A Decoupled Exponential Random Graph Model for Prediction of Structure and Attributes in Temporal Social Networks

Vladimir Ouzienko, Yuhong Guo, Zoran Obradovic*

Center for Data Analytics and Biomedical Informatics, Temple University, Philadelphia, PA 19122, USA

Received 24 July 2010; revised 24 March 2011; accepted 2 June 2011
DOI:10.1002/sam.10130
Published online in Wiley Online Library (wileyonlinelibrary.com).

Abstract: The analysis of social networks often assumes time invariant scenario, while in practice actor attributes and links in such networks often evolve over time and are inextricably dependent on each other. In this article, we propose a new method to predict actor attributes and links in temporal networks. Our approach takes into account the attributes corresponding to the participating actors together with topological and structural changes of the network over time. This is achieved by building two conditional predictors to jointly infer links and actor attributes. The proposed prediction method was significantly more accurate than alternatives when evaluated on synthetic data sets and two well-studied real-life temporal social networks. In addition, the new algorithm is computationally more efficient than a related alternative scaling up linearly with the number of temporal observations and quadratically with the number of actors considered. © 2011 Wiley Periodicals, Inc. Statistical Analysis and Data Mining 4: 470–486, 2011

Keywords: social network analysis; exponential random graph model; decoupled model; temporal social networks; predictor

1. INTRODUCTION

A social network describes the interactions (relationships) between participating social entities (actors). The well-known web applications, Facebook and MySpace, are examples of social networks, where each social actor is a person and two persons can be linked together if there are interactions between them (e.g., email exchanges). Another example of social networks is the informal relationship graph of farming estates [1], where each social actor is a family that owns the farm. In this farming community network, a social visit constitutes a link between two social entities. Actors and links are two essential elements of social networks regardless of their semantics, while directed graphs are the main representation and analytical tool of social network analysis (SNA). A social network can be either static or dynamic. To analyze temporal dynamic social networks, one needs to investigate the evolving patterns of the networks along the time axis, including the network trends, the changes of actor's roles, and the strengthening or weakening of the relationships. The typical prediction problems associated with temporal social networks are inferences of links or actor roles, determining the strengths of the relationships and imputation of the missing links.

While temporal SNA is a well-studied subject [2,3], the topic of prediction in such structures is less explored. Many published works in the field such as Ref. [4] investigate the social networks from the perspective of statistical inference. Their objective is to infer the most probable statistics which would explain the network evolution process. For example, Ref. [4] also addresses the network evolution and changes of the actors attributes, but not to the forecasting end. The objective of our work is to create a state-of-the-art model capable of predicting how the social network will look in the future.

Most published works on predictions of temporal social networks are focused on link predictions. In particular, the time a specified group of social interactions will occur is predicted in a recent study [5]. Another recently studied temporal prediction problem is inferencing network structures at the next yet unseen step [6,7]. The social network link imputation problem is also considered for data stored in relational databases [8]. These models predict how the network evolves. However, the attribute values of the participating actors are typically not considered (an exception is the method proposed by Popescul *et al*. [8]).

Correspondence to: Zoran Obradovic
(zoran.obradovic@temple.edu)

The prediction models employed in previous studies of temporal social networks are solely based on network structures and their topologies, albeit the temporal aspects of the social networks are always considered. However, in many cases, attribute values of actors are also available in social networks and are useful for accurate link prediction. Moreover, predicting the attribute values of participating actors could be as important as predicting the network graphs. However, to the best of our knowledge, none of the published work thus far attempts to predict both the links and the attribute values in a single concise model.

In this article, we develop a novel model that facilitates simultaneous predictions of the links and the attribute values in temporal social networks based solely on historical data.

Given the evolving structure of a temporal network and the changing nonstatic attribute values of the network actors, the goal is to predict the network structure and actor values at the next unobserved time step. Figure 1 shows a graphical representation of the prediction task. This figure demonstrates how the relationship graph of the invariant set of actors and actors' attributes is changing as time progresses from time steps $T-2$ to T. Then based on this historical data, we want to predict the relationship graph and attribute value of each actor at unobserved time step $T+1$. Instead of training a single joint probability prediction model for this prediction task, we build two conditional exponential random graph models, based on the observation over Gibbs sampling inference. These two conditional predictors are mutually dependent on each other, and can then be used to predict the network structures, that is, the links, and the attribute values in an alternative way. Our empirical study suggests that this novel approach achieves better performance than baseline approaches.

Preliminaries of social networks are introduced in Section 2 of this article. Section 3 presents a brief review of related models. In Section 4, a novel approach is proposed that facilitates simultaneous predictions of the links and the attribute values in temporal social networks based solely on historical data. This is followed by a summary of experimental evaluations on synthetic and real-life problems presented in Section 5. Finally, the discussion and some directions for future work are contained in Section 6.

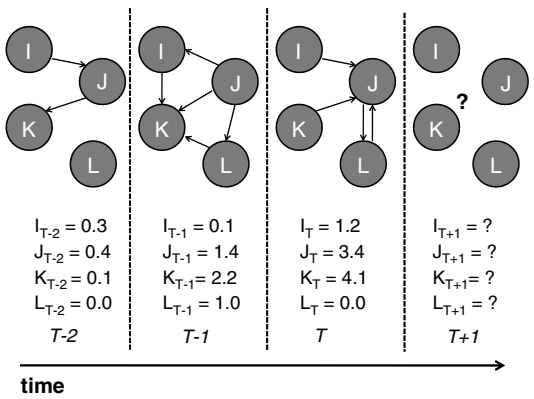

Fig. 1 A conceptual representation of the prediction task addressed in this article.

2. PRELIMINARIES

A typical representation model for analysis of temporal social networks consists of a series of dichotomous adjacency matrices (also called sociomatrices) which define the states of a set of participating agents at each given observation time [9]. For example, to represent T temporal observations of k actors we will use T adjacency matrices, $\mathbf{N}^1 \cdots \mathbf{N}^T$, where each entry $N_{ij}^t = 1$ indicates the presence of a link between the actor i and actor j at time step t; conversely $N_{ij}^t = 0$ indicates the absence of such a link. For example, in a social network over the friendships of a class of students, a sociomatrix $\mathbf{N}^t$ is used to indicate the links between the students based on their friendship status at time t. The entry $N_{ij}^t = 1$ indicates that student i considered student j as his/her friend at time t. It is important to note that a relationship matrix does not have to be symmetric. In this example, we can have $N_{ij}^t = 1$ and $N_{ji}^t = 0$ at the same time, which means that while student i considered student j as his/her friend, student j did not reciprocate the feelings. Moreover, the actors should not have self-referenced relations, thus diagonals of matrices $\mathbf{N}^1 \cdots \mathbf{N}^T$ should be always populated with zeros.

In the past, the $p*$ class of statistical models [10], also known as Exponential Random Graph Models (ERGMs) [9,11,12] has been successfully applied to analyze and describe the sociomatrices discussed above. The ERGM is a log-linear model and is expressed as:

$$P(N) = \frac{1}{Z(\boldsymbol{\theta})} \exp\{\boldsymbol{\theta}' \mathbf{u}(N)\}, \tag{1}$$

which defines the probability of a given social network N from the set $\mathcal{N}$ of all possible social networks. Here $\boldsymbol{\theta}'$ is a transposed parameter vector, $\boldsymbol{u}(N)$ is a vector of sufficient statistics of the network N, and $Z(\boldsymbol{\theta})$ is the normalization constant. An extended version of ERGM, the Temporal Exponential Random Graphical Model (tERGM), is proposed in Ref. [13], which specifically deals with the temporal aspect of social network analysis. This temporal model takes the Markovian assumption that each network matrix $\mathbf{N}^t$, that is, the observation matrix at the time t, is conditionally independent of all other prior observations $\mathbf{N}^1 \cdots \mathbf{N}^{t-2}$ given its immediate prior observed matrix $\mathbf{N}^{t-1}$, which is:

$$P\left(\mathbf{N}^t | \mathbf{N}^{t-1}, \mathbf{N}^{t-2} \cdots \mathbf{N}^1\right) = P\left(\mathbf{N}^t | \mathbf{N}^{t-1}\right). \tag{2}$$

Thus the joint distribution in tERGM can be expressed as:

$$P\left(\mathbf{N}^{1:T}|\boldsymbol{\theta}\right) = P\left(\mathbf{N}^{1}\right)\prod_{t=2}^{T} P\left(\mathbf{N}^{t}|\mathbf{N}^{t-1},\boldsymbol{\theta}\right), \tag{3}$$

where the conditional transition distribution is an extension of the log-linear model in Eq. (1), and can be expressed as

$$P\left(\mathbf{N}^{t}|\mathbf{N}^{t-1},\boldsymbol{\theta}\right) = \frac{1}{Z\left(\boldsymbol{\theta},\mathbf{N}^{t-1}\right)}\exp\left\{\boldsymbol{\theta}'\psi\left(\mathbf{N}^{t},\mathbf{N}^{t-1}\right)\right\}. \tag{4}$$

Here, ψ is a function of $\mathbb{R}_{k\times k}\times\mathbb{R}_{k\times k}\rightarrow\mathbb{R}^{l}$, where k is number of actors, which defines the statistics, that is, the features. In Ref. [13], four statistics are defined, including *density*, *stability*, *reciprocity*, and *transitivity*:

$$\psi_{\mathrm{D}}\left(\mathbf{N}^{t},\mathbf{N}^{t-1}\right) = \frac{1}{k-1}\sum_{ij}^{k} N_{ij}^{t}, \tag{5}$$

$$\psi_{\mathrm{S}}\left(\mathbf{N}^{t},\mathbf{N}^{t-1}\right) = \frac{1}{k-1}\sum_{ij}^{k} \times\left[N_{ij}^{t}N_{ij}^{t-1} + \left(1-N_{ij}^{t}\right)\left(1-N_{ij}^{t-1}\right)\right], \tag{6}$$

$$\psi_{\mathrm{R}}\left(\mathbf{N}^{t},\mathbf{N}^{t-1}\right) = k\frac{\left[\sum_{ij}^{k} N_{ji}^{t}N_{ij}^{t-1}\right]}{\left[\sum_{ij}^{k} N_{ij}^{t-1}\right]}, \tag{7}$$

$$\psi_{\mathrm{T}}\left(\mathbf{N}^{t},\mathbf{N}^{t-1}\right) = k\frac{\sum_{pqr}^{k} N_{pr}^{t}N_{pq}^{t-1}N_{qr}^{t-1}}{\sum_{pqr}^{k} N_{pq}^{t-1}N_{qr}^{t-1}}, \tag{8}$$

where k is number of actors (all statistics are scaled to be in $[0;k]$ range). The conditional probability function in Eq. (4) is a function of the parameters, $\boldsymbol{\theta}=\{\theta_{\mathrm{D}},\theta_{\mathrm{S}},\theta_{\mathrm{R}},\theta_{\mathrm{T}}\}$, which correspond to the statistics $\{\psi_{\mathrm{D}},\psi_{\mathrm{S}},\psi_{\mathrm{R}},\psi_{\mathrm{T}}\}$. Thus θ_{D} controls the density of the network, that is, the number of existing links. θ_{S} controls the stability, or whether a link (or its absence) at time step $t-1$ continues to exist at time step t. θ_{R} drives the reciprocity, which is the degree that a link presented at time $t-1$ from actor i to j will result in a reciprocal link from j to i at time t. θ_{T} governs the transitivity, which is the propensity of the links from p to q and from q to r at $t-1$ resulting in a transitive link from r to p at time t.

