Electronic Preprint for Journal of Information Processing Vol.25

Regular Paper

Link Prediction in Sparse Networks by Incidence Matrix Factorization

Sho Yokoi[1,a)] Hiroshi Kajino[2] Hisashi Kashima[3]

Received: October 12, 2016, Accepted: April 10, 2017

Abstract: Link prediction plays an important role in multiple areas of artificial intelligence, including social network analysis and bioinformatics; however, it is often negatively affected by the data sparsity problem. In this paper, we present and validate our hypothesis, i.e., for sparse networks, incidence matrix factorization (IMF) could perform better than adjacency matrix factorization (AMF), the latter used in many previous studies. A key observation supporting our hypothesis here is that IMF models a partially observed graph more accurately than AMF. Unfortunately, a technical challenge we face in validating our hypothesis is that there is not an obvious method for making link prediction using a factorized incidence matrix, unlike the AMF approach. To this end, we developed an optimization-based link prediction method. Then we have conducted thorough experiments using both synthetic and real-world datasets to investigate the relationship between the sparsity of a network and the predictive performance of the aforementioned two factorization approaches. Our experimental results show that IMF performed better than AMF as networks became sparser, which validates our hypothesis.

Keywords: link prediction, data sparsity problem, matrix factorization, incidence matrix

1. Introduction

Link prediction is the problem of graph mining or network analysis that attempts to predict a link between two nodes based on other observed links and attributes of nodes [4], [8], [17]. There are numerous applications for link prediction, such as recommender systems in the social sciences [15] and protein-protein interaction analysis systems in bioinformatics [24]. In this paper, we focus on link prediction based on a graph structure, which is formulated as a pairwise classification problem as follows. Given partially observed graph $G = (\mathcal{V}, \mathcal{E}_\mathrm{P})$ with the set of nodes $\mathcal{V}$ and the set of *positive links* (i.e., observed links) $\mathcal{E}_\mathrm{P} \subset \mathcal{V}\times\mathcal{V}$, the goal is to learn scoring function $s\colon \mathcal{V}\times\mathcal{V} \to \mathbb{R}$ that predicts a new link on an *unlabeled pair of nodes* in a set $\mathcal{E}_\mathrm{U} := (\mathcal{V}\times\mathcal{V}) \setminus \mathcal{E}_\mathrm{P}$.

As pointed out by many researchers, a central issue in link prediction lies in the *sparsity* of a graph [7], [16], [22]. As networks grow, the number of node pairs that can be linked increases quadratically, whereas the number of actual links often grows only linearly, which degrades the predictive performance of most existing methods [22].

Our idea for countering this sparsity problem is to employ incidence matrix factorization (**IMF**) as a building block of our link prediction method, instead of adjacency matrix factorization (**AMF**), which has been used in a number of previous studies [1], [6], [14], [15], [19]. A key observation supporting our approach here is that IMF can model a partially observed graph more accurately than AMF. We begin by briefly introducing the AMF approach illustrated in **Fig. 1**, [Top]. Given partially observed graph $G = (\mathcal{V}, \mathcal{E}_\mathrm{P})$, AMF learns latent feature vectors of nodes $\{\boldsymbol{x}_k\}_{v_k\in\mathcal{V}}$ by factorizing its adjacency matrix $A \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$, using both positive links (they are represented as ones in the matrix) and unlabeled node pairs (they are represented as zeros) such that

$$\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle \approx \begin{cases} 1 & \text{if } (v_i, v_j) \in \mathcal{E}_\mathrm{P} \\ 0 & \text{if } (v_i, v_j) \notin \mathcal{E}_\mathrm{P} \end{cases}$$

holds in its simplest instantiation; here, $\langle\cdot,\cdot\rangle$ denotes the standard inner product. This approach has a flaw in that the model learned

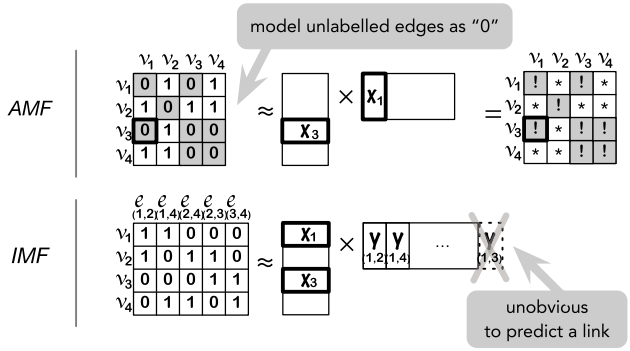

Fig. 1 [Top] An example of link prediction using AMF. Here we learn latent vector $\boldsymbol{x}_k$ for any node $v_k \in \mathcal{V}$, then predict whether v_1 and v_3 are linked based on the magnitude of $\langle \boldsymbol{x}_1, \boldsymbol{x}_3 \rangle$. This model has a flaw in that unlabeled node pairs are modeled as zero.
[Bottom] Even though IMF is promising for accurately capturing a partially observed graph, it is not trivial to predict a link in this direction. By factorizing the incidence matrix, we learn latent vector $\boldsymbol{y}_{(i,j)}$ for any positive link $e_{(i,j)} \in \mathcal{E}_\mathrm{P}$ in addition to latent vector $\boldsymbol{x}_k$ for any node. Predicting a link between v_1 and v_3 requires a latent vector of the unlabeled pair of nodes $(v_1, v_3) \notin \mathcal{E}_\mathrm{P}$, which we cannot obtain through factorization of the incidence matrix.

1 Tohoku University, Sendai, Miyagi 980–8579, Japan
2 IBM Research - Tokyo, Chuo, Tokyo 103–8510, Japan
3 Kyoto University, Kyoto 606–8501, Japan
a) yokoi@ecei.tohoku.ac.jp

from a partially observed graph is inconsistent with the model learned from its fully observed graph. Consider pair of nodes (v_i, v_j) that is not linked in a partially observed graph, but is instead actually positive in its fully observed graph. In the ideal case, latent vectors of v_i and v_j obtained from the partially observed graph satisfy $\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle \approx 0$, whereas those obtained from the fully observed graph satisfy $\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle \approx 1$. As the observation of a graph becomes sparser, the number of such links increases; therefore, this inconsistency can lead to poor performance. Conversely, IMF can avoid this inconsistency because it learns a model by only utilizing positive links. More specifically, IMF learns latent feature vectors of nodes $\{\boldsymbol{x}_k\}_{v_k \in \mathcal{V}}$ and those of positive links $\{\boldsymbol{y}_l\}_{e_l \in \mathcal{E}_\mathrm{P}}$ by factorizing incidence matrix $B \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{E}_\mathrm{P}|}$ such that

$$\langle \boldsymbol{x}_i, \boldsymbol{y}_j \rangle \approx \begin{cases} 1 & \text{if } v_i \in e_j \\ 0 & \text{if } v_i \notin e_j \end{cases}$$

holds in its simplest instantiation. Since this approach does not utilize unlabeled node pairs, the model obtained from a partially observed graph is consistent with the model obtained from a fully observed graph; therefore, the performance of IMF is expected to be robust to the sparsity of the given graph. In this light, we arrive at our hypothesis that IMF can counter the sparsity problem better than AMF. The main purpose of this paper is to confirm our hypothesis by a thorough set of experiments.

