Type-Aware Anchor Link Prediction across Heterogeneous Networks Based on Graph Attention Network

Xiaoxue Li, Yanmin Shang,* Yanan Cao,† Yangxi Li, Jianlong Tan, Yanbing Liu
Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{lixiaoxue, shangyanmin, caoyanan, tanjianlong, liuyanbing}@iie.ac.cn, liyangxi@outlook.com

Abstract

Anchor Link Prediction (ALP) across heterogeneous networks plays a pivotal role in inter-network applications. The difficulty of anchor link prediction in heterogeneous networks lies in how to consider the factors affecting nodes alignment comprehensively. In recent years, predicting anchor links based on network embedding has become the main trend. For heterogeneous networks, previous anchor link prediction methods first integrate various types of nodes associated with a user node to obtain a fusion embedding vector from global perspective, and then predict anchor links based on the similarity between fusion vectors corresponding with different user nodes. However, the fusion vector ignores effects of the local type information on user nodes alignment. To address the challenge, we propose a novel type-aware anchor link prediction across heterogeneous networks (TALP), which models the effect of type information and fusion information on user nodes alignment from local and global perspective simultaneously. TALP can solve the network embedding and type-aware alignment under a unified optimization framework based on a two-layer graph attention architecture. Through extensive experiments on real heterogeneous network datasets, we demonstrate that TALP significantly outperforms the state-of-the-art methods.

Introduction

Anchor Link Prediction (ALP) aims to recognize the accounts of the same natural person across different networks, and the links between these accounts are anchor links (the accounts are anchor nodes). Anchor links play a pivotal role in inter-network applications, such as user profile modeling (Zhan et al. 2017) and recommendation (Fan et al. 2019; Lu et al. 2016). In reality, these networks (such as social networks, academic networks and movie recommendation networks) are heterogeneous networks, which contain various types of nodes and edges. Predicting anchor link across heterogeneous networks is a research hotspot in the industry and academia at present.

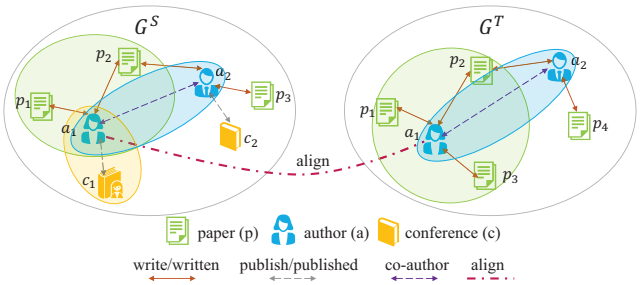

Figure 1: An example of anchor link prediction across heterogeneous networks

With the rise of network embedding, anchor link prediction based on embedding has become the mainstream trend. Based on this trend, the core of existing anchor link prediction methods includes two parts: embedding and alignment. The purpose of embedding part is to obtain the representation vectors of network nodes (accounts) based on network embedding method for each network. The alignment part obtains latent anchor links by estimating pairwise similarity between the embedding representation vectors of nodes in different networks. According to whether the two parts are treated separately, the existing methods can be divided into two categories: unified framework approaches (Liu et al. 2016; Shang et al. 2019) and two-stage approaches (Man et al. 2016; Zhou et al. 2018). All the approaches in above two categories are used to predict anchor links across homogeneous networks which contain only one type of nodes and one type of edges.

In practice, however, heterogeneous networks are ubiquitous. At present, there are few methods for anchor link prediction in heterogeneous networks, especially based on network embedding (Wang et al. 2018; Feng et al. 2019). The idea of anchor link prediction in heterogeneous networks is the same as that in homogeneous networks, however, the difference is how to integrate various types of information into the process of embedding and alignment. Previous methods obtain the embedding vector (named fusion vector) of a user node by fusing information of various types of nodes associated with it from global perspective. Then, anchor links

*Corresponding author: Yanmin Shang
†Corresponding author: Yanan Cao

are predicted based on the similarity of fusion vectors. The researchers have verified the effectiveness of their methods, however, **there also exist defect**: the fusion vector ignores effect of the local type information (information for each type of nodes associated with a user node) on user nodes alignment. This effect is more obvious when there are inconsistent types of nodes in different heterogeneous networks. Take the academic network as an example (Fig.1). G^S contains three types of nodes: author, paper and conference. G^T contains two types of nodes: author and paper. The fusion vector $\vec{f}^S_{a1}$ of author v^S_{a1} in G^S contains three types of information, and the fusion vector $\vec{f}^T_{a1}$ of author v^T_{a1} in G^T contains only two types of information without conference information. So, the information between $\vec{f}^S_{a1}$ and $\vec{f}^T_{a1}$ is inconsistent, which may lead to deviation in estimating their similarity. In addition, for information of author and paper which are included both in $\vec{f}^S_{a1}$ and $\vec{f}^T_{a1}$, each type of information has its own impact on user alignment.

To address the above mentioned challenge, in this paper, we propose a unified framework of type-aware anchor link prediction across heterogeneous networks (TALP) based on graph attention architecture. TALP not only considers the effect of fusion vector on users alignment from global perspective, but also considers the impact of type information on alignment from local perspective. All the considering factors are formulated into a single objective function so that minimizing it can allow network embedding and user nodes alignment to be achieved simultaneously in heterogeneous networks.

Specifically, TALP consists of two parts: *n-tuple representation* and *type-aware alignment*. For *n*-tuple representation, we conduct network embedding on each heterogeneous network to lean the *n*-tuple embedding vectors of each user node. Considering that fusion vectors will lose type information, we use a two-layer Graph Attention architecture (GAT) to learn the fusion vector and type-aware vectors simultaneously. The first layer of GAT aims to integrate the embedding vector which belongs to the same type, and obtain the local representation of the user node on this type information, called type-aware embedding vector. The second layer of GAT aims to fuse different type-aware vectors of the user node to obtain the global embedding vector, called type-fusion embedding vector. For type-aware alignment, we believe that type information and fusion information work together to affect user nodes alignment. In other words, we collaboratively measure the pairwise-similarity of fusion embedding vectors and pairwise-similarity of type-aware embedding vectors, which can guide the *n*-tuple embedding process.

