INFORMS Journal on Computing
Vol. 21, No. 2, Spring 2009, pp. 286–303
ISSN 1091-9856 | EISSN 1526-5528 | 09 | 2102 | 0286

DOI 10.1287/ijoc.1080.0292

The Time-Series Link Prediction Problem with Applications in Communication Surveillance

Zan Huang, Dennis K. J. Lin
Department of Supply Chain and Information Systems, Smeal College of Business, Pennsylvania State University, University Park, Pennsylvania 16802 {zanhuang@psu.edu, dennislin@psu.edu}

The ability to predict linkages among data objects is central to many data mining tasks, such as product recommendation and social network analysis. Substantial literature has been devoted to the *link prediction* problem either as an implicitly embedded problem in specific applications or as a generic data mining task. This literature has mostly adopted a *static graph* representation where a snapshot of the network is analyzed to predict hidden or future links. However, this representation is only appropriate to investigate whether a certain link will ever occur and does not apply to many applications for which the prediction of the *repeated* link occurrences are of primary interest (e.g., communication network surveillance). In this paper, we introduce the *time-series link prediction problem*, taking into consideration temporal evolutions of link occurrences to predict link occurrence probabilities at a particular time. Using Enron e-mail data and high-energy particle physics literature coauthorship data, we have demonstrated that time-series models of single-link occurrences achieve comparable link prediction performance with commonly used static graph link prediction algorithms. Furthermore, a combination of static graph link prediction algorithms and time-series models produced significantly better predictions over static graph link prediction methods, demonstrating the great potential of integrated methods that exploit both interlink structural dependencies and intralink temporal dependencies.

Key words: analysis of algorithms; communication networks; link prediction; statistical analysis; time-series analysis
History: Accepted by Amit Basu, former Area Editor for Knowledge and Data Management; received February 2007; revised January 2008; accepted June 2008. Published online in *Articles in Advance* November 13, 2008.

1. Introduction

Many data mining tasks involve (sometimes implicitly) prediction of linkages among data objects. Examples of explicit link prediction problems include automatic Web hyperlink creation, prediction of genetic or protein-protein interactions, and the record linkage problem. Other well-studied problems can be viewed as an implicit link prediction problem once the data are rendered with a network/graph representation. Examples are abundant. Information retrieval (Salton 1989) can be viewed as dealing with prediction of links between words and documents in a word-document bipartite graph representing word occurrences. Recommender systems (Resnick and Varian 1997) can be viewed as services predicting links between users and items in a user-item bipartite graph representing preferences or purchases. As a generic data mining problem, link prediction has recently received substantial interest in the field of relational learning (Getoor 2003). Relational or multi-relational learning (Dzeroski and Lavrac 2001) deals with richly structured data, which may be described by a relational database or using relational or first-order logic. Objects of multiple types can be linked with each other. Many methods have been developed to exploit the relational structure to predict inherent attributes of the data objects and existence of potential linkages among data objects.

Prior work on link prediction has been primarily formulated based on a *static network* setup, where a partial network structure is known and the objective is to predict the hidden links. In such a static network, link occurrence is typically modeled as a one-time event and the primary interest is on the existence of the link. For example, one may be interested to know whether a customer will purchase a product in the future or whether an author will ever collaborate with another author in the future. In many application settings that involve dynamic evolving networks, however, the link occurrence is preferably modeled as a sequence of binary states or occurrence frequencies, rather than a single binary state regarding its presence. For example, in the surveillance context, the time-series frequency patterns of the communication linkages among a group of monitored targets are much more informative than merely the communication network structure, which represents for each pair of monitored targets whether a communication ever occurred or not. Much richer information could be extracted from the frequency time series of the link

occurrences, such as the periodic patterns and temporal trends of the communication intensities. Generally, link prediction methods based on the static graph representation fall short when the repeated occurrences of the links are of central interest and temporal patterns are the primary feature of the application domain. To the best of our knowledge, no prior work in the literature deals with link prediction taking as input the link occurrence frequency time series. In this paper, we formally introduce the *time-series link prediction problem* and give example applications in which such a formulation may be superior to the existing static graph formulation.

Taking the link occurrence time series of a dynamic network as input, one may naturally attempt to treat the occurrence of each possible link as a stochastic process and adopt the multivariate time-series analysis framework. However, the large number of potential links poses serious challenges on the applicability of the standard time-series analysis techniques. Most importantly, the special graph-based structural dependencies among the links need to be incorporated into a time-series link analysis framework. The link prediction literature is largely based on this fundamental assumption that the network structure itself has predictive information regarding the hidden or future links. Theoretically, an ideal time-series link analysis framework would need to integrate temporal and structural link dependencies simultaneously.

In this study, we propose a hybrid link prediction approach us both the intralink time-series patterns and interlink dependencies, which represents the first step toward the ultimate goal of the integrated time-series link prediction framework. We focus our study on unipartite graphs in which links may represent communication events between a pair of entities (nodes), typically with repeated occurrences. We first investigated a time-series model for link prediction, in which the occurrence of each link is modeled as an independent time series. Specifically, for each link, we built an *autoregressive integrated moving average* (ARIMA) model based on its past occurrence series. Such a model ignores any interlink correlation information, which is the main data pattern employed by the existing static graph link prediction methods (e.g., paths and cluster patterns). Under this model, the future occurrence of a particular link is entirely conditioned on its past occurrences. Despite the simplicity of this univariate link time-series model, we found that this algorithm achieved performance comparable with as existing link prediction methods under the static graph representation in our experiments with e-mail and coauthorship data. We then investigated hybrid link prediction methods that combine the power of the time-series model in predicting repeated link occurrences and the ability of static graph link prediction methods to identify new link occurrences. Our experiments showed that such hybrid methods achieved significantly better performance than the time-series model and static graph methods alone. Our findings present strong evidence for the potential value of time-series information of link occurrences in applications such as communication surveillances. The fact that us a simple univariate link time-series model can already lead to significantly improved link prediction performance provides important justification for further investigation into multivariate link time-series models that incorporate the temporal and structure link dependencies.

The rest of the paper is organized as follows. Section 2 reviews the relevant literature on link prediction and time-series analysis. Section 3 introduces the time-series link prediction problem and discusses example applications. Section 4 presents the use of existing static graph link prediction methods for this problem. Section 5 introduces a time-series link prediction algorithm based on univariate link ARIMA models and the hybrid approaches combining the time series and existing static graph link prediction methods. Section 6 presents the experimental study on predicting e-mail and coauthorship links. Section 7 summarizes the main conclusions and discusses the future directions of our research.

2. Literature Review

In this section, we review relevant research in link prediction and provide a brief overview of time-series analysis.

2.1. Link Prediction

In many contexts, the link structure itself is the critical data pattern exploited for prediction. A wide range of problems can be viewed as prediction of links based on the observed link structure, including information retrieval (Salton 1989) (predicting query-document links based on a document-word network), collaborative filtering (CF) recommendation (Resnick et al. 1994) (predicting user-item links based on a user-item interaction matrix), record linkage problem (Winkler 1994) (predicting links among records with the same identity), and protein/genetic interaction modeling (Goldberg and Roth 2003) (predicting underlying protein/genetic interactions based on interaction networks observed from experiments). Many algorithms developed for these problems also work for the generic link prediction problem. On the other hand, advances in the link prediction problem will have potential implications in these different application domains.

Recently, the link prediction problem has been formulated as a generic data mining task within the field of relational learning. Various relational learning

methods were proposed for link prediction, typically exploiting both the link structure itself and rich descriptive attributes of data objects. Examples of relational learning link prediction algorithms include probabilistic relational models (Getoor and Sahami 1999, Getoor et al. 2002), relational Markov networks (Domingos and Richardson 2001), structural logistic regression models (Popescul and Ungar 2003), and stochastic relational models (Yu et al. 2006). These models can be directly applied to *abstract graphs* (networks with no vertex and edge attributes), where link structure is the only source of predictive data patterns. However, these models are typically unable to capture the complex graph-based data patterns such as paths, cycles, and clusters.

The link prediction problem on abstract graphs is the focus of our study. Liben-Nowell and Kleinberg (2003) studied the abstract graph link prediction problem for social networks of coauthorship networks. Many algorithms developed in fields that deal with problems with an implicit abstract graph representation are also suitable for abstract graph link prediction. CF recommendation algorithms are a prominent example of such implicit link prediction algorithms. A variety of CF algorithms have been proposed in the literature, including standard user-based and item-based neighborhood algorithms (Resnick et al. 1994), cluster and generative models (Hofmann 2004), advanced matrix analysis approaches (Sarwar et al. 2000), and graph-based algorithms (Huang et al. 2004a, b; 2002). These algorithms specifically deal with link prediction in user-item bipartite graphs. They can be directly applied to any bipartite graph link prediction and be applied to unipartite graphs with minor adaptation.

2.2. Time-Series Analysis

Almost all of the entire literature on link prediction has been formulated based on a *static network* setup, where a partial network structure is known and the objective is to predict the hidden links of the underlying complete network. In such a static network, link occurrence is modeled as a one-time event, and the primary interest is on the existence of the linkages rather than the timing of the occurrence or frequency of occurrences. However, in many application settings that involve dynamic evolving networks/graphs, link occurrence is preferably modeled as a sequence of binary states or occurrence frequencies. In this scenario, if we represent the occurrence of a link at a particular time using a random variable (binary or real valued, depending on whether binary occurrences or occurrences are modeled), we are dealing with multivariate time-series data with an exceptionally large number of variables.

Time-series analysis is a well-established field in statistics that provides systematic approaches to modeling of data with time correlations. In this paper, we focus on the time-domain approach because it is typically more appropriate for dealing with possibly nonstationary, shorter time-series with a focus on forecasting future values (Shumway and Stofer 2000), as is the case for our time-series link prediction problem. We specifically focus on the multiplicative models, represented by a systematic class called ARIMA models (Box and Jenkins 1970). These models assume that the observed data result from products of factors involving differential or difference equation operators responding to white noise input. The ARIMA model includes the commonly used autoregressive and moving average models as special cases and is the most widely used model with many applications such as statistical process control (Alwan and Roberts 1988), financial forecasting (Porter-Hudak 1990), biomedical dynamics modeling, and Web traffic modeling and forecasting (Bolot and Hoschka 1996). In this study, we focus on using the standard ARIMA model for time-series link prediction.

2.3. Relevant Literature to Time-Series Link Prediction

The limitation of the static view of the network data has been generally recognized. There have been many recent studies on dynamic or evolving networks that consider temporal connectivity data. Most of these studies convert the temporal connectivity data into a sequence of nonoverlapping network snapshots by aggregating links within each discrete unit of time into a graph. The majority of such studies extend from the main body of network science literature to characterize the time-varying structure of the graph series, such as density and diameter (Leskovec et al. 2007), subgraph and cycle structures (Vazquez et al. 2005), and cluster-formation patterns (Holme et al. 2007), based on empirical temporal network data. One recent study (Berger-Wolf and Saia 2006) also proposed algorithms for deriving information about the evolution of local group formation structures within a temporal graph series.

Most previous studies using temporal network data focus on the characterization of the structural change over time. As previously mentioned, to the best of our knowledge, no prior studies have used the frequency time-series of link occurrences as input for link prediction. We have only identified three studies specifically relevant to link prediction using temporal information. O'Madadhain et al. (2005) predict that links occur in a particular time period by creating features from a static graph, constructed based on events that occurred in a certain number of time periods before the target time period. Although no

sufficient technical details were provided in the paper, it seemed that the temporal information was not exploited in a principled manner because it was only used for static graph construction. Sarkar and Moore (2005) proposed an algorithm for temporal link prediction that extends the latent space (or generative model) algorithm for static graph link prediction to consider temporal correlation of latent locations of nodes but imposing a Markov assumption that the latent location of a node within the state space at time $t+1$ is independent of all previous locations given its latent location at time t. A recent working paper (Potgieter et al. 2009) proposed using a temporal extension of the static graph metrics, such as the percent change of the number of common neighbors over a period of time for link prediction. Although these are interesting ideas for exploiting the temporal information to enhance link prediction, no prior study has used time-series analysis for link prediction by treating the temporal graph data as a collection of a link occurrence frequency series as proposed in this study.