Parameters of the tERGM are estimated from the sequence of temporal network observations $\mathbf{N}^{1}\cdots\mathbf{N}^{T}$ using the Markov Chain Monte Carlo (MCMC) techniques [12,13]. After the model parameters were learned, it is possible to make prediction over the network structure at future time step $T+1$ by applying Gibbs sampling method [12], which samples networks according to the conditional model given in Eq. (4) and the previous observation $\mathbf{N}^{T}$.

The tERGM model introduced above considers only the structures and topologies of the temporal networks, while actor attributes are ignored. However, when the actor attributes are available and useful to know, one needs to make predictions over both the network structures and the attribute values in future steps. For instance, in the student friendship network example, the Grade Point Average (GPA) of each student can be easily obtained at each observation step. Thus, in addition to an array of sociomatrices, one can have T numbers of $k\times 1$ attribute vectors $\boldsymbol{x}^{1:T}$ where $\boldsymbol{x}^{i}$ contains the GPA for all the students at time step i. The GPAs could be a factor that affects the friendships among students. Thus, it is reasonable to assume that the network structures and the attribute values are interactively dependent. Our proposed work is based on such a dependence assumption over the links and the attributes. Our method extends tERGM by building two mutually dependent conditional models which are used to jointly predict both $\mathbf{N}^{T+1}$ and $\boldsymbol{x}^{T+1}$.

3. RELATED WORK

Temporal networks research is mostly applied to the domains of genetics and social network analysis. In genetics, a predominant question addressed by the research community is construction of the genes' pathways (networks) based on temporal observation of gene expressions. A number of interesting models have been developed in this field [14,15]. The question answered by these models is fundamentally different from our research subject. Genetic temporal models recover the network structure, based on underlying time series data. Our temporal model predicts future network structure and actor attributes based on time series of networks and actor attributes. Despite such a fundamental difference, network recovery models consider both links and data, and therefore we pay close attention to these models.

Contrary to genetics, the main question of the temporal SNA community is link prediction. There are two camps in this field. An overwhelming majority of publications on this topic predicts the network structure solely based on network topology [5–7]. The other camp also predicts links, but considers the actor attributes. Reported results that pursue such an approach were mostly concerned with specific types of data sets such as academic collaboration or business cooperation networks [16]. These data sets are textually rich, and therefore specialized models were applied to them. To the best of our knowledge, the SNA community has not yet considered simultaneously predicting data and links for temporal networks.

The brief review of the previous work in this section covers two groups of models. The first group of models

is 'related models' that cannot be readily applied to the problem we are trying to solve, but at the same time is important to us because we drew upon it when we develop our approach. The second type, 'baseline models', is link prediction models that can be applied in a temporal setting. Most of these baseline models do not predict actor attributes. We use them as baselines in our link predictions experiments on temporal social networks.

3.1. Related Models

A novel Hidden Temporal Exponential Random Graph Model (htERGM) was introduced [17] to recover latent temporal network structures based on attributes of observed nodes (actors). Specifically, it is shown how to recover the temporal network of the *Drosophila* gene expressions from the observations over the gene expression levels (node attributes). This approach incorporates the learning of both network structures and node (gene) attributes in a single, combined htERGM expressed as

$$P\left(\mathbf{N}^{1:T}, \boldsymbol{x}^{1:T} \middle| , \mathbf{N}^0\right) = \prod_{t=1}^{T} P\left(\mathbf{N}^t \middle| \mathbf{N}^{t-1}, \boldsymbol{\theta}\right) P\left(\boldsymbol{x}^t \middle| \mathbf{N}^t, \Lambda\right). \quad (9)$$

Note that Eq. (9) consists of two conditional models. The first part is the transition model given in Eq. (4), which defines how the gene network evolves over time. The second model, shown in Eq. (10):

$$P\left(\boldsymbol{x}^t \middle| \mathbf{N}^t, \Lambda\right) = \frac{1}{Z\left(\eta, \mathbf{N}^t\right)} \exp\left\{\eta \sum_{ij} \Phi\left(x_i^t, x_j^t, N_{ij}^t, \Lambda_{ij}\right)\right\} \quad (10)$$

is called an 'emission model', and it defines the dependency of the node attributes over the underlying network topology. The Λ in Eq. (10) is a time invariant activation function specific to Drosophila data set. This function provides the degree of mutual activation or suppression or activation between two genes and has $[-1, 1]$ range. Presence of the activation function Λ is a rather strong assumption, which might be perfectly valid for the gene expression network, but perhaps is inappropriate for the temporal social networks. As two models, transition and emission, are involved so that one needs to learn two sets of unknown parameters: $\boldsymbol{\theta}$ and η, while the activation function Λ is directly estimated from the training data set. Parameters $\boldsymbol{\theta}$ and η are treated as latent variables in htERGM and an Expectation-Maximization (EM) method is used for training. This approach is computationally slow so applications were limited to the retrieval of subnetworks of up to ten genes. The new approach that we propose in this article learns two sets of model parameters separately which makes the learning process much faster and applicable for larger networks.

A follow up work to Ref. [17] was presented by Ahmed and Xing [14], who introduced an algorithm named TESLA to address the slowness of the htERGM. In this new approach, the latent model is simplified to a convex temporal smoothed l_1-regularized logistic regression problem

$$P\left(\mathbf{x}^t \middle| \boldsymbol{\theta}^t\right) = \exp\left(\sum_{i \in V} \theta_{ii}^t x_i^t + \sum_{(i,j) \in E^t} \theta_{ij}^t x_i^t x_j^t - A\left(\boldsymbol{\theta}^t\right)\right), \quad (11)$$

where $A(\boldsymbol{\theta}^t)$ is the log partition function, V is the set of nodes, and E^t is a set of edges at time t. TESLA first learns parameters $\boldsymbol{\theta}$ for each node for all available time steps $t = 1 \cdots T$. Given $\boldsymbol{\theta}$, a simple sign function is applied to determine whether a link is present between nodes i and j at time t. However, in TESLA the transition model is omitted. It can only be applied to recover unknown structures of the temporal network based on the observed node values $\boldsymbol{x}^t$ for $t = 1 \cdots T$. Direct predictions of the future values for $\boldsymbol{x}^{t+1}$ and the edge set E^{t+1} are impossible in this model.

Another approach [4] models the network dynamics as a stochastic actor-driven process where each change occurs at one microstep at a time. This approach models the changes of actor links as a function of node covariates (or 'actor covariates') and characteristics of pairs of nodes ('dyadic covariates'). It takes the Markovian assumption of the network evolution process and posits that changes in the links are actor driven. Its main characteristic is modeling of the network events (appearance or disappearance of a link) as occurring at infinitesimal time intervals where only one actor gets the opportunity to change one link at a time. The modeling process works as follows, the actor is chosen for a given micro time-step who gets the opportunity to change one of his/her outgoing links. The choice of actor can be made either with equal probability among all actors or with probability corresponding to the actor's standing in the network. After the actor is chosen he/she gets the opportunity to change one outgoing link with probability proportional to the objective function. The objective function consists of the network, actor and dyadic covariates effects. To simulate the network evolution the process is repeated until the convergence of the estimation parameters. The stochastic actor-based model is used by researchers to study which of the network effects, or covariates, are prominent factors in evolution of the network. In our work we are more concerned with the predictive power of our proposed approach and other methods. The actor-driven stochastic model has some similarities with our methodology such as the Markovian

assumption of the network evolution and utilizing log-linear functions to model probabilities. The main difference between our model and that in Ref. [4], is that the model in Ref. [4] looks at a longitudinal process and models latent link changes in continuous time as taking place between the actual observations, whereas our model only considers the discrete time steps when the network was observed.

The previous work [18] compares discrete versus continuous time approach. It asserts that discrete observations do not capture the full dynamic of the network evolution because there is a possibility that there could have been many unobserved link changes in-between the network surveys. In Ref. [18] an example is cited where the network surveys done at times t and $t+1$ recorded the creation of a link between two actors with similar characteristic (both nondrinkers in this example). Appearance of such a link in any discrete model is unequivocally classified as a homophily selection. The possibility here is that we did not observe an interim process between times t and $t+1$ where one actor had become a drinker, which caused the other actor to start a therapeutic relationship with him/her (thus creating the outgoing link). This relationship had influenced the actor-drinker to stop consuming alcohol, so at the time $t+1$ we finally observe two nondrinkers in a social relationship. The fact that homophilic selection had nothing to do with what had actually transpired went completely unnoticed. The time-continuous approach is a plausible modeling solution for longitudinal networks, but missing element here is that it cannot be verified with real-life data sets available today. The only way to validate the model assertion that there was an unobserved activity between the surveys would be to retroactively ask network participants a follow up questions. Unfortunately, such retrospective studies are very rare and prone to error [19]. Nevertheless, the question of whether a discrete model can deal with unobserved changes has to be addressed. One way to address it is to determine whether the studied temporal network has too many link changes between observations which would suggest that it is unsuitable for treatment by a discrete model. A social network which evolves too fast between the survey times would not be a good candidate because of the increased likelihood that self-canceling link creations/deletions were not recorded. In Ref. [4] it was proposed to use the Jaccard index to measure the network change rate between the surveys. This measure calculates the rate of change between two observations by

$$\frac{C_{11}}{C_{11}+C_{01}+C_{10}}, \tag{12}$$

where C_{11} is count of links present in both network, C_{01} is number of newly created ties, and C_{10} is number of ties which were terminated. A low value of this index, less than 0.2, suggests that network is undergoing rapid change and it is likely that surveys missed a lot of activity. An index value close to 1.0 tells us that not much has happened between two observations. In our experiments we used two synthetic and two real-life data sets, which are described in great detail in Section 5. We calculated the Jaccard index for all our data sets to ensure that networks are not changing too fast and are suitable for our model. The rate change for synthetic *Data set1* and *Data set2* averaged 0.39 and 0.30, respectively with a small variation of the index along the time axis. For the real-life data sets *Delinquency* and *Teenagers* the average values were 0.40 and 0.35, also with very little variations. These values suggest that changes in temporal networks used in our experiments are adequately gradual and therefore our data sets are good candidates for treatment by a discrete model such as ours.

