DeepWalk Link Prediction Algorithm Based on Matrix Factorization (Postprint)

Authors: Ye Zhonglin, Cao Rong, Zhao Haixing, Zhang Ke, Zhu Yu

Date: 2018-12-13T00:00:00+00:00

Abstract

Existing link prediction methods primarily rely on data sources based on neighbors, paths, and random walks, utilizing node similarity assumptions or maximum likelihood estimation. There remains a notable lack of link prediction research based on neural networks. Some studies grounded in neural networks have demonstrated that the DeepWalk network representation learning algorithm, which is neural network-based, can more effectively mine structural features from networks. Existing research has proven that DeepWalk is equivalent to factorizing a target matrix. Therefore, we propose a DeepWalk link prediction algorithm based on matrix factorization (LPMF). This algorithm first obtains network representation vectors through decomposition using the matrix factorization-based DeepWalk algorithm; then calculates the similarity between each pair of nodes via cosine similarity to construct a similarity matrix for the target network; finally, it conducts link prediction experiments on three real-world citation networks utilizing this similarity matrix. Experimental results demonstrate that the proposed link prediction algorithm outperforms more than 20 existing link prediction algorithms, which fully indicates that LPMF can effectively mine the structural correlations between nodes in networks and can exhibit relatively excellent performance in link prediction for real-world networks.

Full Text

Preamble

Link Prediction Based on Matrix Factorization for DeepWalk

Ye Zhonglin[1,3], , Cao Rong[2,3], , Zhao Haixing[1,2,3], †, Zhang Ke[2,3], , Zhu Yu[2,3],
(1. College of Computer Science, Shaanxi Normal University, Xi' an 710119, China; 2. College of Computer Science, Qinghai Normal University, Xining

810008, China; 3. Key Laboratory of Tibetan Information Processing & Machine Translation of Qinghai Province, Xining 810008, China; 4. Key Laboratory of Tibetan Information Processing, Ministry of Education, Xining 810008, China)

Abstract: Existing link prediction methods primarily rely on neighbor-based, path-based, and random walk approaches, utilizing node similarity assumptions or maximum likelihood estimation. Research on link prediction based on neural networks remains scarce. However, studies on neural networks demonstrate that the DeepWalk network representation learning algorithm can more effectively extract structural features from networks. It has been proven that DeepWalk is equivalent to factorizing a target matrix. Therefore, this paper proposes a link prediction algorithm based on matrix factorization for DeepWalk (LPMF). The algorithm first obtains network representation vectors through matrix factorization based on DeepWalk, then constructs a similarity matrix for the target network by calculating cosine similarity between each pair of nodes, and finally conducts link prediction experiments on three real-world citation networks using this similarity matrix. Experimental results show that the proposed algorithm outperforms over 20 existing link prediction algorithms, demonstrating that LPMF can effectively mine structural correlations between nodes in networks and achieve excellent performance in practical link prediction tasks.

Keywords: link prediction; neural network; DeepWalk; network representation learning; matrix factorization; similarity matrix

0 Introduction

With the continuous development of network technology, the evolution of complex networks has become a hot research topic, with link prediction being a fundamental computational problem in network evolution and modeling. Link prediction in networks refers to predicting the likelihood of a connection forming between two nodes that are not yet connected, based on known network structure and other information [1]. Predicting existing but undiscovered connections is called unknown prediction, while predicting potential future connections is called future prediction. In recent years, link prediction in large-scale networks has become a research focus, with applications in various tasks such as network modeling [2–5], protein network prediction [6,7], social network analysis [8–10], tag classification [11–13], knowledge acquisition [14], anomaly detection [15–17], and recommendation systems [18,19]. Various network modeling methods have been proposed to reveal the evolution mechanisms of real-world networks [20,21], yet it remains difficult to determine which method accurately reflects the generation process of real networks. Benefiting from improved computational performance and public access to large-scale social network data, link prediction has evolved from mining node attributes to mining network attributes, and from small-scale to large-scale social network prediction [22]. However, traditional

link prediction algorithms based on entropy or maximum likelihood estimation [23] suffer from high computational complexity and low accuracy [24]. Moreover, there is still a lack of efficient link prediction algorithms suitable for large-scale datasets and in-depth application-level analysis of large-scale real data. Research in these two areas helps reveal the advantages and limitations of link prediction itself.

