The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Neural Link Prediction Over Aligned Networks

Xuezhi Cao, Haokun Chen, Xuejian Wang, Weinan Zhang,* Yong Yu
APEX Data & Knowledge Management Lab
Shanghai Jiao Tong University
cxz,chenhaokun,xjwang,wnzhang,yyu@apex.sjtu.edu.cn

Abstract

Link prediction is a fundamental problem with a wide range of applications in various domains, which predicts the links that are not yet observed or the links that may appear in the future. Most existing works in this field only focus on modeling a single network, while real-world networks are actually aligned with each other. Network alignments contain valuable additional information for understanding the networks, and provide a new direction for addressing data insufficiency and alleviating cold start problem. However, there are rare works leveraging network alignments for better link prediction. Besides, neural network is widely employed in various domains while its capability of capturing high-level patterns and correlations for link prediction problem has not been adequately researched yet. Hence, in this paper we target at link prediction over aligned networks using neural networks. The major challenge is the heterogeneousness of the considered networks, as the networks may have different characteristics, link purposes, etc. To overcome this, we propose a novel multi-neural-network framework MNN, where we have one individual neural network for each heterogeneous target or feature while the vertex representations are shared. We further discuss training methods for the multi-neural-network framework. Extensive experiments demonstrate that MNN outperforms the state-of-the-art methods and achieves 3% to 5% relative improvement of AUC score across different settings, particularly over 8% for cold start scenarios.

Introduction

Link prediction is a fundamental problem in the area of complex network and data mining (Lü and Zhou 2011). It helps conduct network completion for partially observed networks (Kim and Leskovec 2011) and understand how networks evolve across time (Barabâsi et al. 2002). As network data widely exists in various domains, link prediction contributes to numerous important applications. For example, we can employ link prediction to recommend friends in social networks (Aiello et al. 2012), explore gene expressions in biology networks (Almansoori et al. 2012), etc. Hence, link prediction draws plenty of research attention.

Almost all existing link prediction methods suffer from data insufficiency problems since they only focus on modeling the target network itself. One major limitation is the cold start problem, i.e. the performance is usually poor when a target vertex is rather new (Leroy, Cambazoglu, and Bonchi 2010). Interview process is proposed in order to alleviate this problem (Rashid et al. 2002). However, it can only be applied in certain scenarios thus has limited effect. Another limitation is the lack of comprehensiveness. Each network only reveals the partial information of the vertices, e.g. professional relations in LinkedIn, social ties in Facebook, movie preferences in IMDb, etc. We cannot tell if Alice actually likes the movie "Fast & Furious 8" or just because her boss likes it, using only the partial information revealed in movie rating network without knowing their professional relations. Therefore, researchers are still seeking out solutions to alleviate the data insufficiency problem.

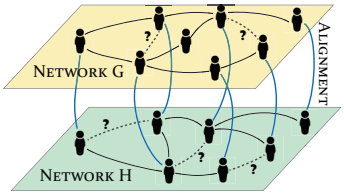

Figure 1: Link Prediction over Aligned Networks

A novel direction for data enrichment is to take advantage of aligned networks (illustrated in Figure 1). Networks are actually alignable in most scenarios. For example, online social networks can be naturally aligned through users, i.e. two accounts in different networks are held by a same user. Online platforms now encourage users to sign in with cross-platform accounts to directly form the alignments. Besides, there also arise literatures on automatically revealing the underlying alignments between social networks (Liu et al. 2014), biology networks (Neyshabur et al. 2013), device networks (Anand and Renov 2015), etc., providing prerequisite alignments for cross-network researches and analysis.

Network alignments provide valuable information for subsequent researches in various domains. Using aligned social networks, researchers propose joint user modeling to improve the quality of online personalized services (Cao and Yu 2016b). In biology, aligned protein-protein interaction networks contribute to the discovery of aging related com-

*Correspondence to Weinan Zhang.

plexes (Faisal, Zhao, and Milenković 2015). These works indicate that modeling the aligned networks can benefit subsequent tasks and applications greatly, hence may become the future trend in network studies.

Therefore, in this paper we target at link prediction over aligned networks. Specifically, given two aligned networks, our goal is to jointly model the two networks to improve the prediction quality for both networks (as shown in Figure 1). The rationale behind is that the aligned networks are consistent to some extent. For example, the social ties are consistent across different social networks (Liu et al. 2014). And for protein-protein interaction networks, there exist common interaction patterns across species (Liao et al. 2009).

Joint modeling over aligned networks is challenging due to heterogeneousness. The meanings of the links in different networks may differ as the links may have various purposes. Also, the link densities of the aligned networks may be unbalanced. Therefore, to directly merge the links and apply traditional link prediction methods may not make full use of the alignments hence does not lead to a satisfying solution. Besides, the networks contain various heterogeneous features including node degrees, distances, etc., which further increase the task's heterogeneousness.

Due to its recency, there are rare works in this direction. The only few works mainly aim at feature engineering using the alignments (Zhang, Yu, and Zhou 2014), or transferring vertices' attribute information across networks (Zhang et al. 2017). These works do not address the heterogeneousness well in aligned networks. Hence, there still exists plenty room for further improvement.

