, note that only minimum and maximum degrees are concerned due to computational complexity, while we argue these two features can approximately obtain the vertices' distance[1]. We use Eq. 2 to measure the distance between u and v, however, any other technique to evaluate distance can be applied in this framework. The weight of each pair is defined as

$$w(u, v) = e^{-\alpha \cdot f(u,v)}, \tag{3}$$

where α is a hyperparameter that controls the distribution of weights. Then we get a weighted, undirected compound network, which we denote as $\tilde{G}$, as depicted in Fig. 2. We consider a biased random walk around $\tilde{G}$, as specified by:

1. Given a probability $q > 0$ to decide whether to walk on the current network or switch to another network (with probability q walking on the current network and probability $1 - q$ for network switching).
2. If the walk is on the current network, the probability of walking from vertex $u \in \mathcal{V}$ to $v \in \mathcal{N}_{G,1}(u)$ is

$$p(u, v) = \frac{1}{|\mathcal{N}_{G,1}(u)|}. \tag{4}$$

[1]Outside this paper, we verified this form in comparison with other forms which incorporate more fine-grained information while we find the used one in the paper is more cost-effective.

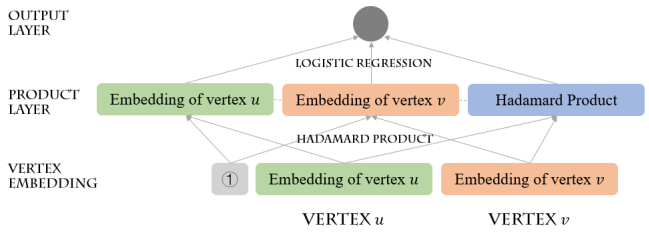


Figure 3: Binary classifier based on product neural network for binary link prediction within a network.

3. Given that the walk will switch networks, if the current vertex $u \in \mathcal{V}$ is a seed vertex, namely $u \in \mathcal{S}$, the probability of walking from vertex $u \in \mathcal{V}$ to $v = \pi(u)$ is:

$$p(u, v) = 1. \tag{5}$$

4. If the walk will switch networks and the current vertex $u \in \mathcal{V}$ is not a seed vertex, then the probability of walking from vertex $u \in \mathcal{V}$ to $v \in \mathcal{V}'$ is:

$$p(u, v) = \frac{w(u, v)}{Z(u)}, \tag{6}$$

where $Z(u)$ is the normalization factor for vertex u:

$$Z(u) = \sum_{v \in G'} w(u, v). \tag{7}$$

Finally we use random walks to train a Skip-gram model to obtain the embedding. In particular, given a vertex, the objective of Skip-gram model is to maximize the average log probability of its context in a sequence, where the vertex's context is given by the nearby vertices in a sequence.

Since basic Skip-gram model suffers from the computational complexity, we use Negative Sampling [Mikolov *et al.*, 2013b], whose objective is defined as follows:

$$\log \sigma(\mathbf{x'}_{v_j}^{\top} \cdot \mathbf{x}_{v_i}) + \sum_{i=1}^{K_{neg}} E_{v_n \sim P_n(v)}[\log \sigma(-\mathbf{x'}_{v_n}^{\top} \cdot \mathbf{x}_{v_i})], \tag{8}$$

where $\mathbf{x}_v$ and $\mathbf{x}'_v$ are the input and output vector representations of v respectively. $\sigma(x) = 1/(1 + e^{-x})$ is the sigmoid function and K_{neg} is the number of negative edges. Then we set $P_n(v) \propto d_v^{3/4}$, where d_v is the degree of vertex v. By Eq. 8, the vertices' vector representations can be learned.

