Social Network Analysis and Mining (2018) 8:16
https://doi.org/10.1007/s13278-018-0494-1

ORIGINAL ARTICLE

Predicting *interval time* for reciprocal link creation using survival analysis

Vachik S. Dave[1] · Mohammad Al Hasan[1] · Baichuan Zhang[2] · Chandan K. Reddy[3]

Received: 15 March 2017 / Revised: 11 January 2018 / Accepted: 14 February 2018

Abstract

The majority of directed social networks, such as Twitter, Flickr and Google+, exhibit *reciprocal altruism*, a social psychology phenomenon, which drives a vertex to create a reciprocal link with another vertex which has created a directed link toward the former. In existing works, scientists have already predicted the possibility of the creation of reciprocal link—a task known as "reciprocal link prediction". However, an equally important problem is determining the *interval time* between the creation of the first link (also called parasocial link) and its corresponding reciprocal link. No existing works have considered solving this problem, which is the focus of this paper. Predicting the reciprocal link *interval time* is a challenging problem for two reasons: First, there is a lack of effective features, since well-known link prediction features are designed for undirected networks and for the binary classification task; hence, they do not work well for the *interval time* prediction; Second, the presence of *ever-waiting* links (i.e., parasocial links for which a reciprocal link is not formed within the observation period) makes the traditional supervised regression methods unsuitable for such data. In this paper, we propose a solution for the reciprocal link *interval time* prediction task. We map this problem to a survival analysis task and show through extensive experiments on real-world datasets that survival analysis methods perform better than traditional regression, neural network-based models and support vector regression for solving reciprocal *interval time* prediction.

Keywords Link prediction · Directed network · Reciprocity · Time prediction · Survival analysis

1 Introduction

Reciprocity is a phenomenon in social psychology which mandates that people should repay voluntarily what another person has provided for them. It is different from *altruism* (Anand et al. 2013) in the way that reciprocity follows from others' initial action, while altruism is a spontaneous action of gift-giving without the hope or expectation of future positive responses. There also exists another social psychology, named *reciprocal altruism*, which is a behavior whereby one performs an act of gift-giving with the expectation that the receiving person will act in a similar manner at a later time (Trivers 1971). People's day-to-day activities on online social networks are filled with many examples of reciprocal altruism: we follow a friend's Twitter feed with the hope that he will follow back our feed; we like a friend's Facebook posts or her Flicker images with the expectation that she will do the same; we endorse our friends for their technical skill in LinkedIn hoping that they will return the favor in a similar manner.

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s13278-018-0494-1) contains supplementary material, which is available to authorized users.

✉ Vachik S. Dave
vsdave@iupui.edu

Mohammad Al Hasan
alhasan@cs.iupui.edu

Baichuan Zhang
baichuan24@fb.com

Chandan K. Reddy
reddy@cs.vt.edu

[1] Department of Computer and Information Science, IUPUI, Indianapolis, USA

[2] Facebook, Menlo Park, USA

[3] Department of Computer Science, Virginia Tech, Arlington, USA

Published online: 09 March 2018

However, reciprocity usually is in conflict with another social phenomenon called *social stratification*, which favors hierarchical arrangement of people in a society based on various factors such as power, wealth and reputation (Hopcroft et al. 2011). This phenomenon is prevalent in online social networks as well, but in a different manner. Apparently, for such networks, the social hierarchy is reflected in various prestige metrics which rank vertices based on their topological bearings, such as PageRank and in-degree. Given this hierarchical arrangement in an online social network, people who are higher up in the hierarchy are sometimes reluctant to perform a reciprocal act for an individual who is lower in the hierarchy; they instead defer the reciprocal action to a later time, or sometimes indefinitely.

For reciprocal link creation, understanding the criteria which control the *interval time* and building learning models which predict the *interval time* are important. From a research standpoint, such studies help scientists to understand the interaction between reciprocity and social stratification phenomena. From the perspective of real-life applications in social network analysis, such prediction models enable better link suggestions, where the *interval time* is also factored in within the suggestion. Reciprocity, along with the *interval time* for reciprocal link creation, is particularly important for recommendation in online dating systems (Xia et al. 2015).

The majority of existing works on link prediction assume an undirected network (Hasan and Zaki 2011; Valverde-Rebaza and de Andrade Lopes 2013), in which the concept of reciprocal edges does not exist. A few works consider reciprocal link prediction (Hopcroft et al. 2011; Gong and Xu 2014) in a directed network where the prediction is binary, yielding a yes/no answer to the question of whether a reciprocal link will be created within a fixed observation window. Several other works utilize reciprocity as a tool for network compression (Chierichetti et al. 2009) and information propagation in social networks (Zhu et al. 2014). Reciprocal links also influence the degree correlations in complex networks; hence, they play an important part in modeling the growth of directed social networks (Zlatić and Štefančić 2009). However, none of the existing works consider predicting the *interval time* for the creation of a reciprocal edge.

Extending a model which solves a binary class reciprocal link prediction problem to a model which predicts the *interval time* of reciprocal links is non-trivial. The major challenge for *interval time* prediction is that typical link prediction features for an undirected network, such as common neighbors, Jaccard's similarity and Adamic–Adar, do not have a well-defined counterpart for directed networks, which makes interval prediction a difficult task. Additionally, for generating the training data for building a prediction model, a network is observed for a finite time window, and the absence of a reciprocal link within that time window does not necessarily mean the absence of that reciprocal edge, because a reciprocal edge might have formed outside (after) the observation time window. This yields numerous right-censored data instances, for which the target variable, i.e., the reciprocal link formation time is not available. Traditional supervised regression models cannot include censored data instances in the training data and hence perform poorly in predicting reciprocal link creation time.

We explain the cases of right-censored data instances in reciprocal *interval time* prediction task using a toy example shown in Fig. 1. In this figure we show a small part of an email communication network consisting of only three vertices representing three persons, A, B and C. Our observation period of this network has five timestamps, $T1$ to $T5$. At $T1$, C sends an email to B, thus creating the first of the directed links (such links are called parasocial links). At T_2, the parasocial link from A to B is created. At $T3$, the reciprocal link from B to C is created; thus, the *interval time* of this edge is $T3 - T1$. At $T3$, another parasocial link ($B \rightarrow C$) is created. More links are created in subsequent time intervals $T4$ and $T5$. At $T5$, we reach the end of our observation period, but the reciprocal link from C to A is yet to be created. The potential reciprocal link $C \rightarrow A$ is an instance of right-censored data for which we only know that the *interval time* is higher than $T5 - T1$; this value, as well, can be infinity in the case that the link is never created. Either way, the exact value of the target variable for this reciprocal edge is unknown. Unfortunately, for any reasonable observation time window, a significantly large number of potential reciprocal links are censored data instances, which is the main challenge for the task of reciprocal link creation time prediction.

In this work, we present a supervised learning model for predicting the *interval time* for the creation of a reciprocal edge between a pair of vertices in an online social network, given that a parasocial edge already exists between the vertex pair. We study real-life networks and validate a collection of topological features that may influence the reciprocal edge creation time. Then, we design the prediction task as a survival analysis problem and propose five censored regression models. Our experimental results show that Cox regression performs better than traditional supervised learning models for reciprocal link prediction. This is an extended version of our previous paper (Dave et al. 2017), which is published in 11th International Conference on Web and Social Media (ICWSM).

2 Related works

The traditional binary classification task of link prediction has received enormous attention over the years since the inception of this problem by Liben-Nowell and Kleinberg (2003). Over the years researchers have solved the link

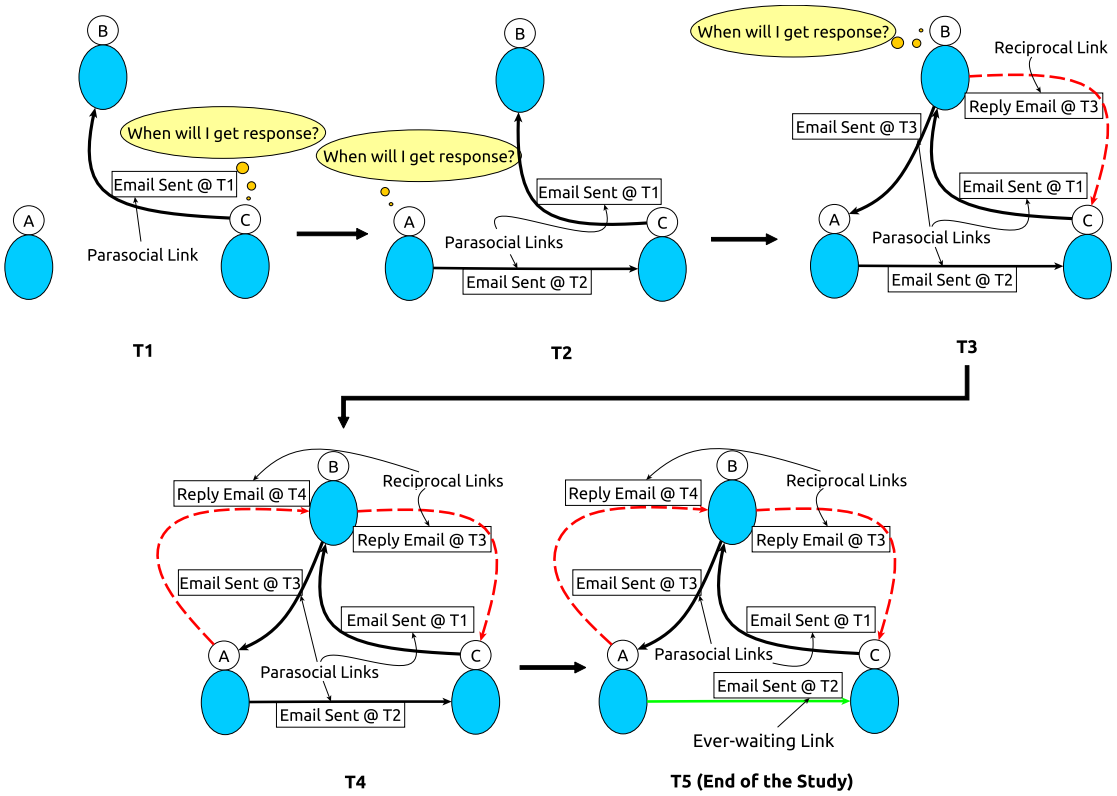

Fig. 1 An illustration of reciprocal link time prediction *RLTP* problem

prediction problem for a variety of graphs—for example link prediction in homogeneous networks (Hasan et al. 2006; Liaghat et al. 2013; Wang et al. 2017b), link prediction in heterogeneous information networks (Sun et al. 2011; Dong et al. 2012) and link prediction for knowledge graphs (Dong et al. 2014; Zhang et al. 2016). Other related problems, such as link/sign prediction and ranking in signed social network (Song and Meyer 2015; Symeonidis and Mantas 2013) and a recommendation system using link prediction techniques (Esslimani et al. 2011), have also been studied.

Reciprocal link prediction is a variant of link prediction which works on directed networks. Even though the majority of social and communication graphs are directed, only a few works exist which consider predicting reciprocal links. In one of the earliest works, Hopcroft et al. (2011) predicted reciprocal edges in a Twitter network. However, many of the features that they proposed are too specific to the Twitter dataset and do not apply to a generic directed network. Gong and Xu (2014) compared reciprocal and parasocial link creation in Google+ and Flickr datasets and solved the reciprocal link prediction problem as an outlier detection task using one-class SVM. Authors of (Cheng et al. 2011) compared structural differences of reciprocal links and parasocial links, and they also studied a Twitter dataset and corresponding node features to predict reciprocal links. In another work (Feng et al. 2014), the authors reported that the majority of reciprocating links are created within a very short time after the creation of corresponding parasocial links. Dumba et al. (2016) studied the structural properties of a reciprocal network and discussed user behavior patterns.