Google' s word2vec [25,26] is a word representation learning algorithm based on a three-layer neural network probabilistic model. Using large-scale language corpora, this algorithm employs neural networks to obtain low-dimensional, dense vector representations of words in a linguistic space. With a fixed window size, it captures adjacent words within the window as context for the current word, then inputs both into a neural network for learning. Neural network-based word representation learning has achieved great success in language models. Subsequently, the DeepWalk [27] network representation learning algorithm was proposed for network space models based on word2vec. This algorithm uses random walks to obtain context nodes for the current node, then inputs both into a neural network model for learning, ultimately obtaining low-dimensional, dense vector representations for each node in the network space model. Network representation learning essentially converts network features into a vector format suitable for processing. These vectors can be visualized in 2D space to demonstrate clustering phenomena among nodes with similar attributes, as shown in Figure 1.

[Figure 1: see original paper]

DeepWalk network representation learning is a neural network-based algorithm that, through deep learning of network structure, enables nodes with similar network structures to have similar representation vectors. Using DeepWalk not only helps better understand structural correlations between nodes but also alleviates training data scarcity caused by network sparsity. Because DeepWalk employs local random walks, it is more efficient for large-scale network structure mining. Later, Yang et al. [28] mathematically proved that DeepWalk is equivalent to factorizing a target matrix, though they did not experimentally validate differences between the two representation learning methods. Since different matrix factorization algorithms yield different network representation learning results, both DeepWalk and matrix factorization can obtain network representation features. The difference is that DeepWalk uses random walk strategies to avoid directly computing and factorizing the matrix, making it suitable for large-scale network representation learning. Factorizing a target matrix has high time complexity, and algorithm accuracy is limited by factorization efficiency.

Based on the equivalence between DeepWalk and matrix factorization, this paper proposes a link prediction algorithm based on matrix factorization. This method introduces matrix factorization-based DeepWalk representation learning into link prediction for the first time, verifying that neural network-like methods can more effectively mine structural correlations in networks, and that the trained network representations also perform excellently in link prediction.

Unlike traditional global random walk [29], random walk with restart [30], and local random walk [31] algorithms used in link prediction, DeepWalk not only uses local random walks to obtain context nodes but also inputs both current and context nodes into neural networks for training, thoroughly and deeply mining network structural features and reflecting structural similarities between nodes. Since Yang et al. [28] proved DeepWalk is equivalent to factorizing matrix M, the matrix factorization-based DeepWalk representation learning algorithm used in this paper avoids random walks and neural network training, instead using efficient matrix factorization methods to factorize the target matrix directly. This approach inherits DeepWalk' s advantages while satisfying the need to directly convert adjacency matrices into network representations.

In summary, this paper' s main contributions are twofold: (a) It introduces matrix factorization-based DeepWalk network representation learning into link prediction, demonstrating that simple matrix factorization can achieve prediction capabilities almost equivalent to neural network algorithms; (b) It conducts link prediction, visualization, and case study experiments on three real citation network datasets. Experimental results show that the proposed method can effectively learn network structural features, enabling better prediction performance.

---

1 Related Work

For link prediction, current commonly used methods are node similarity-based algorithms, which mainly include three types: local information-based similarity indices, path-based similarity indices, and random walk-based similarity indices.

Local information-based similarity indices include Common Neighbors (CN) [32], Adamic-Adar (AA) [33], and Resource Allocation (RA) [34]. CN is the simplest local information-based similarity, defined as: if two nodes share many common neighbors, they are similar. More common neighbors yield higher similarity. Considering node degree influence from different perspectives, six variants exist: Salton (cosine similarity) [35], Jaccard [36], Sorenson [37], Hub Promoted Index (HPI) [38], Hub Depressed Index (HDI) [18], and LHN-I [39]. Local indices have low computational complexity suitable for large networks but limited accuracy due to insufficient information.

Path-based similarity indices include Local Path (LP), Katz [40], and LHN-II [39]. LHN-II assumes that if nodes connected to two nodes are similar, then the two nodes are also similar even without common neighbors. This method requires large amounts of reliable label attributes for training sets, resulting in poor scalability for unlabeled networks. Path-based similarity indices have increasing computational complexity as network scale and path length grow. When LP' s path length approaches infinity, it becomes equivalent to Katz, which considers all paths. Since high-order paths contribute little to similarity,

matrix inversion is used for computation. Sparse matrix inversion algorithms are typically employed.