In a nutshell, the contributions of this paper can be summarized as follows:

- In this paper, we propose a type-aware anchor link prediction framework across heterogeneous networks. This framework predicts anchor links not only based on the pairwise-similarity between type-fusion vectors of user nodes, but also considers the pairwise-similarity between type-aware vectors associated with user nodes according to types.

Table 1: Notations

Notation	Description
G^S, G^T	the input source/target network
V^S, A^S	the node/edge set of G^S
V^T, A^T	the node/edge set of G^S
R^S, R^T	the node type set of G^S/G^T
$\|V\|, \|E\|, \|R\|$	the number of nodes/edges/node types
r, g	a node type
D	the dimension of embeddings
D', D''	the dimension of type-aware/fusion embeddings
K	the number of multi-head
$\mathcal{N}^r$	the r-th type neighborhood set
$\vec{f}, \vec{u}$	the type-fusion/aware embedding
$\vec{x}$	the initial embedding of a node
$\mathcal{B}$	the anchor links set

- For anchor link prediction across heterogeneous networks, this paper proposed a unified framework based on graph attention, which can learn *n*-tuple embedding vectors of each user node while predicting anchor links.
- We evaluate the proposed framework (TALP) on two pairs of real-word heterogeneous networks. The results demonstrate that our method constantly outperforms the state-of-the-art approaches which predict anchor links by only considering the pair-wise similarity between fusion vectors.

Problem Formulation

In this section, we first introduce concepts in heterogeneous networks, and then introduce the embedding representation of nodes (type-aware embedding and type-fusion embedding). Finally, a formal definition of the type-aware anchor link prediction problem is given.

Definition 1. ***Heterogeneous network*** *A heterogeneous network is defined as a network with multiple types of nodes and/or multiple types of links. It can be denoted as* $G = \{V, A, R\}$*, where* V *is a set of nodes,* A *is a set of links, and* R *represents the node type union.*

Take the heterogeneous network G^S in Fig.1 as an example, $G^S = \{V^S, A^S, R^S\}$, $V^S = \{v^S_{p1}, v^S_{p2}, v^S_{p3}, v^S_{a1}, v^S_{a2}, v^S_{c1}, v^S_{c2}\}$, $A^S = \{(v^S_{a1}, v^S_{p1}), (v^S_{a1}, v^S_{p2}), (v^S_{a1}, v^S_{a2}), (v^S_{a1}, v^S_{c1}), (v^S_{a2}, v^S_{p2}), (v^S_{a2}, v^S_{p3}), (v^S_{a2}, v^S_{c2})\}$, $R^S = \{p, c, a\}$

Next, we will take G^S as an example to introduce the type-aware embedding and type-fusion embedding respectively.

PROBLEM 1. ***Type-aware embedding****: Given a user node* v^S_{ai} *in* G^S*(an author node),* $\mathcal{N}^r_{v^S_{ai}}$ *represents the set of* r*-th (*$r \in R^S$*) type neighborhoods of* v^S_{ai}*. For each node* $v_j \in \mathcal{N}^r_{v^S_{ai}}$*, its embedding vector is denoted as* $\vec{e}_j$*, integrating each embedding vector* $\vec{e}_j$ $(j = 1, 2...|\mathcal{N}^r_{v^S_{ai}}|)$ *of node in* $\mathcal{N}^r_{v^S_{ai}}$ *can obtain the type-aware embedding vector of* r*-th type information of* v^S_{ai}*, denoted as* $\vec{u}^{Sr}_{ai}$*.*

PROBLEM 2. ***Type-fusion embedding****: Given a user node* v^S_{ai} *in* G^S*, the type-fusion embedding problem is to integrate*

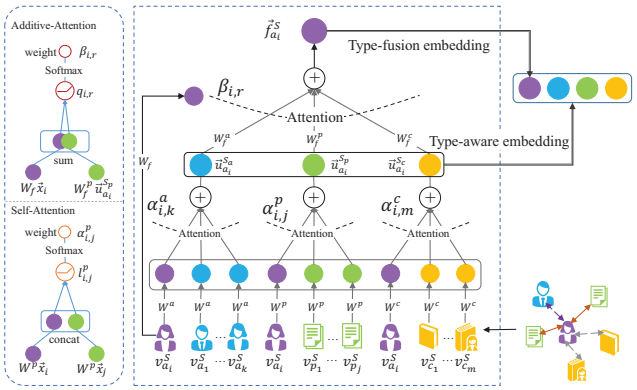

Figure 2: *n*-tuple representation

each type-aware embedding vector $\vec{u}_{ai}^{Sr}$ $(r \in R^S)$ *associated with* v_{ai}^S*, denoted as* $\vec{f}_{ai}^S$.

PROBLEM 3. ***Type-aware Anchor link prediction****: Given two heterogeneous networks:* $G^S = \{V^S, A^S, R^S\}$ *and* $G^T = \{V^T, A^T, R^T\}$*,* (v_{ai}^S, v_{aj}^T) *is an anchor link iff* $v_{ai}^S \in V^S$ *and* $v_{aj}^T \in V^T$ *identify the same nature person. Here, the representations of* v_{ai}^S *and* v_{aj}^T *are n-tuples containing type-fusion embedding vector and type-aware embedding vectors, denoted as* $(\vec{f}_{ai}^S, \vec{u}_{ai}^{Sr}, ...)$ *and* $(\vec{f}_{aj}^T, \vec{u}_{aj}^{Tg}, ...)$*,* $r \in R^S, g \in R^T$*. Type-aware anchor link prediction aims to predict the unobserved anchor links by matching n-tuple representation vectors between each pair of user nodes across* G^S *and* G^T*.*

Proposed Model

In this paper, we propose a unified framework TALP to align anchor user nodes across heterogeneous networks, this framework leverages graph attention to help learn the type-aware vectors and type-fusion vector associated with each user node, and obtain the *n*-tuple representation of each user node. On this basis, we can predict whether there is an anchor link between two user nodes by collaborative measuring the pairwise-similarity of each element vector in their *n*-tuple representations.

n-tuple Representation

In this section, we use two GAT (Graph Attention Network) to learn *n*-tuple representation of each user node in G^S and G^T respectively. The parameters in two GATs are shared, taking G^S as an example, we introduce the acquisition process of *n*-tuple representation for each user node.

