Neural Link Prediction over Aligned Networks

Anonymous Authors
Department
Affiliation
Contact

Abstract

Link prediction is a fundamental problem with a wide range of applications in various domains, which predicts the links that may appear in the future. Most existing works in this direction only focus on modeling single network, while real-world networks are actually aligned with each other. Network alignments contain valuable additional information for understanding the networks, which provide a new direction for data enrichment and alleviating cold start problem. However, there are rarely works leverage network alignments for better link prediction. Besides, neural network is widely employed in various domains while performing 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 of this task is the heterogeneousness of the considered networks, as they 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 learning methods to simultaneously train the multiple neural networks. Experiments using two real-world datasets indicate that MNN outperforms all comparing methods and achieves approximately 3% to 5% relative improvement of AUC across different settings, particularly over 8% for cold start scenarios.

Introduction

Link prediction is a fundamental problem in 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 focus only on modeling the target network itself thus often suffer from data insufficiency. One major limitation is the cold start problem, i.e. the quality is usually poor when the target network or vertex is rather new (Leroy, Cambazoglu, and Bonchi 2010). Interview process is proposed in order to alleviate this problem (Rashid et al. 2002). However, this can be only applied in certain scenarios thus has limited effect. Another limitation is the lack of comprehensiveness. Each network only reveals partial information of the vertexes. 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.

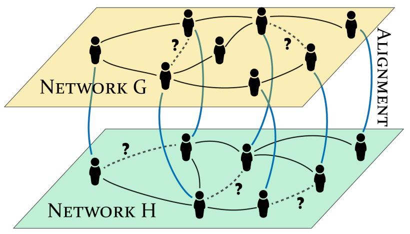

Figure 1: Link Prediction over Aligned Networks

A novel direction for data enrichment is to take advantage of the 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. whether two accounts are held by the same user. Online platforms now encourage users to sign in with cross-platform accounts to directly form the alignment. Besides, there also arise research works on automatically revealing the underlying alignment between social networks (Liu et al. 2014), biology networks (Neyshabur et al. 2013), device networks (Anand and Renov 2015), etc. Hence, using aligned networks for data enrichment is a promising direction.

Subsequent research in multiple domains indicate that network alignments do provide valuable additional information. Using the aligned social networks, researchers propose joint user modeling to improve the quality of online personalized services (Cao and Yu 2016b). In biology, the aligned protein-protein interaction networks contribute the discovery of aging related complexes (Faisal, Zhao, and Milenković 2015). These works indicate that modeling the aligned networks can benefit subsequent tasks and 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 its 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 may not make full use of the alignment hence does not lead to a satisfying solution. Besides, the network itself contains various heterogeneous features including node degrees, distances, local structures etc., which further increase the task's heterogeneousness.

Due to its recency, there are rarely works in this direction. The only works mainly aim at feature engineering using the alignment and does not address the heterogeneous problem in aligned networks (Zhang, Yu, and Zhou 2014). 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 as well as features 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 intuition behind is that despite the heterogeneousness, we should be able to derive the features, link behaviors, etc. by using only the information enclosed in the vertex embeddings, if the vertex embeddings are accurate and comprehensive enough. Each neural network can be considered as an individual worker that tries to incorporate the information from 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 further discuss several learning methods for the joint learning of multiple neural networks.

To demonstrate the performance of our framework MNN, we conduct extensive experiments using two real-world datasets. Experimental results indicate that MNN achieves approximately 3% to 5% relative improvement comparing to best existing 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 vertexes and then conduct link prediction based purely on the resulting scores. The widely used similarities 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 structure or linking mechanisms, and then fit the model with the observed network. The representative models include hierarchical graph organization (Clauset, Moore, and Newman 2008) and latent factor models (Menon and Elkan 2011).

Recently, researchers further extent the model-based methods toward using neural network techniques. DeepWalk considers vertexes 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. follow similar direction and proposed biased random walk to generate the sequences (Grover and Leskovec 2016). Besides using word embedding techniques, SDNE borrows the idea of auto-encoder (Wang, Cui, and Zhu 2016). These works indicate that leveraging neural network 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 (Liu et al. 2014), biological networks (Neyshabur et al. 2013), etc. The aligned networks are of great value for subsequent researches or tasks. E.g, finding evolutionary-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 only 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 alignment (Zhang, Yu, and Zhou 2014). However, they only focused on feature engineering using the alignment and does not address the heterogeneousness of the aligned networks. We compare this method during experiment. Besides, there is also work aiming at transferring the vertexes' attribute information in aligned networks (Zhang et al. 2017), which has different focus with this paper hence not compared during experiment.

