

Simultaneous Link Prediction on Unaligned Networks Using Graph Embedding and Optimal Transport

Luu Huu Phuc Koh Takeuchi Makoto Yamada Hisashi Kashima

Department of Intelligence Science and Technology

Kyoto University

Kyoto, Japan

{phuc, koh, myamada, kashima}@ml.ist.i.kyoto-u.ac.jp

Abstract—Link prediction is an extensively studied topic and various methods have been proposed to tackle the task in both heuristic and more sophisticated statistical learning approaches. However, most of them focus on the setting of one single graph. Combining information on multiple graphs with similar topological structures can improve the performance and robustness of link prediction; nevertheless, the alignment between nodes of different networks is not always available, or is only partially known. This study considers the link prediction problem on two unaligned networks simultaneously. A new framework is proposed to integrate link prediction using graph embedding and node alignment using optimal transport. The integrated objective is optimized at once via an iterative algorithm. A showcase of the proposed framework using LINE embedding method is discussed with experiments on three real datasets. The results demonstrate that the integrated formulation shows better link prediction performance over single-graph link prediction methods as well as existing methods that do not directly aim at link prediction. The framework is flexible and theoretically able to integrate with different graph embedding methods, which is demonstrated in additional experiments using node2vec.

Index Terms—Link prediction, Optimal transport, Graph embedding

I. INTRODUCTION

Graph is a general and ubiquitous data structure that plays a crucial role in various aspects of the world. Since it was first introduced in the 18th century, graph has been an important subject in many research fields. By representing relationships between entities with nodes and links among them, graph is useful to model complex systems of interactive elements. For example, in a social network, a user account can be seen as a node and the friendship between two users can be seen as a link. In a protein-protein interaction network, each node in the network represents a protein and two proteins are linked if they interact with each other. In many real world scenarios, complex networks are only partially observed, i.e. have missing or hidden links. Social network users who know each other in real life might have not been connected in a social network platform yet. In medical and biological studies, interaction graphs are incomplete because they rely only on expensive lab experiment data. Re-establishing such hidden friendships and possible biological interactions undoubtedly derives tremen-

dous benefits in both academic and industrial settings. In the machine learning context, the task of re-discovering those potential relationships is called link prediction.

Like other tasks on graph-structure data, the link prediction task is often challenging due to the difficulty of extracting necessary information from graphs. Traditional approaches for link prediction are often based on heuristic similarity scores over pairs of unconnected nodes such as the Common Neighbor index [1], the Adamic-Adar index [2], and the Jaccard similarity index [3], [4]. These heuristic scores are computationally fast and shown to be effective in capturing the local similarity in social networks [3]. Besides such unsupervised approaches, there have been proposed various supervised [5], [6] and semi-supervised [7], [8] link prediction methods. Other methods apply Matrix factorization to decompose the adjacency matrix of a graph into low-rank matrices [9], [10]. Menon et al. [11] solve the link prediction task as a matrix completion problem using non-negative matrix factorization methods. With the recent advances in deep neural networks and natural language processing [12], graph embedding approaches have shown favorable results on the link prediction task. The basic idea behind the graph embedding approaches is to represent each node as a vector in a low-dimensional latent vector space so that the similarity of nodes in the original graph is preserved as the vector proximity in the latent space. Some of the most prominent methods on this approach are LINE [13], Deepwalk [14], and node2vec [15].

The researches on link prediction problem are profound. However, existing works are specialized to only deal with a single graph. In real world scenarios, it is not rare that multiple graph data from related domains co-exist. An example can be a Facebook network and a Twitter network that share many common users. It is reasonable to think that if there exists a link between a pair of users in one network, there likely also exists a link between the corresponding pair in the other network. Naturally, by sharing link information across related graphs, the link prediction on each individual graph can be greatly improved. The ideal scenario for propagating link information across graphs is when a complete alignment between corresponding nodes is available; e.g., it is clear that

which user in the Facebook network corresponds to which user in the Twitter network. However, such alignment is not always available, or is only partially known. Finding a reasonable alignment is not easy, making link prediction on multiple unaligned graphs a challenging task.

This paper aims to address the link prediction problem on two unaligned graphs via an integrated optimization for learning graph embedding and node alignment at once. The objective is to improve the link prediction performance on each individual graph. Our contribution is to propose a novel framework that allows link information sharing across graphs by combining two recently successful ideas: network embedding and optimal transport. On the one hand, an existing network embedding method such as LINE method [13] is employed to capture link information in each graph. The nodes in each graph will be mapped with latent embedding vectors in low-dimensional space such that the latent vectors of two nodes will stay close in the embedding space if the nodes are linked or similar. On the other hand, link information is propagated across the two graphs by making the latent vectors of corresponding nodes become closer. This process is under the guidance of optimal transport who provides a reasonable alignment in the form of an optimal transport plan between the node sets. Although the embedding process is based on existing network embedding methods, the proposed framework is more desirable thanks to its ability to propagate link information. Instead of being learned independently, the embedding processes of both graphs are correlated and complement each other. An iterative algorithm is designed to optimize the proposed framework in a single optimization problem, which converges to a local optimal solution.

There are a few studies addressing the link prediction problem in unaligned graphs. Xu et al. [16] give another formulation of embedding of unaligned networks using the Gromov-Wasserstein distance, which is the most technically related work, but their embedding aims at preserving the distances between the nodes on each graph and does not focus on preserving link structure. Du et al. [17] consider the most similar problem setting, but their focus is rather on node alignment, and not much optimized for link prediction.

Experiments are conducted on multiple datasets with different numbers of nodes and levels of density. The proposed method and baselines are compared with different values for the size of the training data (i.e., the number of known links) and the number of common nodes between the graphs. Empirically, the proposed method works best with graphs of high variance degree distributions, where the node degrees are diverse and indicate different levels of importance of the nodes in the graphs.

The remainder of this paper is organized as follows. The problem setting and notation are defined in section II. Section III focuses on the proposed method, presenting the graph embedding method and optimal transport approximation method employed in the framework. The effectiveness of the proposed method is demonstrated in section IV with experiments on multiple datasets. Section V reviews related work. Finally,

TABLE I
SUMMARY OF THE SYMBOLS AND NOTATIONS

$G^{(k)}$	one of the unaligned graphs ($k \in \{1, 2\}$)
$V^{(k)}$	the set of nodes in graph $G^{(k)}$
$E^{(k)}$	the set of links in graph $G^{(k)}$
$W^{(k)}$	the weight matrix corresponding to $E^{(k)}$
$n^{(k)}$	the number of nodes in $V^{(k)}$
$v_i^{(k)}$	a node in $V^{(k)}$
$w_{ij}^{(k)}$	the link weight between $v_i^{(k)}$ and $v_j^{(k)}$
d	the embedding dimension
$\mathbf{x}_i^{(k)}$	the embedding vector of node $v_i^{(k)}$
$\mathbf{X}^{(k)}$	the embedding matrix with columns of $\mathbf{x}_i^{(k)}$
$\mathbf{x}'_i^{(k)}$	the ‘‘context’’ embedding vector of node $v_i^{(k)}$
$\mathbf{X}'^{(k)}$	the ‘‘context’’ embedding matrix with columns of $\mathbf{x}'_i^{(k)}$
$\mathbf{p}^{(k)}$	a probability distribution over $V^{(k)}$
P	a transport plan between $\mathbf{p}^{(1)}$ and $\mathbf{p}^{(2)}$
C_{ij}	the transporting cost between nodes $v_i^{(1)}$ and $v_j^{(2)}$
$d_i^{(k)}$	the node weight of $v_i^{(k)}$

concluding remarks and a discussion of future work are presented in section VI.