3. The Time-Series Link Prediction Problem

3.1. Problem Formulation

The time-series link prediction problem is formally introduced as follows. In this paper, we focus exclusively on undirected unipartite graphs and adopt the discrete-graph series representation of the temporal network data. Let V be the list of vertices, $V = (1, 2, \ldots, N)$. Because we are interested in dealing with weighted graphs, we adopt the adjacency matrix representation. A graph series is a list of graphs $(G_1, \ldots, G_T)$ corresponding to a list of symmetric adjacency matrices $(M_1, \ldots, M_T)$. Each M_t is an $N \times N$ matrix with nonzero elements $M_t(i, j)$ corresponding to edges in $E(G_t)$: $(i, j) \in E(G_t)$. The value of $M_t(i, j)$ is the frequency of occurrences of the undirected edge (i, j) during the time period t, which can also be viewed as an integer number label of the edge in G_t. Given such a weighted graph series, the time-series link prediction problem is aimed at predicting the occurrence probabilities of edges at time $T+1$. In many situations, more attention is placed on the ranking of the occurrence probabilities of edges instead of the exact occurrence probability of each edge. In this paper, the output of the link prediction problem will be specified as an $N \times N$ score matrix S with each element $S(i, j)$ being a link occurrence score that is proportional to the predicted occurrence probability of edge (i, j). This setup would encompass nonprobabilistic link prediction methods, as well as the probabilistic methods that directly output link occurrence probabilities.

The prominent feature of the above formulation is the availability of the time-series link occurrence frequency. The extant literature on link prediction exclusively used a static network setup, where the partial structure of the network is used to predict the potential presence of other links. For example, Liben-Nowell and Kleinberg (2003) studied the link prediction problem for social networks, where the goal is to predict future collaborations based on the collaboration history of authors in a research field. In their study, a binary undirected graph is used to represent the collaboration history, where the presence of a collaboration edge between two author vertices represents *at least* one paper coauthored by the two. Under this formulation, the frequency information of the link occurrences and the time dimension are both ignored, because multiple coauthored papers over a long period of time are simply represented as a single binary link in the graph. Corresponding to this representation, the predictive objective is exclusively on the occurrence probability of new collaborations, whereas the repeated old collaborations are generally not studied.

Using a static network representation makes sense for certain application domains. For example, if we are interested in predicting consumer–movie links using a consumer–movie network constructed using past records of movie-seeing activities, it is reasonable to take a static representation and focus on predicting whether a consumer will *ever* see a particular movie that he or she has not seen before (i.e., predicting the occurrence probability of a binary consumer–movie link). However, in many other cases, this static network representation may not serve the modeling purpose. In the next subsection, we discuss the example application domain for communication surveillance, with which the time-series link prediction formulation is critical.

3.2. Example Applications

Time-series link prediction can find important applications in communication network monitoring and surveillance contexts, mainly because of repeated occurrences of communication links and time-sensitive analyses of link occurrences. The link occurrence probability predictions within a communication network can be used in two general ways: (1) identifying links that are not observable or links that will occur in the future, and (2) identifying anomalous links.

The recent surge of academic research in social network modeling has been largely motivated by the task of analyzing and monitoring terrorist networks. One major objective in analyzing terrorist networks is to conjecture that particular individuals are working together (or communicating with each

other) even though their current interactions cannot be observed. This is intuitively a link prediction problem. For a fixed set of monitored targets with repeated interactions over a long period of time, we are not only interested in whether two individuals ever had interactions, but also whether they are interacting with each other at the present time or in the near future, regardless of whether or not they have interacted previously. Here, the temporal pattern of the two individuals' past interactions may provide crucial information for building an accurate link prediction model. Adopting a static network representation and only focusing on the all-time existence of the link between two individuals could be misleading. In other surveillance or criminal investigation contexts, we can find similar situations, where the ability to uncover the unobserved links at the present time or in the near future is desired.

Anomalous link detection, on the other hand, focuses only on the observed links (Rattigan and Jensen 2005). The idea is similar to outlier detection in statistics. The time-series link prediction models give the probability of the event of a link occurring at a particular time. The small-probability event is regarded as anomalous, corresponding to certain special events that rarely occur under normal situations. For computer network traffic monitoring, one can use the predicted probability of communication between two IP addresses based on the network traffic time-series in the past. From these predicted probabilities, the highly anomalous (unlikely) communications can be identified, which could lead to identification of network abuses or malicious intrusions. Similarly, e-mail or other forms of communications between individuals in an organizational context or any kind of group or community can be analyzed to identify anomalous communications. These unusual communications can lead to further investigations. This anomaly detection capability is particularly important for communication channels through which voluminous information is exchanged on the real-time basis.

4. Static Graph Algorithm Approaches

In this section, we briefly discuss commonly used link prediction algorithms under a static graph representation (or static graph link prediction algorithms) and the use of these algorithms in the time-series link prediction context. These algorithms ignore the time dimension and frequency of link occurrences.

To predict links that would have occurred at time $T+1$, under a static graph approach, each of the weighted graph G_t in the graph series $(G_1, \ldots, G_T)$ is first reduced to a single-weighted graph $G_{1\sim T}$ with the corresponding adjacency matrix $M_{1\sim T}$, where $M_{1\sim T}(i,j) = \sum_{t=1}^{T} M_t(i,j)$. The graph $G_{1\sim T}$ is further reduced to a binary graph $G^*_{1\sim T}$ with a binary adjacency matrix $M^*_{1\sim T}$, where $M^*_{1\sim T}(i,j) = 1$ if $M_{1\sim T}(i,j) > 1$, and 0 otherwise. Taking $G^*_{1\sim T}$ as input, static graph link prediction algorithms produce the score matrix S. Most if not all literature adopts such a static graph representation. In this section, we briefly introduce six commonly used static graph link prediction algorithms. Among these, four algorithms, *common neighbors, preferential attachment, Adamic/Adar*, and *Katz*, were included in Liben-Nowell and Kleinberg (2003) for social network link prediction. Common neighbors, Adamic/Adar, and Katz were selected since the study showed that these three algorithms had consistent superior performances compared with a large number of other algorithms, including *graph distance, SimRank, PageRank*, and others. We also included the worst-performing algorithm, *preferential attachment*, in that study. In addition, we also include two commonly used algorithms for link prediciton, *generative model* and *spreading activation* algorithms, which were not included in Liben-Nowell and Kleinberg (2003).

Common Neighbors. In graph $G^*_{1\sim T}$, two nodes, i and j, are neighbors of each other if edge $(i,j) \in E(G^*_{1\sim T})$. The common neighbors (CN) algorithm simply uses the number of common neighbors of a pair of nodes as the score in the score matrix: $S(i,j) = \|\{k \in V \mid (i,k) \in E(G^*_{1\sim T}), (j,k) \in E(G^*_{1\sim T}), k \neq i, j\}\|$. Under the matrix representation, $S = M^*_{1\sim T} \cdot M^*_{1\sim T}$. The CN algorithm captures the basic notion of link transitivity. Intuitively, a direct tie between i and j is considered to be more likely to form when more transitive ties between nodes i and j in the form of $i-k-j$ are observed. Such a notion has been extensively studied in social science regarding the fundamental balance theory of social interactions (Heider 1946).

Preferential Attachment. In graph $G^*_{1\sim T}$, the degree $d(i)$ of node i is defined as the number of edges incident on node i: $d(i) = \|\{j \in V \mid (i,j) \in E(G^*_{1\sim T})\}\|$. The preferential attachment (PA) algorithm is motivated by the preferential attachment phenomena (Barabasi and Albert 1999) discovered in a variety of real-world complex systems. Under this algorithm, the link occurrence score is set to be the product of the degrees of the involved nodes: $S(i,j) = d(i) \times d(j)$. The basic intuition is that links between highly connected nodes are more likely. One can also interpret the PA algorithm as an independence model on link occurrence of nodes:

$$\begin{aligned}&\Pr(\text{a link occurs between } i \text{ and } j)\\ &\quad = \Pr(\text{a link that occurs on } i)\\ &\qquad \times \Pr(\text{a link that occurs on } j).\end{aligned}$$

Adamic/Adar. Adamic and Adar (2003) proposed a measure of similarity between two home pages x

and y as $\sum_{z:\text{features shared by } x,y} (1/\log(\text{frequency}(z)))$. Adapting this for link prediction, we can set the link occurrence score between i and j to be $S(i,j) = \sum_{k\in V,(i,k)\in E(G^*_{1\sim T}),(j,k)\in E(G^*_{1\sim T})} (1/\log(d(k)))$. It is clear that this measure extends from the CN algorithm by assigning a weight for each common neighbor that is inversely related to its log degree (or occurrences in links). One should note that this type of similarity measure has been well studied in the information retrieval literature, where the inverse document frequency is used as the weight for each word for "common-word"-based document similarity computation.

Katz. The Katz (KZ) algorithm (Katz 1953) differs from the CN and Adamic/Adar (AA) algorithms in that it goes beyond paths of length 2 between the target node pair. The Katz algorithm sums over all paths of varying lengths, exponentially damped by length to give short paths higher weights. In our context, $S(i,j) = \sum_{l=1}^{\infty} \beta^l \cdot \|\text{paths}^{\langle l\rangle}_{i,j}\|$, where $\text{paths}^{\langle l\rangle}_{x,y}$ represents the set of paths between i and j of length l, and β controls the exponential damping. It was shown in Liben-Nowell and Kleinberg (2003) that the score matrix can be derived as $(I - \beta M^*_{1\sim T})^{-1} - I$.

Generative Model. Under this approach, latent class variables are introduced to explain the link occurrence patterns (Hofmann 2004, Ungar and Foster 1998). Typically, one can use one latent class variable to represent the unknown cause that governs the link occurrence process. The graph $G^*_{1\sim T}$ is considered to be generated from the following three probabilistic process: (1) select a node i with probability $P(i)$; (2) choose a latent class with probability $P(z \mid i)$; and (3) generate link (i,j) (i.e., adding (i,j) to $E(G^*_{1\sim T})$) with probability $P(j \mid z)$. Thus the probability of observing a link between i and j is given by $P(i,j) = \sum_z P(i)P(z \mid i)P(j \mid z)$. The prior and conditional probabilities, $P(i)$, $P(z \mid i)$, and $P(j \mid z)$, are estimated using the *expectation maximization* algorithm (Dempster et al. 1977) to maximize the log-likelihood function of observed links $L = \sum_{(i,j)\in E} \log P(i,j)$. The link occurrence score matrix is set using the estimated probabilities: $S(i,j) = \Pr(i,j)$.

Spreading Activation. The spreading activation (SA) algorithm explores the ensemble of paths connecting the vertex pair of all lengths and heuristically relates a larger number of paths of different lengths with higher link probability, similar to the KZ algorithm. A *Hopfield network*-based implementation of the SA algorithm has been shown to have competitive performance in previous studies (Huang et al. 2004a). In this approach, each node in the graph is assigned an activation level, μ_j, $j = 1,\ldots,N$. To compute the connectedness score between the node i and all other nodes in the graph, node i is set to have activation level 1 ($\mu_i = 1$), and the activation levels of all other nodes are set to 0. After initialization, the algorithm repeatedly performs the following activation procedure: $\mu_j(t+1) = f_s[\sum_{i=0}^{n-1} t_{ij}\mu_i(t)]$, where f_s is the continuous *SIGMOID* transformation function or other normalization function; $t_{ij} = \eta$ $(0 < \eta < 1)$ if (i,j) is in the edge set of the graph. The algorithm stops when activation levels of all nodes converge. The final activation levels μ_js $(j \neq i)$ give the connectedness scores between i and other nodes.