Random walk-based similarity indices include Average Commute Time (ACT) [29], Cosine similarity (Cos+) [41], Local Random Walk (LRW) [42], and Superposed Random Walk (SRW) [42]. ACT assumes smaller average commute time between two nodes indicates closer proximity. Commute time is the average steps for a random particle to travel from one node to another and back. Cos+ uses Mahalanobis distance to measure vector dissimilarity. As ACT is a global random walk with high computational complexity, Liu et al. [42] proposed LRW, which considers random walks within limited steps, significantly reducing complexity. SRW builds on LRW by superposing previous results with the final step, giving adjacent nodes more opportunities to connect to target nodes, making it a realistic network-aware algorithm.

Other similarity-based methods include Matrix Forest Index (MFI) [43], Transitive Similarity with Consistent Neighborhood (TSCN) [44], Preferential Attachment (PA) [45], and Naive Bayes-based indices (LNBAA, LNBCN, LNBRA) [46]. Traditional common neighbor indices ignore neighbor weights, though different neighbors have varying influence. Liu et al. introduced a role weight function to calculate neighbor influence [46], incorporating it into AA, CN, and RA to propose Naive Bayes-based algorithms.

Since Moore and Newman' s 2008 *Nature* paper [47] and Redner' s commentary [48], link prediction has been a focus in complex network research with many successes. The above algorithms use statistical methods to obtain similarity values, with some showing excellent performance. Currently, some neural network-based algorithms can more efficiently obtain network feature vectors for various machine learning tasks, including link prediction. DeepWalk first introduced this idea to link prediction, demonstrating excellent performance on public real datasets. This paper verifies the feasibility of matrix factorization-based DeepWalk link prediction by leveraging the equivalence between DeepWalk and network feature matrix factorization. While other network representation learning methods like TADW [49], MMDW [50], and NEU [51] outperform DeepWalk, this study aims to achieve equivalent link prediction performance using matrix factorization rather than conducting horizontal comparisons of neural network-based methods.

2 Method

This paper uses matrix factorization-based DeepWalk to obtain vector representations for each network node, then constructs a node similarity matrix using cosine similarity. Finally, the similarity matrix is applied to link prediction, with AUC metrics calculated to verify the feasibility and effectiveness of the proposed LPMF algorithm. Before detailing the method, we first explain matrix factorization-based DeepWalk and prove that DeepWalk is equivalent to

factorizing a target matrix M, which forms the foundation of our approach.

2.1 Matrix Factorization Based DeepWalk Algorithm

SGNS is primarily applied in semantic networks where words have only contextual relationships determined by windows. SGNS collects (word, context) pairs and inputs them into a three-layer shallow neural network for training. Initially, each word is assigned an arbitrary vector representation, which is continuously adjusted as (word, context) pairs are processed through the neural network. SGNS effectively utilizes contextual information to train semantic associations between words.

Inspired by SGNS, DeepWalk modifies it to migrate from semantic networks to general networks like social networks. This generalization enables representation learning across various networks. The only change is context acquisition: SGNS uses sliding windows, while DeepWalk uses random walks; the underlying algorithm remains unchanged. Similarly, (current node, context node) pairs are input into a three-layer shallow neural network.

Subsequently, Levy et al. [52] proved that SGNS word embedding is equivalent to factorizing an SPPMI matrix, denoted as M, where k is the negative sampling count for each (word, context) pair, N is the total number of word occurrences in the training set, N(w) is the count of word w, N(c) is the count of context word c, and N(w,c) is the count of (word, context) pairs.