II. PROBLEM SETTING

In this paper, the setting of simultaneous link prediction problem on unaligned graphs considers two undirected, weighted graphs $G^{(1)} = (V^{(1)}, E^{(1)}, W^{(1)})$ and $G^{(2)} = (V^{(2)}, E^{(2)}, W^{(2)})$, where each graph $G^{(k)}$ ($k \in \{1, 2\}$) has the set of nodes $V^{(k)} = \{v_i^{(k)} \mid i \in \{1, \dots, n^{(k)}\}\}$ and the set of links $E^{(k)}$, and $W^{(k)} = (w_{ij}^{(k)}) \in \mathbb{R}^{n^{(k)} \times n^{(k)}}$ is the weight matrix each of whose element $w_{ij}^{(k)}$ represents the link weight between $v_i^{(k)}$ and $v_j^{(k)}$. The weight $w_{ij}^{(k)} > 0$ if $v_i^{(k)}$ and $v_j^{(k)}$ are connected, and $w_{ij}^{(k)} = 0$ otherwise.

The two graphs are assumed to share similar topological structures so that cross-graph link information can be exploited effectively; more precisely, it is assumed that some of the nodes in $V^{(1)}$ have their respective corresponding nodes in $V^{(2)}$ referring to the same entities, and vice versa. For example, $G^{(1)}$ and $G^{(2)}$ can be the Facebook and Twitter account networks that share many common users; however, the correspondences between the common users are unknown.

The goal is to find hidden (i.e., existing, but unobserved) links in each of the two graphs. The input and output of the proposed framework are summarized as follows:

- **Input:** two undirected, weighted graphs $G^{(1)} = (V^{(1)}, E^{(1)}, W^{(1)})$ and $G^{(2)} = (V^{(2)}, E^{(2)}, W^{(2)})$ that have $n^{(1)}$ nodes and $n^{(2)}$ nodes, respectively.
- **Output:** predicted hidden links $\hat{E}^{(1)} \subseteq V^{(1)} \times V^{(1)} - E^{(1)}$ and $\hat{E}^{(2)} \subseteq V^{(2)} \times V^{(2)} - E^{(2)}$.

This study focuses on the embedding approach that maps the nodes of the two graphs into a d -dimensional vector space. Therefore, the embedding vectors $\mathbf{X}^{(1)} = [\mathbf{x}_1^{(1)}, \mathbf{x}_2^{(1)}, \dots, \mathbf{x}_{n^{(1)}}^{(1)}] \in \mathbb{R}^{d \times n^{(1)}}$ and $\mathbf{X}^{(2)} = [\mathbf{x}_1^{(2)}, \mathbf{x}_2^{(2)}, \dots, \mathbf{x}_{n^{(2)}}^{(2)}] \in \mathbb{R}^{d \times n^{(2)}}$ of the two graphs

are also obtained as the byproducts, where the embedding vector of node $v_i^{(k)}$ is denoted by $\mathbf{x}_i^{(k)} \in \mathbb{R}^d$. The embedding vectors are used for link prediction based on similarity scores between the embeddings. Normally, the closer two nodes' embeddings are, the more likely a link between them exists.

For clarity, the symbols and notations used in the paper are summarized in Table I.

III. PROPOSED METHOD

The proposed framework is built upon two components: graph embedding for link prediction in each graph and optimal transport for unsupervised node alignment between the two graphs. This section starts with brief reviews of the graph embedding and the optimal transport methods employed in the framework, and then introduces an integrated objective formulation as well as its optimization algorithm.

A. Link Prediction Using Graph Embedding

In order to embed each of the two given graphs, the LINE embedding method [13] is specifically employed as it is one of the most popular and highly successful graph embedding methods.

Given an undirected and weighted graph $G = (V, E, W)$, LINE aims to preserve the first-order proximity and the second-order proximity of a graph.

The first-order proximity refers to the direct connections among the nodes given as the links and their weights. LINE defines the first-order proximity between nodes v_i and v_j as:

$$p(v_i, v_j) = \sigma(\mathbf{x}_i^\top \mathbf{x}_j) = \frac{1}{1 + \exp(-\mathbf{x}_i^\top \mathbf{x}_j)},$$

where σ is the sigmoid function. The objective function L to be minimized is defined as the KL-divergence between $p(\cdot, \cdot)$ and the empirical distribution $\hat{p}(v_i, v_j) = \frac{w_{ij}}{\sum_{(i,j)} w_{ij}}$, which results in:

$$L(\mathbf{X}) = - \sum_{(v_i, v_j) \in E} w_{ij} \log p(v_i, v_j). \quad (1)$$

The second-order proximity refers to the similarity of the neighborhoods of two nodes. To quantify the second-order proximity, LINE introduces the ‘‘context’’ embedding \mathbf{x}'_i of each node v_i in addition to the original node embedding \mathbf{x}_i . The conditional probability of the ‘‘context’’ node v_j given node v_i is defined by LINE as:

$$p(v_j | v_i) = \frac{\exp(\mathbf{x}'_j{}^\top \mathbf{x}_i)}{\sum_{k=1}^{|V|} \exp(\mathbf{x}'_k{}^\top \mathbf{x}_i)}.$$

To obtain the embedding for the second-order proximity, LINE optimizes the following objective function:

$$L(\mathbf{X}, \mathbf{X}') = - \sum_{(v_i, v_j) \in E} w_{ij} \log p(v_j | v_i). \quad (2)$$

Since it requires huge computational costs to solve the optimization problems for the two proximity variants, the authors of LINE propose to use sampling approximation and

the stochastic gradient descent. Please refer to the original paper [13] for more details about the LINE method.

Note that, although the proposed framework stands on the LINE embedding method in this paper, most of the other embedding methods can also be used as far as they focus on preserving the link structure in a graph.

B. Node Alignment Using Optimal Transport

In the proposed framework, optimal transport is used to address the problem of node alignment between two graphs. Conceptually, given two ‘‘piles of dirt’’ of a same volume, optimal transport aims to find an optimal transport plan that moves the dirt from one pile to another with the minimum transportation cost. In the machine learning context, optimal transport is a powerful and essential tool for comparing probability distributions.