The six static graph link prediction methods introduced above use an unweighted static graph representation in which only the binary link occurrences information is used. Between this simplest representation and the original weighted graph time-series representation in our proposed time-series link prediction problem, one can have an intermediate representation of weighted static graphs that preserve the cumulated frequencies of link occurrences but ignore the temporal pattern of the frequencies. The standard unweighted static graph link prediction algorithms introduced can be adapted straightforwardly to exploit the link occurrences frequency information. For the CN algorithm, the score matrix can be computed from the matrices corresponding to the weighted static graph: $S = M_{1\sim T} \cdot M_{1\sim T}$. The PA algorithm can be adapted by redefining the degree of a node i in the weighted graph context as $d(i) = \sum_j M_{1\sim T}(i,j)$. The AA algorithm can be adapted similarly by using the degree definition for the weighted graph. The KZ algorithm can simply use $M_{1\sim T}$ in deriving the score matrix as $(I - \beta M_{1\sim T})^{-1} - I$. The generative model (GM) algorithm can be adapted by incorporating the number of occurrences of each link into the likelihood function $L = \sum_{(i,j)\in E} M_{1\sim T}(i,j) \log P(i,j)$ as in Hofmann (1999). The SA algorithm can be adapted by defining t_{ij}: $t_{ij} = \eta^{1/M_{1\sim T}(i,j)}$ $(0 < \eta < 1)$. These adapted algorithms still mainly rely on interlink dependencies to predict links but take the total number of link occurrences into consideration.

5. Time-Series Link Prediction Approaches

In this section, we first introduce the time-series model that exploits the time-series information for link prediction. We then explore hybrid prediction methods integrating the time-series predictions and existing static graph model predictions.

5.1. Link Occurrence Time-Series Model

For each link (i,j), $(M_1(i,j),\ldots,M_T(i,j))$ gives the time-series of its occurrence frequency. To use the time-series frequency information of link occurrences, we use the popular ARIMA model (Box and Jenkins 1970) to fit a time-series model for each link occurrence series. For ease of illustration we denote $M_t(i,j)$

as x_{ijt} in this section. Viewing the occurrence frequency series x_{ijt} as a process, it is said to be ARIMA(p, d, q) if $\phi(B)(1-B)^d x_{ijt} = \alpha + \theta(B)w_t$, where B is the backshift operator: $B^l x_{ijt} = x_{ijt-l}$, $\phi(B)$ and $\theta(B)$ are polynomial functions of the backshift operator: $\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p$ and $\theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^q$, α, $\phi_1, \ldots, \phi_p$, $\theta_1, \ldots, \theta_q$, d are constants, and w_t is a white noise series with mean zero and variance σ_w^2. The ARIMA formulation generalizes from several commonly used time-series models including autoregressive and moving average models. For example, ARIMA$(0, 1, 1)$ corresponds to the exponentially weighted moving averages (EWMA).

For model selection, we set the search space by allowing $p = 0, 1, 2, 3$, $d = 0, 1$, and $q = 0, 1, 2, 3$. We adopted the commonly used AIC measure (Akaike 1974) to assess the quality of the model, which can be derived as

$$\ln\left(\frac{\sum_{t=\max(p,q)+1}^{T}(x_{ijt} - \hat{x}_{ijt})^2}{T - \max(p, q)}\right) + \frac{2(p+q+2)}{T - \max(p, q)}.$$

The model with the smallest AIC score was chosen as the final model which was then used to provide the predicted link occurrence frequency at $T+1$, $\hat{x}_{ijT+1}$, and its prediction error $\mathrm{sd}(\hat{x}_{ijT+1})$. The link occurrence score for the link (i, j) is set to be the predicted probability for the link occurrence frequency at $T+1$ to be greater than 1: $S(i, j) = \Pr(\hat{x}_{ijT+1} > 1)$.

Modeling the weighted graph series as a set of independent link occurrence series gives the simplest time-series model for link occurrences. The straightforward extension would be to consider the temporal correlations among the links. Theoretically the complete model should consider the correlations among all links simultaneously using multivariate time-series analysis frameworks such as vector autoregression and state-space models (Gilbert 1995). However, the large number of links makes such models intractable. In this paper we focus on the independent link occurrence time-series model and assess whether such a minimum model can produce useful predictions regarding future link occurrences. We leave more advanced multivariate time-series analysis for future research. As will be seen later, our approach demonstrates strong predictive power on future link occurrences. It clearly justifies further investigation into more advanced multivariate time-series models for further performance improvement.

5.2. Hybrid Approach to Time-Series Link Prediction

A common feature of existing link prediction approaches under the static graph representation is that the link occurrence probability of the link (i, j) is determined by other links related to it. The CN and AA algorithms rely on the number of occurrences of link pairs of the form $((i, k), (k, j))$. The PA algorithm relies on the degrees of node i and node j (total numbers of links associated with node i and node j). For a densely connected graph, the link (i, j) itself only contributes one degree to each node; therefore the preferential attachment score does not really reflect the information from the link (i, j). The GM algorithm estimates the conditional probabilities associated with each node and the latent-class variable $P(z \mid i)$ and $P(j \mid z)$ and relies on them to derive the joint probability of observing the link (i, j). Again, the link (i, j) itself, as one instance of the data used to fit the conditional probabilities, has only a minor effect on the conditional probability estimates, especially when nodes i and j are linked to many other nodes, because the conditional probability estimates are derived from all links incident on nodes i and j. The KZ and SA algorithms consider all possible paths connecting node i and node j. These paths can be considered as the reachable links from the link (i, j). Although the link (i, j) gets the highest weight among all paths connecting nodes i and j, within a densely connected graph, that single link again only represents a small fraction of the total connectedness score. In summary, these link prediction algorithms under the static graph representation rely heavily on the interlink dependencies to make predictions.

The univariate time-series link prediction model, on the other hand, relies entirely on the temporal dependencies of the individual links themselves and ignores any interlink dependencies. In fact, our time-series algorithm can only give predictions on the links that have occurred in the background graph series. It can be argued that our proposed univariate time-series link prediction model exploits data patterns that are almost conceptually orthogonal to those captured by the static graph link prediction algorithms. It makes intuitive sense to combine the two approaches to achieve improved link prediction performance. Ultimately the graph-based interlink dependencies (e.g., paths and neighbors) should be extended to capture time-series information and be tightly integrated into a time-series link prediction model. We have limited our investigation to simple combination methods such as average and product in this paper and leave more advanced hybrid algorithms for future research.

Figure 1 summarizes the hybrid time-series link prediction algorithm, which takes as input an adjacency matrix series $(M_1, \ldots, M_T)$ and a static link prediction algorithm and produces a link occurrence probability score matrix for time $T+1$. Figure 1 illustrates the algorithm using unweighted static graph representation for static link prediction and uses product to combine the static link and time-series prediction scores. We can also use weighted static graph

Hybrid Time-Series Link Prediction Algorithm

Input: Adjacency matrix series $(M_1, \ldots, M_T)$, where $M_t(i, j)$ gives the occurrence frequency of edge (i, j) during time t, $i, j = 1, 2, \ldots, N$; static graph link prediction algorithm g: $M \to Z$, where M is an adjacency matrix corresponding to a graph and Z is a link occurrence probability score matrix.

Output: S, $S(i, j)$ gives the probability score of edge (i, j) at time $T+1$.

Algorithm:

1. Reduce the adjacency matrix series to a single adjacency matrix corresponding to a static graph representation
   1.1 Compute $M_{1\sim T}$: $M_{1\sim T}(i, j) = \sum_{t=1}^{T} M_t(i, j)$
   1.2 Compute $M^*_{1\sim T}$: $M^*_{1\sim T}(i, j) = 1$ if $M_{1\sim T}(i, j) > 1$, 0 otherwise
2. Compute static graph link predictions: $S_S = g(M^*_{1\sim T})$
3. Estimate univariate time-series link prediction model
   3.1 Construct a zero $N \times N$ matrix S_T
   3.1 Repeat for each edge (i, j)
   3.2 Construct the link occurrence frequency series $\{x_{ijt}\}$, where $x_{ijt} = M_t(i, j)$
   3.3 Repeat for $p = 0, 1, 2, 3$
   3.4 Repeat for $d = 0, 1$
   3.5 Repeat for $q = 0, 1, 2, 3$
   3.6 Estimate model ARIMA(p, d, q) and obtain the AIC score
   3.7 Identify the model with best AIC score and obtain $\hat{x}_{ijT+1}$ and $\mathrm{sd}(\hat{x}_{ijT+1})$
   3.8 $S_T(i, j) = \Pr(\hat{x}_{ijT+1} > 1)$.
4. Combine static graph and time-series prediction to obtain S:
   4.1 Normalize the score matrices: $S_S = S_S/\mathrm{sum}(S_S)$, $S_T = S_T/\mathrm{sum}(S_T)$
   4.2 Identify minimum nonzero scores in S_S and S_T: m_S and m_T
   4.3 Repeat for $i = 1$ to N
   4.4 Repeat for $j = 1$ to N
   4.5 $S(i, j) = (S_S(i, j) + m_S/\alpha) \times (S_T(i, j) + m_T/\alpha)$, $\alpha > 1$.

Figure 1 Hybrid Time-Series Link Prediction Algorithm

representation and use the adapted static graph link prediction algorithms described in §4 by skipping Step 1.2 in the algorithm and producing the link probability score matrix S_S directly from $M_{1\sim T}$. In Step 4 of the algorithm, adding a fraction of the minimum score (dividing by the parameter $\alpha > 1$) to the link probability scores is to prevent loss of predictive information when one of the two scores to be combined is zero. This design is based on the understanding of the complementary nature of the two link prediction approaches. Because only the ranks of the predicted scores matter but not the exact scores, any value of α should generally work as long as it is greater than 1. Setting the value larger for α, we essentially give a heavier penalty for the predicted occurrence probability of the link if one of the two approaches generates a zero score.

6. Experimental Study

6.1. Data

We used two data sets to evaluate our proposed approach to link prediction given a series of link occurrence frequencies. We focus in this paper on presenting the results of the Enron e-mail data set because the capability to predict the likelihood of link (e-mail) occurrences in this context has the direct practical relevance for surveillances purpose. For example, being able to detect the anomalous e-mail communication among a group of surveillance targets may allow the analysts to identify important leads for critical events. We also report the experimental results on the coauthorship links within the field of high-energy particle physics. Here, a coauthored paper represents the extensive communications among the coauthors. We use this data set mainly as a validation data set to understand the behavior of the time-series and hybrid link prediction algorithms in different domains. Meanwhile, the predicted link likelihood among the authors may provide an interesting approach to detect innovative groundbreaking papers in the field, as shown in Rattigan and Jensen (2005).