Inspired by SGNS, Yang et al. [28] mathematically proved that DeepWalk is similar to factorizing SGNS' s feature matrix, i.e., factorizing a target matrix M, where v represents network nodes (instead of words) and c represents context nodes obtained through random walks. For graph G = (V,E) with vertex set V and edge set E, let D be the set of (current node, context node) pairs generated from random walk sequences, where each element is a context node pair (v_i, c_j). Assuming random walk length is t, node v_i is visited N(v_i) times in set D. Since N(v_i) represents node occurrence frequency in random walks, it equals the node' s PageRank value. Additionally, N(v_i,c_j) represents occurrences of node v_i around node v_j within t steps. Defining the PageRank transition matrix as P and node degree as d_i, we can derive:

Let e_i be a |V|-dimensional row vector with 1 in column i and 0 elsewhere. Starting random walk from node v_i with initial state e_i, e_iP^t represents the spatial distribution after t steps, where the j-th element indicates the probability of walking from node v_i to v_j in t steps. Thus, N(v_i,c_j) = e_i(_{r=1}^t P^r)_j represents occurrences of node v_i around node v_j within t steps. The target matrix can be calculated as:

M = log(_{r=1}^t P^r) - log(t)

The time complexity for computing M is $O(|V|^3)$. In practice, DeepWalk uses random walk sampling to avoid accurately computing M, while matrix factorization methods must compute M for factorization. Yang et al. [28] balanced

speed and accuracy to obtain the factorization target matrix:

$M = (A + A^2)/2$

For dense networks, M contains more non-zero elements. Yu et al. [53] proved that with squared loss evaluation, factorization time complexity is proportional to non-zero elements. This paper improves efficiency by factorizing M rather than M^3. Therefore, algorithm complexity comprises two parts: constructing M and factorizing M. Constructing adjacency matrix A has $O(|V|^2)$ complexity. Using SVD to factorize the target matrix yields $O(|V|^3)$ complexity for the factorization part.

2.2 Link Prediction Based on Matrix Factorization

The proposed LPMF algorithm builds upon matrix factorization, using different factorization methods to obtain different network representations. Given a network' s edge representation, it can be converted to an adjacency matrix, from which the target matrix M for factorization is generated. This paper uses matrix methods to decompose M into three matrices. For matrix factorization, we employ the SVDS algorithm, which offers advantages over SVD: (a) SVDS is an SVD variant with reduced computational complexity; (b) it returns specified numbers of largest singular values and vectors; (c) it provides greater customizability and flexibility.

The LPMF framework is illustrated in Figure 2 [Figure 2: see original paper]. The framework consists of five steps:

a) Input a network edge set, split it into training and testing sets, convert the training set to adjacency matrix A, and compute the target matrix M = $(A + A^2)/2$.

b) Use SVDS to factorize M into three matrices: [U, S, V] = svds(M, k).

c) Multiply the decomposed matrices to obtain network representations: Representation = U × S^(1/2).

d) Compute cosine similarity between node pairs: s = sim(a,b) = (a·b)/(||a|| × ||b||), and build a |V|×|V| similarity matrix S.

e) Evaluate link prediction performance using AUC on the testing set.

SVDS represents a complex matrix as the product of three submatrices. For any m×n matrix M, there exists a decomposition: M U_k × S_k × V_k^T, where U_k is an m×k matrix of left singular vectors, S_k is a k×k diagonal matrix of singular values, and V_k^T is a k×n matrix of right singular vectors. Here, |V| represents the number of network nodes, and k is the number of features (vector length). SVDS is widely used in dimensionality reduction and recommendation systems.

The complete algorithm pseudocode is provided below:

Algorithm: LPMF (G, train-ratio, k)

Input:
Network edge set: G
Training ratio: training-ratio
Representation length: k

Output: AUC

1. Get edge set of network G
2. Count nodes, denoted as |V|
3. Split G into training and testing sets
4. Initialize adjacency matrix A for training set
5. Initialize target matrix M: $M = (A + A^2)/2$
6. Factorize M: [U, S, V] = svds(M, k)
7. Compute cosine similarity for each node pair: $s = sim(a,b) = (a \cdot b)/(||a|| \times ||b||)$
8. Build similarity matrix S
9. Compute AUC using testing set

---

3 Experiments

3.1 Experimental Setup

This paper uses three real-world citation network datasets: Citeseer, DBLP, and Cora. Detailed dataset information is shown in Table 1 . All three datasets have approximately 3,000 nodes but differ in edge counts. Citeseer and Cora have similar edge counts, while DBLP has about six times more edges. More edges yield higher network density and average degree. Despite large differences in edge counts, DBLP and Cora have similar network diameters and average path lengths. Based on average degree and density, Citeseer and Cora are sparse networks, while DBLP is dense.