A closely related problem to reciprocal link prediction is online dating recommendation. There exist a few works that solve this problem, mainly by using traditional recommendation methods with novel feature extraction processes. For example, in (Zhao et al. 2014) the authors modified the classical collaborative filtering method for the dating recommendation task. Xia et al. (2015, 2016) proposed different reciprocal score matrices and used them with collaborative filtering for recommendation. The authors in (Tu et al. 2014) proposed an LDA (latent Dirichlet allocation)-based approach to learn latent preferences of users with two side matching-based recommendation. Recently, Zang et al. (2017) proposed a method that extracts profile-based features, topological features and preference features from a dating social network for recommendation. All the existing works discussed so far target the binary classification problem, which predicts whether the reciprocal link will be created or not. On the other hand, our work targets the prediction of time, which is more difficult than the binary classification problem.

To the best of our knowledge, there are only two works that target the time prediction problem; the first one is by Sun et al. (2012) and the second by Li et al. (2016). In both of these works, the authors have extracted unique features for a DBLP-like (author paper) heterogeneous network. Y. Sun et al. proposed meta-path-based topological features and used a generalized linear model (GLM) for the prediction task. Similarly, M. Li et al. proposed a novel time difference labeled path (TDLP)-based method for the knowledge graph. Both methods are designed specifically for DBLP-like networks; hence, they are difficult to apply to other networks. On the other hand, our method is applicable to any general directed network to predict time of a reciprocal link.

3 Our methodology

In this section, we first define the problem of *reciprocal link time prediction* (*RLTP*). Then we present some insight of three real-world datasets that we have used in this work. Then we explain how the *RLTP* can be solved by using a survival analysis framework. After that we discuss different survival analysis methods which we have used for solving the *RLTP* problem. Finally, we provide algorithmic representation of the proposed framework.

3.1 Problem formulation

Definition 1 (*Directed time-stamped network*) Consider a network $G(V, E)$, where V is the set of vertices and E is the set of directed edges. T is a set of time values, and τ is a mapping function, which maps an edge to one of the time values in the set T, i.e., $\tau : E \rightarrow T$. For an edge $e \in E, t_e \in T$ denotes the creation time of the edge e. Collectively, G, T and τ are called a directed time-stamped network. □

For vertices $u, v \in V$ and link $e = (u, v) \in E$ the corresponding time stamp t_e can be represented as t_{uv}. If an edge e is created multiple times, we keep only the oldest (earliest) creation time and assign that to t_e. For a vertex $u \in V$, $\Gamma_{in}(u)$ and $\Gamma_{out}(u)$ are the set of in-neighbors and the set of out-neighbors of u, and $d(u, v)$ is the directed shortest path distance from u to v.

Definition 2 (*Reciprocal/Parasocial link*) For a pair of vertices, u and v, the edge $(u, v) \in E$ is called a parasocial link if the edge $(v, u) \notin E$. On the other hand, if $(v, u) \in E$ and $(u, v) \in E$, and $t_{vu} < t_{uv}$ then (u, v) is called a reciprocal link. □

The objective of the *RLTP* problem is to predict the time of a reciprocal link for the given parasocial link with time. The *interval time* for a reciprocal link (u, v) is defined as $Int(u, v) = t_{uv} - t_{vu}$. Our model for the *RLTP* problem actually predicts $Int(u, v)$, instead of predicting t_{uv} (the reciprocal link creation time). Nevertheless, the reciprocal link creation time t_{uv} can be obtained from the model by using the expression $t_{vu} + Int(u, v)$. The advantage of predicting $Int(u, v)$ instead of predicting t_{uv} is that for predicting $Int(u, v)$ we do not need to use the parasocial link creation time t_{vu} as part of input of the model, which makes the model independent of temporal bias. Thus, the supervised model of our proposed *RLTP* task uses only the topological features of an edge (u, v) as its covariates and the *interval time* $Int(u, v)$ as its target variable, making the model simple.

3.2 Dataset study

In this section, we discuss the datasets that we use in our study. We also provide some statistical analysis of the datasets; specifically, for each of these datasets, we provide the empirical distribution of observed *interval time* and its goodness of fit with known statistical distributions. For the *Enron* dataset, the persons (along with their rank in the company) associated with a vertex is known, so in this dataset we have also performed a qualitative study by checking for the evidences of social stratification phenomenon, which we present at the end of this section.

We used three real-world directed network datasets for our study. We selected datasets where reciprocal link creation is an important (meaningful) event; another selection criterion is that the datasets should have a sufficient number of reciprocal links to train and test the models (Kuhnt and Brust 2014). Our first dataset, *Epinion* is a trust network where a directed link from one vertex to another represents the fact that the former trusts the latter. The *RLTP* task for this dataset is to find the time at which a trusted person acknowledges that (s)he also holds a similar sentiment toward the other person. The dataset was collected from KONECT web page.[1] We have also collected two email datasets: *MC-Email*[2] and *Enron*. Both of these datasets are email communication networks from two distinct enterprises, and for these datasets the *RLTP* task is to predict the response time for an email. More information on these datasets is provided in Table 1, where $|V|$, $|E|$, $|T|$ and *Recipro* are the number of vertices, the number of edges, the number of timestamps (in days) and the reciprocity of the dataset within the observation window, respectively.

For these three datasets, we plot the histogram of the *interval time* for reciprocal links in log scale (Fig. 2). We observed that the majority of the responses are received within a short period of time (within 10 days or less).

[1] http://konect.uni-koblenz.de/networks/.

[2] This is Manufacturing Company email dataset available from R. Michalski's website, https://www.ii.pwr.edu.pl/~michalski.

Table 1 Basic statistics of the datasets used in the paper

Dataset	$\|V\|$	$\|E\|$	$\|T\|$	*Recipro*
Epinion	131, 828	841, 373	938	0.3083
MC-Email	167	5, 783	237	0.876
Enron	182	3, 007	944	0.6053

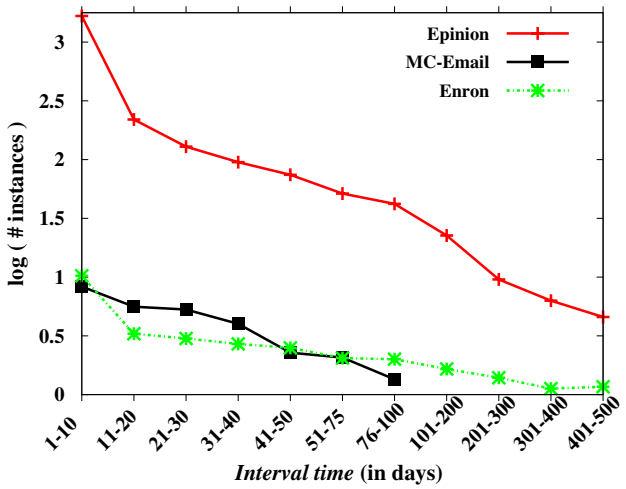

Fig. 2 Histogram of *interval time* of reciprocal link

However, there also exist a few late responders whose reply time is much larger than the average reply time.

3.2.1 Modeling *interval time* using parametric distribution

From the distribution plots in Fig. 2, we observe that the number of reciprocal link instances reduces exponentially with the increment of the *interval time* (note that, y-axis is in log scale). Hence, we fit different exponential family distributions to model the time interval of reciprocal link for all three datasets. Specifically, we fit exponential distribution, normal distribution, logistic distribution, log-normal distribution, log-logistic distribution and Weibull distribution. To evaluate the goodness of fit we use the following four metrics: Kolmogorov–Smirnov (KS) statistic, Cramer–von Mises (CM) statistic, Akaike's information criterion (AIC) (Akaike 1998) and Bayesian information criterion (BIC) (Schwarz 1978). In Fig. 3, we show the quality of fitting results. The results of BIC are very similar to AIC for all three datasets, so we did not show the results of BIC. As depicted in Fig. 3, exponential, normal and logistic distributions (shown in red) have relatively high distance from empirical distribution compared to log-normal, log-logistic and Weibull distributions (shown in black). For the *Enron* dataset, Weibull distribution performs the best over all metrics. Similarly, for the *Epinion* and the *MC-Email* datasets

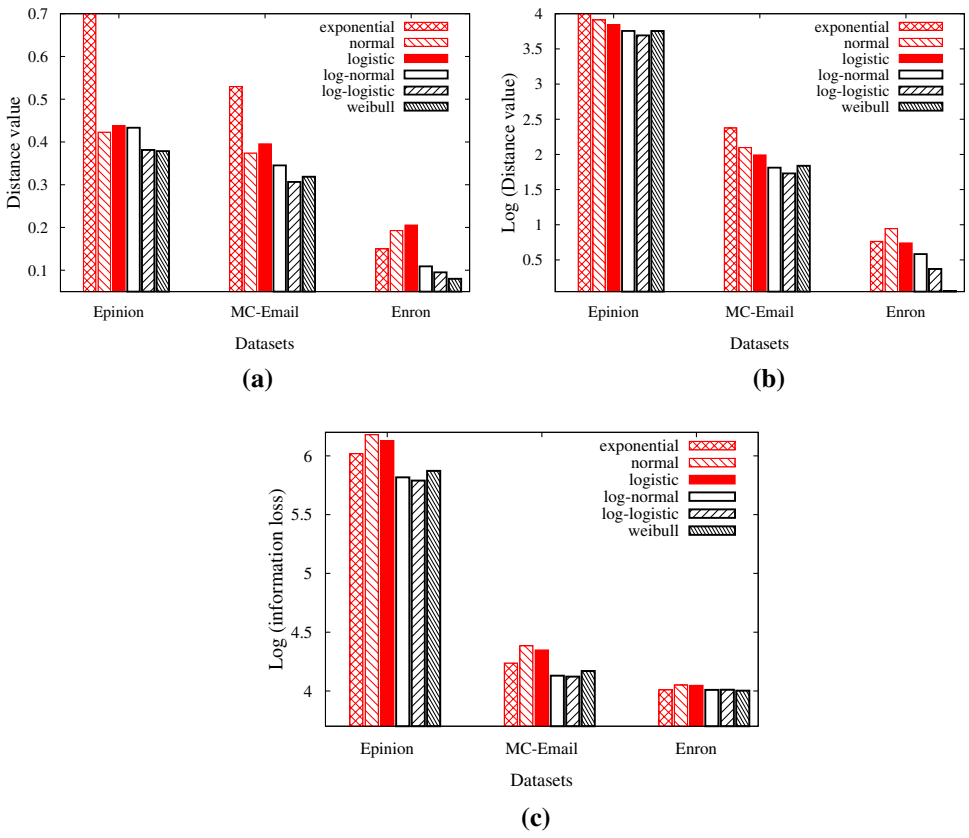

Fig. 3 Goodness of fit comparisons for different distributions. **a** Kolmogorov-Smirnov statistic. **b** Cramer–von Mises statistic. **c** Akaike's information criterion

log-logistic distribution fits the best. Results of log-normal distribution are very similar to both Weibull and log-logistic distributions. Hence, we use log-normal, log-logistic and Weibull distributions for parametric survival models, which are discussed later in Sect. 3.5.

3.2.2 Social stratification in Enron

One of the influencing factors for late responses to a specific user is social stratification—particularly in corporations, people tend to give quicker replies to their superior as compared to their colleagues and other juniors. We study the *Enron* dataset, for which the employee details are available with email communications. In the dataset, "Louise Kitchen" is a president; we observed that her email replying practice follows social stratification phenomenon. She generally takes more than 2–3 days to reply to people with lower ranking positions such as vice-president (VP), employees, etc. For example, she replied to VPs "Kevin Presto", "James Steffes" and "Fletcher Sturm" in 3, 6 and 19 days, respectively. She replied to "Sally Beck" (Chief Operating Officer) in 5 days. On the other hand, she replied to "David Delainey" (Chief Executive Officer (CEO)) on the same (0) day. Another example is "Philip Allen", who is a manager; he replied within a day to higher ranking officers such as "David Delainey" (CEO), "Barry Tycholiz" (VP), "Hunter Shively" (VP) and "Richard Shapiro" (VP). On the other hand, he took 2 to 3 days to reply to "Michael Grigsby" (manager), "Jay Reitmeyer" (employee) and "Matthew Lenhart" (employee).