6.1.1. Enron E-mail Data Set. We used a preprocessed version of the Enron e-mail data set provided by Shetty and Adibi (2005) (available at ftp://ftp.isi.edu/sims/philpot/data/enron-mysqldump.sql.gz). This data set contains 252,759 e-mails from Enron employees, mainly senior managers. In our study, we have focused on e-mails sent from *and* to these 151 people. The final e-mail collection contained 21,254 e-mails during the period from May 11, 1999 to June 21, 2002. We performed the link prediction analysis mainly on the monthly e-mail graphs. In this study, an e-mail graph at month t, G_t, is an undirected graph with edges connecting senders and recipients of e-mails during that month. An edge (i, j) represents that there has been at least one e-mail communication between i and j (either i sending at least one e-mail with recipients including j, or j sending at least one e-mail with recipients including i). The number of e-mails between i and j in month t gives the link occurrence frequency or the weight of the link $M_t(i, j)$. In total, there were 1,526 unique e-mail links with a total number of link occurrences of 47,088. Figure 2 shows the number of e-mail links and link occurrences across the 38 months in our data set marked with major Enron events. The peak level e-mail communication in these data occurred around October 2001, with 531 distinct e-mail links totaling 8,143 link occurrences. The peak month corresponded to Enron's filing \$618 million in losses, transferred from "special purpose entities" as net losses and the start of the SEC inquiry. The Enron e-mail communication context is particularly appropriate for the time-series link prediction setup, because we are interested in the communications among a stable set of individuals with repeated communications—not only whether a

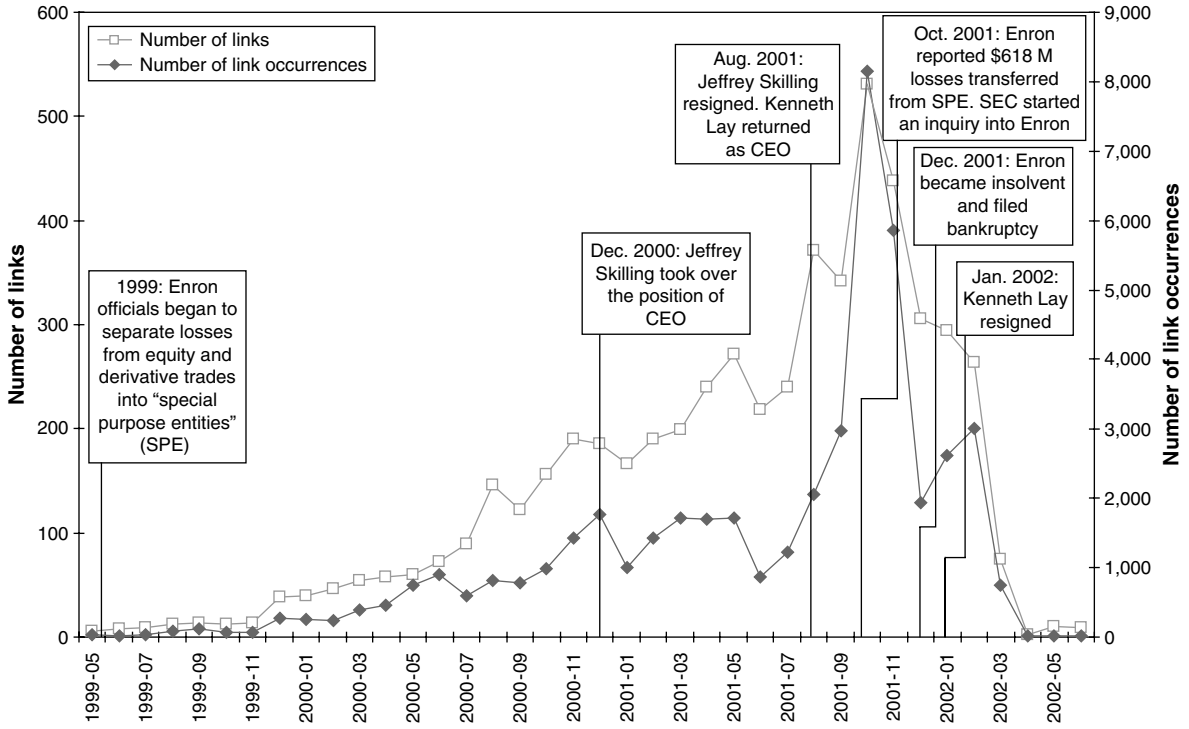

Figure 2 Number of Links and Link Occurrences over Time with Enron Events

pair of individuals has ever communicated or not, but also in who communicated with each other at a particular time.

6.1.2. High-Energy Particle Physics Coauthorship Data Set. The high-energy particle physics coauthorship data set is based on data from the arXiv archive and the Stanford Linear Accelerator Center SPIRES-HEP database provided for the 2003 KDD Cup competition on citation prediction with additional preparation performed by the Knowledge Discovery Laboratory, University of Massachusetts at Amherst (available at http://kdl.cs.umass.edu/data/hepth/hepth-info.html). This data set has been widely used in many studies on modeling of social networks and relational learning in general and on link prediction in particular, including the coauthorship link prediction study (Liben-Nowell and Kleinberg 2003). The complete data set contains 29,555 papers by 9,200 authors with 87,794 coauthorship relationships during the period from 1992 to 2003. Similar to our treatment of the Enron e-mail data set, we decided to focus on the top 102 most productive authors who authored more than 80 papers in this data set. We only included the coauthorship links among these 102 authors in our experiment.

The final data set used for the experiment contained 1,172 papers authored by at least two of the top 102 authors. In the end, 95 authors were included in the experiment data. There are seven top authors who did not coauthor with other top authors. Because some papers had multiple submissions to the database, we used the latest submission time as the timestamp for each paper. The 1,172 papers were submitted between January 1992 and May 2003. We performed the link prediction analysis on the quarterly coauthorship graphs. A coauthorship graph at quarter t, G_t, is an undirected graph with edges connecting coauthors of papers submitted during that quarter. An edge (i, j) in such a graph represents that there has been at least one paper coauthored by i and j. The number of papers containing i and j in the author list in quarter t gives the link occurrence frequency $M_t(i, j)$. In total, there were 280 unique coauthorship links with 1,796 link occurrences. Compared to the e-mail data set, the coauthorship data set contains sparser communication graphs with fewer link occurrences and more stable link occurrences over time.

6.2. Experimental Setup and Implementation Details

In our experiments, we adopted a moving-window approach to evaluate the performance of time-series link prediction algorithms. Given a specified window size T, for each time period t ($t > T$), we use graphs of T previous periods $(G_{t-T}, \ldots, G_{t-1})$ to predict links that occur in the current period (G_t). We refer to $(G_{t-T}, \ldots, G_{t-1})$ as the *background graph series* for G_t. Generally, with longer background graph series our proposed approach is able to capture the potential within-link occurrence dependency patterns to

complement the typical methods based on cross-link dependency patterns. On the other hand, one may argue that the most recent graph G_{t-1} may contain the most critical temporal dependency information for predicting G_t. While the ARIMA model should be able to detect this pattern by arriving at a short-length autocorrelation model in theory, we also included in our experiment a simple algorithm by using the occurrence counts in G_{t-1} as the link occurrence scores for G_t, which we refer to as the $T-1$ *algorithm*. As we will see later in the experimental results, the proposed ARIMA model consistently significantly outperformed the $T-1$ algorithm. For nodes that are not linked to any other nodes in the background graph series, no link prediction algorithm is able to give any informative predictions. Therefore, in this study, we focus on predicting links between nodes that have appeared in the background graph series.

To evaluate the link prediction performance, we construct a receiver operating characteristics (ROC) curve (Bradley 1997) with the x-axis being the percentage of links not occurring in G_t (*negative links*) that are in the selected links (the number of negative links predicted to be positive divided by the total number of negative links) and the y-axis being the percentage of actually occurring links (*positive links*) that are in the selected links (the number of positive links predicted to be positive divided by the total number of positive links). As we move along the x-axis, the larger percentage of links are selected (by lowering the link occurrence score threshold), and the percentage of actually occurring links that are in the selected links monotonically increases. When all links are selected the percentage of the selected links will be 1. Therefore the ROC curve always starts at $(0,0)$ and ends at $(1,1)$. The ROC curve associated with prediction by randomly selecting links to include will result in a 45-degree line. A perfect link prediction algorithm should assign higher link occurrence scores to those links actually occurring in G_t than all other links, which will result in a ROC curve formed by two line segments, $(0,0)$–$(0,1)$ and $(0,1)$–$(1,1)$. The standard measure to assess the quality of the ROC curve is the *area under the curve* (AUC) measure. The AUC measure is strictly bounded between 0 and 1. The perfect algorithm has an AUC measure of 1 and the random algorithm has an AUC measure of 0.5.

We consider the AUC measure to be the preferred measure of the quality of link prediction over other measures such as precision and recall because one needs to set (often arbitrarily) the number of links to include in the set of predicted links to derive the percentage of links in this set to be actual links (the precision measure) and the percentage of actual links appearing in this set (the recall measure). When we construct the ROC curve, we have a complete characterization of all sizes of the set of predicted links. The AUC measure gives a summary metric for the algorithm's overall performance with different prediction set sizes, while a detailed look into the shape of the ROC curve reveals the performance of the algorithm at each prediction set size.

For each time period t ($t > T$) we first prepared the background graph series containing T graphs. We then reduced these background graph series to a single background graph and applied static graph link prediction algorithms. For standard link prediction algorithms, the background graph is unweighted, only representing the binary occurrences of the links in previous T periods. Because our proposed time-series link prediction algorithm utilizes the frequency information, to have a close comparison, we also created a frequency background graph by setting the frequency of a link as the sum of frequencies of that link in the background graph series and applied the frequency-based modification of the standard link prediction algorithms described in §4. We then applied the proposed time-series link prediction algorithm using the original background graph series and compared it with the two sets of performance measures for static graph link prediction algorithms. Last, we investigate the performance of hybrid methods combining the static graph link prediction algorithms and the proposed time-series algorithm. Because our experimental results showed no substantial difference between the performances of the static graph algorithms with the binary and frequency background graph, the static graph algorithm without frequency modification was used to combine with the time-series algorithm.

6.3. Experimental Results

6.3.1. Enron E-mail Data Set Results. Because our data cover 38 months from May 1999 to June 2002, with the moving-window size set to 13 months, we generated predictions for 25 months from June 2000 to June 2002. Table 1 shows the link prediction performances of the static graph link prediction algorithms, the proposed time-series algorithm (TS), and the $T-1$ algorithm measured by AUC when T is set to 13. The static graph algorithms include the common neighbors (CN), preferred attachment (PA), Adamic/Adar (AA), Katz (KZ), generative model (GM), and spreading activation (SA) algorithms. Two sets of results are reported for these algorithms under the binary and weighted static graph representations. In Table 1, the boldface measures indicate the best-performing algorithm for that month. The last three months had an extremely small number of links. We exclude these three months in our comparison of algorithms but still present the results for the sake of completeness.

Table 1 AUC Measures of the Static Graph and Time-Series Algorithms—Enron E-mail Data

