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

Link Prediction via Subgraph Embedding-Based Convex Matrix Completion

Zhu Cao,[*1] **Linlin Wang,**[*1] **Gerard de Melo**[†]
[*] IIIS, Tsinghua University, Beijing, China
[†] Rutgers University, New Brunswick, NJ, USA
{cao-z13, ll-wang13}@mails.tsinghua.edu.cn, gdm@demelo.org

Abstract

Link prediction is of fundamental importance in network science and machine learning. Early methods consider only simple topological features, while subsequent supervised approaches typically rely on human-labeled data and feature engineering. In this work, we present a new representation learning-based approach called SEMAC that jointly exploits fine-grained node features as well as the overall graph topology. In contrast to the SGNS or SVD methods espoused in previous representation-based studies, our model represents nodes in terms of subgraph embeddings acquired via a form of convex matrix completion to iteratively reduce the rank, and thereby, more effectively eliminate noise in the representation. Thus, subgraph embeddings and convex matrix completion are elegantly integrated into a novel link prediction framework. Experimental results on several datasets show the effectiveness of our method compared to previous work.

1 Introduction

Link prediction, which attempts to discover missing links between nodes in a complex network, is of fundamental importance in numerous tasks in countless different domains. For instance, link prediction can identify likely but not yet established links in an evolving social network, thus enabling recommendations to be presented to users.

There are number of challenges that need to be addressed for accurate and efficient link prediction. For one, our knowledge of complex networks is often very partial, i.e., only a small fraction of all true relationships and network structures may be known to us. Additionally, real-world networks are often very large, and hence it is computationally demanding to compute link probabilities using algorithms that account for the global structure.

Previous work on this still suffers from several shortcomings, especially with respect to their ability to capture rich characteristics of the graph both extensively and efficiently. Early work focused on straightforward similarity measures between pairs of nodes (Liben-Nowell and Kleinberg 2003). While these are quick to compute, they only exploit a single topological feature, neglecting crucial additional information that is often needed for accurate prediction. Probabilistic graphical models (Wang, Satuluri, and Parthasarathy 2007) take the network structure into consideration, however at the expense of substantially increased computational effort, with significant limitations in terms of their applicability to larger networks. Path-based approaches (Lichtenwalter, Lussier, and Chawla 2010) capture additional network structure features such as clusters of nodes, the degree of the neighborhood etc., relying on human-engineered features to improve the link prediction accuracy. However, there are important features that are hard to be formalized and human-engineered, such as the intertwining of paths. Capturing such features efficiently has been one of the main impediments to further improving the accuracy of link prediction on challenging real-world datasets.

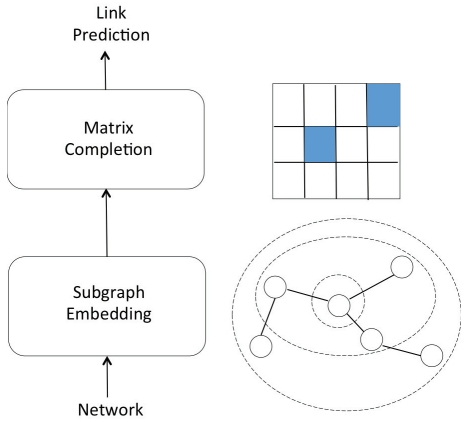

Figure 1: Schematic illustration of our SEMAC link prediction method.

To fill this gap, our work advocates for automatically detecting these inherent but irregular features, using a novel form of representation learning. By learning representations of nodes, we endeavor to capture a larger range of pertinent properties and signals about nodes in the network, drawing on more subtle cues that human-engineered features may overlook. Thus our model is advantageous in that we do not need to heavily rely on expert knowledge and can flexibly apply the same method across a range of heterogeneous kinds of networks. Network representations have recently shown promise for a number of tasks, including vertex clas-

[1] These authors contributed equally to this work.

sification, anomaly detection, and tag recommendation (Perozzi, Al-Rfou, and Skiena 2014; Yang et al. 2015).

We propose a representation-based approach for link prediction called SEMAC. Our algorithm induces a representation of each node by learning embeddings of subgraphs around given nodes in the graph. This allows it to capture suitable fine-grained network features such as a node's neighborhood information, while also accounting for the global community structure. SEMAC proceeds with a form of convex matrix completion to lower the rank of the subgraph embeddings in order to remove noise and improve its generalization abilities. Finally, the representations for different subgraphs of various depths associated with a given node are combined to form the node's overall representation, which can subsequently be used to predict new links in the graph. A simplified illustration of our method is given in Fig. 1.

The main contributions of our paper can be summarized as follows:

- We propose a novel form of representation learning for link prediction. Our SEMAC algorithm introduces the idea of subgraph embeddings to the task of link prediction.
- To the best of our knowledge, this work is also the first to combine the idea of subgraph embedding with convex matrix completion for learning network representations. Note that this is nontrivial due to the sparsity requirement for efficient matrix completion.
- We extensively evaluate our algorithm across a range of heterogeneous real-world datasets, and also demonstrate its scalability on large networks of up to a million nodes. The experiments show that our methods yield state-of-the-art link prediction results on all evaluated datasets.

