DeepLink: A Deep Learning Approach for User Identity Linkage

***Abstract*—The typical aim of User Identity Linkage (UIL) is to detect when users from across different social platforms are actually one and the same individual. Existing efforts to address this problem of practical relevance span from user-profile-based, through user-generated-content-based, user-behavior-based approaches to supervised or unsupervised learning frameworks, to subspace learning-based models. Most of them often require extraction of relevant features (e.g., profile, location, biography, networks, behavior, etc.) to model the user consistently across different social networks. However, these features are mainly derived based on prior knowledge and may vary for different platforms and applications. Inspired by the recent successes of deep learning in different tasks, especially in automatic feature extraction and representation, we propose a deep neural network based algorithm for UIL, called *DeepLink*. It is a novel end-to-end approach in a semi-supervised learning manner, without involving any hand-crafting features. Specifically, DeepLink samples the networks and learns to encode network nodes into vector representation to capture local and global network structures which, in turn, can be used to align anchor nodes through deep neural networks. A dual learning based paradigm is exploited to learn how to transfer knowledge and update the linkage using the policy gradient method. Experiments conducted on several public datasets show that DeepLink outperforms the state-of-the-art methods in terms of both linking precision and identity-match ranking.**

***Index Terms*—user identity linkage, social networks, deep learning, reinforcement learning**

I. Introduction

Online social networks (OSNs) such as Facebook, Twitter, and Instagram allow their respective users to generate and share various contents, and communicate with other users (individuals or public accounts) on topics of mutual interest. Such activities provide a rich data source – both contextual (i.e., text, pictures), as well as networked-related – for many valuable applications. But one example is cross-platform audience targeting in marketing and malicious (fake or duplicated) account detection in cyber-security [1]. All such applications typically involve an important step of User Identity Linkage (UIL) which usually aims to find users across different social platforms that refer to the same individual/entity [2]. UIL has an important impact in multiple applications – e.g., user behavior prediction [3], identity verification and privacy protection [4].

Variety of approaches have been proposed to tackle this practically relevant problem, and majority of them fall into two broad categories: (1) Feature-based approaches: they require to extract a set of independent features from account profiles or activities, e.g., username, gender, writing style, etc... for representing user's identity. This hand-crafting feature engineering is often based on domain knowledge and deep understanding of user activities. For example, Goga et. al [4] combine several characteristics extracted from user's posts, e.g., geo-location, timestamps and language, to profile their identities. Zafarani et. al [5] apply theories from sociology and psychology to model user behavioral patterns to map identities across OSNs. (2) Network-based methods: much attention has been recently paid to make use of user network structural information for correlating accounts across social platforms. For example, COSNET [6] considers the local and global consistency among multiple networks and Adamic/Adar scores are computed to measure neighborhood similarities. IONE [7] learns network embedding from the follower/followee-ship couplings in order to preserve the proximity of users with "similar" relationships. We note that several other works (e.g., [8]–[10]) combine profile features and network information to improve account alignment across OSNs.

Challenges and Our Approach: The drawbacks of the existing approaches addressing the UIL problem can be categorized as follows: (1) They do not provide a comprehensive framework to address the heterogeneity of users and OSNs. Namely, the different social network sites are independent and activities on one site can be very different from other(s). Additionally, users often act differently in multiple sites – thus, mapping the behaviors of cross-site accounts to a particular user is complex. (2) User representation: capturing latent semantic relationships among users based on network structures is difficult, especially if one seeks a meaningful and generalizable model for different networks and applications. Most of the existing works that focus on cross-platform behavior prediction or account correlation are using transfer learning, which assumes that multiple networks are either fully-overlapped or non-overlapped. In reality, however, there may be a very high degree of partial overlaps. (3) Lack of labeled data: Obtaining a set of users with the same identity across platforms is not easy. There is no central repository that would provide an explicit information to this effect. From a complementary perspective – manual labeling is time-consuming and sometimes infeasible.

To address above challenges, in this paper we propose a novel deep learning based approach, called DeepLink which, in principle, operates along the following four components: (1) It samples networks to produce the training "corpus" while preserving network structures at maximum; (2) Each node in the network is represented as a vector in a low-dimensional space via network embedding; (3) The anchor nodes are then fed into a deep neural network to train a non-linear transformation for user alignment across networks;

(4) DeepLink uses a dual-learning process to improve the identity linking performance and boost the supervised training algorithm.

More specifically, we first pretrain two preliminary mapping functions between a given pair of networks A and B, i.e., $\Phi(A \rightarrow B)$ and $\Phi^{-1}(B \rightarrow A)$ using automapping on anchor nodes without their labels (their alignment) . Then the user identity linkage can be formalized as a dual learning game: the embedded representation (vector) of an anchor node $\vec{a}$ in network A can be mapped into a vector $\vec{b'}$ denoting its correspondent in network B using $\Phi(A \rightarrow B)$. The vector $\vec{b'}$ is then compared to its true embedding vector in the network B $\vec{b}$, and A is notified whether $\Phi(A \rightarrow B)$ is a high-quality mapping, followed by mapping $\vec{b'}$ back to A (obtaining $\vec{a'}$) according to $\Phi^{-1}(B \rightarrow A)$. After receiving the vector $\vec{a'}$, network A measures the similarity between $\vec{a}$ and $\vec{a'}$ and feedbacks the result to B the result. The game can also start from a node in B with vector embedding in $B's$ space. Essentially, the two mapping functions get increasingly improved along with more anchor nodes.

The main advantages of DeepLink and contributions of our work are:

- DeepLink takes advantage of deep neural networks to learn latent semantics of both user activities and network structure in an end-to-end manner, thereby avoiding the labor-intensive feature engineering, which is also easy to be generalized to various OSNs.
- DeepLink leverages a semi-supervised graph regularization to predict the context (neighboring structures) of nodes in the network. It encodes structural information of anchor nodes to align networks.
- DeepLink uses a dual-learning process to improve the identity linking performance and boost the supervised training algorithm. In this way, not only unlabeled anchor nodes are being exploited, but also the mapping among multiple networks can get improved through the reinforcement learning procedure. Moreover, the number of aligned (labeled) anchor nodes required can be significantly reduced.
- To demonstrate the effectiveness of DeepLink, we conducted experiments on both real-world and synthetic datasets. The results show that DeepLink can significantly improve the identity accuracy compared to the state-of-the-art approaches, e.g., up to 30% for *top-1* and 40% for *top-5* in terms of linking precision.

In the rest of this paper, Section II reviews the related work, followed by Section III which presents the preliminary background and formalizes the problem statement. In Section IV, we present the main methodology including aligning OSNs and training details. Experimental evaluations quantifying the benefits of our approach are presented in Section V, and Section VI concludes the paper and outlines directions for future work.