Year-month	No. of target links	Binary static graph						Weighted static graph							
		CN	AA	PA	GM	SA	KZ	CN	AA	PA	GM	SA	KZ	TS	$T-1$
200006	61	0.7704	0.7792	0.7740	0.8301	0.7844	0.8751	0.7744	0.7752	0.7774	0.8402	0.8292	**0.8843**	0.8488	0.7839
200007	84	0.7836	0.7952	0.7575	0.8636	0.7914	0.9112	0.8006	0.8112	0.7780	0.8231	0.8586	**0.9173**	0.8175	0.7591
200008	134	0.7698	0.7805	0.7081	0.8095	0.7564	0.8294	0.7724	0.7716	0.6998	0.7883	0.7730	**0.8340**	0.7653	0.7266
200009	121	0.8492	0.8522	0.7840	0.8844	0.8461	0.9267	0.8467	0.8469	0.7682	0.8781	0.8457	**0.9274**	0.8837	0.8161
200010	142	0.8716	0.8814	0.7578	0.8925	0.8437	0.9232	0.8705	0.8721	0.7342	0.8461	0.8226	**0.9251**	0.8861	0.7933
200011	190	0.8332	0.8393	0.6992	0.8188	0.7924	**0.9093**	0.8317	0.8313	0.6693	0.7916	0.7629	0.9027	0.8253	0.7652
200012	183	0.8781	0.8967	0.7215	0.8864	0.8389	**0.9310**	0.8828	0.8786	0.6981	0.8431	0.8065	0.9283	0.8947	0.8409
200101	162	0.8759	0.8818	0.7271	0.8564	0.8388	**0.9157**	0.8689	0.8669	0.6763	0.8555	0.8034	0.9025	0.8235	0.7565
200102	190	0.9014	0.8990	0.7534	0.8849	0.8812	**0.9380**	0.8952	0.8797	0.6994	0.8465	0.8301	0.9218	0.8912	0.7997
200103	199	0.9153	0.9172	0.7782	0.8904	0.8986	**0.9452**	0.9118	0.9042	0.7516	0.8738	0.8504	0.9376	0.9155	0.8384
200104	229	0.9002	0.9053	0.7254	0.8830	0.8887	**0.9322**	0.9034	0.8944	0.7196	0.8374	0.8152	0.9209	0.8585	0.7911
200105	246	0.8100	0.8180	0.6728	0.8170	0.7864	**0.8424**	0.8134	0.7975	0.6366	0.7955	0.7319	0.8289	0.7803	0.7300
200106	192	0.8410	0.8485	0.6427	0.8450	0.7908	**0.8742**	0.8404	0.8334	0.6490	0.7964	0.7102	0.2178	0.7831	0.7369
200107	228	0.8110	0.8279	0.6570	0.8386	0.7925	**0.8639**	0.8181	0.8116	0.6402	0.8063	0.6976	0.1987	0.7982	0.7208
200108	368	0.8363	0.8504	0.6504	0.8321	0.7945	0.8670	0.8377	0.8261	0.6923	0.8068	0.7070	**0.8762**	0.7594	0.6908
200109	342	0.8619	0.8735	0.6716	0.8640	0.8043	**0.9010**	0.8523	0.8307	0.6485	0.8468	0.7076	0.8727	0.8697	0.7775
200110	530	0.8653	0.8699	0.6568	0.8473	0.7933	**0.8876**	0.8514	0.8275	0.6303	0.8317	0.6987	0.8449	0.7986	0.7466
200111	432	0.8864	0.8952	0.6651	0.8559	0.8140	**0.9055**	0.8496	0.8207	0.6398	0.8539	0.6577	0.8422	0.8374	0.7745
200112	303	0.8999	0.9059	0.6909	0.8920	0.8371	**0.9187**	0.8725	0.8502	0.6950	0.8653	0.6469	0.8575	0.8982	0.8342
200201	293	0.8757	0.8817	0.6625	0.8664	0.8143	**0.9024**	0.8598	0.8347	0.6578	0.8539	0.6453	0.8243	0.8415	0.7362
200202	264	0.8401	0.8422	0.6810	0.8241	0.8065	**0.8606**	0.8114	0.7918	0.7027	0.8153	0.5950	0.7911	0.8059	0.7240
200203	75	0.9198	0.9336	0.5740	**0.9470**	0.8525	0.9422	0.8392	0.7815	0.5437	0.9031	0.5477	0.0429	0.9338	0.7920
200204	2	0.4100	0.5218	0.5064	0.7080	0.3977	0.8819	0.4049	0.4607	0.4152	0.8233	0.3055	0.0853	0.9997	**0.9998**
200205	10	0.6908	0.7397	0.5263	0.7226	0.6222	**0.7814**	0.6581	0.6986	0.6711	0.6774	0.6718	0.1178	0.7630	0.7632
200206	9	0.7502	0.7733	0.6003	0.8221	0.6263	0.8279	0.6644	0.6900	0.6342	0.7490	0.7738	0.0815	0.8971	**0.9217**
Average		0.8259	0.8404	0.6818	0.8473	0.7877	**0.8917**	0.8133	0.8075	0.6731	0.8259	0.7238	0.6994	0.8470	0.7848
Average excluding last 3 months		0.8544	0.8625	0.7005	0.8604	0.8203	**0.9001**	0.8456	0.8335	0.6867	0.8363	0.7429	0.7818	0.8416	0.7697

For 21 of the 22 months, the KZ algorithm with the weighted or unweighted graph had the best performance and therefore had the best overall performance. In the 22nd month, the GM algorithm with the unweighted graph had the best performance. On average the KZ algorithm achieved the AUC measure of 0.9001, significantly better than the second-best-performing algorithm (AA) with a paired two-sample t-test p-value smaller than 0.0001. Generally, the standard link prediction algorithms performed better with the binary graph than the weighted graph. In our later analysis we will focus on analyzing the standard static graph link prediction algorithms under the binary setting.

Our proposed time-series link prediction algorithm was able to achieve an average AUC measure of 0.8416, significantly better than the PA (0.7005) and SA (0.8203) algorithms and not significantly different from the CN (0.8544) and AA (0.8652) algorithms at the 0.01 significance level with the paired two-sample t-tests. Given that our time-series algorithm does not consider interlink correlations at all and only relies on the time correlations of occurrence frequencies of the individual links, its competitive performance relative to the existing static graph link prediction algorithms is fairly surprising. It shows that the temporal distribution of the occurrence frequencies of a link contains valuable information for predicting its future occurrence. This relatively straightforward data pattern actually has greater predictive power than several graph-based interlink dependency patterns used by the static graph link prediction algorithms in our study. The performance of the $T-1$ algorithm makes this message even stronger. By simply using the link occurrence frequency in G_{t-1} as the link occurrence score, we achieved the average AUC measure of 0.7697, significantly better than the PA algorithm.

The comparison between our proposed time-series algorithm and the $T-1$ algorithm reveals that the ARIMA model adds additional predictive power by looking into longer time series of link occurrences. The $T-1$ algorithm uses the most recent link occurrence information for link prediction directly. One may wonder how static graph algorithms will perform taking the most recent background graph G_{t-1} as input—in other words, setting $T=1$. Furthermore, we are also interested in the performance of these algorithms and the time-series algorithm as we vary the value of T. Table 2 presents the average AUC measures of the static graph and time-series link predic-

Table 2 Average AUC Measures of Different Values of T—Enron E-mail Data

	Average AUC of $35-T$ months							Average AUC of 15 months						
T	CN	AA	PA	GM	SA	KZ	TS	CN	AA	PA	GM	SA	KZ	TS
1	0.7857	0.7893	0.6995	0.8087	0.7970	0.8635	—	0.8059	0.7558	0.6562	0.7696	0.7565	0.8312	—
3	0.8357	0.8424	0.7240	0.8575	0.8185	0.8935	—	0.8602	0.8342	0.7017	0.8543	0.7801	0.8864	—
5	0.8483	0.8557	0.7171	0.8598	0.8221	0.8995	—	0.8696	0.8487	0.7058	0.8530	0.7882	0.8989	—
8	0.8483	0.8542	0.7185	0.8583	0.8182	0.9002	0.8428	0.8695	0.8578	0.7134	0.8622	0.8199	0.9019	0.8523
13	0.8544	0.8625	0.7005	0.8604	0.8203	0.9001	0.8416	0.8693	0.8767	0.6806	0.8629	0.8262	0.8998	0.8396
15	0.8609	0.8687	0.6916	0.8635	0.8215	0.9001	0.8424	0.8675	0.8752	0.6777	0.8668	0.8242	0.8989	0.8393
20	0.8676	0.8744	0.6729	0.8690	0.8213	0.8977	0.8403	0.8676	0.8744	0.6729	0.8690	0.8213	0.8977	0.8403

tion algorithms by setting the values of T to 1, 3, 5, 8, 13, 15, and 20. We did not include the performance number for the time-series algorithm when $T \leq 5$ because the time-series would be too short to fit a meaningful ARIMA model. As we vary the value of T, predictions change accordingly. Excluding the last three months of data, we obtain predictions for $35-T$ months. We present the average AUC measure over all possible prediction months for each value of T as well as the average AUC measure over the 15 $(35-20)$ common months prior to April 2002 to have a direct comparison of the effect of T on the prediction performance. We observe that the static graph algorithms achieved similar performance as long as at least three previous months of data were used to form the static background graph. This is expected because for the static graph representation many link occurrences only provide redundant information. The only exception is the PA algorithm, which achieved the best performances when T was between three and eight. The time-series algorithm performed similarly when we varied T from 8 to 20.

Note that the proposed time-series link prediction only applies to the situation where abundant time-series link occurrence information is available. In general, the background graph series needs to contain at least eight graphs for the proposed approach to be effective. Therefore with each graph in the series constructed using communications occurring in one day, one week, or one month, we would need eight days, two months, or eight months of historical data, respectively, to apply the proposed approach. The decision on the length of time to form each graph in the background graph series is domain and application dependent. For example, using monthly data to form coauthorship graphs would most likely result in very sparse graphs, losing the cross-link dependency information. On the other hand, the general frequency of link occurrence also determines the prediction target correspondingly. For example, predicting the coauthorship link within a quarter or a year would probably make more sense than predicting coauthorship links within a particular month. For the Enron e-mail data set, we do have sufficiently frequent link occurrences to form weekly e-mail graphs. Table 3 shows the average AUC measures of the static graph, time-series, and hybrid algorithms with T set to be 13 over the period of 118 weeks excluding the beginning and ending quiet weeks. It is observed that the performances of the algorithms were generally consistent with the monthly graph analyses. The time-series link prediction algorithm achieved the second-best average AUC measure, following the KZ algorithm. By incorporating the time-series predictions, all static graph algorithms achieved significant improvements according to the paired two-sample t-test with a p-value smaller than 0.0001, ranging from 0.37% for the KZ algorithm to 15.81% for the PA algorithm.

Figure 3 shows the numbers of links for which each ARIMA model was chosen as the best model based on the background graph series for October 2001. The first four ARIMA models $(0,1,1)$, $(0,0,1)$, $(0,1,0)$, and $(1,1,0)$ accounted for more than 80% of the links. The distribution for background graph series of other months also generally followed a similar pattern. These patterns show that the model space can be drastically reduced for an ARIMA model fitting without significantly reducing the model fit and prediction performance. Figure 4 shows four example link occurrence series that contain the background series of 13 months and the actual link occurrence frequency in October 2001 ($t = 14$). We observe that links 1 and 4 had qualitatively similar time-series patterns, resulting in the same best-fitting ARIMA model of $(0,1,2)$. Comparing the predicted link occurrence frequency with the actual values shown in Figure 4, we can see that the ARIMA models successfully captured the temporal trend. In particular, the time series for links 2 and 3 would be treated almost identically under the static graph approach. The ARIMA model provided the predicted occurrence probability

Table 3 AUC Measures on Weekly Graphs—Enron E-mail Data

	CN	AA	PA	GM	SA	KZ	TS
Standalone	0.8578	0.8635	0.7762	0.8900	0.8505	0.9300	0.8900
Hybrid	0.9314	0.9321	0.9067	0.9266	0.9263	0.9335	—
Improvement (%)	8.58	7.95	16.81	4.11	8.91	0.37	—

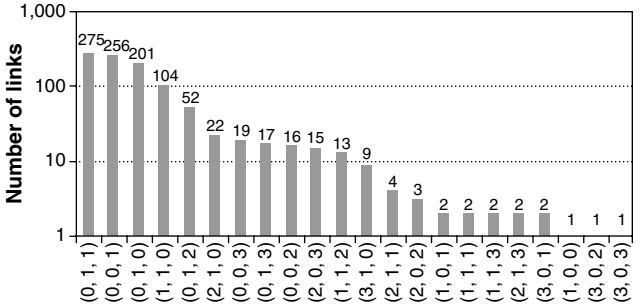

Figure 3 ARIMA Model Distribution—Enron E-mail Data (Oct. 2001)

for link 2 (which did not occur at t) as 0.397, smaller than 0.5 and the other three links.

To better understand the differences among predictions of the algorithms evaluated, we studied the correlation of link occurrence probability score matrices. Because the time-series link prediction algorithm can only make a prediction for links that have occurring previously, we only included predicted scores for these previously occurring links when computing the correlation coefficients. We report the correlation coefficients of the link occurrence probability score matrices for October 2001 in Table 4. These correlation coefficients significantly differ from zero, with p-values smaller than 0.0001 due to the large number of scores included for computation.

As expected, the CN and AA algorithms were highly correlated with a correlation coefficient of 0.9914. The GM algorithm had relatively high correlation with the CN algorithm (0.8316) and the AA algorithm (0.8542). The other algorithm pairs had relatively weak correlations. The correlation between

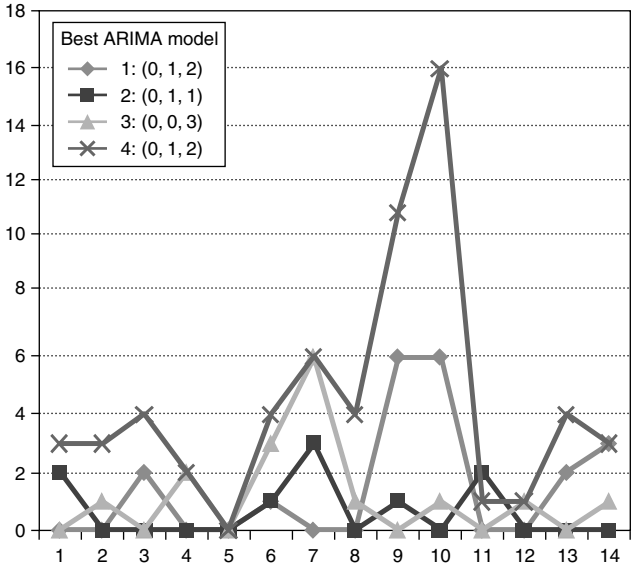

Figure 4 Example Link Occurrence Series and ARIMA Predictions

Note. Predicted frequency for $t = 14$: 1: 1.923, 2: 0.692, 3: 1.154, 4: 3.302; predicted link occurrence probability for $t = 14$: 1: 0.693, 2: 0.387, 3: 0.534, 4: 0.714.