3.3 Topological feature study

In online social networks, user behavior-based features are useful for solving different problems, such as link prediction (Valverde-Rebaza and de Andrade Lopes 2013), personality prediction (Adalı and Golbeck 2014), user attribute prediction (Tuna et al. 2016), link sign prediction (Shahriari et al. 2016), prediction of positive and negative users in Twitter (Roshanaei and Mishra 2015), etc. Hence, we believe social (behavioral) phenomena-based topological features can contribute substantially to solve the *RLTP* problem. Though there are works that study and design user behavior features such as topic-specific modeling (Bogdanov et al. 2014), a behavioral model for Facebook wall posts (Devineni et al. 2017), etc., we assume to have only topological information. Topological features that we use come from two different social phenomena: directed altruism and social stratification. Below we discuss them in two different sections.

3.3.1 Directed altruism-based features

Directed altruism in social networks is described in Leider et al. (2007), where the authors have argued that people are more generous to friends and friends of friends than to a complete stranger. This phenomenon also reflects in people's reciprocal link creation behavior. Below, we define some topological features which quantify the directed altruism phenomena for reciprocal link prediction.

Shortest directed distance: In our problem, one directional link (v, u) already exists, and we are predicting the creation time for the reverse link (u, v). Generally people are more generous to indirect friends than complete strangers. Hence, u is more likely to respond quickly to v for small value of the directed distance from u to v, i.e.,

$$DirectedDist(u, v) = d(u, v)$$

Common in/out neighbors count: The number of common neighbors is a frequently used topological feature for the link prediction task in undirected networks; however, for directed graphs, we have two separate features: common in-neighbors and common out-neighbors. Both of these topological features capture the idea that if a user has more common neighbors with another user, then she is more likely to reply fast. Also, more common friends increase the network flow, which is an important factor for building trust (Leider et al. 2007) and with higher trust people tend to reply faster.

$$Common_{in}(u, v) = |\Gamma_{in}(u) \cap \Gamma_{in}(v)|$$
$$Common_{out}(u, v) = |\Gamma_{out}(u) \cap \Gamma_{out}(v)|$$

Jaccard coefficient (*In/Out*): The Jaccard coefficient is another widely used topological feature for undirected networks. It is the normalized version of common neighbors counts. Similar to the common neighbor count feature, this feature also split into two features due to the directed-ness of the edges. Jaccard coefficients help to predict the trust level between two nodes. Since, higher trust leads to faster response, this is a good feature for the *RLTP* task.

$$Jaccard_{in} = \frac{|\Gamma_{in}(u) \cap \Gamma_{in}(v)|}{|\Gamma_{in}(u) \cup \Gamma_{in}(v)|}$$
$$Jaccard_{out} = \frac{|\Gamma_{out}(u) \cap \Gamma_{out}(v)|}{|\Gamma_{out}(u) \cup \Gamma_{out}(v)|}$$

Local reciprocity: In (Gong and Xu 2014), the authors studied two local reciprocity features and they showed relative influence of both features on linking back probability. The first is *acceptance local reciprocity* (*ALR*), which is defined as:

$$ALR(v) = \frac{|\Gamma_{in}(v) \cap \Gamma_{out}(v)|}{|\Gamma_{in}(v)|}$$

We compute *ALR* for the head node (v) of the reciprocal link (u, v). This feature captures the tendency of node v to accept a link. The second feature is *request local reciprocity* (*RLR*), defined as:

$$RLR(u) = \frac{|\Gamma_{in}(u) \cap \Gamma_{out}(u)|}{|\Gamma_{out}(u)|}$$

We compute *RLR* for the tail node (u) of the reciprocating link (u, v). *RLR* represents the response behavior of the node u and captures its tendency to initiate a reciprocal link.

3.3.2 Social stratification-based features

It is observed that in online social networks people behave according to their status in the network (Hopcroft et al. 2011). A similar behavior is observed in many real-world applications, such as the one described in Sect. 3.2 or in online dating (Xia et al. 2013). We have also shown evidence of social stratification in *Enron* dataset, specifically in connection to the *RLTP* task. The following topological features quantify the extent of social stratification that is practiced by the node u or v.

Preferential attachment: This feature computes a value which reflects the social stratification induced rank order of a given node. The basic idea of preferential attachment is to give more weight to the higher degree nodes. Traditionally, preferential attachment has been computed for undirected networks, so we change the formula to adapt it for directed networks. For undirected graph, it is simply the product of the degrees of the node u and v. For directed graph, we take the product of the out-degree of the tail node (u) and the in-degree of the head node (v) of a prospective reciprocal link (u, v). The formula is given below:

$$PrefAtt(u, v) = |\Gamma_{out}(u)| \times |\Gamma_{in}(v)|$$

Preferential Jaccard (*PrefJacc*) is inspired by both preferential attachment and Jaccard coefficient. It is a trade-off between two concepts—first, high degree nodes are prone to create more edges, and second, nodes prefer to connect with similar nodes (social stratification). Both these phenomena can influence reciprocal edge creation. We calculated *PrefJacc* by using the following equation:

$$PrefJacc(u, v) = \frac{|\Gamma_{out}(u) \cap \Gamma_{in}(v)|}{|\Gamma_{out}(u) \cup \Gamma_{in}(v)|}$$

In/out ratio: A node in the upper hierarchy has a tendency to a create reciprocal edges with other nodes at the same hierarchy level than to nodes which are at a lower hierarchy level (Hopcroft et al. 2011). To reflect this knowledge in our model, we need to find an efficient way for comparing the hierarchy of a pair of nodes, which we compute by the ratio of their in-degrees and the ratio of their out-degrees. Higher *InRatio* is indicative of higher tendency of the numerator node to attract links compared to the denominator node; similarly, higher *OutRatio* represents a higher tendency of the numerator node to create links compared to the denominator node. In this way, these two features capture the relative patterns of link creation and link acceptance by the pair of the vertices. For reciprocating link (u, v), we calculate *InRatio* and *OutRatio* by using the following equations:

$$InRatio = \frac{|\Gamma_{in}(u)|}{|\Gamma_{in}(v)|}$$

$$OutRatio = \frac{|\Gamma_{out}(u)|}{|\Gamma_{out}(v)|}$$

PageRank represents the prestige of the node in the network. We use both, PageRank of u and PageRank of v as features. If $PageRank(u)$ is lower than $PageRank(v)$, then the node u is highly likely to respond faster to the node v.

Table 2 Correlation of features with *interval time*

Features/datasets	*Epinion*	*MC-Emails*	*Enron*
DirectedDist	− 0.04127	− 0.03792	− 0.13336
$Common_{In}$	0.38109	0.33447	0.44398
$Common_{Out}$	0.27254	0.31194	0.27534
$Jaccard_{In}$	0.17161	0.22101	0.24831
$Jaccard_{Out}$	0.11015	0.18925	0.20195
RLR(u)	− 0.00290	0.05820	0.16053
ALR(v)	− 0.06093	0.15383	0.19256
PrefAtt	0.19289	0.23930	0.25443
PrefJacc	0.09136	0.20054	0.25502
InRatio	− 0.03165	− 0.07053	− 0.14302
OutRatio	− 0.01132	0.04269	0.13108
PageRank(u)	0.24783	− 0.07523	− 0.07609
PageRank(v)	0.14300	0.00211	0.02049

3.3.3 Feature analysis

To validate the strength of these features (13 in total) for predicting the *interval time* of reciprocal edges, we compute the Pearson's correlation of the above topological features with the *interval time* value for three real-life graph datasets (Table 1) and show the correlation values in Table 2. As we can see, for the *MC-Emails* dataset most of the features (mainly $Common_{in}$, $Common_{out}$, $Jaccard_{In}$, $Jaccard_{Out}$, *PrefAtt* and *PrefJacc*) have good correlation value (between 0.2 and 0.5). Similarly, for the *Enron* dataset the same set of features is highly related to *interval time*. But, for *Epinion* dataset the correlation values for most of the features are poor except for $Common_{in}$, $Common_{out}$, and *PageRank(u)*; the worst features are *InRatio*, *OutRatio* and *RLR(u)*. To check the influence of these features on reciprocal link creation, we also check the average linking back probability over

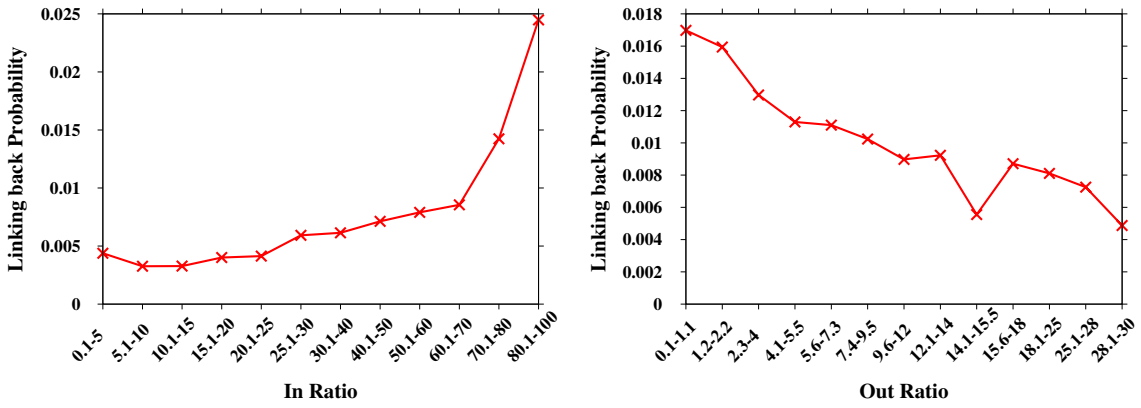

Fig. 4 Relation of *In/OutRatio* and linking back probability in *Epinion* dataset

different range of values for different features. We discuss our observation in the following paragraphs.

In Fig. 4, we plot our observation for two of the features: *InRatio* and *OutRatio*. Here, for each bin of *InRatio*, the linking back probability is calculated as a fraction of reciprocal links over all the links in that bin. Figure 4 clearly shows high linking back probability for higher *InRatio* and lower *OutRatio*, which is expected behavior for these features. In (Gong and Xu 2014), the authors provided a thorough study of some features, such as *RLR(u)* and *ALR(v)*, and proved their significant influence on reciprocal link creation.

In Fig. 5, we show three plots (one for each dataset) of *DirectedDist* vs. *interval time*. Within each plot we have several graphs, each representing the directed distance value between the vertices. Along the x-axis is the *interval time* and along the y-axis is the number of reciprocal link instances that have the corresponding *interval time*. For all dataset, we observe that links with small directed distance value (such as 2 or 3) can have high *interval time*, i.e., the reciprocal link may appear after many days; but as distance increases there are few or almost no instances of reciprocal links with high *interval time*. This observation may appear counterintuitive as we expect short distance to influence a short *interval time*. However, this observation can be explained as follows: people tend to trust other people who are within their circles, and they will ultimately create a reciprocal links with them, even if they do not do it immediately. On the other hand, for people who are outside someone's circle (having a high directed distance value, such as 4 or 5), reciprocal links will be created either in a short *interval time* or will not be created at all. The short *interval time* can be the cases when two strangers meet in-person in a social event and then mutually agree to be connected online (or trust each other). On the other hand, the negative case happens, when a stranger trusts (or sends an invite to) someone, and the second person just ignore that forever. Due to this complex relation, the correlation between directed distance and *interval time* is poor, yet we consider *DirectedDist* to be a useful feature.

3.3.4 Correlation with low and high *interval time*

There are a variety of different social behaviors that influence the *interval time*; hence, some social-based features impact the *interval time* differently over a period. To understand the impact of different features over a period, we split the target variable (*interval time*) into lower and higher range and calculate feature correlations with lower and higher *interval times* separately. For this study, we calculate average *interval time* for each dataset and if the *interval time* is less or equal to average *interval time* we call it low *interval time*, otherwise, we call it high *interval time*. For each dataset and each feature, we calculate the correlation value between the feature and low and high *interval times*; these correlation values are shown in Table 3.

In Table 3, we observe that features like $Common_{in}$, $Common_{out}$, $Jaccard_{In}$, $Jaccard_{Out}$, *PrefAtt* and *PrefJacc* have high correlation with higher *interval time*. For the *Enron* dataset, some of these features ($Common_{in}$, $Jaccard_{In}$ and *PrefJacc*) are highly correlated with lower *interval time* as well. For the *MC-Email* dataset, *DirectedDist*, *ALR(v)*, *OutRatio* and *PageRank(v)* have noticeable correlation with lower *interval time* and other two features (*RLR(u)* and *In Ratio*) are inversely correlated with lower *interval time*. One surprising observation for the *MC-Email* dataset is that *PageRank(v)* is the poorest feature (Table 2), but highly correlated with both lower and higher *interval times*, mainly because the feature is positively correlated for lower *interval time* and inversely correlated with higher *interval time*. From Table 3 we understand that for different datasets user behavior varies and hence a distinct set of features becomes influential to the *interval time* (especially lower *interval time*) of that dataset.