Given two probability distributions $\mathbf{p}^{(1)} \in \mathbb{R}_+^{n^{(1)}}$ and $\mathbf{p}^{(2)} \in \mathbb{R}_+^{n^{(2)}}$ that satisfy $\mathbf{p}^{(1)\top} \mathbf{1}_{n^{(1)}} = \mathbf{p}^{(2)\top} \mathbf{1}_{n^{(2)}} = 1$, $P \in \mathbb{R}_+^{n^{(1)} \times n^{(2)}}$ is called a transport plan between $\mathbf{p}^{(1)}$ and $\mathbf{p}^{(2)}$ if $P \mathbf{1}_{n^{(2)}} = \mathbf{p}^{(1)}$ and $P^\top \mathbf{1}_{n^{(1)}} = \mathbf{p}^{(2)}$. Here, $\mathbf{1}_n$ indicates an n -dimensional vector of ones. For a predefined cost matrix $C \in \mathbb{R}_+^{n^{(1)} \times n^{(2)}}$, the transport cost of a transport plan P is defined as $\langle P, C \rangle = \sum_{i,j} P_{ij} C_{ij}$. A transport plan P that gives the minimum transport cost is an optimal transport. The optimal transport plan P gives a reasonable ‘‘soft’’ matching between the indices of $\mathbf{p}^{(1)}$ and $\mathbf{p}^{(2)}$. Specifically, an index i of vector $\mathbf{p}^{(1)}$ can be seen to be matched with a target index j of vector $\mathbf{p}^{(2)}$ with a probability proportional to P_{ij} .

Therefore, a ‘‘soft’’ node alignment between the nodes of two unaligned graphs can be computed via the optimal transport plan. Given the embedding vectors of two graphs, $\mathbf{X}^{(1)} = [\mathbf{x}_1^{(1)}, \mathbf{x}_2^{(1)}, \dots, \mathbf{x}_{n^{(1)}}^{(1)}] \in \mathbb{R}^{d \times n^{(1)}}$ and $\mathbf{X}^{(2)} = [\mathbf{x}_1^{(2)}, \mathbf{x}_2^{(2)}, \dots, \mathbf{x}_{n^{(2)}}^{(2)}] \in \mathbb{R}^{d \times n^{(2)}}$, we define the cost matrix C using the Euclidean distance between the embedding vectors. Specifically, we use

$$C_{ij} = \|\mathbf{x}_i^{(1)} - \mathbf{x}_j^{(2)}\|_2$$

for first-order LINE embeddings, and

$$C_{ij} = \|\mathbf{x}_i^{(1)} - \mathbf{x}_j^{(2)}\|_2 + \|\mathbf{x}'_i{}^{(1)} - \mathbf{x}'_j{}^{(2)}\|_2$$

for the second-order LINE embeddings, where $\|\cdot\|_2$ denotes the L2-norm.

The computational cost for computing optimal transport plan is often prohibitive. Cuturi [18] has proposed an efficient solution for computing an entropic-regularized approximation of the optimal transport. Instead of the exactly optimal transport $P^* = \arg \min_P \langle P, C \rangle$, they add an entropic regularization into the cost and minimize the entropic-regularized transportation cost M defined as follows:

$$M(P) = \langle P, C \rangle + \frac{1}{\lambda} \sum_{i,j} P_{ij} \log P_{ij}, \quad (3)$$

where $\lambda > 0$ is a hyperparameter and the second term is the negative entropy of matrix P . With a large enough λ , empirically when λ exceeds 50, P^* can be approximated by

the entropic-regularized solution P^λ with high accuracy. P^λ is unique and has the following form:

$$P^\lambda = \mathbf{diag}(\mathbf{u})K\mathbf{diag}(\mathbf{v}),$$

where $\mathbf{diag}(\mathbf{u})$ indicates a diagonal matrix with the elements of \mathbf{u} as the diagonal elements. The $K = e^{-\lambda C}$ is the element-wise exponential of $-\lambda C$. The vectors \mathbf{u} and \mathbf{v} are updated using

$$(\mathbf{u}, \mathbf{v}) \leftarrow \left(\frac{\mathbf{p}^{(1)}}{K\mathbf{v}}, \frac{\mathbf{p}^{(2)}}{K^\top\mathbf{u}} \right),$$

where the division is element-wise.

C. Link Prediction on Unaligned Networks

The proposed framework integrates both node embedding and node alignment processes into one single objective function as:

$$F(\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, P) = L(\mathbf{X}^{(1)}) + L(\mathbf{X}^{(2)}) + \alpha M(\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, P), \quad (4)$$

where $\alpha > 0$ is a hyperparameter to balance link prediction and node alignment.

The first two terms $L(\mathbf{X}^{(1)})$ and $L(\mathbf{X}^{(2)})$ are the loss functions for embedding the two graphs, which are chosen from the ones for the first-order LINE (1), the second-order LINE (2), or others. The third term M is the optimal transport cost (3) for node alignment. Note that the transport cost matrix C depends on the node embeddings $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ to be also optimized in the integrated formulation.

As shown in Figure 1, our optimization procedure consists of two steps: optimization of the embedding vectors $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ (and also $\mathbf{X}'^{(1)}$ and $\mathbf{X}'^{(2)}$ when the second-order LINE is used) and the optimal transport plan P . The embedding vectors are updated when fixing the optimal transport plan, and the optimal transport plan is updated when fixing the embedding vectors; these two steps are iteratively applied.

In the former step, stochastic gradient descent is used to update the embedding vectors through sampling mini-batches of links from the two graphs. The objective functions for the embedding vectors reflect the topological structure of the two graphs, while under the supervision of the optimal transport plan P , the embedding vectors of nodes $v_i^{(1)}$ and $v_j^{(2)}$ across the two graphs are also ‘‘pulled’’ closer to each other according to their ‘‘soft’’ matching probability P_{ij} .