In this paper, we propose a multi-neural-network framework **MNN** for link prediction over aligned networks. We leverage neural network technique to take advantage of its great capacity of capturing high-level correlations. To deal with network heterogeneousness, we extend traditional neural network to a multi-neural-network framework. Within the framework, we consider each heterogeneous target or feature as an individual objective channel. We construct one neural network for each objective channel respectively, while all the neural networks share a common set of vertex embeddings. The rationale behind is that by only using the information enclosed in the vertex embeddings, we should be able to derive the objectives (features, link behaviors, etc.) despite their heterogeneousness, if the vertex embeddings are accurate and comprehensive enough. Each neural network can be considered as an individual worker that incorporates the information of its own target or feature into the common vertex embeddings. They together form the multi-neural-network framework to learn a comprehensive and accurate vertex representation for the aligned networks.

We conduct extensive experiments on two real-world datasets. Due to the limitation of available datasets, we only discuss alignment over two networks to demonstrate the effectiveness of leveraging aligned networks for link prediction task, while our model can be easily extended to model multiple aligned networks as well. Experimental results indicate that MNN achieves 3% to 5% relative improvement compared to the state-of-the-art method across different settings in terms of area under ROC curve (AUC). For cold start scenarios, MNN further yields 8% relative improvement.

Related Works

Link Prediction Link prediction draws plenty of attention due to its importance and wide usage in various domains. Most methods can be categorized into similarity-based methods and model-based methods (Lü and Zhou 2011; Wang et al. 2015). Similarity-based methods assign a similarity score for each pair of vertices and then conduct link prediction based purely on the scores. The widely used similarity-based methods include common neighbors (Lin and others 1998), Leicht-Holme-Newman index (Leicht, Holme, and Newman 2006), node rank based algorithms (Jeh and Widom 2002), etc. Model-based methods define a model based on assumptions of the network structures or linking mechanisms, and then fit the model with the observed network. The representative models include hierarchical graph based models (Clauset, Moore, and Newman 2008) and latent factor models (Menon and Elkan 2011).

Recently, researchers further extend the model-based methods by applying neural networks technique. DeepWalk considers vertices as words and random walk sequences as sentences, then applies word embedding techniques to compute vertex embedding (Perozzi, Al-Rfou, and Skiena 2014). Grover et al. followed similar direction and proposed biased random walk to generate the sequences (Grover and Leskovec 2016). Besides, SDNE further borrows the idea of auto-encoder and simultaneously model both first-order and second-order proximities (Wang, Cui, and Zhu 2016). These works indicate that leveraging neural networks for vertex embedding is promising.

Network Alignment Recently, aligning isolated networks and conducting multi-network analysis are proposed as a new research direction. Researchers successfully align the networks using learning techniques in various domains, including social networks (Zhang and Yu 2015), biological networks (Neyshabur et al. 2013), etc. Aligned networks are of great value for subsequent researches or tasks. E.g, finding evolution-related proteins using aligned protein-protein interaction networks in biology (Liao et al. 2009), conducting joint user modeling for better personalized services in online social networks (Cao and Yu 2016b), etc.

However, there are very limited works on leveraging aligned networks for improving the quality of link prediction. Zhang et al. proposed social meta-paths across networks based on the alignments (Zhang, Yu, and Zhou 2014). However, they only focused on feature engineering using the alignments and did not address the heterogeneousness of the aligned networks. Besides, there also exist work aiming at transferring the vertices' attribute information in aligned networks, including user generated content, user preferences, etc. (Zhang et al. 2017). We will compare these methods during our experiments.

Multi-Task Learning The intuition of using aligned network for further improvement is to some extent similar with the idea of multi-task learning, i.e. trying to model the common patterns or underlying consistencies using multiple heterogeneous behaviors or observations. Evgeniou et

al. conducted multi-task learning using a regularization approach (Evgeniou and Pontil 2004). Collective matrix factorization is later proposed where models of different tasks share common user/item representations (Singh and Gordon 2008). For neural networks, weight-sharing across networks is also proposed for natural language processing (Collobert and Weston 2008). However, there are no works employing the idea of multi-task learning to model aligned networks.

Problem Definition

Definition 1. *(Aligned Network) A pair of aligned networks is defined by* $S = (G, H, A)$ *where* $G = (V_G, E_G)$ *and* $H = (V_H, E_H)$ *are the two networks with* V, E *being the set of vertices and edges respectively,* $A \subset V_G \times V_H$ *is the alignment between* G, H *where* $(u, v) \in A$ *indicates vertices* $u \in V_G$ *and* $v \in V_H$ *are aligned.*

Definition 2. *(Link Prediction over Aligned Networks) Given a pair of aligned networks* $S = (G, H, A)$*, construct* $P_G(u, v)$ *and* $P_H(p, q)$ *to predict the probability of each unobserved link for network* G *and* H *respectively.*

For simplicity and generality, we target at undirected and unweighted network in this paper. Note that our model can also be applied to directed and weighted network by straightforward extensions. For notation, we also define $N_G(u), N_H(v)$ to be the set of neighbors of vertex u, v in network G, H respectively.

