To Be or When to Be? Temporal Relationship Prediction in Heterogeneous Networks

Sina Sajadmanesh ♯1, Sogol Bazargani ♯2, Jiawei Zhang ⋆3, Hamid R. Rabiee ♯4

♯ *Department of Computer Engineering, Sharif University of Technology*
Tehran, Iran
1 sajadmanesh@ce.sharif.edu
2 bazargani@dml.ir
3 rabiee@sharif.edu

⋆ *Department of Computer Science, Florida State University*
Tallahassee, FL, USA
2 jzhang@cs.fsu.edu

***Abstract*—Online social networks, World Wide Web, media and technological networks, and other types of so-called *information networks* are ubiquitous nowadays. These information networks are inherently *heterogeneous* and *dynamic*. They are heterogeneous as they consist of multi-typed objects and relations, and they are dynamic as they are constantly evolving over time. One of the challenging issues in such heterogeneous and dynamic environments, is to forecast those relationships in the network that will appear in the future. In this paper, we try to solve the problem of continuous-time relationship prediction in dynamic and heterogeneous information networks. This implies predicting the time it takes for a relationship to appear in the future, given its features that have been extracted by considering both heterogeneity and temporal dynamics of the underlying network. To this end, we first introduce a feature extraction framework that combines the power of meta-path-based modeling and recurrent neural networks to effectively extract features suitable for relationship prediction regarding heterogeneity and dynamicity of the network. Next, we propose a supervised non-parametric approach, called *Non-Parametric Generalized Linear Model* (Np-Glm), which infers the hidden underlying probability distribution of the relationship building time given its features. We then present a learning algorithm to train Np-Glm and an inference method to answer time-related queries. Extensive experiments conducted on synthetic data and three real-world datasets, namely Delicious.com, Movielens.org, and DBLP, demonstrate the effectiveness of Np-Glm in solving continuous-time relationship prediction problem vis-à-vis competitive baselines.**

I. Introduction

Link prediction is the problem of prognosticating a certain relationship, like interaction or collaboration, between two entities in a networked system that are not connected already [1]. Due to the popularity and ubiquity of networked systems in real world, such as social, economical, or biological networks, this problem has attracted a considerable attention in recent years and has found its applications in various interdisciplinary domains, such as viral marketing, bioinformatics, recommender systems, and social network analysis [2]. For example, suggesting new friends in an online social network [3] or predicting drug-target interactions in a biological network [4] are two quite different problems, but can both cast as the prediction task of friendship links and drug-target links, respectively.

The problem of link prediction has a long literature and is studied extensively in the last decade. Initial works on link prediction problem mostly concentrated on homogeneous networks, which are composed of single type of nodes connected by links of the same type [3], [5], [6]. However, many of the today's networks, such as online social networks or bibliographic networks, are inherently *heterogeneous*, in which multiple types of nodes are interconnected using multiple types of links [7], [8]. For example, a bibliographic network may contain author, paper, venue, etc. as different node types; and write, publish, cite, and so on as diverse link types that bind nodes with different types to each other. In these heterogeneous networks, the concept of a link can be generalized to a relationship, which can be constructed by combining different links with different types. For instance, author-cite-paper relationship can be defined in a bibliographic network as a combination of author-write-paper and paper-cite-paper links. Analogously, one can generalize the link prediction to *relationship prediction* in heterogeneous networks which tries to predict complex relationships instead of links [9].

While most of the studies on the link/relationship prediction in heterogeneous networks utilize a static snapshot of the underlying network, many of these networks are *dynamic* in nature, which means that new nodes and linkages are continually added to the network, and some existing nodes and links may be removed from the network over time. For example, in online social networks such as Facebook, new users are joining in the network every day, and new friendship links are being added to the network, gradually. This dynamic characteristic causes the structure of the network to change and evolve over time, and taking these changes into account can significantly boost the quality of link prediction task [10].

In recent years, newer studies have shifted from traditional link prediction on static and homogeneous networks toward newer domains, considering heterogeneity and dynamicity of the networks [11], [12], [13], [14], [15]. However, most of these works merely focus on one of these aspects, disregarding the other. Although there are quite a few studies that address both the challenges of heterogeneity and dynamicity [16], [17], to the best of our knowledge, all of them have ultimately formulated the link prediction problem as a binary classification task, i.e. predicting *whether* a link will appear in the network in the future. However, in dynamic networks, new links are continually appearing over time. So a much more interesting problem, which we call it *continuous-time link prediction* in this paper, is to predict *when* a link will emerge or appear between two nodes in the network. Examples of this problem include predicting the time at which two individuals become friends in a social network, or the time when two authors collaborate on writing a paper in a bibliographic network [9]. Inferring the link formation time in advance can be very useful in many concrete applications. For example, if a social network recommender system could predict the relationship building time between two people, then it can issue a friendship suggestion close to that time since it will have a relatively higher chance to be accepted. Good continuous-time link prediction results will lead to denser connections among users, and can greatly improve users' engagement that is the ultimate goal of online social networks [18].

In this paper, we aim to solve the problem of continuous-time relationship prediction, in which we forecast the relationship building time between two nodes in a dynamic and heterogeneous environment. This problem is very challenging from the technical perspective, and cannot be solved trivially for three main reasons. First, the formulation of continuous-time relationship prediction is quite different from conventional link prediction due to the involvement of temporal dynamics of the network and the necessity of considering network evolution time-line. Second, we only know the building time of those relationships that are already present at the network and for the rest of them that are yet to happen, which are excessive in number versus the existing ones, we lack such information. Finally, as opposed to the works concerning the binary link prediction, there are very rare works in the literature on continuous-time link prediction that attempt to answer the "when" question. To the best of our knowledge, the only work that has studied the continuous-time relationship prediction problem so far, is proposed by Sun et al. [9]. They infer a probability distribution over time for each pair of nodes given their features, and answer time-related queries about the relationship building time between the two nodes using the inferred distribution. However, the drawback of their method, not to mention neglecting the temporal dynamics of the network, is that it mainly relies on the assumption that relationship building times are coming from a certain probability distribution which must be fixed beforehand. This assumption though simplifying is very restrictive, because in real applications this distribution is unknown, and considering any specific one as a priori could be far from reality or limit the solution generality.

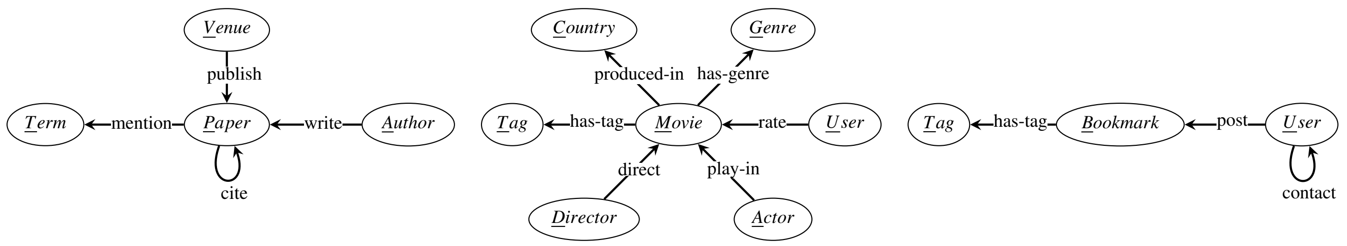
In order to address the above challenges, we propose a supervised non-parametric method to solve the problem of continuous-time relationship prediction. To this end, we first formally define the continuous-time relationship prediction problem and formulate the approach to solve it generally. Then, we introduce our novel feature extraction framework which leverages meta-path-based modeling and recurrent neural networks to deal with heterogeneity and dynamicity of information networks. Next, we present *Non-Parametric Generalized Linear Model* (NP-GLM) which models the distribution of relationship building time given the extracted features. The strength of this non-parametric model is that it is capable of learning the underlying distribution of the relationship building time, as well as the contribution of each extracted feature in the network. Afterwards, we propose an inference algorithm to answer queries, like the most probable time by which a relationship will appear between two nodes, or the probability of relationship creation between them during a specific period. Finally, we conduct comprehensive experiments over a synthetic dataset to verify the correctness of NP-GLM's learning algorithm, and on three real-world dataset - DBLP, Delicious, and MovieLens - to demonstrate the effectiveness and generality of the proposed method in predicting the relationship building time versus the relevant baselines. As a summary, we can enumerate our major contributions as follows:

- The proposed feature extraction framework can tackle heterogeneity of the data as well as capturing the temporal dynamics of the network by incorporating meta-path-based features into a recurrent neural network based autoencoder.
- Our non-parametric model takes a unique approach toward learning the underlying distribution of relationship building time without imposing any significant assumptions on the problem.
- Extensive evaluations over synthetic and real-world datasets is performed to indicate the effectiveness of the proposed method.
- To the best of our knowledge, this paper is the first one which studies the continuous-time relationship prediction problem in both dynamic and heterogeneous network configurations.

The rest of this paper is organized as follows. In Section II, we provide introductory backgrounds on the concept and formally define the problem of continuous-time relationship prediction. Then in Section III, we introduce our novel feature extraction framework. Next, we go through the details of the proposed NP-GLM method in Section IV, explaining its learning method and how it answers inference queries. Experiments on synthetic data and real-world datasets are described in Section V and VI, respectively. Section VII discusses the related works, and finally in Section VIII, we conclude the paper.

II. PROBLEM FORMULATION

In this section, we introduce some important concepts and definitions used throughout the paper and formally define the problem of continuous-time relationship prediction.



(a) DBLP (b) MovieLens (c) Delicious

Fig. 1. Schema of three different heterogeneous networks. Underlined characters are used as abbreviations for corresponding node types.