2.3 Network Alignment

Varieties of alignment methods e.g. [Singh *et al.*, 2008] try to find the best overall alignments, in which each vertex in a network will be matched one or more vertices in another aligned network simultaneously. In contrast, techniques also exist [Koyutürk *et al.*, 2005] that they aim to find similar motifs between aligned networks. Our method will combine the benefits of the above methods and focus on finding the most confident vertex correspondences between networks periodically. Here we further define $\mathcal{N}_k(\mathcal{S})$ as all k-hop neighbors of the vertices in G. Given vertices $u \in \mathcal{V}, v \in \mathcal{V}'$, we also

Algorithm 1: Network Alignment (NA)

Input: Networks $G = (\mathcal{V}, \mathcal{E}, \mathcal{A})$, $G' = (\mathcal{V}', \mathcal{E}', \mathcal{A}')$; seed vertices $\mathcal{S}$ and $\mathcal{S}'$; alignment π; embedding $\{\mathbf{x}_u\}_{u \in \mathcal{V}}$, $\{\mathbf{x}_v\}_{v \in \mathcal{V}'}$; number of vertex pairs T.

```
Compute F = {sim(u, v)}_{u∈V, v∈V'} by Eq. 9 to 13;
while T > 0 do
    T = T - 1;
    Find a new vertex pair (u*, v*) according to F;
    Add new vertex to seed set of G: S = S ∪ {u*};
    Add new vertex to seed set of G': S' = S' ∪ {v*};
    Establish the new correspondence: π(u*) = v*.
```

Output: Updated seed vertices $\mathcal{S}$, $\mathcal{S}'$; alignment π.

define the embedding similarity of two vertices in different networks by cosine similarity

$$\text{sim}_{\text{emb}}(u, v) = \max\left\{\frac{\mathbf{x}_u \cdot \mathbf{x}_v}{||\mathbf{x}_u|| \cdot ||\mathbf{x}_v||}, 0\right\}, \tag{9}$$

where $\mathbf{x}_u$ is the embedding vector of u and $||\cdot||$ is 2-norm of a vector. In order to align the most confident vertices, we only consider 1-hop neighbors of $\mathcal{S}$ and use a variant of Jaccard similarities to rectify Eq. 9, therefore, given u and v in $\mathcal{N}_1(\mathcal{S})$ and $\mathcal{N}_1(\mathcal{S}')$ respectively, the similarity between them is

$$\begin{aligned} \mathcal{CN}_G &= \mathcal{N}_{G,1}(u) \cap \mathcal{S}, \\ \mathcal{CN}_{G'} &= \mathcal{N}_{G',1}(v) \cap \mathcal{S}', \\ \text{sim}_{\text{jc}}(u, v) &= \frac{|\pi(\mathcal{CN}_G) \cap \mathcal{CN}_{G'}|}{|\pi(\mathcal{CN}_G) \cup \mathcal{CN}_{G'}|}, \end{aligned} \tag{10}$$

$$\text{sim}_{\text{graph}}(u, v) = \text{sim}_{\text{emb}}(u, v) \cdot \text{sim}_{\text{jc}}(u, v), \tag{11}$$

where $|\cdot|$ denotes the size of the set. Note Eq. 10 suggests if two vertices in different networks share similar corresponding vertices, they tend to have higher similarity.

When attribute $\mathcal{A}$ is given, the attribute similarity between vertex $u \in \mathcal{N}_1(\mathcal{S})$ and $v \in \mathcal{N}_1(\mathcal{S}')$ can be written as

$$\text{sim}_{\text{attr}}(u, v) = \max\left\{\frac{\mathbf{y}_u \cdot \mathbf{y}_v}{||\mathbf{y}_u|| \cdot ||\mathbf{y}_v||}, 0\right\}, \tag{12}$$

where $\mathbf{y}_u$ is the attribute vector of vertex u. Thus, the similarity between u and v can be further written as

$$\text{sim}(u, v) = \text{sim}_{\text{graph}}(u, v) \cdot \text{sim}_{\text{attr}}(u, v). \tag{13}$$

At last, we will search the most confident pairs of vertices as is shown in Alg. 1. If (u, v) is a new pair of seed vertices, then it will be added to $\mathcal{S}$ and $\mathcal{S}'$ respectively.