The rest of the paper is organized as follows. In Section 2, we discuss related works. In Section 3.1, we define the problem of link prediction rigorously. In Section 3.3, we present our algorithm for solving link prediction. We outline the experiments in Section 4.1 and show their results in Section 4.3. Finally we close with a conclusion in Section 5.

2 Related Work

Traditional algorithms for link prediction can roughly be characterized as belonging to three different categories. The first of these straightforwardly computes the similarity between nodes using simple measures. Examples of such methods include Common Neighbors, Adamic/Adar (i.e., frequency-weighted common neighbors), and the Jaccard coefficient (Liben-Nowell and Kleinberg 2003). Among these, Adamic/Adar fares the best on most datasets. Although such measures are quick to compute, a decisive drawback is that they merely use a single (topological) feature for link prediction, neglecting crucial supplementary cues.

The second category consists of probabilistic graphical models, including Bayesian probabilistic modeling and relational Markov chain solutions (Wang, Satuluri, and Parthasarathy 2007). With methods of this sort, the network structure is taken account of, e.g. via community detection and hierarchical organization. However, training global probabilistic models can be computationally expensive and typically does not scale well to medium-sized, let alone large-scale networks.

The third category consists of path-based link prediction methods, including Propflow, Katz, and Commute Time (Lichtenwalter, Lussier, and Chawla 2010). Compared with the first category, this type of method captures additional features from the network structure, such as clusters of nodes or the degree of the neighborhood. At the same time, they are also time-efficient in comparison with the second category. However, these methods require human-engineered features and may be susceptible to overfitting.

There are also some interesting work that combine the output of several prediction algorithms. One can refer to (Ceci et al. 2015) and (Marbach et al. 2012) for some examples.

In a series of recent studies, network representations have shown promise for a number of network-related tasks. Perozzi et al. presented DeepWalk (Perozzi, Al-Rfou, and Skiena 2014), proposing the notion of a network representation for classification tasks. Their method combines random walks with the word2vec Skip-Gram with Negative Sampling (SGNS) approach. Nonnegative matrix factorization based representation methods have also been explored for tasks such as clustering and recommendation. Another method targeting the task of clustering is GraRep, which is based on DeepWalk, but where random walks are replaced by concatenations of 1-hop to k-hop neighbor information to take into account non-linear relations, and SGNS is replaced by SVD. Compared to GraRep, our method explicitly takes into account the relationships between i-hop and $(i+1)$-hop neighborhoods, and also overcomes the unrealistic assumption for SVD that non-observed entries are zero, by instead relying on a form of convex matrix completion.

Recently, Grover et al. showed that network representations of this sort are useful for link prediction. Their node2vec approach was shown to outperform traditional link prediction methods with an absolute improvement of 10% in terms of AUC on a number of link prediction datasets (Grover and Leskovec 2016). Our work continues this line of inquiry but proposes a novel representation model, which across a range of heterogeneous datasets substantially improves over methods such as node2vec, DeepWalk, and GraRep, thus advancing the state-of-the-art.

3 SEMAC Method

3.1 Problem Definition

Consider an undirected graph $G = (V, E)$, where V is the set of vertices and E is the set of edges. Here, we consider simple graphs without self-loops, in which there can be at most one edge between two nodes. We denote the set of vertices as $V = \{a_1, \ldots, a_n\}$. Entry A_{ij} of the adjacency matrix A is equal to 1 if an edge exists between two nodes a_i, a_j, and 0 otherwise. We further denote all possible $\frac{|V|(|V|-1)}{2}$ edges as $\mathcal{U}$, where $|V|$ is the total number of vertices, and the non-existent edges are referred to as

$\mathcal{U} - E$. Under this setting, link prediction can be formulated as the task of discovering missing edges from the set $\mathcal{U} - E$. In the following sections, we first review necessary background information and then propose our SEMAC algorithm to address this task. Via its novel network representation approach, this model captures both global aspects as well as fine-grained local information to enable high-accuracy predictions.

3.2 Representation Models

We first review some of the technical underpinnings of existing approaches for network representations. The notion of network representations originates from the idea of learning word embeddings in the natural language processing literature (Levy and Goldberg 2014), most notably the word2vec Skip-Gram with Negative Sampling (SGNS) approach. To start with, recall that classic language modeling maximizes

$$\sum_{t=1}^{T} \log Pr(w_{t-c}, \cdots, w_{t+c} \mid w_t), \tag{1}$$

where T is the length of a document. By assuming independence, we obtain

$$Pr(w_{t-c}, \cdots, w_{t+c} \mid w_t) = \prod_{-c \leq j \leq c, j \neq 0} Pr(w_{t+j} \mid w_t). \tag{2}$$

The Skip-Gram model further assumes a softmax probability distribution

$$Pr(w_{t+j} \mid w_t) = \frac{\exp(\Phi_{w_t}^T \Phi'_{w_{t+j}})}{\sum_{w=1}^{\mathcal{V}} \exp(\Phi_{w_t}^T \Phi'_w)}, \tag{3}$$

where $\Phi_w, \Phi'_w \in R^d$ are two vectors corresponding to w, referred to as the input word embedding and the output word embedding of w, respectively. $\mathcal{V}$ is the overall vocabulary.