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. try to model the common patterns or underlying consistencies using multiple heterogeneous behaviors or observations. Evgeniou et al. conduct 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 the aligned networks.

Problem Definition

In this section, we formally define the aligned networks and the corresponding link prediction task.

Definition 1. *(Network) A network, or graph, is denoted as $G = (V, E)$ where $V = \{v_1, v_2, \cdots, v_n\}$ represents n vertexes and $E \subset V \times V$ represents the edges, where $(u, v) \in E$ indicates there exists an edge between vertexes u and v.*

Definition 2. *(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, $A \subset V_G \times V_H$ is the alignment between G, H where $(u, v) \in A$ indicates vertexes $u \in V_G$ and $v \in V_H$ are aligned.*

Definition 3. *(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 or 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 the network G, H respectively.

The link prediction task is actually a binary classification task for the unobserved links $V_G \times V_G - E_G$ and $V_H \times V_H - E_H$. Hence, area under the ROC curve (AUC) is normally used as the evaluation metric.

Our Approach

The Framework

The major challenge for link prediction over aligned networks is its heterogeneousness. As stated previously, the aligned networks may differ on several aspects, including the meaning of the links, link density, etc. Besides, network itself also contains various types of features, including node degrees, common neighbors, specific local structures, etc., which further increase the difficulty of the task. Traditional approaches always directly plug in these features to a classification model, based on the assumption that features have equal contributions towards all the vertexes (Lü and Zhou 2011), which, however, is not always the case.

We propose a multi-neural-network framework **MNN** to mine the underlying consistencies hidden beneath the heterogeneous behaviors or features. The intuition is that although we have heterogeneous links, different types of features, various local structures, etc., they are all explicit vertex behaviors based on their characteristics (user preferences, protein properties, etc.). Hence, 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 target at finding a set of accurate and comprehensive vertex representations as well as the methodologies 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.

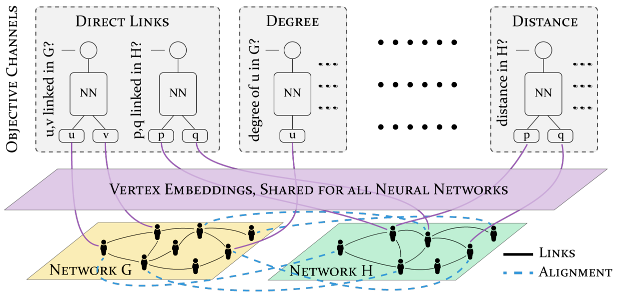

Figure 2: Multi-Neural-Network Framework

We illustrate our multi-neural-network framework for link prediction over aligned networks in Figure 2. At the bottom we have two partially aligned networks G and H. Above them, we have a common vertex embedding layer $E : V_G \cup V_H \rightarrow \mathbb{R}^k$, which is shared among all subsequent neural networks, where k is the embedding dimension. For the aligned vertexes, we assign them with same embedding because they are essentially the same, i.e. $E(u) = E(v)$ for all $(u, v) \in A$. On the top, we have multiple neural networks targeting at various objective channels, including direct links (the target), degrees, distances, etc. Each neural network corresponds to one single objective channel, with the purpose of refining the vertex embeddings based on the supervised information provided by the objective channel. Hence, learning these neural networks together results in a comprehensive representations of the vertexes using all these heterogeneous informations, along with the methodologies to predict these objective channels for unobserved data.

In the following sections, we first introduce the objective channels we use, then give the detailed design of the neural networks. Finally, we discuss the learning 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 vertexes and S is the aligned networks. For each objective channel, a neural network is designed and trained to estimate the function based on the common vertex embedding E, i.e. $\hat{f}(X, E) \sim f(X, S)$. The purpose of the neural network is to incorporate the information provided by the objective channel f into the vertex embedding E, and to acquire the neural network model $\hat{f}$ for estimating such information.

To plug an objective channel into MNN, we need to design the observable function $f_c(X, S)$ and construct the corresponding training samples. For all classification objective channels, we only discuss the generation of positive samples, and use random sampling for the negative ones since our evaluation metric is AUC, which is equivalent to the expected probability of correctly ranking a randomly selected pair of positive and negative samples (Bradley 1997).

Now we introduce the objective channels we use. As the channels are symmetric for both networks, we only discuss with respect to network G.