While the IMF approach is promising, it is not trivial to predict a new link using a factorized incidence matrix as shown in Fig. 1, [Bottom]. By factorizing the incidence matrix of a graph, we learn latent vector $\boldsymbol{x}_k$ of each node $v_k \in \mathcal{V}$ and latent vector $\boldsymbol{y}_{(i,j)}$ of a *positive link* $(v_i, v_j) \in \mathcal{E}_\mathrm{P}$, but we do not learn a latent vector $\boldsymbol{y}_{(i',j')}$ of any *unlabeled node pair* $(v_{i'}, v_{j'}) \notin \mathcal{E}_\mathrm{P}$; therefore, we cannot predict a link using an approach similar to that of AMF. We thereby face a key technical challenge, i.e., how can we utilize the factorized incidence matrix to predict a link? Our idea here is that the link on $(v_{i'}, v_{j'})$ is expected if we can successfully recover its latent vector $\boldsymbol{y}_{(i',j')}$ that is consistent with latent vectors of positive links $\{\boldsymbol{y}_{(i,j)}; (v_i, v_j) \in \mathcal{E}_\mathrm{P}\}$. Given the above, we propose an optimization-based link prediction method that adopts the IMF approach; we describe our method in Section 3. Although the computational cost of IMF is generally more expensive than that of AMF, our method is faster with a simple contrivance, as further detailed in Section 3.3.

We employed the simplest methods for adopting these approaches rather than devising a new model, in order to conduct comparative experiments with large datasets to truly confirm our hypothesis. Moreover, we only focused on the graph structure; i.e., we did not incorporate domain-specific side information into our model to highlight the property of IMF as a building block for link prediction versus that of AMF. In our experiments, we first applied the two methods to synthetic datasets generated by the Barabási–Albert model [5], which is a well-known generative model of graphs. This model enabled us to obtain graphs with scale-free and small-world properties. Next, we applied the two methods to real-world datasets from KONECT [12], which is a repository of a large number of real-world networks. Our experimental results on both the synthetic and real-world datasets demonstrate that performance improvements achieved by the IMF-based method versus the AMF-based method are negatively correlated with the density of the given graphs; in other words, IMF indeed alleviates the sparsity problem.

2. Preliminaries

In this section, we first present our definition of the link prediction problem as a binary classification/ranking problem. Next, we provide formal definitions of the incidence matrix, adjacency matrix, and truncated singular value decomposition (SVD), the latter being a standard method of matrix factorization and low-rank approximation. Finally, we introduce a baseline link prediction method that utilizes the AMF approach.

2.1 Link Prediction

Given partially observed graph $G = (\mathcal{V}, \mathcal{E}_\mathrm{P})$ with the set of nodes $\mathcal{V}$ and the set of positive links $\mathcal{E}_\mathrm{P} \subset \mathcal{V} \times \mathcal{V}$, the goal of the *link prediction* problem is to learn scoring function $s\colon \mathcal{V} \times \mathcal{V} \to \mathbb{R}$ for predicting a new link on an unlabeled pair of nodes in a set $\mathcal{E}_\mathrm{U} := (\mathcal{V} \times \mathcal{V}) \setminus \mathcal{E}_\mathrm{P}$.

2.2 Matrix Representations of a Graph

Adjacency matrix $A = (a_{ij}) \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ of partially observed graph $G = (\mathcal{V}, \mathcal{E}_\mathrm{P})$ is defined as

$$a_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \in \mathcal{E}_\mathrm{P} \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

An adjacency matrix is symmetric here because we focus on undirected graphs.

Next, *incidence matrix* $B = (b_{ij}) \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{E}_\mathrm{P}|}$ of graph $G = (\mathcal{V}, \mathcal{E}_\mathrm{P})$ is defined as

$$b_{ij} = \begin{cases} 1 & \text{if } v_i \in e_j \\ 0 & \text{otherwise.} \end{cases}$$

Here, any column vector $\boldsymbol{b} \in \mathbb{R}^{|\mathcal{V}|}$ of B represents a positive link. By definition, any $\boldsymbol{b}$ has exactly two ones with remaining elements set to zeros because any positive link is incident with exactly two nodes. Note that we exploit this feature in our method, which we present in Section 3.1.

2.3 Truncated SVD

Given matrix $M \in \mathbb{R}^{m \times n}$ and integer $k \leq \operatorname{rank} M$, *truncated SVD* allows us to factorize the matrix approximately into the product of three matrices $U_k \in \mathbb{R}^{m \times k}$, $\Sigma_k \in \mathbb{R}^{k \times k}$, and $V_k \in \mathbb{R}^{n \times k}$ such that

$$M \approx U_k \Sigma_k {V_k}^\top =: \tilde{M},$$

where Σ_k denotes a diagonal matrix $\operatorname{diag}(\sigma_1, \ldots, \sigma_k)$ that consists of the k largest singular values of M. Matrix $\tilde{M}$ is the best rank-k approximation of M in terms of the Frobenius norm, i.e.,

$$\tilde{M}_k = \operatorname*{arg\,min}_{\tilde{M} \in \mathbb{R}^{m \times n}} \left\| M - \tilde{M} \right\|_\mathrm{F}^2 \text{ s.t. } \operatorname{rank} \tilde{M} \leq k.$$

2.4 AMF-based Method

In this subsection, we introduce a baseline method that utilized the AMF approach. This method is composed of the following three steps:

(1) Represent given graph $G = (\mathcal{V}, \mathcal{E}_\mathrm{P})$ by adjacency matrix $A \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$.

(2) Factorize adjacency matrix A via truncated SVD, i.e.,

$$A \approx U_k \Sigma_k {V_k}^\top = XX^\top =: \tilde{A} = (\tilde{a}_{ij}), \tag{2}$$

where $X = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{|\mathcal{V}|})^\top = U_k \Sigma_k^{1/2} = V_k \Sigma_k^{1/2}$ and $\Sigma_k^{1/2} := \mathrm{diag}(\sqrt{\sigma_1}, \ldots, \sqrt{\sigma_k})$. Note that V_k equals U_k because adjacency matrix A is symmetric. Each row vector ${\boldsymbol{x}_i}^\top$ of matrix X denotes a latent vector of node $v_i \in \mathcal{V}$.

(3) Predict a link by scoring function $s_\mathrm{AMF}\colon \mathcal{V}\times\mathcal{V} \to \mathbb{R}$, which is defined as

$$s_\mathrm{AMF}(v_i, v_j) := \tilde{a}_{ij} = \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle,$$

where $\langle \cdot, \cdot \rangle$ denotes the standard inner product.

3. IMF-based Method

In this section, we propose an optimization-based link prediction method that adopts the IMF approach. As we noted out in Section 1 above, IMF is expected to counter the sparsity problem substantially better than AMF; however, a technical challenge we face in implementing IMF is that, unlike AMF, there is not an obvious method for making link predictions using a factorized incidence matrix. IMF only provides latent vectors of nodes and positive links; i.e., IMF lacks those of unlabeled pairs of nodes, which are necessary for link prediction, as illustrated in Fig. 1, [Bottom]. To address this challenge, we propose that a link between an unlabeled node pair is expected if we can successfully recover its latent vector that is consistent with those of the positive links. Our approach here is simple, fast, and appropriate for conducting a large number of comparative experiments to confirm our hypothesis.