3.2. Baseline Models

A comprehensive description of a static graph link prediction algorithm which can also be applied to temporal network link prediction problem was recently published [20]. The article [20] introduced a new approach which enhances the accuracy of temporal network link prediction by combining a time-series approach with time-invariant algorithms. We provide brief descriptions of these methods because we will use them as baselines in our experiments on real-life and synthetic data sets. The output of these algorithms is always score matrix $\mathbf{S}$, where each entry $\mathbf{S}(i, j)$ assigns the link occurrence score proportional to the predicted probability of the link from actor i to actor j at time step $T+1$. Contrary to our approach, these methods do not exploit information available as attributes of actors.

Finally, here we also describe an alternative approach based on tensor factorization which predicts both links and attributes (Section 3.2.4).

3.2.1. Time invariant models

In order to predict links at $T+1$, the time invariant link prediction algorithms reduce series of sociomatrices $\mathbf{N}^1 \cdots \mathbf{N}^T$ to a time invariant adjacency matrix $\mathbf{M}^{1:T}$, where $\mathbf{M}^{1:T}(i, j) = \sum_{t=1}^{T} \mathbf{N}^t(i, j)$. The matrix $\mathbf{M}^{1:T}$ is further reduced to the binary matrix $\mathbf{M}^*$ where $\mathbf{M}^*(i, j) = 1$ if $\mathbf{M}^{1:T}(i, j) > 1$, and 0 otherwise. The time invariant link prediction algorithms usually use such matrix representation.

The *Common Neighbor* algorithm assigns the probabilistic score for each entry in the score matrix $\mathbf{S}$ as the number of common neighbors shared by each possible pair of actors. In a matrix form: $\mathbf{S} = \mathbf{M}^* * \mathbf{M}^*$. *Common Neighbor* exploits the notion of transitivity; the higher count of common neighbors shared by two actors, the more likely these actors will be connected. The concept of transitivity

as applied to social relationships was thoroughly studied in Ref. [21].

The *Adamic-Adar* method [22] is a measure similar to *Common Neighbor*. Adapted for the link prediction problem, *Adamic-Adar* assigns the link score between actors i and j as

$$\mathbf{S}(i, j) = \sum_{k \in V, (i,k) \in E(\mathbf{M}^*), (j,k) \in E(\mathbf{M}^*)} \frac{1}{\log(d(k))}, \quad (13)$$

where d is degree of the actor. The *Adamic-Adar* measure is well known in information retrieval domain.

The *Katz* measure [23] is based on a principle similar to *Common Neighbor* and *Adamic-Adar*. However, it also considers paths beyond length 2 between pairs of actors. The *Katz* sums paths of all possible lengths between two actors and exponentially dampens the longer paths to give them less weight than to the shorter paths. For the purpose of links prediction

$$\mathbf{S}(i, j) = \sum_{l=1}^{\infty} \beta^l ||\text{paths}_{i,j}^{<l>}||, \quad (14)$$

where l is the length of the paths and β is a dampening parameter which has to be estimated from the training data. In Ref. [6] it was shown that a score matrix based on *Katz* measure can be derived as:

$$\mathbf{S} = (I - \beta \mathbf{M}^*)^{-1} - I. \quad (15)$$

The *Preferential Attachment* method sets the link score in matrix $\mathbf{S}$ as the product of the degrees of participating actors, namely $\mathbf{S}(i, j) = d(i) * d(j)$, where function d is the actor's degree. The *Preferential Attachment* algorithm assumes that the highly connected actors are more likely to be linked. This assumption is based on a preferential attachment phenomenon discovered in real-world networks [24].

3.2.2. Time series model

An application of the *autoregressive integrated moving average* ARIMA model [25] is also proposed for the link prediction in temporal social networks [20]. Each possible link (i, j) between actors i and j during observations $t = 1 \cdots T$ can be viewed as a time series of the link occurrence frequency. The ARIMA model then can be fitted by exploring its parameter space $p = 0, 1, 2, 3; d = 0, 1; q = 0, 1, 2, 3$, where p is the number of autoregressive terms, d is the number of nonseasonal differences and q is the number of lagged forecast errors in the prediction equation. To determine the quality of the model the *Akaike information criterion* AIC measure is used [20,26]. The model with the lowest AIC score is selected to predict the link frequency at time $T + 1$. This prediction is defined as $\hat{N}_{ij}^{T+1}$, and the prediction error as $sd(\hat{N}_{ij}^{T+1})$. The link occurrence score of matrix $\mathbf{S}$ is populated with probabilities of the link frequency at $T + 1$ to be greater than 1: $\mathbf{S}(i, j) = \Pr(\hat{N}_{ij}^{T+1} > 1)$. The time series model [20] is simple in the sense that it does not consider the temporal networks' topology. It only looks into a link's occurrence frequency independently from other actors. Therefore, its concern is the temporal nature of the link occurrence, whereas models described in the previous section were only considering the network's topology and did not consider the temporal aspect. Thus, the time-series model and models described in Section 3.2.1 are 'orthogonal'.

3.2.3. Combined model

The same study [20] exploits orthogonality of time invariant link prediction models and a time-series model by combining score matrices produced by both to yield a more accurate predictor. Authors introduced the *Hybrid Time Series Link Prediction Algorithm* which combines the score matrix generated by any of the time invariant algorithms (*Common Neighbor, Preferential Attachment, Adamic-Adar*, and *Katz*) and time-series algorithm introduced in prior section. Namely, the new score matrix is derived as

$$\mathbf{S}(i, j) = \left(\mathbf{S}_{\mathrm{S}}(i, j) + \frac{ms}{\alpha}\right) * \left(\mathbf{S}_{\mathrm{T}}(i, j) + \frac{mt}{\alpha}\right), \quad (16)$$

where α is a parameter greater than 1, $\mathbf{S}_{\mathrm{S}}$ is normalized score matrix outputted by any of the static graph link prediction algorithms, $\mathbf{S}_{\mathrm{T}}$ is a normalized score matrix generated by time-series link prediction algorithm and ms and mt are the minimum nonzero scores from corresponding matrices. The comprehensive set of experiments on two real-life data sets demonstrated the advantage of this approach. Experiments have shown that a combination of time-series *ARIMA* and time invariant *Katz* yields the most accurate combined model.

3.2.4. Tensor factorization model

A robust and innovative approach by Dunlavy *et al.* [27] based on CANDECOMP/PARAFAC (CP) tensor decomposition [28] avoids the loss of the temporal information. Its output is also a score matrix S, but instead of collapsing the temporal networks into the time time-invariant adjacency matrix it takes the three-way tensor representation $\mathbb{Z}$ (the size of $k \times k \times T$, where k is number of actors) of the temporal network. We define $\mathbb{Z}(i, j, t) = 1$ if there was a link from actor i to actor j at time t and $\mathbb{Z}(i, j, t) = 0$ otherwise. Given such a tensor its L components CP decomposition is

given by

$$\mathbb{Z} \approx \sum_{l=1}^{L} \lambda_l \boldsymbol{a}_l \circ \boldsymbol{b}_l \circ \boldsymbol{c}_l. \tag{17}$$

The symbol $\circ$ denotes the outer vector product, λ_l is a positive real number, $\boldsymbol{a}_l$ and $\boldsymbol{b}_l$ are k-size real value vectors and $\boldsymbol{c}_l$ is real value vector of size T. The CP tensor decomposition is analogous to the Singular Value Decomposition (SVD) with some notable differences (for more details see Ref. [27]). This approach is using the extracted components to assign a score proportional to the likelihood of future link appearance to each pair of actors. The outer product of vectors $\boldsymbol{a}_l$ and $\boldsymbol{b}_l$ captures the relationship between the actors in the component l, the temporal interactions are stored in vectors $\boldsymbol{c}_l$. Authors propose a simple heuristic to account for the temporal activity stored in temporal profiles $\boldsymbol{c}_l$ by averaging the scores for the last three observations. Therefore the score matrix $\mathbf{S}_{\mathrm{L}}$ for L-component decomposition is calculated as

$$\mathbf{S}_{\mathrm{L}} = \sum_{l=1}^{L} \gamma_l \lambda_l \boldsymbol{a}_l \circ \boldsymbol{b}_l, \tag{18}$$

where

$$\gamma_l = \frac{1}{T_0} \sum_{t=T-T_0+1}^{T} \boldsymbol{c}_l(t), \tag{19}$$

and T_0 is customarily set to 3. Here, it is impossible to determine the number of components L '*a priori*'. Instead, authors use an ensemble approach where the score matrices $\mathbf{S}_{\mathrm{L}}$ are calculated for various values of $L \in \{l, l_{+1}, l_{+2} \ldots l_{\max}\}$ and the final matrix $\mathbf{S}$ is composed as

$$\mathbf{S} = \sum_{L \in \{l, l_{+1}, l_{+2}, \ldots, l_{\max}\}} \frac{\mathbf{S}_{\mathrm{L}}}{||\mathbf{S}_{\mathrm{L}}||_{\mathrm{F}}}, \tag{20}$$

where $||\mathbf{S}_{\mathrm{L}}||_{\mathrm{F}}$ is the Frobenius norm of the score matrix $\mathbf{S}_{\mathrm{L}}$. Besides the link prediction, the tensor decomposition can also be used for the prediction of actors' attributes. In our experiments we use the tensor decomposition (implementation provided by MATLAB Tensor Toolbox [29]), as a baseline for both link and attribute predictions.

4. THE PROPOSED MODEL

A new efficient approach to make mutual predictions over the social network structure and actor attributes based on the historical data is described in this section. Specifically, given the social network link observations $\mathbf{N}^1 \cdots \mathbf{N}^T$, where $\mathbf{N}^t$ is a binary $k \times k$ sociomatrix, and the actor attribute observations $\boldsymbol{x}^1 \cdots \boldsymbol{x}^T$, where $\boldsymbol{x}^t$ is a k-length vector and k is a number of participating actors, we aim to make accurate predictions of the $\mathbf{N}^{T+1}$ and $\boldsymbol{x}^{T+1}$ in the future step.

Note that the network structure and the attribute values in a social network are mutually dependent on each other. Learning a joint distribution over them will end up predicting each of them alternatively using derived conditional models from the joint model. On the basis of this observation, we propose to learn directly two interdependent conditional prediction models, link prediction model, and actor prediction model, which can then be used to predict the network structure and attribute values interdependently to avoid the expensive inference in htERGM. Our overall model will be called extended tERGM (etERGM), as the conditional models are still formulated under the similar framework as in tERGM.

