2009 IEEE International Conference on Data Mining Workshops

Link Prediction on Evolving Data using Matrix and Tensor Factorizations

Evrim Acar
Informatics and Decision Sciences
Sandia National Laboratories
Livermore, CA 94551-9159
eacarat@sandia.gov

Daniel M. Dunlavy
Computer Science & Informatics
Sandia National Laboratories
Albuquerque, NM 87123-1318
dmdunla@sandia.gov

Tamara G. Kolda
Informatics and Decision Sciences
Sandia National Laboratories
Livermore, CA 94551-9159
tgkolda@sandia.gov

***Abstract*—The data in many disciplines such as social networks, web analysis, etc. is link-based, and the link structure can be exploited for many different data mining tasks. In this paper, we consider the problem of temporal link prediction: Given link data for time periods 1 through T, can we predict the links in time period $T+1$? Specifically, we look at bipartite graphs changing over time and consider matrix- and tensor-based methods for predicting links. We present a weight-based method for collapsing multi-year data into a single matrix. We show how the well-known Katz method for link prediction can be extended to bipartite graphs and, moreover, approximated in a scalable way using a truncated singular value decomposition. Using a CANDECOMP/PARAFAC tensor decomposition of the data, we illustrate the usefulness of exploiting the natural three-dimensional structure of temporal link data. Through several numerical experiments, we demonstrate that both matrix- and tensor-based techniques are effective for temporal link prediction despite the inherent difficulty of the problem.**

***Keywords*-link mining, link prediction, evolution, tensor factorization, CANDECOMP, PARAFAC**

I. Introduction

The data in different analysis applications such as social networks, web analysis, and collaborative filtering consists of relationships, which can be considered as links, between objects. For instance, two people may be linked to each other if they exchange emails or phone calls. These relationships can be modeled as a graph, where nodes correspond to the data objects (e.g., people) and edges correspond to the links (e.g., a phone call was made between two people). The link structure of the resulting graph can be exploited to detect underlying groups of objects, predict missing links, rank objects, and handle many other tasks [1].

Dynamic interactions over time introduce another dimension to the challenge of mining and predicting link structure. Here we consider the task of link prediction in time. Given link data for T time periods, can we predict the relationships at time $T+1$? This problem has been considered in a variety of contexts [2], [3], [4]. Collaborative filtering is also a related task, where the objective is to predict interest of users to objects (movies, books, music) based on the interests of similar users [5], [6]. The *temporal link prediction problem* is different from *missing link prediction*, which has no temporal aspect and where the goal is to predict missing connections in order to describe a more complete picture of the overall link structure in the data [7].

Time-evolving link data can be organized as a third-order tensor, or multi-dimensional array. In the simplest case, we can define a tensor $\mathcal{Z}$ of size $M \times N \times T$ such that

$$\mathcal{Z}(i,j,t) = \begin{cases} 1 & \text{if object } i \text{ links to object } j \text{ at time } t, \\ 0 & \text{otherwise.} \end{cases}$$

It is also possible to use weights to indicate the strength of the links. The matrix slice of $\mathcal{Z}$ corresponding to time t is represented by $\mathbf{Z}_t$. The goal is to predict the links at time $T+1$ by analyzing the link structure of $\mathcal{Z}$.

We consider both matrix- and tensor-based methods for link prediction. For the matrix-based methods, we collapse the data into a single matrix by summing (with and without weights) the matrices corresponding to the time slices. As a baseline, we consider a low-rank approximation as produced by a truncated singular value decomposition (TSVD). Next, we consider the Katz method [8] (extended to bipartite graphs), which has proven to be highly accurate in previous work on link prediction [9], [3], [10]; however, it is not always clear how to make the method scalable. Therefore, we present a novel scalable technique for computing approximate Katz scores based on a truncated spectral decomposition (TKatz). For the tensor-based methods, we consider the CANDECOMP/PARAFAC (CP) tensor decomposition [11], [12], which does not collapse the data but instead retains its natural three-dimensional structure. Tensor factorizations are higher-order extensions of matrix factorizations that capture the underlying patterns in multi-way data sets and have proved to be successful in diverse disciplines including chemometrics, neuroscience and social network analysis [13], [14]. Moreover, CP yields a highly interpretable factorization that includes a time dimension.

There are many possible applications for link prediction, such as predicting the web pages a web surfer may visit on a given day based on past browsing history, the places that a traveler may fly to in a given month, or the patterns of computer network traffic. As an example application for link prediction, we consider computer science conference publication data with a goal of predicting which authors will publish at which conferences in year $T+1$ given the

978-0-7695-3902-7/09 $26.00 © 2009 IEEE
DOI 10.1109/ICDMW.2009.54

publication data for the previous T years. In this case, we assume we have M authors and N conferences. All of the methods produce scores for each (i, j) author-conference pair for a total of MN prediction scores for year $T+1$. For large values of M or N, computing all possible scores is impractical due to the large memory requirements of storing all MN scores. However, we note that it is possible to easily compute subsets of the scores. For example, these methods can answer specific questions such as "Who is most likely to publish at ICDM next year?" or "Where is Abigail Sellen most likely to publish next year?" using only $O(M + N)$ memory. This is how we envision link prediction methods being used in practice.

The main contributions of this paper can be summarized as follows: 1) Weighted methods for collapsing temporal data into a matrix are shown to outperform straight summation (inspired by the results in [15]). 2) Katz is extended to the case of bipartite graphs and its relationship to the matrix SVD is derived. 3) Using the truncated SVD, we devise a novel scalable method for calculating a "truncated" Katz score. 4) The CP tensor decomposition is applied to temporal data and its interpretability is demonstrated with several examples. 5) Seven different matrix- and tensor-based methods are compared on DBLP bibliometric data in terms of link prediction performance and relative expense.