3.1 Overview

Figure 2 shows an overview of our method. Given incidence matrix B of the given graph, IMF first factorizes matrix $B \approx XY$ using truncated SVD, which provides us latent vectors of nodes $\{\boldsymbol{x}_k\}_{v_k\in\mathcal{V}}$ and those of positive links $\{\boldsymbol{y}_{(i,j)}\}_{(v_i,v_j)\in\mathcal{E}_\mathrm{P}}$ such that $\boldsymbol{b}_{(i,j)} \approx X\boldsymbol{y}_{(i,j)}$ for any positive link (v_i, v_j), where $\boldsymbol{b}_{(i,j)} := (0, \ldots, 0, \underset{i}{1}, 0, \ldots, 0, \underset{j}{1}, 0, \ldots, 0)^\top$ is a column vector of B. Our idea here is to predict a link on unlabeled node pair (v'_i, v'_j) based on how well we can recover its latent vector $\boldsymbol{y}_{(i',j')}$ that is consistent with the positive links, i.e.,

$$\boldsymbol{b}_{(i',j')} \approx X\boldsymbol{y}_{(i',j')}.$$

Since we have matrix X and binary vector $\boldsymbol{b}_{(i',j')}$ (whose i'th and j'th elements are one, with remaining elements set to zero), we formulate our above idea as following scoring function

$$s_\mathrm{IMF}(v_i, v_j) := -\min_{\boldsymbol{y}\in\mathbb{R}^k} \left\|\boldsymbol{b}_{(i,j)} - X\boldsymbol{y}\right\|_2^2.$$

Here

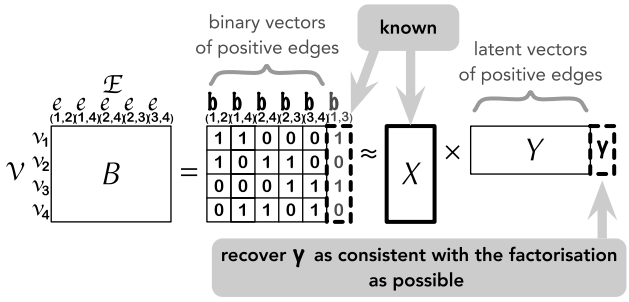

Fig. 2 An overview of our link prediction method based on IMF. For any positive link $(v_i, v_j) \in \mathcal{E}_\mathrm{P}$, equation $\boldsymbol{b}_{(i,j)} \approx X\boldsymbol{y}_{(i,j)}$ holds. For unlabeled node pair $(v_1, v_3) \notin \mathcal{E}_\mathrm{P}$, if we can find latent vector $\boldsymbol{y}_{(1,3)}$ such that $\boldsymbol{b}_{(1,3)} \approx X\boldsymbol{y}_{(1,3)}$, then we consider there to be a link between v_1 and v_3. Note that matrix X and binary vector $\boldsymbol{b}_{(1,3)}$ are known.

$$\nabla_{\boldsymbol{y}} \left\|\boldsymbol{b}_{(i,j)} - X\boldsymbol{y}\right\|_2^2 = 0 \iff \boldsymbol{y} = \left(X^\top X\right)^{-1} X^\top \boldsymbol{b}_{(i,j)},$$

because

$$\begin{aligned}\nabla_{\boldsymbol{y}} \left\|\boldsymbol{b}_{(i,j)} - X\boldsymbol{y}\right\|_2^2 &= \nabla_{\boldsymbol{y}} \left(\boldsymbol{y}^\top (X^\top X)\boldsymbol{y} - 2(X^\top \boldsymbol{b}_{(i,j)})^\top \boldsymbol{y} + 2\right) \\ &= 2X^\top X\boldsymbol{y} - 2X^\top \boldsymbol{b}_{(i,j)}.\end{aligned}$$

Thus, the optimization problem can be solved in a closed form as

$$s_\mathrm{IMF}(v_i, v_j) = -\langle \boldsymbol{w}_i + \boldsymbol{w}_j, \boldsymbol{w}_i + \boldsymbol{w}_j \rangle, \tag{3}$$

where $\boldsymbol{w}_i$ is the i'th column of matrix W, which is given as

$$W := X(X^\top X)^{-1} X^\top - I_{|\mathcal{V}|} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}, \tag{4}$$

and $I_{|\mathcal{V}|}$ is the $|\mathcal{V}| \times |\mathcal{V}|$ identity matrix.

3.2 Algorithm

Similar to the AMF-based method, the procedure of our IMF-based method is composed of the three steps that follow:

(1) Represent given graph $G = (\mathcal{V}, \mathcal{E}_\mathrm{P})$ by incidence matrix $B \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{E}_\mathrm{P}|}$.

(2) Factorize incidence matrix B into the product of two matrices X and Y using truncated SVD, i.e.,

$$B \approx U_k \Sigma_k {V_k}^\top = XY^\top, \tag{5}$$

where $X := U_k\Sigma_k$ and $Y := V_k$. Next, calculate matrix $W \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ (i.e., Eq. (4)).

(3) Predict a link by using scoring function $s_\mathrm{IMF}\colon \mathcal{V}\times\mathcal{V} \to \mathbb{R}$ in conjunction with the i'th and j'th columns of matrix W (i.e., Eq. (3)).

3.3 Computational Efficiency

At first glance, the computational cost of IMF appears to be more expensive than that of AMF because the size of incidence matrix ($|\mathcal{V}| \times |\mathcal{E}_\mathrm{P}|$) is larger than that of adjacency matrix ($|\mathcal{V}| \times |\mathcal{V}|$); however, with a simple contrivance, the cost of the matrix factorization of our method can be as small as that of the AMF-based method. Observing that we only need matrices U_k and Σ_k of Eq. (5), it is sufficient to apply truncated SVD to positive semi-definite symmetric matrix $BB^\top$. We then obtain U_k and Σ_k as

$$U_k = Q_k, \tag{6}$$

$$\Sigma_k = \Lambda_k^{1/2}, \tag{7}$$

where

$$BB^\top \approx Q_k \Lambda_k {Q_k}^\top. \tag{8}$$

Since the size of $BB^\top$ is the same as adjacency matrix A, the computation time required for matrix factorization in our method is the same as that of AMF.

Moreover, the construction of matrix $BB^\top$ requires almost the same computation time as that of constructing adjacency matrix A because

$$BB^\top = A + D \tag{9}$$

holds, where D denotes diagonal matrix $\mathrm{diag}(d_1, \ldots, d_{|\mathcal{V}|})$, and each d_i corresponds to the degree of node v_i.

4. Experiments

To demonstrate that IMF actually counters the sparsity problem better than AMF, we conducted comparative experiments using large datasets. To examine the capabilities of IMF versus that of AMF as a building block in link prediction, we employed the simplest methods described above. We used synthetic datasets in our first set of experiments, then used real-world datasets in our second set of experiments. Further, we conducted comparative experiments to compare IMF with AMF in terms of the computation times required to learn the given models so as to clarify the usefulness of IMF in practical applications.

4.1 Datasets

In the subsection that folow, we introduce the datasets we used in our experiments.

4.1.1 Synthetic Datasets

In the first set of experiments, we generated synthetic graphs that possess scale-free and small-world properties by utilizing the Barabási–Albert preferential attachment model [5]. Further, we used the graph generator available in NetworkX, a Python language software package [9]. We generated 10 synthetic graphs using parameters $n = 10,000$ and $k \in \{1, 2, 3, 4, 5, 7, 10, 15, 25, 100\}$, where n denotes the number of nodes, and k denotes the number of links between a new node and existing nodes; i.e., each generated graph has approximately $k \times n$ positive links.