Type-aware Embedding The GAT we used contains two attention layers: the first layer aims to learn the type-aware embedding and the second layer aims to learn the type-fusion embedding (Fig.2).

For a user node v_{ai}^S in G^S, we initialize feature vectors of v_{ai}^S and its each neighbor $v_j \in \mathcal{N}_{v_{ai}^S}^r$ to the same dimension D firstly. The initial feature vector is extracted according to information contained in node. Specifically, Word2vec is used for nodes containing text information, and nodes with unclear text information adopt random assignment method to obtain their initial feature vectors. The initial feature vectors of v_{ai}^S and v_j are denoted as $\vec{x}_i$ and $\vec{x}_j$ respectively. To learn the r-th type-aware embedding vector $\vec{u}_{ai}^{Sr}$ of v_{ai}^S, we fed all $\vec{x}_j, 1 \leq j \leq |\mathcal{N}_{v_{ai}^S}^r|$ and $\vec{x}_i$ into the first attention layer.

Specially, for each node-pair v_{ai}^S and its arbitrary neighbor v_j, we first use a linear transformation, parameterized by weight matrix $W^r \in \mathbb{R}^{D' \times D}$, to transfer the initial features into higher-level features, and then compute the importance score of v_j to v_{ai}^S with self-attention (Vaswani et al. 2017) according to Eq.(1):

$$l_{i,j}^r = \vec{o}^T [W^r \vec{x}_i || W^r \vec{x}_j] \tag{1}$$

where $\cdot^T$ represent transposition, $||$ is the concatenation operation, $\vec{o} \in \mathbb{R}^{2D'}$ is a weight vector, D' is the dimension of each type-aware vector.

Then we compute attention coefficient between node v_{ai}^S and its neighbor v_j. We inject the adjacency matrix of G^S into the attention mechanism by performing *masked attention*(Velickovic et al. 2017), and the normalized attention coefficient is expressed as:

$$\alpha_{i,j}^r = \frac{exp(LeakyReLU(l_{i,j}^r))}{\sum_{t=1}^{|\mathcal{N}_{v_{ai}^S}^r|} exp(LeakyReLU(l_{i,t}^r))} \tag{2}$$

The r-th type-aware embedding vector $\vec{u}_{ai}^{Sr}$ of v_{ai}^S is computed as:

$$\vec{u}_{ai}^{Sr} = \sigma\Big(\sum_{j=1}^{|\mathcal{N}_{v_{ai}^S}^r|} \alpha_{i,j}^r W^r \vec{x}_j\Big) \tag{3}$$

where σ is the Elu activation function.

To stabilize the learning process of self-attention, we employ multi-head attention on computing the type-aware embedding vectors, and K is the number of attention mechanisms. So, we represent the Eq.(3) as a form with multi-head attention mechanisms:

$$\vec{u}_{ai}^{Sr} = \sigma\Big(\frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{|\mathcal{N}_{v_{ai}^S}^r|} \alpha_{i,j}^{k,r} W^{k,r} \vec{x}_j\Big) \tag{4}$$

where $\alpha_{i,j}^{k,r}$ is the normalized attention coefficient computed by the k-th attention mechanism, and $W^{k,r}$ is the weighted matrix of k-th attention mechanism. In training, K attention mechanisms are independent and parallel.

Type-fusion Embedding To obtain the type-fusion embedding vector of v_{ai}^S, we fed all type-aware embedding vectors $\vec{u}_{ai}^{Sr}(r \in R^S)$ and the initial feature $\vec{x}_i$ into the second attention layer of GAT, and then aggregate these vectors with attention coefficients.

Specially, for each vector-pair $\vec{x}_i$ and $\vec{u}_{ai}^{Sr}$, as their dimensions are different (D and D' respectively), we adopt additive-attention (Bahdanau, Cho, and Bengio 2015) to

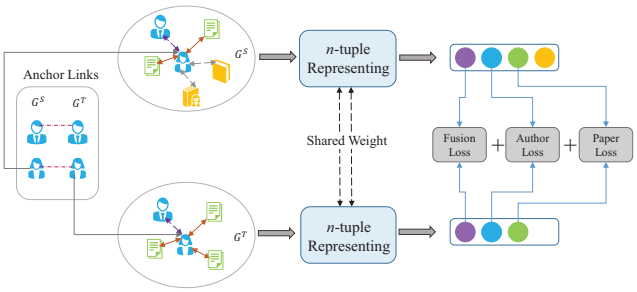

Figure 3: Type-aware alignment

compute attention coefficient between $\vec{x}_i$ and $\vec{u}_{ai}^{Sr}$. First, the importance score of $\vec{u}_{ai}^{Sr}$ to the type-fusion vector is computed as follows:

$$q_{i,r} = \vec{m}^T tanh(W_f \vec{x_i} + W_f^r \vec{u}_{ai}^{Sr}) \tag{5}$$

where $\vec{m} \in \mathbb{R}^{D''}$ is a weight vector, $W_f \in \mathbb{R}^{D'' \times D}$ and $W_f^r \in \mathbb{R}^{D'' \times D'}$ are weight matrices, D'' is the dimension of the type-fusion vector of v_{ai}^S.