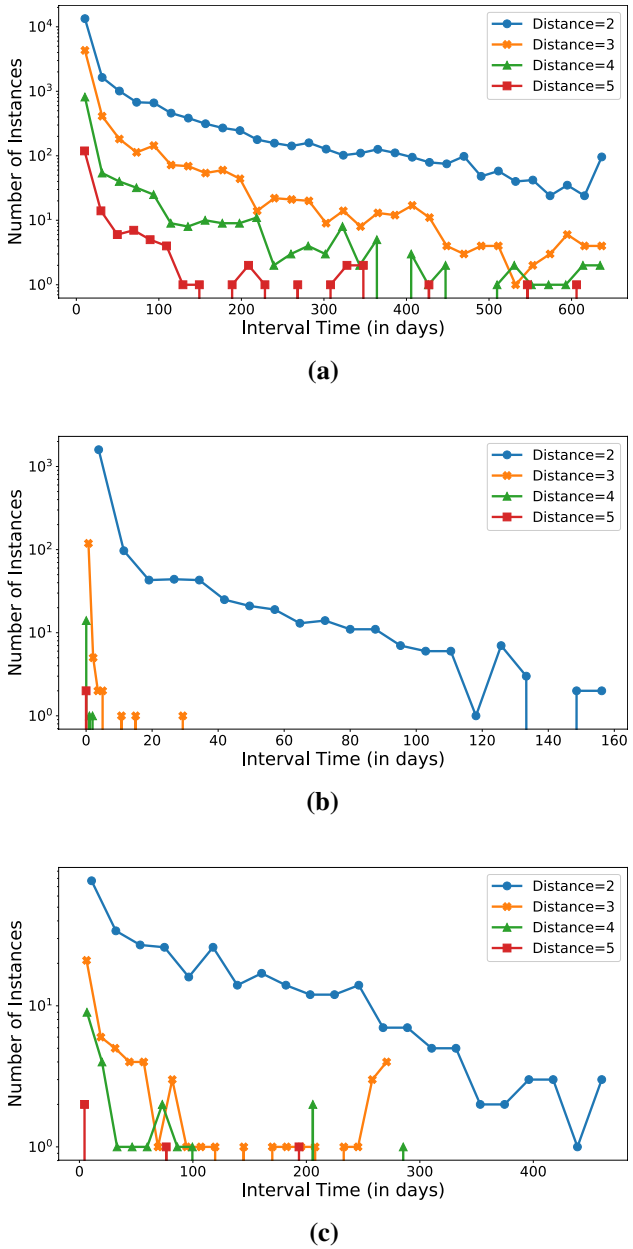

Fig. 5 *DirectedDist* versus *interval time*. **a** *Epinion* dataset. **b** *MC-Email* dataset. **c** *Enron* dataset

3.4 Proposed methodology using survival analysis

Survival analysis is widely used in the medical domain to predict survival time or time to a specific event (such as death) for patient datasets (Vinzamuri and Reddy 2013; Wang et al. 2017a). In the survival analysis setup, for a set of instances under observation, events happen over a time period, from which a survival model learns the temporal patterns of these events and predicts the survival time. Here, we propose a novel method to map the *RLTP* problem to a survival analysis task and explain survival analysis concepts from a reciprocal link creation perspective. For these concepts, we also provide suitable terminology for the *RLTP* problem to describe our approach clearly.

Beginning of graph expansion and study period: At the first time stamp, a given directed time-stamped network is static (initialized); the *beginning of graph expansion* is the second time stamp from when new links are added to the static network. Survival analysis assumes a starting time of the study, from when a model starts to observe for the events. In the *RLTP* problem, the *beginning of graph expansion* serves as the starting time of the study. For the *RLTP* problem, we divide the time stamps of the network into train and test time periods, and we observe the network for the reciprocal link creation till the end of the train period, so the last time stamp in the train period is considered to be the *end of the study*. Thus, the time window from the beginning of graph expansion to the last time stamp of train period is considered to be the *study period* which is the same as the train period.

Reciprocal event: For a parasocial link (v, u), if a reciprocal link (u, v) is created during the training period, we call it a *reciprocal event*, which is the event of interest in the *RLTP* problem. In the *RLTP* problem each parasocial link is a data instance, and time stamp of a parasocial link generation is the time when the data instance is considered into the network for study. Hence, the time stamp of a parasocial link generation is called the *starting time of observation* for that data instance (an ordered pair of vertices).

Ever-waiting links: We study the network for a limited time window (train period), and hence, for a set of parasocial links, the corresponding reciprocal event may not be observed before the end of the study (last time stamp of training period). We call these links *ever-waiting links*. *Ever-waiting* links carry the information that the reciprocal link creation event did not happen till the end of the train period. In the survival analysis terminology the *ever-waiting* links are also called censored instances; we use both of these terms interchangeably in this paper.

In a traditional regression task, *ever-waiting* links may either be ignored, because the target value (the *interval time*) for these instances are unknown, or they may be retained with an arbitrarily chosen large *interval time*, which is higher than the time difference between the end of the study time and the starting time of observation for that parasocial link. The first of the above approaches ignores important information; specifically, the ignored fact is that the *interval time* for *ever-waiting* links is higher than the time difference between the end of the study and the starting time of observation for that parasocial link. The second approach is simply a crude approximation of the target value. As mentioned before, the main reason to map the *RLTP* problem into survival regression analysis framework is to exploit the important information provided by the *ever-waiting* links.

Table 3 Correlation of features with Low and High *interval times*

Datasets features	*Epinion*		*MC-Emails*		*Enron*	
	Low	High	Low	High	Low	High
DirectedDist	− 0.00387	− 0.04587	0.14453	− 0.06022	− 0.04023	− 0.15364
$Common_{In}$	0.06671	0.33821	0.00018	0.41728	0.22168	0.35640
$Common_{Out}$	0.07231	0.24064	0.06639	0.27446	0.04738	0.20793
$Jaccard_{In}$	0.07765	0.15312	− 0.04154	0.29774	0.17726	0.10033
$Jaccard_{Out}$	0.06829	0.13183	− 0.07426	0.22517	0.07820	0.06360
RLR(u)	− 0.03937	0.06628	− 0.17897	0.02467	0.07949	0.06348
ALR(v)	− 0.01783	− 0.07657	0.15905	0.09455	0.08760	0.06401
PrefAtt	0.03049	0.14439	− 0.00163	0.31220	0.06305	0.32053
PrefJacc	0.04258	0.13021	− 0.06545	0.23248	0.16523	0.09010
InRatio	− 0.01251	− 0.01751	− 0.15333	− 0.04297	− 0.10385	− 0.09571
OutRatio	− 0.00700	− 0.02610	0.29600	− 0.06979	0.00578	0.12331
PageRank(u)	0.06118	0.20674	− 0.07606	− 0.09756	− 0.00557	− 0.12452
PageRank(v)	0.02399	0.14362	0.31830	− 0.13432	0.01606	0.03715

Target value of survival regression model The time difference between the starting time of observation (parasocial link generation time) and the time stamp of the reciprocal edge creation is the *interval time* which we want to predict in the *RLTP* task. For a reciprocal edge (u, v), the *interval time* is defined as $Int(u, v)$, as is discussed in Sect. 3.1. In a traditional survival model, the *interval time* is called the life span of an instance as for these models "death" is the event of interest. Hence, survival models that predict survival time can be adopted to predict the *interval time* for the *RLTP* problem. For training the prediction model, we need a feature vector for each data instance, along with the survival time and a binary event indication value (event occurred or not). For the *RLTP* problem, the feature vector of a parasocial edge is $\mathbf{x}_i \in \mathbb{R}^d$, a vector of topological features (Sect. 3.3) for the i'th parasocial link in training data, where feature dimension d is 13 (number of topological features). For each parasocial links of the training period, if the reciprocal event has occurred during training period then life span of parasocial link is the *interval time* with the event indication value set to 1; otherwise, for *ever-waiting* links, the time difference between the last time stamp of training and time stamp of the parasocial link generation is the survival time with event indication value set to 0. Given this training dataset, the target value (the *interval time*) of test instances are predicted by using a trained survival model. We use various survival models, which we discuss in the next subsection.

3.5 Survival models for the *RLTP* problem

As explained in the previous section, any survival model can be adopted to solve the *RLTP* problem. There are two types of widely used survival models: (1) semi-parametric models and (2) parametric models. Parametric models assume that *interval time* follows a known statistical distribution; hence, if the *interval time* for a dataset follows a distribution then parametric models perform very good for the dataset compared to a semi-parametric models. However, for many real-world datasets, it is difficult to find a suitable statistical distribution that fits well to the *interval time*, for these datasets semi-parametric models perform better than parametric models, because semi-parametric models do not assume any underlying distribution, rather they try to learn the actual distribution from the data. As we discussed in Sect. 3.2, some of our datasets are good fit for a statistical distribution but others are not. Hence, we conduct experiments with both semi-parametric and parametric models to offer a comprehensive study of the *RLTP* problem. In this section, we describe these selected semi-parametric and parametric models and their adaptation for solving the *RLTP* problem. Broadly, all types of models try to predict the survival time of an instance in the data by modeling three functions: (1) survival function, (2) hazard function and (3) event density function. Definitions and relationship between these three functions are described below:

Survival function $S(t)$: Survival models provide a principled approach for *interval time* prediction by modeling a survival function, which is defined as the probability value that the reciprocal edge creation does not happen for a given parasocial link before a specified time t.

$$S(t) = Pr(\boldsymbol{T} \geq t)$$

Here, $\boldsymbol{T}$ is a random variable representing the time of the reciprocal edge creation event.

Hazard function $\lambda(t)$ is the reciprocal event rate at time t conditional on the fact that the reciprocal event has not occurred until that time t,

$$\lambda(t) = \frac{f(t)}{S(t)} \tag{1}$$

where $f(t)$ is the *reciprocal event density function*, which is given as follows:

$$f(t) = \frac{d}{dt}(1 - S(t)) = -\frac{d}{dt}S(t)$$

For a given parasocial link if corresponding reciprocal link is not likely to be created at time t, then *Survival function* value for t is high. On the other hand, if the corresponding reciprocal link is highly likely to be created at time t then the *reciprocal event density function* value should be high and that leads to a higher value of the *Hazard function*. We can observe that both *survival function* and *hazard function* are interrelated and we can model either function for the *interval time* prediction. Next, we describe how semi-parametric Cox regression models the *hazard function* to solve the *RLTP* problem. Later we discuss parametric methods (BJ model and AFT models) and their approach for modeling the *survival function* with the help of different statistical distributions.