A. Heterogeneous Information Networks

An information network is *heterogeneous* if it contains multiple kinds of nodes and links. Formally, it is defined as a directed graph $G = (V, E)$ where $V = \bigcup_i V_i$ is the set of nodes comprising the union of all the node sets V_i of type i. Similarly, $E = \bigcup_j E_j$ is the set of links constituted by the union of all the link sets E_j of type j. Now we bring the definition of the *network schema* [19] which is used to describe a heterogeneous information network at a meta-level:

Definition 1. *(Network Schema) The schema of a heterogeneous network G is a graph $\mathcal{S}_G = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of different node types and $\mathcal{E}$ is the set of different link types in G.*

In this paper, we focus on three different heterogeneous and dynamic networks: (1) DBLP bibliographic network[1]; (2) Delicious bookmarking network[2]; and (3) MovieLens recommendation network[3]. The schema of these networks is depicted in Fig. 1. As an example, in the bibliographic network, $\mathcal{V} = \{Author, Paper, Venue, Term\}$ is the set of different node types, and $\mathcal{E} = \{\text{write}, \text{publish}, \text{mention}, \text{cite}\}$ is the set of different link types.

As we mentioned in the Introduction section about heterogeneous networks, the concept of a link can be generalized to a relationship. In this case, a relationship could be either a single link, or a composite relation constituted by concatenation of multiple links that together have a particular semantic meaning. For example, the co-authorship relation in the bibliographic network with the schema shown in Fig. 1a, can be defined as the combination of two *Author-write-Paper* links, making *Author-write-Paper-write-Author* relation. When dealing with link or relationship prediction in heterogeneous networks, we must exactly specify what kind of link or relationship we are going to predict. This specific relation to be predicted is called the *Target Relation* [9]. For example, in DBLP bibliographic network we aim to predict if and when an author will cite a paper from another author. Thus the target relation in this case would be *Author-write-Paper-cite-Paper-write-Author*.

[1]http://dblp.uni-trier.de/
[2]http://delicious.com/
[3]https://movielens.org/

B. Dynamic Information Networks

An information network is *dynamic* when its nodes and linkage structure can change over time. That is, in a dynamic information network, all nodes and links are associated with a birth and death time. More formally, a dynamic network at the timestamp τ is defined as $G^\tau = (V^\tau, E^\tau)$ where V^τ and E^τ are respectively the set of nodes and the set of links existing in the network at the timestamp τ.

In this paper, we consider the case that an information network is both dynamic and heterogeneous. This means that all network entities are associated with a type, and can possibly have birth and death times, regardless of their types. The bibliographic network is an example of both dynamic and heterogeneous one. Whenever a new paper is published, a new *Paper* node will be added to the network, alongside with the corresponding new *Author*, *Term*, and *Venue* nodes (if they don't exist yet). New links will be formed among these newly added nodes to indicate the *write*, *publish* and *mention* relationships. Some linkages might also form between the existing nodes and the new ones, like new *cite* links connecting the new paper with the existing papers in its reference list.

C. Continuous-Time Relationship Prediction

Suppose that we are given a dynamic and heterogeneous information network as G^τ lastly observed at the timestamp τ, together with its network schema S_G. Now, given the target relation R, the aim of continuous-time relationship prediction is to continuously forecast the relationship building time $t \geq \tau$ of R for any node pair of G^τ provided in a query.

To solve this problem, we first introduce a feature extraction framework to cope with both dynamicity and heterogeneity of the network, and then propose a non-parametric model which utilizes the extracted features to perform predictions about the building time of the target relationship.

III. Feature Extraction Framework

In this section, we present our framework to extract features which is designed to have three major characteristics: First, it effectively considers different type of nodes and links available in a heterogeneous information network and regard their impact on the building time of the target relationship. Second, it takes the temporal dynamics of the network into account and leverages the network evolution history instead of simply aggregating it into a single snapshot. Finally, the

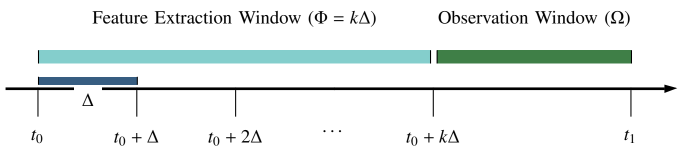

Feature Extraction Window ($\Phi = k\Delta$)
Observation Window (Ω)
Δ
t_0
$t_0 + \Delta$
$t_0 + 2\Delta$
$\cdots$
$t_0 + k\Delta$
t_1

Fig. 2. The evolutionary timeline of the network data.

extracted features are suitable for not only the link prediction problem, but also the generalized *relationship prediction*. We will incorporate these features in the proposed non-parametric model in Section IV to solve the continuous-time relationship prediction problem.

A. Data Preparation For Feature Extraction

To solve the problem of continuous-time relationship prediction in dynamic networks, we need to pay attention to the temporal history of the network data from two different point of views. First, we have to mind the evolution history of the network for feature extraction, so that the extracted features reflect the changes made in the network over time. Second, we have to specify the exact relationship building time for each pair of nodes, because our goal is to propose a supervised method to predict a continuous variable, which in this case is the relationship building time. Hence, for each sample pair of nodes, we need a feature vector $\mathbf{x}$, associated with a target variable t which indicates the building time of the target relationship between them.

Suppose that we have observed a dynamic network G^τ recorded in the interval $t_0 < \tau \leq t_1$. According to Fig 2, we split this interval into two parts: the first part for extracting the feature $\mathbf{x}$, and the second for determining the target variable t. We refer to the first interval as *Feature Extraction Window* whose length is denoted by Φ, and the second as *Observation Window*, whose length is denoted by Ω. Now, based on the existence in the observation window, target relationships fall within one of the following three different groups:

1) Relationships that are already formed before the beginning of the observation window (formed in the feature extraction window).
2) Relationships that will be built in the observation window for the first time (not existing before).
3) Relationships that will not form at all (neither in the feature extraction window nor in the observation window).

Those pairs of nodes that act as the starting and ending nodes of the relationships in the 2nd and 3rd categories constitute our data samples, and will be used in the learning procedure. For such pairs, we extract their feature vector $\mathbf{x}$ using the history available in the feature extraction window. For each node pair in the 2nd category, we have seen that the target relationship between them is created at a time like $t_r \in (t_0 + \Phi, t_1]$. So we set $t = t_r - (t_0 + \Phi)$ as the time it takes for the relationship to form since the beginning of the observation window. For these samples, we also set an auxiliary variable $y = 1$ which indicates that we have *observed* their exact building time. On the other hand, For node pairs in the 3rd category, we haven't seen their exact building time, but we know the it is definitely after t_1. For such samples, which we call *censored* samples, we set $t = t_1 - (t_0 + \Phi)$ that is equal to the length of the observation window Ω, and set $y = 0$ to indicate that the recorded time is in fact a lower bound on the true relationship building time. These type of samples though look useless, are also of interest because their features will give us some information about their time falling after t_1. As a result, each final sample is associated with a triple $(\mathbf{x}, y, t)$ representing its feature vector, observation status, and the time it takes for the target relationship to form, respectively.

TABLE I
SIMILARITY META-PATHS IN DIFFERENT NETWORKS

Network	Meta-Path	Semantic Meaning
DBLP	$A \to P \leftarrow A$	Authors co-write a paper
	$A \to P \to A \leftarrow P \leftarrow A$	Authors have common co-author
	$A \to P \to V \leftarrow P \leftarrow A$	Authors publish in the same venue
	$A \to P \to T \leftarrow P \leftarrow A$	Authors use the same term
	$A \to P \to P \leftarrow P \leftarrow A$	Authors cite the same paper
	$A \to P \leftarrow P \to P \leftarrow A$	Authors are cited by the same paper
Delicious	$U \leftrightarrow U \leftrightarrow U$	Users have common contact
	$U \to B \leftarrow U$	Users post the same bookmark
	$U \to B \to T \leftarrow B \leftarrow U$	Users post bookmarks with the same tag
MovieLens	$M \to A \leftarrow M$	Movies share an actor
	$M \to C \leftarrow M$	Movies belong to the same country
	$M \to D \leftarrow M$	Movies have the same director
	$M \to G \leftarrow M$	Movies have the same genre
	$M \to T \leftarrow M$	Movies have the same tag
	$U \to M \leftarrow U$	Users rate common movie
	$U \to M \to A \leftarrow M \leftarrow U$	Users rate movies sharing an actor
	$U \to M \to C \leftarrow M \leftarrow U$	Users rate movies from the same country
	$U \to M \to D \leftarrow M \leftarrow U$	Users rate movies of the same director
	$U \to M \to G \leftarrow M \leftarrow U$	Users rate movies with the same genre
	$U \to M \to T \leftarrow M \leftarrow U$	Users rate movies with the same tag

B. Dynamic Feature Extraction

In this part, we describe how to utilize the temporal history of the network in the feature extraction window in order to extract features for continuous-time relationship prediction problem. We first begin with the meta-path-based feature set for heterogeneous information networks, and then incorporate these features into a *recurrent neural network based autoencoder* to exploit the temporal dynamics of the network as well. Hereby, we begin by defining the concept of meta-path [19]:

Definition 2 (Meta-Path). *In a heterogeneous information network, a meta-path is a directed path following the graph of the network schema to describe the general relations that can be derived from the network. Formally speaking, given a network schema $\mathcal{S}_G = (\mathcal{V}, \mathcal{E})$, the sequence $v_1 \xrightarrow{\varepsilon_1} v_2 \xrightarrow{\varepsilon_2} \ldots v_{k-1} \xrightarrow{\varepsilon_{k-1}} v_k$ is a meta-path defined on S_G where $v_i \in \mathcal{V}$ and $\varepsilon_i \in \mathcal{E}$.*

Meta-paths are commonly used in heterogeneous information networks to describe multi-typed relations that have concrete semantic meanings. For example, in the bibliographic network whose schema are show in Fig 1a, we can define the co-authorship relation by the following meta-path:

$$Author \xrightarrow{write} Paper \xleftarrow{write} Author$$

or simply by $A \rightarrow P \leftarrow A$. Another example is the author citation relation, which in this paper is used as the target relation for DBLP network. It can be specified as:

$$Author \xrightarrow{write} Paper \xrightarrow{cite} Paper \xleftarrow{write} Author$$

abbreviated as $A \rightarrow P \rightarrow P \leftarrow A$.