Essentially, the link prediction task is a binary classification task for the unobserved links. Hence, area under the ROC curve (AUC) is normally used as the evaluation metric (Hanley and McNeil 1982).

Methodology

The Framework

The major challenge of link prediction over aligned networks is heterogeneousness. As stated previously, aligned networks may differ in several aspects, including the purposes of the links, link density, etc. Besides, a network itself also contains various types of features, including node degree, common neighbor, specific local structures, etc., which further increase the heterogeneousness of this task. Most traditional approaches directly incorporate these features to train a classification model, based on the assumption that these features have equal contributions towards all vertices (Lü and Zhou 2011). As there is no theoretical or practical support for such assumption, the heterogeneousness problem has not been well addressed by existing works.

Hence, we propose a multi-neural-network framework **MNN** to mine the underlying consistencies beneath the heterogeneous behaviors or features. The rationale behind is that although we have heterogeneous features, they are all explicit vertex behaviors based on vertex characteristics (user preferences, protein properties, etc.). Thus, we should be able to derive these information purely based on the corresponding vertex representations, providing that the vertex representations are accurate and comprehensive enough. Following this idea, we tackle the task by finding a set of accurate and comprehensive vertex representations as well as the models to derive the aforementioned heterogeneous information using such representations. Specifically, we use vertex embeddings as the vertex representations and learn a neural network for each objective channel (target or feature), forming the multi-neural-network framework.

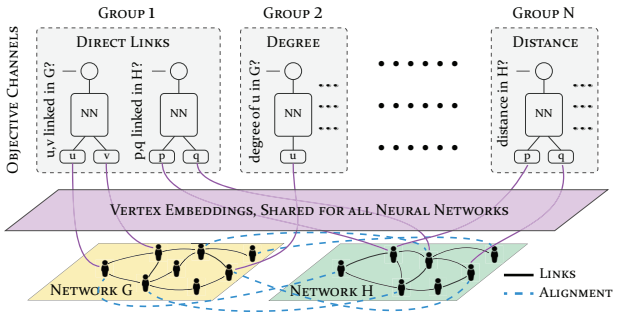

Figure 2: Multi-Neural-Network Framework

We illustrate our multi-neural-network framework in Figure 2. At the bottom we have two partially aligned networks G and H. Above them, we have a shared vertex embedding layer $E : V_G \cup V_H \to \mathbb{R}^k$, where k is the embedding dimension. The embeddings are shared among all subsequent neural networks. For aligned vertices, we assign them with a same embedding because their characteristics are essentially the same, i.e. $E(u) = E(v)$ for all $(u, v) \in A$. At the top, we have multiple neural networks targeting at various objective channels, including direct links, degrees, distances, etc. Each neural network corresponds to one single objective channel, with the purpose of refining the vertex embeddings using the information provided by its corresponding objective channel. Training these neural networks together results in an accurate and comprehensive vertex embeddings capturing all the heterogeneous information, along with the models to estimate these objectives.

In the following sections, we first introduce the objective channels we use, then present the detailed design of the neural networks. Finally, we discuss the training methods.

Objective Channels

Formally, each objective channel c is a fully or partially observable function $f_c(X, S)$, where X is a list of vertices and S is the aligned networks. For each objective channel, a neural network is designed to estimate the function based on the shared vertex embeddings E, i.e. $\hat{f}(X, E) \sim f(X, S)$. The purpose of each neural network is to incorporate the information provided by the objective f into the vertex embeddings E, and to acquire the model $\hat{f}$ for prediction.

To plug an objective channel c into MNN, we need to design the function $f_c(X, S)$ as well as generate the training samples. For all classification-based objective channels, we only discuss the generation of positive samples, and use random sampling to generate negative ones.

There are overall 7 groups of objective channels in our setting. As the channels are symmetric for both networks, we only discuss the channels with respect to network G.

Direct Links (DL). Direct links, as the target of link prediction task, provide the most important information. Specifically, we have objective channel DL_G for network G, where $DL_G(u,v) = 1$ if $(u,v) \in E_G$ and 0 otherwise. We use all observed links $(u,v) \in E_G$ as positive samples.

Multi-Step Links (ML). Besides direct links, we also include multi-step links to model the pairs of vertices that are close together but may not be directly linked. Specifically, we define $ML_G^d(u,v)$ to represent whether there exists a path of length d between vertices u and v in network G. To generate the positive samples, we conduct α random walks with length of d starting from each vertex, where α is the sampling rate. We use the starting and ending vertices of the random walks as positive samples, resulting in $\alpha \cdot |V_G|$ pairs. For experiments, we use ML with length 2 and 3.

Cross-Network Multi-Step Links (CL). To leverage the information of aligned networks, we extend the multi-step link channels to cross-network multi-step link channels. With this, we further capture the closeness of vertices according to the links from both networks. Similarly, we define $CL_G^d(u,v)$ to represent whether there exists a cross-path of length d between vertex u in G and vertex v in H, where cross-path is a path from a vertex in G to a vertex in H using an aligned vertex as the bridge. Formally, $(u_0, \cdots u_k, v_0, \cdots v_l)$ is a cross-path of length $k+l$ if and only if $u_i \in V_G, v_i \in V_H, (u_i, u_{i+1}) \in E_G, (v_i, v_{i+1}) \in E_H$ and $(u_k, v_0) \in A$. Similar with ML, we also use random walk to generate the training samples.