2.4 Link Prediction

In general, given a network G, link prediction refers to predicting the probability of each unobserved link, which can be written by $P_G(u_1, u_2)$, with u_1, u_2 being two unlinked vertices in G. In our task, a natural idea is that new links can be identified between u_1 and u_2 in one graph if the corresponding $v_1 = \pi(u_1), v_2 = \pi(u_2)$ were already connected in G'.

Therefore, we propose a link prediction method within a network for alignment that takes into account cross-graph information, as shown in Alg. 2.

Algorithm 2: Cross-graph Link Prediction (LP)

Input: Networks $G = (\mathcal{V}, \mathcal{E}, \mathcal{A})$, $G' = (\mathcal{V}', \mathcal{E}', \mathcal{A}')$; seed vertices $\mathcal{S}$, $\mathcal{S}'$; alignment π; threshold thr.

Randomly sample a group of existent links $\mathcal{E}_{ext}$, $\mathcal{E}'_{ext}$ and nonexistent links $\mathcal{E}_{mis}$, $\mathcal{E}'_{mis}$ in G and G' respectively and their labels o, o'; Randomly initialize link prediction classifier's parameters $W = (w, b)$ and $W' = (w', b')$;

while *W and W' not converged* **do**

  Compute loss function with training links $\mathcal{E}_{ext}$, $\mathcal{E}_{mis}$ and $\mathcal{E}'_{ext}$, $\mathcal{E}'_{mis}$ by Eq. 14;

  Update W and W' by Eq. 15;

Construct edge lists $\mathcal{E}_{test} = \mathcal{S} \times \mathcal{S}$ and $\mathcal{E}'_{test} = \mathcal{S}' \times \mathcal{S}'$;

for *(u_1, u_2) in $\mathcal{E}_{test}$ and $(\pi(u_1), \pi(u_2))$ in $\mathcal{E}'_{test}$* **do**

  Compute objective $l_2(u_1, u_2)$ and $l'_2(\pi(u_1), \pi(u_2))$ with parameters W' and W respectively by Eq. 14.

  //Note that W and W' are reversed.

  if *$l_2(u_1, u_2) > thr$ and $l'_2(\pi(u_1), \pi(u_2)) > thr$* **then**

    $\mathcal{E} = \mathcal{E} \cup \{e = (u_1, u_2)\}$;

    $\mathcal{E}' = \mathcal{E}' \cup \{e' = (\pi(u_1), \pi(u_2))\}$;

Output: Updated networks G and G'.

Given a group of existent and missing edges in graph G, we construct a training set for link prediction learning. As illustrated in Fig. 3, for edge $e = (u_1, u_2) \in \mathcal{E}$ and the node embedding of u_1 and u_2, we adopt a product layer [Qu *et al.*, 2016] to extract the latent interaction between these two vertices, which is followed by a Logistic regression layer for binary classification. Formally, for input edge $e = (u_1, u_2) \in \mathcal{E}$ and its label $o_e \in \{0, 1\}$, we have

$$\begin{aligned} l_0(e) &= \text{concat}(\mathbf{x}_{u_1}, \mathbf{x}_{u_1}, \mathbf{x}_{u_1} \circ \mathbf{x}_{u_2}), \\ l_1(e) &= w \cdot l_0(e) + b, \\ l_2(e) &= \sigma[l_1(e)] = \frac{1}{1 + e^{-l_1(e)}}, \end{aligned} \tag{14}$$

where $\circ$ is Hadamard product, formally $\mathbf{x}_{u_1} \circ \mathbf{x}_{u_2}$ multiplies each corresponding elements in $\mathbf{x}_{u_1}$ and $\mathbf{x}_{u_2}$ with $(\mathbf{x}_{u_1} \circ \mathbf{x}_{u_2})_i = (\mathbf{x}_{u_1})_i \cdot (\mathbf{x}_{u_2})_i$. The objective for link prediction classifier can be defined as:

$$\min_{w,b} \frac{1}{|\mathcal{E}|} \sum_{e \in \mathcal{E}} -o_e \log l_2(e) - (1 - o_e) \log \big(1 - l_2(e)\big). \tag{15}$$

We denote the above classifier as $\mathcal{C}$. To approximate the links in G and G', $\mathcal{C}$ is utilized to perform cross prediction. Specifically, given $u_1, u_2 \in \mathcal{S}$, $e = (u_1, u_2) \in \mathcal{E}$, if $\mathcal{C}$ predicts that $e' = (\pi(u_1), \pi(u_2))$ links in G', then $e' \in \mathcal{E}'$.