To maximize $Pr(w_{t+j} \mid w_t)$, it is equivalent to maximize

$$\log \exp(\Phi_{w_t}^T \Phi'_{w_{t+j}}) + \log \sum_{w=1}^{\mathcal{V}} \exp(-\Phi_{w_t}^T \Phi'_w). \tag{4}$$

More generally, the exponential function may be replaced by an arbitrary nonlinear function σ, and thus the objective becomes

$$\log \sigma(\Phi_{w_t}^T \Phi'_{w_{t+j}}) + \log \sum_{w=1}^{\mathcal{V}} \sigma(-\Phi_{w_t}^T \Phi'_w). \tag{5}$$

Since the second term is time-consuming to compute, it is typically replaced by negative sampling,

$$\log \sigma(\Phi_{w_t}^T \Phi'_{w_{t+j}}) + k \cdot \mathbb{E}_{w \sim P_D} \log \sigma(-\Phi_{w_t}^T \Phi'_w), \tag{6}$$

where D is the observed data and $P_D(w) = \frac{\#(w)}{|D|}$ is the distribution of words.

The first network representation work, DeepWalk (Perozzi, Al-Rfou, and Skiena 2014), relied on random walks to select paths of nodes, which served as analogs of sentences. Based on these, it then simply invokes SGNS to obtain node embeddings in the same way as word embeddings are obtained.

Levy and Goldberg proved that SGNS, under certain conditions, can be regarded as an implicit form of matrix factorization (Levy and Goldberg 2014). Multiple works provided empirical corroboration showing that SVD and their variants can outperform SGNS (Cao, Lu, and Xu 2015; Tu et al. 2016). Following Levy et al., DeepWalk can be modified to optimize

$$\sum_w \sum_c \log \left(\frac{\#(w,c) \cdot |\mathcal{D}|}{\#(w) \cdot \#(c)} \right) - \log(k). \tag{7}$$

GraRep (Cao, Lu, and Xu 2015) adopts this idea of replacing SGNS by SVD, but further improves over DeepWalk by observing that taking the average of the context items within a window of the target item is suboptimal. They instead treat each context item with distance k to the target item separately ($1 \leq k \leq K$) and concatenate the resulting vectors to capture weights and non-linearity.

3.3 Our SEMAC Algorithm

Our SEMAC approach improves over previous representation learning methods by introducing a number of key innovations, especially the use of subgraph embeddings and the use of a novel form of convex matrix factorization. It applies to both undirected and directed graphs. The overall procedure consists of five steps, for which we provide the high-level story in Algorithm 1. Of these five steps, the first four can be prepared offline, while the fifth one is activated on demand given a pair of nodes to be evaluated.

Algorithm 1 Overall Algorithm

1. Determine target subgraphs and radial contexts.
2. Compute the PPMI matrix.
3. Nuclear norm regularization-based matrix completion.
4. Form node embeddings from subgraph embeddings.
5. Compute link prediction score given query node pair.

In the first step, we retrieve subgraphs $g_{v,d}$ at different depths d around nodes v, for which we will subsequently compute subgraph embeddings that later give rise to embeddings for nodes. Subgraph embeddings were first proposed in the context of graph classification, clone detection, and malware detection and have enjoyed success in those tasks (Narayanan et al. 2016). We adapt this idea to the task of link prediction, but will rely on novel techniques to learn the embeddings.

The first step of obtaining subgraphs is presented in more detail in Algorithm 2. By invoking breadth-first search, we capture multiple subgraphs for each node at different depth levels. These constitute our target *vocabulary* $\mathcal{V}$, in analogy to vocabularies in word representation learning. For each subgraph with a center node v and a radial depth d, we also determine its *radial context* $C_{v,d}$. By accounting for the radial context around each node in this way, we capture the natural correlations between different depth levels. Such correlations can generally help the node representation learning, giving our approach an advantage over methods such as

GraRep, which treat the representations of different lengths as independent.

Algorithm 2 $\mathcal{V}, \mathcal{C}$=Vocabulary(G,D)

Input: A graph $G = (V, E)$ and depth D.
for $(v, d) \in V \times \{1, \ldots, D\}$ **do**
  $g_{v,d} \leftarrow$ BFS with v as the root and d as the depth
for $(v, d) \in V \times \{1, \ldots, D\}$ **do**
  $C_{v,d} \leftarrow \{g_{v,d'} \mid d' \in \{d-1, d+1\} \cap \{1, \ldots, D\}\}$
    $\cup \{g_{u,d} \mid (u, v) \in E\}$
Output: Output $\{g_{v,d}\}$ as the vocabulary $\mathcal{V}$ and $\{C_{v,d}\}$ as the set of radial contexts $\mathcal{C}$.