Vertex Distances (DT). Since multi-step link channels only consider paths of limited length, they are still focusing on local structures. To capture global structures between vertices, we further include vertex distances as another objective channel, i.e. $DT_G(u,v)$ is the distance between vertices u, v in network G. For training, we randomly sample $\alpha \cdot |V_G|$ pairs of vertices.

Neighborhood Jaccard (NJ). Another important feature to measure whether two vertices are close together is the portion of their neighbors that are shared. We employ Jaccard Similarity Coefficient to model it. Formally, we have:

$$NJ_G(u,v) = \frac{|N_G(u) \cap N_G(v)|}{|N_G(u) \cup N_G(v)|} \tag{1}$$

As we already have multi-step links with length 2 to represent whether the two vertices share a common neighbor, we now focus on (u,v) pairs with $NJ_G(u,v) > 0$. We sample the training pairs by sampling α pairs of $u, v \in N_G(w)$ for all $w \in V_G$, leading to $\alpha \cdot |V_G|$ samples.

Cross-Network Jaccard (CJ). We also extend Neighborhood Jaccard to cross-network Jaccard by considering the common neighbors according to the alignment. Specifically, for vertices in different networks, we count how many of their neighbors are aligned according to the alignment and then calculate the Jaccard Coefficient. Formally, for $u \in V_G$ and $v \in V_H$, $CJ_G(u,v)$ is defined by:

$$\begin{aligned} CN_G(u,v) &= \{(p,q) \in A | p \in N_G(u), q \in N_H(v)\} \\ CJ_G(u,v) &= \frac{|CN_G(u,v)|}{|N_G(u)| + |N_H(v)| - |CN_G(u,v)|} \end{aligned} \tag{2}$$

The sampling strategy is the same with NJ.

Graphlet Degree Signature (GP). Graphlet degree signature is a vector signature for each vertex representing its neighborhood structures (Milenkoviæ and Pržulj 2008), which is widely used in biological networks. The signature contains the count of different local structures around the target vertex, which is an extension of traditional vertex degrees. We include up to 4-node graphlets in our model (in total 15 structures, numbered from 0 to 14 in Figure 3). We use graphlet for all vertices to form the training set.

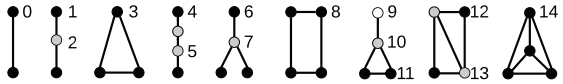

Figure 3: Graphlets up to 4-nodes used in GP.

To summarize, there are 7 groups of objective channels and 18 neural networks in total to form our framework MNN. The details are given in Table 1. These groups cover heterogeneous targets and features, indicating that MNN can be used as a general framework to leverage heterogeneous information. For unlisted features or additional dataset-dependent information, we can also easily include them in the framework following similar approaches.

Network Design

Despite the objective channels have different training targets, they all mine the underlying information or relations based on the vertex embeddings. From the viewpoint of model, the objective channels only differ with each other in the following aspects: (i) number of input vertices m, (ii) number of output units n, (iii) target type (classification or regression). Hence, we propose a unified neural network structure for all the objective channels.

In Figure 4, we depict an example of the neural network structure with 2 vertices as input ($m = 2$). For the input layer, we use one-hot encoding for the m input vertices $\{v_1, \cdots, v_m\}$ respectively. By multiplying each one-hot input by the common embedding matrix E, we form the embedding layer $\{E(v_1), \cdots, E(v_m)\}$. To model the direct interactions, we further insert a product layer proposed in product neural network (Qu et al. 2016). Specifically, it is the concatenation of $\{E(v_i)\}$ and $\{E(v_i) \circ E(v_j), i \neq j\}$, where $\circ$ indicates the element-wise product of the two embeddings (a.k.a. Hadamard product). Upon the product layer, we have multiple fully connected hidden layers with ReLU as the activation function to mine the high-level relations. Formally, for input vertices $X = \{v_i\}$, we have:

$$\begin{aligned} h_0(X) &= \text{concat}(\{E(v_i)\}, \{E(v_i) \circ E(v_{j \neq i})\}) \\ h_{i+1}(X) &= \text{relu}(h_i(X) W_i + B_i) \end{aligned} \tag{3}$$

where the last layer (l^{th} layer) serves as the raw output and has exact n units corresponding to the objective channel's setting, i.e. $o(X) = h_l(X) \in \mathbb{R}^n$.