Table 1: Dataset Descriptions

Dataset	Nodes	Edges	Classes	Avg. Degree	Diameter	Avg. Path Length	Density	Avg. Clustering Coefficient
Citeseer	3312	4732	6	2.86	28	9.32	0.0009	0.144
DBLP	3119	39516	4	25.34	8	5.26	0.0081	0.632
Cora	2708	5429	7	4.01	19	5.69	0.0015	0.238

Most link prediction algorithms listed in related work use statistical methods to obtain similarity values. The proposed LPMF uses a neural network-like approach to obtain network structural feature matrices, then applies matrix factorization to construct node similarities. Therefore, LPMF is comparable to the methods listed in related work. We set the representation vector length to 100 and training ratios to 0.7, 0.8, and 0.9. Results are shown in Table 2 .

Table 2: Link Prediction Results on Citeseer, DBLP, and Cora

Algorithm	Citeseer (0.7/0.8/0.9)	DBLP (0.7/0.8/0.9)	Cora (0.7/0.8/0.9)
Salton	66.32/72.73/74.44	86.00/87.92/90.74	69.38/72.13/77.89
Jaccard	66.51/72.25/74.33	85.92/88.26/90.98	69.25/72.00/77.09
LHN-I	66.47/72.93/74.46	85.80/87.87/89.95	69.19/72.16/77.30
LHN-II	95.76/96.85/96.20	90.86/91.80/92.80	89.41/90.37/93.64
LNBAA	66.37/72.64/74.52	86.07/88.42/91.12	69.42/72.50/78.01
LNBCN	66.70/72.27/74.25	85.60/88.47/90.80	69.50/72.19/77.79
LNBRA	66.05/72.23/74.27	85.86/88.91/91.23	69.32/72.84/77.74
ACT	75.88/75.59/73.79	79.00/80.07/80.84	74.11/73.67/74.00
Cos+	88.57/89.38/88.49	91.53/93.47/95.08	90.25/90.98/93.22
LRW	87.21/90.13/91.25	92.75/93.35/94.09	88.48/90.58/93.63
SRW	86.34/90.05/90.47	90.50/92.25/94.06	88.40/90.50/93.62
TSCN	84.26/85.68/86.27	91.25/91.03/92.34	88.35/90.64/92.98
LPMF	**87.18/90.64/94.98**	**93.42/94.70/95.13**	**89.57/92.13/93.93**

3.2 Experimental Results Analysis

LPMF is compared with 21 common link prediction algorithms. Analysis shows that on Citeseer, DBLP, and Cora, while LPMF effectively mines network features, MFI-based feature mining better captures network growth essence. LPMF performs similarly to Katz, which excels in networks with short average path lengths. LPMF can obtain more network features through longer random walks. Thus, Katz outperforms LPMF on Citeseer, while LPMF surpasses Katz on DBLP and Cora. Overall, LPMF outperforms the other 19 algorithms by fully utilizing network structural features.

3.3 Distribution Visualization

The most fundamental network parameter is vertex degree. Degree distribution (the frequency distribution of vertex degrees) is closely related to network topology and helps determine network type. For example, most networks are scale-free, with power-law distributions determined by degree distribution exponents. We visualized degree distributions for Citeseer, DBLP, and Cora using MATLAB, as shown in Figure 3 [Figure 3: see original paper].

Figure 3 shows degree values on the x-axis and occurrence frequency on the y-axis. In Cora, the maximum degree is below 170, but each degree value appears more frequently than in DBLP and Citeseer, with the highest degree appearing over 570 times. In Citeseer and DBLP, nodes with degree <50 appear frequently, while those with degree 50-200 appear rarely. Thus, most nodes have small degrees, with few high-degree nodes. Citeseer and Cora are not highly dense networks.

3.4 Parameter Tuning and Analysis

Two parameters require setting: vector length k and training ratio. The training ratio splits data for AUC calculation. During training, only the training set is converted to an adjacency matrix to compute the target matrix. Figure 4 [Figure 4: see original paper] shows parameter influence experiments.