II. Matrix Techniques

We consider different matrix techniques by collapsing the matrices over time into a single matrix. In §II-A, we present two techniques (unweighted and weighted) for combining the multi-year data into a single matrix. In §II-B, we present the technique of using a truncated SVD to generate link scores. In §II-C, we extend the Katz method to bipartite graphs and show how it can be computed efficiently using a low-rank approximation.

A. Collapsing the data

Suppose that our data set consists of matrices $\mathbf{Z}_1$ through $\mathbf{Z}_T$ of size $M \times N$ and the goal is to predict $\mathbf{Z}_{T+1}$. The most straightforward way to collapse that data into a single $M \times N$ matrix $\mathbf{X}$ is to sum all the entries across time, i.e.,

$$\mathbf{X}(i,j) = \sum_{t=1}^{T} \mathbf{Z}_t(i,j). \tag{1}$$

We call this the *collapsed tensor (CT)* because it collapses (via a sum) the entries of the tensor $\mathcal{Z}$ along the time mode. This is similar to the approach in [3].

We propose an alternative approach to collapsing the tensor data, motivated by [15], where the link structure is damped backward in time according to the following formula:

$$\mathbf{X}(i,j) = \sum_{t=1}^{T} (1-\theta)^{T-t}\, \mathbf{Z}_t(i,j). \tag{2}$$

The parameter $\theta \in (0, 1)$ can be chosen by the user or according to experiments on various training data sets. We call this the *collapsed weighted tensor (CWT)* because the slices in the time mode are weighted in the sum. This gives greater weight to more recent links.

The numerical results in §IV demonstrate improved performance using CWT versus CT.

B. Truncated SVD

One of the methods compared in this paper is a low-rank approximation of the matrix $\mathbf{X}$ produced by (1) or (2). Specifically, suppose that the compact SVD of $\mathbf{X}$ is given by

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathsf{T}}, \tag{3}$$

where R is the rank of $\mathbf{X}$, $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices of sizes $M \times R$ and $N \times R$, respectively, and $\mathbf{\Sigma}$ is a diagonal matrix of singular values $\sigma_1 > \sigma_2 > \cdots > \sigma_R > 0$. It is well known that the best rank-K approximation of $\mathbf{X}$ is then given by the truncated SVD

$$\mathbf{X} \approx \mathbf{U}_K \mathbf{\Sigma}_K \mathbf{V}_K, \tag{4}$$

where $\mathbf{U}_K$ and $\mathbf{V}_K$ comprise the first K columns of $\mathbf{U}$ and $\mathbf{V}$ and $\mathbf{\Sigma}_K$ is the $K \times K$ principal submatrix of $\mathbf{\Sigma}$.

A matrix of scores for predicting future links can then be calculated as

$$\mathbf{S} = \mathbf{U}_K \mathbf{\Sigma}_K \mathbf{V}_K. \tag{5}$$

We call these the *Truncated SVD (TSVD)* scores. Low-rank approximations based on the matrix SVD have proven to be an effective technique in many data applications—latent semantic indexing [16] is one such example. This technique is called "low-rank approximation: matrix entry" in [3].

C. Katz

The Katz measure [8] is arguably one of the best link predictors available because it has been shown to outperform many other methods [3]. Suppose that we have an undirected graph $G(V, E)$ on $P = |V|$ nodes. Then the Katz score of a potential link between nodes i and j is given by

$$\hat{\mathbf{S}}(i,j) = \sum_{\ell=1}^{+\infty} \beta^{\ell} |\text{path}_{i,j}^{(\ell)}|, \tag{6}$$

where $|\text{path}_{i,j}^{(\ell)}|$ is the number of paths of length ℓ between nodes i and j, and $\beta > 0$ is a user-defined parameter.

The Katz scores for all pairs of nodes can be expressed in matrix terms as follows. Let $\hat{\mathbf{X}}$ be the $P \times P$ symmetric adjacency matrix of the graph. Then the scores are given by

$$\hat{\mathbf{S}} = \sum_{\ell=1}^{+\infty} \beta^{\ell} \hat{\mathbf{X}}^{\ell} = (\mathbf{I} - \beta\hat{\mathbf{X}})^{-1} - \mathbf{I}. \tag{7}$$

If the graph under consideration has weighted edges, $\hat{\mathbf{X}}$ is replaced by a weighted adjacency matrix.

We address two problems with the formulation of the Katz measure. First, the method is not scalable because it requires the inversion of a $P \times P$ matrix at a cost of $O(P^3)$ operations. We show that we can replace $\hat{\mathbf{X}}$ by a low-rank approximation in order to compute the Katz scores more efficiently. Second, the method is only applicable to square symmetric matrices representing undirected graphs. We show that it can also be applied to our situation: a rectangular matrix representing a bipartite graph.