The attention coefficient can be computes as:

$$\beta_{i,r} = \frac{exp(q_{i,r})}{\sum_{t \in R^S} exp(q_{i,t})} \tag{6}$$

The type-fusion embedding vector of v_{ai}^S is denoted as $\vec{f}_{ai}^S$, and it can be computed according to the following equation:

$$\vec{f}_{ai}^S = \sum_{r \in R^S} \beta_{i,r} \vec{u}_{ai}^{Sr} \tag{7}$$

So far, we can obtain the n-tuple representation $(\vec{f}_{ai}^S, \vec{u}_{ai}^{Sr}, ...)$ of v_{ai}^S, $r \in R^S$. Similarly, the n-tuple representation of each user node in G^T can be obtained. It's important to note that the weight matrices $W^{k,r}$, W_f and W_f^r and the weight vector $\vec{o}$, $\vec{m}$ are shared in G^S and G^T, which ensures that the embedding vector tuples of user nodes in two networks are in the same embedding space.

Type-aware Alignment

Given two heterogeneous networks G^S and G^T, $v_{ai}^S \in G^S$ and $v_{aj}^T \in G^T$, if there is an anchor link between v_{ai}^S and v_{aj}^T, the embedded vector tuples corresponding to them $(\vec{f}_{ai}^S, \vec{u}_{ai}^{Sr}, ...)$ and $(\vec{f}_{aj}^T, \vec{u}_{aj}^{Tg}, ...)$ should be as close as possible. Just as important, if $v_{ai}^{'S} \in G^S$ and $v_{aj}^{'T} \in G^T$ do not identify to the same nature person, the distance between their n-tuple embedding representation should be as far as possible. In other words, the distances between aligned user nodes should be minimized and those of unaligned user nodes should be maximized. In practice, if the types of nodes in two heterogeneous networks are inconsistent, for the heterogeneous network lacking a certain type of nodes, the corresponding type-aware embedding vector in n-tuple of a user node in this network is supplemented to $\vec{0}$ when tuples alignment. That is to say, the tuple length must be the same when aligning, as shown in Fig.3. We use R^F to denote the length of tuple on type-aware alignment, where $R^F = R^S \cup R^T$. Therefore, the objective function is:

$$\begin{aligned}\mathcal{L} = \sum_{(v_{ai}^S, v_{aj}^T) \in \mathcal{B}} \sum_{(v_{ai}^{'S}, v_{aj}^{'T}) \notin \mathcal{B}} \{\omega[d(\vec{f}_{ai}^S, \vec{f}_{aj}^T) + \\ \xi - d(\vec{f}_{ai}^{'S}, \vec{f}_{aj}^{'T})] + \sum_{r \in R^F} \lambda^r [d(\vec{u}_{ai}^{Sr}, \vec{u}_{aj}^{Tr}) + \\ \xi - d(\vec{u}_{ai}^{'Sr}, \vec{u}_{aj}^{'Tr})]\}\end{aligned} \tag{8}$$

where $\mathcal{B}$ is the set of known anchor links, (v_{ai}^S, v_{aj}^T) denote an anchor link and $(v_{ai}^{'S}, v_{aj}^{'T})$ is not an anchor link. $\vec{f}_{ai}^S, \vec{f}_{aj}^T, \vec{f}_{ai}^{'S}, \vec{f}_{aj}^{'T}$ are the type-fusion embedding vector of $v_{ai}^S, v_{aj}^T, v_{ai}^{'S}$ and $v_{aj}^{'T}$ respectively. $\vec{u}_{ai}^{Sr}, \vec{u}_{aj}^{Tr}, \vec{u}_{ai}^{'Sr}, \vec{u}_{aj}^{'Tr}$ are their type-aware embedding vector of r-th type information. $d(,)$ is a distance formula, in this paper $d(\vec{f}_{ai}^S, \vec{f}_{aj}^T) = ||\vec{f}_{ai}^S - \vec{f}_{ai}^T||_1$. ξ is a margin hyper-parameter separating anchor links and unanchored links. ω and λ^r are hyper-parameters balancing the importance between type-fusion similarity and type-aware similarity for anchor link prediction, here, $\omega + \sum_{r \in R} \lambda^r = 1$.

Algorithm 1 The TALP algorithm.

Input: Heterogeneous network G^S and G^T; Adjacency matrix A^S and A^T; Anchor links set $\mathcal{B}$; Iteration Γ; Hyper-parameter ω, λ^r, ξ, K, D,D',D''

Output: parameter set $\Theta^* = \{\vec{o}, \vec{m}, W^{k,r}, W_f, W_f^r\}$;

```
Extract initial feature matrix E^S and E^T;
Learn initial parameter Θ^0;
Initial t ← 1
while t < Γ do
    for v_ai^S ∈ G^S do
        for r ∈ R^S do
            update u_ai^Sr with Eq.(4).
        end for
        update f_ai^S with Eq.(7)
    end for
    for v_aj^T ∈ G^T do
        for r ∈ R^T do
            update u_aj^Tr with Eq.(4).
        end for
        update f_aj^T with Eq.(7)
    end for
    updated L with Eq.(8)
    update parameter Θ^t with L.
end while
```

We summarize our algorithm in Algorithm.1. The time complexity of our n-tuple representation on source and target network are $O(R^S(|V^S|DD' + |E^S|D')D'')$ and $O(R^T(|V^T|DD' + |E^T|D')D'')$ respectively, which are linear to the sum number of edges and nodes. Computing embedding vectors of source network and target network are parallel, the time complexity depends on the network with a larger number of nodes and edges. Besides, as the time

complexity of type aware is caused by calculation similarity, which can be ignored. Therefore, the time complexity of TALP mainly depends on the sum of nodes number and edges number in source network or target network.