3.5.1 Cox regression

Cox regression model (Cox 1972) is the most widely used semi-parametric model for predicting the (interval) time taken for a reciprocal event to occur. The basic Cox model follows the proportional hazard assumption, for which the hazard function $\lambda(t \mid \mathbf{x}_i)$ takes the following form:

$$\begin{aligned}\lambda(t \mid \mathbf{x}_i) &= \lambda_0(t) \times \exp(\beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_d x_{id}) \\ &= \lambda_0(t) \times \exp(\mathbf{x}_i^T \boldsymbol{\beta})\end{aligned} \tag{2}$$

where $\mathbf{x}_i$ is the (topological) feature vector of a parasocial link represented as i'th data instance in the training data and d is the dimensionality of the features. Here, $\lambda_0(t)$ is called baseline hazard function, and $\boldsymbol{\beta}$ is the model parameter which Cox regression model learns. The Cox regression is called semi-parametric because the baseline hazard function $\lambda_0(t)$ can be any non-negative function of time. The probability of occurrence of reciprocal event for the ith parasocial link (data instance) at time t can be represented as ratio $\frac{\lambda(t|\mathbf{x}_i)}{\sum_{j \in R_t} \lambda(t|\mathbf{x}_j)}$, where R_t is the set of all instances for which the reciprocal event did not happen until t. The product of these probabilities gives the partial likelihood function:

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{N} \left[\frac{\exp(\mathbf{x}_i^T \boldsymbol{\beta})}{\sum_{j \in R_t} \exp(\mathbf{x}_j^T \boldsymbol{\beta})} \right]^{C_i} \tag{3}$$

Here N is the total number of parasocial links appeared during the training period and C_i is an event indicator value, i.e., if reciprocal link for the ith parasocial link appear during training period then $C_i = 1$ otherwise $C_i = 0$. The model parameter $\boldsymbol{\beta}$ is learnt by minimizing the negative log likelihood function. If $\widehat{\boldsymbol{\beta}}$ is the optimal model parameter, we have:

$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \quad \frac{1}{N} \sum_{i=1}^{N} \left[-C_i(\mathbf{x}_i^T \boldsymbol{\beta}) + C_i \log \left(\sum_{j \in R_t} \exp(\mathbf{x}_j^T \boldsymbol{\beta}) \right) \right] \tag{4}$$

Regularized Cox model: For complex model with high dimensional real-world datasets, over-fitting is a frequent problem. To avoid this, we need a regularization term in the objective function (Eq. 4). We observe in Sect. 3.3.3 that only a few features have a strong correlation with the target variable, so we want to use a sparse regularization model. In this work we use elastic net regularization. In literature, a Cox model with elastic net regularization is also known as Cox model with elastic net (EN) penalty (Zou and Hastie 2005). The penalty term P_{EN} is:

$$P_{EN}(\boldsymbol{\beta}) = \sum_{k=1}^{d} \left[\alpha|\beta_k| + \frac{1}{2}(1 - \alpha)\beta_k^2 \right] \tag{5}$$

where, $0 < \alpha \leq 1$ and with EN penalty the objective function in Eq. 4, becomes

$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \quad \frac{1}{N} \sum_{i=1}^{N} \left[-C_i(\mathbf{x}_i^T \boldsymbol{\beta}) + C_i \log \left(\sum_{j \in R_t} \exp(\mathbf{x}_j^T \boldsymbol{\beta}) \right) \right] + \gamma \cdot P_{EN}(\boldsymbol{\beta}) \tag{6}$$

Here, $\gamma > 0$ is a regularization constant. For solving this optimization task, we can use the maximum partial

Table 4 Density, survival and hazard functions for the distributions used with AFT model

Distributions	Density function	Survival function	Hazard function
Weibull	$\lambda k t^{k-1} \cdot \exp(-\lambda t^k)$	$\exp(-\lambda t^k)$	$\lambda k t^{k-1}$
Log-normal	$\frac{1}{\sqrt{2\pi}\sigma t} \exp(-\frac{(log(t)-\mu)^2}{2\sigma^2})$	$1 - \Phi(\frac{log(t)-\mu}{\sigma})$	$\frac{\frac{1}{\sqrt{2\pi}\sigma t} \exp(-\frac{(log(t)-\mu)^2}{2\sigma^2})}{1-\Phi(\frac{log(t)-\mu}{\sigma})}$
Log-logistic	$\frac{\lambda k t^{k-1}}{(1+\lambda t^k)^2}$	$\frac{1}{1+\lambda t^k}$	$\frac{\lambda k t^{k-1}}{1+\lambda t^k}$

Here, λ is scale parameter and k is shape parameter for both Weibull and log-logistic distribution. For log-normal distribution μ is the mean (location parameter), σ^2 is the variance and Φ is cumulative distribution function of normal distribution

likelihood estimator proposed by Cox (1972); it uses the Newton-Raphson method to iteratively find the estimated $\widehat{\boldsymbol{\beta}}$ which minimizes Eq. (6).

3.5.2 Parametric models

The main idea behind a parametric model is that it assumes that the *interval time* follows a specific statistical distribution. There are two ways to relate *interval time* and a statistical distribution: first, assume that the actual *interval time* for all parasocial links follows a distribution; and second, assume that the logarithm of the *interval time* follows a distribution. The models under the first assumption are referred to as linear regression models, and the models under later assumption are called accelerated failure time (AFT) models.

Generally, parametric models use maximum likelihood estimation (MLE) approach to learn model parameters. Let's assume all the parameters of a model are represented by $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots)^T$. For a given parasocial link (say ith link in the training data), if it is an *ever-waiting* link the corresponding survival function $S(t, \boldsymbol{\beta})$ at time t (in fact, any t value during a training period) should be near to 1, and if it is not an *ever-waiting* link then the reciprocal event density function $f(t_i, \boldsymbol{\beta})$ at time t_i (time of reciprocal event for the ith parasocial link) should be high (near to 1) for that link. Hence, the likelihood of all parasocial links of a training period is the product of their reciprocal event density functions or survival functions based on their state (whether the link is *ever-waiting* or not), i.e.,

$$L(\boldsymbol{\beta}) = \prod_{C_i=1} f(t_i, \boldsymbol{\beta}) \cdot \prod_{C_i=0} S(t_i, \boldsymbol{\beta}) \tag{7}$$

Linear regression model: The statistical linear regression with the least squares estimation is widely used for a variety of regression tasks. However, the issue with the model is that it cannot use information from *ever-waiting* links. For *interval time* prediction this issue can be handled by using a specific survival model such as the Buckley–James model (BJ model). The BJ model first estimates the *interval time* of training *ever-waiting* links using the Kaplan–Meier (KM) (Kaplan and Meier 1958) estimation method and then by using all parasocial links from training period to train a linear model. This linear model can be trained through MLE as described above. For more practical use, Wang and Wang (2010) proposed twin boosting method with BJ estimator, we use this method to solve the *RLTP* problem.

Accelerated failure time (AFT) model: An AFT model assumes that the logarithm of the *interval time* $\log(\boldsymbol{T})$ follows a statistical distribution, and it is linearly related to the (topological) feature vectors. The general form for AFT regression model is

$$log(\boldsymbol{T}) = \boldsymbol{X} \cdot \boldsymbol{\beta} + \sigma \cdot \epsilon \tag{8}$$

where $\boldsymbol{X}$ is the covariate matrix of size $N \times d$ where ith row of $\boldsymbol{X}$ is $\mathbf{x}_i$, $\boldsymbol{\beta}$ is a d dimensional coefficient vector (model parameters), σ ($\sigma > 0$) is an unknown scale parameter, and ϵ is an error variable which follows a similar distribution to $\log(\boldsymbol{T})$. For our problem, we use the three most suitable distributions (see Fig. 3) for *interval time*, the details of which are given in Table 4.

3.6 Algorithmic framework

Algorithm 1 Our Framework

1: For time-stamps (t_0 to t_M) of the input graph, divide the starting $p\%$ time-stamps as training period (t_0 - t_p) and remaining as testing period (t_{p+1} - t_M).
2: *train-set* ← all parasocial links generated before/at time-stamp t_p.
3: *test-set* ← all parasocial links generated after time-stamp t_p and immortal links.
4: Sort edge in *train-set* and *test-set* based on its edge creation time. {optional}
5: **for each** (edge) $e \in$ *train-set* **do**
6: G_{te} ← create a snapshot of the network at $t_e - 1$. {sorting helps in this step}
7: x_e ← generate topological features (Section 3.3) for edge e from the snapshot G_{te}.
8: add x_e to $\boldsymbol{X}$.
9: **end for**
10: Similarly, generate topological features for edges in the *test-set* and generate $\boldsymbol{X_{test}}$.
11: **for each** $e \in$ *train-set* **do**
12: **if** e has a reciprocal link e_r in dataset **then**
13: **if** $t_{er} \leq t_p$ **then**
14: $y_e \leftarrow Int(e) \leftarrow t_{er} - t_e$ {target value for the parasocial edge (data instance)}
15: $C_e \leftarrow 1$ {event indicator value for the parasocial edge (data instance)}
16: **else**
17: $y_e \leftarrow t_p - t_e$ {target value for the *ever-waiting* link (data instance)}
18: $C_e \leftarrow 0$
19: **end if**
20: **else**
21: $y_e \leftarrow t_p - t_e$ {target value for the *ever-waiting* link (data instance)}
22: $C_e \leftarrow 0$
23: **end if**
24: add y_e to $\boldsymbol{T}$.
25: add C_e to $\boldsymbol{C}$.
26: **end for**
27: **for each** $e \in$ *test-set* **do**
28: **if** e has a reciprocal link e_r in dataset **then**
29: $y_e \leftarrow Int(e) \leftarrow t_{er} - t_e$ {target value for the parasocial edge (data instance)}
30: $C_e \leftarrow 1$
31: **else**
32: $y_e \leftarrow t_M - t_e$ {target value for the *ever-waiting* link (data instance)}
33: $C_e \leftarrow 0$
34: **end if**
35: add y_e to $\boldsymbol{T_{test}}$.
36: add C_e to $\boldsymbol{C_{test}}$.
37: **end for**
38: {Use one of the methods among Cox, BJ, and AFT; below we call all three methods}
39: $\boldsymbol{cox}$ ← cocktail($\boldsymbol{X}$, $\boldsymbol{T}$, $\boldsymbol{C}$) {method of *fastcox* (*R* package)}
40: {For given distribution *dist*}
41: $\boldsymbol{AFT_{dist}}$ ← survreg($\boldsymbol{X}$, $\boldsymbol{T}$, $\boldsymbol{C}$,distribution=*dist*) {method of *survival* (*R* package)}
42: $\boldsymbol{BJmodel}$ ← bujar($\boldsymbol{X}$, $\boldsymbol{T}$, $\boldsymbol{C}$) {method of *bujar* (*R* package)}
43: The $\boldsymbol{cox}$, $\boldsymbol{AFT_{dist}}$ and $\boldsymbol{BJmodel}$ contain the model parameters $\boldsymbol{\beta}$.
44: *test-res* ← predict interval time y_e for each edge $e \in$ *test-set* using $\boldsymbol{X_{test}}$ and $\boldsymbol{\beta}$.
45: evaluate *test-res* using $\boldsymbol{T_{test}}$ and $\boldsymbol{C_{test}}$.

In Algorithm 1, we describe a general framework of our proposed method. First, we divide the time stamps of the input graph into train and test periods as mentioned in line 1 of Algorithm 1. After that we create training data instances (*train-set*) and test data instance (*test-set*) from the corresponding train and test periods (Lines 2–4). Then we calculate topological features for each parasocial link (data instance) in the *train-set* and *test-set* as described in Lines 5–10 of Algorithm 1. After that we generate target variable for each data instance (Lines 11–26), for which we observe the corresponding reciprocal link in the graph. For a parasocial link $e \in$ *train-set*, if the corresponding reciprocal link is generated during train period then *interval time Int*(e) (Sect. 3.1) is the target value with event indicator value $C_e = 1$; otherwise, time difference between the link creation and end of training period acts as the target value with event indicator value $C_e = 0$. Similarly, we generate target values for data instance of *test-set* as explained in Lines 27–35 of Algorithm 1. Then, we use *R* libraries to train the survival models with training data and predict target values for the test data to generate the test results (*test-res*) and lastly we evaluate that *test-res*.

4 Experiments and results

We conducted a set of rigorous experiments to demonstrate the benefit of using censored information and the superiority of proposed survival models to solve the *RLTP* problem. We used five proposed survival models: Cox regression model, AFT model with Weibull, log-normal and log-logistic distributions and Buckley–James (BJ) regression model. To prove the fact that the proposed survival models are better suited for solving the *RLTP* problem, we compared them with traditional regression models such as ridge regression (RidgeReg), lasso regression (LassoReg), feed-forward neural networks (FFNN) and support vector regression (SVR). Note that these traditional regression models cannot use censored information (*ever-waiting* links). We also compare proposed Cox regression model with generalized linear model (GML), which is an adopted model from (Sun et al. 2012).

In addition to the suitability of the proposed survival models for the *RLTP* problem, we also demonstrate the usability of the *ever-waiting* links. For that, we conducted experiments where we train the survival models without censored information and compare the performance of the models on the test dataset. We report the improvement in the performance when the *ever-waiting* links are used for training the survival models.

Lastly, we conduct an experiment to show that reciprocal links with short *interval time* contain enough information required for training the survival models.