1) Truncated Katz: Assume $\hat{\mathbf{X}}$ has rank $R \leq P$. Let the eigendecomposition of $\hat{\mathbf{X}}$ be given by

$$\hat{\mathbf{X}} = \mathbf{W}\mathbf{\Lambda}\mathbf{W}^{\mathsf{T}}, \tag{8}$$

where $\mathbf{W}$ is a $P \times P$ orthogonal matrix and $\mathbf{\Lambda}$ is a diagonal matrix with $|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_R| > \lambda_{R+1} = \cdots = \lambda_P = 0$. Then the Katz scores in (7) become

$$\begin{aligned}\hat{\mathbf{S}} &= (\mathbf{I} - \beta\mathbf{W}\mathbf{\Lambda}\mathbf{W}^{\mathsf{T}})^{-1} - \mathbf{I} \\ &= \mathbf{W}\left[(\mathbf{I} - \beta\mathbf{\Lambda})^{-1} - \mathbf{I}\right]\mathbf{W}^{\mathsf{T}} = \mathbf{W}\hat{\mathbf{\Gamma}}\mathbf{W}^{\mathsf{T}},\end{aligned}$$

where $\hat{\mathbf{\Gamma}}$ is a $P \times P$ diagonal matrix with diagonal entries

$$\hat{\gamma}_p = \frac{1}{1 - \beta\lambda_p} - 1 \quad \text{for} \quad p = 1, \ldots, P.$$

This shows a close relationship between the Katz measure and the eigendecomposition and gives some hint as to how to incorporate a low-rank approximation. The best rank-L approximation of $\hat{\mathbf{X}}$ is given by replacing $\mathbf{\Lambda}$ in (8) with a matrix $\mathbf{\Lambda}_L$ where all but the L largest magnitude diagonal entries are set to zero. The mathematics carries through as above, and the end result is that the Katz scores based on the rank-L approximation are

$$\hat{\mathbf{S}} = \mathbf{W}_L\hat{\mathbf{\Gamma}}_L\mathbf{W}_L^{\mathsf{T}}$$

where $\hat{\mathbf{\Gamma}}_L$ is the $L \times L$ principal submatrix of $\hat{\mathbf{\Gamma}}$, and $\mathbf{W}_L$ is the $P \times L$ matrix containing the first L columns of $\mathbf{W}$.

Since it is possible to construct a rank-L approximation of the adjacency matrix in $O(L|E|)$ operations (using an Arnoldi or Lanczos technique [17]), this technique can be applied to large-scale problems at a relatively low cost. We note that in [3], Katz is applied to a low-rank approximation of the adjacency matrix which is equivalent to what we discuss here, but its computation is not discussed — specifically, the fact that it can be computed efficiently via the formula above is not mentioned. Thus, we assume that calculation was done directly on the dense low-rank approximation matrix given by

$$\hat{\mathbf{X}}_L = \mathbf{W}_L\mathbf{\Lambda}_L\mathbf{W}_L^{\mathsf{T}}.$$

We contract to the approach of Wang et al. [18] who discuss an approximate Katz measure given by truncating the sum in (6) to the first L (they recommend $L = 4$) terms, i.e., $\hat{\mathbf{S}} = \sum_{\ell=1}^{4} \beta^\ell \hat{\mathbf{X}}^\ell$; the main drawback of this approach is the power matrices may be dense, depending on the connectivity of the graph.

2) Bipartite Katz & Truncated Bipartite Katz: Our problem is different than what we have discussed so far because we are considering a bipartite graph, represented by a weighted adjacency matrix from (1) or (2). This can be considered as a graph on $P = M + N$ nodes where the weighted adjacency matrix is given by

$$\hat{\mathbf{X}} = \begin{bmatrix} \mathbf{0} & \mathbf{X} \\ \mathbf{X}^{\mathsf{T}} & \mathbf{0} \end{bmatrix}.$$

If $\mathbf{X}$ is rank R and its SVD is given as in (3), then the eigenvectors and eigenvectors of $\hat{\mathbf{X}}$ are given by

$$\mathbf{W} = \begin{bmatrix} \frac{1}{\sqrt{2}}\mathbf{U} & -\frac{1}{\sqrt{2}}\mathbf{U} \\ \frac{1}{\sqrt{2}}\mathbf{V} & \frac{1}{\sqrt{2}}\mathbf{V} \end{bmatrix} \quad \text{and} \quad \mathbf{\Lambda} = \begin{bmatrix} \mathbf{\Sigma} & \mathbf{0} \\ \mathbf{0} & -\mathbf{\Sigma} \end{bmatrix}.$$

Note that the eigenvalues in $\mathbf{\Lambda}$ are not sorted by magnitude and the rank of $\bar{\mathbf{X}}$ is $2R$. The square matrix of Katz scores is given by

$$\hat{\mathbf{S}} = \begin{bmatrix} \mathbf{0} & \mathbf{U}\mathbf{\Gamma}\mathbf{V}^{\mathsf{T}} \\ \mathbf{V}\mathbf{\Gamma}\mathbf{U}^{\mathsf{T}} & \mathbf{0} \end{bmatrix},$$

where $\mathbf{\Gamma}$ is a diagonal matrix with entries

$$\gamma_p = \frac{1}{1 - \beta\sigma_p} - 1 \quad \text{for} \quad p = 1, \ldots, R. \tag{9}$$

The link scores for the bipartite graph can be extracted and are given by

$$\mathbf{S} = \mathbf{U}\mathbf{\Gamma}\mathbf{V}^{\mathsf{T}}. \tag{10}$$

We call these the *Katz* scores.

We can replace $\mathbf{X}$ by its best rank-K approximation as in (4), and the resulting Katz scores then become

$$\mathbf{S} = \mathbf{U}_K\mathbf{\Gamma}_K\mathbf{V}_K^{\mathsf{T}}, \tag{11}$$

where $\mathbf{\Gamma}_K$ is the $K \times K$ principal submatrix of $\mathbf{\Gamma}$. We call these the *Truncated Katz (TKatz)* scores. It is interesting to note that TKatz is very similar to using TSVD except that the diagonal weights have been changed. Related methods for scaling have also been proposed in the area of information retrieval (e.g., [19], [20]) where exponential scaling of singular values led to improved performance.