In the second step, we learn representations of the target subgraphs in $\mathcal{V}$. To this end, we compute a positive PMI (PPMI) matrix (Levy and Goldberg 2014) as $M = (A+A^2)/2$. Unlike previous work (Yang et al. 2015), where A is the original adjacency matrix for a graph, we instead operate at the level of subgraphs and form an adjacency matrix of subgraphs. Algorithm 3.3 below describes this procedure.

Algorithm 3 PPMI Matrix M= Rep($\mathcal{V}, \mathcal{C}$)

Input: a vocabulary $\mathcal{V}$.
Step 1: Define a graph $G' = (V', E')$ with subgraphs in $\mathcal{V}$ as vertices and edges between two subgraphs representing that one subgraph is in the radial context of another subgraph.
Step 2: Let d_v be the degree of node v for all $v \in V'$. Set all A_{ij} to 0 if $(v_i, v_j) \notin E'$, and $1/d_{v_i}$ if $(v_i, v_j) \in E'$.
Output: PPMI matrix $M = (A + A^2)/2$.

In Algorithm 3.3, we first form a new graph $G' = (V', E')$ where the node set V' is the vocabulary $\mathcal{V} = \{g_{v,d}\}$ obtained in Algorithm 2. For the edge set, an edge $\{g_{u,d}, g_{v,d'}\}$ exists in E' iff $g_{v,d'} \in C_{u,d}$. Then we define a matrix A based on G' which induces a PPMI matrix M.

In the third step, we learn a low-rank representation of the PPMI matrix M to remove white noise. Let W denote such a representation. However, instead of SVD or max-margin matrix factorization (MMMF), we propose relying on a form of convex matrix completion called nuclear norm regularization (NNR) (Mazumder, Hastie, and Tibshirani 2010). Let Ω be the set of nonzero entries. For ease of later discussion, for a matrix Y, define its decomposition as

$$Y = P_\Omega(Y) + P_\Omega^\perp(Y), \tag{8}$$

where

$$P_\Omega(Y)(i,j) = \begin{cases} Y(i,j), & (i,j) \in \Omega \\ 0, & (i,j) \notin \Omega \end{cases}. \tag{9}$$

The objective of NNR is to minimize

$$\frac{1}{2}||P_\Omega(M) - P_\Omega(W)||_F^2 + \lambda||W||_*, \tag{10}$$

where $||\cdot||_*$ is the nuclear norm and $||\cdot||_F$ is the Frobenius norm. This objective is convex and thus lends itself to tractable global optimization. There are three advantages of this approach over SVD and MMMF. First, SVD presupposes that the entries are exact. In particular, non-observed entries are presumed to be zero. This assumption is inadequate in our setting and is avoided by NNR. Second, MMMF is not convex and thus one often falls into local minima, while NNR is convex. Third, NNR automatically learns the rank instead of sweeping the rank. Hence, its run-time is reduced. Algorithm 4 presents the details of the SOFT-IMPUTE algorithm for NNR. At a high level, it iteratively replaces missing elements with those of a soft-thresholded SVD. The complexity of the NNR algorithm is $O(|\Omega|r+(m+n)r^2)$, where m, n are the number of columns and rows, respectively, r is the rank.

Algorithm 4 W=SOFT-IMPUTE(M)

$Z^{old} := 0$.
for $\lambda_1 > \lambda_2 > \cdots > \lambda_K$ **do**
  $Z^{new} := S_{\lambda_k}(P_\Omega(M) + P_\Omega^\perp(Z^{old}))$.
  while $\frac{||Z^{new} - Z^{old}||_F^2}{||Z^{old}||_F^2} > \epsilon$ **do**
    $Z^{old} := Z^{new}$.
    $Z^{new} := S_{\lambda_k}(P_\Omega(M) + P_\Omega^\perp(Z^{old}))$.
  $\hat{Z_{\lambda_k}} := Z^{new}$.
Output: $\hat{Z_{\lambda_K}}$ expressed as its SVD decomposition.

Algorithm 4 provably finds an optimal solution to Eq. (10) modulo minor computational inaccuracy. Within Algorithm 4, λ_k equals $\alpha \cdot \eta^k$, where α is a parameter to be tuned for each dataset, and η is the learning rate. $S_{\lambda_i}(Y)$ stands for the sum of the components of Y that have singular values larger than λ_i. In the algorithm, $P_\Omega^\perp(Z^{old})$ is not stored directly, but obtained from Z^{old}. Note that Z^{old} is a low-rank matrix, and hence can be stored compactly using its SVD decomposition. The outer iteration of the algorithm allows us to gradually decrease the threshold λ_i of the soft-thresholded SVD $S_{\lambda_i}(Y)$.

Next, in the fourth step, we create node embeddings from the previous subgraph embeddings. Algorithm 5 takes the subgraph representations of all depths $1 \le d \le D$ for a given node generated by the third step, and concatenates them.

Algorithm 5 $M = Concat(W_1, \ldots, W_D)$

Input: Embeddings $W_1, \ldots, W_D$ for subgraphs of various depths.
Output: A concatenated embedding $M_u = (W_1(u), \cdots, W_D(u))$ for each vertex u.

Finally, we predict links by calculating the cosine value of the angle between the vector representations of the two vertices for a given link under consideration. These scores are closely related to the probability of this link existing. This is presented in Algorithm 6.