Table 4 Correlation of Link Occurrence Probability Scores—Enron E-mail Data

	CN	AA	PA	GM	SA	KZ
AA	0.9914					
PA	0.6075	0.6025				
GM	0.8316	0.8542	0.5273			
SA	0.7156	0.6791	0.5783	0.5687		
KZ	0.6386	0.6561	0.3333	0.6313	0.4351	
TS	0.5297	0.5449	0.2617	0.5117	0.3371	0.7409

the time-series algorithm and the static graph algorithms ranged from 0.2617 (with the PA algorithm) to 0.7409 (with the KZ algorithm). These observations are generally consistent with our analysis that the proposed time-series link prediction algorithm utilizes the information set that compliments the ones used by the static graph algorithms.

Figure 5 shows the ROC curves of the static graph algorithms and the time-series algorithm for October 2001. It can be seen that the ROC curve of the time-series algorithm shows a different shape from the ones of the static graph algorithms. The ROC curves for the static graph algorithms are generally smooth, while there is clearly a turning point for the ROC curve of the time-series algorithm when the percentage of positive links included reaches 70%. This turning point corresponds roughly to the percentage of links in the October 2001 e-mail graph that occurred previously in its background graph series. This result shows that the time-series algorithm achieves better performance on previously occurring links than other algorithms, including the best-performing KZ algorithm. The segment of the ROC curve for the time-series algorithm after the turning point is close to a straight line. This is fully expected because the time-series algorithm assigns zero scores for the

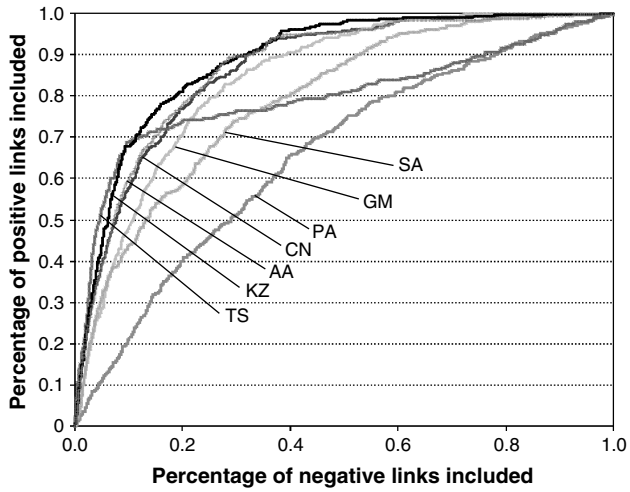

Figure 5 ROC Curves for October 2001—Enron E-mail Data

Table 5 Performance of Hybrid Link Prediction—Enron E-mail Data

Year-month	No. of target links	TS	CN	CN ∗ TS	AA	AA ∗ TS	PA	PA ∗ TS	GM	GM ∗ TS	SA	SA ∗ TS	KZ	KZ ∗ TS
200006	61	0.8488	0.7704	0.8741	0.7752	0.8810	0.7740	0.8537	0.8301	**0.8907**	0.7844	0.8603	0.8751	0.8774
200007	84	0.8141	0.7836	0.8975	0.8112	0.8976	0.7575	0.8644	0.8636	0.8904	0.7914	0.8839	0.9112	**0.9143**
200008	134	0.7719	0.7698	0.8408	0.7716	**0.8465**	0.7081	0.7905	0.8095	0.8223	0.7564	0.8140	0.8294	0.8319
200009	121	0.8864	0.8492	**0.9327**	0.8469	0.9325	0.7840	0.8981	0.8844	0.9276	0.8461	0.9129	0.9267	0.9296
200010	142	0.8790	0.8716	0.9298	0.8721	**0.9332**	0.7578	0.8732	0.8925	0.9211	0.8437	0.9025	0.9232	0.9253
200011	190	0.8253	0.8332	0.9076	0.8313	**0.9139**	0.6992	0.8241	0.8188	0.8596	0.7924	0.8571	0.9093	0.9127
200012	183	0.8942	0.8781	0.9349	0.8786	**0.9436**	0.7215	0.8796	0.8864	0.9236	0.8389	0.9134	0.9310	0.9344
200101	162	0.8239	0.8759	0.9106	0.8669	0.9140	0.7271	0.8541	0.8564	0.8887	0.8388	0.8881	0.9157	**0.9173**
200102	190	0.8889	0.9014	0.9417	0.8797	**0.9420**	0.7534	0.8885	0.8849	0.9244	0.8812	0.9172	0.9380	0.9388
200103	199	0.9162	0.9153	0.9482	0.9042	0.9468	0.7782	0.9158	0.8904	0.9372	0.8986	0.9345	0.9452	**0.9484**
200104	229	0.8585	0.9002	**0.9371**	0.8944	0.9366	0.7254	0.8735	0.8830	0.9177	0.8887	0.9297	0.9322	0.9346
200105	246	0.7803	0.8100	0.8443	0.7975	0.8450	0.6728	0.8016	0.8170	0.8256	0.7864	0.8408	0.8424	**0.8461**
200106	192	0.7832	0.8410	0.8768	0.8334	**0.8814**	0.6427	0.7898	0.8450	0.8615	0.7908	0.8586	0.8742	0.8786
200107	228	0.7995	0.8110	0.8628	0.8116	0.8721	0.6570	0.8182	0.8386	**0.8778**	0.7925	0.8696	0.8639	0.8679
200108	368	0.7594	0.8363	**0.8745**	0.8261	0.8726	0.6504	0.7948	0.8321	0.8629	0.7945	0.8520	0.8670	0.8724
200109	342	0.8683	0.8619	0.9086	0.8307	**0.9140**	0.6716	0.8617	0.8640	0.9103	0.8043	0.8980	0.9010	0.9064
200110	530	0.8056	0.8653	**0.8959**	0.8275	0.8958	0.6568	0.8139	0.8473	0.8813	0.7933	0.8674	0.8876	0.8922
200111	432	0.8417	0.8864	0.9162	0.8207	**0.9191**	0.6651	0.8602	0.8559	0.9113	0.8140	0.8972	0.9055	0.9119
200112	303	0.8982	0.8999	0.9282	0.8502	**0.9301**	0.6909	0.8922	0.8920	0.9200	0.8371	0.9151	0.9187	0.9258
200201	293	0.8415	0.8757	0.9121	0.8347	**0.9132**	0.6625	0.8647	0.8664	0.9016	0.8143	0.9021	0.9024	0.9120
200202	264	0.8053	0.8401	0.8722	0.7918	0.8692	0.6810	0.8483	0.8241	0.8471	0.8065	**0.8766**	0.8606	0.8740
200203	75	0.9338	0.9198	0.9520	0.7815	0.9557	0.5740	0.9171	0.9470	0.9574	0.8525	0.9518	0.9422	**0.9537**
200204	2	**0.9999**	0.4100	0.9280	0.4607	0.9282	0.5064	0.9580	0.7080	0.9359	0.3977	0.9264	0.8819	0.9908
200205	10	0.7625	0.6908	0.8143	0.6986	0.8351	0.5263	0.8020	0.7226	**0.8561**	0.6222	0.8072	0.7814	0.8199
200206	9	**0.8971**	0.7502	0.8721	0.6900	0.8813	0.6003	0.8794	0.8221	0.8871	0.6263	0.8483	0.8279	0.8758
Average		0.8473	0.8259	0.9005	0.8075	**0.9040**	0.6818	0.8567	0.8473	0.8936	0.7877	0.8850	0.8917	0.9037
Average excluding last 3 months		0.8420	0.8544	0.9045	0.8335	**0.9071**	0.7005	0.8535	0.8604	0.8936	0.8203	0.8883	0.9001	0.9048
Improvement (%)				5.86		8.82		**21.85**		3.86		8.29		0.52

remaining links that did not occur previously, essentially picking all links at random to include.

Given the superior performance of the time-series algorithm on predicting the occurrence of previously occurring links and the fact that time-series predictions and static graph predictions only weakly correlate with each other, algorithms that combine the two approaches hold the promise to achieve better prediction performances. We next examine simple hybrid link prediction algorithms using product of link occurrence probability scores produced by the two approaches. Table 5 shows the performance of the hybrid link prediction using score products. For example, the column under CN ∗ TS shows the AUC measure of the hybrid algorithm combining the time-series algorithm and the CN algorithm. Two link occurrence probability score matrices were produced by each individual algorithm, S_{TS} and S_{CN}. Following the algorithm description in Figure 1, we first normalized the two score matrices and then identified the smallest nonzero element m_T and m_S for the two score matrices. The link prediction score matrix was then updated by adding m_T/α (m_S/α) to each element of the matrix. We then obtained the hybrid score matrix $S_{\mathrm{TS*CN}}$ by multiplying the TS and CN scores: $S_{\mathrm{TS*CN}}(i,j) = (S_{\mathrm{TS}}(i,j) + m_T/\alpha) \times (S_{\mathrm{CN}}(i,j) + m_S/\alpha)$. Table 5 shows the experimental results with $\alpha = 2$.

Within the 22 months of our data used in analysis, the best-performing algorithm was always a hybrid algorithm. However, there was no single winning hybrid algorithm. Even though the KZ algorithm was clearly the best-performing algorithm in Table 1, the KZ ∗ TS algorithm only achieved the best performance for 5 of the 22 months, with an average AUC measure of 0.9048. The AA ∗ TS algorithm actually achieved the best performance in the greatest number of months (10 out of 22 months) with the highest average AUC measure of 0.9071. For all six algorithms, the incorporation of time-series prediction scores resulted in improvement in performance, ranging from 0.52% with the KZ algorithm to 21.85% with the PA algorithm. All improvements were statistically significant according to the paired two-sample t-test with p-values smaller than 0.0001. These results are consistent with the analysis of the correlation between the predictions generated by different algorithms. With the highest correlation of 0.7409, the KZ algorithm did not benefit much from the incorporation of the time-series prediction. On the other hand, the worst-performing PA algorithm, with the smallest

correlation of 0.2617, had the greatest improvement by incorporating the time-series prediction.

Other values of α (from 2 to 100) were also tested, and the results obtained were almost identical. We also experimented with using the average instead of the product to combine the static graph and time-series link prediction scores and obtained similar results as shown in Table 5.

6.3.2. Coauthorship Data Set Results. The coauthorship data set spanned over 46 quarters from 1992 to 2003. The window size of 13 produced 33 quarters of prediction data from the second quarter of 1995 to the second quarter of 2003. Table 6 shows the link prediction performances of the static graph link prediction algorithms, the proposed time-series algorithm, and the $T-1$ algorithm measured by AUC. We noticed that the last quarter had an unusually small number of target links, possibly because of incomplete data collection. For analysis purposes, we excluded the results of this quarter. Similar to the e-mail data set results, the KZ algorithm was generally the best-performing algorithm. The KZ algorithm achieved the best performance for 23 of the 32 quarters with the weighted graph. It also achieved the best performance for two quarters with the unweighted graph. The time-series algorithm demonstrated stronger relative performance than with the e-mail data set. It achieved the best performance for the test of the seven quarters. Overall, the KZ algorithm achieved the highest average AUC measure of 0.9318 with the weighted graph (0.9285 with the unweighted graph), followed by the time-series algorithm with an average AUC measure of 0.8998. Similar to the e-mail data set results, the PA algorithm had the worst performance, even compared with the $T-1$ algorithm. For the coauthorship data set, the static graph link prediction algorithms had a slightly better performance with the weighted graph.

