Topic-Based Influence Computation in Social Networks under Resource Constrains

Kaan Bingöl, Bahaeddin Eravcı, Hakan Ferhatosmanoğlu,
Buğra Gedik
Department of Computer Engineering
Bilkent University
Ankara, Turkey

Çağrı Özgenç Etemoğlu
AveaLabs
Istanbul,Turkey

Jul, 2015

Abstract

As social networks are constantly changing and evolving, methods to analyze dynamic social networks are becoming more important in understanding social trends. However, due to the restrictions imposed by the social network service providers, the resources available to fetch the entire contents of a social network are typically very limited. As a result, analysis of dynamic social network data requires maintaining an approximate copy of the social network for each time period, locally. In this , we study the problem of dynamic network and text fetching with limited probing capacities, for identifying and maintaining influential users as the social network evolves. We propose an algorithm to probe the relationships (required for global influence computation) as well as posts (required for topic-based influence computation) of a limited number of users during each probing period, based on the influence trends and activities of the users. We infer the current network based on the newly probed user data and the last known version of the network maintained locally. Additionally, we propose to use

link prediction methods to further increase the accuracy of our network inference. We employ PageRank as the metric for influence computation. We illustrate how the proposed solution maintains accurate PageRank scores for computing global influence, and topic-sensitive weighted PageRank scores for topic-based influence. The latter relies on a topic-based network constructed via weights determined by semantic analysis of posts and their sharing statistics. We evaluate the effectiveness of our algorithms by comparing them with the true influence scores of the full and up-to-date version of the network, using data from the micro-blogging service Twitter. Results show that our techniques significantly outperform baseline methods (80% higher accuracy for network fetching and 77% for text fetching) and are superior to state-of-the-art techniques from the literature (21% higher accuracy).

1 Introduction

Analysis of social networks have attracted significant research attention in recent years due to the popularity of online social networks among users and the vast amount of social network data publicly available for analysis. Applications of social network analyses are abound, such as influential user detection, community detection, information diffusion, network modeling, user recommendation, to name a few.

Influential user detection is a key social analysis used for opinion mining, targeted advertising, churn prediction, and word-of-mouth marketing. Social networks are dynamic and constantly evolving via user interactions. Accordingly, the influence of users within the network are also dynamic. Beyond the current influence of users, tracking the influence trends provides greater insights for deeper analysis. By combining the patterns of the past with the current information, comprehensive analysis on customers, marketing plans, and business models can be performed more accurately. For example, forecasting future user influences can be used to detect 'rising stars', who can be employed in upcoming on-line advertisement campaigns.

In this report, we address the problem of identifying and tracking influential users in dynamic social networks under real-world data acquisition resource limits. The current approaches for influence analysis mostly assume that the graph structure is static, or even when it is dynamic, the data is completely known and stored in a local database. However, in many cases, analysts are third-party clients and do not own the data. They cannot keep the data completely fresh as changes happen, since it is typically gathered from a service provider with limitations on resources or even on the amount of data provided. Third-party data acquisition

tools access the data via rate-limited APIs, which constraint the fetching capacity of clients. These externally enforced limits prevent the collection of entire up-to-date data within a predetermined period. To this end, we present an effective solution to rate-limited fetching of evolving network relations and user posts. Our system maintains a local, partially fresh copy of the data and calculates influence scores based on inferred network and text data. The proposed solution probes limited number of active users whose influence scores are changing significantly within the network. By combining previous and the newly probed network data, we are able to infer the current network accurately. The local network copy is maintained while consuming resources within allowed limits, and at the same time, influence values of the users are computed as accurately as possible.

While computing and maintaining influence scores, we consider both global and topic-based influence. Active and influential users mostly affect the general opinion with respect to their topics of authority. For instance, a company marketing sports goods will be interested in locating users who have high influence in sports, rather than the global community. While this leads us to consider topic-based analyses in our problem setting, general influence scores of users are still of interest as well. For instance, a politician would prefer a broader audience and identify a list of globally influential users to promote her cause. In our system, we utilize both global and topic-based networks and compute global as well as topic-based influences.