4.1. Actor Attribute Prediction Model

For actor attribute prediction we use the following log-linear model:

$$P\left(\boldsymbol{x}^t | \boldsymbol{x}^{t-1}, \mathbf{N}^t, \boldsymbol{\gamma}\right) = \frac{1}{Z(\boldsymbol{x}^{t-1}, \mathbf{N}^t, \boldsymbol{\gamma})} \exp\left\{\boldsymbol{\gamma}' \boldsymbol{\psi}\left(\boldsymbol{x}^t, \boldsymbol{x}^{t-1}, \mathbf{N}^t\right)\right\} \mathbb{N}(\boldsymbol{x}^t). \tag{21}$$

It describes the transition of attributes from time $t-1$ to time t, conditioning on the network structure $\mathbf{N}^t$ at time t. Here, $Z(\boldsymbol{x}^{t-1}, N^t, \boldsymbol{\gamma})$ is a normalization constant, and $\mathbb{N}(\boldsymbol{x}^t)$ is the regularization prior. As the basis for the prior, we used Gaussian multivariate distribution. The mean and covariance for our Gaussian regularized prior are estimated from training data. We choose Gaussian as a prior because of its smoothening effect on actor attributes along the time axis and thus inhibiting the oscillation of the predicted values. Our choice of prior regularizes the attribute predictions such that they stay within the range of their domain.

This model encodes the dependency of the attribute values $\boldsymbol{x}^t$ over the network structure represented by $\mathbf{N}^t$ in a direct way. The transposed model parameter $\boldsymbol{\gamma}'$ is a vector corresponding to the statistic vectors $\boldsymbol{\psi}(\boldsymbol{x}^t, \boldsymbol{x}^{t-1}, N^t)$ which encodes the dependencies between the links among actors and theirs attributes. We used three statistics ψ_{links}, ψ_{sim}, and ψ_{dyads}:

$$\psi_{\text{links}}(\mathbf{N}^t, \boldsymbol{x}^t) = k \frac{\sum_{i<j}^{k} N_{ij}^t N_{ji}^t \mathbb{I}(|x_i^t - x_j^t| < \sigma)}{\sum_{i<j}^{k} N_{ij}^t N_{ji}^t}, \tag{22}$$

$$\psi_{\text{sim}}\left(\boldsymbol{x}^t, \boldsymbol{x}^{t-1}\right) = \sum_{i}^{k} \mathbb{I}\left(x_i^t, x_i^{t-1}, \sigma\right), \tag{23}$$

$$\psi_{\text{dyads}}\left(\mathbf{N}^t, \boldsymbol{x}^t\right) = k \frac{\sum_{ij}^{k} N_{ij}^t \mathbb{I}(|x_i^t - x_j^t| < \sigma)}{\sum_{ij}^{k} N_{ij}^t}. \tag{24}$$

To derive the ψ_{links} statistics we exploited the homophily effect often found in social network, also stated as 'birds of feather flock together'. We tested our assumption on two real-life data sets. For each data set we measured the average of the absolute values of the actor attributes differences, defined as $|x_i^t - x_j^t|$, for three distinct cases. We measured the average distance between the actors that are not connected, that is, $N_{ij}^t = N_{ji}^t = 0$; that are partially connected by a single link: $N_{ij}^t = 1$ or $N_{ji}^t = 1$; and that are fully connected: $N_{ij}^t = N_{ji}^t = 1$. We discovered that fully connected actors on average have lesser absolute attribute difference than actors who are connected partially or not connected at all. On the basis of this discovery, we defined the ψ_{links} statistics as the ratio of the count of the fully connected pairs where actors attributes are similar, to the count of the fully connected pairs in graph $\mathbf{N}^t$. We deem actor's i and j attributes similar if $|x_i^t - x_j^t| < \sigma$, where σ is a parameter and is estimated from the training data. In our experiments the training data were normalized to one standard deviation and we customarily set $\sigma = 0.3$. For example, if the number of actors is $k = 50$, the total number of fully linked pairs is 40 and out of those ten are between actors with similar attributes, the value of ψ_{links} is $50 \times \frac{10}{40} = 12.5$.

ψ_{sim} captures temporal stability of actors' attributes. If actors' attributes do not change between the observations, ψ_{sim} is large and is small otherwise. $\mathbb{I}$ is the indicator function which returns 1 if actor value at times t and $t-1$ is similar, that is, $|x_i^t - x_i^{t-1}| < \sigma$ and returns 0 otherwise. For example, if $x_i^t = 0.6$, $x_i^{t-1} = 0.4$, and parameter σ is set to 0.3, we increase ψ_{sim} by 1.

Statistics ψ_{dyads} measures the similarity of attributes for the connected actors. It is a fraction of the total count of the linked pairs, which have similar attributes (as defined by $\mathbb{I}(x_i^t, x_j^t, \sigma)$) to the total count of all linked pairs of the graph. For example, if the number of actors is $k = 50$, the total number of links in the network $\mathbf{N}^t$ is 25 and out of those there are 12 links between actors having similar attributes (as defined by indicator function $\mathbb{I}$), the value of ψ_{dyads} is $50 \times \frac{12}{25} = 24$.

All three statistics are scaled such that their values are always in the $[0; k]$ range, where k is the number of actors.

4.2. Link Prediction Model

The links are predicted by a log-linear model defined as

$$P(\mathbf{N}^t|\mathbf{N}^{t-1}, \boldsymbol{x}^t, \boldsymbol{\theta}) = \frac{1}{Z(\mathbf{N}^{t-1}, \boldsymbol{x}^t, \boldsymbol{\theta})} \exp\left\{\boldsymbol{\theta}' \boldsymbol{\psi}\left(\mathbf{N}^t, \mathbf{N}^{t-1}, \boldsymbol{x}^t\right)\right\}. \quad (25)$$

Similar to the tERGM model, this link prediction model defines the transition from $\mathbf{N}^{t-1}$ to $\mathbf{N}^t$. However, different from before, we incorporate the dependency of $\mathbf{N}^t$ over the attributes $\boldsymbol{x}^t$ into the model directly.

In this log-linear model, $\boldsymbol{\psi}(\mathbf{N}^t, \mathbf{N}^{t-1}, \boldsymbol{x}^t)$ denotes a list of statistics. Here, we reused the four statistics already used at tERGM, which were shown in Eqs (5)–(8). Statistics defined in Eqs (5)–(8) capture only dependencies between matrices $\mathbf{N}^{t-1}$ and $\mathbf{N}^t$ and they do not link attribute values to the network. Thus, we define the additional statistics to capture such linkage ψ_{links}—the same statistics that were used in the actor attribute prediction model.

4.3. Learning Algorithm

The actor attribute prediction model and link prediction model proposed in Sections 4.1 and 4.2 are both log-linear models. Two sets of parameters, $\boldsymbol{\theta}$ and $\boldsymbol{\gamma}$, need to be learned there.

4.3.1. Learning the link prediction model parameter

The parameter $\boldsymbol{\theta}$ of the link prediction model consists of five coefficients $\{\theta_{\text{D}}, \theta_{\text{S}}, \theta_{\text{R}}, \theta_{\text{T}}, \theta_{\text{links}}\}$ corresponding to the statistics $\{\psi_{\text{D}}, \psi_{\text{S}}, \psi_{\text{R}}, \psi_{\text{T}}, \psi_{\text{links}}\}$, respectively. To learn $\boldsymbol{\theta}$ we can apply Newton's optimization method demonstrated in Ref. [13] in straightforward fashion. The algorithm works as following: the $\boldsymbol{\theta}$ parameter is randomly initialized to a sensible value, then in an iterative manner it approximates the expectation using Gibbs sampling [9], followed by an update of parameter $\boldsymbol{\theta}$ such that it increases log-likelihood function of the observed temporal network. The algorithm repeats approximation with updated parameters until updates to $\boldsymbol{\theta}$ become very small (we reach convergence). The Gibbs sampling of sociomatrices used in learning link prediction model parameters was well described in Ref. [12] and here we provide its brief description. To draw sociomatrix samples from posterior distribution initial matrix $\mathbf{N}^{(1)}$ is chosen, and each element of this matrix is stochastically updated. The updating algorithm circles through the matrix $\mathbf{N}^{(1)}$ defining the Markov chain stochastic process which asymptotically approximates required random graph distribution. At each update step only one element of the matrix is considered and stochastically updated, therefore the matrices $\mathbf{N}^{(u)}$ and $\mathbf{N}^{(u+1)}$ differ in one element at update step u. If element $\mathbf{N}_{ij}^{(u)}$ at time u is being updated, then the probability of this element being 0 or 1 is defined by this conditional distribution:

$$P_{\boldsymbol{\theta}}\left(N_{ij}^{(u+1)} = a\right)|\mathbf{N}^{(u)}\right) = P_{\boldsymbol{\theta}}\left(N_{ij} = a|\ N_{hk}^{(u)} \forall (h,k) \neq (i,j)\right) \quad (a = 0, 1). \quad (26)$$

The update process works as following, for a given sociomatrix $\mathbf{N}$ define two sociomatrices $\mathbf{N}^{ij0}$ and $\mathbf{N}^{ij1}$, where both matrices are exact copies of matrix $\mathbf{N}$, except

the i, j element of matrices $\mathbf{N}^{ij0}$ and $\mathbf{N}^{ij1}$, defined, respectively, as N_{ij}^{ij0} and N_{ij}^{ij1}, are set to: $N_{ij}^{ij0} = 0$, $N_{ij}^{ij1} = 1$. The conditional distribution given in Eq. (26) is defined as

$$\begin{aligned}&\operatorname{logit}\{P_{\boldsymbol{\theta}}(N_{ij}^{(u+1)} = 1 |\ N_{hk}^{(u)}\ \ \forall (h,k) \neq (i,j))\} \\ &\quad = \boldsymbol{\theta}^{'}(\psi(N^{ij1}) - \psi(N^{ij0})). \end{aligned} \tag{27}$$

The formula (27) defines the Gibbs sampling process for the time-invariant sociomatrix. We redefine formula (27) for our link prediction model as:

$$\begin{aligned}&\operatorname{logit}\{P_{\boldsymbol{\theta}}(N_{ij}^{t,(u+1)} = 1 |\ N_{hk}^{t,(u)}\ \ \forall (h,k) \neq (i,j), \mathbf{N}^{t-1}, \boldsymbol{x}^t)\} \\ &\quad = \boldsymbol{\theta}'\{\boldsymbol{\psi}(\mathbf{N}^{t,(u),ij1}, \mathbf{N}^{t-1}, \boldsymbol{x}^t) \\ &\qquad -\boldsymbol{\psi}(\mathbf{N}^{t,(u),ij0}, \mathbf{N}^{t-1}, \boldsymbol{x}^t)\}. \end{aligned} \tag{28}$$

We set i, j element of sampled sociomatrix $\mathbf{N}^{t,(u+1)}$ to 1 at update step u with probability defined in formula (28), while leaving all other element of the matrix intact. Sampling one link at a time requires recalculation of model statistics, which can be slow. To speed up the sociomatrix sampling process the techniques 'big update' and 'inversion step' were described in Ref. [12]. The 'big update' technique specifies the update of large set of cells of the sampled sociomatrix instead of the single link. The 'inversion step' technique randomly inverts the whole sampled sociomatrix with a very small probability. It has been shown that 'big update' speeds up the sampling process and 'inversion step' leads to better Markov chain mixing [12]. We implemented both techniques in our approach.