II. Related Work

Several variants of the User Identity Linkage (UIL) [11] problem have been studied in the literature, also known as Link Inference/Prediction [12], [13], Correlating Accounts [4], Node Alignment [10], [14], etc. The existing research efforts can be broadly classified into two main categories: profile-based and network-based identity linkage.

Profile-based methods: leverage user's profile information (e.g., username [5], spatio-temporal patterns [15], posts [4], writing style [16], etc.) to link accounts across different sites. For instance, Reza et al. find correlating accounts using user names by modeling the naming process and rules from the perspective of information redundancy [5]. However, user names can be deliberately selected and modified at any time, which increases the difficulty of completing the UIL task. Writing style identification is another promising way to localize multi-account users while revealing various camouflage behaviors. Arvind et al. used writing style of user generated text, e.g., grammatical structure and frequency of letters to identify users [16]. However, this method results in over-fitting, especially for short texts such as tweets, because it involves too many features. Recent research results investigated the security issues of multi-accounts. For instance, Jiang et al. propose a semi-supervised transfer learning method to predict cross-platform behaviors through a sparse overlapped crowds [3]. Qian et al. protect against de-anonymization attack by discovering the approach for inferring privacy using knowledge graphs [1]. Luo et al. presents a uni-class classification-based approach to detect multi-account users across OSN sites [17]. Riederer et al. use the timestamped location data generated by users to infer the user identities across the OSNs [15]. Zhang et al. presents an energy-based model to link user identities by extracting distance-based profile features [6].

Almost all previous methods focus on either writing-style analysis or user behavior inference – however, in addition to the risk(s) of privacy leakage [1], they have a drawback in terms coping with potential inconsistencies [2].

Network-based methods: are becoming increasingly promising in tackling the UIL problem and have received much attention, because they only require structural information to align networks based on anchor nodes. For instance, BIG-ALIGN [18] introduces the problem of aligning bipartite graphs and proposes a gradient-descent based solution. In [19], Tan et al. model user relationship using a hyper-graph and project the manifolds of two OSNs onto a common embedded space to correlate accounts. Neighborhood-based features seem like a natural choices for the UIL problem [6], [9], [20], relying on computing the Adamic/Adar scores to measure the neighborhood similarity [21]. CLF [22] predicts both anchor nodes and social links by transferring information related to social links formed by anchor nodes in the source network to the target network.

Inspired by word embedding techniques (e.g., word2vec [23] and Glove [24]) in natural language processing, a number of approaches have been proposed

to embed the graph, e.g., DeepWalk [25], Line [26], SDNE [27], SDAE [28], node2vec [29], MM-DeepWalk [30], M-NMF [31], TransNet [32], CANE [33] – to name a few. Recently, some researchers have exploited Convolutional Neural Networks (CNN) and spectral graph theory to learn the representation of arbitrary networks, such as Planetoid [34], GraphCNN [35], Patchy-SAN [36], GCN [37], etc.

In the context of UIL task, Man et al. [8] used network embedding techniques to capture latent structural regularities of observed anchor links and further learn a cross-network mapping for predicting anchor links. Liu et al. proposed IONE which also embeds two OSNs onto a common space to capture the social contacts of users [7]. PCT [38] aims at inferring potential corresponding connections linking multiple kinds of shared entities across networks simultaneously through combination of both profile and network features.

Existing network-based methods embed structures of nodes from their local context by preserving the first and/or second-order proximity to link accounts across OSNs. The local structure of a network contains rich information for a group of nodes, but it is hard for existing UIL algorithms to discriminate the real user identity from its neighborhoods. Previous works only leverage the partial anchor nodes for supervised training, including embedding and network alignment while using the rest for testing which incurs insufficient training and inefficient linking. Moreover, many UIL algorithms including IONE use anchor nodes to embed and align non-anchor nodes [7]. However, their anchor nodes may become deviated (not aligned any more) after training.

Compared to these work, our approach is capable of embedding the global network structure and achieves higher correlation accuracy. In addition, DeepLink learns more sophisticated non-linear mapping among multiple networks, when compared to the traditional embedding-based methods. To make full use of unlabeled anchor nodes, we borrow an idea of dual-learning mechanism, a policy gradient-based method to boost the training while updating the linkage.

III. Preliminary Background

In this section, we introduce the basic terminology in the setting of our proposed approach, and present a few formal definition in that context.

A. Problem Definition

We consider a set of s different social networks as $\mathcal{G}_1, ..., \mathcal{G}_s$, each of which is defined as a Social Network Graph (SNG).

Definition 1. ***Social Network Graph*** *Let $\mathcal{G} = (V, E)$ be an unweighted and undirected graph, where V is the set of vertices, each representing a user, and E is the set of edges connecting users. $e_{i,j} \in E$ indicates the relationship between user u_i and user u_j.*

We represent each SNG with an unique latent user space according to the Network Embedding Model (NEM). NEM learns the probabilistic distributions of nodes and uses low-dimensional vectors to represent them in a latent space. Formally:

Definition 2. ***Network Embedding Model*** *Given a set of $u_1, u_2, ..., u_m$ users in $\mathcal{G}$, NEM learns to represent each u_i with a vector $\mathbf{v}(u_i) \in \mathbb{R}^d$, where d is the dimensionality of the latent space.*

After embedding each of the SNGs $\mathcal{G}_1, ..., \mathcal{G}_s$, we learn the prediction functions among them as the task of User Identity Linkage (UIL), similar to the definition in [11].

Definition 3. ***User Identity Linkage*** *Given any two social networks $\mathcal{G}^s$ and $\mathcal{G}^t$, the goal of UIL is to predict that any pair of user identities u^s and u^t chosen from $\mathcal{U}^s$ and $\mathcal{U}^t$ respectively belong to the same real natural person (i.e., $u^s = u^t$). That is, UIL learns a binary function $\Phi_U : \mathcal{U}^s \times \mathcal{U}^t \to 0, 1$ such that*

$$\Phi_U(\mathcal{U}^s, \mathcal{U}^t) = \begin{cases} 1, & u^s = u^t, \\ 0, & otherwise. \end{cases}$$

where $\Phi_U(\mathcal{U}^s, \mathcal{U}^t) = 1$ means a successful linkage (a good identity matching).

However, perfect prediction function Φ_U is hard to obtain, as the latent space of each $\mathcal{G}$ is unknown to the others. In real implementations, most UIL algorithms try to find an approximate graph mapping (projection) function Φ [9].