To demonstrate the effectiveness of our solutions, we use Twitter [39]. Twitter is a good fit for research on dynamic user influence detection due to its large user base and highly dynamic user activity. One can collect two-way friendship relations as well as one-way follow, re-tweet, and favorite relations via the publicly available Twitter APIs. These APIs have well-defined resource limits [40], which motivates the need for our probing algorithms. We calculate PageRank [32] on the Twitter network as the influence score for the users. To generate topic-based influence scores, we adapt the weighted PageRank [47], and adjust the initial scores and transition probabilities based on topic relevance scores of the users. The topic relevance scores are computed based on user posts, using text mining techniques, as well as the re-tweet and favorite counts of the tweets.

To further improve the accuracy of our network inference, we perform link prediction using trends on user relationships. The proposed solution shows increased accuracy on Twitter data when compared with other methods from the literature. Estimated network structure is shown to be very close to the actual up-to-date network, with respect to influential users. The proposed solutions address not only the limitations of data fetching via public APIs, but also local processing

when the resources are limited to fetch the entire data. We summarize our major contributions as follows:

- We estimate global and topic-based influence of users within a dynamic social network. For topic-based influence estimation, we construct topic-based networks via semantic analyses of tweets and the use of re-tweet and favorite statistics for the topic of interest.
- We propose efficient algorithms for collecting dynamic network and text data, under limited resource availability. We leverage both latest known user influence values, as well as the past user influence trends in our probing strategy. We further improve our probing techniques by applying link prediction methods.
- We evaluate our proposed algorithms and compare results to several alternatives from the literature. The experimental results for relationship fetching show that the proposed algorithms perform 80% better than the baseline methods, and 21% better than the state-of-the-art method from the literature in terms of mean squared error. For tweet fetching methods used for topic-based influence detection, our algorithms perform 77% better than the alternative baselines in terms of the Jaccard similarity measure.

The rest of this report is organized as follows. Section 2 describes the resource constraint problem for data collection. Section 3 gives the overall system architecture and presents influence estimation techniques. Section 4 explains algorithms and strategies proposed for the network and text fetching problems. Section 5 discusses results obtained from experiments run on real data. Section 6 discusses related work. Section 7 concludes the report.

2 Problem Definition

Our goal is to determine top-m influential users in the network, under a constrained probing setting. Among various methods to calculate a user's influence in the network, we have chosen PageRank based methods, since PageRank is well understood and used widely in the literature for various network structures. While computing influence, PageRank naturally considers the number of followers a user has, but more importantly it takes into account the topological place of the user within the network. Therefore, we assume that a user's influence in the network corresponds to its PageRank score. As a result, the top-m influential user determination problem turns into identifying the top-m users with the highest PageRank

scores. One can also utilize other approaches that can outperform PageRank for estimating social influence within our framework. These approaches need to produce a single score that will be calculated periodically for every user.

PageRank score calculation requires having access to all the relationships present between the users of the network. This means that we need to have the complete network data to compute exact PageRank scores. Moreover, if the network is dynamic, the calculation needs up-to-date network data for each time step in order to perform accurate influence analysis.

Our system continuously collects social network data (relations, tweets, retweets, etc.) via the publicly available Twitter API. Twitter enforces certain limitations on data acquisition using the Twitter APIs. There are different limitations for different types of data acquisition requests:

- *Relations*: 15 calls per 15 minutes, where each call is for retrieving a user's relations. Moreover, if the user has more than $5K$ followers, we need an extra call for each additional $5K$ followers. This means that we can update relations with a maximum rate of 1 user per minute ($R_{rel} = 1$ user/min).

- *Tweets*: 180 calls per 15 minutes, where each call is for retrieving a user's tweets. Moreover, if the user has more than $3.2K$ tweets, we need an extra call for each additional $3.2K$ tweets. This means that we can update tweets with a maximum rate of 12 users per minute ($R_{twt} = 12$ user/min).

Assuming that we update the network with a period of P days, we need the following condition to hold, in order to be able to capture the entire network of relations:

$$\text{Number of Users} \leq R_{rel} \cdot P \cdot 1440 \tag{1}$$

For getting the recent tweets of the users, we need:

$$\text{Number of Users} \leq R_{twt} \cdot P \cdot 1440 \tag{2}$$

One can easily calculate that for a network as small as $250K$ users, we need 174 days to update the complete network in the best case[1]. This analysis shows that the rate limits hinder the timeliness of the data collection process, which in turn affects the timeliness of the calculation process to find and track influential users in the network. Furthermore, Twitter is a highly dynamic network that evolves at a fast rate, which means that refreshing the network infrequently will

[1] if all users have $\leq$ 5K followers, requiring a single call per user.

result in significant degradation in the accuracy of the influence scores. Current resource limits prohibit the system to collect the network data in a reasonable period of time. Therefore, the evolving network's relationships and the tweet sets are not fully observable at every analysis time step.