Among the possible meta-paths that can be defined on a network schema, there are some that capture the similarity between two nodes. For example, the co-authorship meta-path $A \rightarrow P \leftarrow A$ in a bibliographic network creates a sense of similarity between two *Author* nodes. These type of meta-paths, called *similarity meta-paths*, are widely used to define topological features for link prediction problem in heterogeneous networks [20], [21], [13]. Table I presents a number of similarity meta-paths that can be defined on DBLP, Delicious, and MovieLens networks to capture the heterogeneous similarity between different node types.

The concept of similarity meta-paths can be extended to define heterogeneous features suitable for relationship prediction problem, where we have a target relation. Here we follow the same approach as in [9] which suggests the following three meta-path-based blocks to describe features for relationship prediction problem, given a target relation between two nodes of type A and B:

1) $A \overset{\text{similarity}}{\rightsquigarrow} A \overset{\text{target}}{\rightsquigarrow} B$
2) $A \overset{\text{target}}{\rightsquigarrow} B \overset{\text{similarity}}{\rightsquigarrow} A$
3) $A \overset{\text{relation}}{\rightsquigarrow} C \overset{\text{relation}}{\rightsquigarrow} B$

where $\rightsquigarrow$ denotes a meta-path, with labels *similarity* and *target* denoting a similarity meta-path and the target relation, respectively. The *relation* label denotes an arbitrary meta-path relating two nodes of possibly different types. The first block tells that there are some nodes of type A similar to a single node of the same type that has made the target relationship with a node of type B. Therefore, those similar nodes may also form the target relation with the type B node. An analogous intuition is behind the second block. For the third, it says that some nodes of type A are in relation with some type C nodes, which are themselves in relation with some nodes of type B. Hence, it is likely that type A nodes form some relationships, such as the target relationship, with type B nodes.

As an example in DBLP bibliographic network, for the target relation we use $A \rightarrow P \rightarrow P \leftarrow A$ as the meta-path denoting the author citation relation. In Addition, Paper-cite-Author ($P \rightarrow P \rightarrow A$) and Author-cite-Paper ($A \rightarrow P \rightarrow P$) are also used as the arbitrary relations, and the similarity meta-paths for DBLP network from Table I are used to define the features for author citation relationship prediction.

After specifying the suitable meta-paths, we need a method to quantify them as features. Here, due to the dynamicity of the network, different links are emerging and vanishing from the network over time. Therefore, the quantifying method must handle this dynamicity. Here, we formally define *Time-Aware Meta-Path-based Features*:

Definition 3 (Time-Aware Meta-Path-based Feature)**.** *Suppose that we are given a dynamic heterogeneous network G^τ along with its network schema $\mathcal{S}_G = (\mathcal{V}, \mathcal{E})$, and a target Relation $A \rightsquigarrow B$. For a given pair of nodes $a \in A$ and $b \in B$, and a meta-path $\Psi = v_1 \xrightarrow{\varepsilon_1} v_2 \xrightarrow{\varepsilon_2} \ldots v_{k-1} \xrightarrow{\varepsilon_{k-1}} v_k$ defined on $\mathcal{S}_G$, the time-aware meta-path-based feature at the timestamp τ is calculated as:*

$$f_\Psi^\tau(a,b) = I(a,A)I(b,B) \sum_{n_1 \in v_1, n_2 \in v_2, \ldots, n_k \in v_k} \prod_{i=1}^{k-1} I_\in\Big((n_i, n_{i+1}), \varepsilon_i\Big) I_<\left((n_i, n_{i+1}), \tau\right) \quad (1)$$

where:

$$I_\in(u, \mathcal{U}) = \begin{cases} 1 & \textit{if } u \in \mathcal{U} \\ 0 & \textit{otherwise} \end{cases}$$

checks whether a network entity u (node or link) belongs to type $\mathcal{U}$, and:

$$I_<(e, \tau) = \begin{cases} 1 & \textit{if } birth(e) < \tau \le death(e) \\ 0 & \textit{otherwise} \end{cases}$$

checks whether a link exists in the network at the timestamp τ. Here $birth(e)$ and $death(e)$ denotes the birth time and the death time of the link e, respectively.

By using the above definition, we will be able to quantify the number of instances of any particular meta-path at any specific timestamp. If we set this timestamp to the end of the feature extraction window, it is as though we are aggregating the whole network into a single snapshot observed at time $t_0 + \Phi$. In order to avoid such an aggregation, we divide the feature extraction window into a sequence of k contiguous intervals of a constant size Δ, as shown in Fig. 2. By doing such discretization, we intend to extract time-aware features in each sub-window which results in a multivariate time series containing the information about the temporal evolution of the topological features between any pair of nodes. With this in mind, we define *Dynamic Meta-Path-based Time Series* as follows:

Definition 4 (Dynamic Meta-Path-based Time Series)**.** *Suppose that we are given a dynamic heterogeneous network G^τ observed in a feature extraction window of size Φ ($t_0 < \tau \le t_0 + \Phi$), along with its network schema $\mathcal{S}_G = (\mathcal{V}, \mathcal{E})$ and a target relation $A \rightsquigarrow B$. Also suppose that the feature extraction window is divided into k fragments of size Δ. For a given pair of nodes $a \in A$ and $b \in B$ in $G^{t_0+\Phi}$, and a meta-path Ψ defined on $\mathcal{S}_G$, the dynamic meta-path-based time series of (a, b) is calculated as:*

$$x_\Psi^i(a,b) = f_\Psi^{t_0+i\Delta}(a,b) - f_\Psi^{t_0+(i-1)\Delta}(a,b) \qquad i = 1 \ldots k \quad (2)$$

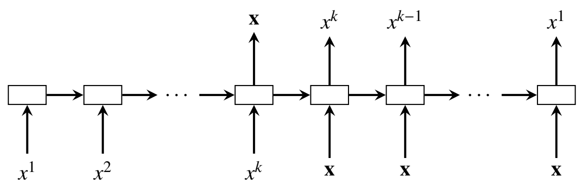


Fig. 3. The architecture of the LSTM Autoencoder used for dynamic feature extraction. The learned representation of the k^{th} stage is used as the dynamic feature $\mathbf{x}$.

For each unique meta-path designed using the triple building blocks described before, we get a unique time series. For each time step, we put the corresponding values from all time series into a vector. Consequently, we get a multivariate time series where each time step is vector-valued. For example, if we have d meta-paths Ψ_1 to Ψ_d, then each time step of the resulting time series will be of the form $\mathbf{x}^i = [x^i_{\Psi_1}, \dots, x^i_{\Psi_d}]^T$. Such multivariate time series reflect how topological features between two nodes change across different snapshots of the network. Based on the level of the network dynamicity, it can capture increasing/decreasing trends or even periodic/re-occurring patterns.

Now it's the time to convert this multivariate time series into a single feature vector so that we can use it as the input of our non-parametric model that is discussed in the next section. A trivial solution would be to stack all vectors of the multivariate time series into a single one, and feed our model with this single vector. However, this approach will result in a very high dimensional vector as the number of time steps increases, and can lead to difficulties in the learning procedure due to the curse of dimensionality. This is in contrast with our expectation that more time steps means more information about the history of the network and should result in a better prediction model. To overcome this problem, we combine the power of recurrent neural networks, especially Long Short Term Memory (LSTM) units [22], which have proven to be very successful in handling time series and sequential data, and autoencoders [23], which are widely used to learn alternative representations of the data such that the learned representation can reconstruct the original input.

Inspired by the work of Dai and Le on semi-supervised sequence learning [24], we design an LSTM autoencoder which takes a multivariate time series as input, and tries to encode it to a latent representation, so that it can then predict the input time series from the learned vector. The architecture of such autoencoder is illustrated in Fig 3. The benefits of using this autoencoder is two-fold: (1) since the autoencoder can reconstruct the original time series, which reflects the temporal dynamics of the network, we get minimum information loss in the learned vector; and (2) as we can set the dimensionality of the encoded vector to any desired value, we can evade the curse of dimensionality. We explain our proposed non-parametric model in the next section that takes this learned representation as the feature vector $\mathbf{x}$ and attempts to predict the corresponding time t.

IV. Proposed Non-Parametric Model

In this section we introduce our proposed model, called *Non-Parametric Generalized Linear Model*, to solve the problem of continuous-time relationship prediction based on the extracted features. Since the relationship building time is treated as a continuous random variable, we attempt to model the probability distribution of this time, given the features of the target relationship. Thus, if we denote the target relationship building time by t and its features by $\mathbf{x}$, our aim is to model the probability density function $f_T(t \mid \mathbf{x})$. A conventional approach to modeling this function is to fix a parametric distribution for t (e.g. Exponential distribution) and then relate $\mathbf{x}$ to t using a Generalized Linear Model [9]. The major drawback of this approach is that we need to know the exact distribution of the relationship building time, or at least, we could guess the best one that fits. The alternative way that we follow is to learn the shape of $f_T(t \mid \mathbf{x})$ from the data using a non-parametric solution.

In the rest of this section, we first bring the necessary theoretical backgrounds related to the concept, then we go through the details of the proposed model. At the end, we explain the learning and inference algorithms of Np-Glm.