4.1.2 Real-World Datasets

In the second set of experiments, we used real-world datasets from KONECT (the Koblenz Network Collection) [12], which is a de facto online graph repository that provides a large number of real-world graphs. We extracted all unweighted and undirected graphs from KONECT, and from this group, we selected the 24 smallest graphs in terms of $|\mathcal{V}|$. **Table 1** summarizes all of the datasets we used in the second set of experiments.

4.2 Predictive Performance

In this subsection, we describe the experiments to examine the relationship between predictive performance and graph sparsity, all of which supports the validation of our hypothesis that IMF performs better than AMF as networks become sparser.

Table 1 Statistics for real-world datasets extracted from KONECT, sorted by the Sparsity measure ($|\mathcal{E}_\mathrm{P}|/|\mathcal{V}|^2$), and AUC scores (complete). Here, $|\mathcal{V}|$ represents the number of nodes and $|\mathcal{E}_\mathrm{P}|$ represents the number of positive links. For each row, the best result is shown in bold.

Name	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}_\mathrm{P}\rvert$	Sparsity	AMF	IMF
Zebra	27	111	1.5e-01	**0.867**	0.748
Jazz musicians	198	2,742	7.0e-02	**0.913**	0.740
Zachary karate club	34	78	6.7e-02	0.628	**0.766**
Contiguous USA	49	107	4.5e-02	**0.676**	0.448
Dolphins	62	159	4.1e-02	**0.714**	0.580
David Copperfield	112	425	3.4e-02	**0.716**	0.715
Reactome	6,327	147,547	3.7e-03	**0.978**	0.890
arXiv hep-ph	28,093	4,596,803	5.8e-03	**0.988**	0.907
PDZBase	212	244	5.4e-03	0.576	**0.694**
arXiv hep-th	22,908	2,673,133	5.1e-03	**0.963**	0.905
U. Rovira i Virgili	1,133	5,451	4.2e-03	**0.819**	0.760
Hamsterster friendships	1,858	12,534	3.6e-03	**0.870**	0.847
Hamsterster full	2,426	16,631	2.8e-03	**0.890**	0.815
Euroroad	1,174	1,417	1.0e-03	**0.530**	0.492
Human protein (Vidal)	3,133	6,726	6.9e-04	0.677	**0.745**
Protein	1,870	2,277	6.5e-04	0.575	**0.687**
arXiv astro-ph	18,771	198,050	5.6e-04	**0.937**	0.840
Facebook (NIPS)	2,888	2,981	3.6e-04	0.523	**0.996**
Route views	6,474	13,895	3.3e-04	0.621	**0.836**
US power grid	4,941	6,594	2.7e-04	**0.563**	0.519
Pretty Good Privacy	10,680	24,316	2.1e-04	0.755	**0.780**
Facebook friendships	63,731	817,035	2.0e-04	**0.880**	0.864
CAIDA	26,475	53,381	7.6e-05	0.655	**0.941**
Brightkite	58,228	214,078	6.3e-05	0.764	**0.872**

4.2.1 Performance Measure

To quantitatively measure performance, we used the area under the receiver operating characteristic curve (ROC-AUC, AUC) to evaluate the performance of scoring function s.

In general, for a binary classification problem, given a set of positive examples Δ_p and a set of negative examples Δ_n in a test set, the AUC of scoring function $s\colon \Delta \to \mathbb{R}$ is calculated as

$$\frac{1}{|\Delta_\mathrm{p}||\Delta_\mathrm{n}|} \sum_{x_\mathrm{p} \in \Delta_\mathrm{p}, x_\mathrm{n} \in \Delta_\mathrm{n}} \mathbb{I}[s(x_\mathrm{p}) > s(x_\mathrm{n})],$$

where $\mathbb{I}[\mathrm{condition}] = 1$ if the condition is true and $\mathbb{I}[\mathrm{condition}] = 0$ otherwise.

In the link prediction problem for a partially observed graph, since negative examples do not exist, we substitute a set of randomly selected $\min\{1,000, |\mathcal{E}_\mathrm{U}|\}$ unlabeled node pairs $\mathcal{E}_\mathrm{U}^{(1,000)} \subset \mathcal{E}_\mathrm{U}$ as a set of negative examples. With a test set $\mathcal{E}_\mathrm{P}^{(\mathrm{test})} \subset \mathcal{E}_\mathrm{P}$ as a set of positive examples, AUC is calculated as

$$\frac{1}{|\mathcal{E}_\mathrm{P}^{(\mathrm{test})}||\mathcal{E}_\mathrm{U}^{(1,000)}|} \sum_{e_\mathrm{p} \in \mathcal{E}_\mathrm{P}^{(\mathrm{test})}, e_\mathrm{n} \in \mathcal{E}_\mathrm{U}^{(1,000)}} \mathbb{I}[s(e_\mathrm{p}) > s(e_\mathrm{n})].$$

Lichtenwalter et al. noted that AUC is an appropriate performance measure for link prediction because it does not rely on an arbitrary or unjustified threshold [16].

4.2.2 Experimental Procedures

We conducted five fold cross validation to measure the performance of IMF and AMF using the following protocol.

(1) Given partially observed graph $G = (\mathcal{V}, \mathcal{E}_\mathrm{P})$, we randomly divide G into
- a training set $G^{(\mathrm{train})} = \left(\mathcal{V}, \mathcal{E}_\mathrm{P}^{(\mathrm{train})}\right)$,
- a development set $G^{(\mathrm{dev})} = \left(\mathcal{V}, \mathcal{E}_\mathrm{P}^{(\mathrm{dev})}\right)$,

Table 2 AUC scores on the synthetic datasets sorted by sparsity measure ($|\mathcal{E}_\mathrm{P}|/|\mathcal{V}|^2$). Here, n and k correspond to the parameters of the Barabási–Albert model introduced in Section 4.1.1. For each row, the best result is shown in bold.

n	k	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}_\mathrm{P}\rvert$	Sparsity	AMF	IMF
10,000	100	10,000	989,999	9.9e-03	0.722	**0.723**
10,000	25	10,000	249,374	2.5e-03	0.712	**0.727**
10,000	15	10,000	149,774	1.5e-03	0.693	**0.723**
10,000	10	10,000	99,899	1.0e-03	0.674	**0.722**
10,000	7	10,000	69,950	7.0e-04	0.657	**0.713**
10,000	5	10,000	49,974	5.0e-04	0.629	**0.711**
10,000	4	10,000	39,983	4.0e-04	0.610	**0.707**
10,000	3	10,000	29,990	3.0e-04	0.571	**0.708**
10,000	2	10,000	19,995	2.0e-04	0.533	**0.698**
10,000	1	10,000	9,998	1.0e-04	0.495	**0.682**

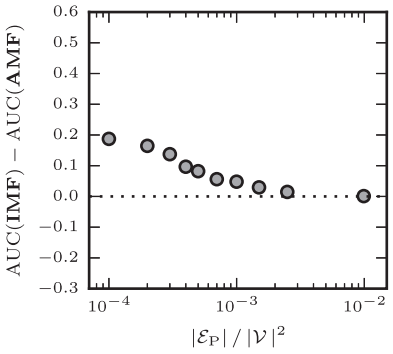

Fig. 3 A scatter plot illustrating the relationship between sparsity (x-axis) and the performance improvements of IMF over AMF (y-axis). Each point corresponds to a result for each synthetic dataset. Here, Spearman's $\rho = -1.0$ ($p = 0.0 < 0.01$).