To overcome this limitation, we propose to determine a small subset of users during each data collection period, whose information is to be updated. This data collection process, which does not violate the rate limits of the API, is sufficient to maintain an approximate network with a reasonable data collection period, while at the same time providing good accuracy for the estimated influence scores.

We apply the concept of *probing* for efficient fetching of the dynamic network and the user tweets. We denote a network at time t as $G_t = \{V_t, E_t\}$, where V_t is the set of users and $E_t \subset V_t \times V_t$ is the set of edges representing the follower relationship within the network. In other words, $(u, v) \in E_t$ means that the user $u \in V_t$ is following the user $v \in V_t$. Our model uses an evolving set of networks in time, represented as $\{G_t \mid 0 \leq t \leq T\}$. However, we assume that we have fully[2] observed the network only at time $t = 0$. G_t where $t > 0$, can only be observed partially by probing. At each time period, we use an algorithm to determine a subset of k users and probe them via API calls. We then update the existing local network with the new information obtained from the probed users. In effect, we maintain a partially observed network G'_t, which can potentially differ from the actual network G_t. Larger k values bring the partial network G'_t closer to the actual network G_t. However, using large k values is not feasible due to rate limits outlined earlier. Our probing strategy should select a relatively small number of users to probe, so that the data collection process can be completed within the period P (as determined by Eq. 1). Furthermore, these probed users should bring the most value in terms of performing accurate influence detection.

Dynamic Network Fetching Problem Definition: We assume that complete network information is available only at time 0, i.e., G_0 is known. The problem is defined as determining a subset of users of size k at time t, denoted by $U_t \subset V_t$ s.t. $|U_t| = k$, by analyzing the local graph G'_{t-1}. The system will update the relationships of the users included in this subset to construct the local network at time t, that is G'_t. Specifically, this new network G'_t is constructed by replacing the relationships of the users in G'_{t-1} with the newly fetched relationships from the probing of the users in U_t. We aim to choose U_t such that the influence

[2]The initial probing of the network can be accelerated via the use of multiple cooperating fetchers. However, this is clearly not a sustainable and feasible approach for continued probing of the network, as it requires large number of accounts, which are subject to bot detection and suspension.

scores of the estimated network G'_t will be as close as possible to the true scores of the real network G_t. The final objective is to estimate the PageRank scores $PR'_v(t), \forall v \in G_t$ as accurately as possible, using partial knowledge about G_{t-1}, that is G'_{t-1}.

In order to track topic-specific influence scores of the users, we analyze their latest tweets. One needs to collect predetermined amount of tweets for all of the users to be able to compute exact influence scores. However, due to the rate limitations (see Eq. 2), we cannot fetch all the tweets within the desired period. Instead of retrieving tweets of every user, we determine a subset of users so that by collecting tweets of this subset, the topic scores of the users will be as close to the true scores as possible. We denote the tweet set at time t as T_t. We again assume that we have observed this set fully only at time 0, that is T_0 is known. The other snapshots can only be observed partially by probing. I.e., we locally maintain partial tweet sets T'_t, where $t > 0$.

Dynamic Tweet Fetching Problem Definition: Given the tweets T_0 of all users in the network at time 0, the problem is defined as determining a subset of users of size k at time t, denoted by $U_t \subset V_t$ s.t. $|U_t| = k$, by analyzing the tweet set T'_{t-1} and local graph G'_{t-1}. By collecting tweets of the users included in U_t, we construct an approximate tweet set T'_t and update the topic-based network accordingly. The final objective is to estimate the topic-based influence scores of the users in the network as accurately as possible. Thus, the goal is to pick the subset U_t, so as to maximize the accuracy of the influence scores computed on the estimated topic-based network.

3 Overall System Architecture

In this section we briefly describe our system architecture, which depicted in Figure 1.