Note that classifiers $\mathcal{C}$ and $\mathcal{C}'$ only predict the links in their respective graphs that are connected to the seed vertices with established correspondences. This strategy avoids error accumulation by noise. The overall procedure is shown in Alg. 3.

3 Related Work

We discuss related work in node embedding, network alignment and link prediction, as involved in our approach.

Algorithm 3: Cross-graph Node Embedding for Joint Network Alignment and Link Prediction (CENALP)

Input: Networks $G = (\mathcal{V}, \mathcal{E}, \mathcal{A})$, $G' = (\mathcal{V}', \mathcal{E}', \mathcal{A}')$; seed vertices $\mathcal{S}$ and $\mathcal{S}'$; alignment π.

while *new vertex correspondences can be found* **do**

  Update $\{\mathbf{x}_u\}_{u \in \mathcal{V}}$, $\{\mathbf{x}_v\}_{v \in \mathcal{V}'}$ by Eq. 4 to 8;

  Search new correspondences by Alg. 1;

  Predict links within a network by Alg. 2;

Output: Updated seed vertices $\mathcal{S}$, $\mathcal{S}'$; alignment π;node embedding $\{\mathbf{x}_u\}_{u \in \mathcal{V}}$, $\{\mathbf{x}_v\}_{v \in \mathcal{V}'}$.

Node Embedding. There are scalable node embedding methods e.g. DeepWalk [Perozzi *et al.*, 2014] based on random walk and node2vec [Grover and Leskovec, 2016] inspired by Skip-gram model [Mikolov *et al.*, 2013a]. In particular, LINE [Tang *et al.*, 2015] explicitly defines *first-order* proximity and *second-order* proximity and builds heuristics models for the two proximities. However, the techniques above cannot obtain similarity between vertices in separate networks. Though struc2vec [Figueiredo *et al.*, 2017] learns node representations from structural identity and allows for learning representations in separate networks, it ignores topological neighbors that are crucial to network alignment tasks. In this paper, we propose a novel node embedding technique that can learn from both structural and topological neighbors.

Network Alignment. Network alignment has recently become an active area. Different features have been used to network alignment e.g. network structures[Singh *et al.*, 2008], node attributes [Zhang *et al.*, 2015] and edge attributes [Si and Tong, 2016]. It is common for a user getting involved in multiple social networks e.g. Foursquare, Facebook and Twitter. Such shared users among different networks are referred to as 'anchor users' [Kong *et al.*, 2013; Tong *et al.*, 2016]. In particular, [Zhang and Philip, 2015] connects the users by both social links and anchor links simultaneously. However, some positive anchor links have to be used to train their models and the metrics AUC and Precision@30 (i.e. correct alignments in top-30 choices) do not find a one-to-one correspondence. In our approach, we propose a cross-graph embedding-based aligning technique which enables the model to learn in an unsupervised way.

Link Prediction. Link prediction has been a well- studied area since the seminal work [Liben-Nowell and Kleinberg, 2003]. A line of works formulates link prediction as a supervised classification task [Al Hasan *et al.*, 2006], whereby different types of links are labeled for prediction according to their physical meaning [Backstrom and Leskovec, 2011]. Meanwhile, unsupervised methods [Xiang *et al.*, 2010] are also developed for estimating the link strength. While in our case, we perform binary prediction to estimate whether the link shall be formed or not, regardless of their detailed types.