Definition 4. ***Graph Mapping Function*** *The function Φ is defined as a mapping from $\mathcal{G}^s$ to $\mathcal{G}^t$, such that for each $u_i \in \mathcal{G}^s$ and its latent space vector $\mathbf{v}(u_i)$, we have $\Phi(\mathbf{v}(u_i)) = \mathbf{v}(u'_i), u'_i \in \mathcal{G}^t$. We also denote the inverse mapping as $\Phi^{-1}(\mathbf{v}(u_j)) = \mathbf{v}(u'_j)$, where $u_j \in \mathcal{G}^t$ and $u'_j \in \mathcal{G}^s$.*

Generally, the mapping function Φ is unknown for a given $\mathcal{G}$ – and the objective of our work is to learn a bilateral mapping (Φ and Φ^{-1}) such that the two SNGs $\mathcal{G}^s$ and $\mathcal{G}^t$ are aligned by maximizing the similarity of all aligned pairs $(\mathbf{v}(u_i), \mathbf{v}(u_j))$. Note that fully aligned networks hardly exist in the real world, hence, we leverage the partially aligned anchor nodes (labeled data) to map the user latent spaces of two SNGs.

IV. DeepLink: The Proposed Model

As shown in Figure 1, DeepLink consists of four main components: *network sampling*, *users latent space embedding*, *unsupervised automapping* and *supervised linkage dual learning* – which we discuss in detail in the sequel.

A. Network Structure Sampling

To embed users into a latent space, we first generate multiple social sequences for each user $u_i \in V$ in several rounds of random walks. Each sequence encodes the social relationships among users in the SNG. All the sequences combined across the different SNGs are called "corpus" and are used to learn the embedding vectors for users.

Sampling process works as follows: it starts at vertex (user) u_i and proceeds along a randomly selected edge at each step, until length L is reached. This can implicitly extract some hidden structural social information, e.g., friendship and community in a network. As discussed in [25], random walks not only capture basic network information, but can also scale

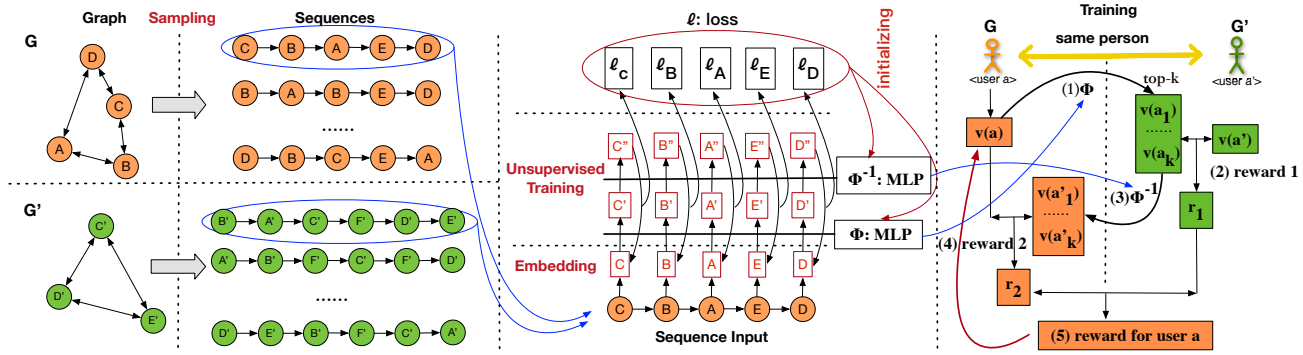


Fig. 1. Overview of DeepLink: (1) first, it uses the random walk to generate samples of node sequences to form corpus. (2) then, in the initialization phase, it embeds and represents each node in a low-dimensional latent space and pretrains two mapping functions from $G \rightarrow G'$ using anchor nodes in G (without knowing their aligned correspondents in G'), similarly for $G' \rightarrow G$ – to improve Φ and Φ^{-1}. (3) finally, it learns the node alignment across networks based on pairs of aligned anchor nodes using a policy gradient-based method in a supervised manner (with known alignments).

in the sense of accommodating minor changes in a graph with no need to re-compute everything. From a complementary perspective, generating social sequences is time consuming for a large network, but can be implemented in parallel by assigning several simultaneous walkers.

There are several recent works on learning structural features of connected graph by leveraging various sequence generating algorithms, such as LINE [26], GraRep [39] and node2vec [29]. node2vec samples network neighborhoods of each node using the *biased random walks*. This can capture the community information of that node. In this work, we still use traditional random walks to generate social sequences due to its computational efficiency compared to the exhaustive search used in works such as node2vec [29].

B. User Latent Space Embedding

After generating the social sequence $\mathcal{S}^r_{u_i}$ for user u_i in the r^{th} round, DeepLink uses Skip-gram model to update its social representation. Skip-gram model is originally used to predict the context of a word [23] by maximizing the average log probability in the domain of word representation. Formally, given a sequence of users $u_1, u_2, ..., u_m \in \mathcal{G}$, DeepLink aims to maximize the log probability using the following equation:

$$\frac{1}{m}\sum_{t=1}^{m}\sum_{j=-w}^{w}\log p(u_{t+j} \mid u_t), j \neq 0 \tag{1}$$

where w is the size of the sliding window and indicates that increasing the training context w may lead to a higher accuracy at the cost of training time. A softmax function is used to define the conditional probability $p(u_{t+j} \mid u_t)$ as the occurrence of the *j-hop* neighbor u_{t+j} given user u_t:

$$p(u_{t+j} \mid u_t) = \frac{\exp(\mathbf{v}^{\mathrm{T}}_{u_{t+j}}\mathbf{v}'_{u_t})}{\sum_{i=1}^{m}\exp(\mathbf{v}^{\mathrm{T}}_{u_i}\mathbf{v}'_{u_t})} \tag{2}$$

where $\mathbf{v}_{u_i}$ and $\mathbf{v}'_{u_i}$ are, respectively, the input and output vector representations of user u_i, and m is the number of users in a SNG.

To improve the training efficiency, Negative Sampling [40] is adopted. Specifically, we maximize the objective function as [8]

$$\log[\sigma(\mathbf{v}^{\mathrm{T}}_{u_{t+j}}\mathbf{v}'_{u_t})] + \sum_{i=1}^{K}\mathbb{E}_{u_i\sim p_n(u)}[\log(1-\sigma(\mathbf{v}^{\mathrm{T}}_{u_i}\mathbf{v}'_{u_t}))]. \tag{3}$$

where there are K negative samples. Empirically, each node is sampled with probability $p_n(u) \sim d_{u_i}^{3/4}$, where d_{u_i} is the degree of node u_i [23].