Experiment

Experiment Setup

Datasets and Evaluation Metrics We conduct our experiment on two pairs of real-word heterogeneous networks: Aminer-Mag and Twitter-Foursquare. Aminer-Mag (Tang et al. 2008) is a pair of citation networks. In Aminer, there are three types of nodes: conference nodes, paper nodes and author nodes, while in Mag, the types of nodes are paper and author. Twitter-Foursquare (Zhang and Yu 2015) is a pair of social networks, the types of nodes in them are user, tweet and location. Table 2 illustrates the statistics of these data sets. We use *Precision@k* (*P@k*) and Mean Average Precision (*MAP*) (Zhou et al. 2018) to evaluate the performance on ALP.

Table 2: Statistics of the Datasets

Datasets	Nodes	#Nodes	Rel.	#Rel.	#Anc.
Mag	Conf.	280	C-P	9,490	873
	Paper	9,490	P-P	94,312	
	Auth.	1,365	A-P	7,695	
AMiner	Auth.	1,456	A-P	8,348	
	Paper	8,936	P-P	85,040	
Twitter	User	5,220	U-U	164,919	3148
	Tweet	9,490,707	U-T	9,490,707	
	Loc	297,183	U-L	615,515	
Foursq.	User	5315	U-U	76,972	
	Tweet	48,755	U-T	48,756	
	Loc	38,921	U-L	48,756	

Baselines and Settings

We compare our proposed model TALP with the following recent anchor link predicting methods:

- **MAG** (Tan et al. 2014) MAG uses manifold alignment on graph to map users for homogeneous network. The dataset Twitter-Foursquare used in MAG is the same as that in our paper, however the source code of MAG is not public, so in our paper, we directly copy the experimental results reported in MAG to compare with our method.
- **IONE** (Liu et al. 2016) IONE predicts anchor links by learning the follower-ship embedding and followee-ship embedding of a user simultaneously, it is also proposed for homogeneous network. In this paper, for each heterogeneous network, we only keep user nodes and links between them, and input the directed sub-network into to IONE.
- **DeepLink** (Zhou et al. 2018) As a ALP method for homogeneous networks, DeepLink employs unbiased random walk to generate embeddings, and then uses MLP to map users. Similarly to MAG, we directly copy the experimental results reported in DeepLink to compare with our method because of the source code is not public and the shared Twitter-Foursquare datasets between our paper and DeepLink.
- **HAN** (Wang et al. 2019) HAN is a heterogeneous network embedding model which is based on GAT. In this paper, we use it to obtain a embedding vector for each user node in heterogeneous network, and then map user nodes by estimating the pairwise similarity between their embedding vectors.
- **PME** (Chen et al. 2018) PME is also a heterogeneous network embedding method which projects various types of links into different sub-spaces and eventually gets an overall embedding vector for each node. In this paper, we use PME to obtain the embedding vectors and then map them.
- **HHNE** (Wang, Zhang, and Shi 2019) HHNE uses naive active learning to obtain the embedding vector of each node in heterogeneous network. In this paper, we use HHNE to obtain the embedding vectors and then align them.
- **TALP**$_f$ and **TALP**$_a$ are the variants of TALP. TALP$_f$ only uses the type-fusion embedding vector to align user nodes across heterogeneous networks. TALP$_a$ only uses various types of type-aware embedding vectors to align user nodes. We take them as baseline methods to analyze the importance of type-fusion vector and type-aware embedding vectors for anchor link prediction respectively.

Performance Comparison

In the experiment, all the hyper-parameters of both compared methods and our method TALP are tuned to perform the best on test set. For our model, $D = 300$, $D' = D'' = 128$, $\xi = 3$, $\omega = 0.4$ and $K = 3$, and for all baseline methods we set the parameters the same as original works.

Table 3 gives the convinced results of anchor link prediction. From this table, we can observe that our model TALP consistently outperforms all baselines on two pairs of datasets. More specially:

- TALP is significantly better than previous anchor links prediction methods (HAN, PME, HHNE) for heterogeneous networks. The reason lies in that previous methods align anchor user nodes only based on the pairwise-similarity of fusion vectors. By comparison, TALP utilizes type-aware embedding vectors and type-fusion embedding vector to align anchor user nodes simultaneously. This precisely shows the effect of type information matching on anchor links. In addition, same as HAN, PME, HHNE, the variant TALP$_f$ also only uses the fusion vector to align users, but its performance are better than them. The reason for this is that our graph attention architecture can better model fusion vectors of user nodes. Meanwhile, HAN,PME and HHNE outperform homogeneous networks ALP methods (MAG, IONE, DeepLink), which indicates that the embedding vector of a user node in heterogeneous network includes richer information than homogeneous networks.
- TALP performs better than TALP$_f$ and TALP$_a$. For TALP and TALP$_f$, the only difference between them is whether the matching of type-aware vectors is introduced. Obviously, introducing matching of type-aware vectors can im-

Table 3: Performance comparison on anchor link prediction

Dataset	Twitter-Foursquare						Aminer-Mag					
Metrics(%)	*P@1*	*P@5*	*P@9*	*P@21*	*P@30*	*MAP@30*	*P@1*	*P@5*	*P@9*	*P@21*	*P@30*	*MAP@30*
MAG	6.38	13.62	17.05	27.08	32.29	-	-	-	-	-	-	-
IONE	22.38	40.33	46.38	55.71	59.70	32.79	34.18	39.27	49.56	57.81	63.42	39.19
DeepLink	34.47	59.42	66.09	70.00	70.48	47.78	-	-	-	-	-	-
HAN	38.69	60.38	71.16	75.49	78.33	50.22	42.90	64.33	70.27	76.91	80.16	53.25
PME	40.51	59.89	73.18	78.54	80.95	52.42	45.91	65.81	73.92	79.38	82.97	55.96
HHNE	38.72	60.45	69.92	75.96	79.13	51.28	42.65	65.19	72.87	75.34	81.96	53.35
TALP$_f$	42.66	62.32	88.50	90.40	93.79	55.89	50.56	70.34	90.07	95.23	97.68	66.25
TALP$_a$	43.37	69.82	89.96	92.57	94.27	57.68	63.85	76.49	95.72	95.23	98.81	79.49
TALP	**43.79**	**72.32**	**93.22**	**95.20**	**98.69**	**59.33**	**71.19**	**77.68**	**96.32**	**96.89**	**99.87**	**82.39**