4 Experiments

Popular datasets are used i.e. Twitter/Facebook, Douban (online and offline communities as China's popular social network), and the DBLP benchmark. For better visualization, a

Datasets		Nodes	Edges	Attribute	Node pairs
DBLP & distrubed copy	G	2,151	6,306	8	2,151
	G'	2,151	5,676		
Facebook & Twitter	G	1,256	4,734	0	1,043
	G'	1,043	4,860		
Douban online & offline	G	3,906	8,164	538	1,118
	G'	1,118	1,511		

Table 1: Statistics of the datasets tested in the experiments.

simple example is given in Fig. 4 to illustrate our method.

4.1 Protocols

The statistics of the used datasets are summarized in Table 1.

1) DBLP. It is collected by [Prado *et al.*, 2013], which can be treated as a co-authorship network. Each author can be considered as a vertex and authors' academic cooperation as the links. Each author is associated with an attribute vector representing the number of publications in computer science conferences. To generate a similar but slightly different network to align, we randomly drop 10% edges and flip 10% attribute information in line with [Si and Tong, 2016]. **2) Facebook/Twitter.** A cross-graph constructed from two real-world social networks as collected and published by [Cao and Yong, 2016]. Facebook and Twitter are the most popular worldwide online social network and micro-blog website respectively. Each social account is treated as a vertex and accounts' friend relationship as edges. If a real-world user owns both Facebook and Twitter accounts, these two accounts will be treated as an alignment between two networks. In this task, no attribute information is used. **3) Douban online/offline.** A real-world social network extracted from which is collected and published by [Zhong *et al.*, 2012]. It has an online social network and an offline social network to align. Users' locations are treated as attributes.

We compare CENALP and its variant CENA to baselines by accuracy i.e. proportion of correct node correspondences of the total correspondences. We also evaluate the impact of the missing ratio of links in our experiments.

We compare our approach with the following methods:

1) DeepWalk. Random walk based network embedding model [Perozzi *et al.*, 2014] inspired by language model; **2) Struc2vec.** Node embedding method [Figueiredo *et al.*, 2017] based on structural identity; **3) IsoRank.** Global alignment method initially with application to protein interaction networks [Singh *et al.*, 2008]; **4) FINAL.** State-of-the-art graph alignment algorithm [Si and Tong, 2016] considering both node attributes, edge attributes and graph structures. FINAL needs a prior alignment as input, which is not applied in our approach. So we treat it as attribute similarity alignment. And if not, the uniform is used; **5) CENALP.** Our proposed framework with joint alignment and link prediction over iterations; **6) CENA.** A variant of our approach with only cross-graph alignment one time without link prediction.

The parameters commonly used in the compared methods are set the same for a fair comparison. Specifically, the dimension of node embeddings, including DeepWalk, struc2vec and our proposed method, is universally set as 64. The maximum depth of neighbors to hop is set as $K = 2$. The parameter in Eq. 3 is set as $\alpha = 5$. The probability controlling whether to switch networks is set as $q = 0.3$.

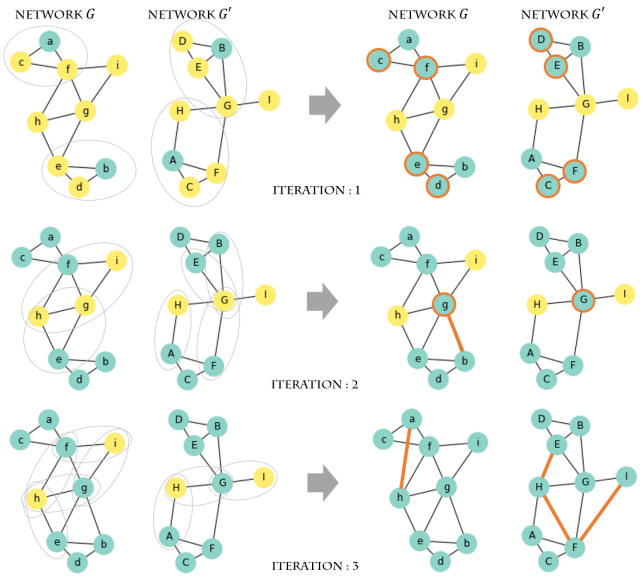
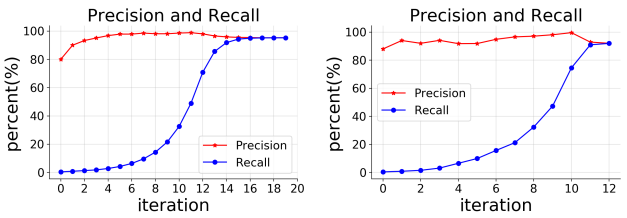