- and a test set $G^{(\mathrm{test})} = \left(\mathcal{V}, \mathcal{E}_\mathrm{P}^{(\mathrm{test})}\right)$, such that $|\mathcal{E}_\mathrm{P}^{(\mathrm{train})}| : |\mathcal{E}_\mathrm{P}^{(\mathrm{dev})}| : |\mathcal{E}_\mathrm{P}^{(\mathrm{test})}| = 3 : 1 : 1$.

(2) With the training set $G^{(\mathrm{train})}$, we learn scoring function s_k for each $k \in \left\{2^0, 2^1, \ldots, \min\left\{2^{14}, 2^{\lfloor \log_2(\mathrm{rank}\, M) \rfloor}\right\}\right\}$, where k is the rank of truncated SVD and matrix M is the incidence or adjacency matrix of $G^{(\mathrm{train})}$. For each k, we evaluate scoring function s_k by AUC with the development set $G^{(\mathrm{dev})}$, then select the best k.

(3) We calculate AUC of $s_{\mathrm{best}\,k}$ with the test set $G^{(\mathrm{test})}$ as the results.

We repeat this process five times, then we report the mean of AUC of $s_{\mathrm{best}\,k}$.

4.2.3 Experimental Results

In this subsection, we present our experimental results for both types of datasets and discuss the validity of our hypothesis.

Synthetic Datasets

Table 2 and **Fig. 3** show our experimental results on the synthetic datasets generated by the Barabási–Albert model. To confirm our hypothesis that IMF counters the sparsity problem better than AMF, we calculated Spearman's rank correlation coefficient (Spearman's ρ) between the sparsity measure ($|\mathcal{E}_\mathrm{P}|/|\mathcal{V}|^2$) and the performance improvements of IMF over AMF (AUC(IMF) − AUC(AMF)). If Spearman's ρ is negative, we conclude that the performance gain of IMF over AMF increases as the original graph becomes sparser; i.e., the hypothesis is supported. Spearman's ρ was calculated as -1.0 and its p-value is $0.0\,(< 0.01)$, which strongly supports our hypothesis.

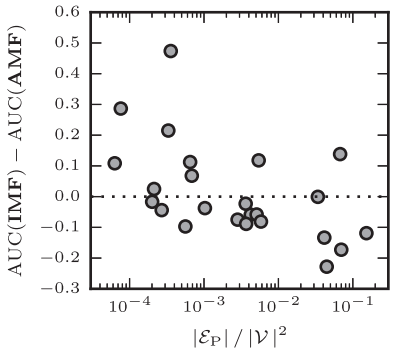

Fig. 4 A scatter plot illustrating the relationship between the sparsity (x-axis) and the performance improvements of IMF over AMF (y-axis). Each point corresponds to result for each real-world dataset. Here, Spearman's $\rho = -0.55$ ($p = 0.0054 < 0.01$).

Table 3 An excerpt of experimental results for AUC scores on the real-world datasets. For each row, the best result is shown in bold. We compared results of all pairs of graphs such that the relative difference of their sizes ($|\mathcal{V}|$) was < 20%, and one graph had more than five times as many positive links ($|\mathcal{E}_\mathrm{P}|$) as the other graph.

Name	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}_\mathrm{P}\rvert$	Sparsity	AMF	IMF
Jazz musicians	198	2,742	7.0e-02	**0.913**	0.740
PDZBase	212	244	5.4e-03	0.576	**0.694**
Hamsterster friendships	1,858	12,534	3.6e-03	**0.870**	0.847
Protein	1,870	2,277	6.5e-04	0.575	**0.687**
Hamsterster full	2,426	16,631	2.8e-03	**0.890**	0.815
Facebook (NIPS)	2,888	2,981	3.6e-04	0.523	**0.996**
Reactome	6,327	147,547	3.7e-03	**0.978**	0.890
Route views	6,474	13,895	3.3e-04	0.621	**0.836**
arXiv hep-th	22,908	2,673,133	5.1e-03	**0.963**	0.905
CAIDA	26,475	53,381	7.6e-05	0.655	**0.941**
arXiv hep-ph	28,093	4,596,803	5.8e-03	**0.988**	0.907
CAIDA	26,475	53,381	7.6e-05	0.655	**0.941**

Further, Table 2 demonstrates that (i) the AUC score using IMF was more robust to the sparsity of the original graph versus that of AMF and (ii) the performance of AMF decreased further as the original graph became sparser. These observations suggest that if the original graph possesses scale-free or small-world properties, IMF is able to achieve nearly consistent performance regardless of the sparsity of the graph, whereas the performance of AMF degrades by the sparsity.

Real-world Datasets

Table 1 and **Fig. 4** show the complete set of experimental results on the real-world datasets that we used. Similar to our experiments with the synthetic datasets, Spearman's ρ was -0.55 with a p-value of $0.0054\,(< 0.01)$; therefore, we conclude that IMF also alleviated the sparsity problem for the real-world datasets.

An excerpt of our experimental results are shown in **Table 3**; these results also support our hypothesis. More Specifically, Table 3 shows all pairs of graphs whose sizes were in the same range and whose sparsity measures differed radically. Here, the relative difference of their sizes ($|\mathcal{V}|$) was less than 20%, and one graph had more than five times as many positive links ($|\mathcal{E}_\mathrm{P}|$) as the other graph. For all sparser datasets, IMF outperformed AMF in terms of AUC, which also supports our hypothesis.

4.2.4 Discussion: Strengths and Weaknesses

In this subsection, we discuss the strengths and weaknesses of IMF as compared to AMF in terms of network statistics. We analyze our experimental results for real-world networks by examin-

Table 4 Correlations between network statistics and predictive performance on real-world datasets. Here, the highest correlated method in each row is shown in bold.

	AMF	IMF
Normalized edge distribution entropy	−0.08	**−0.81**
Gini coefficient	0.54	**0.85**
Clustering coefficient	**0.68**	−0.08
Average degree (another measure of density)	**0.97**	0.40

ing the correlation between various types of statistics of networks and the predictive performance of IMF and AMF. Next we detect some statistics with large differences in correlation with IMF and AMF; i.e., we find effective features of a network to make a judgment as to which method is more suitable for the given network, the results of which are summarized in **Table 4**.

Strengths

Non-uniformity: IMF has the advantage in predicting a link on *non-uniform* networks. It is well known that the degree distribution and edge distribution of real-world networks are non-uniform. In other words, only a few nodes have a considerable number of links, whereas most nodes have very few links. As noted by Kunegis and Preusse, the normalized edge distribution entropy and Gini coefficient are appropriate to use here to measure the non-uniformity of a network [13].

Interestingly, these measures of non-uniformity are highly correlated with the performance of IMF on real-world datasets, as shown in Table 4. For non-uniform networks, i.e., networks with *low* normalized edge distribution entropy or *high* Gini coefficient, IMF is expected to predict links with great accuracy.

This observation also shows that the predictive performance of IMF is stably high on synthetic datasets (Table 2) because the Barabási–Albert model generates scale-free networks with degree distribution following a power law, which is a typical non-uniform distribution.

Weaknesses

Clustering Coefficient: AMF has the advantage in predicting links on networks with high clustering coefficients. According to our experimental results for real-world networks, the clustering coefficient is highly correlated with AMF performance ($\rho = 0.68$). Complex networks typically have high clustering coefficients, as well as the scale-free property.