4.1 Datasets

For the experiments, we use three real-world datasets *Epinion*, *MC-Email* and *Enron*. We discuss these datasets in Sect. 3.2, and basic statistics of the datasets are shown in Table 1.

Generating a synthetic dataset for the *RLTP* problem is a challenging task, because in the literature most of the synthetic graph generation models try to mimic basic real-world properties such as power-law degree distribution (Faloutsos et al. 1999), community structures (Leskovec et al. 2007), etc. All these methods generate directed networks with extremely low reciprocity—generally, less than 1%. Durak et al. have proposed a synthetic network generation algorithm which also considers reciprocity (Nurcan Durak 2013). We use this algorithm for generating three synthetic graphs where the vertex count varies between 10,000 (10K) to 30,000 (30K) with increments of 10K. Edges of these synthetic networks have no time stamps; hence, we assign random time stamps between 0 to 100 to parasocial links. The time stamps of reciprocal links of these synthetic networks are selected by matching the reciprocal link *interval time* of the *Epinion* dataset through the best fit Weibull distribution.

Table 5 *Epinion* dataset: TD-AUC results [mean (± standard deviation)] with different splits used for training period

Method/split	60%	70%	80%
RidgeReg	0.6185 (± .0018)	0.6086 (± .0013)	0.6060 (± .0018)
LassoReg	0.6169 (± .0013)	0.6020 (± .0014)	0.6039 (± .0017)
FFNN	0.5510 (± .1296)	0.5048 (± .0822)	0.4456 (± .0725)
SVR	0.4791 (± .0005)	0.4871 (± .0039)	0.4914 (± .0030)
BJ Model	0.7312 (± .0010)	0.7339 (± .0020)	0.7416 (± .0021)
Weibull	0.3807 (± .0763)	0.5210 (± .1446)	0.5232 (± .1282)
Log-normal	0.3660 (± .0388)	0.4461 (± .0283)	0.4283 (± .0305)
Log-logistic	0.4901 (± .0098)	0.5110 (± .0196)	0.5132 (± .0188)
Cox	**0.7364** (± .0025)	**0.7436** (± .0016)	**0.7485** (± .0028)

Bold values are specifying the best performance among all models

Table 6 *MC-Email* dataset: TD-AUC results [mean (± standard deviation)] with different splits used for training period

Method/split	60%	70%	80%
RidgeReg	0.6213 (± .0087)	0.6083 (± .0146)	0.6014 (± .0125)
LassoReg	0.5884 (± .0100)	0.5709 (± .0201)	0.5686 (± .0074)
FFNN	0.4199 (± .0800)	0.4609 (± .0964)	0.5069 (± .0915)
SVR	0.5462 (± .0154)	0.5737 (± .0187)	0.5530 (± .0150)
BJ Model	0.5898 (± .0087)	0.5910 (± .0146)	0.6103 (± .0059)
Weibull	0.6139 (± .0075)	0.6171 (± .0069)	0.6315 (± .0166)
Log-normal	0.6391 (± .0053)	0.6463 (± .0015)	0.6695 (± .0116)
Log-logistic	0.6380 (± .0121)	0.6494 (± .0062)	0.6747 (± .0201)
Cox	**0.6527** (± .0097)	**0.6558** (± .0125)	**0.6797** (± .0062)

Bold values are specifying the best performance among all models

Table 7 *Enron* dataset: TD-AUC results [mean (± standard deviation)] with different splits used for training period

Method/split	60%	70%	80%
RidgeReg	0.5732 (± .0073)	0.5847 (± .0159)	0.5318 (± .0164)
LassoReg	0.5740 (± .0076)	0.5850 (± .0152)	0.5309 (± .0178)
FFNN	0.4900 (± .0258)	0.5407 (± .0434)	0.5363 (± .0561)
SVR	0.5490 (± .0080)	0.5680 (± .0176)	0.5608 (± .0136)
BJ Model	0.5292 (± .0120)	0.6096 (± .0076)	0.5599 (± .0121)
Weibull	0.5710 (± .0168)	**0.6319** (± .0050)	**0.5980** (± .0096)
Log-normal	0.5713 (± .0146)	0.6146 (± .0097)	0.5862 (± .0129)
Log-logistic	0.5787 (± .0171)	0.6224 (± .0069)	0.5917 (± .0101)
Cox	**0.5854** (± .0166)	0.6311 (± .0110)	0.5919 (± .0084)

Bold values are specifying the best performance among all models

4.2 Experimental setting

For our experiments, we divide the time stamps of a dataset into two non-overlapping continuous partitions, where the earlier partition is the train period and the latter is the test period. In three different experiments, we use, respectively, 60, 70 and 80% of the earlier time stamps as the train periods and the remaining time stamps as the test period. For synthetic datasets, a 70:30 split of the time stamps is used as the train and test period of our experiments. For calculating the topological features explained in Sect. 3.3 for a parasocial link (data instance), we take a snapshot of the network until the time stamp of the corresponding reciprocal link or end of the train period (whichever is earlier).

Like any other link prediction task, *RLTP* also suffers from the class imbalance issue, where the number of positive instances ($C_i = 1$) is much smaller than that of the negative instances. To alleviate this problem, we use the well-known majority undersampling (Bunkhumpornpat et al. 2011) strategy as discussed below: all the reciprocal links generated during a train period are considered in the training data pool as positive instances and only 50% of the parasocial links generated during the same period are censored negative instances ($C_i = 0$) in the pool. The test data pool (and their labels) is also generated similarly from the test period. As train and test data instances need to be from their corresponding time periods, we use a modified K-fold cross-validation, where each fold contains a random subset of train and test data instances from their respective pools. For all our experiments, we used 5-fold cross validation in this manner.

For minimizing the objective function [Eq. (4)] of censored problem formulation of *RLTP*, for the Cox regression model, we used cocktail algorithm (Yang and Zou 2013) [the library is provided by the authors of Yang and Zou (2013)]. For AFT models and BJ regression, we used *Survival* package[3] and *Bujar* package[4], respectively, available in **R**. For RidgeReg, LassoReg and SVR, we used scikit-learn python library and for FFNN, we used MATLAB NN toolbox. We used TopCom indexing method (Dave and Hasan 2015, 2016) to find shortest directed distance feature. To choose the best parameters of SVR, we used grid search, where the cost parameter C takes values from $\{0.0001, 0.001, 0.01, 0.1, 1.0\}$ and Epsilon (ϵ) takes values from $\{0.0001, 0.001, 0.01, 1.0\}$.

4.3 Evaluation metrics

Datasets generated from directed time-stamped networks are longitudinal data, and for the *RLTP* problem the datasets also contain censored information. Evaluating models on these datasets using traditional evaluation metrics is not suitable, instead we use time-dependent AUC (also known as c-Index), which is widely used in longitudinal data analysis (Pencina and D'Agostino 2004).

For a pair of data instances, assume (y_i, y_j) and $(\hat{y}_i, \hat{y}_j)$ are the target and the predicted values, respectively. The time-dependent AUC is defined as the probability of $\hat{y}_i > \hat{y}_j$ given $y_i > y_j$. If target y_i has only 2 possible values, then time-dependent AUC is the same as the popular AUC (Area Under ROC Curve) metric for classification. Similar to the AUC metric, time-dependent AUC takes values between 0 and 1, where 1 is the best possible value for this metric. Time-dependent AUC (TD-AUC) is calculated as follows:

$$TD\text{-}AUC = \frac{1}{N_{cnt}} \sum_{i:C_i=1} \sum_{y_j>y_i} mathbb1(\hat{y}_j > \hat{y}_i) \tag{9}$$

where, N_{cnt} is total count of (y_i, y_j) pairs such that $C_i = 1$ (the event has occurred) and $y_j > y_i$ holds.

For the Cox regression model, the predicted value is the hazard value and for a higher hazard value the event occurs earlier; hence, the time-dependent AUC for Cox can be calculated as:

$$TD\text{-}AUC = \frac{1}{N_{cnt}} \sum_{i:C_i=1} \sum_{y_j>y_i} mathbb1(\mathbf{x}_i^T \hat{\boldsymbol{\beta}} > \mathbf{x}_j^T \hat{\boldsymbol{\beta}}) \tag{10}$$

Table 8 TD-AUC results [mean (± standard deviation)] for various methods on synthetic datasets

Method	*10K*	*20K*	*30K*
RidgeReg	0.5210 (± .0029)	0.4949 (± .0018)	0.5203 (± .0023)
LassoReg	0.5150 (± .0034)	0.4876 (± .0059)	0.5091 (± .0048)
FFNN	0.4999 (± .0517)	0.4967 (± .0151)	0.5068 (± .0631)
SVR	0.5379 (± .0021)	0.4963 (± .0026)	0.5473 (± .0015)
BJ Model	0.5589 (± .0011)	0.5232 (± .0013)	0.5557 (± .0008)
Weibull	0.5641 (± .0036)	0.4954 (± .0027)	0.5559 (± .0015)
Log-normal	**0.5670** (± .0027)	0.4991 (± .0030)	**0.5618** (± .0011)
Log-logistic	0.5597 (± .0029)	0.4985 (± .0042)	0.5576 (± .0019)
Cox	0.5604 (± .0025)	**0.5282** (± .0026)	0.5558 (± .0016)

Bold values are specifying the best performance among all models

4.4 Comparison results of survival models and regression models

We compared proposed survival models with four other traditional regression models, and our results are shown in Tables 5, 6 and 7, where columns represent different training splits and each row represents a prediction model. A horizontal bar separates the traditional regression models in the upper part and the survival-based models in the lower

[3] https://cran.r-project.org/web/packages/survival/index.html.

[4] https://cran.r-project.org/web/packages/bujar/index.html.

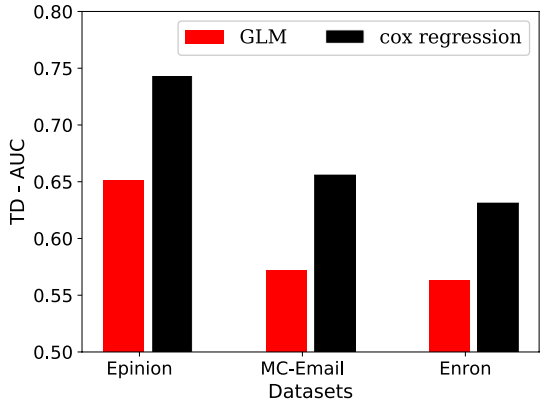

Fig. 6 Comparison of GLM and cox regression

Table 9 Time-dependent AUC results [mean (± standard deviation)] for survival analysis methods with and without *ever-waiting* links on real datasets

Model	w/o *ever-waiting*	with *ever-waiting*	%incr
Epinion			
BJ model	0.4580 (± .0042)	0.7416 (± .0021)	61.94
Weibull	0.4096 (± .0090)	0.5232 (± .1282)	27.73
Log-normal	0.4218 (± .0035)	0.4283 (± .0305)	1.53
Log-logistic	0.3767 (± .0024)	0.5132 (± .0188)	36.23
Cox	0.4975 (± .0024)	0.7485 (± .0028)	50.45
MC-Email			
BJ model	0.4787 (± .0131)	0.6103 (± .0059)	27.51
Weibull	0.5517 (± .0101)	0.6315 (± .0166)	14.46
Log-normal	0.6342 (± .0146)	0.6695 (± .0116)	5.56
Log-logistic	0.6331 (± .0152)	0.6747 (± .0201)	6.57
Cox	0.6102 (± .0137)	0.6797 (± .0062)	11.38
Enron			
BJ model	0.5499 (± .0134)	0.5599 (± .0121)	1.82
Weibull	0.5330 (± .0237)	0.5980 (± .0096)	12.20
Log-normal	0.5344 (± .0070)	0.5862 (± .0129)	9.71
Log-logistic	0.5379 (± .0053)	0.5917 (± .0101)	10.01
Cox	0.5481 (± .0234)	0.5919 (± .0084)	7.99

part. Here, each table cell shows mean and standard deviation for TD-AUC values. For most of the cases, the Cox regression model performs the best.