Figure 4: Iterative alignment and link prediction. Vertex a to i corresponds to A to I. Cyan vertices represent those with established correspondences between two graphs, while the yellows are not aligned. Vertices with red rings represent newly-aligned vertices and red links mean newly predicted links. For simplicity, the maximum depth of neighbors to hop is set by $K = 0$. In this ideal case, our approach jointly aligns new vertices and predict new links, and eventually makes the two networks exactly the same in structure.

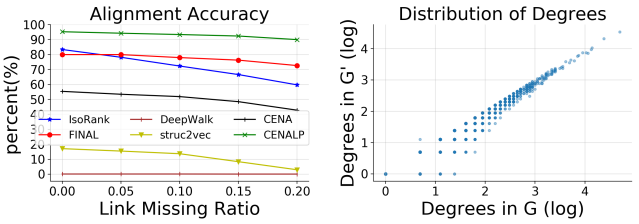


(a) DBLP & disturbed copy (b) Facebook/Twitter

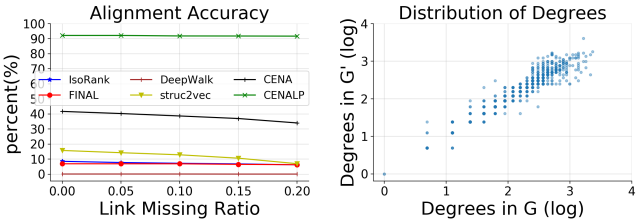
Figure 5: Precision and recall over iterations by our CENALP on DBLP and Facebook/Twitter. Note the recall grows notably. One can see that the recall grows steadily in the beginning because there is a few reliable pairs, while it grows rapidly later as seed pairs increases and eventually it becomes saturate.

4.2 Results and Discussion

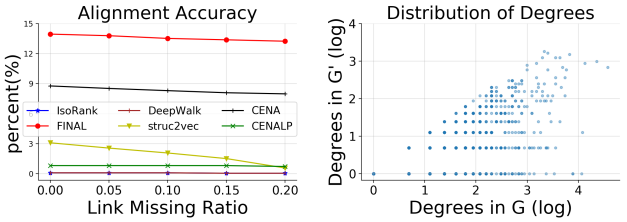
Recall and precision. As shown in Fig. 5, by joint link prediction and network alignment, recall increases while precision is stable. We evaluate alignment accuracy and influence of missing links for different methods. The results are depicted in Fig. 6. The proposed CENALP achieves the highest alignment accuracy in DBLP and Facebook/Twitter networks while CENA achieves the second highest in Douban, under different values for the ratio of missing links against all links in the datasets. In contrast, DeepWalk and struc2vec tend to drop as link missing ratio grows. Also, the scatter plots of node degrees in two networks are given on the right, from which one can see that CENALP works well when the scatters are more linearly distributed (DBLP and Facebook/Twitter