3.1 Social Network Data Collection

We use the Twitter network and tweets to analyze user influence. A Twitter network is a directed, unweighted graph where the nodes represent users and the edges denote follower relationships in Twitter. When a user u follows a user v, it is obvious that v would have a influence on u. Moreover, the user u also would have an effect on v's influence, since the number of people v reaches would potentially increase. This interaction has an effect on both users' influence scores. In order to construct our network, we first determine a small set of users called the *core seeds*. For illustration, we started with some popular Turkish Twitter accounts

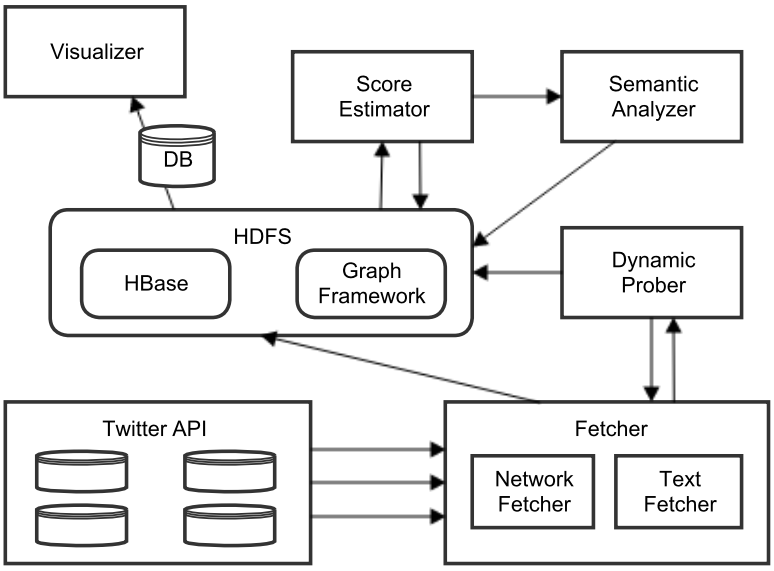


Figure 1: Overall system architecture.

including newspapers, TV channels, politicians, sport teams, and celebrities. Second, we collect one-hop relations of the core seeds and add the unique users to a set called the *main seeds*. We iterate once more to collect one-hop relations of the main seeds with a filter to avoid unrelated and inactive users. This filter has three conditions: *a*) a user must have at least five followers, *b*) a user must have at least one tweet within the last three months, and *c*) the tweet language of a user must be Turkish. As a result of this process, we have determined our *seed users* set, which includes approximately 2.8 million unique users. In the final step of the data collection phase, we acquire the relations of the seed users to determine G_0, that is the social network graph at time 0. Furthermore, we collect tweets of the seed users in order to construct T_0, that the tweet set at time 0.

We implemented the proposed methods using a distributed system with HBase and HDFS serving as the database and file system backends. The system consists of six main parts: *a*) local copy of the social network data on HDFS, *b*) data fetcher, *c*) dynamic prober, *d*) score estimator, *e*) semantic analyzer, and *f*) visualizer. Data fetcher component, as the name implies, fetches the data (network relations and tweets) via rate-limited Twitter APIs, periodically. Dynamic prober makes a dynamic probing analysis, decides which users are going to be fetched and notifies data fetcher to bring the information, accordingly. Score estimator calculates users' influence and the related parameters of the proposed algorithms,