In the latter step, the optimal transport plan P is sequentially updated based on the new transport cost matrix calculated from the updated embedding vectors. The Sinkhorn algorithm [18] is used. One should keep in mind that in order to calculate P , it is necessary to assign probability distributions $\mathbf{p}^{(1)}$ and $\mathbf{p}^{(2)}$ over the node sets $V^{(1)}$ and $V^{(2)}$. The parametric forms of $\mathbf{p}^{(1)}$ and $\mathbf{p}^{(2)}$ are specifically set as

$$p_i^{(k)} = \frac{(d_i^{(k)})^r}{\sum_j (d_j^{(k)})^r},$$

where the node weight $d_i^{(k)}$ of node $v_i^{(k)}$ is defined as the total weights of the links connected to $v_i^{(k)}$, and r is a hyperparameter. The two steps are performed iteratively together until a

Algorithm 1: Embedding learning for two unaligned graphs

Input: $G^{(1)} = (V^{(1)}, E^{(1)}, W^{(1)})$,
 $G^{(2)} = (V^{(2)}, E^{(2)}, W^{(2)})$, d
1 $n^{(1)}, n^{(2)} = |V^{(1)}|, |V^{(2)}|$
2 Initialize $\mathbf{p}^{(1)} \in \mathbb{R}_+^{n^{(1)}}$, $\mathbf{p}^{(2)} \in \mathbb{R}_+^{n^{(2)}}$
3 $P = \mathbf{0} \in \mathbb{R}^{n^{(1)} \times n^{(2)}}$
4 Initialize $\mathbf{X}^{(k)} \in \mathbb{R}^{n^{(k)} \times d}$ (and $\mathbf{X}^{(k)'} \in \mathbb{R}^{n^{(k)} \times d}$ for LINE-2nd), $k \in \{1, 2\}$
5 **for** $i = 0, 1, \dots$ **do**
6 **for** $j = 0, 1, \dots$ **do**
7 Minimize objective function F in (4) w.r.t $\mathbf{X}^{(k)}$
7 (and $\mathbf{X}^{(k)'}$) using SGD
8 **end**
9 Update P using the Sinkhorn algorithm
10 **end**
Output: $\mathbf{X}^{(1)}, \mathbf{X}^{(2)}$

stopping criterion is met. The whole optimization process is summarized in Algorithm 1.

After the final embedding vectors are obtained, they are used for making link prediction. For two unconnected nodes $v_i^{(k)}$ and $v_j^{(k)}$ in graph $G^{(k)}$, the link prediction score is defined as the cosine similarity $\frac{\langle \mathbf{x}_i^{(k)}, \mathbf{x}_j^{(k)} \rangle}{\|\mathbf{x}_i^{(k)}\|_2 \cdot \|\mathbf{x}_j^{(k)}\|_2}$.

IV. EXPERIMENTS

A. Datasets

Experiments are conducted on three real datasets to investigate the performance of the proposed method.

1) *Small-facebook*: This dataset is a small portion extracted from Facebook network dataset¹ in Stanford Large Network Dataset Collection. A sub-network of the original dataset with 532 nodes is used and denoted as the Small-facebook network for convenience. The network contains 4,812 links with an average node degree of approximately 18. Its node degree distribution is right-skewed and has a large deviation, which indicates a large diversity of the node degrees.

2) *Facebook-Twitter*: This dataset is used by Du et al. [17] and is constructed from the real-world Facebook and Twitter social networks that are collected and published in ASNETS dataset². It represents the relationship of 1,043 users on the two platforms; the Facebook network and the Twitter network have 4,734 links and 4,860 links, respectively. Both networks have average node degrees of approximately 9 and are sparser than the Small-facebook network. The two networks have fairly balanced node degree distributions with small deviations, which indicates that most of the nodes have similar node degrees.

¹<https://snap.stanford.edu/data/ego-Facebook.html>.

²<http://apex.sjtu.edu.cn/datasets/12>.

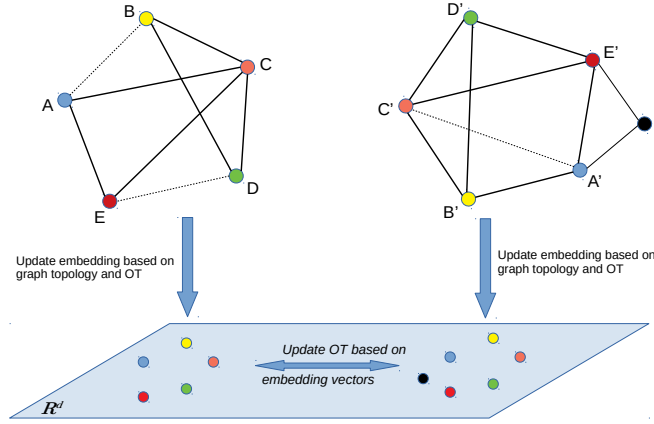


Fig. 1. The proposed method embeds two unaligned graphs into a common embedding space \mathbb{R}^d , where the embedding vectors correspond to the nodes of the original graphs. In the embedding space, optimal transport is used for seeking the most reasonable node alignment. The embedding vectors and the optimal transport plan are learned jointly in the proposed formulation.

3) *MC3 Communication network*: This dataset is extracted from a large dataset used in Mini-Challenge 3 of VAST Challenge 2018³, which contains phone call and email communication information of employees in a company. A subset of 500 employees is considered in this experiment⁴. Following Xu et al. [16], the links are filtered based on their weights by removing links with weights of 8 or smaller. This results in two networks of 6,569 links and 9,953 links. For convenience, they are denoted the Calls network and the Emails network, respectively. The Calls network has an average node degree of 26.3 and the Emails network has an average node degree of 39.8. Both networks are considerably denser than those of the previous datasets. The node degree distributions have very large deviations, which reveals that both networks have varying node degrees with several nodes of extremely big degrees.

A summary of the property of the datasets is presented in Table II.

B. Experimental Setup

Two scenarios are set up for the experiments, where the link prediction performance is investigated when changing the size of the training data as well as when reducing the number of common nodes between the networks.

In the first scenario, the observed links of each network are divided into training data, validation data, and testing data. Three configurations for the training data's size of 30%, 50%, and 70% are tested. In the validation and testing phrases, the observed links are treated as positive samples. The same number of pairs of unconnected nodes are randomly sampled and treated as negative samples. The embeddings are tested on the validation data after every outer iteration in Algorithm 1. The embeddings with the best validation AUC are used for evaluation on the testing data. The whole learning process

(model training, validation, and testing) is repeated 5 times for measuring the final average AUC score.

In the second scenario, the models' performances are observed against the number of common nodes between the two graphs. To control the percentage of common nodes, a half number of nodes are randomly selected from each original graph so that the node-overlapped percentage is in $\{100\%, 75\%, 50\%, 25\%, 0\%\}$. The new pairs of graphs G_1 and G_2 are induced as subgraphs on the selected node sets. The AUC scores are then calculated as described above with the training data's size of 50%.

In the case of the Small-facebook dataset, the Small-facebook network is treated as both G_1 and G_2 . For the Facebook-Twitter dataset, the Facebook network is treated as G_1 while the Twitter network is treated as G_2 . For the MC3 dataset, the Calls network is treated as G_1 and the Emails network is treated as G_2 .

1) *Proposed method*: LINE is employed as the base embedding method⁵. In Algorithm 1, the number of inner iteration is fixed to 100. The values of hyperparameters r and α are learned using the validation data with the grid search over $r \in \{0.75, 1, 1.5, 3, 5\}$ and $\alpha \in \{5, 10, 15, 20\}$. For each learned value of α , the proposed model is also evaluated with $r = 0.0$ (a uniform distribution). Embedding dimension d is set to 64 and the mini-batch size is set to 32. As suggested in [13], the number of the "negative" nodes K is 5, and the initial learning rate is set to 0.025. The number of the outer iterations in Algorithm 1 is set to 500 for the Facebook-Twitter dataset and to 100 for the other two datasets.