With the above approach, SEMAC addresses two key challenges in link prediction: (1) By generating subgraphs

Algorithm 6 Rep2Score(M, u, v)

Input: a vertex representation of all nodes M, a pair of nodes (u, v), between which we ask whether a link exists.
Step 1: $M_u, M_v \leftarrow$ representation vectors for u and v
Output: Score$(u, v) = \frac{|M_u \cdot M_v|}{|M_u||M_v|}$

from each node, our method gathers pertinent structural information from the network, learning representations that capture fine-grained local neighborhood information, while adaptively accounting for overall properties of the graph structure. (2) By relying on a modified convex matrix completion to find a low-rank approximation of the features, SEMAC is able to disregard uninformative signals, while improving the model's generalization abilities.

3.4 Extensions

Before concluding this section, we discuss some possible variants of our algorithm. The first option is to replace convex matrix completion by another dimensionality reduction method, using the Johnson-Lindenstrauss lemma (Krahmer and Ward 2011). The Johnson-Lindenstrauss lemma projects vectors of size $O(n)$ to vectors of size $O(\log n)$ with little distortion. Said projection is achieved in two phases: First, randomly projecting the vector to an appropriate dimension, and, second, multiplying it by a suitable scalar to preserve the norm. Replacing our SOFT-IMPUTE(A) by this procedure yields a fast variant of our algorithm. Two further modifications are to rely on *streaming* and *non-random walk* variants, which can be obtained analogously to modifications of previous work (Perozzi, Al-Rfou, and Skiena 2014).

Secondly, our algorithm can be adapted to weighted graphs. When constructing the matrix A in Algorithm 3.3, we set the entries in each row to equal values. For weighted graphs, this can be set differently according to the edge weights. We leave detailed investigation to future work.

Finally, we remark that our algorithm can easily incorporate additional side information in the form of node features. Surprisingly, by adding these additional features, both the speed and performance of the algorithm can be improved, which also holds true in other matrix completion tasks, such as multi-class labeling (Xu, Jin, and Zhou 2013). Specifically, denote the representation of the network as $A \in \mathbb{R}^{n \times n}$, and the node-feature matrix as $\Phi \in \mathbb{R}^{n \times k}$. Here, the feature dimensionality of the node k is much smaller than the number of nodes n. Under certain mild assumptions, it can be shown that matrix completion on $\Phi^T A \Phi$, which is a much smaller matrix, is equivalent to matrix completion on A. Additionally, the information in Φ can be utilized in comparing two nodes to compute their similarity. Thus, both the speed and the effectiveness can be improved.

4 Experimental Evaluation

We evaluated our SEMAC approach on a series of heterogeneous real-world datasets and compared it against both classic methods and recent representation-based approaches.

4.1 Datasets

We consider the following datasets:

Facebook (McAuley and Leskovec 2012; Leskovec and Krevl 2014): We use a Facebook social network dataset, which consists of 4,039 nodes, each corresponding to a Facebook account. This sparse graph contains 88,234 edges representing the friendship relation. In order to examine the effect of graph sizes and levels of sparsity, we also consider small subsets by randomly selecting nodes from the original data set (which is denoted as FOrig), to generate the sample datasets F200, F400, F800 and F1600, which are connected components with 200, 400, 800, and 1600 nodes, respectively. By checking the degree distributions of these datasets, we find that they still follow a power law, and thus preserve properties of a social network, albeit with a smaller size.

Wikipedia (Leskovec and Krevl 2014): This real-world dataset is collected from Wikipedia and consists of 7,115 nodes and 103,689 edges. We denote this dataset as Wiki.

Coauthorship (Leskovec and Krevl 2014): This real-world dataset is formed from the coauthor network of general relativity section on arXiv. It consists of 5,242 nodes and 14,496 edges. We denote this dataset as Coauth.

PPI (Breitkreutz et al. 2008): This protein-protein interaction network consists of 19,706 nodes and 390,633 edges. It is denoted as PPI.

4.2 Experiment Setup

We conduct a series of experiments and evaluate SEMAC at different data scale levels. For each dataset, the observed edges $\mathcal{E}$ are split into two parts $\mathcal{E}^T$ and $\mathcal{E}^P$, where $\mathcal{E}^T$ is used for training and $\mathcal{E}^P$ for testing. The splitting is performed with 5-fold cross-validation. That is, the observed edges are split to five equal parts. Then we repeat 5 times, each time take one part as the test set and the rest four parts as the training set.

To assess the effectiveness of different link prediction approaches, we rely on the standard area under the receiver operating characteristic curve (AUC) metric (Hanley and Mcneil 1982). We repeat experiments on each dataset for 100 times and report the average AUC score. These experiments are run on a laptop with 2.8 GHz CPU and 8G memory.