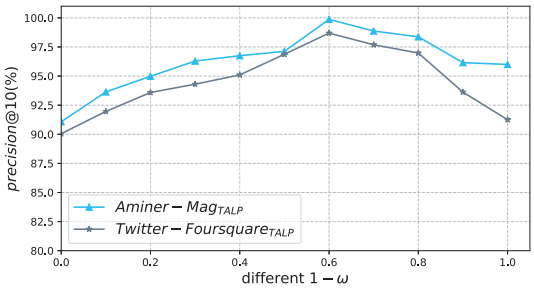

Figure 4: Performance of TALP on different $(1-\omega)$

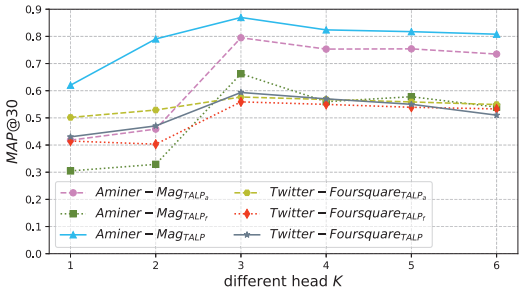

Figure 5: Performance of TALP on different number of K

prove the performance of anchor user nodes alignment. For TALP and TALP$_a$, the difference between them is whether the matching of fusion vectors is introduced. From table 2, we can see clearly that TALP is better than TALP$_a$, showing that type-fusion information is also beneficial for ALP.

- TALP$_a$ outperforms TALP$_f$. By contrast, TALP$_a$ has better performance than TALP$_f$, which indicates that type-aware information is more efficient than information of type-fusion for anchor links prediction across heterogeneous networks.
- The performance improvement on Aminer-Mag is obviously higher than Twitter-Foursquare. The difference between the two pairs of datasets is that the data types of Aminer and Mag are inconsistent. This proves that our model TALP can better predict anchor links in heterogeneous networks with inconsistent data types.

Discussion

In this section, we evaluate how different choices of parameters affect our model's performance. In the following experiments, except for the parameter being tested, the rest parameter are set as the optimal configurations.

Performance on different ω and λ^r In our model, ω is to weight the importance of type-fusion similarity for ALP, and λ^r is to weight the importance of r-th type-aware similarity for ALP. As $\omega+\sum_{r\in R}\lambda^r=1$, we only evaluate the effect of the change in ω on alignment performance of TALP. From Fig.4, we found that (1) TALP achieves the worst performance under $(1-\omega=0)$ setting, which indicates that only using type-fusion similarity ($\omega=1$) is not enough for anchor link prediction, it is necessary to introduce type-aware information; (2) With the growth of $(1-\omega)$, the performance of TALP increase firstly, which indicates that type-aware information indeed can predict anchor links more accurately. However, with $(1-\omega)$ further increase, the performance drops gradually, and this shows that it is very important to balance the information of type-aware and the information of type-fusion.

Performance on different K For learning type-aware vectors, we use multi-head attention mechanism. The number of head K also affect the performance on anchor links prediction. From Fig.5, we can see that TALP, TALP$_a$ and TALP$_f$ on two pairs of datasets achieve the best performance when $K=3$, indicating that $K=3$ best express the type-aware information of user nodes and delivers alignment characteristics of user nodes across heterogeneous networks. The performance on all datasets begins to gradually rise to the highest point and then declines as the number of head K grows. This mainly because that too small K can not capture the richer type-aware information and larger K may introduce noisy.

Performance on different training ratio For different training-to-test ratios, as observed from Fig.6(a) and Fig.7(a), TALP outperforms all the baselines on two pairs of datasets. Even for the ratio as low as 10% to 20%, the performance of them still superior to the baselines. In addition, TALP, TALP$_a$ and TALP$_f$ achieve best results when

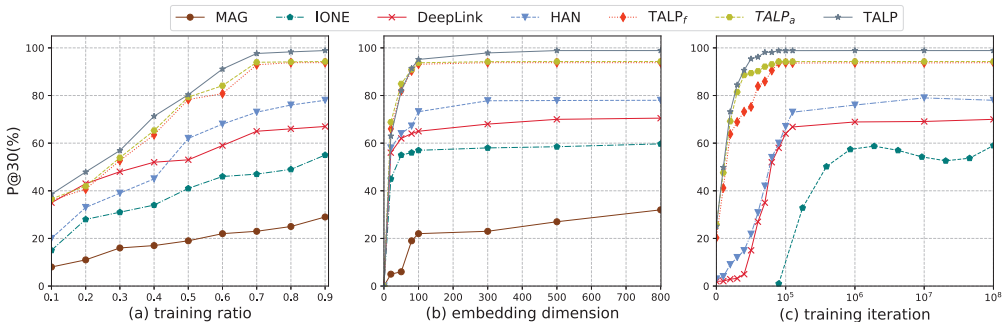

Figure 6: Detailed performance on Twitter-Foursquare

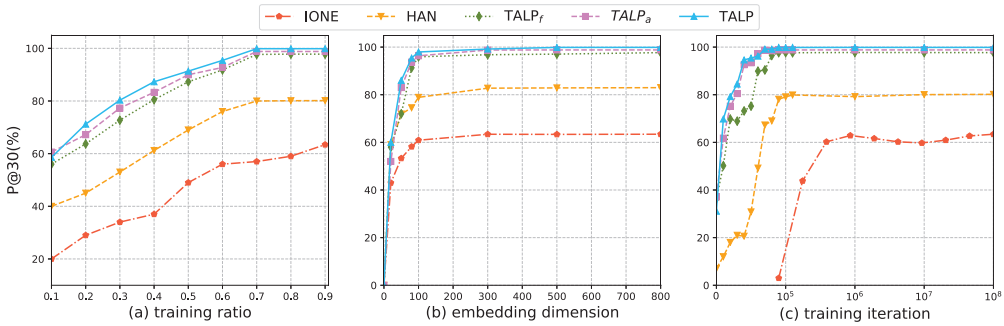

Figure 7: Detailed performance on Aminer-Mag

the ratio rose to 70% while other baselines achieve good performance when the training ratio is around 90%, which demonstrates the robustness of our model.