4.3.2. *Learning the actor attribute prediction model parameter*

Learning $\boldsymbol{\gamma}$ coefficients $\{\gamma_{\text{links}}, \gamma_{\text{sim}}, \gamma_{\text{dyads}}\}$ of actor attribute prediction model which correspond to the statistics $\{\psi_{\text{links}}, \psi_{\text{sim}}, \psi_{\text{dyads}}\}$ is done similarly to learning $\boldsymbol{\theta}$ of the link prediction model. Let,

$$\begin{aligned}&L(\boldsymbol{\gamma}; \boldsymbol{x}^1, \ldots, \boldsymbol{x}^T) \\ &\quad = \log P(\boldsymbol{x}^2, \boldsymbol{x}^3, \ldots, \boldsymbol{x}^T | \boldsymbol{x}^1, \mathbf{N}^1 \cdots \mathbf{N}^T, \boldsymbol{\gamma}), \end{aligned} \tag{29}$$

$$M(t, \boldsymbol{\gamma}) = \mathbb{E}_{\boldsymbol{\gamma}}[\boldsymbol{\psi}(\underline{\boldsymbol{x}}^t, \boldsymbol{x}^{t-1}, \mathbf{N}^t) | \boldsymbol{x}^{t-1}, \mathbf{N}^t], \tag{30}$$

$$\begin{aligned}C(t, \boldsymbol{\gamma}) = \mathbb{E}_{\boldsymbol{\gamma}}[&\boldsymbol{\psi}(\underline{\boldsymbol{x}}^t, \boldsymbol{x}^{t-1}, \mathbf{N}^t)\boldsymbol{\psi} \\ &\times (\underline{\boldsymbol{x}}^t, \boldsymbol{x}^{t-1}, \mathbf{N}^t)' | \boldsymbol{x}^{t-1}, \mathbf{N}^t], \end{aligned} \tag{31}$$

where expectations are taken based on samples from random variable $\underline{\boldsymbol{x}}^t$. We note, that

$$\begin{aligned}&\nabla L(\boldsymbol{\gamma}; \boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^T) \\ &\quad = \sum_{t=2}^{T} (\boldsymbol{\psi}(\boldsymbol{x}^t, \boldsymbol{x}^{t-1}, \mathbf{N}^t) - M(t, \boldsymbol{\gamma})), \end{aligned} \tag{32}$$

and

$$\begin{aligned}&\nabla^2 L(\boldsymbol{\gamma}; \boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^T) \\ &\quad = \sum_{t=2}^{T} (M(t, \boldsymbol{\gamma}) M(t, \boldsymbol{\gamma})' - C(t, \boldsymbol{\gamma})). \end{aligned} \tag{33}$$

Similarly to Ref. [13] for learning link prediction model we apply Newton's optimization procedure to learn parameters $\boldsymbol{\gamma}$. The following procedure is used to approximate the expectations and update parameter values so that the log-likelihood function (29) is increased:

1. Randomly initialize $\boldsymbol{\gamma}$
2. For $i = 1$ up until convergence
3. For $t = 2, 3, \ldots, T$
4. Sample $\hat{\boldsymbol{x}}_{(i)}^{t,1}, \ldots, \hat{\boldsymbol{x}}_{(i)}^{t,C} \sim P(\underline{\boldsymbol{x}}^t | \boldsymbol{x}^{t-1}, \mathbf{N}^t, \boldsymbol{\gamma}^i)$
5. $\hat{\boldsymbol{\mu}}_{(i)}^t = \frac{1}{C} \sum_{c=1}^{C} \boldsymbol{\psi}(\hat{\boldsymbol{x}}_{(i)}^{t,c}, \boldsymbol{x}^{t-1}, \mathbf{N}^t)$
6. $\hat{C}_{(i)}^t = \frac{1}{C} \sum_{c=1}^{C} \boldsymbol{\psi}(\hat{\boldsymbol{x}}_{(i)}^{t,c}, \boldsymbol{x}^{t-1}, \mathbf{N}^t) \boldsymbol{\psi}(\hat{\boldsymbol{x}}_{(i)}^{t,c}, \boldsymbol{x}^{t-1}, \mathbf{N}^t)'$
7. $\hat{H}_{(i)} = \sum_{t=2}^{T} [\hat{\boldsymbol{\mu}}_{(i)}^t \hat{\boldsymbol{\mu}}_{(i)}^{t\,\prime} - \hat{C}_{(i)}^t]$
8. $\boldsymbol{\gamma}^{(i+1)} \leftarrow \boldsymbol{\gamma}^{(i)} - \alpha \hat{H}_{(i)}^{-1} \sum_{t=2}^{T} [\boldsymbol{\psi}(\boldsymbol{x}^t, \boldsymbol{x}^{t-1}, \mathbf{N}^t) - \hat{\boldsymbol{\mu}}_{(i)}^t]$

The parameter C in this algorithm specifies the number of samples drawn from posterior distribution. A greater value of C provides a finer update of parameter $\boldsymbol{\gamma}$ and usually achieves faster convergence. Also, for brevity, we omit the 'burn-in' parameter. 'Burn-in' specifies the number of samples that had to be thrown out for the good MCMC chain mixing. The main modification of our algorithm as compared to [13] is line 4. In a previous study [13], Gibbs sampling was used to sample from conditional distribution of sociomatrices. Here, we replaced Gibbs sampling with the Metropolis–Hastings algorithm to sample from $P(\underline{\boldsymbol{x}}^t | \boldsymbol{x}^{t-1}, \mathbf{N}^t, \boldsymbol{\gamma}^i)$ distribution. Parameter α shown at line 8 specifies the learning rate of the optimization procedure.

For high dimensional data (large number of actors), it is useful to use the 'block-at-a-time' technique discussed in Ref. [30]. In fact, this technique was implemented in our experiments.

4.4. Inference Algorithm

The goal of learning the etERGM is to make predictions over the social network structure and attribute values at time step $t + 1$; that is to infer the $\mathbf{N}^{t+1}$ and $\boldsymbol{x}^{t+1}$. As the actor prediction model and the link prediction

model are interdependent, we developed the following iterative algorithm to predict the network structure and actor attributes alternatively in a Gibbs sampling manner:

1. Randomly initialize $\hat{\boldsymbol{x}}^{t+1}$, $\hat{\mathbf{N}}^{t+1}$
2. $iter = 1$
3. Do while $iter < maxiterations$
4. Randomly pick and sample one link (u, v): $\hat{\mathbf{N}}^{t+1,(u,v)} \sim P\left(\underline{\mathbf{N}}^{t+1} | \mathbf{N}^t, \hat{\boldsymbol{x}}^{t+1}, \boldsymbol{\theta}\right)$
5. Set $\hat{\mathbf{N}}^{t+1}(u, v) = \hat{\mathbf{N}}^{t+1,(u,v)}$
6. Randomly pick and sample one attribute (w): $\hat{\boldsymbol{x}}^{t+1,(w)} \sim P\left(\underline{\boldsymbol{x}}^{t+1} | \boldsymbol{x}^t, \hat{\mathbf{N}}^{t+1}, \boldsymbol{\gamma}\right)$
7. Set $\hat{\boldsymbol{x}}^{t+1}(w) = \hat{\boldsymbol{x}}^{t+1,(w)}$
8. if all attributes in $\hat{\boldsymbol{x}}^{t+1}$ were updated
9. add $\hat{\boldsymbol{x}}^{t+1}$ to $\mathbb{X}$
10. if all links in $\hat{\mathbf{N}}^{t+1}$ were updated
11. add $\hat{\mathbf{N}}^{t+1}$ to $\mathbb{N}$
12. $iter = iter + 1$
13. $\boldsymbol{x}^{t+1} = mean(\mathbb{X})$
14. $S = \sum_{\hat{N} \in \mathbb{N}} \hat{N}$

The algorithm starts by randomly initializing the attributes $\boldsymbol{x}^{t+1}$ and sociomatrix $\mathbf{N}^{t+1}$. We initialize the sociomatrix $\mathbf{N}^{t+1}$ such that its link density is equal to the density of $\mathbf{N}^t$. In lines 4–7 of the algorithm we iteratively sample one link and one attribute at a time in a Gibbs sampling manner. In line 4 of the algorithm we sample one randomly selected link and use the resulting updated sociomatrix as input into the sampling distribution on line 6. In line 6 we repeat the process by sampling the attribute value of one randomly selected actor and input the updated vector of attributes into the sampling distribution in line 4. At each iteration we check if all attributes or links were updated by our sampling process. If they were, we add sampled actors' attributes vector to the collection $\mathbb{X}$ and sociomatrix to the collection $\mathbb{N}$. We continue collecting samples of sociomatrices and actors' attributes until we reach the maximum number of iterations. Our prediction for the actors' attributes is mean of the collected samples (line 13). Our score matrix of the link probabilities is the sum of the collected sociomatrices (line 14). Our algorithm is in essence a Gibbs sampling from a joint posterior distribution. We sample one link and one attribute at a time but to speed up the sampling process it is also possible to use a 'block-at-a-time' technique [30] for actors' attributes as well as a 'big update' and 'inversion step' techniques for sociomatrices [12].

4.5. Convergence

In the previous section we described the etERGM's inference algorithm. Its main idea is an iterative substitution of the drawn samples into interlocking probability distributions. The inference algorithm is intuitive by itself but we need to ensure that our inference procedure indeed converges. To evaluate convergence property of our algorithm we applied the standard convergence measurement for ERGM's [12]. It works as following: (i) estimate the ERGM's parameter vector $\boldsymbol{\theta}$, (ii) draw the multiple samples from the estimated model, (iii) for each drawn sample calculate the model's statistics ψ_k, (iv) estimate t-ratio as

$$t_k = \frac{E_\theta(\psi_k) - \psi_0}{SD_\theta(\psi_k)}, \tag{34}$$

where ψ_0 is the actual value of the statistic. It was suggested [12] that $|t_k| \leq 0.1$ indicates an excellent convergence, $0.1 < |t_k| \leq 0.2$ is good and $0.2 < |t_k| \leq 0.3$ is fair.

To calculate t-ratios for each statistic of both prediction models we trained etERGM on a single transition step from $t = 2$ to $t = 3$ of the *Delinquency*, a real-life data set which will be described in the experiments section. Using estimated model parameters $\boldsymbol{\theta}$ and $\boldsymbol{\gamma}$ we ran our inference algorithm. After the sufficient burn-in we sampled 1000 sociomatrices and 1000 vectors of the actors attributes. In Table 1 we report average differences between the simulated statistics and true values, the standard deviations of the simulated statistic and statistics' t-ratios.