D. Computational Complexity and Memory

Computing a sparse rank-K TSVD via an Arnoldi or Lanczos method requires $O(\mathrm{nnz}(\mathbf{X}))$ work per iteration where $\mathrm{nnz}(\mathbf{X})$ is the number of nonzeros in the adjacency matrix $\mathbf{X}$, which is equal to the number of edges in the bipartite graph. The number of iterations is typically a small multiple of K but cannot be known in advance. The storage of the factorization requires only $K(M + N + 1)$ space for the singular values and two factor matrices. Because TKatz is based on the TSVD, it requires the same amount of computation and storage for a rank-K approximation. The only difference is that TKatz stores $\mathbf{\Gamma}_K$ rather than $\mathbf{\Sigma}_K$. Katz, on the other hand, requires $O(M^2N + MN^2 + N^3)$ operations to compute (7) if $M > N$. Furthermore, it stores all of the scores explicitly, using $O(MN)$ storage.

III. Tensor Techniques

The data set consisting of matrices $\mathbf{Z}_1$ through $\mathbf{Z}_T$ is three-way, so this lends itself to a multi-dimensional interpretation. One of the most common and useful tensor models is CP [11], [12]; see also reviews [13], [14]. The advantage of a three-way model is that we can explicitly model the time dimension and have no need to collapse the data as discussed in §II-A.

A. CP Tensor Model

Given a three-way tensor $\mathcal{Z}$ of size $M \times N \times T$, its K-component CP decomposition is given by

$$\mathcal{Z} \approx \sum_{k=1}^{K} \lambda_k \, \mathbf{a}_k \circ \mathbf{b}_k \circ \mathbf{c}_k. \tag{12}$$

Here the symbol $\circ$ denotes the outer product,[1] $\lambda_k \in \mathbb{R}_+$, $\mathbf{a}_k \in \mathbb{R}^M$, $\mathbf{b}_k \in \mathbb{R}^N$, and $\mathbf{c}_k \in \mathbb{R}^T$ for $k = 1, \ldots, K$. Each summand $(\lambda_k \ \mathbf{a}_k \circ \mathbf{b}_k \circ \mathbf{c}_k)$ is called a *component*, and the individual vectors are called *factors*. We assume $\| \mathbf{a}_k \| = \| \mathbf{b}_k \| = \| \mathbf{c}_k \| = 1$ and therefore λ_k contains the scalar weight of the kth component. The CP tensor decomposition can be considered an analogue of the SVD because it decomposes a tensor as a sum of rank-one tensors just as the SVD decomposes a matrix as a sum of rank-one matrices Nevertheless, there are also important differences between them. The columns of $\mathbf{U}$ and $\mathbf{V}$ are orthogonal in the SVD while there is no orthogonality constraint in the CP model. Despite the CP model's lack of orthogonality, Kruskal [14] has shown that it is unique, up to permutation and scaling, under mild conditions.

We compute the CP model via an Alternating Least Squares (ALS) approach using the Tensor Toolbox for Matlab [21].

B. CP Scoring

We make use of the components extracted by the CP model to assign scores to each pair (i, j) according to their likelihood of linking in the future. The outer product of $\mathbf{a}_k$ and $\mathbf{b}_k$, i.e., $\mathbf{a}_k \mathbf{b}_k^\mathsf{T}$ quantifies the relationship between object pairs in component k. The components have temporally different profiles in the vectors $\mathbf{c}_k$; e.g., they may have increasing, decreasing, or steady profiles. Therefore, we scale $\mathbf{a}_k \mathbf{b}_k^\mathsf{T}$ so that objects occurring together at components with increasing trends are assigned higher similarity scores than the objects occurring together at components with decreasing trends. In short, we define the similarity score for objects i and j using a K-component CP model in (12) as the (i, j) entry of the following matrix:

$$\mathbf{S} = \sum_{k=1}^{K} \gamma_k \lambda_k \mathbf{a}_k \mathbf{b}_k^\mathsf{T}, \quad \text{where} \quad \gamma_k = \sum_{t=T-2}^{T} \mathbf{c}_k(t). \tag{13}$$

[1] A three way outer product is defined as follows: $\mathcal{X} = \mathbf{a} \circ \mathbf{b} \circ \mathbf{c}$ means $\mathcal{X}(i, j, k) = \mathbf{a}(i)\mathbf{b}(j)\mathbf{c}(k)$.

We have used a simple scaling approach here. However, we can make better use of temporal profiles by using time series models such as the Holt-Winters method [22], which models the trends and seasonality in the data, to forecast the time points in near future. This is a future research topic.

C. Computational Complexity and Memory

The computational complexity of CP is similar to TSVD. It is proportional to $K\mathrm{nnz}(\mathcal{Z}) + K^2(M + N + T)$ work per iteration. Since we normally assume $\mathrm{nnz}(\mathcal{Z}) \gg M+N+T$, the computational complexity is $O(\mathrm{nnz}(\mathcal{Z}))$. As with TSVD, we cannot predict the number of iterations in advance. The storage required for CP is $K(M+N+T+1)$, for the three factor matrices and the scalar λ_k values.

IV. Experiments

We use the DBLP data set[2] to assess the performance of various link predictors discussed in §II and §III. All experiments were performed using Matlab 7.6 on a Linux Workstation (RedHat 5.2) with 2 Quad-Core Intel Xeon 3.0GHz processors and 32GB RAM.

A. Data

At the time the DBLP data was downloaded for this work, it contained publications from 1936 through the end of 2007. We only consider publications of type *inproceedings* between 1991 and 2007[3].