A. Background

Here we define some essential concepts that are necessary to study before we proceed to the proposed model. Generally, the formation of a relationship between two nodes in a network can simply be considered as an event with its occurring time as a random variable T coming from a density function $f_T(t)$. Regarding this, we can have the following definitions:

Definition 5 (Survival Function)**.** *Given the density $f_T(t)$, the survival function denoted by $S(t)$, is the probability that an event occurs after a certain value of t, which means:*

$$S(t) = P(T > t) = \int_t^\infty f_T(t)dt \tag{3}$$

Definition 6 (Intensity Function)**.** *The intensity function (or failure rate function), denoted by $\lambda(t)$, is the instantaneous rate of event occurring at any time t given the fact that the event has not occurred yet:*

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{P(t \le T \le t + \Delta t \mid T \ge t)}{\Delta t} \tag{4}$$

Definition 7 (Cumulative Intensity Function)**.** *The cumulative intensity function, denoted by $\Lambda(t)$, is the area under the intensity function up to a point t:*

$$\Lambda(t) = \int_0^t \lambda(t)dt \tag{5}$$

The relations between density, survival, and intensity functions come directly from their definitions as follows:

$$\lambda(t) = \frac{f_T(t)}{S(t)} \tag{6}$$

$$S(t) = \exp(-\Lambda(t)) \tag{7}$$

$$f_T(t) = \lambda(t)\exp(-\Lambda(t)) \tag{8}$$

TABLE II
CHARACTERISTICS OF SOME PROBABILITY DISTRIBUTIONS USED FOR EVENT-TIME MODELING

Distribution	Density function $f_T(t)$	Survival function $S(t)$	Intensity function $\lambda(t)$	Cumulative intensity $\Lambda(t)$
Exponential	$\alpha\exp(-\alpha t)$	$\exp(-\alpha t)$	α	αt
Rayleigh	$\frac{t}{\sigma^2}\exp(-\frac{t^2}{2\sigma^2})$	$\exp(-\frac{t^2}{2\sigma^2})$	$\frac{t}{\sigma^2}$	$\frac{t^2}{2\sigma^2}$
Gompertz	$\alpha e^t \exp\{-\alpha(e^t-1)\}$	$\exp\{-\alpha(e^t-1)\}$	αe^t	αe^t
Weibull	$\frac{\alpha t^{\alpha-1}}{\beta^\alpha}\exp\left\{-(\frac{t}{\beta})^\alpha\right\}$	$\exp\left\{-(\frac{t}{\beta})^\alpha\right\}$	$\frac{\alpha t^{\alpha-1}}{\beta^\alpha}$	$(\frac{t}{\beta})^\alpha$

Table II shows the density, survival, intensity, and cumulative intensity functions of some widely-used distributions for event time modeling.

B. Model Description

Looking at Eq. 6, we see that the density function can be specified uniquely with its intensity function. Since the intensity function often has a simpler form than the density itself, if we learn the shape of the intensity function, then we can infer the entire distribution eventually. Therefore, we focus on learning the shape of the conditional intensity function $\lambda(t \mid \mathbf{x})$ from the data, and then accordingly infer the conditional density function $f_T(t \mid \mathbf{x})$ based on the learned intensity. In order to reduce the hypothesis space of the problem and avoid the curse of dimensionality, we assume that $\lambda(t \mid \mathbf{x})$, which is a function of both t and $\mathbf{x}$, can be factorized into two separate positive functions as the following:

$$\lambda(t \mid \mathbf{x}) = g(\mathbf{w}^T\mathbf{x})h(t) \tag{9}$$

where g is a function of $\mathbf{x}$ which captures the effect of features via a linear transformation using coefficient vector $\mathbf{w}$ independent of t, and h is a function of t which captures the effect of time independent from x. This assumption, referred to as proportional hazards condition [25], holds in GLM formulations of many event-time modeling distributions, such as the ones shown in Table II. Our goal is now to fix the function g and then learn both the coefficient vector $\mathbf{w}$ and the function h from the training data. In order to do so, we begin with the likelihood function of the data which can be written as follows:

$$\prod_{i=1}^{N} f_T(t_i \mid \mathbf{x}_i)^{y_i} P(T \geq t_i \mid \mathbf{x}_i)^{1-y_i} \tag{10}$$

The likelihood consists of the product of two parts: The first part is the contribution of those samples for which we have seen their exact building time, in terms of their density function. The second part on the other hand, is the contribution of the censored samples. For these samples, we use the probability of the building time being greater than the recorded one. Applying the Eq. 6, 7, and 9, the likelihood function becomes:

$$\prod_{i=1}^{N}\left[g(\mathbf{w}^T\mathbf{x}_i)h(t_i)\right]^{y_i}\exp\{-g(\mathbf{w}^T\mathbf{x}_i)\int_0^{t_i}h(t)dt\} \tag{11}$$

Since we don't know the form of $h(t)$, we cannot directly calculate the integral appeared in the likelihood function. To deal with this problem, we treat $h(t)$ as a non-parametric function by approximating it with a piecewise constant function that changes just in t_i s. Therefore, the integral over $h(t)$, denoted by $H(t)$, becomes a series:

$$H(t_i) = \int_0^{t_i} h(t)dt \simeq \sum_{j=1}^{i} h(t_j)(t_j - t_{j-1}) \tag{12}$$

assuming samples are sorted by t in increasing order, without loss of generality. The function $H(t)$ defined above plays an important role in both learning and inference phases. In fact, both the learning and inference phases rely on $H(t)$ instead of $h(t)$, which we will see later in this paper. Replacing the above series in the likelihood and taking the logarithm, we end up with the following log-likelihood function:

$$\begin{aligned}\log\mathcal{L} = \sum_{i=1}^{N}\Big\{&y_i\left[\log g(\mathbf{w}^T\mathbf{x}_i) + \log h(t_i)\right] \\ &- g(\mathbf{w}^T\mathbf{x}_i)\sum_{j=1}^{i}h(t_j)(t_j - t_{j-1})\Big\}\end{aligned} \tag{13}$$

The log-likelihood function depends on the vector $\mathbf{w}$ and the function $h(t)$. In the next part, we explain an iterative learning algorithm to learn both $\mathbf{w}$ and $h(t)$ collectively.

C. Learning Algorithm

Maximizing the log-likelihood function (Eq. 13) rely on the choice of the function g. There are no particular limits on the choice of g except that it must be a non-negative function. For example, both quadratic and exponential functions of $\mathbf{w}^T\mathbf{x}$ will do the trick. Here, we proceed with $g(\mathbf{w}^T\mathbf{x}) = \exp(\mathbf{w}^T\mathbf{x})$ since it makes the log-likelihood a convex function with respect to $\mathbf{w}$. Subsequent equations can be derived for other choices of g analogously.

Setting the log-likelihood derivative with respect to $h(t_k)$ to zero yields a closed form solution for $h(t_k)$:

$$h(t_k) = \frac{y_k}{(t_k - t_{k-1})\sum_{i=k}^{N}\exp(\mathbf{w}^T\mathbf{x}_i)} \tag{14}$$

Applying Eq. 12, we get the following for $H(t_i)$:

$$H(t_i) = \sum_{j=1}^{i}\frac{y_j}{\sum_{k=j}^{N}\exp(\mathbf{w}^T\mathbf{x}_k)} \tag{15}$$

Algorithm 1: The learning algorithm of Np-Glm

```
Input: $\mathbf{X}_{N\times d} = (\mathbf{x}_1, \dots \mathbf{x}_N)^T$ as $d$-dimensional feature vectors,
       $\mathbf{y}_{N\times 1}$ as observation states, and $\mathbf{t}_{N\times 1}$ as recorded times.
Output: Learned parameters $\mathbf{w}_{d\times 1}$ and $\mathbf{H}_{N\times 1}$.
$converged \leftarrow False$;
$threshold \leftarrow 10^{-4}$;
$\tau \leftarrow 0$;
$\log \mathcal{L}^{(\tau)} = -\infty$;
Initialize $\mathbf{w}^{(\tau)}$ with random values;
while Not converged do
    $\tau \leftarrow \tau + 1$;
    Use Eq. 15 to obtain $\mathbf{H}^{(\tau)}$ using $\mathbf{w}^{(\tau-1)}$;
    Optimize Eq. 16 to obtain $\mathbf{w}^{(\tau)}$ using $\mathbf{H}^{(\tau)}$;
    Use Eq. 13 to obtain $\log \mathcal{L}^{(\tau)}$ using $\mathbf{w}^{(\tau)}$ and $\mathbf{H}^{(\tau)}$;
    if $\left\|\log \mathcal{L}^{(\tau)} - \log \mathcal{L}^{(\tau-1)}\right\| < threshold$ then
        $converged \leftarrow True$;
    end
end
$\mathbf{w} \leftarrow \mathbf{w}^{(\tau)}$;
$\mathbf{H} \leftarrow \mathbf{H}^{(\tau)}$;
```

which depends on the vector $\mathbf{w}$. On the other hand, we cannot obtain a closed form solution for $\mathbf{w}$ from the log-likelihood function. Therefore, we turn to use Gradient-based optimization methods to find the optimal value of $\mathbf{w}$. The negative log-likelihood function with respect to $\mathbf{w}$, denoted by $NL(\mathbf{w})$ is as follows:

$$NL(\mathbf{w}) = \sum_{i=1}^{N} \left\{ \exp(\mathbf{w}^T \mathbf{x}_i) H(t_i) - y_i \mathbf{w}^T \mathbf{x}_i \right\} \tag{16}$$

which depends on the function H. As the learning of both $\mathbf{w}$ and H depends on each other, they should be learned collectively. Here, we use an iterative algorithm to learn $\mathbf{w}$ and H alternatively. We begin with a random vector $\mathbf{w}^{(0)}$. Then in each iteration τ, we first update $H^{(\tau)}$ via Eq. 15 using $w^{(\tau-1)}$. Next, we optimize Eq. 16 using the values of $H^{(\tau)}(t_i)$ to obtain $\mathbf{w}^{(\tau)}$. We continue this routine until convergence. The pseudo code of the learning procedure is available in Algorithm 1.