Applying convergence criteria from Ref. [12], the results on average are fair or close to fair. The notable exception is the ψ_{dyads} statistics of the attribute prediction model which showed poor convergence characteristic with the t-ratio of 0.79. We do have to consider, however, that the average difference for ψ_{dyads} is 2.00, which is not a lot considering that statistic can range from 0 to 26 (*Delinquency* has 26 actors so all statistics are scaled proportional to $k = 26$, from 0 to 26). Overall, the results in Table 1 suggest that

Table 1. Convergence estimates of the inference algorithm on time steps $t = 2, 3$ of the real-life data set *Delinquency*.

Statistic	Average difference	Standard deviation	t-ratio
ψ_{links}	−1.09	2.42	−0.45
ψ_{sim}	1.35	2.18	0.62
ψ_{dyads}	2.00	2.54	0.79
ψ_{D}	0.12	0.36	0.33
ψ_{S}	−0.16	0.36	−0.44
ψ_{R}	0.42	1.13	0.37
ψ_{T}	0.19	0.86	0.22
ψ_{links}	−0.70	2.18	−0.32

link prediction model has better convergence properties than attribute prediction model.

The complex question of good chain mixing for Markov's processes is well studied [31]. In our experiments the number of throw away samples and total number of samples drawn from the Markov chain were derived empirically. During training we recorded the total number of samples versus the number of throw away samples drawn from posterior distribution and evaluated it against achieved accuracy. As a rule of thumb we set the number of throw away examples to be a half of the total number of the drawn samples.

4.6. Sufficiency

The often cited work Ref. [9] described the now canonical sufficient statistics for the ERGM's such as triad counts and 2-stars. Many more sufficient statistics for the nontemporal networks were published in later works [32]. The sufficient statistics for ERGM's can be derived because of the discovery made by Besag [33], who states that the probabilities of a random graph (social network in our case) are sufficiently described by cliques of its dependence graph. Here, the dependence graph D of a network N describes the conditional dependencies between links in N. The vertices in D are links connecting nodes in N and edges are conditional dependencies between the links given the rest of the graph N. The application of the dependence graph to define sufficient statistic was applied so far to the static nontemporal networks. It is not clear how it can be used for the temporal models such as ours because most of the statistics in etERGM are calculated on two temporally adjacent networks.

One possible way to determine the sufficiency for tERGM/etERGM could be to create a composition network $\mathbb{C}$ containing all T temporal sequences. The networks within $\mathbb{C}$ could be tied to each other by creating links between nodes at time step t and their own copies in the adjacent time steps $t-1$ and $t+1$. The dependence graph on such a network could be analyzed to confirm sufficiency of tERMG's statistics or to create a new statistics. The sufficiency question for the temporal models is a separate research topic by itself so that we do not address it in this article; however, this direction is very worthy future investigation.

5. EXPERIMENTS

To evaluate the proposed method, we conducted experiments on two synthetic data sets and two well-studied real-life data sets.

5.1. Synthetic Data Sets

The purpose of experiments on synthetic data was to investigate the proposed model under controlled conditions. The synthetic data sets were generated by applying Markovian process, that is, each generated network (time step) was derived from the previous network. To derive the next network from the previous one, we randomly selected 50% of the links in the previous network and reversed their direction. We proceeded by randomly selecting 10% of the links in the resulting network and deleting them. We counted the number of the links we have just deleted and randomly added the same number of the links to the actor pairs that were not connected. Next, we identified all incomplete transitive relationships in the network , that is, relationships where links were present from actor p to actor q and from actor q to actor r, but link from actor r to actor p was absent. The 20% of those identified incomplete transitive relationships were randomly picked and we completed their transitivity by adding link from actor r to actor p. To keep the density of the generated network steady, we counted the number of the transitive links we just have added and randomly deleted the same number of the links from resulting network. This completed generation of one time step. The consequent network was generated by going through the same procedure, and we used the network we have just created as the starting point. The first network in the sequence was derived from the random graph. The data values of the actors were generated after the network time series were completed. Similarly to how the sequence of the networks was generated, the values of the actors from the previous time step were used to populate the next time step. To transform the k-length vector of actor attributes, where k is the number of actors, from the previous time step into the next, we randomly picked 30% of the actors from the previous time step and added to their values zero mean one standard deviation random Gaussian noise. Additionally, the fully linked actor pairs were identified, that is, where the link was present from actor i to actor j and from actor j to actor i, and for these actors we set the value of one actor equal to the value of its neighbor plus small random Gaussian noise. This procedure completed the generation of actor attributes for one time step. The consecutive time step was generated by applying the same procedure, where the actor attributes vector we just created was used as the starting point. To generate the first time step k-length vector of actor attributes we started off from the random Gaussian vector with zero mean and standard deviation one.

We created two synthetic data sets by following our Markovian procedure. *Data set1* consists of 30 networks (average network density was 30%), each network consisting of 30 actors. Here, density is defined as the proportion of links to the possible number of links. *Data set2* is a

time series graph of 100 actors observed over 30 time steps. The average network density in *Data set2* is 10%. The purpose of our synthetic experiments was to investigate the prediction accuracy of actors attributes and test our conjecture that the proposed model, which in interlocking fashion learns network structure and actors values, is superior in link prediction task to the models that only use network topology and/or time-series information. To test this assumption, we compared our algorithm with algorithms of *Common Neighbor* (CN), *Preferential Attachment* (PA), *Adamic-Adar* (AA), *Katz* (KZ), *ARIMA* (AR), *Hybrid Time Series Link Prediction Algorithm* (HA), *Tensor Factorization* (TF), and *Temporal Exponential Random Graph Model* (tERGM). For HA (Section 3.2.3) we used *ARIMA* and *Katz* as its input algorithms, because it was reported that these combined predictors achieve the highest accuracy [20]. For *Tensor Factorization* (Section 3.2.4) we used the number of decompositions in increments of 10: $L \in \{10, 20, \ldots, k\}$. We also compared our algorithm to tERGM, which uses a similar exponential model framework as our model, but does not consider the actors' attributes, whereas our approach does. Also, a baseline '$T-1$' approach was used which assumes $\mathbf{N}^{T+1}$ is network $\mathbf{N}^{T}$. As we shall see, the '$T-1$' algorithm in many circumstances performs quite well.

To measure link prediction performance we used the *area under curve* (AUC). The AUC is a preferred way to measure performance of the link prediction algorithms because social networks usually have a low link density (as low as 0.1%), which makes these data sets imbalanced [20,27]. The output of a CN, PA, KZ, AR, HA, and TF algorithms is a score matrix $\mathbf{S}$, where each entry $\mathbf{S}(i, j)$ assigns the link occurrence score proportional to the predicted probability of the link from actor i to actor j at time step $T+1$. Our inference algorithm can also produce such score matrix. At line 4 of our inference procedure (Section 4.4) we can add the sampled matrices which will result in score matrix $\mathbf{S}$: $\mathbf{S} = \sum_{k=i}^{B} \hat{\mathbf{N}}^{i}$. To derive AUC from score matrix $\mathbf{S}$, we moved the threshold parameter in small increments from matrix's smallest to its largest value. Each time we incremented the threshold, we created the intermediate binary matrix and set its entries for all i and j where $\mathbf{S}(i, j) < threshold$ to 0 and the rest of the entries to 1. Thus, initially, our binary prediction matrix contained all 1's and at the end it contained all 0's. While continuously moving our threshold parameter, we recorded the percentage of *true positive* links, that is, the number of correctly predicted links divided by the total number of positive links in the target matrix, and percentage of *false positive* links, that is, the number of predicted links that were not in the target matrix divided by the total number of negative links. The *Receive Operating Characteristics* (ROC) curve [34] is constructed by using the x-axis for false positive percentages and the y-axis for true positive ones. We calculated AUC, bounded between 0 and 1, based on generated ROC. A perfect algorithm will have AUC = 1, whereas a random algorithm will have AUC = 0.5. A better predictor will always have larger AUC. A similar procedure was used to obtain the prediction score matrix $\mathbf{S}$ from tERGM.

We calculated the mean square error (MSE) between the predicted attribute vector and true attribute vector to measure the accuracy of actor attributes prediction. The proposed method was compared with three baseline methods. One baseline method assumes that the actor attributes do not change between time steps T and $T+1$, that is, $\boldsymbol{x}^{T+1} = \boldsymbol{x}^{T}$. The second baseline was to use the history mean as the prediction, that is, $\boldsymbol{x}^{T+1} = mean(\boldsymbol{x}^{1} : \boldsymbol{x}^{T})$. Although these two baseline predictors are simple they are difficult to beat in practice. Our third baseline was the *Tensor Factorization* technique from Section 3.2.4. We found that low values of L (number of decompositions) are preferable when using *Tensor Factorization* to predict attributes and in our experiments we set $L = 3$.

We also investigated how the length of the historical data used for training influences our predictors. We compared predictors by training on 1, 2, 5, and 10 previous time steps by setting up sliding windows of 1, 2, 5, and 10 training time steps. For example, in the experiments with five training time steps we trained predictors on time steps from 6 to 10 to predict time step 11, from 7 to 11 to predict time step 12 and so on until we predicted 30th time step. Because we only did predictions for the time steps from 11 to 30 we collected 20 AUC and MSE values for each experiment. The AUC average and sample standard deviation of each predictor on synthetic *Data set1* are presented in Table 2. We report the MSE averages and sample standard deviation of each predictor actor attribute predictor on synthetic *Data set1* in Table 3.

The t-test pairwise statistics, comparing etERGM results with that of the second best predictor, for each set of experiments reported in Tables 2 and 3 were statistically significant with p-values less than 0.01.

Some of the entries in Table 2 are not filled in. We did not conduct *ARIMA* experiments using 1 and 2 training time steps because such short time-series do not provide enough data points for meaningful time-series predictions. The corresponding entries for the hybrid algorithm (HA) are not filled in either, because it uses an *ARIMA* score matrix as its input. For the *Tensor Factorization* baseline we only considered tensors with at least two time steps.

Our model requires at least one transition step for training, that is, two time steps, that is why the first column in etERGM row in Tables 2 and 3 is absent. All entries for the '$T-1$' algorithm are the same, because no matter how many time steps used for training, '$T-1$' needs just one

Table 2. Links prediction on synthetic *Data set1* : AUC averages and sample standard deviations of 20 experiments.