Performance on different embedding dimension For different embedding dimension, according to Fig.6(b) and Fig.7(b), we observe that a low dimensionality is sufficient for all the methods except MAG. It is well known that the complexity of the learning algorithm is highly dependent on spatial dimensions. In this paper, we select 128 as the optimal dimension.

Performance on different training iteration Fig.6(c) and Fig.7(c) show how the performance of our model and baselines methods changes with different training iterations. We observe that the performance of all the methods consistently achieve better results as the iteration number increases. The number of training iterations reflects the convergence speed of algorithms. TALP converges to the best result sooner than all baselines.

A case study

To better understand and gain deeper insights into the effect of node type information difference on alignment process, we randomly sample two pairs of real anchor users and show their neighbors in Fig.8. The yellow and white rows represent Aminer and Mag datasets respectively. In particular, we observe that:

- *Type-aware alignment could predict anchor users that type-fusion method can not*. For the neighbors of paper type, user "I**r Ivanov" has both the same papers and different papers in two datasets. The role of different papers will be magnified. For example, "I**r Ivanov" links a paper "5-Selenization of salicylic acid ***" in Aminer but links another paper "Experimental ionization of atomic ***" in Mag. This difference leads to the similarity of this user's type-fusion vectors in two datasets are only 0.49, which is hard to determine whether there exists an anchor link. By comparison, considering type information difference with type-aware method, the similarity is 0.61, which makes it easier to determine an anchor link. This again verifies why our method can improve alignment precision.
- *For the anchor links that can be predicted by both methods, type-aware alignment achieves higher similarity than that of type-fusion*. Although user "E**er Zartzer" has the same paper information in two datasets, the author information is different. In specific, author "A**sM. Yinnon" in Aminer is different from that in Mag ("A**s.M. Yinnon"). Besides, conference information in Mag is "NULL" which is different from that in Aminer. Type-fusion method ignores these type information differences which was focused on by the type-aware method, so the similarity between anchor users are lower.

Related Work

Heterogeneous Network Embedding

Heterogeneous network embedding refers to learning representation of nodes/edges in heterogeneous network. In re-

Anchor users	Paper	Author	Conference	$TALP_f$	$TALP_a$	TALP
E**er Zartzer	Prospective Evaluation of Colonization ***	R**en Friedmann; D**id Raveh; A**sM. Yinnon; …	Infection Control and Hospital Epidemiology	0.87	0.92	0.95
E**er Zartzer	Prospective Evaluation of Colonization ***	R**en Friedmann; D**id Raveh; A**s.M. Yinnon; …	NULL			
I**r Ivanov	5-Selenization of salicylic acid ***; Probing dimerization and structural ***;	Sun** Yu; H**ut Kuhn; W** Shang; S**ne Stehling; J**zy Jankun; …	Organic &bimolecular chemistry; Journal of Molecular biology	0.49	0.61	0.65
I**r Ivanov	Experimental ionization of atomic ***; Probing dimerization and structural ***;	K**s Bart; Mi**el Pullen; D**el Weflen; W**g Shang; S**ne Stehling; J**zy Jankun ; …	NULL			

Figure 8: A case of type-fusion and type-aware alignment. In order to protect the privacy of authors, we use "**" instead of the real letter of authors' and papers' name

cent years, many researchers have done a lot of work in this area. PME (Chen et al. 2018) and (Sun, Zhao, and Liu 2015) project various relations into different embedding subspace and then map them into the same embedding space via translation or coordinate matrix. HAN (Wang et al. 2019) provides a meta-path based GAT model to learn the embedding vectors through node-level and semantic-level attention mechanism. EGNN (Gong and Cheng 2018) jointly encodes both nodes and edges into an unified low-dimensional space via GCN. GaAN (Zhang et al. 2018) apply attention mechanism into gated neural network to solve node classification problem. Unlike the traditional multi-head attention mechanism, which equally consumes all attention heads, GaAN uses a convolution sub-network to control the importance of each attention head. EOE(Xu et al. 2017) learns the embedding representation of two networks,and incorporates a harmonious embedding matrix to transform the representation of different networks into the same space.

Anchor Link Prediction

The traditional ALP methods mainly compute pair-wise similarity based on well-design hand-crafted features, for example, MNA (Kong, Zhang, and Yu 2013) extracts features from the social structural and text content information. (Koutra, Tong, and Lubensky 2013) extracts features form various node attributes, e.g., user-name, typing patterns and language patterns, etc. Though achieving great performance, they are time-consuming, labor expensive and usually suffer from inflexible extension.

Different from the above hand-crafted features methods, the embedding based methods could learn node's features automated, which includes embedding and alignment parts. According to whether the two parts are treated separately, existing methods can be divided into two categories: the first category is to predict anchor links by taking those two parts as two independent steps, such as PALE (Man et al. 2016) firstly learns network embedding via capturing each node's major structural regularity, and then learning a mapping function across the two learned low-dimensional spaces. DeepLink (Zhou et al. 2018) 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. The second category is to solve embedding and alignment process simultaneously based on a unified framework. For example, IONE (Liu et al. 2016) considers both follower/followee-ship in network and anchor users across networks via formulating them into a single objective function. PAAE (Shang et al. 2019) devices an auto-encoder to capture major structural regularity in one network via an adversarial regularization and then formulates both embedding and alignment problem into a single objective function.