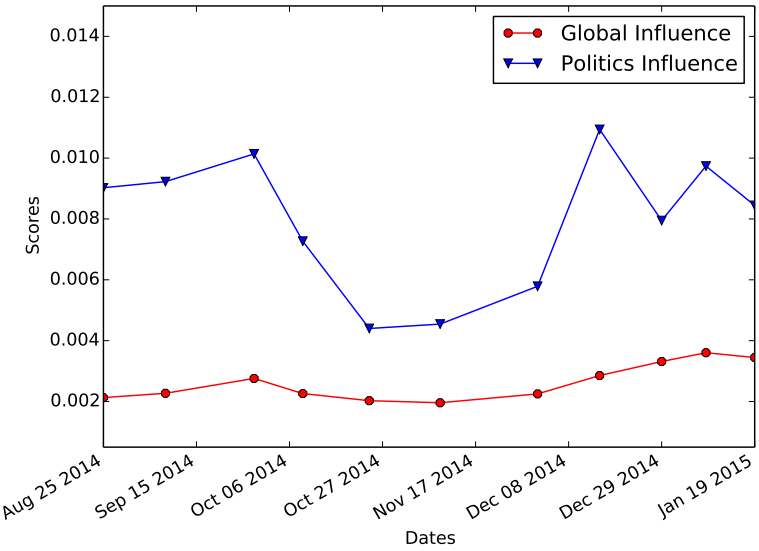


Figure 2: Past influence scores of a user

which are essential parts of the probing method. Semantic analyzer performs keyword extraction and calculates the related parameters for constructing topic-based networks. Finally, visualizer provides a graphical user interface for result analysis.

3.2 Score Analysis

We calculate influence scores of users based on their relationships and the overall impact of their tweets in the network. We analyze topic activities of the users from their tweets and determine topic-based user influence scores. Overall, we are using two types of scores, namely *global influence* and *topic-based influence*, which can be interpreted together for a more detailed analyses.

Global Influence Score. This score is a measure of the user's overall influence within the network. For this purpose we use the PageRank (PR) algorithm. PageRank value $PR_v(t)$ at time t for a user $v \in G_t$ directly corresponds to the global influence score of it and will be used interchangeably throughout the report.

Figure 2 illustrates the evolving nature of the influence score by showing the global and topic-based influence scores history of a user, which is selected by our algorithm as one of the most important users that should be probed during the first collection period. This is the official account of the president of the Republic of Turkey. Besides the account's high impact, we observe that its influence also varies significantly over time, which further justifies the need to probe this

account frequently. A reason of the variation in influence score is that the time period shown in the figure matches with the elections for the Presidency (10 August 2014). After becoming the new president, the account's influence has further increased. During this period, it is always selected as a top user to be probed by our proposed approach. This is intuitive, as it is a popular account with changing influence scores over time.

Topic-Based Influence Score. The system calculates topic-based influence scores representing user activity and impact on a specific topic. We perform semantic analysis on user tweets by taking re-tweets and favorite counts into consideration as well. A re-tweet (RT) is a re-posting of someone else's tweet, which helps users quickly share a tweet that they are influenced by or like. A favorite (FAV) is another feature that represents influence relation between users, wherein a user can mark a tweet as a favorite. These two features help estimate the influence of an individual tweet. Since Twitter is a micro-blogging platform, users are generally tweeting on specific topics. While many tweets are mostly conversational and reflect self-information [29, 1], some are being used for information sharing, which is important in harvesting knowledge. RTs and FAVs are effective in separating relevant and irrelevant tweets. Accordingly, we use them in our topic weight analysis to estimate influence of a tweet on a specific topic.

Topic-based network construction process consists of three main phases: *a*) keyword extraction on tweets, *b*) correlation of keywords with topic dictionaries, and *c*) weight calculation.

In the first phase, keywords are extracted from tweets by using information retrieval techniques, including word stemming and stop word elimination. The output from this phase is a keyword analyzed tweet corpus for each individual user and the related histogram which captures the frequencies of the related keywords (K). These corpora are further analyzed in the second phase.

We have created a keyword dictionary (D_j) for each topic (C_j), in order to score tweets against topics. As part of each dictionary, we have assigned normalized weights to words, representing their topic relevance. In the second phase, using the weights from the dictionaries and the users' keyword histograms, we obtain the normalized raw topic scores of users for each one of the topics.

In the third phase, we calculate a value called the RT-FAV total for each user, which is the summation of the number of re-tweets and favorites received by a user's tweets. We then multiply the normalized raw topic score by the RT-FAV total of the user, in order to find the number of RT-FAVs the user gets on a topic of interest. The final normalized results are used as the in-edge weights of the users

on each topic, when forming the topic-based network.

Once the topic-based network construction is complete, we execute the weighted PageRank [47] (WPR) algorithm which also considers the importance of the incoming and outgoing edges in the distribution of the rank scores. The resulting weighted PageRank values of users, denoted by $WPR_v(t)$ at time t for $v \in G_t$, is assigned as their topic-based influence scores.