We compare our model with the following state-of-the-art baselines: Common Neighbor, Salton Index (Salton and Mcgill 1983), Jaccard Index (Jaccard 1901), Sorensen Index (Sorensen 1948), and Resource Allocation (Zhou, Lü, and Zhang 2009). We also compare against matrix factorization methods, including matrix factorization for recommender systems (MFRS) (Koren, Bell, and Volinsky 2009), link prediction via matrix factorization (LPMF) (Menon and Elkan 2011), distributed stochastic gradient descent (DSGD) (Gemulla et al. 2011), and greedy asynchronous stochastic gradient descent (GASGD) (Petroni and Querzoni 2014). Finally, we consider DeepWalk (Perozzi, Al-Rfou, and Skiena 2014), GraRep (Cao, Lu, and Xu 2015), and node2vec (Grover and Leskovec 2016). Note that DeepWalk is originally intended to be used for classification tasks and not for link prediction. Here, the baseline DeepWalk means that after the embeddings of the nodes are learnt through Deep-

Walk, the cosine similarity score is computed as in Algorithm 6.

4.3 Link Prediction Result Analysis

The experimental results of this comparison are given in Table 1. For these experiments, the learning rate η and the ratio of regularization λ are optimized to be 0.3 and 0.001, respectively, according to cross validation within the training set. The depth D is taken to be 3 for the datasets FOrig, Wiki, Coauth, and 2 for the dataset PPI.

From Table 1, we conclude that our method SEMAC achieves favourable results on predicting links in diverse networks of heterogeneous structure and at different scale levels in terms of the number of nodes. SEMAC consistently outperforms previous methods across all datasets. Specifically, our algorithm has an absolute improvement over the second best algorithm by 1.5%, 1.7%, 1.2%, 0.8%, 0.9%, 2.2%,2.9%, and 2.4% for the datasets F200, F400, F800, F1600, Wiki, Coauth, Twitter, and PPI, respectively. This is due to our model's ability to learn representations that capture more neighborhood information, while dynamically adapting to different network topologies and structures, while our convex matrix completion mitigates the effects of noise and increases the generalization abilities of our representations.

For further understanding, we can compare network types and structures among the datasets. For the Wikipedia dataset, which saw the weakest results, we observed that the presence of an edge has an overloaded interpretation in the sense that there are actually links of many different kinds. Protein interaction (PPI) data to some extent exhibits similar behaviour. Thus, datasets that conflate heterogeneous kinds of links remain challenging for current methods. A potential solution for this sort of data, with different types of links, is to combine the information on the link. Further work is thus needed. in order to discover suitable application scope and make proper model selection decisions. Note however, that even in these cases, our approach still eclipses previous methods.

Parameter Sensitivity To further evaluate the effect of SEMAC's parameters, we vary these on F400. For these tests, we fix the number of inner iterations to an appropriate value ($i = 100$), which ensures convergence. We then proceed to vary the learning rate α and the ratio of the regularization term λ to determine their effects on the performance of our algorithms.

Learning rate. Figure 2(a) shows the effect of different choices of learning rate α, ranging from 0.1 to 0.5. We can see that initially when learning rate is 0.1, the performance is a bit worse for the given number of iterations. This phenomenon can be explained by the slow convergence, which is in turn caused by the small learning rate. As the learning rate increases, the performance gradually becomes better. But at some point (when rate=0.4), the performance decreases. It can be seen that for both SEMAC, the best learning rate is between 0.2 and 0.3.

Regularization. Figure 3(b) shows the effect of the regularization ratio, which ranges from 10^{-9} to 1. When the ratio

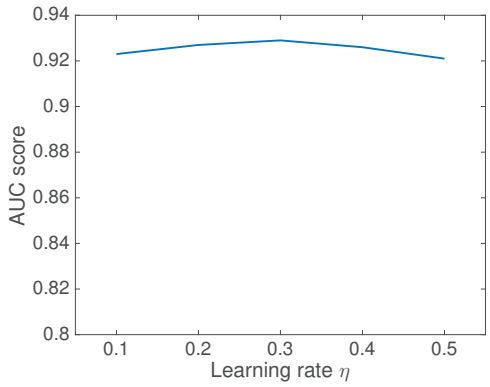

Figure 2: Varying the learning rates for SEMAC on F400.

grows too large (≥ 0.1), the AUC scores rapidly degrade because the objective function then plays a too small role compared to the regularization, and the prediction is no longer accurate. As the ratio of regularization decreases (shown in log scale on the figure), the performance gradually improves, and finally stays at a constant level. This implies that for this specific task of link prediction, it would not hurt to omit the regularization term. We leave the theoretical understanding of this fact as future work.

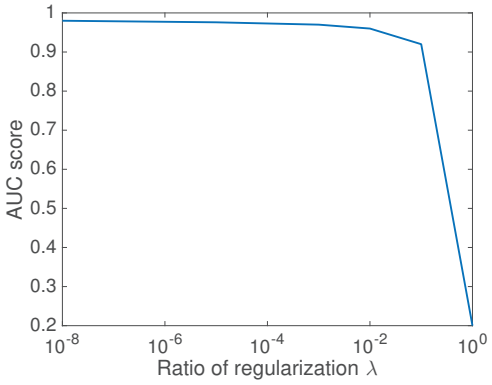

Figure 3: Varying the ratio of regularization for SEMAC on F400.