Direct Links (DL). Without any doubts, direct links, as the target of link prediction task, provide the most important information. Specifically, we have two objective channels DL_G and DL_H for the two networks, where $DL_G(u,v) = 1$ if $(u,v) \in E_G$ and 0 otherwise. For training, we use all observed links $(u,v) \in E_G$ as the positive samples.

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

Cross-Network Multi-Step Links (CL). To leverage the aligned network for further improvement, we extend the multi-step links to cross-network multi-step links. Similarly, we define $CL_G^d(u,v)$ to judge 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 vertex in G to 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 $(u_k, v_0) \in A$. We also use random walk to generate the training samples as the same with ML.

Vertex Distances (DT). Because for multi-step links we can only consider limited length paths, they still focus on local structures. For global structure, we further include vertex distances as another objective channel, i.e. $DT_G(u,v)$ is the distance between vertexes u, v according to G. For training, we randomly sample $\alpha \cdot |V_G|$ pairs of vertexes.

Neighborhood Jaccard (NJ). An important feature to measure whether two vertexes are close together is the ratio of their common neighbors. 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 judge whether the two vertexes share a common neighbor, we now focus on estimating for (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 vertexes in different networks, we count how many of their neighbors are aligned together 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 similar 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 for biological networks. The signature includes the count of different local structures around the target vertex. We include up to 4-node graphlets in our model (in total 15 structures, depicted in Figure 3). We use all vertexes to form the training set.

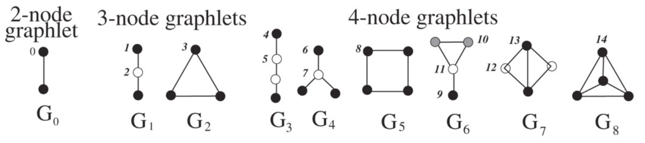

Figure 3: Graphlets up to 4-nodes used in GP.

To summarize, we have 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 for leveraging heterogeneous information. For other unlisted features or additional dataset-dependent information, we can also easily leverage them into the framework following similar methodologies.

Network Design

Despite the objective channels have different training targets, they all mine the required information or relations based on the vertex embeddings. From the viewpoint of the model, the objective channels only differ with each other in following aspects: (i) number of input vertexes 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.

We depict an example of the neural network structure with 2 vertexes as input ($m = 2$) in Figure 4. For the input layer, we use one-hot encoding for the m input vertexes $\{v_1, \cdots, v_m\}$ respectively. Multiplied 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 among the vertexes, we further insert a product layer proposed in product neural network (PNN) (Qu et al. 2016). Specifically, the product layer is the concatenation of $\{E(v_i)\}$ and $\{E(v_i) \circ E(v_j), i \neq j\}$, where $\circ$

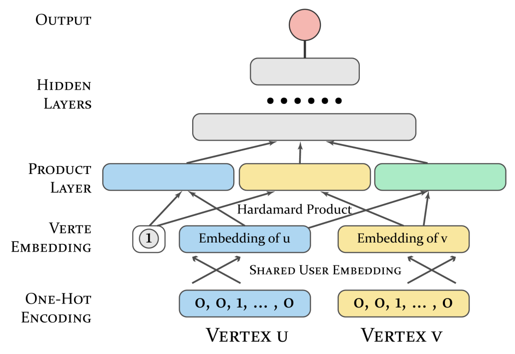

Figure 4: Neural Network Structure for Objective Channels

Table 1: Summary of Objective Channels used in LPAN, 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 $u, v \in N_G(w)$ for $w \in V_G$
Cross Links (CL)	4	$u \in V_G, v \in V_H$	$[0, 1]$	Classification	Sample $u \in N_G(p), v \in N_H(q)$ for $(p, q) \in A$
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 $u \in V_G$

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 activation function to mine the high-level relations. Formally, with input vertexes $X = \{v_i\}$, we have:

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

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.

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

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 direct use the raw output as the final estimation and employ square loss as the loss function.

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

Learning 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 is the parameters. For the weighting, we set $w_q = \beta$ for the target objective channel (direct link) and $w_q = 1 - \beta$ for the other channels, where β is the weighting parameter to adjust the balance between the targets and features.

There are two ways to carry out the training: jointly update all the networks together or update them one by one stochastically. Specifically, for joint learning we feed the training data for all objective channels together and optimize for the overall loss $\mathcal{L}_{tot}$. And for stochastic learning, we update each neural network separately by optimizing $w^q \mathcal{L}^q + \gamma \|\Theta_q\|$ using its own training data. Analogous to full-batch gradient decent and the stochastic gradient decent that differs on whether to conduct the partition over training data, here we partition over the multi-neural-networks. The benefit of joint updating is its theoretical optimality. However, it consumes larger computational resource. The advantage of stochastic learning is that it naturally supports distributed computing and online extension (adding additional objective channels).