Given this, IMF is not suitable for networks with high clustering coefficients, because AMF is expected to predict links with high accuracy on these networks. Conversely, if the clustering coefficient of a network is low, we can utilize IMF instead of AMF. For example, from Table 1, the clustering coefficients of PDZBase, Facebook (NIPS), Route views, and CAIDA are quite low (< 0.01), and the corresponding predictive performance of AMF is low while the AUC of IMF is relatively high for these networks.

Average Degree: AMF has the advantage in predicting links on networks that have high average degrees. On real-world datasets, the AUC of AMF strongly correlates to the average degree of the graph ($\rho = 0.97$), as shown in Table 4.

Note that the average degree is sometimes called *density*. Therefore, AMF can perform on *dense* networks and not be expected to predict links well on *sparse* networks, at least in terms of average degree. This phenomenon also reinforces our main claim; i.e., IMF performs better than AMF as the given network becomes *sparser* in terms of average degree.

4.3 Computation Time

In Section 3.3, we theoretically compare the computational efficiency of IMF with that of AMF. In this subsection, we confirm the actual computation time.

4.3.1 Experimental Settings

We focus on the time required to learn the IMF and AMF models, which is comprised of matrix construction, matrix factorization, and matrix multiplication.

- AMF
  - Matrix construction: A (see Eq. (1)).
  - Matrix factorization: $U_k \Sigma_k {V_k}^\top \approx A$ (see Eq. (2)).
  - Matrix multiplication: $\tilde{A} = U_k \Sigma_k {V_k}^\top$ (see Eq. (2)).
- IMF
  - Matrix construction: $BB^\top = A + D$ (see Eq. (9)).
  - Matrix factorization: $Q_k \Lambda_k {Q_k}^\top \approx BB^\top$ (see Eq. (8)).
  - Matrix multiplication: $W = X(X^\top X)^{-1} X^\top - I_{|\mathcal{V}|}$, where $X = Q_k \Lambda_k^{1/2}$ (see Eqs. (4), (6), and (7)).

As in the case of Table 3, we handle all pairs of graphs such that the relative difference of their sizes ($|\mathcal{V}|$) is less than 20% and one graph has more than five times as many positive links ($|\mathcal{E}_\mathrm{P}|$) as the other graph. We conducted our experiments using the following common conditions: $k = \min\{1,024, 2^{\lfloor \log_2(\mathrm{rank}\, M) \rfloor}\}$, where matrix M is the incidence or adjacency matrix; the Python's version was 2.7.11; and the algorithm used for truncated SVD was randomized SVD [*1]. Finally, note that the computer we used was with Ubuntu 14.04, Xeon E5-2680 v2 (2.8 GHz) CPU, and 256 GB RAM.

4.3.2 Experimental Results

Table 5 shows our results of measuring computation times. Regardless of density, the computation time of matrix factorization is almost the same between IMF and AMF for all networks. As shown theoretically in Section 3.3, even in a network where the size of its incidence matrix becomes extremely large, such as arXiv hep-ph or arXiv hep-th, the empirical computation time of matrix factorization of IMF is almost equal to that of AMF, primarily because IMF learns the model through the factorization of matrix $BB^\top$ of the same size as adjacency matrix $A \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ (Section 3.3).

For matrix construction, i.e., loading data, IMF generally takes twice as long as AMF, because unlike the adjacency matrix, it is necessary to consider the diagonal components of $BB^\top$ (Eq. (9)). The cost of matrix multiplication of IMF is also higher than that of AMF, because IMF must solve the linear equation whereas AMF must only calculate simple matrix products in the matrix multiplication phase.

Finally, looking at the total time, learning with IMF completed within a few minutes, like the simplest instantiation of AMF with no expansion, even for huge networks with millions of edges (i.e., incidence matrix of the given graph is huge). Therefore, despite

[*1] We used the sklearn.utils.extmath.randomized_svd() function implemented in scikit-learn, a machine learning library in Python (http://scikit-learn.org/stable/developers/utilities.html).

Table 5 Empirical computation times for learning the given models. The use of "construct," "factorize," and "multiply" here represents the time for matrix construction, matrix factorization, and matrix multiplication, respectively; similarly, the use of "Total" represents the sum of these. Units are in seconds. The times listed in the table are the average execution times in five fold cross validation.

Name	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}_\mathrm{P}\rvert$		AMF	IMF
Jazz musicians	198	2,742	*Total*	*0.109*	*0.154*
			construct	0.038	0.093
			factorize	0.071	0.057
			multiply	0.000	0.004
PDZBase	212	244	*Total*	*0.066*	*0.083*
			construct	0.004	0.010
			factorize	0.061	0.069
			multiply	0.001	0.004
Hamsterster friendships	1,858	12,534	*Total*	*1.555*	*1.981*
			construct	0.154	0.350
			factorize	1.123	1.078
			multiply	0.278	0.553
Protein	1,870	2,277	*Total*	*1.414*	*1.671*
			construct	0.037	0.080
			factorize	1.116	1.102
			multiply	0.261	0.489
Hamsterster full	2,426	16,631	*Total*	*2.139*	*2.925*
			construct	0.203	0.469
			factorize	1.517	1.653
			multiply	0.419	0.803
Facebook (NIPS)	2,888	2,981	*Total*	*4.360*	*2.625*
			construct	0.047	0.096
			factorize	3.773	1.388
			multiply	0.540	1.141
Reactome	6,327	147,547	*Total*	*9.264*	*14.645*
			construct	1.626	4.135
			factorize	5.048	5.079
			multiply	2.590	5.431
Route views	6,474	13,895	*Total*	*7.690*	*11.034*
			construct	0.186	0.386
			factorize	4.896	4.967
			multiply	2.608	5.681
arXiv hep-th	22,908	2,673,133	*Total*	*84.145*	*160.527*
			construct	33.745	82.091
			factorize	21.569	19.630
			multiply	28.831	58.806
CAIDA	26,475	53,381	*Total*	*51.776*	*90.675*
			construct	0.665	1.537
			factorize	19.880	6.331
			multiply	31.231	82.807
arXiv hep-ph	28,093	4,596,803	*Total*	*88.305*	*221.525*
			construct	58.198	138.912
			factorize	13.466	13.406
			multiply	16.641	69.207
CAIDA	26,475	53,381	*Total*	*51.776*	*90.675*
			construct	0.665	1.537
			factorize	19.880	6.331
			multiply	31.231	82.807

the disadvantages noted above, we believe that IMF can withstand practical use sufficiently.

5. Supplemental Experiments

For completeness, we also compared the performance of IMF with a variant of AMF, i.e., the alternating least squares (ALS) matrix factorization, which is referred to as either *ALS matrix factorization* [10] or *regularized SVD* [21] in the collaborative filtering domain. The objective of our supplemental experiments are to confirm whether our hypothesis holds even if the baseline method is ALS. In this section, we first introduce ALS, then show our experimental results.