Table 6 AUC Measures of the Static Graph and TS Algorithms—Coauthorship Data

		Binary static graph						Weighted static graph							
Year-quarter	No. of target links	CN	AA	PA	GM	SA	KZ	CN	AA	PA	GM	SA	KZ	TS	$T-1$
1995q2	21	0.7241	0.7059	0.6726	0.8562	0.8547	0.8755	0.7242	0.7205	0.7177	0.7789	0.8614	**0.8809**	0.8315	0.6753
1995q3	14	0.6734	0.6552	0.5395	0.8172	0.7859	0.8483	0.6878	0.6874	0.5422	0.7746	0.8051	**0.8611**	0.7966	0.539
1995q4	42	0.7757	0.7831	0.6747	0.7898	0.8659	0.9305	0.7788	0.779	0.7015	0.8041	0.8708	**0.9368**	0.8363	0.6204
1996q1	34	0.8429	0.8232	0.7153	0.7859	0.8342	0.8758	0.8185	0.8418	0.705	0.8271	0.8368	0.8727	**0.8921**	0.8564
1996q2	39	0.8231	0.8427	0.6785	0.8337	0.8156	0.9025	0.8377	0.8188	0.6827	0.82	0.8361	**0.9062**	0.8753	0.7308
1996q3	28	0.6696	0.6873	0.5788	0.7739	0.7637	0.8985	0.6671	0.6839	0.5885	0.75	0.7678	**0.899**	0.8656	0.6722
1996q4	43	0.7706	0.7662	0.6932	0.7797	0.8142	0.8865	0.7585	0.7654	0.6807	0.7487	0.8178	0.8927	**0.9114**	0.8326
1997q1	34	0.7287	0.7396	0.5355	0.7711	0.7325	0.8857	0.7317	0.7425	0.5401	0.786	0.7323	**0.8872**	0.8102	0.6687
1997q2	36	0.8141	0.8204	0.6596	0.8262	0.8423	**0.9459**	0.8144	0.8231	0.6501	0.7895	0.8532	0.9445	0.9091	0.769
1997q3	38	0.8453	0.852	0.7009	0.8567	0.8327	**0.9473**	0.8428	0.8469	0.687	0.8416	0.8497	0.9456	0.9327	0.7112
1997q4	37	0.8198	0.8402	0.6271	0.791	0.7886	0.9311	0.8221	0.8395	0.6397	0.8157	0.7997	**0.9336**	0.8838	0.7403
1998q1	39	0.8287	0.8416	0.6364	0.8124	0.7968	0.9563	0.8337	0.836	0.634	0.84	0.8177	**0.9571**	0.9389	0.7756
1998q2	53	0.8512	0.8623	0.7173	0.8039	0.8386	0.9439	0.8595	0.8616	0.7317	0.8371	0.8641	**0.9461**	0.8857	0.8096
1998q3	27	0.8244	0.8352	0.7734	0.8361	0.8613	0.9371	0.8168	0.8009	0.7746	0.8433	0.8883	**0.9369**	0.8862	0.8094
1998q4	30	0.7639	0.7701	0.611	0.8185	0.772	0.937	0.7549	0.7668	0.6313	0.7699	0.7904	**0.9403**	0.9388	0.6951
1999q1	31	0.7693	0.778	0.6439	0.8126	0.7651	0.916	0.7855	0.7858	0.6704	0.8741	0.7713	**0.9247**	0.8972	0.628
1999q2	37	0.8061	0.8182	0.6278	0.7966	0.7864	0.9284	0.8124	0.8198	0.6282	0.7989	0.8263	**0.93**	0.8523	0.6884
1999q3	23	0.7809	0.7974	0.6326	0.8798	0.8408	0.9782	0.7931	0.8011	0.6541	0.8701	0.8553	0.9824	**0.9876**	0.8051
1999q4	41	0.8018	0.8164	0.644	0.8568	0.801	0.9569	0.8038	0.8152	0.6357	0.857	0.8058	**0.9582**	0.9464	0.7272
2000q1	35	0.8582	0.8691	0.6863	0.859	0.834	0.9575	0.8656	0.8731	0.7002	0.8485	0.8448	**0.9649**	0.8707	0.7751
2000q2	35	0.8277	0.8355	0.6801	0.8067	0.8145	0.9391	0.8296	0.8362	0.6824	0.8282	0.8305	**0.9425**	0.9352	0.8074
2000q3	19	0.796	0.7961	0.6317	0.7915	0.8295	0.9348	0.7976	0.8008	0.679	0.8701	0.8491	**0.9372**	0.8891	0.7209
2000q4	39	0.8353	0.851	0.6763	0.861	0.8195	0.9637	0.8396	0.8484	0.6968	0.8735	0.8571	**0.9679**	0.9439	0.7035
2001q1	24	0.7894	0.8069	0.6931	0.8154	0.8292	0.964	0.8082	0.8085	0.6996	0.8395	0.8516	**0.9677**	0.9427	0.8157
2001q2	28	0.8564	0.865	0.7169	0.8674	0.8441	0.9759	0.858	0.8654	0.7262	0.8806	0.8533	**0.9777**	0.9596	0.7625
2001q3	21	0.8807	0.8849	0.7516	0.8763	0.8814	0.9793	0.8797	0.8844	0.7632	0.896	0.8957	**0.9806**	0.9165	0.7363
2001q4	22	0.7774	0.7823	0.7139	0.864	0.8552	0.9472	0.7805	0.7565	0.705	0.8603	0.8587	0.9496	**0.9588**	0.7343
2002q1	22	0.8485	0.8528	0.7721	0.8278	0.846	0.8922	0.8516	0.854	0.7949	0.872	0.8604	0.9198	**0.9599**	0.9204
2002q2	19	0.8233	0.8234	0.7496	0.8388	0.8619	0.9657	0.8192	0.821	0.7475	0.8573	0.8678	0.9681	**0.9683**	0.811
2002q3	30	0.787	0.7992	0.6348	0.8446	0.8053	0.9095	0.7875	0.7972	0.6242	0.7895	0.8095	0.8941	**0.9348**	0.6689
2002q4	23	0.6768	0.6699	0.5666	0.7429	0.7974	0.9216	0.6757	0.6801	0.5892	0.7791	0.8087	**0.9273**	0.8381	0.766
2003q1	18	0.7247	0.733	0.5651	0.8241	0.7815	0.881	0.7228	0.7249	0.624	0.7774	0.8227	**0.8855**	0.7968	0.7055
2003q2	6	0.748	0.7477	0.5678	0.8543	0.8521	0.9766	0.746	0.7465	0.5635	0.7777	0.8434	0.9794	**0.9906**	0.8204
Average		0.7922	0.7985	0.6596	0.8234	0.8195	0.9300	0.7941	0.7980	0.6694	0.8235	0.8334	**0.9333**	0.9025	0.7425
Average excluding last quarter		0.7936	0.8001	0.6625	0.8224	0.8185	0.9285	0.7956	0.7996	0.6727	0.8250	0.8331	**0.9318**	0.8998	0.7401

Table 7 Performance of Hybrid Link Prediction—Coauthorship Data Set

Year-quarter	No. of target links	TS	CN	CN ∗ TS	AA	AA ∗ TS	PA	PA ∗ TS	GM	GM ∗ TS	SA	SA ∗ TS	KZ	KZ ∗ TS
1995q2	21	0.8315	0.724	0.8316	0.7059	0.8312	0.673	0.8495	0.856	0.85692	0.85469	**0.88195**	0.8755	0.8749
1995q3	14	0.7966	0.673	0.8609	0.6552	0.8629	0.54	0.841	0.817	**0.8815**	0.78586	0.85838	0.8483	0.861
1995q4	42	0.8363	0.776	0.8954	0.7831	0.8841	0.675	0.8911	0.79	0.91174	0.86587	0.90434	0.9305	**0.9325**
1996q1	34	0.8921	0.843	0.8881	0.8232	0.8697	0.715	0.8712	0.786	**0.90835**	0.83423	0.88803	0.8758	0.8796
1996q2	39	0.8753	0.823	0.8986	0.8427	0.8997	0.679	0.866	0.834	0.89452	0.81564	0.89193	0.9025	**0.9039**
1996q3	28	0.8656	0.67	0.8717	0.6873	0.8706	0.579	0.8805	0.774	0.9036	0.76366	0.88545	0.8985	**0.9037**
1996q4	43	**0.9114**	0.771	0.905	0.7662	0.8768	0.693	0.8678	0.78	0.8404	0.81422	0.88858	0.8865	0.8906
1997q1	34	0.8102	0.729	0.885	0.7396	0.8886	0.535	0.8164	0.771	0.87505	0.73249	0.87237	0.8857	**0.8941**
1997q2	36	0.9091	0.814	0.9261	0.8204	0.93	0.66	0.8884	0.826	0.89097	0.84228	**0.95844**	0.9459	0.9499
1997q3	38	0.9327	0.845	0.9584	0.852	**0.9598**	0.701	0.9373	0.857	0.94528	0.83265	0.94442	0.9473	0.9476
1997q4	37	0.8838	0.82	0.9144	0.8402	0.9109	0.627	0.8894	0.791	0.89967	0.78864	0.9245	0.9311	**0.9349**
1998q1	39	0.9389	0.829	0.9521	0.8416	0.9501	0.636	0.9319	0.812	0.94675	0.79678	0.95043	0.9563	**0.9616**
1998q2	53	0.8857	0.851	0.9332	0.8623	0.9276	0.717	0.9056	0.804	0.88915	0.83862	0.9416	0.9439	**0.9461**
1998q3	27	0.8862	0.824	0.9178	0.8352	0.9177	0.773	0.9248	0.836	0.86459	0.86129	**0.95601**	0.9371	0.9415
1998q4	30	0.9388	0.764	0.9376	0.7701	0.9376	0.611	0.927	0.819	0.89917	0.77197	**0.94969**	0.937	0.9455
1999q1	31	0.8972	0.769	0.9213	0.778	**0.9262**	0.644	0.901	0.813	0.91675	0.76509	0.90856	0.916	0.9212
1999q2	37	0.8523	0.806	0.934	0.8182	**0.938**	0.628	0.8558	0.797	0.88529	0.78643	0.9162	0.9284	0.9329
1999q3	23	**0.9876**	0.781	0.9858	0.7974	0.9862	0.633	0.9839	0.88	0.98128	0.84079	0.98749	0.9782	0.9875
1999q4	41	0.9464	0.802	0.9587	0.8164	0.9578	0.644	0.9514	0.857	0.95413	0.80097	0.95594	0.9569	**0.9664**
2000q1	35	0.8707	0.858	0.9546	0.8691	0.956	0.686	0.8977	0.859	0.93939	0.83397	0.94819	0.9575	**0.9647**
2000q2	35	0.9352	0.828	**0.9594**	0.8355	0.9534	0.68	0.9369	0.807	0.91835	0.81449	0.94197	0.9391	0.9465
2000q3	19	0.8891	0.796	0.9232	0.7961	0.916	0.632	0.9081	0.791	**0.95879**	0.82946	0.94489	0.9348	0.9405
2000q4	39	0.9439	0.835	0.9668	0.851	0.9655	0.676	0.9541	0.861	0.94693	0.8195	0.96316	0.9637	**0.9685**
2001q1	24	0.9427	0.789	0.9755	0.8069	0.973	0.693	0.9531	0.815	0.95398	0.82925	**0.97759**	0.964	0.9741
2001q2	28	0.9596	0.856	0.9847	0.865	**0.9855**	0.717	0.9731	0.867	0.97598	0.84406	0.98028	0.9759	0.984
2001q3	21	0.9165	0.881	**0.9851**	0.8849	0.9849	0.752	0.9602	0.876	0.95296	0.88143	0.97281	0.9793	0.9833
2001q4	22	0.9588	0.777	0.9564	0.7823	0.9564	0.714	**0.9639**	0.864	0.95642	0.85519	0.95867	0.9472	0.9557
2002q1	22	0.9599	0.849	0.9555	0.8528	0.9592	0.772	**0.9804**	0.828	0.92936	0.84601	0.93755	0.8922	0.9259
2002q2	19	0.9683	0.823	**0.9834**	0.8234	0.9791	0.75	0.9376	0.839	0.93173	0.86188	0.96119	0.9657	0.9715
2002q3	30	0.9348	0.787	0.9309	0.7992	0.931	0.635	0.9175	0.845	**0.937**	0.80529	0.9108	0.9095	0.9212
2002q4	23	0.8381	0.677	0.8569	0.6699	0.8598	0.567	0.8642	0.743	0.83997	0.79745	0.91482	0.9216	**0.9284**
2003q1	18	0.7968	0.725	0.8517	0.733	0.8542	0.565	0.8164	0.824	0.80373	0.78148	0.88426	0.881	**0.8911**
2003q2	6	**0.9906**	0.748	0.9858	0.7477	0.9849	0.568	0.9856	0.854	0.98633	0.85214	0.99003	0.9766	0.9904
Average		0.9025	0.7922	0.9287	0.7985	0.9268	0.6596	0.9100	0.8234	0.9144	0.8195	0.9318	0.9300	**0.9370**
Average excluding last quarter		0.8998	0.7936	0.9269	0.8001	0.9250	0.6625	0.9076	0.8224	0.9122	0.8185	0.9300	0.9285	**0.9353**
Improvement (%)				16.80		15.61		36.99		10.91		13.63		0.73