Depth. The maximum depth D affects both the quality of the results and the time complexity. More precisely, increasing D improves the performance but significantly increase the time complexity and the space complexity. In our experiments, we set the maximum D possible for each data size given our hardware environment.

Iterative rank minimization. Though matrix completion is sometimes misinterpreted as matrix factorization such as SVD, we emphasize that these two concepts are different in that matrix completion uses an iterative process to reduce the rank (Ma, Goldfarb, and Chen 2011). This is of particular importance because the exact rank of the considered matrix is often unknown and it is desirable to automatically reduce the rank. We analyse how the scores develop for dif-

AUC	F200	F400	F800	F1600	FOrig	Wiki	Coauth	PPI
Common Neighbor	70.2(4)	76.9(6)	80.2(5)	82.4(8)	87.2(3)	83.2(6)	85.1(2)	86.2(7)
Salton Index	69.7(8)	76.8(9)	80.1(5)	82.4(4)	86.8(7)	84.1(3)	84.8(5)	85.7(8)
Jaccard Index	69.6(3)	76.6(3)	80.5(9)	82.3(3)	86.9(5)	83.7(5)	85.3(2)	85.4(5)
Sorensen Index	69.8(7)	76.5(6)	80.3(7)	82.1(5)	86.9(2)	82.8(9)	84.6(4)	86.1(5)
Resource Allocation	70.3(2)	76.5(3)	80.2(4)	82.8(9)	87.5(6)	84.5(4)	84.7(8)	86.3(3)
MFRS	73.2(5)	78.3(8)	78.9(5)	79.5(2)	82.1(4)	69.4(2)	71.7(4)	71.9(4)
LPMF	90.9(5)	91.4(6)	91.9(6)	92.5(2)	92.7(5)	81.5(6)	82.9(7)	81.7(5)
DSGD	73.5(7)	78.7(6)	79.2(7)	79.7(8)	82.6(7)	70.3(9)	72.4(6)	72.5(7)
GASGD	73.9(3)	78.8(2)	79.4(5)	79.8(3)	82.9(5)	70.5(6)	72.4(8)	72.7(2)
DeepWalk	79.4(2)	86.5(6)	92.7(9)	95.4(7)	96.8(8)	87.8(4)	88.1(7)	88.2(9)
GraRep	79.1(6)	87.7(8)	93.1(7)	95.3(9)	97.3(3)	88.2(8)	87.4(4)	88.9(3)
node2vec	80.5(8)	86.2(7)	92.9(4)	96.8(3)	96.8(3)	87.0(6)	89.6(8)	87.7(4)
SEMAC	**92.4(9)**	**93.1(2)**	**94.3(3)**	**97.6(4)**	**98.2(7)**	**90.4(9)**	**92.5(7)**	**91.3(7)**

Table 1: AUC scores of link prediction. Confidence intervals of p-value 0.05 are shown in the bracket.

ferent numbers of iterations, which illustrates the denoising power and generalization abilities of the matrix completion method. The results for SEMAC on the Facebook data is shown in Fig. 4. We observe that the AUC score reaches a peak of 98% within only 10 outer iterations.

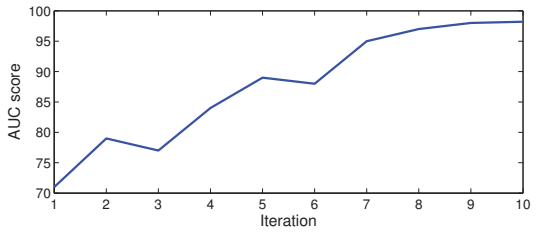

Figure 4: The rank minimization across different iterations.

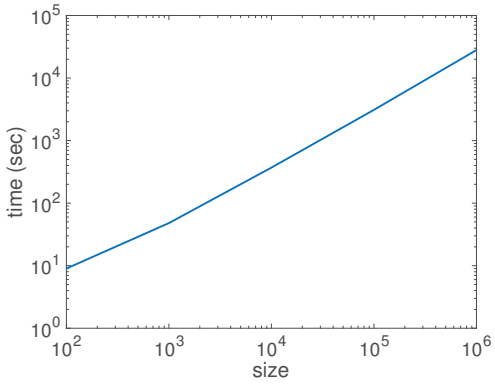

Figure 5: Runtime for Erdős-Rényi graphs of different sizes.

Scalability Finally, we examine the running time of our algorithm. Since the prediction time is much shorter than the training time according to our experiments, we focus on examining the theoretical time complexity of the training time. Theoretically, the runtime of the representation learning process scales approximately linearly with $O(n)$, because matrix completion runs in time $O(|\Omega|r + (n+m)r^2)$, which is linear with respect to $n+m$ when the rank r is small and $|\Omega| < (n+m)r$. We evaluate this experimentally on randomly generated Erdős-Rényi graphs with 100 to 1,000,000 nodes and an average node degree of 10. The running time of our algorithm is plotted in Fig. 5. It takes 7.4 hours to run on a graph of size 1,000,000. This shows that our method can scale to networks with a million nodes. Moreover, our algorithm is also parallellizable on multi-core clusters, as is the case for other matrix factorization algorithms as well.