2) *Baseline methods*: For comparing the performance of the proposed model, LINE with the first-order and the second-order proximity are employed as baseline methods. Gromov-Wasserstein Learning with cosine distance (GWL-C) [16] is also compared.

³<http://vacommunity.org/VAST+Challenge+2018+MC3>.

⁴A different subset of 622 employees also is used in [16].

⁵Our implementation for the LINE model partially refers to the codes at <https://github.com/snowkylin/line>.

TABLE II
SUMMARY OF THE DATASET PROPERTIES

Dataset		#nodes	#links	Type
Small-facebook		532	4812	unweighted
Facebook-Twitter	Facebook network	1043	4734	unweighted
	Twitter network	1043	4860	unweighted
MC3	Calls network	500	6569	weighted
	Emails network	500	9953	weighted

The performance of LINE is evaluated by simply setting $\alpha = 0$ in the proposed model. For GWL-C, the implementation provided at the authors’ github⁶ is used. Since the performance of GWL-C on the link prediction task seems to improve when being trained with more epochs, the training epoch number is set to 250. The other hyperparameters are kept unchanged from the default values.

C. Results

1) Convergence of Training loss and Sinkhorn distance:

As shown in Figures 2 and 3, the total loss F of the proposed model and the Sinkhorn distance M between distributions over the node sets of the Small-facebook and Facebook-Twitter datasets gradually decrease and converge. The convergence could be foreseeable; when the model is trained, the base embedding method (LINE) is expected to capture the structure of each graph and the embedding vectors of corresponding nodes between the two graphs are expected to come closer due to the influence of optimal transport. Therefore, the embedding loss $L(\mathbf{X}^{(k)})$ and the optimal transportation cost (the Sinkhorn distance) M are projected to reduce. The loss and the Sinkhorn distance of the MC3 dataset converge quickly and behave similarly to that of Figure 2; therefore, we exclude them for the sake of presentation.

2) Link prediction with different sizes of the training data:

Tables III, IV, and V show the link prediction accuracy by different methods in terms of AUC. The proposed method with the first-order LINE (LINE-1st) and the second-order LINE (LINE-2nd) are denoted as Proposed-1st and Proposed-2nd, respectively.

In the small-facebook dataset, the proposed method shows a considerable improvement over the base LINE method. For the first-order proximity, the Proposed-1st method significantly outperforms LINE-1st even when the node distributions are set to uniform ($r = 0.0$). The AUCs further improve when the node weight information is integrated ($r > 0.0$) to enhance node alignment with optimal transport. For the second-order proximity, the improvement is milder. The AUCs of Proposed-2nd slightly drops for $r = 0.0$, but increase and improve over those of LINE-2nd when the node weight is considered. As expected, when the amount of training data increase, the performance of both LINE and the proposed model increases as well, in which the proposed model consistently improves over the LINE method. One factor contributing to the performance of the proposed method is the diversity of the node degrees of

the dataset. Important nodes with large degrees (equivalently, large node weights) are filtered out by being assigned with large probabilities in $\mathbf{p}^{(k)}$ (for $r > 0.0$), which is beneficial for optimal transport in aligning corresponding nodes. GWL-C performs worse than both LINE and the proposed method in this dataset.

In the Facebook-Twitter dataset, the proposed method gives better AUCs than LINE for the first-order proximity. For the second-order, the proposed method does not provide much benefit as the AUCs stay almost the same for all the configurations. In this dataset, the structures of both graphs might not be captured well enough by LINE, because the dataset is substantially sparser than the Small-facebook dataset and LINE is known to struggle with sparse graphs. Besides, the nodes of the Facebook and Twitter graphs have a similar node degree, which provides limited information for optimal transport in aligning corresponding nodes. These combined factors contribute to the reasons why the proposed method does not clearly improve over the LINE method. As demonstrated in Figure 3, the convergence of the Sinkhorn distance is slow and might solely be due to the fact that the embedding vectors become closer during the training rather than a reasonable matching P between the nodes has been learned. GWL-C still under-performs in this dataset.

In the MC3 dataset, the LINE-1st and Proposed-1st methods with the first-order proximity are not able to capture the structures of the networks. The learned embeddings only give the AUCs between 0.4 and 0.5, which is close to that of random embeddings. the GWL-C method performs really well and outperforms the other methods with big margins. Nevertheless, the Proposed-2nd method shows a significant improvement over the base LINE-2nd method. Its AUCs slightly reduce compared to that of LINE-2nd for $r = 0.0$, but significantly increase for $r > 0.0$. Similar to the Small-facebook dataset, the MC3 dataset has degree distributions of large deviations, which highly benefits optimal transport in aligning corresponding nodes. Strangely, when the amount of training data increases, the AUC scores of the LINE-2nd and Proposed-2nd methods do not increase but slightly decrease.

3) Link prediction with different percentages of common nodes:

As demonstrated in Figure 4, it is interesting to see that the performance does not degrade when the amount of overlapping node is small, especially even when there is no node overlapping. It can be explained that even when there is no “real” matching (common nodes) between the two graphs, if the topological structures of G_1 and G_2 are similar, it is still reasonable to consider alignment between nodes of

⁶<https://github.com/HongtengXu/gwl>.

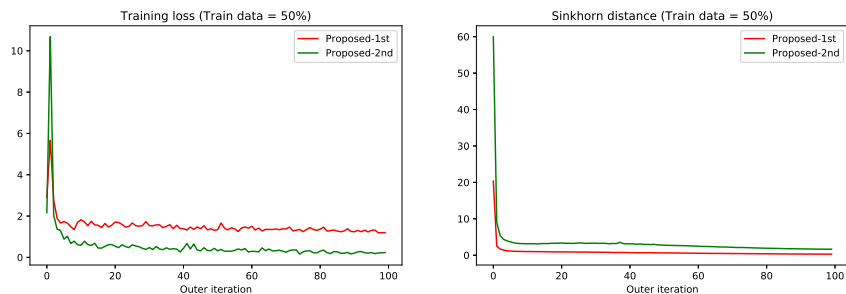


Fig. 2. The total loss and the Sinkhorn distance by the proposed method converge quickly after several outer iterations for the Small-facebook dataset. Here, the total loss is calculated when the model is trained with mini-batches, while the Sinkhorn distance is computed between the whole node sets of the graphs.