By maximizing Eq. 3 over all the nodes in SNGs independently, we can approximate the objective function Eq. 1 to train the representation vector for each u_i with stochastic gradient descent algorithm.

C. Neural Mapping Learning

After obtaining the latent embedding space for each SNG, DeepLink turns to learn the mapping functions between any two SNGs based on the anchor nodes by using two Multi-Layer Perceptrons (MLP). Given each labeled anchor node pair (u_i, u_j) and their representation vectors $(\mathbf{v}(u_i), \mathbf{v}(u_j))$, DeepLink learns a mapping $\Phi(\mathbf{v}(u_i)$ by minimizing the loss function below:

$$\ell(\mathbf{v}(u_i), \mathbf{v}(u_j)) = \min(1 - \cos(\Phi(\mathbf{v}(u_i)), \mathbf{v}(u_j))) \tag{4}$$

where $\cos(\cdot)$ is the cosine similarity between mapped vector $\Phi(\mathbf{v}(u_i))$ from $\mathcal{G}_s$ and the embedding representation $\mathbf{v}(u_j)$ in $\mathcal{G}_t$. The loss of $\ell(\cdot)$ ranges from 0 meaning exactly the same to 2 meaning exactly opposite.

Suppose we have n anchor nodes, their embedding vector matrices are respectively denoted by $\mathbf{A} \in \mathbb{R}^{d\times n}$ and $\mathbf{B} \in \mathbb{R}^{d\times n}$, where d is the embedding dimensionality. The mapping learning minimizes the following objective function

$$\ell(\mathbf{A}, \mathbf{B}) = \arg\min_{\mathbf{W},\mathbf{b}}(1 - \cos(\Phi(\mathbf{A}), \mathbf{B}; \mathbf{W}, \mathbf{b})) \tag{5}$$

where $\mathbf{W}$ and $\mathbf{b}$ are parameter and bias to learn. In a real implementation, we use a batch size of h ($0 < h \ll n$) vectors to feed MLP at each step until the n anchor nodes have been visited, which would be repeated R times – until convergence.

D. Linkage Dual Learning

The MLP based anchor mapping from $\mathcal{G}_s$ to $\mathcal{G}_t$ introduced above can be straightforwardly applied so that we can learn a reverse mapping Φ^{-1} from $\mathcal{G}_t$ to $\mathcal{G}_s$ with the same process. However, this automapping process does not take into account anchor nodes that are used for training. In other words, assuming that we have n anchor nodes, when $TR\%$ used for training then $TE\%$(=1$-TR\%$) are used for evaluation/testing. To fully exploit these anchor nodes, we employ a variation of dual-learning [41], originally used in natural language translation, for the task of user identity linkage.

Suppose we have two weak mapping functions Φ and Φ^{-1} – e.g., pretrained with partial anchor nodes – that can project vectors from $\mathcal{G}^s$ to $\mathcal{G}^t$ and vice versa. We then improve two mapping functions by leveraging the duality of Φ and Φ^{-1}. The two steps achieving this goal are described next.

1) Unsupervised UIL Pretraining: Most of existing work (e.g., [7], [8]) employ the labeled anchor nodes – i.e., the alignment for a pair of nodes is known – for aligning two networks. However, the *unlabeled* anchor nodes, normally used for testing, are also informative in network alignment. Specifically, for each anchor node (labeled and unlabeled) u_b in $\mathcal{G}^s$, we first obtain $\mathbf{v}'(u_b)$ via the Φ mapping: $\mathbf{v}'(u_b) = \Phi(\mathbf{v}(u_b))$ and then map back from $\mathcal{G}^t$ via $\Phi^{-1}(\mathbf{v}'(u_b))$ to get a vector $\mathbf{v}''(u_b)$. Note that no labels are required in this unsupervised learning process, therefore there is no difference between labeled and unlabeled anchor nodes. The loss of this automapping (similar to autoencoder in [42]) is calculated based on the difference between $\mathbf{v}(u_b)$ and $\mathbf{v}''(u_b)$. In addition, the anchor nodes in one network are blinded to with respect to the other. We use the same method to pretrain a $\mathcal{G}^t \rightarrow \mathcal{G}^s$ model. After this unsupervised pretraining, we have two weak mapping functions Φ and Φ^{-1} that will be further improved in next step – described below.

2) Supervised UIL Learning: The *labeled* anchor nodes are used to improve the mapping function Φ and Φ^{-1} by playing a dual learning game. Specifically, we set h batches for n anchor nodes where each batch has $\lfloor n/h \rfloor$ labeled nodes. Each batch forms an *episode*, where one anchor node u_a denotes a *state* s_a. Note that the state transition in this case is deterministic, i.e., the current state has probability of 1 transferring to the next state (anchor node) and the *action* is defined as selecting an anchor node. We use $\mathbf{v}(u_a)$ and $\mathbf{v}'(u_a)$ as the vectors respectively representing u_a in networks $\mathcal{G}^s$ and $\mathcal{G}^t$. The basic idea of our dual identity linkage learning is that given a batch of anchor nodes, two mapping functions try to align two user latent spaces according to the rewards of mapping anchor nodes in the batch. Additionally, it provides an explicit convergence and termination conditions, in contrast to those completely empirically chosen in existing works such as IONE [7] and ULink [9].

More formally, for a game starting with u_a in $\mathcal{G}^s$ (agent A), we use Φ to map its vector on the space of $\mathcal{G}^t$ and search its k nearest vectors $S(\mathbf{v}'(u_a)) = \text{Top}(\Phi(\mathbf{v}(u_a)))$, indicating the most similar k embedding vectors of anchor nodes in $\mathcal{G}^t$. Here, k vectors are the candidates for real users and the probability of successful linkage gets higher with training on more anchor nodes. The agent B (in $\mathcal{G}^t$) then computes a *reward* $r^i_{s,t}$ as:

$$r^a_{s,t} = \frac{1}{k}\sum_{i=1}^{k} \log(\cos(\mathbf{v}(u_i), \mathbf{v}'(u_a)) + 1) \tag{6}$$

where $\cos(\cdot) + 1$ denotes the similarity value between two vectors and ranges from 0 to 2. We note that the reason for searching and averaging *top-k* vectors for $\Phi(\mathbf{v}(u_a))$ is that it is difficult for the mapping function to exactly match the real identity of $u_a \in \mathcal{G}^t$ (vector $\mathbf{v}'(u_a)$) – however, it has a larger probability to include the real identity in the *top-k* nearest ones.