Due to the nature of the PageRank algorithm, some of the globally influential users also turn out to be highly influential for most or all of the topics. These users have a lot of followers and they are also followed by some of the influential accounts of the specific topics, which cause them to score high for topic-based analysis as well. Therefore, they can get high topic-based influence scores even if they do not actively tweet about the topic itself. To eliminate this effect, we apply one more level of filtering to remove these globally effective accounts from the topic-sensitive influence lists. In particular, if the number of tweets a user posted that are related with the topic at hand is less than a predefined percentage, e.g., $\%40$[3], of the total number of tweets posted by the user, then the user is discarded for that topic. This filtering process significantly reduces the noise level in the analysis.

As a result, for each topic, we construct a weighted network in which an edge ((u, v)) represents the amount of topic-specific influence a user (v) has on a follower user (u). Thus, the results of weighted PageRank algorithm gives us the overall topic-influence scores on the network.

Figure 2 also shows the topic-based score history of the official account of the president of the Republic of Turkey. According to our analysis, $\%80$ of the account's topic activity is related to politics. Since it could not pass our applied activity filter on other topic categories, the system only calculates its topic influence scores for politics. We can see from the figure that the change on the topic-based scores are more dramatic compared to the global scores. This is intuitive, as the topic-sensitive scores are depending on users' tweets and sharing statistics. A user might be very active on some weeks about a specific topic so that her influence on the topic might increase dramatically. Likewise, when she posts something important, it might achieve high sharing rates. On the other hand, when she just posts regular things which are not shared, her influence on the topic might decrease quickly.

[3]Note that a tweet can be related to zero or more topics.

4 Dynamic Data Fetching

In this section, we introduce our algorithms for probing dynamic social networks. In order to efficiently determine a subset of vertices to probe, we develop heuristics for both dynamic network fetching and dynamic tweet fetching problems given in Section 2.

Since we have chosen the PageRank score as the indicator of influence in a social network, we analyze its change as the network evolves. PageRank value of a specific vertex v is given as follows:

$$PR(v) = \alpha \sum_{\forall (u,v) \in E_{in}(v)} \frac{PR(u)}{|E_{out}(u)|} + \frac{1-\alpha}{n}, \tag{3}$$

where $PR(v)$ denotes the PageRank value, $E_{in}(v)$ denotes the in-edge set, and $E_{out}(v)$ denotes the out-edge set for v.

Figure 3 shows an example network, which will be used to demonstrate the effects of network changes on PageRank values.

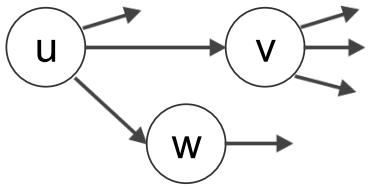


Figure 3: A sample graph for analysis.

Assume that an edge (u, v) is added due to the evolving nature of the network. Here, we analyze the effect of this addition on the PageRank values of the out neighbors of u. We see that the PageRank value of v is as follows per Equation 3:

$$\begin{aligned} PR^{new}(v) &= \alpha \left(\sum_{\forall (i,v) \in E_{in}(v)} \frac{PR(i)}{|E_{out}(i)|} + \frac{PR(u)}{|E_{out}(u)|+1} \right) + \frac{1-\alpha}{n} \\ &= PR(v) + \alpha \frac{PR(u)}{|E_{out}(u)|+1} \end{aligned}$$

We can easily extend this analysis to multiple new edges since the total effect will be a superposition of the effect of the new individual in-edges of vertex v.

$$PR^{new}(v) = PR(v) + \alpha \sum_{\forall(u,v)\in E_{in}^{new}(v)} \frac{PR(u)}{|E_{out}(u)|+1}$$

PageRank values of out neighbors of u other than v, such as w, are impacted as follows:

$$PR(w) = \alpha \left(\sum_{\forall(i,w)\in E_{in}(w)\setminus(u,w)} \frac{PR(i)}{|E_{out}(i)|} + \frac{PR(u)}{|E_{out}(u)|} \right) + \frac{1-\alpha}{n}$$

$$PR^{new}(w) = \alpha \left(\sum_{\forall(i,w)\in E_{in}(w)\setminus(u,w)} \frac{PR(i)}{|E_{out}(i)|} + \frac{PR(u)}{|E_{out}(u)|+1} \right) + \frac{1-\alpha}{n}$$