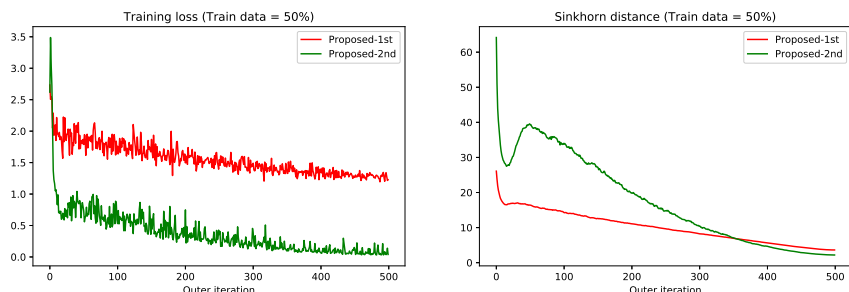


Fig. 3. For the Facebook-Twitter dataset, the convergence speed is slower. The total loss and the Sinkhorn distance slowly decrease and start to converge after around 400 outer iterations.

TABLE III
ROC-AUC SCORES OF THE SMALL-FACEBOOK DATASET

In all configurations for the size of the training data, the proposed method significantly improves the ROC-AUC scores for both orders of LINE, especially for the case of the first order. The proposed method also outperforms GWL-C in this dataset.

Train	Graphs	LINE (1st)	Proposed-1st		LINE (2nd)	Proposed-2nd		GWL-C
			$r = 0.0$	$r > 0.0$		$r = 0.0$	$r > 0.0$	
30%	G1	0.6609 ± 0.0051	0.6714 ± 0.0064	0.7052 ± 0.0102	0.6936 ± 0.0106	0.6665 ± 0.0069	0.7187 ± 0.0087	0.6892 ± 0.0009
	G2	0.6719 ± 0.0109	0.6708 ± 0.0090	0.7094 ± 0.0036	0.6994 ± 0.0068	0.6629 ± 0.0035	0.7126 ± 0.0028	0.7024 ± 0.0013
50%	G1	0.7354 ± 0.0070	0.7508 ± 0.0073	0.8033 ± 0.0119	0.7907 ± 0.0072	0.7565 ± 0.0040	0.8332 ± 0.0070	0.7194 ± 0.0019
	G2	0.7359 ± 0.0019	0.7594 ± 0.0029	0.8010 ± 0.0071	0.8018 ± 0.0036	0.7697 ± 0.0098	0.8202 ± 0.0110	0.7205 ± 0.0003
70%	G1	0.7752 ± 0.0085	0.8084 ± 0.0070	0.8584 ± 0.0085	0.8515 ± 0.0071	0.8207 ± 0.0030	0.8775 ± 0.0066	0.7608 ± 0.0020
	G2	0.7726 ± 0.0067	0.8034 ± 0.0121	0.8420 ± 0.0067	0.8518 ± 0.0041	0.8219 ± 0.0077	0.8594 ± 0.0070	0.7452 ± 0.0015

similar local neighborhood. Overall, the AUC scores remain almost the same for different values of the common node percentage. Similar to the results shown in the previous tables, the proposed method significantly improves over the LINE method and achieves the highest AUC scores for the Small-facebook dataset and the second-highest AUC scores for the MC3 dataset. However, for the second order in the Small-facebook dataset, only G_1 benefits from the proposed method while the AUC scores of G_2 remains the same as that of the LINE-2nd method. In the case of the Facebook-Twitter dataset,

consistent improvement is observed for the first order.

Throughout the three datasets, the proposed method consistently shows advantages over the base LINE method. The considerable increase in performance when the node weight is considered firmly verifies the benefit of node alignment by optimal transport in the learning process. Nevertheless, one major issue of the proposed method is that when the base embedding method fails to capture the topological structures of graphs, the proposed method also fails to perform. This is depicted in the first-order proximity in the MC3 dataset.

TABLE IV
ROC-AUC SCORES OF THE FACEBOOK-TWITTER DATASET

For the Facebook-Twitter dataset, a mild improvement of the ROC-AUC scores by the proposed model is observed for the first order. However, the ROC-AUC scores remain similar in the case of the second order. GWL-C still underperforms in this dataset.

Train	Graphs	LINE (1st)	Proposed-1st		LINE (2nd)	Proposed-2nd		GWL-C
			$r = 0.0$	$r > 0.0$		$r = 0.0$	$r > 0.0$	
30%	G1	0.5947 ± 0.0156	0.5982 ± 0.0068	0.6022 ± 0.0051	0.5665 ± 0.0035	0.5609 ± 0.0064	0.5577 ± 0.0060	0.5527 ± 0.0034
	G2	0.6111 ± 0.0080	0.6192 ± 0.0119	0.6212 ± 0.0040	0.5731 ± 0.0050	0.5741 ± 0.0056	0.5620 ± 0.0088	0.5458 ± 0.0013
50%	G1	0.6817 ± 0.0057	0.6846 ± 0.0082	0.6901 ± 0.0057	0.6397 ± 0.0043	0.6404 ± 0.0065	0.6471 ± 0.0059	0.5849 ± 0.0024
	G2	0.6709 ± 0.0022	0.6833 ± 0.0045	0.6793 ± 0.0103	0.6332 ± 0.0098	0.6347 ± 0.0096	0.6336 ± 0.0052	0.5769 ± 0.0017
70%	G1	0.7266 ± 0.0069	0.7389 ± 0.0105	0.7371 ± 0.0107	0.6937 ± 0.0081	0.6894 ± 0.0079	0.6891 ± 0.0123	0.6141 ± 0.0030
	G2	0.7219 ± 0.0056	0.7315 ± 0.0105	0.7296 ± 0.0129	0.6906 ± 0.0051	0.6878 ± 0.0115	0.6796 ± 0.0080	0.6058 ± 0.0026

TABLE V
ROC-AUC SCORES OF THE MC3 DATASET

For the MC3 dataset, GWL-C outperforms all the others including the proposed methods with large margins. However, a significant improvement of ROC-AUC scores by the proposed method is observed in the case of the second order. For the first order, LINE fails to capture the graphs' structures, which results in the failure of the proposed method.

Train	Graphs	LINE (1st)	Proposed-1st		LINE (2nd)	Proposed-2nd		GWL-C
			$r = 0.0$	$r > 0.0$		$r = 0.0$	$r > 0.0$	
30%	G1	0.4454 ± 0.0044	0.4406 ± 0.0031	0.4414 ± 0.0070	0.6348 ± 0.0054	0.6259 ± 0.0082	0.7801 ± 0.0080	0.8430 ± 0.0018
	G2	0.4411 ± 0.0038	0.4407 ± 0.0049	0.4423 ± 0.0028	0.6486 ± 0.0110	0.6140 ± 0.0096	0.7505 ± 0.0159	0.8290 ± 0.0013
50%	G1	0.4115 ± 0.0102	0.4167 ± 0.0051	0.4151 ± 0.0040	0.6387 ± 0.0070	0.6270 ± 0.0069	0.7676 ± 0.0112	0.8615 ± 0.0010
	G2	0.4253 ± 0.0059	0.4227 ± 0.0053	0.4187 ± 0.0085	0.6317 ± 0.0132	0.6087 ± 0.0052	0.7328 ± 0.0183	0.8496 ± 0.0009
70%	G1	0.3890 ± 0.0097	0.3852 ± 0.0055	0.3952 ± 0.0188	0.6188 ± 0.0095	0.6162 ± 0.0181	0.7609 ± 0.0123	0.8610 ± 0.0013
	G2	0.3937 ± 0.0091	0.3881 ± 0.0059	0.3821 ± 0.0043	0.6157 ± 0.0068	0.6039 ± 0.0093	0.7174 ± 0.0158	0.8406 ± 0.0020

Moreover, balanced graphs with nodes of similar degrees are likely to hinder the improvement of the proposed method, as demonstrated in the case of the Facebook-Twitter dataset.