Intuitively, we can also calculate the mapping Φ^{-1} of $\mathbf{v}'(u_i)$ back to $\mathcal{G}^s$ and leverage the duality of the mapping to produce the second reward $r^a_{t,s}$, i.e., the average similarity between $\Phi^{-1}(\mathbf{v}'(u_i))$ and $\mathbf{v}(u_a)$, as:

$$r^a_{t,s} = \frac{1}{k}\sum_{i=1}^{k} \log(\cos(\Phi^{-1}(\mathbf{v}'(u_i)), \mathbf{v}(u_a)) + 1) \tag{7}$$

Thus, the action-value r^a of selecting user (state) u_a is a linear combination of $r^a_{s,t}$ and $r^a_{t,s}$ which indicates the estimated probability of correct real identity linkage by the mapping functions. In particular, it exploits the duality of two mapping functions to guide the anchor nodes training process. Now the expected reward $\mathbb{E}[r_h]$ for the h^{th} batch is:

$$\mathbb{E}[r_h] = \sum_{a=1}^{\lfloor n/h \rfloor} (\alpha r^a_{s,t} + (1-\alpha) r^a_{t,s}) \tag{8}$$

Algorithm 1 User Identity Linkage Dual Learning.

Input: anchor nodes $\mathbf{A}$ and $\mathbf{B}$ ($|\mathbf{A}| = |\mathbf{B}| = n$), batch size h, two initialized pretrained weak functions: Φ and Φ^{-1}, discount rates $\gamma^h_{s,t}$ and $\gamma^h_{t,s}$.
Output: $\Phi_{\mathbf{W},\mathbf{b}}$ and $\Phi^{-1}_{\mathbf{W},\mathbf{b}}$, hyper-parameter α.

repeat
  h = 1, 2, ...
  for anchor vector $\mathbf{v}(u_a)$ in batch h **do**
    for each $u_a \in \mathcal{G}^s$ **do**
      state $s_a = u_a$;
      Search k most similar vectors $S(\mathbf{v}'(u_k))$ in $\mathcal{G}^t$:
      $S(\mathbf{v}'(u_k)) = \text{Top}(\Phi(\mathbf{v}(u_a)))$.
      Compute reward $r^a_{s,t}$ according to Eq.6.
      Compute reward $r^a_{t,s}$ according to Eq.7.
      Compute stochastic gradient of $\triangledown\Phi_{\mathbf{W},\mathbf{b}}\mathbb{E}[r_{s,t}]$ as:
      $\frac{1}{k}\sum_{i=1}^{k}[\alpha\triangledown\Phi_{\mathbf{W},\mathbf{b}} r^i_{s,t}; \mathbf{W}, \mathbf{b})]$;
      Update $\Phi_{\mathbf{W},\mathbf{b}} \leftarrow \Phi_{\mathbf{W},\mathbf{b}} + \gamma^h_{s,t}\triangledown\Phi_{\mathbf{W},\mathbf{b}}$.
      Compute stochastic gradient of $\triangledown\Phi_{\mathbf{W},\mathbf{b}}\mathbb{E}[r_{t,s}]$ as:
      $\frac{1}{k}\sum_{j=1}^{k}[(1-\alpha)\triangledown\Phi_{\mathbf{W},\mathbf{b}} r^j_{t,s}; \mathbf{W}, \mathbf{b})]$;
      Update $\Phi_{\mathbf{W},\mathbf{b}} \leftarrow \Phi_{\mathbf{W},\mathbf{b}} + \gamma^h_{t,s}\triangledown\Phi_{\mathbf{W},\mathbf{b}}$.
    end for
  end for
until convergence;

where hyper-parameter α is learned and tuned from the training, similarly to [41].

The process of supervised user identity linkage learning is summarized in Algorithm 1. Since the reward of the game can be considered as a function of $\mathbf{v}'(u_a)$, $\mathbf{v}(u_a)$, Φ and Φ^{-1}, we can optimize the parameter $\mathbf{W}$ and $\mathbf{b}$ in two mapping functions based on maximizing the expected reward (line 12 and line 15 in Algorithm 1 where $\gamma^h_{s,t}$ and $\gamma^h_{t,s}$ are discount rates using policy gradient methods [43]).

We also use the same method to train DeepLink from the other direction $\mathcal{G}^s \rightarrow \mathcal{G}^t$ in order to alleviate overfitting. Empirically, we found that averaging the linkage results is informative to help aligning two networks efficiently.

E. Discussion

As discussed so far, our proposed algorithm DeepLink links users across two networks. However, in many real applications, it is often the case that more than two platforms are involved for user identity linkage [9]. We note that this can also be addressed by DeepLink in two ways:
(1) A straightforward "chain rule", i.e., sequentially aligning several networks such as $\mathcal{G}^1 \rightarrow \mathcal{G}^2 \rightarrow \mathcal{G}^3$....
(2) Training different versions of DeepLink for every pair of networks. For example, suppose we have three networks $\mathcal{G}^1$, $\mathcal{G}^2$ and $\mathcal{G}^3$. The dual mappings between all possible pairs, i.e., $(\mathcal{G}^1, \mathcal{G}^2)$, $(\mathcal{G}^1, \mathcal{G}^3)$ and $(\mathcal{G}^2, \mathcal{G}^3)$, can be trained in parallel. Then the user identity linkage across three networks can be easily obtained via transitivity.

DeepLink is a network-based UIL approach, but it can easily incorporate features from users to improve the linkage efficiency, such as usernames and/or locations, if available. These features can be embedded into the vector obtained via network embedding. We note that the existing network structure based embedding may suffer from *top-k mess*, meaning that embedded vectors for neighboring nodes are too close to distinguish from each other. The features irrelevant to network structures might be helpful to discriminate the *top-k* candidates. For instance, we can use an *edit distance* of usernames as a constraint to screen out the *top-1* result from the most similar k vectors.

V. Experiments

We now describe several real-world and synthetic datasets used in our experiments, and then present in detail the experimental observations regarding the advantages of DeepLink from two aspects: linking precision and ranking performance. We also discuss the effects of different parameters' combinations on model performance[1].

A. Datasets

To compare the performance of all the different methods we used the following ground truth social network collections in our experiments (Table I describes the corresponding graphs):

- **Foursquare - Twitter**: This dataset is provided by Zhang et al. [38], where nodes (users) of two social networks (Foursquare and Twitter) are partially aligned.
- **Lastfm-MySpace and Livejournal-MySpace**: This dataset is published by Zhang et al. [6] and available online[2]. It contains 5 networks, however, for the sake of privacy, they only provide partial anchor nodes for true identity linkage. We randomly choose two network pairs for our evaluation, including Lastfm-MySpace and Livejournal-MySpace.

TABLE I
Statistics of datasets.