D. Inference Queries

In this part, we explain how to answer the common inference queries based on the inferred distribution $f_T(t \mid \mathbf{x})$. Suppose that we have learned the vector $\mathbf{w}$ and the function H using the training samples $(\mathbf{x}_i, y_i, t_i), i = 1 \dots N$ following Algorithm 1. Afterwards, for a testing relationship R associated with a feature vector $\mathbf{x}_R$, the following queries can be answered:

Ranged Probability. What is the probability for the relationship R to be formed between time t_α and t_β? This is equivalent to calculating $P(t_\alpha \le T \le t_\beta \mid \mathbf{x}_R)$, which by definition is:

$$\begin{aligned} P(t_\alpha \le T \le t_\beta \mid \mathbf{x}_R) &= S(t_\alpha \mid \mathbf{x}_R) - S(t_\beta \mid \mathbf{x}_R) \\ &= \exp\{-g(\mathbf{w}^T \mathbf{x}_R) H(t_\alpha)\} - \exp\{-g(\mathbf{w}^T \mathbf{x}_R) H(t_\beta)\} \end{aligned} \tag{17}$$

The problem here is to obtain the values of $H(t_\alpha)$ and $H(t_\beta)$, as t_α and t_β may not be among t_is of the training samples, for which H is estimated. To calculate $H(t_\alpha)$, we find $k \in \{1, 2, \dots, N\}$ such that $t_k \le t_\alpha < t_{k+1}$. Due to the piecewise constant assumption for the function h, we get:

$$h(t_\alpha) = \frac{H(t_\alpha) - H(t_k)}{t_\alpha - t_k} \tag{18}$$

On the other hand, since h only changes in t_is, we have:

$$h(t_\alpha) = h(t_{k+1}) = \frac{H(t_{k+1}) - H(t_k)}{t_{k+1} - t_k} \tag{19}$$

Combining Eq. 18 and 19, we get:

$$H(t_\alpha) = H(t_k) + (t_\alpha - t_k) \frac{H(t_{k+1}) - H(t_k)}{t_{k+1} - t_k} \tag{20}$$

Following the similar approach, we can calculate $H(t_\beta)$, and then answer the query using Eq. 17. The dominating operation here is to find the value of k. Since we have t_is sorted beforehand, this operation can be done using a binary search with $O(\log N)$ time complexity.

Quantile. By how long the target relationship R will be formed with probability α? This question is equivalent to find the time t_α such that $P(T \le t_\alpha \mid x_R) = \alpha$. By definition, we have:

$$1 - P(T \le t_\alpha \mid \mathbf{x}_R) = S(t_\alpha \mid \mathbf{x}_R) = \exp\{-g(\mathbf{w}^T \mathbf{x}_R) H(t_\alpha)\} = 1 - \alpha$$

Taking logarithm of both sides and rearranging, we get:

$$H(t_\alpha) = -\frac{\log(1 - \alpha)}{g(\mathbf{w}^T \mathbf{x}_R)} \tag{21}$$

To find t_α, we first find k such that $H(t_k) \le H(t_\alpha) < H(t_{k+1})$. We eventually have $t_k \le t_\alpha < t_{k+1}$ since H is a non-decreasing function due to non-negativity of the function h. Therefore, we again end up with Eq. 20, which by rearranging we get:

$$t_\alpha = (t_{k+1} - t_k) \frac{H(t_\alpha) - H(t_k)}{H(t_{k+1}) - H(t_k)} + t_k \tag{22}$$

By combining the Eq. 21 and 22, we can obtain the value of t_α which is the answer to the quantile query. It worth mentioning that if $\alpha = 0.5$ then t_α becomes the median of the distribution $f_T(t \mid \mathbf{x}_R)$. Here again the dominant operation is to find the value of k, which due to the non-decreasing property of the function H can be found using a binary search with $O(\log N)$ time complexity.

V. Evaluations on Synthetic Data

We use synthetic data to verify the correctness of Np-Glm and its learning algorithm. Since Np-Glm is a non-parametric method, we generate synthetic data using various parametric models with previously known random parameters, and evaluate how well Np-Glm can learn the parameters and the underlying distribution of the generated data.

Algorithm 2: Synthetic dataset generation algorithm.

```
Input: The number of observed samples $N_o$, the number of
       censored samples $N_c$, the dimension of the feature
       vectors $d$, and the desired distribution $dist$
Output: Synthetically generated data $\mathbf{X}_{N\times d}$, $\mathbf{y}_{N\times 1}$, and $\mathbf{t}_{N\times 1}$.
$N \leftarrow N_o + N_c$;
Draw a weight vector $\mathbf{w} \sim \mathcal{N}(0, \mathbf{I}_d)$, where $\mathbf{I}_d$ is the
  $d$-dimensional identity matrix;
Draw scalar intercept $b \sim \mathcal{N}(0, 1)$;
for $i \leftarrow 1$ to $N$ do
    Draw feature vector $\mathbf{x}_i \sim \mathcal{N}(0, \mathbf{I}_d)$;
    Set distribution parameter $\alpha_i \leftarrow \exp(\mathbf{w}^T\mathbf{x}_i + b)$;
    if $dist == Rayleigh$ then
        Draw $t_i \sim \alpha_i\ t \exp\{-0.5\alpha_i t^2\}$;
    else if $dist == Gompertz$ then
        Draw $t_i \sim \alpha_i\ e^t \exp\{-\alpha_i(e^t - 1)\}$;
end
Sort pairs $(\mathbf{x}_i, t_i)$ by $t_i$ in ascending order;
for $i \leftarrow 1$ to $N_o$ do
    $y_i \leftarrow 1$;
end
for $i \leftarrow (N_o + 1)$ to $N$ do
    $y_i \leftarrow 0$;
end
```

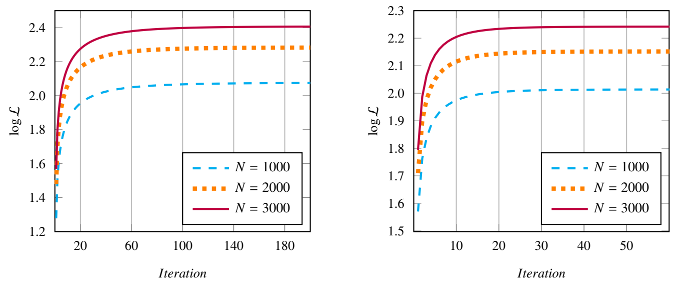
A. Experiment Setup

We consider generalized linear models of two widely used distributions for event-time modeling, Rayleigh and Gompertz, as the ground truth models for generating synthetic data. Algorithm 2 is used to generate a total of N data samples with d-dimensional feature vectors, consisting N_o non-censored (observed) samples and remaining $N_c = N - N_o$ censored ones. For all synthetic experiments, we generate 10-dimensional feature vectors ($d = 10$) and set $g(\mathbf{w}^T\mathbf{x}) = \exp(\mathbf{w}^T\mathbf{x})$. We repeat every experiment 100 times and report the average.

B. Experiment Results

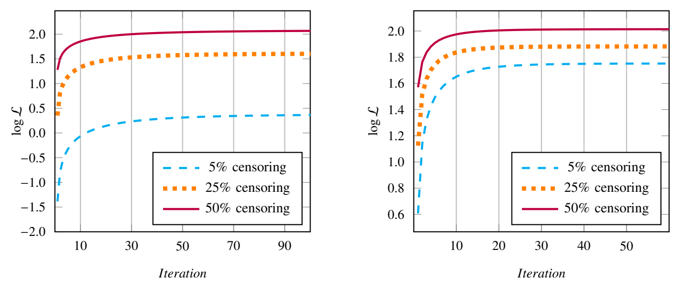
Since Np-Glm's learning is done in an iterative manner, we first analyzed whether this algorithm converges as the number of iterations increase. We recorded the log-likelihood of Np-Glm, averaged over the number of training samples N in each iteration. We repeated this experiment for $N \in \{1000, 2000, 3000\}$ with a fixed censoring ratio of 0.5, which means half of the samples are censored. The result is depicted in Fig. 4. We can see that the algorithm successfully converges with a rate depending on the underlying distribution. For the case of Rayleigh, it requires about 100 iterations to converge but for Gompertz, this reduces to about 30. Also, we see that using more training data results in achieving more log-likelihood as expected.

In Fig. 5, we fixed $N = 1000$ and performed the same experiment this time using different censoring ratios. According to the figure, we see that by increasing the censoring ratio, the convergence rate increases. This is because Np-Glm infers the values of $H(t)$ for all t in the observation window. Therefore, as the censoring ratio increases, the observation window is decreased, so Np-Glm has to infer a fewer number of parameters, leading to a faster convergence. Note that as opposed to



(a) Rayleigh distribution (b) Gompertz distribution

Fig. 4. Convergence of Np-Glm's average log-likelihood (log $\mathcal{L}$) for different number of training samples (N). Censoring ratio has been set to 0.5.



(a) Rayleigh distribution (b) Gompertz distribution

Fig. 5. Convergence of Np-Glm's average log-likelihood (log $\mathcal{L}$) for different censoring ratios with 1K samples.

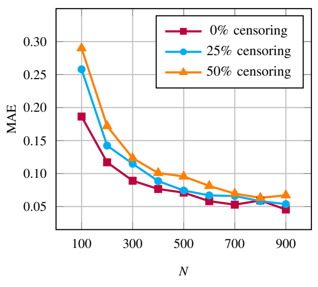
Fig. 4, here a higher log-likelihood doesn't necessarily indicate a better fit, due to the likelihood marginalization we get by the censored samples.

Next, we evaluated how good Np-Glm can infer the parameters used to generate the synthetic data. To this end, we varied the number of training samples N and measured the mean absolute error (MAE) between the learned weight vector $\hat{\mathbf{w}}$ and the ground truth. Fig. 6 illustrates the result for different censoring ratios. It can be seen that as the number of training samples increases, the MAE gradually decreases. The other point to notice is that more censoring ratio results in higher error due to the information loss we get by censoring.