4) *Node2vec as the base embedding method:* Instead of the LINE embedding method, node2vec is used as the base embedding method in the proposed model⁷. Experiments with the same settings described previously were conducted. Table VI reports a result for the MC3 dataset when the size of the training data varies. The proposed method works surprisingly well in this scenario. For all training data sizes, the proposed method with $r > 0.0$ improves the AUC scores of node2vec significantly and also outperforms GWL-C, whose AUC scores are the highest when LINE is used as the base embedding method, with big margins. The degradation of the AUC scores when $r = 0.0$ again demonstrates the importance of the node weight in the node matching process via optimal transport.

⁷Our implementation of node2vec refers to <https://github.com/aditya-grover/node2vec> and <https://github.com/Andras7/word2vec-pytorch>.

V. RELATED WORK

1) *Heuristic approaches for link prediction:* Many methods for link prediction are designed on carefully handcrafted heuristic scores. Defined on pairs of unconnected nodes, the scores are expected to be proportional to the probability that a link exists between those nodes. Heuristic methods are mostly based on heuristic observations such as two nodes who share many common neighbor nodes are likely to be connected. Some of the popular scores like Common Neighbors [1], Adamic-Ada Index [2], Jaccard Similarity Index [3], [4] are derived from the local similarity between the nodes. Other scores like Katz Index [19] and SimRank [20] are derived from the global similarity. Those methods are fast, highly parallelizable, and perform well in many settings. However, when heuristic assumptions are not met, the performances drop significantly.

2) *Graph Embedding:* Recently, graph-embedding methods are successfully applied to link prediction. They are not

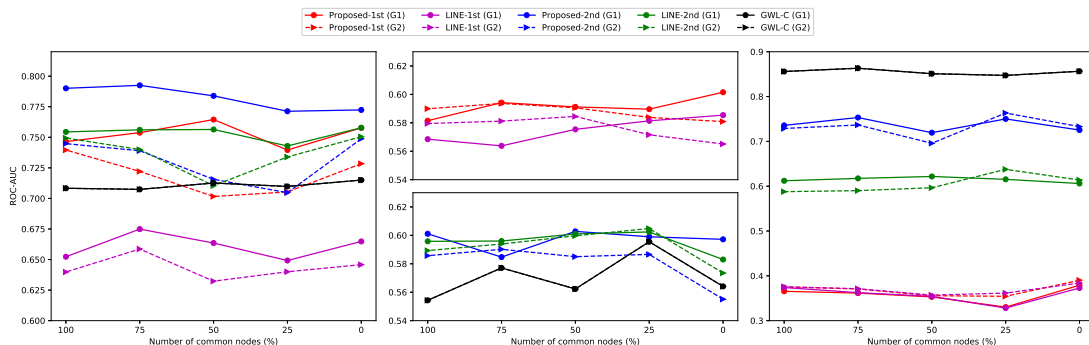


Fig. 4. ROC-AUC scores w.r.t. the percentage of common nodes between the two graphs (From left to right are of the Small-facebook, Facebook-Twitter, and MC3 datasets, respectively. The Facebook-Twitter dataset’s figure is divided into two parts for clearer visualization.) Interestingly, the performance remains similar for different amount of common nodes between the graphs.

TABLE VI
ROC-AUC SCORES OF THE MC3 DATASET WITH NODE2VEC

The proposed method with node2vec as the base embedding method works surprisingly well for the MC3 dataset. When the node weight is considered ($r > 0.0$), the proposed method outperforms both node2vec and GWL-C with big margins to achieve the highest ROC-AUC scores.

Train	Graphs	node2vec	Proposed-node2vec		GWL-C
			$r = 0.0$	$r > 0.0$	
30%	G1	0.6093 ± 0.0219	0.6186 ± 0.0130	0.9080 ± 0.0042	0.8430 ± 0.0018
	G2	0.6317 ± 0.0115	0.6196 ± 0.0062	0.9029 ± 0.0025	0.8290 ± 0.0013
50%	G1	0.6858 ± 0.0197	0.6548 ± 0.0171	0.9183 ± 0.0042	0.8615 ± 0.0010
	G2	0.7129 ± 0.0165	0.6250 ± 0.0100	0.9121 ± 0.0034	0.8496 ± 0.0009
70%	G1	0.7554 ± 0.0208	0.6534 ± 0.0071	0.9257 ± 0.0047	0.8610 ± 0.0013
	G2	0.7693 ± 0.0184	0.6403 ± 0.0200	0.9154 ± 0.0033	0.8406 ± 0.0020

designed on heuristic scores and aims to automatically learn the graph structure of data instead. Nodes are mapped into low-dimensional vector space, in which connected or closer nodes are presented with more similar vectors. Popular methods like Deepwalk [14] or node2vec [15] try to approximate the probability of visiting a node on a fixed-length random walk. If node v is close to node u in a graph, when traveling randomly along the links started from node u , node v will likely be visited with high probability. Therefore, node u and node v will be mapped to similar vectors. Another method like LINE [13] aims to approximate the “first-order” and the “second-order” node proximities. The “first-order” proximity of two nodes is proportional to the weight of their link, while the “second-order” proximity considers the resemblance between two nodes’ neighborhoods. Hence, two proteins that interact intensely with each other or two users who share many common friends will be presented with similar embedding vectors. With the ability to extract structural information efficiently in an automatic and data-driven way, the graph-embedding based approach has shown favorable results and achieved state-of-the-art performance in a variety of settings.

3) *Cross-graph approaches*: All of the above approaches only consider information in one single graph. In reality, when

two or more graphs of a similar topological structure are available (e.g: two networks describe relationships between entities of a same set), using complementary information among those graphs can improve the performance and robustness of link prediction. A few studies have been conducted following this manner. Du et al. [17] define a heuristic comparability score based on a variant of the Jaccard Similarity Index [3], [4] for node alignment. A binary classifier is then employed to predict new possible links. Both tasks are performed jointly via cross-graph embedding and have shown to benefit each other. Following this cross-graph research line, Xu et al. propose the Gromov-Wasserstein Learning framework [16] for learning node embedding and node alignment simultaneously. Aiming to minimize Gromov-Wasserstein discrepancy [21] between two graphs, the authors apply optimal transport for learning node alignment and enhancing the quality of embedding vectors. Although the framework is originally proposed to deal with node alignment only and is not intended for handling link prediction, the learned embedding vectors can be employed for the link prediction task.