Dataset	$\lvert V \rvert$	$\lvert E \rvert$	anchor nodes
Foursquare	5,120	76,972	3,148
Twitter	5,313	164,920	
Lastfm	2,659	102,506	1,561
MySpace	2,161	90,438	
Livejournal	1,366	22,299	667
MySpace	867	18,163	

B. Baselines and Settings

For DeepLink, we use a 3-layer perceptron as the mapping function. Each perceptron has 2 hidden layers which produce a non-linear transformation by projecting the input (e.g., $800d$) to intermediate vectors: $1200d$ (first hidden layer) and $1600d$ (second hidden layer) before reducing to its original dimensionality ($800d$). The learning rate for training is 0.001, and the batch size is set to 32.

We compare our DeepLink approach with several network-based methods which also require only network structural information. We note that there are many methods using both profile-based feature and network information, which is also applicable in DeepLink as we discussed in section IV-E. In this paper, we concentrate on the network-based UIL, and evaluate the performance of DeepLink by comparing it with the following baselines:

- **IONE**: Input-Output Network Embedding (IONE) is a network embedding and partial network alignment method proposed in [7] achieving the state-of-the-art linking results compared to previous works. In IONE, the followers and followees of users are represented with three vectors: node vector, input vector and output vector. The user latent space is obtained with negative sampling and constraints on common users of the networks, where gradient descent is used to train and align two networks with the anchor nodes.
- **ONE**: ONE is simplified version of IONE where only node vector and output vector representation of a user are considered for alignment.
- **MAH**: Manifold Alignment on Hypergraph (MAH) is the network embedding method proposed in [19] which represents nodes into a common low dimensional space (dimensionality is 400 in our repeated implementation

[1]We respectfully note that supplemental material, implementation details and instructions are available online, but the URL is removed from this submission version due to the double-blind review process.

[2]http://aminer.org/cosnet

which is the best result reported in [19]) and infers account correlation by comparing distances between two vectors across networks in the embedding space. MAH uses hypergraph to model high-order relations, e.g., same interest group or same activity. For a target user in one network, MAH ranks all users in the other network by their probability of referring to the same user.

- **MAG**: Manifold Alignment on traditional Graphs (MAG) [19] builds a social graph for each network by computing user-to-user pairwise weights as $w(u_i, u_j) = \mid R_{u_i} \cap R_{u_j} \mid / (\mid R_{u_i} \mid + \mid R_{u_j} \mid)$ where R_{u_i} is relation set containing user u_i. The ranking result of users is obtained via the manifold alignment.
- **CRW**: Collective Random Walk (CRW) [22] predicts the formation of social links among users in the target network as well as anchor links aligning the target network with other external social networks. CRW consists of two phases: (1) collective link prediction of anchor and social links; and (2) propagation of predicted links across the partially aligned probabilistic networks with collective random walk.

C. Evaluation Metrics

In a similar spirit to [2], we choose three standard metrics to evaluate the performance, including *Precision@k(P@k)*, *MAP*, *AUC* and *Hit-Precision*. Note that the higher the values of each of these measures, the better the performance.

Precision@k is the metric for evaluating the linking accuracy, defined as:

$$P@k = \sum_{i}^{n} \mathbb{1}_i\{success@k\}/n \tag{9}$$

where $\mathbb{1}_i\{success@k\}$ measures whether the positive matching identity exists in *top-k* ($k <= n$) list, and n is the number of testing anchor nodes. Note that since *top-k* is a metric of the true positive prediction rate, *Precision@k* is exactly the same as *Recall@k*, as well as the $F_1@k$, in the context of UIL.

MAP, *AUC* and *Hit-Precision* are used for evaluating the ranking performance of the algorithms, defined as:

$$\begin{aligned} MAP &= (\sum^{n} \frac{1}{ra})/n \\ AUC &= (\sum^{n} \frac{m+1-ra}{m})/n \\ \textit{Hit-Precision} &= (\sum^{n} \frac{m+2-ra}{m+1})/n \end{aligned} \tag{10}$$

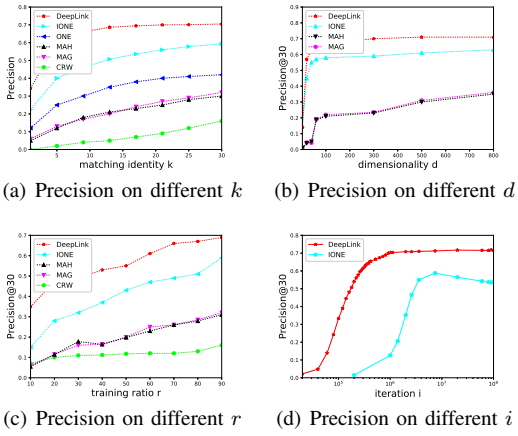
where ra is the rank of that the positive matching identity and m is the number of negative user identities. The goal of all three metrics is to rank the true matching user identities as high as possible.

D. Results

We first examine the performance of various methods on linking precision, as illustrated in Table II, from which we notice that the embedding based methods achieve higher precision compared to the traditional link based prediction method (CRW). Among the latent space learning approaches, DeepLink achieves the highest accuracy in all *top-k* identity matching. Table II randomly reports 6 different k values between 0 and 30 for lack of space, while Figure 2(a) compares the performance of different algorithms by varying the value of k.

TABLE II
COMPARISONS OF LINK PRECISION ON FOURSQUARE-TWITTER DATA.

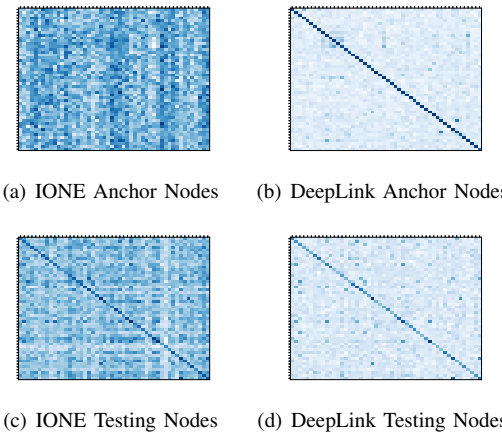
	Precision					
	P@1	***P@5***	***P@9***	***P@13***	***P@21***	***P@30***
CRW	0.0000	0.0219	0.0476	0.0538	0.0909	0.1603
MAG	0.0638	0.1362	0.1705	0.2081	0.2708	0.3229
MAH	0.0500	0.1219	0.1886	0.2148	0.2513	0.0003
ONE	0.1229	0.2533	0.3038	0.3510	0.4071	0.4270
IONE	0.2238	0.4033	0.4638	0.5010	0.5571	0.5970
DeepLink	**0.3447**	**0.5942**	**0.6609**	**0.6866**	**0.7000**	**0.7048**