(a) DBLP and its disturbed copy

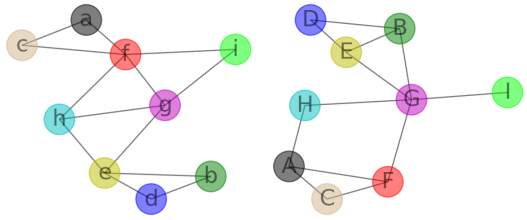


(b) Facebook/Twitter

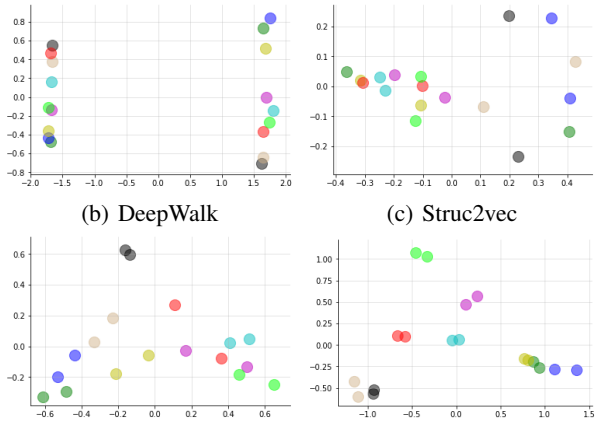


(c) Douban online/offline

Figure 6: Alignment accuracy by varying: edges missing ratio levels (left) and degree scatters after log transformation (right) of networks.



(a) Illustration by using two toy networks for alignment



(b) DeepWalk (c) Struc2vec

(d) CENA (e) CENALP

Figure 7: 2-D plot of different methods' embeddings by PCA dimension reduction. Embeddings by DeepWalk from two separate networks are distant to each other. Our cross-graph method CENA can project correspondences closely – more closely by CENALP.

in contrast to Douban). This helps users to decide if our method is applicable for their practical tasks. In general, the alignment accuracy of DeepWalk is low and sometimes approaches random guess. In our analysis, this is because DeepWalk cannot obtain similarities in separate networks. For struc2vec, it can only learn structural neighbors and ignores topological neighbors, thus not suitable for aligning tasks, though it performs better than FINAL and IsoRank in Facebook/Twitter. FINAL and IsoRank are two aligning methods that depend on both structure and prior alignment, therefore in our expectation, they perform badly in Facebook/Twitter due to the lack of attribute information and prior alignment. In summary, though our proposed methods do not always perform best, it is suitable for aligning networks whose degree scatter plots are narrowed, or in case when the attribute information is absent. Our approach learns not only information from topological neighbors (within graph), but also the one from structural ones (cross-graph), since the biased random walk is both within and cross-graph, and based on Eq. 4, 5 and 6. We compare the distribution of embedded vertices with different embedding techniques as depicted in Fig. 7, and tend to conclude that our methods perform better in obtaining both topological and structural information.

Time overhead. Though our method covers embedding, alignment and link prediction, the time cost mainly depends on sampling time for random walks. Specifically, the task for DBLP and its disturbed copy with 2,151 nodes and 22 iterations can be finished in average 94 seconds per iteration. Douban online/offline with 1,118 nodes and 18 iterations spend in average 116 seconds per iteration and Facebook/Twitter with 1,043 nodes and 12 iterations spend in average 53 seconds per iteration on our desktop with 2.1GHz CPU and 16G memory. By comparison, for the task DBLP and its disturbed copy, IsoRank and FINAL spend around 7 seconds and 16 seconds respectively while the embedding-based methods DeepWalk and struc2vec spend 86 seconds and 183 seconds respectively. Recall that our method CENA can be regarded as the running of the first iteration, which is still competitive against baselines as shown in Fig. 6. By our design, CENA focuses more on accuracy to ensure the subsequent iteration will not be influenced by wrong vertex correspondences. In fact, our method in each iteration has the same time complexity with Deepwalk.

5 Conclusion

We have presented a joint link prediction and network alignment framework for improving the recall for both of the two tasks. We also develop a cross-graph embedding technique based on structural and topological neighbors to effectively enable the node to embed from separate graphs.

We also empirically show the condition under which our bootstrapping method can perform in expectation and suggest the possible failure case. We believe this is important as inherently our method may encounter error accumulation which is common to many self-learning like methods.