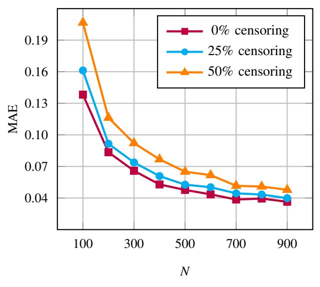
Finally, we investigated whether censored samples are informative or not. For this purpose, we fixed the number of observed samples N_o and changed the number of censored samples from 0 to 200. We measure the MAE between the learned $\mathbf{w}$ and the ground truth for $N_o \in \{200, 300, 400\}$. The result is shown in Fig. 7. It clearly demonstrates that adding more censored samples causes the MAE to dwindle up to an extent, after which we get no substantial improvement. This threshold is dependent on the underlying distribution. In this case, for Rayleigh and Gompertz it is about 80 and 120, respectively.

VI. Experiments on Real-World Data

We apply Np-Glm with the proposed feature set on a number of real-world datasets to evaluate its effectiveness and compare its performance in predicting the relationship building time

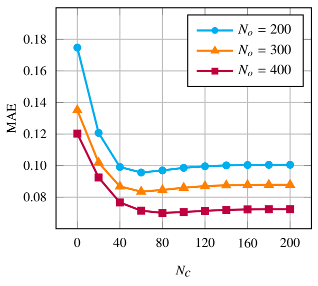


(a) Rayleigh distribution

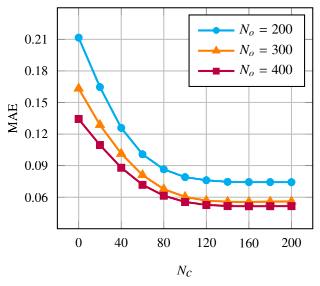


(b) Gompertz distribution

Fig. 6. Np-Glm's mean absolute error (MAE) vs the number of training samples (N) for different censoring ratios.



(a) Rayleigh distribution



(b) Gompertz distribution

Fig. 7. Np-Glm's mean absolute error (MAE) vs the number of censored samples (N_c) for different number of observed samples (N_o).

vis-à-vis state-of-the-art models.

A. Dataset

1) DBLP: We use DBLP bibliographic citation network, provided by [26], which has both attributes of dynamicity and heterogeneity. The network contains four types of objects: authors, papers, venues, and terms. The network schema of this dataset is depicted in Fig 1a. Each paper is associated with a publication date, with a granularity of one year. Based on the publication venue of the papers, we limited the original DBLP dataset to those papers that are published in venues relative to the theoretical computer science. This will result in about 37k papers ranging from 1969 to 2016, published in 38 venues.

2) Delicious: Another dynamic and heterogeneous dataset we use in our experiments is the Delicious bookmarking dataset from [27], with a network schema presented in Fig 1c. It contains three types of objects, namely users, bookmarks, and tags. The dataset includes bookmarking timestamps from May 2006 to October 2010. For the sake of convenience, we convert the scale of time durations from timestamp to month.

3) MovieLens: The third heterogeneous dataset with dynamic characteristics has been extracted from MovieLens personalized movie recommendations website, provided by [28]. The dataset comprises seven types of objects, that are users, movies, tags, genres, actors, directors, and countries, as illustrated by the network schema in the Fig 1b. It contains user-movie rating timestamps ranging from September 1997 to January 2009. Similar to Delicious dataset, timestamp differences are converted to months.

TABLE III
Demographic Statistics of Real-World Datasets

Dataset	Time Span	Entity	Title	Count
DBLP	From 1969 to 2016	Nodes	*Author*	15,929
			Paper	37,077
			Venue	38
			Term	12,028
		Links	write	100,797
			cite	165,904
			publish	42,872
			mention	284,156
Delicious	From May 2006 to Oct 2010	Nodes	*User*	1,714
			Tag	21,956
			Bookmark	30,998
		Links	contact	15,329
			post	437,594
			has-tag	437,594
MovieLens	From Sep 1997 to Jan 2009	Nodes	*User*	1,421
			Movie	5,660
			Actor	6,176
			Director	2,401
			Genre	19
			Tag	5,561
			Country	63
		Links	rate	855,599
			play-in	231,743
			direct	10,156
			has-genre	20,810
			has-tag	47,958
			produced-in	10,198

B. Experiment Settings

1) Comparison Methods: To challenge the performance of Np-Glm, we use the state of the art generalized linear model-based framework proposed in [9] with Exponential, Rayleigh, and Weibull as distributions with different shapes, denoted as Exp-Glm, Ray-Glm, and Wbl-Glm, respectively. To examine the effect of considering the dynamicity of the network on the performance of the models, we evaluate each one with two different feature sets: *dynamic* and *static*. Dynamic feature set is extracted using our proposed features extraction framework and captures the dynamicity of the network. On the contrary, static feature set only uses the very last snapshot of the network just before the beginning of the observation window, failing to reflect its temporal dynamics. For all models, we consider the median of the distribution $f_T(t \mid \mathbf{x}_{test})$ as the predicted time for any test sample and then compare it to the ground truth time t_{test}.

2) Performance Measures: We assess different methods using a number of evaluation metrics which are described in the following:

- Mean Absolute Error (MAE): this measures the expected absolute error between the predicted time values and the ground truth, and is computed as:

$$MAE(\mathbf{t}, \hat{\mathbf{t}}) = \frac{1}{N} \sum_{i=1}^{N} \left| t_i - \hat{t}_i \right|$$

- Mean Relative Error (MRE): this measures the expected relative absolute error between the predicted time values and the ground truth, and is computed as:

$$MRE(\mathbf{t}, \hat{\mathbf{t}}) = \frac{1}{N} \sum_{i=1}^{N} \left| \frac{t_i - \hat{t}_i}{t_i} \right|$$

- Root Mean Squared Error (RMSE): this measures the root of the expected squared error between the predicted time values and the ground truth, and is computed as:

$$RMSE(\mathbf{t}, \hat{\mathbf{t}}) = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (t_i - \hat{t}_i)^2}$$

- Mean Squared Logarithmic Error (MSLE): this measures the expected value of the squared logarithmic error between the predicted time values and the ground truth, and is computed as:

$$RMSE(\mathbf{t}, \hat{\mathbf{t}}) = \frac{1}{N} \sum_{i=1}^{N} (\log(1 + t_i) - log(1 + \hat{t}_i))^2$$

- Median Absolute Error (MDAE): this measures the median of the absolute errors between the predicted time values and the ground truth, and is computed as:

$$MDAE(\mathbf{t}, \hat{\mathbf{t}}) = median(|t_1 - \hat{t}_1| \dots |t_N - \hat{t}_N|)$$

- Concordance Index (CI): this metric is one of the most widely used performance measures for survival models that estimates how good the model performs at ranking predicted times [29]. It can be seen as the fraction of all pairs of samples whose predicted times are correctly ordered among all samples that can be ordered, and is considered as the generalization of the Area Under Receiver Operating Characteristic Curve (AUC) when we are dealing with censored data [30].
- Maximum Threshold Prediction Accuracy (ACC): this measures for what fraction of samples, a model have a lower absolute error than a given threshold:

$$ACC(\mathbf{t}, \hat{\mathbf{t}}) = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\left(|t_i - \hat{t}_i| < threshold\right)$$

3) Experiment Setup: For DBLP dataset, we confine the data samples to those authors who have published more than 5 papers in the feature extraction window of each experiment. Following the triple building blocks described for feature extraction in Section III, and using the similarity meta-paths in Table I, we start the feature extraction process with 19 meta-paths. In all experiments, the author citation relation ($A \rightarrow P \rightarrow P \leftarrow A$) is chosen as the target relation. For the case of Delicious dataset, we select user-user relation ($U \leftrightarrow U$) as the target relation, and design 6 meta-paths via the similarity meta-paths in Table I. Regarding the MovieLens dataset, we limit the actor list to the top three for each movie. To imply a notion of "like" relation between user and movie, we only consider ratings above 4 in scale of 5. For this dataset, the target relation is set to user rate movie ($U \rightarrow M$), based on which, we design 11 final meta-paths.