Table 6 Results of our supplemental experiments. The AUC scores on the synthetic datasets, sorted by the sparsity measure ($|\mathcal{E}_\mathrm{P}|/|\mathcal{V}|^2$). Here, n and k correspond to the parameters of the Barabási–Albert model introduced in Section 4.1.1. For each row, the best result is shown in bold.

n	k	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}_\mathrm{P}\rvert$	Sparsity	AMF	ALS	IMF
100	10	100	899	9.0e-02	0.651	0.640	**0.675**
100	9	100	818	8.2e-02	0.660	0.651	**0.689**
100	8	100	737	7.8e-02	0.642	0.628	**0.670**
100	7	100	650	6.5e-02	0.635	0.637	**0.687**
100	6	100	563	5.6e-02	0.648	0.632	**0.718**
100	5	100	474	3.7e-02	0.620	0.623	**0.695**
100	4	100	385	3.9e-02	0.621	0.624	**0.697**
100	3	100	290	2.9e-02	0.554	0.569	**0.654**
100	2	100	195	2.0e-02	0.486	0.506	**0.700**
100	1	100	98	9.8e-03	0.495	0.470	**0.637**

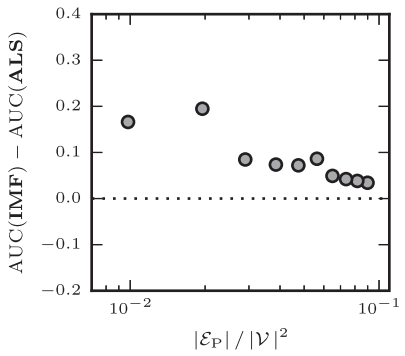

Fig. 5 A scatter plot illustrating the relationship between the sparsity (x-axis) and the performance improvements of IMF over ALS (y-axis). Each point corresponds to a result for each synthetic dataset. Here, Spearman's $\rho = -0.92$ ($p = 0.0 < 0.01$).

5.1 ALS Matrix Factorization

ALS learns latent vectors of the given nodes by using gradient descent to minimize following loss function

$$\mathcal{L}(X) = \frac{1}{2} \sum_{(v_i,v_j)\in\mathcal{E}_\mathrm{P}} (\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle - 1)^2 + \frac{\lambda}{2} \|X\|_\mathrm{F}^2 ,$$

where λ is a regularization hyperparameter. To predict a link, we use scoring function $s_\mathrm{ALS} : \mathcal{V} \times \mathcal{V} \to \mathbb{R}$ defined as

$$s_\mathrm{ALS}(v_i, v_j) = \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle,$$

which is a similar approach to that of AMF.

5.2 Experimental Setting

For our experiments, we generated 10 synthetic graphs again using the Barabási–Albert model, this time with parameters $n = 100$ and $k \in \{1, 2, \ldots, 10\}$. We essentially used the same experimental procedure described in Section 4.2.2. Specifically, for ALS, we employed a grid search to tune two hyperparameters k and $\lambda \in \{1.0, 0.1, 0.01, 0.001\}$.

5.3 Experimental Results

Table 6 shows the resulting AUC scores for AMF, ALS, and IMF on the generated synthetic graphs, **Fig. 5** illustrates the performance improvements of IMF over ALS.

From these results, we identify two key findings. First, Fig. 5 shows that IMF performed better than ALS as the graph became sparser. More specifically, Spearman's ρ was -0.92 with a p-value of 0.0. These results show that our hypothesis holds for

IMF and ALS on small synthetic datasets. Second, the relationship between the sparsity and the performance of ALS is quite similar to results for AMF. These findings can be quantitatively supported by Spearman's ρ between the AUC scores of AMF and those of ALS, i.e., $\rho = 0.95$ ($p = 0.0$). This second finding also allows us to reason that the first finding is also valid for large-scale datasets. Therefore, we conclude that our hypothesis may indeed hold for ALS and IMF.

6. Related Work

In this section, we review existing studies on (i) link prediction methods based on graph structures and (ii) devising countermeasures against the sparsity problem; further, we state the relationships these previous works have with our present research.

6.1 Link Prediction Based on Graph Structure

To learn scoring function $\mathcal{V} \times \mathcal{V} \to \mathbb{R}$ for predicting a link on an unlabeled node pair, there are typically three approaches to constructing the features of node pair (v_i, v_j) from the topology of the graph; these three approaches are based on(i) neighboring nodes of v_1 and those of v_2,(ii) the ensembles of paths from v_1 to v_2, and(iii) factorizing a matrix of the given graph. We follow this third line of research in our present work.

Typical examples of node-neighborhoods-based features are "common neighbors" and Adamic/Adar [2]. A probabilistic model of node neighborhoods using a Markov random field has also been presented by Wang et al. [23].

Katz is one of the features based on the ensembles of paths between a pair of nodes [11]. Further, Rooted PageRank [15] and PropFlow [16] are based on path information and are both founded on PageRank.

Matrix factorization is also used to calculate the feature on a node pair in a number of studies [1], [6], [14], [19]. Our study follows this line of research because of its computational efficiency. There are a number of various computationally efficient techniques for matrix factorization [18]. The modern real-world networks we want to model and use are massive; therefore, computational efficiency is of considerable importance. In this study, we utilized truncated SVD, a lightweight approach to the simplest matrix factorization that we use, to conduct thorough experiments and investigate the difference between the capabilities of IMF and AMF. When we put IMF to practical use, we can utilize various matrix factorization techniques to improve the performance.

Further, we utilize the incidence matrix rather than the adjacency matrix, the latter of which most studies have applied for link prediction. To the best of our knowledge, only Nori et al. used an incidence matrix for link prediction [20]. They represented multi-relational data as an incidence matrix and embedded the matrix into a low-dimensional space to avoid local optimal solutions from occurring when using tensor decomposition methods in multi-relation prediction. They noted that their method was potentially robust to data sparsity. The contribution of our research compared with theirs is that we (i) propose a new approach for utilizing a factorized incidence matrix for prediction and formulate our idea as a simple optimization problem that can be efficiently solved, and (ii) experimentally demonstrate that IMF actually alleviates the sparsity problem on both synthetic and real-world datasets.

For more detailed information about existing link prediction methods that make use of graph structure, there are several comprehensive surveys that address extensive information regarding link prediction [4], [17]. Further, some studies have compared the performance of multiple methods using several real-world networks [15], [19].

6.2 Sparsity Problem

As Rattigan and Jensen have noted, a fundamental problem in link prediction is a highly skewed class distribution, i.e., the data sparsity problem [22]. There are three main approaches to addressing this sparsity problem, i.e., (i) using an ensemble of classifiers, (ii) down-sampling, and (iii) combining with domain-specific side information. Our approach does not conflict with these three approaches, because some of them can be combined with our method, and the others cannot be applied to our problem setting.

First, Lichtenwalter et al. concluded that using an ensemble of classifiers is useful for sparse graphs after careful investigation of various issues that involve link prediction such as degrees of imbalance and variance reduction [16]. Our method can be incorporated into their framework.

Another set of approaches for addressing the sparsity problem including down-sampling and up-sampling [3], [23]. These techniques cannot be applied to our problem setting because they require both positive and negative examples, whereas only positive examples are available in our problem setting.

The third approach is to incorporate domain-specific side information—i.e., feature vectors of nodes or of node pairs—into graph-structure-based methods. Menon and Elkan reported that the combination of graph structure and side information was expected to be useful in predicting links, especially when a node was only sparsely connected [19]. They presented a method for incorporating side information into an adjacency matrix factorization approach; they experimentally demonstrated that this combination was able to yield better performance results than models based on either graph topology or node attributes. Our method can also be combined with side information by modifying Step 2 of our approach in a similar way to that of Menon's work. Combining side information is a fascinating expansion of our work.