For binary classification tasks, the raw output serves as the logits. Log likelihood is then used as the loss function.

$$\mathcal{L}_c = -\sum_{X,y} \sum_i y_i \ln(\sigma(o(X)_i)) + (1-y_i) \ln(1-\sigma(o(X)_i)) \tag{4}$$

Table 1: Summary of Objective Channel Groups used in MNN, with respect to network G

Objective Channel	#Nets	Input	Output	Type	Generation of (Positve) Training Samples
Direct Links (DL)	2	$u, v \in V_G$	$[0, 1]$	Classification	All $(u, v) \in E_G$
Multi Links (ML)	4	$u, v \in V_G$	$[0, 1]$	Classification	Sample by random walk
Cross Links (CL)	4	$u \in V_G, v \in V_H$	$[0, 1]$	Classification	Sample by random walk
Neighbor Jaccard (NJ)	2	$u, v \in V_G$	$\mathbb{R}$	Regression	Sample $u, v \in N_G(w)$ for $w \in V_G$
Cross Jaccard (CJ)	2	$u \in V_G, v \in V_H$	$\mathbb{R}$	Regression	Sample $u \in N_G(p), v \in N_H(q)$ for $(p, q) \in A$
Distance (DT)	2	$u, v \in V_G$	$\mathbb{R}$	Regression	Sample $(u, v) \in V_G \times V_G$
Graphlet (GP)	2	$u \in V_G$	$\mathbb{R}^{15}$	Regression	All $u \in V_G$

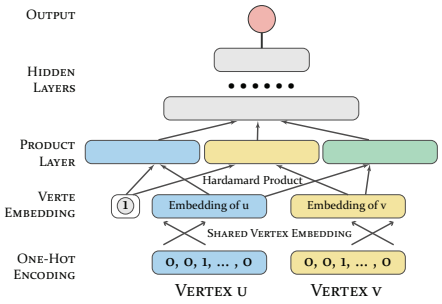

Figure 4: Neural Network Structure for Objective Channels

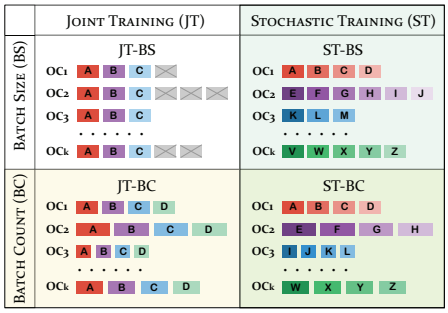

Figure 5: Training Methods

where $\sigma(x) = 1/(1 + exp(-x))$ is the sigmoid activation function and $y \in [0, 1]^n$ is the ground truth label.

For regression tasks, we directly use the raw output as the final estimation and employ square loss as the loss function.

$$\mathcal{L}_r = \sum_{X,y} \sum_i (o(X)_i - y_i)^2 \tag{5}$$

Training Methods

Our goal is to minimize the overall loss of all neural networks and the regularization term, formally $\mathcal{L}_{tot} = \sum_q w^q \mathcal{L}^q + \gamma \sum_\Theta \|\Theta\|_2$, where $\mathcal{L}^q$ and w^q denotes the loss function and the weight for q^{th} objective channel, Θ denotes the parameters. For the weighting, we set $w_q = \beta$ for the target objective channels (direct links) and $w_q = 1-\beta$ for other channels, where β is the weighting parameter to balance the targets and the features.

There are two ways to perform the training: jointly updating all the networks together or updating them individually and stochastically. Specifically, for joint training we use the training data of all objective channels together to optimize the overall loss $\mathcal{L}_{tot}$. For stochastic training, we update each neural network separately by optimizing $w^q \mathcal{L}^q + \gamma \|\Theta_q\|$ using its own training data. Their difference is analogous to the difference between full-batch gradient decent and stochastic gradient decent, which is whether to conduct the partition over training data. The benefit of joint updating is its theoretical optimality. However, it consumes larger computational resource. The advantage of stochastic training is that it naturally supports distributed computing and online extension by adding additional objective channels.

Since we have different numbers of training samples for each objective channel, how to divide the training batches may also lead to different results. If we fix the batch size for all channels, those channels with more training samples will have more batches, hence draw more attention in the resulting model. The other approach is to divide the training data for each channel into fixed number of batches B.

By combining joint training and stochastic training with the two batching strategies respectively, we have four different training methods. We illustrate them in Figure 5. Within each sub figure, each row corresponds to one objective channel while the colored blocks indicate partitioned training samples. Samples with the same color form one training batch. As the training methods have different characteristics, we will conduct experiments to explore their performances.

Despite the training methods, the time complexity is linear to the number of training samples. For the target channel (DL), we have $|E_G| + |E_H|$ positive samples. For vertex-based channels, we have one training sample for each vertex, hence $|V_G| + |E_H|$ samples. For link-based channels, we have $\alpha \cdot (|V_G| + |E_H|)$ positive samples for each channel according to our sampling strategies. The random sampling for negative samples also generate the same scale of samples with the positive ones. Hence, the time complexity of our framework is $O(N + M)$ in total, where N, M denote the numbers of vertices and links respectively.

Experiments

Datasets We conduct experiments using two sets of aligned social networks, provided by (Cao and Yu 2016a).