With k=50, all datasets achieve worst AUC performance. With k=300, overall performance is best. Since Citeseer and Cora are sparse networks, AUC performs better at training ratio 0.9. DBLP is dense, showing minimal AUC variation across training ratios. AUC variation is larger on Citeseer and Cora than on DBLP. In summary, vector length and training ratio significantly impact sparse networks but have limited impact on dense networks.

3.5 Network Representation Visualization

We randomly selected 4 categories from each dataset, sampling 150 nodes per category (600 nodes total). T-SNE projected these nodes into 2D space, with different categories in different colors, as shown in Figure 5 [Figure 5: see original paper].

Figure 5 shows clear regional boundaries, indicating that node vectors trained by matrix factorization-based methods have distinct discriminative capabilities. Nodes with the same label (color) exhibit clear clustering, with similar nodes having closer distances in 2D projection. This demonstrates that matrix factorization-based network representation learning effectively captures network structure information, placing structurally similar nodes closer in representation space and dissimilar nodes farther apart. Visualization confirms that LPMF-trained node representations have clustering properties that enhance link prediction accuracy.

3.6 Case Study

DBLP is a citation network divided into 4 fields: databases (SIGMOD, ICDE, VLDB, etc.), data mining (KDD, ICDM, etc.), AI (IJCAI, AAAI, etc.), and computer vision (CVPR, ICCV, etc.). We randomly selected a target node with title "Querying Object-Oriented Databases," then computed cosine similarity to find its 5 most similar neighbors and extracted their titles. With vector length 100 and training ratio 0.9, results are shown in Table 3 .

Table 3: Case Study

Rank	Title	Similarity	Field
1	A Powerful and Simple Database Language	0.7476	Database
2	A General Framework for The Optimization of Object-Oriented Queries	0.7165	Database
3	Towards an Effective Calculus for Object Query Languages	0.7065	Database
4	A Functional Execution Model for Object Query Languages	0.7065	Database
5	A query Language for Multidimen-sional Arrays Design Imple-mentation and Optimization	0.7065	Database

The 5 most similar titles share high structural similarity with the target, all belonging to the database field. The target paper proposes a novel structured language for querying object-oriented databases. Therefore, citing or cited papers should satisfy at least one condition: "query language" or "database." The

returned titles contain these keywords, demonstrating that matrix factorization-based DeepWalk effectively mines structural correlations, enabling similar network structures to have closer spatial distances.

4 Conclusion

This paper proves that DeepWalk network representation learning is essentially matrix factorization. Based on this, we propose LPMF, a link prediction algorithm using matrix factorization-based DeepWalk. Experiments on Citeseer, DBLP, and Cora demonstrate LPMF' s excellent link prediction performance, surpassing most existing algorithms. Visualization experiments show that node representations trained by matrix factorization-based DeepWalk exhibit clear clustering. The case study proves that trained node vectors fully reflect network features, placing structurally similar nodes closer in space. In summary, LPMF is an effective and feasible algorithm for practical link prediction tasks. Future work includes: (1) integrating the algorithm with distributed frameworks like cloud computing for ultra-large-scale link prediction; (2) using matrix factorization algorithms that incorporate external information (e.g., Inductive Matrix Completion, Dependent Probabilistic Matrix Factorization) to better mine network features.

References

[1] Getoor L, Diehl C P. Link mining: a survey [J]. ACM SIGKDD Explorations Newsletter, 2005, 7(2): 3-12.
[2] Albert R, Barabasi A L. Statistical mechanics of complex networks [J]. Reviews of Modern Physics, 2002, 74(51): 47-97.
[3] Dorogovtsev S N, Mendes J F F. Evolution of networks [J]. Advances in Physics, 2002, 51(4): 1079-1187.
[4] Leicht E A, Holme P, Newman M E. Vertex similarity in networks [J]. Physical Review E: Statistical Nonlinear & Soft Matter Physics, 2006, 73(2): 026120.
[5] Zhang Luming, Gao Yue, Hon Chaoqun. Feature correlation hypergraph: exploiting high-order potentials for multimodal recognition [J]. IEEE Trans on Cybernetics, 2017, 44(8): 1408-1419.
[6] Yu Haiyuan, Braun P, Yildirim M A, et al. High-quality binary protein interaction map of the yeast interactome network [J]. Science, 2008, 322(5898): 104-110.
[7] Stumpf M P, Thorne T, De S E, et al. Estimating the size of the human interactome [J]. Proceedings of the National Academy of Sciences of the United States of America, 2008, 105(19): 6959-6964.
[8] Xie Xiaoqin, Li Yijia, Zhang Zhiqiang, et al. A joint link prediction method for social network [C]//Proc of International Conference of Young Computer