(a) Precision on different k (b) Precision on different d

(c) Precision on different r (d) Precision on different i

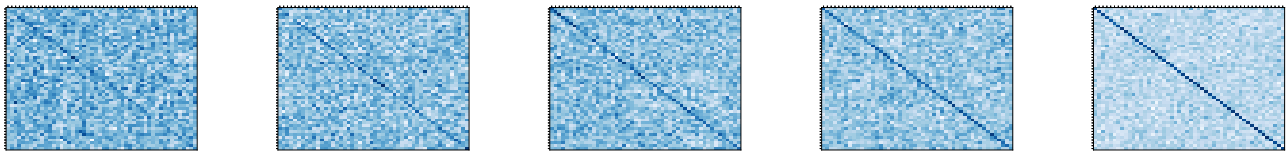
Fig. 2. Linking precision results on Foursquare-Twitter data. Parameters: k is the predicted k top matching identities; d is the embedding dimensionality; r is the percentage of anchor nodes used for training; i is i^{th} training iteration.



(a) IONE Anchor Nodes (b) DeepLink Anchor Nodes

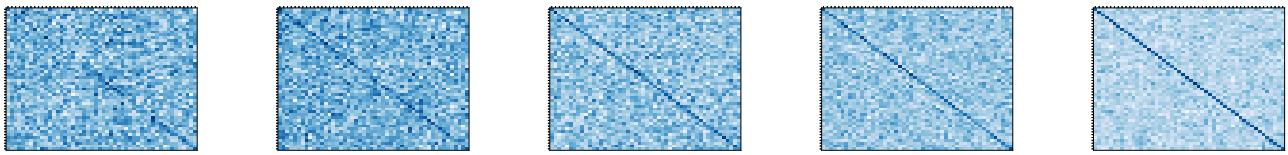
(c) IONE Testing Nodes (d) DeepLink Testing Nodes

Fig. 3. Heatmap comparison on Foursquare-Twitter data. Best viewed when zoomed digitally. Note that IONE may disrupt linkage of anchor nodes after training which makes the network "partially" alignment. The ideal result is that all pairs of anchor nodes are perfectly diagonally aligned.



(a) $\mu = 0.3$ (b) $\mu = 0.4$ (c) $\mu = 0.5$ (d) $\mu = 0.6$ (e) $\mu = 0.9$

Fig. 4. Heatmap (DeepLink) on Lastfm-MySpace data.



(a) $\mu = 0.3$ (b) $\mu = 0.4$ (c) $\mu = 0.5$ (d) $\mu = 0.6$ (e) $\mu = 0.9$

Fig. 5. Heatmap (DeepLink) on Livejournal-MySpace data.

In addition to k, we also investigate the impact of space dimensionality d, the percentage of anchor nodes used in training r and iteration i on the performance of algorithms, which are illustrated in Figure 2 and show that:

- A small number of dimensionality is sufficient for DeepLink and IONE;
- The percentage of anchor nodes used for training greatly affects the performance of all algorithms, while DeepLink achieves the best results due to its ability of leveraging all the anchor nodes (both labeled and unlabeled) for pretraining and dual learning process;
- The number of training iterations required for algorithm convergence is another important factor, as illustrated in Figure 2(d). Two important observations are: (1) DeepLink converges to the best result sooner than IONE; and (2) an interesting finding is that IONE is an empirical method and there is no explicit way to stop training, which might easily incur overfitting problem (declining of the precision) if the value of i becomes larger.

TABLE III
RANKING PERFORMANCE COMPARISON

	MAP	AUC	Hit-Precision
IONE	0.003	0.926	0.926
DeepLink	**0.022**	**0.991**	**0.991**

We also investigated the ranking performance of IONE and DeepLink, and the findings are illustrated in Table III[3]. As discussed in Section IV-E, the existing methodologies for addressing the UIL problem, including IONE, suffer from *top-k mess* problem. DeepLink significantly outperforms IONE in terms of *Hit-Precision* which measures the ranking performance on identity prediction.

[3]Due to a lack of space, we omit the findings with respect to the other algorithms.

Finally, we visualize our observations regarding the performance of DeepLink and IONE in Figure 3, which consists of two components. First, we randomly select 50 anchor nodes (for training) and plot their pair-wise cosine similarity in two networks after alignment (Figure 3(a) and 3(b)). As can be seen, IONE may disrupt the alignment of anchor nodes after training, while DeepLink maintains their user latent space linkage relatively well. Complementary to this, Figure 3(c) and 3(d) depict the alignment results of 50 randomly selected testing anchor nodes – which further demonstrated the advantage of DeepLink (having less predicting noise) compared to IONE.

DeepLink focuses on utilizing network features to link user identities. To further evaluate the performance, we generate several partially simulated synthetic sub-networks using node and edge sampling strategies. The construction process is as follows: (1) Given a pair of real social networks (e.g., Lastfm-MySpace), we first include all n anchor nodes and randomly select 50% of them as non-anchor nodes in the new sub-networks. Therefore, we have two subnetworks A and B with $n/2$ anchor nodes and $n/2$ non-anchor nodes and aligned accordingly. (2) For each of the $n/2$ anchor nodes in network A, we use a random probability ρ to decide whether we should add edges on both networks (we fix $\rho = 0.9$ in our experiments). If both networks for the anchor node are going to be augmented with an edge, we use another parameter μ to control the probability that added edges are connecting to exactly the same pair of nodes for both networks. The parameter μ is mainly used for adjusting the degree of similarity between A and B. (3) To make network A and B more realistic, we randomly add some extra nodes and edges for either A or B.

Figure 4 exhibits the heatmap of DeepLink on 20 randomly chosen testing nodes (from $n/2$ non-anchor nodes) and shows that the similarity of two networks on structures plays a paramount role of user identity linkage. We have similar

results for Livejournal-MySpace shown in Figure 5.

VI. Conclusions and Future Work

We proposed a novel deep reinforcement learning based algorithm – DeepLink – to study the UIL problem by leveraging the duality of mapping between any two networks. DeepLink is an end-to-end network alignment approach and a semi-supervised user identity linkage learning algorithm that does not require a heavy feature engineering and can easily incorporate profile-based features. Despite the challenges of linking identity across different sites and running such mapping in a data-driven manner, we believe that the deep structure learning framework and cross-site reinforcement learning paradigm presented in this work can be beneficial for network alignment and many identity-based applications. Our experiments demonstrated that the proposed framework outperforms various state-of-the-art UIL methods in both linkage precision and ranking matching user identity.