References

[Al Hasan *et al.*, 2006] Mohammad Al Hasan, Vineet Chaoji, Saeed Salem, and Mohammed Zaki. Link prediction using supervised learning. In *SDM06: workshop on link analysis, counter-terrorism and security*, 2006.

[Backstrom and Leskovec, 2011] Lars Backstrom and Jure Leskovec. Supervised random walks: predicting and recommending links in social networks. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 635–644. ACM, 2011.

[Cao and Yong, 2016] Xuezhi Cao and Yu Yong. *BASS: A Bootstrapping Approach for Aligning Heterogenous Social Networks*. 2016.

[Figueiredo *et al.*, 2017] Daniel R. Figueiredo, Leonardo F. R. Ribeiro, and Pedro H. P. Saverese. struc2vec: Learning node representations from structural identity. *KDD*, 2017.

[Grover and Leskovec, 2016] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864. ACM, 2016.

[Kong *et al.*, 2013] Xiangnan Kong, Jiawei Zhang, and Philip S. Yu. Inferring anchor links across multiple heterogeneous social networks. In *Acm International Conference on Information & Knowledge Management*, 2013.

[Koyutürk *et al.*, 2005] Mehmet Koyutürk, Ananth Grama, and Wojciech Szpankowski. Pairwise local alignment of protein interaction networks guided by models of evolution. In *International Conference on Research in Computational Molecular Biology*, 2005.

[Liben-Nowell and Kleinberg, 2003] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. In *Conference on Information and Knowledge Management (CIKM'03)*, pages 556–559, 2003.

[Mikolov *et al.*, 2013a] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *Computer Science*, 2013.

[Mikolov *et al.*, 2013b] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[Perozzi *et al.*, 2014] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014.

[Prado *et al.*, 2013] Adriana Prado, Marc Plantevit, Celine Robardet, and J. F. Boulicaut. Mining graph topological patterns: Finding covariations among vertex descriptors. *IEEE Transactions on Knowledge & Data Engineering*, 25(9):2090–2104, 2013.

[Qu *et al.*, 2016] Yanru Qu, Han Cai, Kan Ren, Weinan Zhang, Yong Yu, Ying Wen, and Jun Wang. Product-based neural networks for user response prediction. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, pages 1149–1154. IEEE, 2016.

[Si and Tong, 2016] Zhang Si and Hanghang Tong. Final: Fast attributed network alignment. In *Acm Sigkdd International Conference*, 2016.

[Singh *et al.*, 2008] Rohit Singh, X. U. Jinbo, and Bonnie Berger. Global alignment of multiple protein interaction networks. *Proceedings of the National Academy of Sciences of the United States of America*, 105(35):12763–12768, 2008.

[Tang *et al.*, 2015] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1067–1077. International World Wide Web Conferences Steering Committee, 2015.

[Tong *et al.*, 2016] Man Tong, Huawei Shen, Shenghua Liu, Xiaolong Jin, and Xueqi Cheng. Predict anchor links across social networks via an embedding approach. In *International Joint Conference on Artificial Intelligence*, 2016.

[Xiang *et al.*, 2010] Rongjing Xiang, Jennifer Neville, and Monica Rogati. Modeling relationship strength in online social networks. In *Proceedings of the 19th international conference on World wide web*, pages 981–990. ACM, 2010.

[Zhang and Philip, 2015] Jiawei Zhang and S Yu Philip. Integrated anchor and social link predictions across social networks. In *IJCAI*, pages 2125–2132, 2015.

[Zhang *et al.*, 2015] Yutao Zhang, Jie Tang, and Zhiling Yang. Cosnet: Connecting heterogeneous social networks with local and global consistency. In *Acm Sigkdd International Conference on Knowledge Discovery & Data Mining*, 2015.

[Zhong *et al.*, 2012] Erheng Zhong, Wei Fan, Junwei Wang, Lei Xiao, and Yong Li. Comsoc: Adaptive transfer of user behaviors over composite social network. In *Acm Sigkdd International Conference on Knowledge Discovery & Data Mining*, 2012.