For the *Epinion* dataset, as depicted in Table 5, the Cox regression model performs the best with mean TD-AUC 0.7364, 0.7463 and 0.7485 for training period with 60, 70 and 80% splits of time stamps, respectively. Here, with increase in the training data we can clearly see improvement in the performance, which is an expected behavior because with more training examples the model learns better. BJ model is the next best with performance very close to the Cox model. For this model also, the mean TD-AUC

Table 10 Time-dependent AUC results [mean (± standard deviation)] for survival analysis methods with and without *ever-waiting* links on synthetic datasets

Model	w/o *ever-waiting*	with *ever-waiting*	%incr
10K			
BJ model	0.5730 (± .0045)	0.5589 (± .0011)	− 2.46
Weibull	0.4847 (± .0096)	0.5641 (± .0036)	16.37
Log-normal	0.5564 (± .0102)	0.5670 (± .0027)	1.89
Log-logistic	0.5546 (± .0128)	0.5597 (± .0029)	0.92
Cox	0.4910 (± .0037)	0.5604 (± .0025)	14.14
20K			
BJ model	0.4956 (± .0062)	0.5232 (± .0013)	5.57
Weibull	0.4951 (± .0025)	0.4954 (± .0027)	0.06
log-normal	0.4984 (± .0018)	0.4991 (± .0030)	0.15
Log-logistic	0.4965 (± .0055)	0.4985 (± .0042)	0.41
Cox	0.4938 (± .0098)	0.5282 (± .0026)	6.97
30K			
BJ model	0.5548 (± .0049)	0.5557 (± .0008)	0.17
Weibull	0.4544 (± .0020)	0.5559 (± .0015)	22.35
Log-normal	0.5270 (± .0044)	0.5618 (± .0011)	6.61
Log-logistic	0.5243 (± .0042)	0.5576 (± .0019)	6.35
Cox	0.4637 (± .0051)	0.5558 (± .0016)	19.86

improves from 0.7312 to 0.7416 as we increase the training data. Similar behavior is observed for other survival models, but the performance of the AFT models is, unfortunately, not good for the dataset. This can be attributed to the fact that AFT models make strict distribution assumptions on the data and such assumption may not be suitable for the *Epinion* dataset (Fig. 3).

For the *Epinion* dataset, among the traditional regression-based methods, ridge regression performs better than any other competing methods with mean TD-AUC in the range between 0.60 and 0.61. But, when we compare its performance over different training splits, we see that its performance does not improve as we increase the training data. The same behavior holds for other traditional regression methods, such as Lasso regression and FFNN. One possible explanation for this behavior is model under-fitting; that is, the majority of the errors in the traditional regression models are coming from the bias error, so the error does not improve much with a larger dataset which reduces variance error only. On the other hand, survival analysis-based models are more sophisticated, which enables them to design complex functions for predicting the time, thus overcoming the under-fitting issue.

For the *MC-Email* dataset, the overall behavior of the models is very similar to the *Epinion* dataset. Here again the Cox regression model performs the best with mean TD-AUC between 0.65 and 0.68 and its results are improved for larger training data. Performance of different AFT models

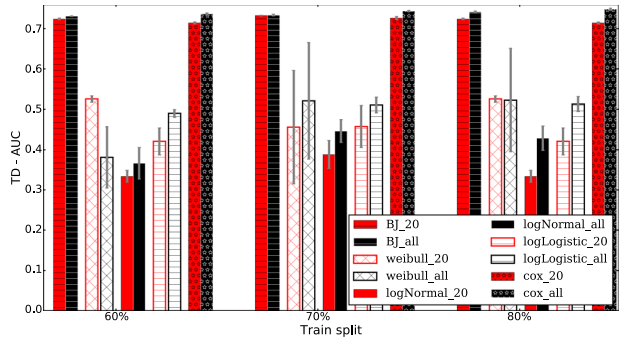

Fig. 7 *Epinion* dataset: comparison of training with top 20% reciprocal links and all reciprocal links

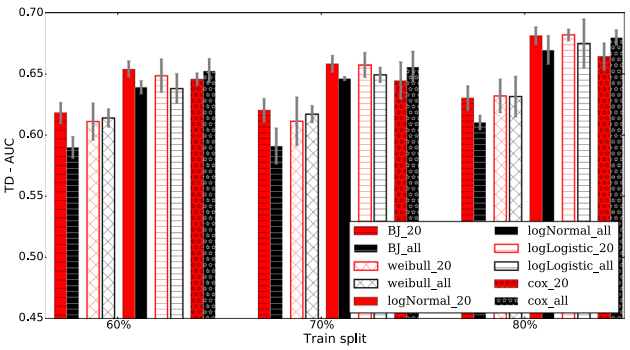

Fig. 8 *MC-Email* dataset: comparison of training with top 20% reciprocal links and all reciprocal links

varies, but they all perform better than all of the traditional regression methods. In particular, AFT with log-logistic and log-normal distributions perform great and their mean TD-AUC is very close to the results of the Cox regression as shown in Table 6. The performance of all survival models improves as we provide more training data. On the other hand, best among the competing methods is ridge regression with a mean TD-AUC between 0.60 and 0.62. As we have discussed earlier, this model suffers from under-fitting problem.

For the *Enron* dataset, results are shown in Table 7. Here, for the training period with 60% split, Cox regression performs the best with mean TD-AUC 0.58. For the other two splits, AFT model with Weibull distribution performs the best with mean TD-AUC 0.63 and 0.59. The BJ model performs poorly compared to the other survival models with mean TD-AUC ranging from 0.52 to 0.6, but the performance of BJ model is still better than all competing regression methods for training period with 70% and 80% splits of time stamps. For this dataset, for 80% training split, none of the models have better performance than the other splits. This is due to the fact that this dataset is extremely sparse and it has only 3007 links created during 944 time stamps

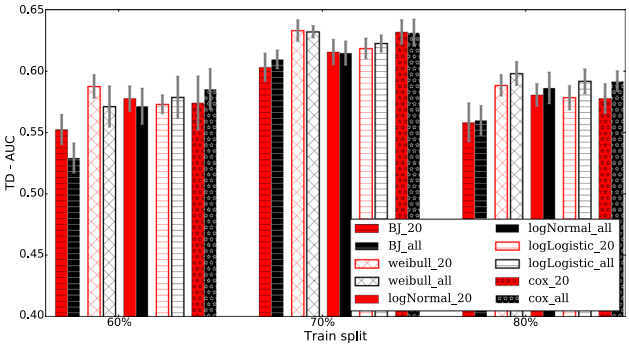

Fig. 9 *Enron* dataset: comparison of training with top 20% reciprocal links and all reciprocal links

(Table 1). Hence, even the 80% split does not provide more informative training samples to perform good prediction on remaining data.

The results for synthetic networks are shown in Table 8 by using the mean TD-AUC and standard deviation metrics. As we observe the results in this table, we can easily conclude that survival models always perform better than traditional regression methods. For two datasets with 10K and 30K node instances, the AFT model with log-normal distribution performs the best among all, while for the dataset with 20K nodes the Cox regression performs the best. The performance of survival models is consistently very similar except for dataset with 20K node where Cox and BJ models clearly perform better than AFT models. Among competing methods, SVR always performs better than others.

4.4.1 Comparison with GLM

Sun et al. (2012) proposed a method to predict link generation time in a heterogeneous network, where they design a unique feature for the task and use the feature with generalized linear model (GLM) for the prediction task. This proposed feature is designed based on meta-path (a simple path with link label information) in a heterogeneous network. We adopted this feature for a homogeneous network, and the adopted feature can be described as a number of simple paths of size k between two nodes. Counting the number of simple paths is an extremely costly operation, especially for a large dataset such as the *Epinion* network; hence, for this experiment, we use k upto 5, i.e., $k \in \{2, 3, 4, 5\}$ for all three networks, *Epinion*, *MC-Email* and *Enron*. We provide these homogeneous feature values to GLM (with gamma distribution) to solve the *RLTP* problem. For this experiment we use a 70% split of time stamps as train period and remaining 30% as test period. The results of this experiment are depicted in Fig. 6, where GLM is compared with the Cox regression model for all three datasets. From Fig. 6, we observe that the Cox regression outperforms the GLM

Table 11 Time-dependent AUC results [mean (± standard deviation)] for survival analysis methods with top5-features and all features

Model	Top5-features	All features	%incr
Epinion			
BJ model	0.6541 (± .0027)	0.7339 (± .0020)	12.20
Weibull	0.4878 (± .0872)	0.5210 (± .1446)	6.80
Log-normal	0.3157 (± .0062)	0.4461 (± .0283)	41.32
Log-logistic	0.3360 (± .0032)	0.5110 (± .0196)	52.10
Cox	0.6292 (± .0056)	0.7436 (± .0016)	18.19
MC-Email			
BJ model	0.4728 (± .0059)	0.5910 (± .0146)	25.00
Weibull	0.5503 (± .0102)	0.6171 (± .0069)	12.13
Log-normal	0.5802 (± .0074)	0.6463 (± .0015)	11.40
Log-logistic	0.5917 (± .0125)	0.6494 (± .0062)	9.76
Cox	0.5738 (± .0187)	0.6558 (± .0125)	14.29
Enron			
BJ model	0.5995 (± .0061)	0.6096 (± .0076)	1.69
Weibull	0.5985 (± .0140)	0.6319 (± .0050)	5.58
Log-normal	0.5972 (± .0130)	0.6146 (± .0097)	2.92
Log-logistic	0.6043 (± .0070)	0.6224 (± .0069)	2.99
Cox	0.5964 (± .0069)	0.6311 (± .0110)	5.82

model for all three datasets by noticeable margins. We believe one of the main reasons for the poor performance of the GLM-based method is that the feature proposed by Sun et al. (2012) is carefully designed for an author paper-based heterogeneous network and its adoption in a homogeneous network is not very useful.

4.5 Importance of *ever-waiting* links

We conducted experiments to show the importance of *ever-waiting* links, and the results are depicted in Tables 9 and 10. Table 9 shows the increment in TD-AUC up to 62% in the real-world datasets, when the survival models are provided with censored information (*ever-waiting* links) during the training, as compared to when the models are trained without censored information. For the *Epinion* dataset, the increment in the results is significant (more than 27% for all models) except AFT with log-normal distribution. Similarly, for the *MC-Email* and the *Enron* datasets the increment is up to 27%, which is substantial. As shown in Table 10, for the synthetic datasets we also have very similar increment in the results except for the BJ model with datasets of 10K nodes. For the most part the increment in performance is high for the Cox regression and the AFT with Weibull distribution. However, for other models the increment is limited to around 10%. The modest contribution of *ever-waiting* links for the case of synthetic networks can be attributed to the network generation model. We used Durak et al's model (Nurcan Durak 2013), which selects pairs of vertices for reciprocal link creation based on only degree distribution without considering any of the social phenomena, so the features that we are using may be not very effective for the synthetic datasets.

4.6 Importance of reciprocal links with small *interval time*

For the *RLTP* problem, reciprocal links carry very useful information and this information is not distributed uniformly over all reciprocal links. We described in Sect. 3.2 that for most of the reciprocal links the corresponding time interval is relatively small, and very few have high time interval as depicted in Fig. 2. The reciprocal links for which the corresponding time interval is equal to or less than 20% of the maximum time interval among all the time intervals of reciprocal links in the dataset are called "top 20%" reciprocal links. We trained survival models with top 20% reciprocal links (with *ever-waiting* links) and compared the results of these models with results of models trained with all reciprocal links (with *ever-waiting* links).

Results for these experiments are shown in Figs. 7, 8, 9, where all red bars represent different models trained with top 20% reciprocal links and all black bars represent the same models trained using all reciprocal links. We can see that, for all three datasets, survival models trained with top 20% reciprocal links perform very similar or better to the models trained with all reciprocal links. This observation supports our argument that all reciprocal links do not carry same amount of information, but notable amounts of information lie in the reciprocal links with short *interval time*.