There are several directions to be investigated in the future. First, DeepLink chooses random walk and word2vec as the basic method for capturing the network structure – which perform well and are natural choices in existing works. It will be interesting to investigate some other types of network representation models such as the one based on *spectrum of the graph Laplacian* [34] or *convolutional neural networks* [36]. Another important implication of DeepLink is its ability to underpin novel patterns of cross-site analysis by leveraging the duality of pair-networks such as improving advertising accuracy and product recommendation efficiency. Thus, developing new cross site applications might also be worthwhile pursuing.

VII. Acknowledgements

References

[1] J. Qian, X.-Y. Li, C. Zhang, and L. Chen, "De-anonymizing social networks and inferring private attributes using knowledge graphs," in *IEEE INFOCOM*, 2016.

[2] K. Shu, S. Wang, J. Tang, R. Zafarani, and H. Liu, "User identity linkage across online social networks: A review," *SIGKDD Explorations Newsletter*, vol. 18, no. 2, pp. 5–17, 2017.

[3] M. Jiang, P. Cui, N. J. Yuan, X. Xie, and S. Yang, "Little is much: Bridging cross-platform behaviors through overlapped crowds," in *AAAI*, 2016.

[4] O. Goga, H. Lei, S. H. K. Parthasarathi, G. Friedland, R. Sommer, and R. Teixeira, "Exploiting innocuous activity for correlating users across sites," in *ACM WWW*, 2013.

[5] R. Zafarani and H. Liu, "Connecting users across social media sites: A behavioral-modeling approach," in *ACM SIGKDD*, 2013.

[6] Y. Zhang, J. Tang, Z. Yang, J. Pei, and P. S. Yu, "Cosnet: Connecting heterogeneous social networks with local and global consistency," in *ACM SIGKDD*, 2015.

[7] L. Liu, W. K. Cheung, X. Li, and L. Liao, "Aligning users across social networks using network embedding," in *IJCAI*, 2016.

[8] T. Man, H. Shen, S. Liu, X. Jin, and X. Cheng, "Predict anchor links across social networks via an embedding approach," in *IJCAI*, 2016.

[9] X. Mu, F. Zhu, E. P. Lim, J. Xiao, J. Wang, and Z. H. Zhou, "User identity linkage by latent user space modelling," in *ACM SIGKDD*, 2016, pp. 1775–1784.

[10] S. Zhang and H. Tong, "Final: Fast attributed network alignment," in *ACM SIGKDD*, 2016.

[11] S. Liu, S. Wang, F. Zhu, J. Zhang, and R. Krishnan, "Hydra: large-scale social identity linkage via heterogeneous behavior modeling," in *ACM SIGMOD*, 2014, pp. 51–62.

[12] X. Kong, J. Zhang, and P. S. Yu, "Inferring anchor links across multiple heterogeneous social networks," in *ACM CIKM*, 2013, pp. 179–188.

[13] Y. Shen and H. Jin, "Controllable information sharing for user accounts linkage across multiple online social networks," in *ACM CIKM*, 2014, pp. 381–390.

[14] J. Zhang and P. S. Yu, "Multiple anonymized social networks alignment," in *IEEE ICDM*, 2015, pp. 599–608.

[15] C. Riederer, Y. Kim, A. Chaintreau, N. Korula, and S. Lattanzi, "Linking users across domains with location data: Theory and validation," in *ACM WWW*, 2016.

[16] A. Narayanan, H. Paskov, N. Z. Gong, J. Bethencourt, E. Stefanov, E. C. R. Shin, and D. Song, "On the feasibility of internet-scale author identification," in *IEEE S&P*, 2012.

[17] X. Luo, F. Zhou, M. Liu, Y. Liu, and C. Xiao, "Efficient multi-account detection on ugc sites," in *IEEE ISCC*, 2016.

[18] D. Koutra, H. Tong, and D. Lubensky, "Big-align: Fast bipartite graph alignment," in *ICDM*, 2013, pp. 389–398.

[19] S. Tan, Z. Guan, D. Cai, X. Qin, J. Bu, and C. Chen, "Mapping users across networks by manifold alignment on hypergraph," in *AAAI*, 2014.

[20] R. Zafarani, L. Tang, and H. Liu, "User identification across social media," *ACM Trans. Knowl. Discov. Data*, vol. 10, no. 2, 2015.

[21] X. Zhou, X. Liang, H. Zhang, and Y. Ma, "Cross-platform identification of anonymous identical users in multiple social media networks," *IEEE Transactions on Knowledge & Data Engineering*, vol. 28, no. 2, pp. 411–424, 2016.

[22] J. Zhang and P. S. Yu, "Integrated anchor and social link predictions across social networks," in *IJCAI*, 2015.

[23] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS*, 2013.

[24] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014.

[25] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *ACM SIGKDD*, 2014.

[26] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *ACM WWW*, 2015.

[27] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *ACM SIGKDD*, 2016.

[28] S. Cao, W. Lu, and Q. Xu, "Deep neural networks for learning graph representations," in *AAAI*, 2016.

[29] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *ACM SIGKDD*, 2016.

[30] C. Tu, W. Zhang, Z. Liu, and M. Sun, "Max-margin deepwalk: Discriminative learning of network representation," in *IJCAI*, 2016.

[31] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding," in *AAAI*, 2017.

[32] C. Tu, Z. Zhang, Z. Liu, and M. Sun, "Transnet: Translation-based network representation learning for social relation extraction," in *IJCAI*, 2017.

[33] C. Tu, H. Liu, Z. Liu, and M. Sun, "Cane: Context-aware network embedding for relation modeling," in *Proceedings of ACL*, 2017.

[34] Z. Yang, W. Cohen, and R. Salakhudinov, "Revisiting semi-supervised learning with graph embeddings," in *ICML*, 2016, pp. 40–48.

[35] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *NIPS*, 2016.

[36] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *ICML*, 2016.

[37] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR*, 2017.

[38] J. Zhang and P. S. Yu, "Pct: Partial co-alignment of social networks," in *ACM WWW*, 2016.

[39] S. Cao, W. Lu, and Q. Xu, "Grarep: Learning graph representations with global structural information," in *ACM CIKM*, 2015.

[40] A. Mnih and Y. W. Teh, "A fast and simple algorithm for training neural probabilistic language models," in *ICML*, 2012.

[41] D. He, Y. Xia, T. Qin, L. Wang, N. Yu, T. Liu, and W.-Y. Ma, "Dual learning for machine translation," in *NIPS*, 2016, pp. 820–828.

[42] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, p. 504, 2006.

[43] R. S. Sutton, D. Mcallester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *NIPS*, 1999.