4) *Link prediction on multiplex graphs*: Similar to the works on the cross-graph approaches, works on multiplex graphs (sometimes known as “multi-view” graphs) also con-

sider information from multiple graphs for the link prediction problem. A multiplex graph allows different types of links between the nodes and can be modeled as a multiple-layer graph, in which each layer is a homogeneous graph containing links of only one specific type [22]. Link prediction on multiplex graphs aims to predict potential hidden links in one layer by leveraging information from other layers in the graph [23], [24], [25]. However, the layers in a multiplex graph need to be aligned through inter-layer links, which makes the multiplex graph methods difficult to apply to the setting of multiple unaligned graphs.

VI. CONCLUSION

Although the link prediction problem has been studied extensively, existing works do not leverage the setting where multiple related graph data are available. In this paper, we follow the graph embedding approach and present a new framework that addresses the problem on two unaligned graphs simultaneously. Via optimal transport, link information in the form of vectors' closeness in the latent space is propagated between graphs, which allows the embedding processes of the two graphs to be associated and enhance each other. Experiments on three datasets in different scenarios show consistent advantages of the proposed method over the base LINE embedding method, whose embedding process is independent for each individual graph. The proposed method is flexible. Additional experiments have demonstrated the promising performance of the proposed method combined with node2vec. Technically, it can be extended for combining with different existing graph embedding methods which can be chosen according to characteristics of the graphs.

Currently, the performance of the proposed method is still modest when the degree distributions of the two graphs are balanced. To alleviate this limitation, we are looking into an extension that employs Gromov-Wasserstein discrepancy [21] as the alternating optimal transport approximation method, whose objective is to minimize the structural dissimilarity between different graphs. A solution where the probability distributions $\mathbf{p}^{(k)}$ are adaptively computed along with the optimization instead of being predefined with the hyperparameter r is also under investigation. For future works, we plan to address a more general problem setting of three and more unaligned graphs, where the multi-marginal optimal transport is necessary.

VII. ACKNOWLEDGMENT

This work was partially supported by JSPS KAKENHI Grant Number 20H04244 and the JST PRESTO program JPMJPR165A.

REFERENCES

[1] M. E. J. Newman, "Clustering and preferential attachment in growing networks," *Physical Review E*, vol. 64, no. 2, p. 025102, 2001.
 [2] L. A. Adamic and E. Adar, "Friends and neighbors on the web," *Social Networks*, vol. 25, no. 3, pp. 211–230, 2003.

[3] D. Liben-Nowell and J. M. Kleinberg, "The link prediction problem for social networks," in *Proceedings of the 22nd International Conference on Information and Knowledge Management (CIKM)*, 2003, pp. 556–559.
 [4] R. A. Baeza-Yates and B. A. Ribeiro-Neto, *Modern Information Retrieval*. ACM Press/Addison-Wesley, 1999.
 [5] Z. Lu, B. Savas, W. Tang, and I. S. Dhillon, "Supervised link prediction using multiple sources," in *Proceedings of the 10th International Conference on Data Mining (ICDM)*, 2010, pp. 923–928.
 [6] H. R. de Sa and R. B. C. Prudêncio, "Supervised link prediction in weighted networks," in *Proceedings of the 2011 International Joint Conference on Neural Networks (IJCNN)*, 2011, pp. 2281–2288.
 [7] H. Kashima, T. Kato, Y. Yamanishi, M. Sugiyama, and K. Tsuda, "Link propagation: A fast semi-supervised learning algorithm for link prediction," in *Proceedings of the 2009 SIAM International Conference on Data Mining*, 2009, pp. 1100–1111.
 [8] C. Brouard, F. d'Alché-Buc, and M. Szafranski, "Semi-supervised penalized output kernel regression for link prediction," in *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2011, pp. 593–600.
 [9] J. Kunegis and A. Lommatzsch, "Learning spectral graph transformations for link prediction," in *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, 2009, pp. 561–568.
 [10] J. Kunegis and J. Fliege, "Predicting directed links using nondiagonal matrix decompositions," in *Proceedings of the 12th International Conference on Data Mining (ICDM)*, 2012, pp. 948–953.
 [11] A. K. Menon and C. Elkan, "Link prediction via matrix factorization," in *Proceedings of the 2011 European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, 2011, pp. 437–452.
 [12] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *CoRR*, vol. abs/1310.4546, 2013.
 [13] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: large-scale information network embedding," in *Proceedings of the 24th International Conference on World Wide Web (WWW)*, 2015, pp. 1067–1077.
 [14] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: online learning of social representations," in *Proceedings of the 20th International Conference on Knowledge Discovery (KDD)*, 2014, pp. 701–710.
 [15] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd International Conference on Knowledge Discovery and Data Mining (KDD)*, 2016, pp. 855–864.
 [16] H. Xu, D. Luo, H. Zha, and L. Carin, "Gromov-Wasserstein learning for graph matching and node embedding," in *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019, pp. 6932–6941.
 [17] X. Du, J. Yan, and H. Zha, "Joint link prediction and network alignment via cross-graph embedding," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*, 2019, pp. 2251–2257.
 [18] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," in *Advances in Neural Information Processing Systems 26 (NeurIPS)*, 2013, pp. 2292–2300.
 [19] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, no. 1, pp. 39–43, 1953.
 [20] G. Jeh and J. Widom, "Simrank: a measure of structural-context similarity," in *Proceedings of the Eighth International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2002, pp. 538–543.
 [21] G. Peyré, M. Cuturi, and J. Solomon, "Gromov-Wasserstein averaging of kernel and distance matrices," in *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, 2016, pp. 2664–2672.
 [22] Y. Cui, Y. Liu, J. Hu, and H. Li, "A survey of link prediction in information networks," in *2018 IEEE International Conference on Smart Internet of Things (SmartIoT)*, 2018, pp. 29–33.
 [23] A. Hajibagheri, G. Sukthankar, and K. Lakkaraju, "A holistic approach for link prediction in multiplex networks," in *Social Informatics - Eighth International Conference (SocInfo)*, 2016, pp. 55–70.
 [24] D. Hristova, A. Noulas, C. Brown, M. Musolesi, and C. Mascolo, "A multilayer approach to multiplexity and link prediction in online geo-social networks," *EPJ Data Sci.*, vol. 5, no. 1, p. 24, 2016.
 [25] M. Jalili, Y. Orouskhani, M. Asgari, N. Alipourfard, and M. Perc, "Link prediction in multiplex online social networks," *Royal Society Open Science*, vol. 4, no. 2, p. 160863, 2017.