4.7 Contribution of top5-features

In Sect. 3.3.3, we study correlation of different features with *interval time*. Through this experiment, we study the contribution of top five highly correlated features (top5-features) to solve the *RLTP* problem. From Table 2, we can find these top5-features for each real-world dataset. We can see, for *Epinion* dataset $Common_{in}$, $Common_{out}$, $Jaccard_{In}$, *PrefAtt* and *PageRank(u)* are highly correlated features. Similarly, for both *MC-Email* and *Enron* datasets $Common_{in}$, $Common_{out}$, $Jaccard_{In}$, *PrefAtt* and *PrefJacc* are the top5-features (Table 2). For this comparison study, we prepared train and test instances similarly as described in Sect. 4.2 with 70% training split, but the difference is, here each data instance is represented by only corresponding top5-features. We use proposed survival models (Sect. 3.5) with top5-features data to solve the *RLTP* problem.

Table 11 shows results for the comparison experiment with mean TD-AUC value and standard deviation for 5 independent runs. The last column in Table 11 shows the

increment in the mean TD-AUC value from top5-features data to all features data. This table clarifies the importance of the other features with lower correlation values (Table 2), because for both the *Epinion* and the *MC-Email* datasets the increment in the results is noticeable. But for the *Enron* dataset the increment is not very impressive; we believe low number of data instances and very high correlation of top5-features are the main reasons for this shortcoming.

5 Conclusion and future works

In this paper, we proposed a novel problem, namely reciprocal link time prediction (*RLTP*), which has wide applicability in email, social and other directed networks. We designed various socially meaningful topological features specifically for directed networks, which are useful to solve the *RLTP* problem. We mapped the *RLTP* problem into a survival analysis task and through experiments on three real-life network datasets, we showed that such a framework is better suited than traditional regression-based approaches for solving the *RLTP* problem. We demonstrated that using *ever-waiting* links for training adds valuable information to the prediction models. We also investigated the information contributed by the reciprocal links and showed that the majority of the required information lies in the top few percent (20%) of the reciprocal links. To the best of our knowledge, this is the first study on time interval prediction for reciprocal links, which is useful to answer response time for emails or friend requests. It can also be used for recommendation in trust networks for suggesting a new connection (parasocial link) for which, the predicted response time is very small.

The *RLTP* is a novel problem, and this is one of the earliest and comprehensive study of the problem. There do exist some opportunities to extend our work. For example, we have used basic survival analysis-based regression models, but one can study the timing patterns and design complex regression model by considering the timing patterns. Also, one can design sophisticated time-dependent topological features that carry more information to solve the *RLTP* problem and study different sparse prediction models to find suitable features for the prediction.

Acknowledgements This research is supported by National Science Foundation (NSF) career award (IIS-1149851) and in part by the NSF Grants IIS-1707498, IIS-1619028 and IIS-1646881.

References

Adalı S, Golbeck J (2014) Predicting personality with social behavior: a comparative study. Soc Netw Anal Min 4(1):159

Akaike H (1998) Information theory and an extension of the maximum likelihood principle. Springer, New York, pp 199–213

Anand S, Chandramouli R, Subbalakshmi KP, Venkataraman M (2013) Altruism in social networks: good guys do finish first. Soc Netw Anal Min 3(2):167–177

Bogdanov P, Busch M, Moehlis J, Singh AK, Szymanski BK (2014) Modeling individual topic-specific behavior and influence backbone networks in social media. Soc Netw Anal Min 4(1):204

Bunkhumpornpat C, Sinapiromsaran K, Lursinsap C (2011) Mute: Majority under-sampling technique. In: International conference on information communications and signal processing (ICICS), IEEE, pp 1–4

Cheng J, Romero DM, Meeder B, Kleinberg J (2011) Predicting reciprocity in social networks. In: IEEE 3rd international conference on privacy, security, risk and trust (PASSAT) and 2011 IEEE 3rd inernational conference on social computing (SocialCom), 2011, pp 49–56

Chierichetti F, Kumar R, Lattanzi S, Mitzenmacher M, Panconesi A, Raghavan P (2009) On compressing social networks. In: ACM SIGKDD international conference on Knowledge discovery and data mining, pp 219–228

Cox DR (1972) Regression models and life-tables. J Roy Stat Soc Ser B (Methodol) 34(2):187–220

Dave VS, Hasan MA (2015) Topcom: Index for shortest distance query in directed graph. Database and expert systems applications. Springer, Cham, pp 471–480

Dave VS, Hasan MA (2016) Topcom: index for shortest distance query in directed graph. CoRR abs/1602.01537, http://arxiv.org/abs/1602.01537

Dave VS, Al Hasan M, Reddy CK (2017) How fast will you get a response? predicting interval time for reciprocal link creation. In: AAAI international conference on web and social media (ICWSM)

Devineni P, Koutra D, Faloutsos M, Faloutsos C (2017) Facebook wall posts: a model of user behaviors. Soc Netw Anal Min 7(1):6

Dong X, Gabrilovich E, Heitz G, Horn W, Lao N, Murphy K, Strohmann T, Sun S, Zhang W (2014) Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In: Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '14, pp 601–610

Dong Y, Tang J, Wu S, Tian J, Chawla NV, Rao J, Cao H (2012) Link prediction and recommendation across heterogeneous social networks. In: Proceedings of the IEEE 12th international conference on data mining, ICDM '12, pp 181–190

Dumba B, Golnari G, Zhang ZL (2016) Analysis of a reciprocal network using Google+: structural properties and evolution. Springer, Berlin, pp 14–26

Esslimani I, Brun A, Boyer A (2011) Densifying a behavioral recommender system by social networks link prediction methods. Soc Netw Anal Min 1(3):159–172

Faloutsos M, Faloutsos P, Faloutsos C (1999) On power-law relationships of the internet topology. ACM SIGCOMM Comput Commun Rev 29:251–262

Feng X, Zhao J, Fang Z, Xu K (2014) Time-aware reciprocity prediction in trust network. In: Advances in social networks analysis and mining (ASONAM), pp 234–237

Gong NZ, Xu W (2014) Reciprocal versus parasocial relationships in online social networks. Soc Netw Anal Min 4(1):1–14

Hasan MA, Zaki MJ (2011) A survey of link prediction in social networks. In: Agaarwal CC (ed) Social network data analytics. Springer, New York

Hasan MA, Chaoji V, Salem S, Zaki M (2006) Link prediction using supervised learning. In: In Proceedings of the SDM 06 workshop on link analysis, counterterrorism and security

Hopcroft J, Lou T, Tang J (2011) Who will follow you back?: Reciprocal relationship prediction. In: Proceedings of the CIKM, pp 1137–1146

Kaplan EL, Meier P (1958) Nonparametric estimation from incomplete observations. J Am Stat Assoc 53(282):457–481. http://www.jstor.org/stable/2281868

Kuhnt MR, Brust OA (2014) Low reciprocity rates in acquaintance networks of young adults: fact or artifact? Soc Netw Anal Min 4(1):167

Leider S, Mobius MM, Rosenblat T, Do QA (2007) How much is a friend worth? directed altruism and enforced reciprocity in social networks. Revision of NBER working paper 13135, Cambridge, Mass, National Bureau of Economics Research

Leskovec J, Kleinberg J, Faloutsos C (2007) Graph evolution: densification and shrinking diameters. ACM Trans Knowl Discov Data (TKDD) 1(1). https://doi.org/10.1145/1217299.1217301

Li M, Jia Y, Wang Y, Zhao Z, Cheng X (2016) Predicting links and their building time: a path-based approach. In: Proceedings of the 30th AAAI conference on artificial intelligence, AAAI Press, AAAI'16, pp 4228–4229

Liaghat Z, Rasekh AH, Mahdavi A (2013) Application of data mining methods for link prediction in social networks. Soc Netw Anal Min 3(2):143–150

Liben-Nowell D, Kleinberg J (2003) The link prediction problem for social networks. In: Proceedings of the CIKM, pp 556–559

Nurcan Durak APCS, Kolda Tamara G (2013) A scalable null model for directed graphs matching all degree distributions: In, out, and reciprocal. IEEE workshop on network science. IEEE Press, Piscataway, NJ, pp 23–30

Pencina MJ, D'Agostino RB (2004) Overall-c as a measure of discrimination in survival analysis: model specific population value and confidence interval estimation. Stat Med 23(13):2109–2123

Roshanaei M, Mishra S (2015) Studying the attributes of users in twitter considering their emotional states. Social Network Analysis and Mining 5(1):34

Schwarz G (1978) Estimating the dimension of a model. Ann Stat 6(2):461–464

Shahriari M, Sichani OA, Gharibshah J, Jalili M (2016) Sign prediction in social networks based on users reputation and optimism. Soc Netw Anal Min 6(1):91

Song D, Meyer DA (2015) Link sign prediction and ranking in signed directed social networks. Soc Netw Anal Min 5(1):52

Sun Y, Barber R, Gupta M, Aggarwal CC, Han J (2011) Co-author relationship prediction in heterogeneous bibliographic networks. In: International conference on advances in social networks analysis and mining, pp 121–128

Sun Y, Han J, Aggarwal CC, Chawla NV (2012) When will it happen?: relationship prediction in heterogeneous information networks. In: ACM international conference on web search and data mining, WSDM, pp 663–672

Symeonidis P, Mantas N (2013) Spectral clustering for link prediction in social networks with positive and negative links. Soc Netw Anal Min 3(4):1433–1447

Trivers RL (1971) The evolution of reciprocal altruism. Q Rev Biol 46:33–57

Tu K, Ribeiro B, Jensen D, Towsley D, Liu B, Jiang H, Wang X (2014) Online dating recommendations: matching markets and learning preferences. In: Proceedings of the 23rd international conference on world wide web, ACM, New York, NY, USA, WWW '14 Companion, pp 787–792

Tuna T, Akbas E, Aksoy A, Canbaz MA, Karabiyik U, Gonen B, Aygun R (2016) User characterization for online social networks. Soc Netw Anal Min 6(1):104

Valverde-Rebaza J, de Andrade Lopes A (2013) Exploiting behaviors of communities of twitter users for link prediction. Soc Netw Anal Min 3(4):1063–1074

Vinzamuri B, Reddy CK (2013) Cox regression with correlation based regularization for electronic health records. In: International conference on data mining (ICDM), pp 757–766

Wang P, Li Y, Reddy CK (2017a) Machine learning for survival analysis: a survey. ACM Computing Surveys

Wang Z, Wang C (2010) Buckley-James boosting for survival analysis with high-dimensional biomarker data. Stat Appl Genet Mol Biol 9(1):1–31

Wang Z, Chen C, Li W (2017b) Predictive network representation learning for link prediction. In: Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval, SIGIR '17, pp 969–972

Xia P, Ribeiro B, Chen C, Liu B, Towsley D (2013) A study of user behavior on an online dating site. In: 2013 IEEE/ACM international conference on advances in social networks analysis and mining (ASONAM 2013), pp 243–247

Xia P, Liu B, Sun Y, Chen C (2015) Reciprocal recommendation system for online dating. In: IEEE/ACM international conference on advances in social networks analysis and mining, ACM, ASONAM '15, pp 234–241

Xia P, Zhai S, Liu B, Sun Y, Chen C (2016) Design of reciprocal recommendation systems for online dating. Soc Netw Anal Min 6(1):32

Yang Y, Zou H (2013) A cocktail algorithm for solving the elastic net penalized cox regression in high dimensions. Stat Interface 6(2):167–173

Zang X, Yamasaki T, Aizawa K, Nakamoto T, Kuwabara E, Egami S, Fuchida Y (2017) You will succeed or not? matching prediction in a marriage consulting service. In: 2017 IEEE 3rd international conference on multimedia big data (BigMM), pp 109–116

Zhang B, Choudhury S, Hasan MA, Ning X, Agarwal K, Purohit S, Cabrera PGP (2016) Trust from the past: Bayesian personalized ranking based link prediction in knowledge graphs. In: SDM workshop on mining networks and graphs (MNG 2016)

Zhao K, Wang X, Yu M, Gao B (2014) User recommendations in reciprocal and bipartite social networks-an online dating case study. IEEE Intell Syst 29(2):27–35

Zhu YX, Zhang XG, Sun GQ, Tang M, Zhou T, Zhang ZK (2014) Influence of reciprocal links in social networks. PloS One 9(7):e103007

Zlatić V, Štefančić H (2009) Influence of reciprocal edges on degree distribution and degree correlations. Phys Rev E 80(1):016117

Zou H, Hastie T (2005) Regularization and variable selection via the elastic net. J R Stat Soc B 67:301–320