- **Facebook-Twitter** Facebook and Twitter are the most popular world-wide online social network and microblog

Table 2: Performance Comparison in General Cases, AUC as Metric (* indicates p-value less than $1e^{-6}$ for significant test)

Dataset	Method	Training Ratio			
		80%	60%	40%	20%
Facebook ↕ Twitter	CN	91.72% / 90.91%	89.99% / 88.38%	69.86% / 67.18%	69.86% / 67.18%
	DeepWalk	92.77% / 90.60%	92.32% / 90.02%	90.74% / 88.34%	87.03% / 82.95%
	LINE	92.19% / 90.29%	91.11% / 90.16%	90.61% / 89.30%	82.28% / 83.77%
	SDNE	92.16% / 90.15%	90.92% / 89.59%	87.54% / 77.31%	73.78% / 72.06%
	node2vec	93.29% / 91.68%	92.07% / 90.95%	90.52% / 89.70%	85.09% / 84.36%
	MLI	94.16% / 92.04%	91.59% / 90.76%	87.71% / 87.48%	77.73% / 75.17%
	SLAM	93.39% / 92.29%	91.39% / 91.11%	88.22% / 89.50%	82.09% / 86.55%
	MNN	**96.96% / 95.40%**	**96.32% / 94.88%**	**94.73% / 93.81%**	**92.62% / 91.15%**
	Improve	+2.97%* / +3.37%*	+4.33%* / +4.14%*	+4.40%* / +4.58%*	+6.42%* / +5.31%*
Weibo ↕ Douban	CN	84.32% / 86.69%	82.11% / 83.98%	77.37% / 78.50%	66.61% / 66.27%
	DeepWalk	86.06% / 85.81%	84.78% / 83.70%	83.05% / 81.58%	76.11% / 74.09%
	LINE	86.94% / 86.30%	85.18% / 85.69%	83.74% / 85.36%	82.36% / 84.14%
	SDNE	86.07% / 89.98%	85.41% / 87.80%	84.22% / 83.55%	81.52% / 78.62%
	node2vec	91.59% / 92.87%	89.72% / 91.76%	87.86% / 88.99%	83.81% / 83.86%
	MLI	90.04% / 91.55%	88.68% / 90.68%	86.65% / 88.51%	79.85% / 83.61%
	SLAM	90.31% / 91.28%	89.01% / 90.23%	88.18% / 88.90%	86.37% / 86.34%
	MNN	**96.46% / 97.14%**	**94.59% / 96.70%**	**94.04% / 96.18%**	**92.39% / 94.37%**
	Improve	+5.32%* / +4.60%*	+5.23%* / +5.388%*	+6.65%* / +8.08%*	+6.97%* / +9.30%*

site. We have 4,137 active users with average degree of 13.91 and 35.71 in the networks in this set.

- **Weibo-Douban** Weibo and Douban are Chinese largest microblog and movie rating site respectively. For this set, we have 50,552 vertices with average degree of 45.47 and 52.52 respectively.

Comparing Methods

- **Common Neighbor (CN)**: Similarity-based method using the number of common neighbors as similarity function (Liben-Nowell and Kleinberg 2007).
- **DeepWalk**: Skip-gram based vertex embedding method which considers vertices as words and random walks as sentences (Perozzi, Al-Rfou, and Skiena 2014).
- **LINE**: Network embedding method with objective function that preserves both the first-order and second-order proximities (Tang et al. 2015).
- **SDNE**: Auto-encoder based vertex embedding method (Wang, Cui, and Zhu 2016).
- **node2vec**: Word-to-vector approach with biased random walk (Grover and Leskovec 2016).
- **MLI**: Link prediction method for aligned networks using meta-path as features (Zhang, Yu, and Zhou 2014).
- **SLAM**: Link prediction for aligned networks using sparse and low rank matrix estimation (Zhang et al. 2017).

For fair comparison, we also extend the single-net link prediction methods to using the alignment information. We apply them to both the original isolated networks and the merged network (union of the links based on the alignment), then report the best performances. The parameters of each model are tuned separately for best performances.

Parameters For our multi-neural-network model (MNN), we set the embedding dimension $k = 80$, sampling rate $\alpha = 100$, weighting parameter $\beta = 0.5$ and the regularization term $\gamma = 0.1$. We design each neural network to have 2 hidden layers between the product layer and output layer, with width of 100 and 50 respectively. The source code as well as the datasets are available online[1].

Evaluation Metric As link prediction is essentially a binary classification task, we use Area Under the Curve (AUC) as the evaluation metric. 80% links are used for training.

General Case We list the experimental results for all comparing methods in Table 2. In all settings, our framework MNN achieves the best performance, with p-value $< 1e^{-6}$ for significance test (Hanley and McNeil 1982). When training ratio is 80%, we achieve relative improvements of 2.97% to 5.32% in different settings compared to the best existing method, indicating that MNN can leverage the heterogeneous objective channels for a comprehensive vertex embeddings over aligned networks. Comparing experiments with different training ratios, the performances of all methods decrease as the training ratio drops. Compared to existing methods, the performance drop of our model MNN is rather moderate, i.e. MNN is more robust to data insufficiency.

Cold Start Scenarios We also expect MNN to alleviate cold start problem. We depict the performances for vertices with different degrees in Figure 6. As expected, all approaches encounter a performance drop when dealing with cold start scenarios where the target vertex's degree is limited. The results indicate that our model is the most robust one to data insufficiency. For vertices with degree equal to 2, we achieve a relative improvement of 8.53%.