Besides, there are some works about anchor link prediction across heterogeneous networks. LHNE (Wang et al. 2018) embeds cross-network structural and content information into a unified space by jointly capturing the friend-based and interest-based user co-occurrence in intra-network and inter-network, respectively. And then align users based on those embedding vectors. DPLink (Feng et al. 2019) proposes an end-to-end deep neural network, which solves anchor link prediction based on heterogeneous mobile data collected from services with different natures.

Conclusion

In this paper, we present a type-aware anchor link prediction framework across heterogeneous networks, which considers the effects of the local type information on user nodes alignment. This framework predicts anchor links not only based on the pairwise-similarity between type-fusion vectors of user nodes, but also considering the pairwise-similarity between type-aware vectors of different types of nodes associated with user nodes. For each user node, TALP can learn a n-tuple representation based on two-layer graph attention architecture. Anchors are used to supervise the objective function which aims to minimizing the distance between anchors. On this basis, we can predict whether there is an anchor link between two user nodes via measuring the pairwise-similarity of each element vector in their n-tuple representations. Experiments on real-world heterogeneous network datasets demonstrate the effectiveness and

efficiency of TALP. In future, we plan to extend our model to anchor link prediction across multiple (more than two) heterogeneous networks.

Acknowledgments

This work is supported by the National Key $R\&D$ Problem of China (NO.2018YFB1004703), the National Natural Science Foundation of China (NO.U1736106, NO.61602466). We thank all authors for their contributions and all anonymous reviewers for their constructive comments.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Chen, H.; Yin, H.; Wang, W.; Wang, H.; Nguyen, Q. V. H.; and Li, X. 2018. PME: projected metric embedding on heterogeneous networks for link prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, 1177–1186.

Fan, S.; Zhu, J.; Han, X.; Shi, C.; Hu, L.; Ma, B.; and Li, Y. 2019. Metapath-guided heterogeneous graph neural network for intent recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019.*, 2478–2486.

Feng, J.; Zhang, M.; Wang, H.; Yang, Z.; Zhang, C.; Li, Y.; and Jin, D. 2019. Dplink: User identity linkage via deep neural network from heterogeneous mobility data. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, 459–469.

Gong, L., and Cheng, Q. 2018. Adaptive edge features guided graph attention networks.

Kong, X.; Zhang, J.; and Yu, P. S. 2013. Inferring anchor links across multiple heterogeneous social networks. In *22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*, 179–188.

Koutra, D.; Tong, H.; and Lubensky, D. 2013. BIG-ALIGN: fast bipartite graph alignment. In *2013 IEEE 13th International Conference on Data Mining, Dallas, TX, USA, December 7-10, 2013*, 389–398.

Liu, L.; Cheung, W. K.; Li, X.; and Liao, L. 2016. Aligning users across social networks using network embedding. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, 1774–1780.

Lu, C.; Xie, S.; Shao, W.; He, L.; and Yu, P. S. 2016. Item recommendation for emerging online businesses. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, 3797–3803.

Man, T.; Shen, H.; Liu, S.; Jin, X.; and Cheng, X. 2016. Predict anchor links across social networks via an embedding approach. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, 1823–1829.

Shang, Y.; Kang, Z.; Cao, Y.; Zhang, D.; Li, Y.; Li, Y.; and Liu, Y. 2019. PAAE: A unified framework for predicting anchor links with adversarial embedding. In *IEEE International Conference on Multimedia and Expo, ICME 2019, Shanghai, China, July 8-12, 2019*, 682–687.

Sun, M.; Zhao, Y.; and Liu, Z. 2015. Representation learning for measuring entity relatedness with rich information. In *International Conference on Artificial Intelligence*.

Tan, S.; Guan, Z.; Cai, D.; Qin, X.; Bu, J.; and Chen, C. 2014. Mapping users across networks by manifold alignment on hypergraph. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada.*, 159–165.

Tang, J.; Zhang, J.; Yao, L.; Li, J.; Zhang, L.; and Su, Z. 2008. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008*, 990–998.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, 5998–6008.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2017. Graph attention networks. *CoRR* abs/1710.10903.

Wang, Y.; Feng, C.; Ling, C.; Yin, H.; Guo, C.; and Chu, Y. 2018. User identity linkage across social networks via linked heterogeneous network embedding. *World Wide Web-internet & Web Information Systems* (2):1–22.

Wang, X.; Ji, H.; Shi, C.; Wang, B.; Ye, Y.; Cui, P.; and Yu, P. S. 2019. Heterogeneous graph attention network. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, 2022–2032.

Wang, X.; Zhang, Y.; and Shi, C. 2019. Hyperbolic heterogeneous information network embedding. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019.*, 5337–5344.

Xu, L.; Wei, X.; Cao, J.; and Yu, P. S. 2017. Embedding of embedding (EOE): joint embedding for coupled heterogeneous networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM 2017, Cambridge, United Kingdom, February 6-10, 2017*, 741–749.

Zhan, Q.; Zhang, J.; Yu, P.; and Xie, J. 2017. Community detection for emerging social networks. *World Wide Web-internet & Web Information Systems* 20(6):1409–1441.

Zhang, J., and Yu, P. S. 2015. Integrated anchor and social link predictions across social networks. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, 2125–2132.

Zhang, J.; Shi, X.; Xie, J.; Ma, H.; King, I.; and Yeung, D. 2018. Gaan: Gated attention networks for learning on large and spatiotemporal graphs. In *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence, UAI 2018, Monterey, California, USA, August 6-10, 2018*, 339–349.

Zhou, F.; Liu, L.; Zhang, K.; Trajcevski, G.; Wu, J.; and Zhong, T. 2018. Deeplink: A deep learning approach for user identity linkage. In *2018 IEEE Conference on Computer Communications, INFOCOM 2018, Honolulu, HI, USA, April 16-19, 2018*, 1313–1321.