TABLE IV
PERFORMANCE COMPARISON OF DIFFERENT METHODS ON DIFFERENT DATASETS

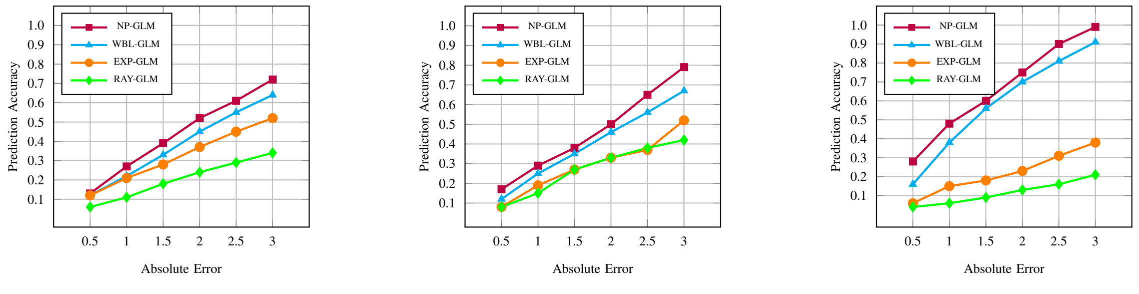
Dataset	Feature	Model	MAE	MRE	RMSE	MSLE	MDAE	CI
DBLP	Dynamic	NP-GLM	**1.99**	**0.95**	**2.43**	**0.30**	**1.73**	**0.62**
		WBL-GLM	2.33	1.10	2.85	0.36	2.08	0.58
		EXP-GLM	3.11	1.39	3.88	0.52	2.58	0.50
		RAY-GLM	4.02	1.83	4.70	0.66	3.72	0.35
	Static	NP-GLM	2.76	1.35	3.07	0.44	2.88	0.26
		WBL-GLM	2.81	1.38	3.16	0.45	2.88	0.48
		EXP-GLM	3.28	1.57	3.70	0.53	3.30	0.14
		RAY-GLM	5.04	2.28	5.26	0.85	5.12	0.01
Delicious	Dynamic	NP-GLM	**2.10**	**1.20**	**2.55**	**0.35**	**2.05**	**0.70**
		WBL-GLM	2.37	1.31	2.89	0.40	2.16	0.57
		EXP-GLM	3.21	1.58	3.84	0.54	2.89	0.55
		RAY-GLM	3.90	2.07	4.66	0.68	3.91	0.40
	Static	NP-GLM	2.33	1.46	2.80	0.41	2.17	0.61
		WBL-GLM	2.65	1.62	3.23	0.47	2.26	0.43
		EXP-GLM	3.35	1.91	4.17	0.59	2.75	0.35
		RAY-GLM	4.81	2.61	5.27	0.85	4.28	0.12
MovieLens	Dynamic	NP-GLM	**2.48**	**3.08**	**3.04**	**0.55**	**2.14**	**0.70**
		WBL-GLM	3.06	3.61	3.79	0.65	2.60	0.56
		EXP-GLM	3.79	2.70	4.60	0.78	3.48	0.45
		RAY-GLM	4.98	3.58	5.63	1.05	4.83	0.33
	Static	NP-GLM	2.92	3.44	3.45	0.67	3.36	0.50
		WBL-GLM	2.99	3.52	3.51	0.69	3.37	0.49
		EXP-GLM	3.42	2.89	3.86	0.78	3.82	0.49
		RAY-GLM	5.32	4.06	5.62	1.17	5.70	0.20

We implemented the LSTM autoencoder using Keras deep learning library [31]. We used mean square error loss function and Adadelta optimizer [32] with default parameters. For all datasets, we set the dimension of the encoded feature as twice as the input dimension. For Np-Glm the data samples were ordered according to their corresponding time variables, as the model needs the samples sorted by their recorded time. In all experiments, we pick an equal number of censored samples as the observed ones, uniformly at random. We use 5-fold cross-validation and report the average results for all the experiments in this section.

C. Experiment Results

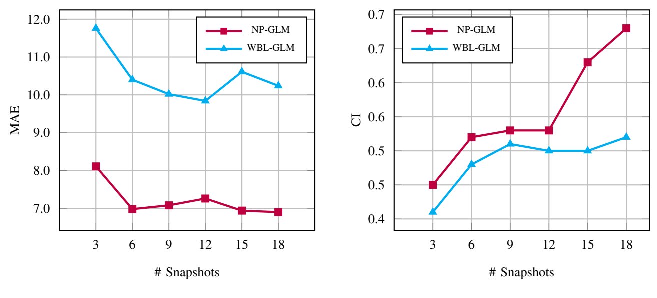
In the rest of this section, we first assess how well different methods perform on various datasets and compare their performance based on different measures. Next, we analyze the effect of different parameters and problem configurations on the performance of competitive methods.

Comparative Performance Analysis. In the first set of experiments, we evaluated the prediction power of different models on DBLP, Delicious and MovieLens datasets. For all configurations, we set $\Delta = 1$ and $\Omega = 6$. For DBLP dataset, the number of snapshots k, was set to 6, while for the other two datasets we set $k = 12$. MAE, MRE, RMSE, MSLE, MDAE and CI of all models using both dynamic and static feature sets has been shown in Table IV. We see that in all three networks, NP-GLM using the dynamic features is superior to the other models under all performance measures. For instance, our model NP-GLM can obtain an MAE of 1.99 for the DBLP dataset, which is 15% lower than the MAE obtained by its closest competitor, WBL-GLM. As of CI, NP-GLM achieves 0.62 on DBLP, which is 7% better than WBL-GLM. On Delicious



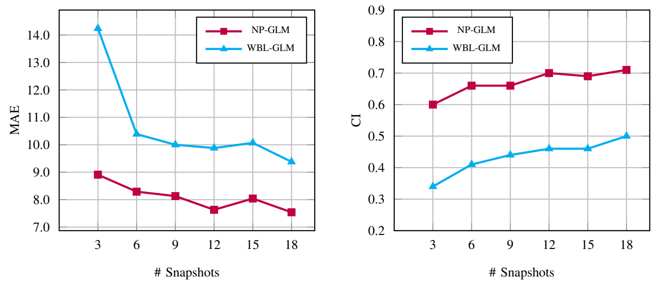
(a) DBLP (b) Delicious (c) MovieLens

Fig. 8. Prediction accuracy of different methods vs the maximum tolerated absolute error on different datasets.



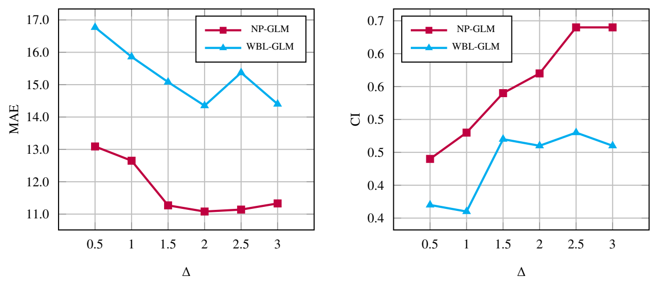
(a) Mean Absolute Error (b) Concordance Index

Fig. 9. Effect of choosing different number of snapshots on performance of different methods using Delicious dataset.



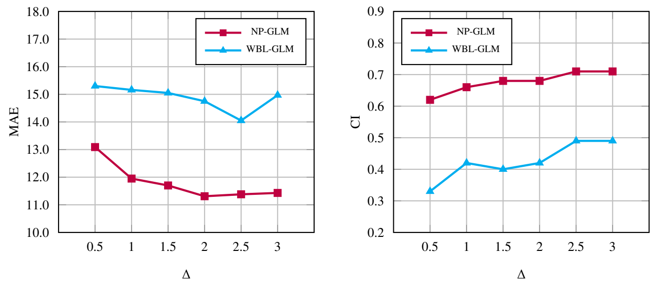
(a) Mean Absolute Error (b) Concordance Index

Fig. 10. Effect of choosing different number of snapshots on performance of different methods using MovieLens dataset.



(a) Mean Absolute Error (b) Concordance Index

Fig. 11. Effect of choosing different values for Δ on performance of different methods using Delicious dataset.



(a) Mean Absolute Error (b) Concordance Index

Fig. 12. Effect of choosing different values for Δ on performance of different methods using MovieLens dataset.

dataset, NP-GLM improves MAE and CI by 11% and 23%, respectively, relative to WBL-GLM. Similarly, NP-GLM reduces MAE by 19% and increases CI by 25%. Comparable results hold for other performance measures as well. Moreover, in this table it is evident that using the dynamic features has a positive impact on the performance of all models.

In the next experiment, we investigated the performance of different methods using the dynamic feature set under maximum threshold prediction accuracy. In other words, to evaluate the prediction accuracy of a model, we record the fraction of test samples for which the difference between their true times and the predicted ones are lower than a given threshold, called *tolerated error*. The parameter settings for feature extraction (Δ, Ω, and k) is the same as the configuration in Table IV. The results for different tolerated errors in range $\{0.5, 1.0, \ldots, 3.0\}$ were plotted in Fig 8. The result demonstrates that NP-GLM can achieve a higher accuracy in all cases relative to other baselines.

Parameter Setting Analysis. The performance of different models is influenced by two parameters, the number of snapshots k, and the time difference between snapshots Δ as these parameters determine the length of the feature extraction window Φ. In this set of experiments, we investigate how these parameters affect the performance of our model NP-GLM and its closest competitor WBL-GLM on Delicious and MovieLens datasets.

The effect of increasing the number of snapshots on achieved MAE and CI by NP-GLM and WBL-GLM using Delicious and MovieLens datasets is illustrated in Fig 11 and Fig 12, respectively. For both datasets, we set $\Delta = 1.5$ and $\Omega = 18$ and changed the number of snapshots in the range of 3 to 18. As we can see in both figures, increasing the number of snapshots results in lower prediction error and higher accuracy. This is due to the fact that as the number of snapshots grows, a longer history of the network is taken into account.

Finally, the impact of choosing different values for Δ is analyzed on the performance of Np-Glm and Wbl-Glm in terms of MAE and CI. The results for Delicious and MovieLens datasets are depicted in Fig 9 and Fig 10, respectively. In this experiment, number of snapshots and observation window length are accordingly set to 6 and 24. Different values of Δ are selected from the set $\{0.5, 1.0, \ldots, 3.0\}$. As illustrated in both figures, by increasing Δ up to an extent, we witness that the performance of models improves gradually. That is because increasing the value of Δ leads to a wider feature extraction window. However, since the number of snapshots is constant, we see no performance improvement when the value of Δ becomes greater than a certain threshold. This is due to that fact that short term temporal evolution of the network will be neglected when the value of Δ is too wide.

VII. Related Works

The problem of link prediction has been studied extensively in recent years and many approaches have been proposed to solve this problem [33], [34]. Previous works on time-aware link prediction have mostly considered temporality in analyzing the long-term network trend over time [35]. Authors in [10] have shown that temporal metrics are an extremely valuable new contribution to link prediction, and should be used in future applications. Dunlavy *et al.* focused on the problem of periodic temporal link prediction [36]. They focused on bipartite graphs that evolve over time and also considered weighted matrix that contained multilayer data and tensor-based methods for predicting future links. Oyama *et al.* solved the problem of cross-temporal link prediction, in which the links among nodes in different time frames are inferred [37]. They mapped data objects in different time frames into a common low-dimensional latent feature space, and identified the links on the basis of the distance between the data objects. Özcan *et al.* proposed a novel link prediction method for evolving networks based on NARX neural network [38]. They take the correlation between the quasi-local similarity measures and temporal evolutions of link occurrences information into account by using NARX for multivariate time series forecasting. Yu *et al.* developed a novel temporal matrix factorization model to explicitly represent the network as a function of time [39]. They provided results for link prediction as an specific example and showed that their model performs better than the state-of-the-art techniques.