	Number of training time steps			
Predictor	1	2	5	10
AR	—	—	0.58 ± 0.02	0.55 ± 0.03
PA	0.62 ± 0.03	0.61 ± 0.03	0.60 ± 0.02	0.58 ± 0.03
T-1	0.71 ± 0.02	0.71 ± 0.02	0.71 ± 0.02	0.71 ± 0.02
CN	0.53 ± 0.03	0.53 ± 0.03	0.54 ± 0.03	0.55 ± 0.03
AA	0.51 ± 0.07	0.52 ± 0.03	0.54 ± 0.03	0.54 ± 0.04
KZ	0.71 ± 0.03	0.73 ± 0.03	0.68 ± 0.03	0.65 ± 0.02
HA	—	—	0.69 ± 0.03	0.65 ± 0.02
TF	—	0.75 ± 0.03	0.75 ± 0.02	0.73 ± 0.02
tERGM	—	0.78 ± 0.03	0.83 ± 0.02	0.84 ± 0.01
etERGM	—	**0.83 ± 0.01**	**0.87 ± 0.01**	**0.88 ± 0.01**

Table 3. Attributes prediction on synthetic *Data set1* : MSE averages and sample standard deviations of 20 experiments.

	Number of training time steps			
Predictor	1	2	5	10
Previous network	1.29 ± 0.04	1.29 ± 0.04	1.29 ± 0.04	1.29 ± 0.04
Average	1.29 ± 0.04	0.79 ± 0.04	1.01 ± 0.04	1.12 ± 0.02
TF	—	0.75 ± 0.45	1.02 ± 0.42	1.15 ± 0.26
etERGM	—	**0.74 ± 0.03**	**0.70 ± 0.04**	**0.67 ± 0.03**

previous time step to make prediction. Results presented in Table 2 are in many ways similar to results reported in Ref. [20] on real-life data. Just like in Ref. [20], we see that when predicting from a short history (two previous networks) *Katz* performs exceptionally well, and when it is used in conjunction with *ARIMA* as a hybrid link prediction algorithm (HA), the result is even better. We also noticed close correlations between *Common Neighbor* and *Adamic-Adar*, which were also reported in Ref. [20]. The *Tensor Factorization* algorithm also performs very well. We can see from Table 2 that in the link prediction task our model outperformed models that do not exploit actor attributes. etERGM performed better as we added more historical data to the training data set, whereas time-invariant algorithms show the reverse trend. This observation makes sense, as under Markovian assumption our model will perform better given more historical steps to train, whereas for the time-invariant algorithms the excess of historical data will appear as noise. The results for attributes predictions reported in Table 3 are somewhat similar to the link prediction results. The only exception is *Tensor Factorization* which shows great variance predicting 20 time steps, its variance is decreasing however as the volume of training data increases. We can see that etERGM accuracy is getting better as more history was provided for training. Also, the Markovian nature of the data set is evident by poor accuracy of the 'Average' predictor as compared to the 'Previous Network'.

We repeated the same set of experiments on *Data set2*. This data set also consists of 30 time steps, with networks of 100 actors. The results of these experiments are reported in Tables 4 and 5.

Here we observed trends similar to experiments on *Data set1*. The pairwise t-test comparison showed that etERGM accuracy improvement are statistically significant compared to that of the second best predictor (p-values were less than 0.01).

5.2. Real-Life Data Sets

We have also conducted experiments on two real-life data sets. The first, *Delinquency*, is a well-studied data set [4] consisting of four temporal observation of 26 students in a Dutch school class. For each observation, the researches asked each student to identify up to 12 pupils as friends. Also, the researchers collected delinquency measures every time the questioning was done. Delinquency measure is a five-point scale score ranging from 1 to 5, defined as a rounded average of stealing, vandalizing, fighting, and graffiti. The delinquency score 1–5 was assigned based on frequency of incidents over the last three months, where: 1 = never, 2 = once, 3 = 2–4 times, 4 = 5–10 times, 5 = more than 10 times. The distribution of the delinquency varied between each measurement and was highly skewed. The density of the network formed using student relationships was low (on average between 13 and 17%). The objective was to predict the relationship network and delinquency score of each student at the observation time step $t = 4$, based only on previous three surveys.

Table 4. Links prediction on synthetic *Data set2*: AUC averages and sample standard deviations of 20 experiments.

	Number of training time steps			
Predictor	1	2	5	10
AR	—	—	0.66 ± 0.03	0.64 ± 0.03
PA	0.68 ± 0.04	0.66 ± 0.04	0.67 ± 0.04	0.64 ± 0.04
T-1	0.71 ± 0.01	0.71 ± 0.01	0.71 ± 0.01	0.71 ± 0.01
CN	0.62 ± 0.05	0.59 ± 0.05	0.64 ± 0.05	0.57 ± 0.04
AA	0.62 ± 0.05	0.58 ± 0.07	0.62 ± 0.06	0.58 ± 0.03
KZ	0.75 ± 0.03	0.73 ± 0.03	0.75 ± 0.02	0.70 ± 0.03
HA	—	—	0.76 ± 0.03	0.71 ± 0.03
TF	—	0.78 ± 0.02	0.80 ± 0.01	0.79 ± 0.01
tERGM	—	0.79 ± 0.02	0.81 ± 0.03	0.82 ± 0.03
etERGM	—	**0.86 ± 0.01**	**0.87 ± 0.01**	**0.88 ± 0.01**

Table 5. Attributes prediction on synthetic *Data set2*: MSE averages and sample standard deviations of 20 experiments.

	Number of training time steps			
Predictor	1	2	5	10
Previous network	0.82 ± 0.03	0.82 ± 0.03	0.82 ± 0.03	0.82 ± 0.03
Average	0.82 ± 0.03	0.35 ± 0.01	0.46 ± 0.01	0.57 ± 0.02
TF	—	0.35 ± 0.15	0.46 ± 0.15	0.58 ± 0.11
etERGM	—	**0.30 ± 0.01**	**0.29 ± 0.01**	**0.27 ± 0.03**

The second real-life data set used in our experiments is called *Teenagers* [35,36]. This data set consists of three temporal observations of 50 teenagers. Just like in *Delinquency*, teenagers were asked to identify their 12 best friends. The other measurement was the student's alcohol consumption. This measurement was defined on a 5 points scale: 1 = none, 2 = once or twice a year, 3 = once a month, 4 = once a week, and 5 = more than once a week. The goal in this data set was to predict the relationship graph and the teenager's alcohol consumption score at the observation time step $t = 3$, based on two previous observations. The *Teenagers*' network density is even lower than in *Delinquency*, and is hovering at about 4.5%.

To learn the model, we have used parameters specified in Table 6 for *Delinquency* data and Table 7 for *Teenagers*. Once parameters were obtained, we used our inference technique (Section 4.4) to obtain predictions reported at Tables 8 and 9. Table 8 contains the accuracies of each baseline prediction model and our proposed approach measured by AUC. In Table 9 we report the average and sample standard deviation for TF, tERGM, and etERGM predictors based on 20 runs per each experiment. We can see that our approach had achieved the higher accuracy in links prediction than any other predictor on *Delinquency*. The pairwise t-test comparison showed that etERGM accuracy improvement are statistically significant compared to that of the *Tensor Factorization*—the second best predictor (p-values were less than 0.01). For the *Teenagers* data set, the results of TF and etERGM were the same. Table 9 contains the averages, sample standard deviation, and analysis of statistical significance of the actor attributes predictions on both real-life data sets, which are based on the same 20 runs. Here, the proposed approach, as measured by MSE, outperforms baseline predictors. The *Tensor Factorization* algorithm performs very well in link prediction task, however, for the attribute prediction its results seem to correlate with *Average* predictor.

Overall, in both experiments on real-life data sets the etERGM outperformed the conventional predictors in prediction of actor's attributes and the link prediction in most cases.

Table 6. Parameters of *Delinquency* data set.

Parameter	Parameter value	Description
K	26	Number of actors
B	1000	Number of network samples
C	10000	Number of data samples
T	4	Number of time steps
σ	0.3	Similarity threshold

Table 7. Parameters of *Teenagers* data set.

Parameter	Parameter value	Description
K	50	Number of actors
B	1000	Number of network samples
C	10000	Number of data samples
T	3	Number of time steps
σ	0.3	Similarity threshold

Table 8. Links prediction on *Delinquency* and *Teenagers* data sets: AUC averages and sample standard deviations of 20 experiments.

	Link prediction AUC							
	PA	$T-1$	CN	AA	KZ	TF	tERGM	etERGM
Delinquency	0.68	0.76	0.68	0.68	0.76	0.83 ± 0.00	0.82 ± 0.00	**0.84 ± 0.00**
Teenagers	0.62	0.76	0.60	0.61	0.69	**0.84 ± 0.00**	0.83 ± 0.00	**0.84 ± 0.00**

Table 9. Attributes prediction on *Delinquency* and *Teenagers* data sets: MSE averages and sample standard deviations of 20 experiments.

	Actor attributes (MSE)				
	Previous network	Average	TF	etERGM	p-value
Delinquency	1.12	1.01	1.00 ± 0.00	**0.94 ± 0.02**	$p < 0.05$
Teenagers	0.90	0.87	0.87 ± 0.00	**0.83 ± 0.01**	$p < 0.05$

5.3. Scalability

Empirical observations suggest that our approach is scalable to handle networks with hundreds of actors. The largest network size we conducted our experiments on had 500 actors and it is possible to handle a number slightly larger than that. This is sizable improvement compared to the htERGM model [17], which we consider to be closely related to ours. htERGM can only handle networks of up to 10 nodes (actors). The increase in efficiency over htERGM was achieved by decoupling link prediction and attribute prediction probabilistic models, whereas htERGM's model was joint.

Our approach is not directly scalable to the networks the size of Youtube or Facebook. This is attributable to the fact that most of our statistics/features have runtime of $O(n^2)$. Also, the features are constantly recalculated as part of the Gibbs sampling, which is inherently slow. However, it has been noted that networks of such size are impractical for the problem we are addressing here, because humans are incapable of handling more than couple hundred relations simultaneously. Consider a class with 500 students where researchers conducted multiple temporal observations of social relationships by asking each student to identify each student's friends. We know that any given student will indicate an absence of a link to the majority of other students. The links will be absent not because of the student's personal dislike, but simply because he/she never had a chance to get to know so many pupils. The similar conclusion that social networks with an invariant set of actors should not exceed a couple hundred nodes can be found in Ref. [4]. One possible way to scale up our approach to handle large networks is to run a community detection algorithm which would find the network clusters containing up to a few hundred nodes. The etERGM then can be applied on detected communities separately to make predictions. Such an approach would make it unnecessary for etERGM to consider all $n^2 - n$ relationships, which is a valid assumption because we know that people in large networks (such as Facebook) are only aware of the people within or close to their social circle. How to scale the etERGM to the large networks is out of scope of this work, but we will try to address it in our future research.

In recently published work on tERGM [37], our predecessor, authors discovered that with a smart choice of statistics for link transition probabilities it is possible to completely avoid costly parameter estimation procedure described in the previous sections. In a special case the choice of the tractable ψ function makes it unnecessary to perform expensive sampling steps, because it becomes possible to do exact Newton's updates. We leave investigation of such statistics and their empirical effects on the prediction accuracy for future research.