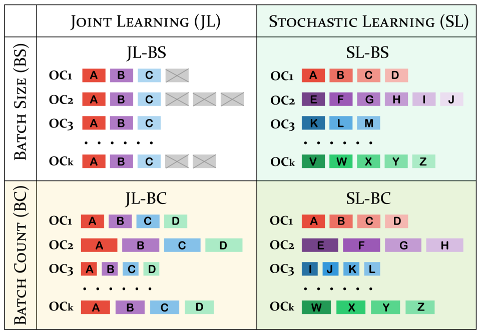

Figure 5: Learning Methodologies

Because we have different number of training samples for each objective channel, how to divide the training batches may also lead to different learning behaviors. If we fix the batch size for all objective channels, the ones with more training samples will have more training batches. Hence, the resulting model would focus more on these objective channels. Besides using fixed batch size, we may also divide the training data for each objective channel into fixed number of batches B, with each batch having $1/B$ training samples.

By combining joint learning and stochastic learning with the two batching strategies respectively, we have four different learning methodologies. We give an illustration of the four learning methodologies in Figure 5. Within each sub figure, each row corresponds to one objective channel while the colored blocks indicate partitioned training samples. Data samples with same color form one training batch.

Despite the learning method, 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 our sampling strategies. The random sampling for negative samples also generate same scale of samples with the positive ones. Hence, in total the time complexity of our framework is $O(N + M)$, where N, M indicate the number of vertexes and links respectively.

Table 2: Performance Comparison in General Scenarios with Different Training Ratios, AUC as Metric

Dataset	Method	Training Ratio			
		80%	60%	40%	20%
Facebook $\updownarrow$ 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%
	SDNE	92.16% / 90.15%	90.92% / 89.59%	87.54% / 77.31%	73.78% / 72.06%
	LINE	88.87% / 89.55%	88.78% / 89.06%	85.44% / 86.92%	72.21% / 77.83%
	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%
	MNN	**96.96% / 95.40%**	**96.32% / 94.88%**	**94.73% / 93.81%**	**92.62% / 91.15%**
	Improve	+2.97% / +3.65%	+4.33% / +4.32%	+4.40% / +4.58%	+6.42% / +9.21%
Weibo $\updownarrow$ 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%
	SDNE	86.07% / 89.98%	85.41% / 87.80%	84.22% / 83.55%	81.52% / 78.62%
	LINE	xx.xx% / xx.xx%	xx.xx% / xx.xx%	xx.xx% / xx.xx%	xx.xx% / xx.xx%
	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%
	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%	+7.03% / +8.08%	+10.24% / +12.53%

Experiments

Datasets The experiments are carried out using two sets of aligned online social networks covering both English and Chinese populations, provided by (Cao and Yu 2016a).

- **Facebook-Twitter** Facebook and Twitter are the most popular world-wide online social network and microblog 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 sites respectively. For this set, we have 50,552 vertexes with average degree of 45.47 and 52.52 respectively.

Comparing Methods

- **Common Neighbor (CN)**: Similarity-based method using common neighbor as similarity function (Liben-Nowell and Kleinberg 2007).
- **LINE**: Network embedding method with objective function that preserves both the first-order and second-order proximities (Tang et al. 2015).
- **DeepWalk**: It considers vertexes as words and random walk as sentences, then apply skip gram technique for vertex embeddings (Perozzi, Al-Rfou, and Skiena 2014).
- **node2vec**: Word-to-vector approach with biased random walk (Grover and Leskovec 2016).
- **SDNE**: Auto-encoder based vertex embedding method (Wang, Cui, and Zhu 2016).
- **MLI**: Link prediction for aligned networks using meta-path as features (Zhang, Yu, and Zhou 2014).

For fair comparison, we also extend the single-net link prediction methods to use the alignment information. We apply them to both the original isolated network 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 ratio $\alpha = 100$, weighting parameter $\beta = 0.5$ and the regularization term $\gamma = 0.1$. Each neural network is designed 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 actually a binary classification task for the unobserved links, we use Area Under the Curve (AUC) as the evaluation metric. The score can also be interpretative as the probability that a randomly chosen positive example receives higher prediction score than a randomly chosen negative one (Hanley and McNeil 1982). 80% links are used for training unless indicated otherwise.

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 comparing to best existing method, indicating that MNN can leverage the heterogeneous objectives for a comprehensive network modeling.