The data is organized as a third-order tensor $\mathcal{Z}$ of size $M \times N \times T$. We let $\mathcal{C}(i, j, t)$ denote the total number of papers by author i at conference j in year t. In order to decrease the effect of large numbers of publications, we preprocess the data so that

$$\mathcal{Z}(i, j, t) = \begin{cases} 1 + \log(\mathcal{C}(i, j, t)) & \text{if } \mathcal{C}(i, j, t) > 0, \\ 0 & \text{otherwise.} \end{cases}$$

Using a sliding window approach, we divide the data into seven training/test sets such that each training set contains $T = 10$ years and the corresponding test set contains the following 11th year. Table I shows the size and density of the training and testing sets. We only keep those authors that have at least 10 publications (i.e., an average of one per year) in the training period, and each test set contains only the authors and conferences available in the corresponding training set.

B. Interpretation of CP

Before addressing the link prediction problem, we first discuss how to use the CP model for exploratory analysis of the temporal data. The primary advantage of the CP model is its interpretability, as illustrated in Figure 1, which contains three example components from the 50-component CP

[2] http://www.informatik.uni-trier.de/~ley/db/index.html

[3] The publications between 1936 and 1990 comprise only 6% of publications of type *inproceedings*.

Test Year	# Authors	# Conf.	# Train Links (%)	# Test Links (%)	# New Test Links (%)
2001	7108	1103	113k (.14)	13k (.16)	5k (.06)
2002	8368	1211	135k (.13)	16k (.16)	7k (.07)
2003	9929	1342	162k (.12)	20k (.15)	9k (.07)
2004	11836	1491	197k (.11)	27k (.16)	13k (.07)
2005	14487	1654	245k (.10)	35k (.15)	17k (.07)
2006	17811	1806	308k (.10)	40k (.13)	19k (.06)
2007	21328	1934	377k (.09)	41k (.10)	20k (.05)

Table I: Data on training and test sets formed from the DBLP data set. The training data consists of data from the ten years preceding the test year.

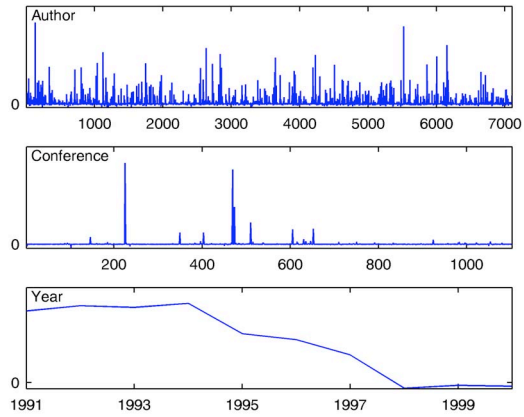
model of the tensor representing publications from 1991 to 2000. The factor $\mathbf{a}_k$ captures a certain group of authors while $\mathbf{b}_k$ extracts the conferences, where the authors captured by $\mathbf{a}_k$ publish. Finally, $\mathbf{c}_k$ corresponds to the temporal signature and shows what kind of time pattern the publication history of those authors at those conferences follows. Therefore, the CP model can address the link prediction problem well by capturing the evolution of the links between objects using the factors in the time mode.

Figure 1a shows the third component with authors in the top plot ($\mathbf{a}_k$), conferences in the middle ($\mathbf{b}_k$), and time on the bottom ($\mathbf{c}_k$). The highest scoring conferences are DAC, ICCAD and ICCD, which are related conferences on computer design. Many authors publish in these conferences between 1991 and 2000, but the top are Vincentelli, Brayton, and others listed in the caption. This author/conference combination has a peak in the early 1990s and starts to decline in mid-'90s. Note that the author and conference scores are mostly positive. Figure 1b shows another example component, which actually has very similar conferences to the component discussed above. The leading authors, however, are different. Further, the time profile is different with an increasing trend after the mid-'90s. Figure 1c shows a component that detects related conferences that take place only in even years. Again we see that the components are primarily positive. A nice feature of the CP model is that it does not have any constraints (like orthogonality in the SVD) that artificially impose a need for negative entries in the components.

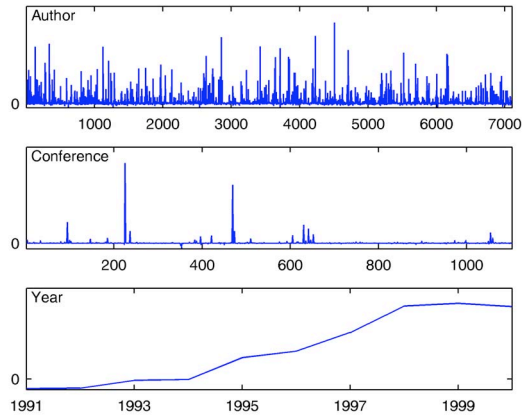
C. Methods and Parameter Selection

The goal of a link predictor in this study is to predict whether the ith author is going to publish at the jth conference during the test year. Therefore, each nonzero entry in the test set is treated as 1, i.e., a positive link, regardless of the actual number of publications; otherwise, it is 0 indicating that there is no link between the corresponding author-conference pair.