Scientists, Engineers and Educators. Berlin: Springer, 2015.
[9] Schafer L, Graham J W. Missing data: our view of the state of the art [J]. Psychol. Methods, 2007, 7(2): 147-152.
[10] Kossinets G. Effects of missing data in social networks [J]. Social Networks, 2003, 28(3): 247-268.
[11] Kumar R, Novak J, Tomkins A. Structure and evolution of online social networks [M]//Link Mining: Models, Algorithms, and Applications. New York: Springer, 2006: 337-357.
[12] Gallagher B, Tong Hanghang, Eliassi-Rad T, et al. Using ghost edges for classification in sparsely labeled networks [C]//Proc of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM Press, 2008: 256-264.
[13] Dasgupta K, Singh R, Viswanathan B, et al. Social ties and their relevance to churn in mobile telecom networks [C]//Proc of the 11th International Conference on Extending Database Technology. 2008.
[14] Zadeh P M, Kobti Z. A knowledge based framework for link prediction in social networks [C]//Proc of International Symposium on Foundations of Information and Knowledge Systems. Austria: Springer, 2016: 255-268.
[15] Zhang Xue, Zhao Chengli, Wang Xiaojie, et al. Identifying missing and spurious interactions in directed networks [M]//Wireless Algorithms, Systems, and Applications. Harbin: Springer, 2014: 470-481.
[16] Guimerà R, Salespardo M. Missing and spurious interactions and the reconstruction of complex networks [J]. Proceedings of the National Academy of Sciences of the United States of America, 2009, 106(52): 22073-22078.
[17] Mering C V, Krause R, Snel B, et al. Comparative assessment of large-scale data sets of protein interactions [J]. Nature, 2002, 417(6887): 399-403.
[18] Zhou Tao, Liu Linyuan, Zhang Yicheng. Predicting missing links via local information [J]. European Physical Journal B, 2009, 71(4): 623-630.
[19] Leskovec J, Huttenlocher D, Kleinberg J. Predicting positive and negative links in online social networks [C]//Proc of International Conference on World Wide Web. New York: ACM Press, 2010: 641-650.
[20] Barabási A, Albert R. Emergence of scaling in random networks [J]. Science, 1999, 286(5439): 509-512.
[21] Garlaschelli D, Capocci A, Caldarelli G. Self-organized network evolution coupled to extremal dynamics [J]. Nature Physics, 2008, 3(11): 813-817.
[22] Liben-Nowell D, Kleinberg J. The link-prediction problem for social networks [J]. Journal of the Association for Information Science & Technology, 2007, 58(7): 1019–1031.
[23] Bianconi G. Entropy of network ensembles [J]. Physical Review E: Statistical Nonlinear & Soft Matter Physics, 2009, 79(2): 036114.
[24] Liu Weiping, Lu Linyuan. Link prediction based on local random walk [J]. Europhysics Letters, 2010, 89(5): 58007-58012.
[25] Mikolov T, Chen Kai, Corrado G, et al. Efficient estimation of word representations in vector space [EB/OL]. [2017-09-05] https://arxiv.org/abs/1301.3781.
[26] Mikolov T, Sutskever I, Chen Kai, et al. Distributed representations of words and phrases and their compositionality [J]. Advances in Neural