Comparing the experiments using different training ratios, the performances of all methods decrease as the training ratio drops. Comparing to the existing methods, the performance drop of our model MNN is rather moderate, i.e. MNN is more robust to data insufficiency.

Cold Start Scenarios Besides improving under general settings, we also expect MNN to leverage the alignment information to alleviate cold start problem. We depict the performances for vertexes with different degrees in Figure 6. Due to page limit, we only show the result for Facebook here and leave the others in our project website[1]. As expected, all approaches encounter a performance drop when dealing

[1]The url link is omitted for blind review.

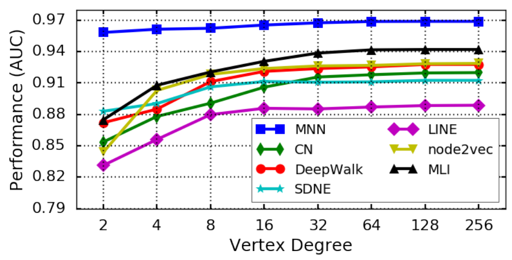

Figure 6: Performance on Cold-Start Scenarios

with cold start scenarios where the target vertex's degree is limited. Results indicate that our model is the most robust one to such data insufficient impact. For vertex with degree less than or equal to 2, we achieve a relative improvement of 5.56% comparing to the best existing method (MLI).

Evaluating Objective Channels Now we evaluate the contribution of each objective channel. Recall that besides the link prediction target (direct link, DL), we have six additional objective channels for various types of features (listed in Table 1). In Table 3, we show the performance gained by adding the additional objective channels to the basic setting where only direct link is used. Result indicates that the objective channels we propose do have positive effect on the performances. It also proves that multi-neural-network can successfully leverage all these heterogeneous features for a comprehensive vertex representation.

Table 3: Contribution of Different Objective Channels

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%**

Learning Methods Recall that we have 4 different learning methods for MNN (Figure 5). We show their performances in Table 4. Results indicate that stochastic learning achieves better performances comparing to joint learning, while fixed batch count out performs fix batch size. The reason would be that fixed batch count and stochastic learning gives each objective channel an equal chance to tune the embeddings, hence may better capture heterogeneous information instead of focusing on only the strongest ones.

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

Hyperparameter Analysis Now we examine the influence of two key hyperparameters in our framework: the sampling rate α, the weighting parameter β and the embedding size. Results in Figure 8 indicate that setting sampling rate α to around 100 provides the best performance. For weighting parameter β, the best setting is in $[0.3, 0.5]$, which matches with our intuition that we need to balance between the target (direct link) and the other features. For embedding size, $k = 80$ provides the best result.

Table 4: Comparing Learning Methodologies

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

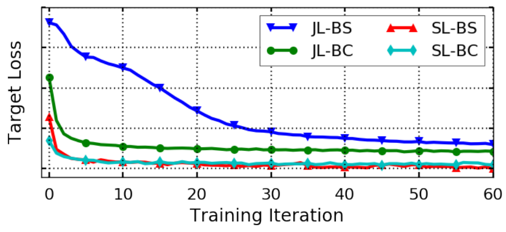

Figure 7: Training Curve with Different Learning Methods

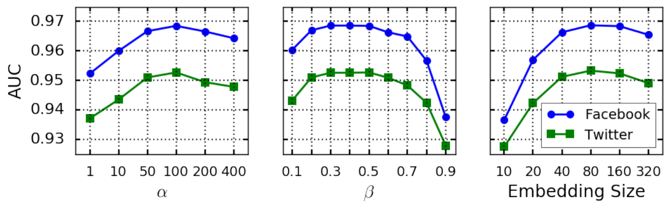

Figure 8: Hyperparameter Analysis

Conclusion & Future Works

In this paper, we target at link prediction over multiple aligned networks. The aligned networks provide valuable additional information for modeling the structure 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 common set of vertex embeddings, hence leveraging the heterogeneous information to achieve an accuracy and comprehensive vertex representation. Experiments with two real-world datasets indicate that our framework outperforms existing link prediction methods for both single-network and aligned networks scenarios.

The multi-neural-network framework we propose may also contribute to other tasks besides link prediction. We essentially propose a novel method of using multiple neural networks to handle heterogeneous features. For future works, we plan to extend it for more general scenarios.

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.

Bradley, A. P. 1997. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition* 30(7):1145–1159.

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 *Proceedings of the 7th international conference on Intelligent user interfaces*, 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.; 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 *Data Engineering (ICDE), 2017 IEEE 33rd International Conference on*, 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.