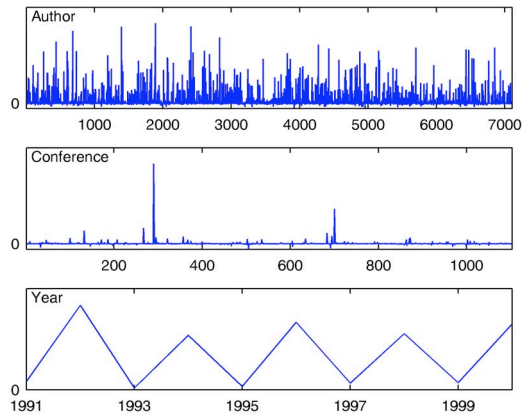
The common parameter for all link predictors, except Katz-CT/CWT, is the number of components, K. In our experiments, instead of using a specific value of K, which cannot be determined systematically, we use an ensemble



(a) Factors from component 3: Top authors are Alberto L. Sangiovanni Vincentelli, Robert K. Brayton, Sudhakar M. Reddy, and Irith Pomeranz. Top conferences are DAC, ICCAD, and ICCD.



(b) Factors from component 4: Top authors are Miodrag Potkonjak, Massoud Pedram, Jason Cong, and Andrew B. Kahng. Top conferences are DAC, ICCAD, and ASPDAC.



(c) Factors from component 46: Top authors are Franz Baader, Henri Prade, Didier Dubois, and Bernhard Nebel. Top conferences are ECAI and KR.

Figure 1: Examples from 50-component CP model of publications from 1991 to 2000.

approach. Let $\mathbf{S}_K$ denote the matrix of scores computed for $K = 10, 20, ...100$. Then the matrix of ensemble scores, $\mathbf{S}$, used for link prediction is calculated as $\mathbf{S} = \sum_{K \in \{10,20,...100\}} \frac{\mathbf{S}_K}{\| \mathbf{S}_K \|_F}$. In addition to the number of components, the parameter β used in the Katz scores in (10) and (11) needs to be determined. We use $\beta = 0.001$, which was chosen such that $\gamma_p > 0$ for all $p = 1, \ldots, R$ in (9) for the data in our experiments. We have observed that if $\gamma_p < 0$ then the scores contain entries with large magnitudes but negative values, which degrades the performance of Katz measure. Finally, θ is the parameter used for weighting slices while forming the CWT in (2). We set $\theta = 0.2$ according to preliminary tests on the training data sets.

D. Link Prediction Results

Two different experimental set-ups are used to evaluate the performance of the methods.

- *Predicting All Links*: The first approach compares the methods in terms of how well they predict positive links in the test set.
- *Predicting New Links*: The second approach addresses a more challenging problem, i.e., how well the methods predict the links that have not been previously seen at any time in the training set.

As an evaluation metric for link prediction performance, we use the area under the receiver operating characteristic curve (AUC) because it is viewed as a robust measure in the presence of imbalance [23], which is important since less than 0.2% of all possible links exist in our testing data. Figure 2 shows the performance of each link predictor in terms of AUC when predicting all links (blue bars) and new links (red bars). As expected, the AUC values are much lower for the new links. Among all methods, the best performing method in terms of AUC is Katz-CWT. Further, CWT is consistently better than the corresponding CT methods, which shows that giving more weight to the data in recent years improves link prediction.

In Figure 3, we show the ROC (receiver operating characteristic) curves; for the purposes of clarity, we omit the CT results. When predicting all links, Figure 3a shows that all methods perform similarly initially, but Katz-CWT is best as the false positive rate increases. TKatz-CWT and TSVD-CWT are only slightly worse than Katz-CWT. Finally, CP starts having false-positives earlier than the other methods. Figure 3b shows the behavior for just the new links. In this case, the relative performance of the algorithms is mostly unchanged.

In order to understand the behavior of different link predictors better, we also compute how many *correct links* (true positives) are in the first top 1000 scores. Table II shows that CP, TSVD-CWT, TKatz-CWT and Katz-CWT achieve close to 75% accuracy over all links. The accuracy of the methods goes down to 10% when we remove all previously seen links from the test set. Note that the best

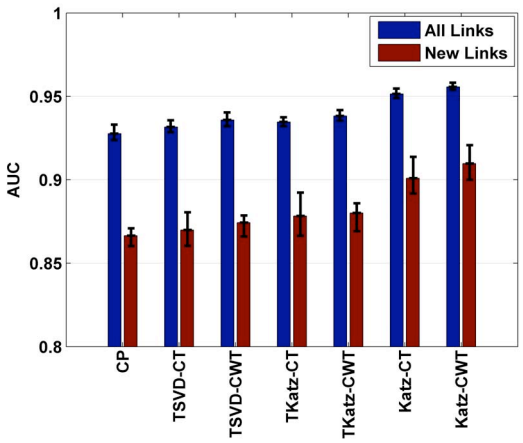
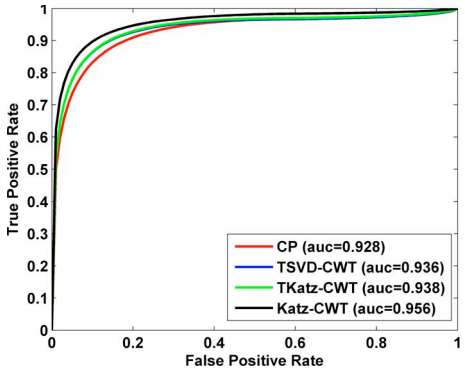
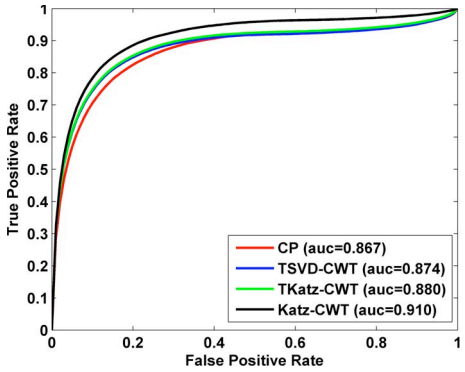


Figure 2: Average link prediction performance of each method across all seven training/test set pairs (black bars show absolute range).