The most relevant works to this study are available in [14], [16], [17], [9]. The Authors in [14] approaches the problem of time series link prediction by extracting simple temporal features from the time series, such as mean, (weighted) moving average, and exponential smoothing besides some topological features like common neighbor and adamic-adar. But their method is designed for homogeneous networks and fail to consider the heterogeneity of the modern networks. Aggarwal *et al.* [16] tackle the link prediction problem in both dynamic and heterogeneous information networks using a dynamic clustering approach alongside with content-based and structural models. However they aim to solve the conventional link prediction problem, not the continuous-time relationship prediction problem studied in this paper. In [17], the authors proposed a feature set, called TMLP, well suited for link prediction in dynamic and heterogeneous information networks. Although their proposed feature set cope with both dynamicity and heterogeneity of the network, it cannot be extended for the generalized problem of relationship prediction and is only designed for solving the simpler link prediction problem.

Most of the aforementioned works answered the question of *whether* a link will appear in the network. To the best of our knowledge, the only work that has focused on the continuous-time relationship prediction problem is proposed by Sun *et al.* [9], in which a generalized linear model based framework is suggested to model the relationship building time. They consider the building time of links as independent random variables coming from a pre-specified distribution and model the expectation as a function of a linear predictor of the extracted topological features. A shortcoming of this model is that we need to exactly specify the underlying distribution of times. We came over this problem by learning the distribution from the data using a non-parametric solution. Furthermore, we considered the temporal dynamics of the network which has been entirely ignored in their work.

VIII. Conclusion

In this paper, we studied the problem of continuous-time relationship prediction in both dynamic and heterogeneous information networks. To effectively tackle this problem, we first introduced a novel feature extraction framework based on meta-path modeling and recurrent neural network autoencoders to systematically extract features suitable that takes both the temporal dynamics and heterogeneous characteristics of the network into account for solving the continuous-time relationship problem. We then proposed a supervised non-parametric model, called Np-Glm, which exploits the extracted features to predict the relationship building time in information networks. The strength of our model is that it does not impose any significant assumption on the underlying distribution of the relationship building time given its features, but tries to infer it from the data via a non-parametric approach. Extensive experiments conducted on synthetic dataset and real-world datasets from DBLP, Delicious, and MovieLens demonstrated the correctness of our method and its effectiveness in predicting the relationship building time.

References

[1] L. Lü and T. Zhou, "Link prediction in complex networks: A survey," *Physica A: Statistical Mechanics and its Applications*, vol. 390, no. 6, pp. 1150–1170, 2011.

[2] S. Wasserman and K. Faust, *Social network analysis: Methods and applications*. Cambridge university press, 1994, vol. 8.

[3] D. Liben-Nowell and J. Kleinberg, "The link prediction problem for social networks," *journal of the Association for Information Science and Technology*, vol. 58, no. 7, pp. 1019–1031, 2007.

[4] X. Chen, M.-X. Liu, and G.-Y. Yan, "Drug–target interaction prediction by random walk on the heterogeneous network," *Molecular BioSystems*, vol. 8, no. 7, pp. 1970–1978, 2012.

[5] C. Wang, V. Satuluri, and S. Parthasarathy, "Local probabilistic models for link prediction," in *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*. IEEE, 2007, pp. 322–331.

[6] R. N. Lichtenwalter, J. T. Lussier, and N. V. Chawla, "New perspectives and methods in link prediction," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010, pp. 243–252.

[7] B. Taskar, M.-F. Wong, P. Abbeel, and D. Koller, "Link prediction in relational data," in *Advances in neural information processing systems*, 2004, pp. 659–666.

[8] C. Shi, Y. Li, J. Zhang, Y. Sun, and S. Y. Philip, "A survey of heterogeneous information network analysis," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 1, pp. 17–37, 2017.

[9] Y. Sun, J. Han, C. C. Aggarwal, and N. V. Chawla, "When will it happen?: Relationship prediction in heterogeneous information networks," in *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, ser. WSDM '12. New York, NY, USA: ACM, 2012, pp. 663–672.

[10] A. Potgieter, K. A. April, R. J. Cooke, and I. O. Osunmakinde, "Temporality in link prediction: Understanding social complexity," *Emergence: Complexity and Organization*, vol. 11, no. 1, p. 69, 2009.

[11] Y. Dong, J. Tang, S. Wu, J. Tian, N. V. Chawla, J. Rao, and H. Cao, "Link prediction and recommendation across heterogeneous social networks," in *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE, 2012, pp. 181–190.

[12] D. Davis, R. Lichtenwalter, and N. V. Chawla, "Multi-relational link prediction in heterogeneous information networks," in *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*. IEEE, 2011, pp. 281–288.

[13] S. Sajadmanesh, H. R. Rabiee, and A. Khodadadi, "Predicting anchor links between heterogeneous social networks," in *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, Aug 2016, pp. 158–163.

[14] A. Hajibagheri, G. Sukthankar, and K. Lakkaraju, "Leveraging network dynamics for improved link prediction," in *Social, Cultural, and Behavioral Modeling: 9th International Conference, SBP-BRiMS 2016, Washington, DC, USA, June 28-July 1, 2016, Proceedings 9*. Springer, 2016, pp. 142–151.

[15] B. Moradabadi and M. R. Meybodi, "A novel time series link prediction method: Learning automata approach," *Physica A: Statistical Mechanics and its Applications*, vol. 482, pp. 422–432, 2017.

[16] C. Aggarwal, Y. Xie, and P. S. Yu, "On dynamic link inference in heterogeneous networks," in *Proceedings of the 2012 SIAM International Conference on Data Mining*. SIAM, 2012, pp. 415–426.

[17] N. Sett, S. Basu, S. Nandi, and S. R. Singh, "Temporal link prediction in multi-relational network," *World Wide Web*, pp. 1–25, 2017.

[18] H. Kwak, C. Lee, H. Park, and S. Moon, "What is twitter, a social network or a news media?" in *Proceedings of the 19th international conference on World wide web*. ACM, 2010, pp. 591–600.

[19] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks," *Proceedings of the VLDB Endowment*, vol. 4, no. 11, pp. 992–1003, 2011.

[20] Y. Sun, R. Barber, M. Gupta, C. C. Aggarwal, and J. Han, "Co-author relationship prediction in heterogeneous bibliographic networks," in *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*. IEEE, 2011, pp. 121–128.

[21] J. Zhang, P. S. Yu, and Z.-H. Zhou, "Meta-path based multi-network collective link prediction," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 1286–1295.

[22] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[23] Y. Bengio *et al.*, "Learning deep architectures for ai," *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[24] A. M. Dai and Q. V. Le, "Semi-supervised sequence learning," in *Advances in Neural Information Processing Systems*, 2015, pp. 3079–3087.

[25] N. E. Breslow, "Analysis of survival data under the proportional hazards model," *International Statistical Review / Revue Internationale de Statistique*, vol. 43, no. 1, pp. 45–57, 1975.

[26] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "Arnetminer: Extraction and mining of academic social networks," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '08. New York, NY, USA: ACM, 2008, pp. 990–998.

[27] I. Cantador, P. Brusilovsky, and T. Kuflik, "2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011)," in *Proceedings of the 5th ACM conference on Recommender systems*, ser. RecSys 2011. New York, NY, USA: ACM, 2011.

[28] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 4, pp. 19:1–19:19, Dec. 2015.

[29] F. E. Harrell Jr, R. M. Califf, D. B. Pryor, K. L. Lee, R. A. Rosati *et al.*, "Evaluating the yield of medical tests," *Jama*, vol. 247, no. 18, pp. 2543–2546, 1982.

[30] H. Steck, B. Krishnapuram, C. Dehing-oberije, P. Lambin, and V. C. Raykar, "On ranking in survival analysis: Bounds on the concordance index," in *Advances in neural information processing systems*, 2008, pp. 1209–1216.

[31] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[32] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[33] P. Wang, B. Xu, Y. Wu, and X. Zhou, "Link prediction in social networks: the state-of-the-art," *Science China Information Sciences*, vol. 58, no. 1, pp. 1–38, 2015.

[34] T. Wang and G. Liao, "A review of link prediction in social networks," in *Management of e-Commerce and e-Government (ICMeCG), 2014 International Conference on*. IEEE, 2014, pp. 147–150.

[35] Y. Dhote, N. Mishra, and S. Sharma, "Survey and analysis of temporal link prediction in online social networks," in *Advances in Computing, Communications and Informatics (ICACCI), 2013 International Conference on*. IEEE, 2013, pp. 1178–1183.

[36] D. M. Dunlavy, T. G. Kolda, and E. Acar, "Temporal link prediction using matrix and tensor factorizations," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 5, no. 2, p. 10, 2011.

[37] S. Oyama, K. Hayashi, and H. Kashima, "Cross-temporal link prediction," in *Data Mining (ICDM), 2011 IEEE 11th International Conference on*. IEEE, 2011, pp. 1188–1193.

[38] A. Özcan and Ş. G. Öğüdücü, "Temporal link prediction using time series of quasi-local node similarity measures," in *Machine Learning and Applications (ICMLA), 2016 15th IEEE International Conference on*. IEEE, 2016, pp. 381–386.

[39] W. Yu, C. C. Aggarwal, and W. Wang, "Temporally factorized network modeling for evolutionary network analysis," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 2017, pp. 455–464.