Table 7 shows the performance of the hybrid link prediction using score products. The results are similar to the e-mail data set results. The KZ ∗ TS algorithm had generally the best performance with an average AUC measure of 0.9353 but did not consistently outperform other hybrid algorithms across the 32 quarters. In fact, all six hybrid algorithms achieved the best performance in at least two quarters. By incorporating the time-series prediction, the six static graph link prediction algorithms had improvements ranging from 0.73% (the KZ algorithm) to 36.99% (the PA algorithm). All improvements are significant according to the paired two-sample t-test with p-values smaller than 0.0001.

7. Conclusions and Future Directions

The time-series link prediction problem is formally introduced in this paper. Our formulation extends from existing link prediction studies to explicitly modeling the time-series patterns of the link occurrence frequencies. Such a formulation can lead to improved link prediction quality in application domains where the repeated link occurrences are of central interest, such as communication surveillance. We proposed a univariate time-series model to estimate ARIMA models for individual links as a minimum model that can capture the time-series frequency pattern of link occurrences. This model produces predictions on future occurrences of links based on the intralink dependencies over time. The existing link prediction algorithms based on a static graph representation can also be used for a time-series link prediction problem by reducing the weighted graph series to a single static graph. The static link prediction approach mainly relies on interlink dependency patterns (such as paths and

clusters) to perform link prediction. As the time-series and static graph approaches employ conceptually orthogonal data patterns, we proposed hybrid time-series link prediction combining the predictions from both approaches for improved link prediction performance.

We experimented with our proposed univariate TS and hybrid algorithms on the monthly e-mail graphs based on the Enron data set and quarterly coauthorship graphs based on the high-energy theoretical physics literature. Our experimental results showed that the univariate time-series model, despite being the minimum model capturing the time-series information, achieved comparable performance with the best-performing ones of six commonly used static graph link prediction algorithms: CN, AA, PA, KZ, SA, and GM algorithms. Consistent with our analysis, the time-series and the static graph predictions were shown to have only weak correlations, indicating the possibility of prediction performance improvement with hybrid approaches. The experimental results showed that the hybrid TS algorithm achieved significantly improved prediction performance than the time-series approach or static graph approach alone. Combining the time-series prediction model with each of the six static graph algorithms in our study had significantly better average performances. The combination of the KZ and TS algorithms had the overall best performance for both data sets. However, there was no consistent winner among the hybrid algorithm across all the time periods we evaluated.

Our study represents an initial effort toward building an integrated time-series link prediction algorithm that can exploit the interlink dependencies (network/graph structural patterns such as paths and clusters) and intralink time dependencies simultaneously for improved link prediction performance in application domains such as communication surveillance. Our experiments of the univariate time-series model presented strong evidence for the potential of the predictive value of the time-series link occurrence patterns. In future research, we will explore multivariate time-series models that can incorporate more complex covariance structures that have been well studied in the time-series analysis literature. More importantly, we will explore covariance structures motivated by the commonly used network/graph-based interlink dependency structures to build truly integrated hybrid time-series link prediction algorithms.

Acknowledgments

This work was sponsored in part by a research grant from the Smeal College of Business of the Pennsylvania State University. The first author acknowledges support from NNSFC 60621001 and 60573078; MOST 2006CB705500, 2004CB318103, and 2006AA010106; and CAS 2F05N01, and 2F07C01. The authors also thank the editors and three anonymous reviewers for their detailed and constructive comments.

References

Adamic, L., E. Adar. 2003. Friends and neighbors on the Web. *Soc. Networks* **25**(3) 211–230.

Akaike, H. 1974. A new look at the statistical model identification. *IEEE Trans. Automatic Control* **19**(6) 716–723.

Alwan, L. C., H. V. Roberts. 1988. Time-series modeling for statistical process control. *J. Bus. Econom. Statist.* **6**(1) 87–95.

Barabasi, A.-L., R. Albert. 1999. Emergence of scaling in random networks. *Science* **286**(5439) 509–512.

Berger-Wolf, T. Y., J. Saia. 2006. A framework for analysis of dynamic social networks. *Proc. 12th ACM SIGKDD Internat. Conf. Knowledge Discovery and Data Mining, Philadelphia*, ACM, New York, 523–528.

Bolot, J.-C., P. Hoschka. 1996. Performance engineering of the World Wide Web: Application to dimensioning and cache design. *Comput. Networks ISDN Systems* **28**(7–11) 1397–1405.

Box, G. E. P., G. Jenkins. 1970. *Time Series Analysis, Forecasting, and Control*. Holden-Day, Oakland, CA.

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

Dempster, A., N. Laird, D. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *J. Roy. Statist. Soc. Ser. B* **39**(1) 1–38.

Domingos, P., M. Richardson. 2001. Mining the network value of customers. *Proc. Seventh ACM SIGKDD Internat. Conf. Knowledge Discovery and Data Mining, San Francisco*, ACM, New York, 57–66.

Dzeroski, S., N. Lavrac. 2001. *Relational Data Mining*. Springer-Verlag, Berlin.

Getoor, L. 2003. Link mining: A new data mining challenge. *SIGKDD Explorations* **5**(1) 84–89.

Getoor, L., M. Sahami. 1999. Using probabilistic relational models for collaborative filtering. *Proc. WebKDD'99, San Diego.*

Getoor, L., N. Friedman, D. Koller, B. Taskar. 2002. Learning probabilistic models of link structure. *J. Machine Learn. Res.* **3** 679–707.

Gilbert, P. D. 1995. Combining VAR estimation and state space model reduction for simple good predictions. *J. Forecasting* **14** 229–250.

Goldberg, D. S., F. P. Roth. 2003. Assessing experimentally derived interactions in a small world. *Proc. National Acad. Sci.* **100**(8) 4372–4376.

Heider, F. 1946. Attitudes and cognitive organization. *J. Psych.* **21** 107–112.

Hofmann, T. 1999. Probabilistic latent semantic analysis. K. Laskey, H. Prade, eds. *Proc. 15th Conf. Uncertainty Artificial Intelligence, Stockholm*, Morgan Kaufmann, San Fransisco, 289–296.

Hofmann, T. 2004. Latent semantic models for collaborative filtering. *ACM Trans. Inform. Systems* **22**(1) 89–115.

Holme, P., S. M. Park, B. J. Kim, C. R. Edling. 2007. Korean university life in a network perspective: Dynamics of a large affiliation network. *Physica A* **373** 821–830.

Huang, Z., H. Chen, D. Zeng. 2004a. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Trans. Inform. Systems* **22**(1) 116–142.

Huang, Z., D. Zeng, H. Chen. 2004b. A link analysis approach to recommendation with sparse data. *Proc. Americas Conf. Inform. Systems, New York*, Association of Information Systems, Atlanta, 1997–2005.

Huang, Z., W. Chung, T.-H. Ong, H. Chen. 2002. A graph-based recommender system for digital library. *Proc. Second ACM/IEEE-CS Joint Conf. Digital Libraries, Portland, OR*, ACM, New York, Association of Information Systems, Atlanta, 65–73.

Katz, L. 1953. A new status index derived from sociometric analysis. *Psychometrika* **18**(1) 39–43.

Leskovec, J., J. Kleinberg, C. Faloutsos. 2007. Graph evolution: Densification and shrinking diameters. *ACM Trans. Knowledge Discovery Data* **1**(1) 2.

Liben-Nowell, D., J. Kleinberg. 2003. The link prediction problem for social networks. *Proc. 12th Internat. Conf. Inform. Knowledge Management* (*CIKM*), *New Orleans*, ACM, New York, 556–559.

O'Madadhain, J., J. Hutchins, P. Smyth. 2005. Prediction and ranking algorithms for event-based network data. *ACM SIGKDD Explorations* **7**(2) 23–30.

Popescul, A., L. Ungar. 2003. Statistical relational learning for link prediction. *Proc. Workshop Learn. Statist. Models Relational Data Internat. Joint Conf. Artificial Intelligence, Acapulco, Mexico*, ACM, New York, 81–90.

Porter-Hudak, S. 1990. An application of the seasonal fractionally differenced model to the monetary aggregates. *J. Amer. Statist. Assoc.* **85**(410) 338–344.

Potgieter, A., K. A. April, R. J. E. Cooke, I. O. Osunmakinde. 2009. Temporality in link prediction: Understanding social complexity. *Emergence Complexity Organ.* Forthcoming.

Rattigan, M., D. Jensen. 2005. The case for anomalous link detection. *Proc. 4th Multi-Relational Data Mining Workshop, 11th ACM SIGKDD Internat. Conf. Knowledge Discovery and Data Mining, Chicago*, ACM, New York, 69–74.

Resnick, P., H. Varian. 1997. Recommender systems. *Comm. ACM* **40**(3) 56–58.

Resnick, P., N. Iacovou, M. Suchak, P. Bergstorm, J. Riedl. 1994. GroupLens: An open architecture for collaborative filtering of netnews. *Proc. ACM Conf. Comput.-Supported Cooperative Work*, ACM, New York, 175–186.

Salton, G. 1989. *Automatic Text Processing*: *The Transformation, Analysis, and Retrieval of Information by Computer*. Addison Wesley, Reading, MA.

Sarkar, P., A. W. Moore. 2005. Dynamic social network analysis using latent space models. *ACM SIGKDD Explorations* **7**(2) 31–40.

Sarwar, B., G. Karypis, J. Konstan, J. Riedl. 2000. Application of dimensionality reduction in recommender systems: A case study. *Proc. WebKDD Workshop at the ACM SIGKKD, Boston*, ACM, New York.

Shetty, J., J. Adibi. 2005. The Enron data set database schema and brief statistical report, http://www.isi.edu/~adibi/Enron/Enron_Dataset_Report.pdf.

Shumway, R., D. S. Stofer. 2000. *Time Series Analysis and Its Applications*. Springer-Verlag, New York.

Ungar, L. H., D. P. Foster. 1998. A formal statistical approach to collaborative filtering. *Proc. Conf. Automated Learn. Discovery* (*CONALD*), *Pittsburgh*.

Vazquez, A., J. G. Oliveira, A.-L. Barabasi. 2005. The inhomogeneous evolution of subgraphs and cycles in complex networks. *Phys. Rev. Lett. E* **71** 025103.

Winkler, W. E. 1994. Advanced methods for record linkage. Technical report, Statistical Research Division, U.S. Census Bureau, Wasington, D.C.

Yu, K., W. Chu, S. Yu, V. Tresp, Z. Xu. 2006. Stochastic relational models for discriminative link prediction. *Adv. Neural Inform. Processing Systems* **19** 1553–1560.