We have investigated the runtime behavior of our approach on two sets of experiments. In the first experiment we measured how the length of the historical data used for training influences the runtime of our approach. We achieved this by running our algorithm on the synthetic data set with 30 actors. For each run we changed the number of time steps used for training and we recorded the time the algorithm took to learn the parameters of both models and to do the inference. Results of this experiment are shown in Fig. 2. We can see that there is an obvious linear trend between the runtime and the amount of historical data used for training. In the second experiment we used the synthetic data set with four previous networks, where three were used for training and one for prediction.

In this experiment we were gradually changing the number of actors from 30 to 100 in increments of 5 and for each change we recorded the time it took for the algorithm to run. The results of the second experiment are reported in Fig. 3. Here we observe the quadratic trend between the runtime and the network size. This is expected, because as the number of actors grows we considered the quadratic increase in the number of possible relationships.

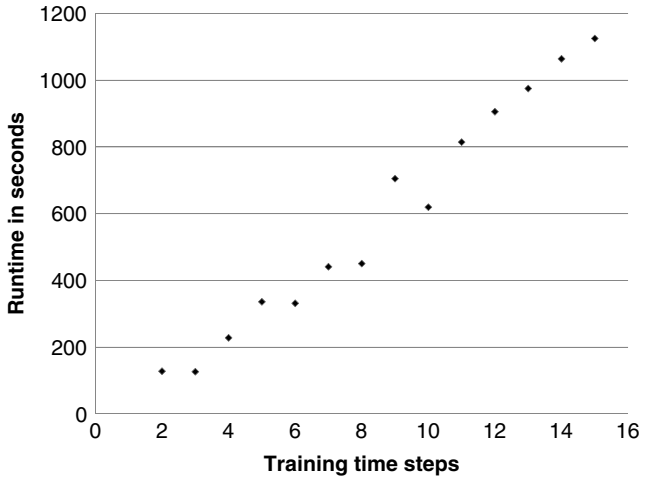

Fig. 2 etERGM runtime in seconds versus the number of time steps used for training for network with 30 actors.

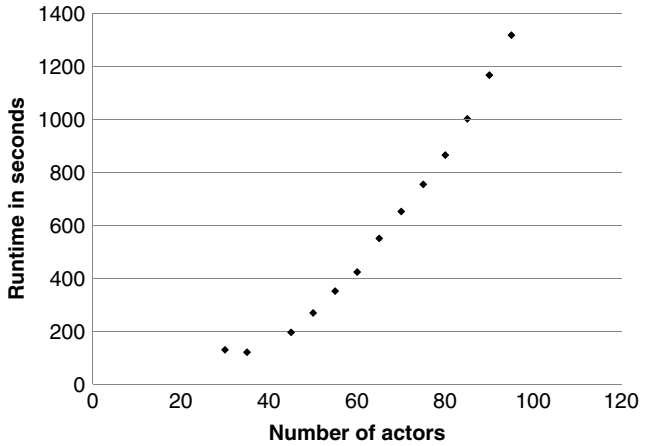

Fig. 3 etERGM runtime in seconds versus the number of actors in the network (three time steps used for training).

Despite the network size limitation, there are real-life problems that could benefit from our approach. Besides the two real-life data sets presented in this article, there are temporal network of similar size in sociological and biological domain. We are presently investigating a problem from zoological domain which could benefit from our approach.

6. DISCUSSION AND FUTURE WORK

We have shown that the etERGM is a viable predictor for links and attributes in temporal social networks. One of its core strengths is the separate learning of its two component models, which makes it applicable to larger problems. To the best of our knowledge, no published work had attempted to do simultaneous attributes and link predictions for temporal social networks. We foresee a number of improvements and future work directions based on our proposed method. As we have already mentioned, the Gibbs sampling technique for learning sociomatrix transition parameters is inherently slow.

In our study, only actors with a single real-valued attribute were considered. In many settings, more information is collected about participating actors. As a more general task in our ongoing research we are exploring the effects of augmenting etERGM to include multivariate attributes. Despite the fact that we specifically conducted the testing on temporal social networks, it would be also interesting to find out how the new method would perform on gene expression networks.

ACKNOWLEDGMENTS

This project is funded in part under a grant to Z. Obradovic with the Pennsylvania Department of Health. The Department specifically disclaims responsibility for any analyses, interpretations, or conclusions. We thank Aleksandar Obradovic for very useful comments on a draft of this manuscript.

REFERENCES

[1] G. W. Hill, Turrialba, social systems and the introduction of change, Rural Sociol 19 (1954), 45.
[2] W. de Nooy, A. Mrvar, and V. Batagelj, Exploratory Social Network Analysis with Pajek, (1st ed.), New York, NY, Cambridge University Press, 2005.
[3] S. Dynes, P. A. Gloor, R. Laubacher, and Y. Zhao, Temporal visualization and analysis of social networks, In Proceedings of the North American Association for Computational Social and Organizational Science Conference, North American Association for Computational Social and Organizational Science, 2004.
[4] T. Snijders, C. Steglich, and G. van de Bunt, Introduction to stochastic actor-based models for network dynamics, Social Networks 32 (2009), 44–60.
[5] M. Lahiri and T. Y. Berger-Wolf, Structure prediction in temporal networks using frequent subgraphs, In Proceedings of the Institute of Electrical and Electronics Engineers Symposium on Computational Intelligence and Data Mining, 2007.
[6] D. L. Nowell and J. Kleinberg, The link prediction problem for social networks, Proceedings of the 12th International Conference on Information and Knowledge Management, New York, NY, Association for Computing Machinery, 2003.
[7] T. Tylenda, R. Angelova, and S. Bedathur, Towards time-aware link prediction in evolving social networks, Proceedings of the 3rd workshop of Social Network Mining and Analysis—The 13th Association for Computing Machinery International Conference on Knowledge Discovery and Data Mining, New York, NY, Association for Computing Machinery, 2009.

[8] A. Popescul, R. Popescul, and L. H. Ungar, Statistical relational learning for link prediction, In Proceedings of the 18th International Joint Conference on Artificial Intelligence, Acapulco, Mexico, 2003.
[9] O. Frank and D. Strauss, Markov graphs, J Am Stat Assoc 81 (1986), 832–842.
[10] S. Wasserman and P. Pattison, Logit models and logistic regressions for social networks: I. An introduction to markov graphs and p*, Psychometrika 61 (1996), 401–425.
[11] P. J. Carrington, J. Scott, and S. Wasserman, Models and Methods in Social Network Analysis, (1st edn.), New York, NY, Cambridge University Press, 2005.
[12] T. A. B. Snijders, Markov chain monte carlo estimation of exponential random graph models, J Social Struct 3 (2002), 1–40.
[13] S. Hanneke and E. Xing, Discrete temporal models of social networks, Proceedings of the International Conference on Machine Learning Workshop on Statistical Network Analysis, New York, NY, Springer-Verlag, 2006.
[14] A. Ahmed and E. P. Xing, Recovering time-varying networks of dependencies in social and biological studies, Proc Natl Acad Sci U S A 106 (2009), 11878–11883.
[15] S. Kim, S. Imoto, and S. Miyano, Dynamic Bayesian network and nonparametric regression model for inferring gene networks from time series microarray data, Biosystems 75 (2003), 104–113.
[16] A. Bartal, E. Sasson, and G. Ravid, Predicting links in social networks using text mining and SNA, Social Network Analysis and Mining, International Conference on Advances in Social Network Analysis and Mining, 30, 2009.
[17] F. Guo, S. Hanneke, W. Fu, and E. P. Xing, Recovering temporally rewiring networks: a model-based approach, Proceedings of the 24th International Conference on Machine Learning, New York, NY, Association for Computing Machinery, 2007.
[18] Christian Steglich, Tom A. B. Snijders, and Michael Pearson, Dynamic networks and behavior: separating selection from influence, Sociol Methodol 40(1) (2010), 329–393.
[19] H. R. Bernard, P. Killworth, D. Kronenfeld, and L. Sailer, The problem of informant accuracy: the validity of retrospective data, Ann Rev Anthropol 13(1) (1984), 495–517.
[20] Z. Huang and D. K. J. Lin, The time-series link prediction problem with applications in communication surveillance, Inst Oper Res Manage Sci J Comput 21 (2009), 286–303.
[21] F. Heider, Attitudes and cognitive organization, J Psychol 21 (1946), 107–112.
[22] L. A. Adamic and E. Adar, Friends and neighbors on the web, Social Networks 25 (2001), 211–230.
[23] L. Katz, A new status index derived from sociometric analysis, Psychometrika 18 (1953), 39–43.
[24] A. Barabasi and R. Albert, Emergence of scaling in random networks, Science 286 (1999), 509–512.
[25] G. E. P. Box and G. M. Jenkins, Time series analysis; forecasting and control, (1st ed.), San Francisco, CA, Holden-Day, 1970.
[26] H. Akaike, A new look at the statistical model identification, Inst Elect Elect Eng Trans Autom Contr 19 (1974), 716–723.
[27] Daniel M. Dunlavy, Tamara G. Kolda, and Evrim Acar, Temporal link prediction using matrix and tensor factorizations, ACM Trans Knowledge Disc Data 5 (2011), 1–10.
[28] J. Carroll and Jih-Jie Chang, Analysis of individual differences in multidimensional scaling via an n-way generalization of Eckart-Young decomposition, Psychometrika 35 (1970), 283–319.
[29] B. W. Bader and T. G. Kolda, Efficient Matlab computations with sparse and factored tensors, SIAM J Sci Comput 30 (2007), 205–231.
[30] W. K. Hastings, Monte Carlo sampling methods using Markov chains and their applications, Biometrika 57 (1970), 97–109.
[31] D. A. Levin, Y. Peres, and E. L. Wilmer, Markov chains and mixing times, (1st ed.), New York, NY, American Mathematical Society, 2008.
[32] Tom A. B. Snijders, Philippa E. Pattison, Garry L. Robins, and Mark S. Handcock, New specifications for exponential random graph models, Sociol Methodol 36 (2006), 99–153.
[33] J. Besag, Spatial interaction and the statistical analysis of lattice systems, J R Stat Soc Ser B Methodol 36 (1974), 192–236.
[34] A. P. Bradley, The use of the area under the roc curve in the evaluation of machine learning algorithms, Pattern Recognit 30 (1997), 1145–1159.
[35] L. Michell, and A. Amos, Girls, pecking order and smoking, Social Sci Med 44 (1997), 1861–1869.
[36] M. L. Pearson and L. Michell, Smoke rings: social network analysis of friendship groups, smoking and drug-taking, Drugs Educat Prevent Policy 7 (2000), 21–37.
[37] S. Hanneke, W. Fu, and E. P. Xing, Discrete temporal models of social networks, Electron J Stat 4 (2010), 585–605.