Information Processing Systems, 2013, 26(14): 3111-3119.
[27] Berozzi B, Al-Rfou R, Skiena S. DeepWalk: online learning of social representations [C]//Proc of ACM SIGKDD Conference on Knowledge Discovery and Data Mining. New York: ACM Press, 2014: 701-710.
[28] Yang Cheng, Liu Zzhiyuan. Comprehend deepwalk as matrix factorization [EB/OL]. [2017-09-15] https://arxiv.org/abs/1301.3781.
[29] Klein D J, Randic M. Resistance distance [J]. Journal of Mathematical Chemistry, 1993, 12(1): 81-95.
[30] Brin S, Page L. The anatomy of a large-scale hypertextual Web search engine [J]. computer network and ISDN system, 1998, 30(1): 107-117.
[31] Liu Weiping, Lü Linyuan. Link prediction based on local random walk [J]. Europhysics letters, 2010, 89(5): 58007-58012.
[32] François Lorrain, Harrison C. White. Structural equivalence of individuals in social networks [J]. Social Networks, 1977, 1(1): 67-98.
[33] Casey R G C B. Friends and neighbors [J]. Foreign Affairs, 2005, 46(3): 45-58.
[34] Zhou Tao, Lü Lingyuan, Zhang Yicheng. Predicting missing links via local information [J]. European Physical Journal B, 2009, 71(4): 623-630.
[35] Salton G, McGill M J. Introduction to modern information retrieval [J]. Program, 2004, 55(3): 239-240.
[36] Jaccard P. Etude de la distribution florale dans une portion des Alpes et du Jura [J]. Bulletin De La Societe Vaudoise Des Sciences Naturelles, 1901, 37(142): 547-579.
[37] Sorensen T. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on Danish commons [J]. Biologiske Skrifter, 1957, 5(4): 1-34.
[38] Ravasz E, Somera A L, Mongru D A, et al. Hierarchical organization of modularity in metabolic networks [J]. Science, 2002, 297(5586): 1551-1555.
[39] Leicht E A, Holme P, Newman M E. Vertex similarity in networks [J]. Physical Review E Statistical Nonlinear & Soft Matter Physics, 2006, 73(2): 116-120.
[40] Katz L. A new status index derived from sociometric analysis [J]. Psychometrika, 1953, 18(1): 39-43.
[41] Fouss F, Pirotte A, Renders J M, et al. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation [J]. IEEE Trans on Knowledge & Data Engineering, 2007, 19(3): 355-369.
[42] Li Ling. Link prediction based on random walks [J]. Journal of Computational Information Systems, 2015, 11(5): 1757-1764.
[43] Chebotarev P, Shamis E. The matrix-forest theorem and measuring relations in small social groups [J]. Automation & Remote Control, 2006, 58(9): 1505-1514.
[44] Sun Duo, Zhou Tao, Liu Jianguo, et al. Information filtering based on transferring similarity [J]. Physical Review E Statistical Nonlinear & Soft Matter Physics, 2009, 80(2): 017101.
[45] Fortunato S, Flammini A, Menczer F. Scale-free network growth by ranking

[J]. Physical Review Letters, 2006, 96(21): 218701.
[46] Lü Linyuan. Link prediction in complex networks: a local naive Bayes model [J]. Europhysics Letters, 2011, 96(4): 48007.
[47] Clauset A, &Amp C M, Newman M E J. Hierarchical structure and the prediction of missing links in networks [J]. Nature, 2008, 453(7191): 98-101.
[48] Redner S. Networks: teasing out the missing links [J]. Nature, 2008, 453(7191): 47-48.
[49] Yang Cheng, Liu Zhiyuan, Zhao Deli, et al. Network representation learning with rich text information [C]//Proc of International Joint Conference on Artificial Intelligence. San Francisco: Morgan Kaufmann, 2015: 2111-2117.
[50] Tu Ccunchao, Zhang Weicheng, Liu Zhiyuan, et al. Max-margin deepwalk: discriminative learning of network representation [C]//Proc of International Joint Conference on Artificial Intelligence. San Francisco: Morgan Kaufmann, 2016: 3889-3895.
[51] Yang Cheng, Sun Maosun, Liu Zhiyuan, et al. Fast network embedding enhancement via high order proximity approximation [C]//Proc of International Joint Conference on Artificial Intelligence. San Francisco: Morgan Kaufmann, 2017: 3894-3900.
[52] Levy O, Goldberg Y. Neural word embedding as implicit matrix factorization [J]. Advances in Neural Information Processing Systems, 2014, 3(17): 2177-2185.
[53] Yu H F, Jain P, Kar P, et al. Large-scale multi-label Learning with Missing Labels [C]//Proc of International Conference on Machine Learning. New York: ACM Press, 2013: 593-601.

Note: Figure translations are in progress. See original paper for figures.