7. Conclusion

To address the sparsity problem in link prediction, we presented a new direction that utilizes incidence matrix factorization (IMF) as a building block rather than adjacency matrix factorization (AMF), the latter of which has been used in a number of previous studies.

The key technical challenge was to establish a method for predicting a new link using the factorized incidence matrix. Our idea here was that a link on an unlabeled node pair is expected if we can successfully recover a latent vector of the pair that is consistent with latent vectors of positive links. We formulated this idea into a simple optimization problem that we then solved efficiently.

To validate our hypothesis, we conducted comparative experiments using synthetic datasets generated by the Barabási–Albert model and real-world datasets obtained from KONECT. Our experimental results showed that Spearman's ρ between sparsity measure $|\mathcal{E}_P|/|\mathcal{V}|^2$ and the performance gain of IMF over AMF was negative ($p < 0.01$) for both synthetic and real-world datasets. Therefore, we concluded that the IMF approach is able to successfully counter the sparsity problem better than the AMF approach. Our experimental results also showed that IMF was very robust to the sparsity of the synthetic datasets, whereas that of AMF worsened further as the graph became sparser. Therefore, we concluded that if the original graph has scale-free or small-world properties, IMF is more robust to the sparsity of the graph than AMF.

An interesting direction for future work is to apply our method to multinomial relation prediction problems. The sparsity problem is more serious in such problems because of the inherent high dimensionality. We expect our approach as being able to overcome the sparsity problem in multi-relation prediction because multi-relational data can quite naturally be represented as an incidence matrix.

Another future study we would like to conduct involves a set of large-scale comparative experiments with other link prediction methods, including ALS, which we discussed in Section 5. While this paper showed the promising property that IMF is more robust to network sparsity as compared with AMF, experimental verification of whether IMF always has this advantage versus other link prediction methods is important future research.

Acknowledgments This work was supported by JSPS KAKENHI Grant Number 15H01704.

References

[1] Acar, E., Dunlavy, D.M. and Kolda, T.G.: Link Prediction on Evolving Data Using Matrix and Tensor Factorizations, *Large-scale Data Mining: Theory and Applications at ICDM*, pp.262–269, IEEE (2009).

[2] Adamic, L.A. and Adar, E.: Friends and neighbors on the Web, *Social Networks*, Vol.25, No.3, pp.211–230 (2003).

[3] Al Hasan, M., Chaoji, V., Salem, S. and Zaki, M.: Link Prediction using Supervised Learning, *Workshop on Link Analysis, Counterterrorism and Security at SDM* (2006).

[4] Al Hasan, M. and Zaki, M.J.: A Survey of Link Prediction in Social Networks, *Social Network Data Analytics*, pp.243–275, Springer (2011).

[5] Barabási, A.-L. and Albert, R.: Emergence of Scaling in Random Networks, *Science*, Vol.286, No.5439, pp.509–512 (1999).

[6] Dong, E., Li, J. and Xie, Z.: Link Prediction via Convex Nonnegative Matrix Factorization on Multiscale Blocks, *Journal of Applied Mathematics*, Vol.2014, pp.786156:1–786156:9 (2014).

[7] Getoor, L.: Link Mining: A New Data Mining Challenge, *ACM SIGKDD Explorations Newsletter*, Vol.5, No.1, pp.84–89 (2003).

[8] Getoor, L. and Diehl, C.P.: Link Mining: A Survey, *ACM SIGKDD Explorations Newsletter*, Vol.7, No.2, pp.3–12 (2005).

[9] Hagberg, A.A., Schult, D.A. and Swart, P.J.: Exploring network structure, dynamics, and function using NetworkX, *Proc. 7th Python in Science Conference* (*SciPy*), pp.11–15 (2008).

[10] Hu, Y., Volinsky, C. and Koren, Y.: Collaborative Filtering for Implicit Feedback Datasets, *Proc. 8th IEEE International Conference on Data Mining* (*ICDM*), pp.263–272 (2008).

[11] Katz, L.: A new status index derived from sociometric analysis, *Psychometrika*, Vol.18, No.1, pp.39–43 (1953).

[12] Kunegis, J.: KONECT: The Koblenz Network Collection, *Proc. 1st International Web Observatory Workshop at WWW*, pp.1343–1350, ACM (2013).

[13] Kunegis, J. and Preusse, J.: Fairness on the Web: Alternatives to the Power Law, *Proc. 4th Annual ACM Web Science Conference*, pp.175–184, ACM (2012).

[14] Kunegis, J. and Lommatzsch, A.: Learning Spectral Graph Transformations for Link Prediction, *Proc. 26th Annual International Conference on Machine Learning* (*ICML*), pp.561–568, ACM (2009).

[15] Liben-Nowell, D. and Kleinberg, J.: The Link-Prediction Problem for Social Networks, *Journal of the American Society for Information Science and Technology*, Vol.58, No.7, pp.1019–1031 (2007).

[16] Lichtenwalter, R.N., Lussier, J.T. and Chawla, N.V.: New Perspectives and Methods in Link Prediction, *Proc. 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (*KDD*), pp.243–252, ACM (2010).

[17] Lü, L. and Zhou, T.: Link prediction in complex networks: A survey, *Physica A: Statistical Mechanics and its Applications*, Vol.390, No.6, pp.1150–1170 (2011).

[18] Mahoney, M.W.: Randomized algorithms for matrices and data, *Foundations and Trends in Machine Learning*, Vol.3, No.2, pp.123–224 (2011).

[19] Menon, A.K. and Elkan, C.: Link Prediction via Matrix Factorization, *Machine Learning and Knowledge Discovery in Databases - European Conference* (*ECML PKDD*), pp.437–452, Springer (2011).

[20] Nori, N., Bollegala, D. and Kashima, H.: Multinomial Relation Prediction in Social Data: A Dimension Reduction Approach, *Proc. 26th AAAI Conference on Artificial Intelligence* (*AAAI*), pp.115–121, AAAI Press (2012).

[21] Paterek, A.: Improving regularized singular value decomposition for collaborative filtering, *Proc. KDD Cup and Workshop*, pp.2–5 (2007).

[22] Rattigan, M.J. and Jensen, D.: The Case For Anomalous Link Discovery, *ACM SIGKDD Explorations Newsletter*, Vol.7, No.2, pp.41–47 (2005).

[23] Wang, C., Satuluri, V. and Parthasarathy, S.: Local Probabilistic Models for Link Prediction, *Proc. 7th IEEE International Conference on Data Mining* (*ICDM*), pp.322–331, IEEE (2007).

[24] Yamanishi, Y., Vert, J.-P. and Kanehisa, M.: Protein network inference from multiple genomic data: A supervised approach, *Bioinformatics*, Vol.20, No.suppl 1, pp.i363–i370 (2004).

Sho Yokoi is a doctoral student at Graduate School of Information Sciences, Tohoku University. He received his B.Eng. degree from Kyoto University in 2015 and his M.Info.Sci. degree from Tohoku University in 2017. His research focuses on machine learning and natural language processing.

Hiroshi Kajino is a researcher at IBM Research - Tokyo. He received his Ph.D. degree from The University of Tokyo in 2016. His research focuses on machine learning, data mining, and human computation.

Hisashi Kashima is a professor at Department of Intelligence Science and Technology, Kyoto University. He received his Ph.D. degree from Kyoto University in 2007. He was a researcher at IBM Tokyo Research Laboratory during 1999–2009, and was an associate professor at The University of Tokyo during 2009–2014. His research focuses on machine learning, data mining, and human computation.