[1] http://apex.sjtu.edu.cn/projects/34

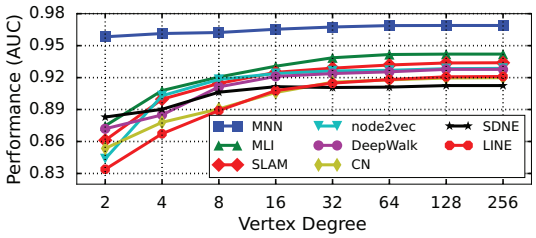

Figure 6: Performance on Cold Start Scenarios

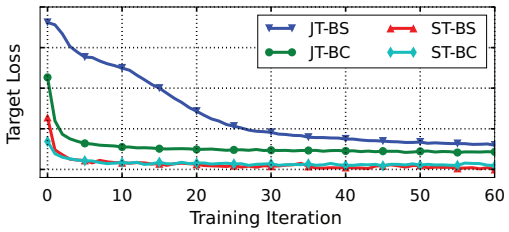

Figure 7: Training Curves with Different Training Methods

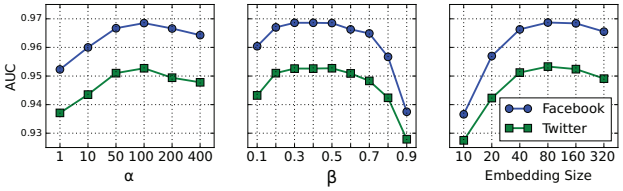

Figure 8: Hyperparameter Analysis

Evaluating Objective Channels Now we evaluate the contribution of each objective channel. In Table 3, we show the performances gained by adding additional objective channels. The results indicate that the channels we propose do have positive effect on the performances. It also proves that MNN can successfully leverage all these heterogeneous features for comprehensive vertex embeddings.

Table 3: Contribution of Objective Channel Groups

Objective Channels	Networks			
	Facebook	Twitter	Weibo	Douban
DL (Basic)	94.82%	93.43%	93.69%	95.84%
DL+ML	95.95%	94.60%	95.17%	96.42%
DL+CL	95.55%	94.20%	94.35%	96.30%
DL+DT	95.40%	94.68%	94.15%	96.23%
DL+GP	95.20%	93.71%	94.12%	96.05%
DL+NJ	95.58%	94.28%	94.62%	96.29%
DL+CJ	95.76%	94.26%	94.46%	96.34%
ALL	**96.96%**	**95.40%**	**96.46%**	**97.14%**

Training Methods Recall that we have 4 different training methods for MNN (Figure 5). We show their performances in Table 4. The results indicate that stochastic training achieves better performances compared to joint training, while fixed batch count methods outperforms fixed batch size methods. Since fixed batch count and stochastic training give each objective channel an equal chance to tune the embeddings, it may better capture heterogeneous information instead of focusing on only the strongest ones.

Table 4: Comparing Training Methodologies

Training Method	Networks			
	Facebook	Twitter	Weibo	Douban
JT-BS	94.59%	92.93%	94.17%	96.07%
JT-BC	95.11%	93.78%	94.19%	96.22%
ST-BS	95.92%	94.89%	95.19%	96.84%
ST-BC	**96.24%**	**95.02%**	**96.46%**	**97.14%**

We also plot the training curves in Figure 7, which indicate that stochastic training also achieves faster convergence. As stochastic training can be naturally conducted in parallel or distributed computing, it also has a great advantage in time-complexity aspect compared to joint training.

Hyperparameter Analysis Now we examine the influence of three key hyperparameters in our framework: the sampling rate α, the weighting parameter β and the embedding size. The results in Figure 8 indicate that setting sampling rate α to 100 provides the best performance. For weighting parameter β, the best setting is in $[0.3, 0.5]$, which matches our intuition that we need to balance between the target objective channel group (direct links) and the other objective channel groups. For embedding size, $k = 80$ provides the best performances.

Conclusion

In this paper, we target at link prediction over multiple aligned networks. The aligned networks provide valuable additional information for modeling the structures and understanding the networks. To address the heterogeneousness of the aligned networks, we propose a multi-neural-network framework MNN. Specifically, we consider each heterogeneous learning target as well as network-based features as an individual objective channel, and then construct one neural network for each objective channel respectively. All the neural networks jointly learn a shared set of vertex embeddings, hence leverage the heterogeneous information to achieve accurate and comprehensive vertex representations. Experiments with two real-world datasets indicate that our framework outperforms state-of-the-art link prediction methods for both single-network and aligned-network, especially in cold start scenarios.

We essentially propose a novel method of using multiple neural networks to handle heterogeneous features. The MNN framework may also contribute to other tasks besides link prediction, which we leave as future works.

Acknowledgement

This work is financially supported by NSFC (61772333, 61702327) and Shanghai Sailing Program (17YF1428200).

References

Aiello, L. M.; Barrat, A.; Schifanella, R.; Cattuto, C.; Markines, B.; and Menczer, F. 2012. Friendship prediction and homophily in social media. *TWEB* 6(2):9.

Almansoori, W.; Gao, S.; Jarada, T. N.; Elsheikh, A. M.; Murshed, A. N.; Jida, J.; Alhajj, R.; and Rokne, J. 2012. Link prediction and classification in social networks and its application in healthcare and systems biology. *Network Modeling Analysis in Health Informatics and Bioinformatics* 1(1-2):27–36.