(a) Prediction of all links in the test sets.



(b) Prediction of only new links in the test sets.

Figure 3: Average ROC curves showing the performance of link prediction methods across all training/test sets.

methods in terms of AUC, i.e., Katz-CT and Katz-CWT, perform worse than CP, TSVD-CWT and TKatz-CWT for predicting new links. We also observe that CP is among the best methods when we look at the top predictions even if it starts giving false-positives earlier than other methods.

Test Year	CP	TSVD -CT	TSVD -CWT	TKatz -CT	TKatz -CWT	Katz -CT	Katz -CWT
All Links							
2001	671	617	685	611	684	629	705
2002	668	660	674	656	672	668	713
2003	723	697	743	693	748	708	755
2004	783	726	777	724	773	723	780
2005	755	716	776	718	770	706	780
2006	807	729	801	731	796	706	798
2007	721	681	755	682	753	656	731
Mean	733	689	744	688	742	685	752
New Links							
2001	87	80	104	79	107	55	77
2002	97	84	124	88	124	86	104
2003	78	80	96	82	102	67	75
2004	99	79	105	85	105	70	82
2005	116	89	117	90	117	67	89
2006	91	77	110	75	109	69	92
2007	83	71	95	73	99	49	67
Mean	93	80	107	82	109	66	84

Table II: Correct predictions in top 1000 scores.

The TSVD and TKatz methods are quite fast. Average timings across all training sets are TSVD-CT: 60.6 sec.; TSVD-CWT: 61.4 sec.; TKatz-CT: 123.9 sec. and TKatz-CWT: 126.8 sec. CP takes 1304.1 sec., and the Katz-CT and Katz-CWT requires 1623.1 and 2081.4 sec., respectively.

V. Related Work

Getoor and Diehl [1] present a survey of link mining tasks, including node classification, group detection, and numerous other tasks including link prediction. Sharan and Neville [15] consider the goal of node classification for temporal-relational data, suggesting the idea of a "summary graph" of weighted snapshots in time which we have incorporated into this work.

The seminal work of Liben-Nowell and Kleinberg [3] examines numerous methods for link prediction on co-authorship networks in arXiv bibliometric data. However, temporal information was unused (e.g., as in [15]) except for splitting the data The proportion of new links ranged from 0.1–0.5% and is thus comparable to what we see in our data (0.05–0.07%). According to Liben-Nowell and Kleinberg, Katz and its variants are among the best link predictors; this observation has been supported by other work as well [9], [18]. We note that Wang, Satuluri and Parthasarathy [18] use the truncated sum approximate Katz measure discussed in §II-C and recommend AUC as one evaluation measure for link prediction because it does not require any arbitrary cut-off. Rattigan and Jensen [24] contend that the link mining problem is too difficult, in part because the proportion of actual links is very small compared to the number of possible links; specifically, they consider co-author relationships in DBLP data and observe that the proportion of new links is less than 0.01%. (Although we also use DBLP data, we consider author-conference links which has 0.05% or more new links.)

Another way to approach link prediction is to treat it as a straightforward classification problem by computing features for possible links and using a state-of-the-art classification engine like support vector machines. Al Hasan et al. [2] use this approach in the task of author-author link prediction. They randomly pick equal sized sets of linked and unlinked pairs of authors. They compute features such as keyword similarity, neighbor similarity, shortest path, etc. However, it would likely be computationally intractable to use such a method for computing *all* possible links due to the size of the problem and imbalance between linked and unlinked pairs of authors. Clauset, Moore, and Newman [7] predict links (or anomalies) using Monte-Carlo sampling on all possible dendrogram models of a graph.

Modeling the time evolution of graphs has been considered, e.g., by Sakar et al. [4] who create time-evolving co-occurrence models that map entities into an evolving latent space. Tong et al. [25] also compute centrality measures on time evolving bipartite graphs by aggregating adjacency matrices over time in similar approaches to those in §II-A.

Link prediction is also related to the task of collaborative filtering. In the Netflix contest, for example, Bell and Koren [26] consider the "binary view of the data" as important as the ratings themselves. In other words, it is important to first predict who is likely to rate what before focusing on the ratings. This was a specific task in KDD Cup 2007 [5].

Tensor factorizations have been previously applied in web link analysis [27] and also in social networks for the analysis of chatroom [28] and email communications [29], [30]. In these applications tensor factorizations are used as exploratory analysis tools and do not address the link prediction problem.

VI. Conclusions

In this paper, we explore several matrix- and tensor-based approaches to solving the link prediction problem. We consider author-conference relationships in bibliometric data as an example application, but this has many applications in other domains such as predicting Internet traffic, flight reservations, and more. For the matrix methods, our results show that using a weighted collapse function to combine multiple time slices is superior to simple summation. We also show how to extend Katz to bipartite graphs and to efficiently compute an approximation to Katz based on the truncated SVD. However, none of the matrix-based methods fully leverages and exposes the temporal signatures in the data. We present an alternative: the CP tensor factorizations. Temporal information can be incorporated into the CP tensor-based link prediction analysis to gain a perspective not available when computing using matrix-based approaches.

We have considered these methods in terms of their AUC scores and the number of correct predictions in the top 1000 scoring links. In both cases, we can see that all the

methods do quite well. Katz has the best AUC but is not computationally tractable for large-scale problems; however, the other methods are not far behind. Moreover, TKatz-CWT is best for predicting new links. Our numerical results also show that the tensor-based methods are competitive with the matrix-based methods in terms of link prediction performance. The current drawback of the tensor-based approach is that there is typically a higher computational cost incurred, but the software for these methods is quite new and will no doubt be improved in the near future.