5 Conclusion

In this paper, we have proposed a new link prediction method called SEMAC, which is based on a novel representation learning approach that captures rich information about the network structure around each node, obviating the need for manual feature engineering. Our approach rests on the idea of learning subgraphs embeddings and relying on nuclear norm regularization-based matrix completion. Experiments on a series of heterogeneous datasets from different domains show the effectiveness of our method.

Acknowledgement

Gerard de Melo's research is funded by the DARPA SocialSim program.

References

Breitkreutz, B.-J.; Stark, C.; Reguly, T.; Boucher, L.; Breitkreutz, A.; Livstone, M.; Oughtred, R.; Lackner, D. H.; Bähler, J.; Wood, V.; et al. 2008. The biogrid interaction database: 2008 update. *Nucleic acids research* 36(suppl 1):D637–D640.

Cao, S.; Lu, W.; and Xu, Q. 2015. GraRep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, 891–900. ACM.

Ceci, M.; Pio, G.; Kuzmanovski, V.; and Džeroski, S. 2015. Semi-supervised multi-view learning for gene network reconstruction. *PloS one* 10(12):e0144031.

Gemulla, R.; Nijkamp, E.; Haas, P. J.; and Sismanis, Y. 2011. Large-scale matrix factorization with distributed stochastic gradient descent. *KDD* 69–77.

Grover, A., and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 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.

Jaccard, P. 1901. Etude comparative de la distribution florale dans une portion des alpes et du jura. *Bulletin de la Societ Vaudoise des Sciences Naturelles* 547–579.

Koren, Y.; Bell, R.; and Volinsky, C. 2009. Matrix factorization techniques for recommender systems. ieee, computer journal, 42(8), 30-37. *Computer* 42(8):30–37.

Krahmer, F., and Ward, R. 2011. New and improved johnsonlindenstrauss embeddings via the restricted isometry property. *Siam Journal on Mathematical Analysis* 43(3):1269–1281.

Leskovec, J., and Krevl, A. 2014. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data.

Levy, O., and Goldberg, Y. 2014. Neural word embedding as implicit matrix factorization. In *Advances in Neural Information Processing Systems 27*. Curran Associates, Inc. 2177–2185.

Liben-Nowell, D., and Kleinberg, J. 2003. The link prediction problem for social networks. In *Twelfth International Conference on Information & Knowledge Management*, 556–559.

Lichtenwalter, R. N.; Lussier, J. T.; and Chawla, N. V. 2010. New perspectives and methods in link prediction. *KDD* 243–252.

Ma, S.; Goldfarb, D.; and Chen, L. 2011. Fixed point and bregman iterative methods for matrix rank minimization. *Mathematical Programming* 128(1-2):321–353.

Marbach, D.; Costello, J. C.; Küffner, R.; Vega, N. M.; Prill, R. J.; Camacho, D. M.; Allison, K. R.; Kellis, M.; Collins, J. J.; Stolovitzky, G.; et al. 2012. Wisdom of crowds for robust gene network inference. *Nature methods* 9(8):796–804.

Mazumder, R.; Hastie, T.; and Tibshirani, R. 2010. Spectral regularization algorithms for learning large incomplete matrices. *Journal of machine learning research* 11(Aug):2287–2322.

McAuley, J., and Leskovec, J. 2012. Learning to discover social circles in ego networks. *Advances in Neural Information Processing Systems* 539–547.

Menon, A. K., and Elkan, C. 2011. *Link Prediction via Matrix Factorization*. Springer Berlin Heidelberg.

Narayanan, A.; Chandramohan, M.; Chen, L.; Liu, Y.; and Saminathan, S. 2016. subgraph2vec: Learning distributed representations of rooted sub-graphs from large graphs. *arXiv preprint arXiv:1606.08928*.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 701–710.

Petroni, F., and Querzoni, L. 2014. Gasgd: stochastic gradient descent for distributed asynchronous matrix completion via graph partitioning. In *Proceedings of the 8th ACM Conference on Recommender systems*, 241–248.

Salton, G., and Mcgill, M. J. 1983. *Introduction to modern information retrieval*. McGraw-Hill.

Sorensen, T. 1948. A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on danish commons. *Biol. Skr.* 1.

Tu, C.; Zhang, W.; Liu, Z.; and Sun, M. 2016. Max-margin deepwalk: Discriminative learning of network representation. In *Proceedings of IJCAI*.

Wang, C.; Satuluri, V.; and Parthasarathy, S. 2007. Local probabilistic models for link prediction. In *icdm*, 322–331.

Xu, M.; Jin, R.; and Zhou, Z.-H. 2013. Speedup matrix completion with side information: Application to multi-label learning. In *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc. 4999–5007.

Yang, C.; Liu, Z.; Zhao, D.; Sun, M.; and Chang, E. Y. 2015. Network representation learning with rich text information. In *IJCAI*.

Zhou, T.; Lü, L.; and Zhang, Y. C. 2009. Predicting missing links via local information. *European Physical Journal B* 71(4):623–630.