Anand, T. R., and Renov, O. 2015. Machine learning approach to identify users across their digital devices. In *ICDMW*, 1676–1680. IEEE.

Barabâsi, A.-L.; Jeong, H.; Néda, Z.; Ravasz, E.; Schubert, A.; and Vicsek, T. 2002. Evolution of the social network of scientific collaborations. *Physica A: Statistical mechanics and its applications* 311(3):590–614.

Cao, X., and Yu, Y. 2016a. Asnets: A benchmark dataset of aligned social networks for cross-platform user modeling. In *CIKM*, 1881–1884. ACM.

Cao, X., and Yu, Y. 2016b. Joint user modeling across aligned heterogeneous sites. In *RecSys*, 83–90. ACM.

Clauset, A.; Moore, C.; and Newman, M. E. 2008. Hierarchical structure and the prediction of missing links in networks. *Nature* 453(7191):98–101.

Collobert, R., and Weston, J. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, 160–167. ACM.

Evgeniou, T., and Pontil, M. 2004. Regularized multi–task learning. In *SIGKDD*, 109–117. ACM.

Faisal, F. E.; Zhao, H.; and Milenković, T. 2015. Global network alignment in the context of aging. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 12(1):40–52.

Grover, A., and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *SIGKDD*, 855–864. ACM.

Hanley, J. A., and McNeil, B. J. 1982. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology* 143(1):29–36.

Jeh, G., and Widom, J. 2002. Simrank: a measure of structural-context similarity. In *SIGKDD*, 538–543. ACM.

Kim, M., and Leskovec, J. 2011. The network completion problem: Inferring missing nodes and edges in networks. In *Proceedings of the 2011 SIAM International Conference on Data Mining*, 47–58. SIAM.

Leicht, E. A.; Holme, P.; and Newman, M. E. 2006. Vertex similarity in networks. *Physical Review E* 73(2):026120.

Leroy, V.; Cambazoglu, B. B.; and Bonchi, F. 2010. Cold start link prediction. In *SIGKDD*, 393–402. ACM.

Liao, C.-S.; Lu, K.; Baym, M.; Singh, R.; and Berger, B. 2009. Isorankn: spectral methods for global alignment of multiple protein networks. *Bioinformatics* 25(12):i253–i258.

Liben-Nowell, D., and Kleinberg, J. 2007. The link-prediction problem for social networks. *journal of the Association for Information Science and Technology* 58(7):1019–1031.

Lin, D., et al. 1998. An information-theoretic definition of similarity. In *ICML*, volume 98, 296–304. Citeseer.

Liu, S.; Wang, S.; Zhu, F.; Zhang, J.; and Krishnan, R. 2014. Hydra: Large-scale social identity linkage via heterogeneous behavior modeling. In *SIGMOD*, 51–62. ACM.

Lü, L., and Zhou, T. 2011. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications* 390(6):1150–1170.

Menon, A., and Elkan, C. 2011. Link prediction via matrix factorization. *Machine Learning and Knowledge Discovery in Databases* 437–452.

Milenkoviæ, T., and Pržulj, N. 2008. Uncovering biological network function via graphlet degree signatures. *Cancer informatics* 6:257.

Neyshabur, B.; Khadem, A.; Hashemifar, S.; and Arab, S. S. 2013. Netal: a new graph-based method for global alignment of protein–protein interaction networks. *Bioinformatics* 29(13):1654–1662.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *SIGKDD*, 701–710. ACM.

Qu, Y.; Cai, H.; Ren, K.; Zhang, W.; Yu, Y.; Wen, Y.; and Wang, J. 2016. Product-based neural networks for user response prediction. *arXiv preprint arXiv:1611.00144*.

Rashid, A. M.; Albert, I.; Cosley, D.; Lam, S. K.; McNee, S. M.; Konstan, J. A.; and Riedl, J. 2002. Getting to know you: learning new user preferences in recommender systems. In *IUI*, 127–134. ACM.

Singh, A. P., and Gordon, G. J. 2008. Relational learning via collective matrix factorization. In *SIGKDD*, 650–658. ACM.

Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; and Mei, Q. 2015. Line: Large-scale information network embedding. In *WWW*, 1067–1077. International World Wide Web Conferences Steering Committee.

Wang, P.; Xu, B.; Wu, Y.; and Zhou, X. 2015. Link prediction in social networks: the state-of-the-art. *Science China Information Sciences* 58(1):1–38.

Wang, D.; Cui, P.; and Zhu, W. 2016. Structural deep network embedding. In *SIGKDD*, 1225–1234. ACM.

Zhang, J., and Yu, P. S. 2015. Multiple anonymized social networks alignment. In *ICDM*.

Zhang, J.; Chen, J.; Zhi, S.; Chang, Y.; Philip, S. Y.; and Han, J. 2017. Link prediction across aligned networks with sparse and low rank matrix estimation. In *ICDE*, 971–982. IEEE.

Zhang, J.; Yu, P. S.; and Zhou, Z.-H. 2014. Meta-path based multi-network collective link prediction. In *SIGKDD*, 1286–1295. ACM.