ACKNOWLEDGMENTS

This work was funded by the Laboratory Directed Research & Development (LDRD) program at Sandia National Laboratories, a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under Contract DE-AC04-94AL85000.

REFERENCES

[1] L. Getoor and C. P. Diehl, "Link mining: a survey," *ACM SIGKDD Explorations Newslett.*, vol. 7, pp. 3–12, 2005.

[2] M. A. Hasan, V. Chaoji, S. Salem, and M. Zaki, "Link prediction using supervised learning," in *Proc. SIAM Data Mining Workshop on Link Analysis, Counterterrorism, and Security*, 2006.

[3] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *J. Amer. Soc. Inform. Sci. Technolgy*, vol. 58, pp. 1019–1031, 2007.

[4] P. Sarkar, S. M. Siddiqi, and G. J. Gordon, "A latent space approach to dynamic embedding of co-occurrence data," in *Proc. AI-STATS'07*, 2007.

[5] Y. Liu and Z. Kou, "Predicting who rated what in large-scale datasets," *ACM SIGKDD Explorations Newslett.*, vol. 9, pp. 62–65, 2007.

[6] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Computer*, vol. 42, pp. 30–37, 2009.

[7] A. Clauset, C. Moore, and M. Newman, "Hierarchical structure and the prediction of missing links in networks," *Nature*, vol. 453, 2008.

[8] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, pp. 39–43, 1953.

[9] Z. Huang, X. Li, and H. Chen, "Link prediction approach to collaborative filtering," in *Proc. JCDL'05*, 2005, pp. 141–142.

[10] Z. Huang and D. K. J. Lin, "The time-series link prediction problem with applications in communication surveillance," *INFORMS J. Computing*, vol. 21, pp. 286–303, 2009.

[11] J. D. Carroll and J. J. Chang, "Analysis of individual differences in multidimensional scaling via an N-way generalization of 'Eckart-Young' decomposition," *Psychometrika*, vol. 35, pp. 283–319, 1970.

[12] R. A. Harshman, "Foundations of the PARAFAC procedure: Models and conditions for an 'explanatory' multi-modal factor analysis," *UCLA working papers in phonetics*, vol. 16, pp. 1–84, 1970.

[13] E. Acar and B. Yener, "Unsupervised multiway data analysis: A literature survey," *IEEE Trans. Knowl. Data Eng.*, vol. 21, pp. 6–20, 2009.

[14] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, pp. 455–500, 2009.

[15] U. Sharan and J. Neville, "Temporal-relational classifiers for prediction in evolving domains," in *Proc. ICDM'08*, 2008, pp. 540–549.

[16] S. T. Dumais, G. W. Furnas, T. K. Landauer, S. Deerwester, and R. Harshman, "Using latent semantic analysis to improve access to textual information," in *Proc. CHI'88*, 1988, pp. 281–285.

[17] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*. Manchester University Press, 1992.

[18] C. Wang, V. Satuluri, and S. Parthasarathy, "Local probabilistic models for link prediction," in *Proc. ICDM'07*, 2007, pp. 322–331.

[19] H. Bast and D. Majumdar, "Why spectral retrieval works," in *Proc. SIGIR'05*, 2005, pp. 11–18.

[20] H. Yan, W. I. Grosky, and F. Fotouhi, "Augmenting the power of LSI in text retrieval: Singular value rescaling," *Data Knowledge and Eng.*, vol. 65, pp. 108–125, 2008.

[21] B. W. Bader and T. G. Kolda, "Efficient MATLAB computations with sparse and factored tensors," *SIAM J. Scientific Computing*, vol. 30, pp. 205–231, 2007.

[22] C. Chatfield, *The Analysis of Time Series*. Chapman & Hall/CRC, 2003.

[23] M. Stager, P. Lukowicz, and G. Troster, "Dealing with class skew in context recognition," in *Proc. ICDCSW'06*, 2006, p. 58.

[24] M. J. Rattigan and D. Jensen, "The case for anomalous link discovery," *ACM SIGKDD Explorations Newslett.*, vol. 7, pp. 41–47, 2005.

[25] H. Tong, S. Papadimitriou, P. S. Yu, and C. Faloutsos, "Proximity tracking on time-evolving bipartite graphs," in *Proc. SDM'08*, 2008, pp. 704–715.

[26] R. M. Bell and Y. Koren, "Lessons from the Netflix Prize Challenge," *ACM SIGKDD Explorations Newslett.*, vol. 9, pp. 75–79, 2007.

[27] T. G. Kolda, B. W. Bader, and J. P. Kenny, "Higher-order web link analysis using multilinear algebra," in *Proc. ICDM'05*, 2005, pp. 242–249.

[28] E. Acar, S. A. Çamtepe, and B. Yener, "Collective sampling and analysis of high order tensors for chatroom communications," in *Proc. ISI'06*, 2006, pp. 213–224.

[29] B. W. Bader, M. W. Berry, and M. Browne, "Discussion tracking in Enron email using PARAFAC," in *Survey of Text Mining: Clustering, Classification, and Retrieval, Second Edition*, M. W. Berry and M. Castellanos, Eds. Springer, 2007, pp. 147–162.

[30] J. Sun, S. Papadimitriou, C.-Y. Lin, N. Cao, S. Liu, and W. Qian, "Multivis: Content-based social network exploration through multi-way visual analysis," in *Proc. SDM'09*, 2009, pp. 1063–1074.