$$PR^{new}(w) = PR(w) - \alpha \frac{PR(u)}{|E_{out}(u)|.(|E_{out}(u)|+1)}$$

These effects are the immediate responses on the vertices that are considered. These residual PageRanks will ripple out to all the vertices in all the paths from v and w in each iteration of the PageRank algorithm. But the effect will decease as the residuals will be divided by the number of outgoing edges for each vertex visited. We will analyze the effects of the first iteration of the algorithm to simplify the problem and to get a general feel of the change in PageRank values. Considering expected value of $\overline{E_{out}} = E[|E_{out}(u)|]$ as the average out-degree for vertices, the differential PageRanks are given as follows:

$$\nabla PR(v) = \alpha \frac{PR(u)}{\overline{E_{out}}} \tag{4}$$

$$\nabla PR(w) = -\alpha \frac{PR(u)}{\overline{E_{out}}^2} \tag{5}$$

We can see from Equations 4 and 5 that we should select the vertices, say u, with the following properties for accurate G'_t and $PR'_u(t)$ estimations:

- vertices with high PageRank values ($PR(u)$);
- vertices whose PageRank values change over time;
- vertices with high out-degrees ($E_{out}(u)$);
- vertices whose out-degrees change over time.

PageRank, when computed until the values converge in steady state, considers both incoming and outgoing edges. The parameters related to out-degree values are intrinsically taken into account when PageRank is computed. Hence, in our dynamic fetching approach, we focus only on PageRank values and their changes to cover all the cases listed above.

Based on these observations, we will define a utility function that incorporates the above findings. We will find the vertices that maximize this utility function, which will be probed and used to estimate the influence scores of the evolving network. We analyze two sub-problems of the general case specific for our application: network fetching and tweet fetching. These sub-problems and the solutions will be addressed in the subsequent sections.

4.1 Dynamic Network Fetching using Influence Past

We aim to probe a subset, U_t, update the edges incident on vertices in U_t to form G'_t, and calculate PageRank values $PR'_v(t),\ \forall v \in G_t$. In order to determine this subset, we use a time series of past PageRank values for a vertex v, named the *influence past* of v. Formally, we have $IP_v = [\ldots, PR'_v(t-2), PR'_v(t-1)]$.

In our strategy for determining U_t, we consider the vertices whose PageRank values change considerably over time. We first explored building time-series models over sequences of scores to forecast their future values. There are some well-known methodologies in the literature for forecasting using this kind of time-series data, such as ARIMA models. However, these models typically require much longer sequences for accurate predictions. Therefore, in order to quantify this *change* for a vertex v, we calculate the standard deviation of the time series IP_v, that is:

$$Change_v = \sigma_{IP_v} = \sqrt{Var(PR'_v)} \tag{6}$$

Choosing the best vertices to probe can be performed by calculating a score that is a linear combination of the PageRank value and the change in PageRank values, as given in Equation 7. Here, θ parameter balances the importance of the two aspects. We assume that influence past that contains at least two data points is available for every user, in order to calculate the score changes.

$$Score(v) = (1-\theta)PR'_v(t-1) + \theta\, Change_v \tag{7}$$

After the selection of the users with respect to the ranking of $Score(v)$, we probe their current relations and form G'_t.

Round-Robin & Change Probing. Change Probing could cause the system to

focus on a particular portion of the network and may discard the changes developing in other parts. This is because the probing scores of some vertices will be stale and as a result these vertices may consistently rank below the top-k, despite changes in their real scores. This bias could end up accumulating errors in the influence scores of these vertices and start to have an impact on the entire network. Therefore, we propose to use Change Probing together with Round-Robin Probing, in which users are probed in a random order with equal frequency. In this way, we aim to probe every vertex at least once within a specific period P. Round-Robin Change algorithm probes some portion of the network randomly and marks all probed users. Thus, any probed users are not probed randomly again, until all users are probed at least once within P. In this method, we control the balance between change vs. random selection by using a parameter $\beta \in [0, 1]$. In particular, we choose $\beta * k$ users to probe with Change Probing and $(1 - \beta) * k$ users with Round-Robin Probing.

Network Inference. Since we are able to fetch data only for a limited number of users, there is a high probability that other users in the network have changed their connections as well. To take these possible changes into account, we have incorporated *link prediction* into our solution. Link prediction algorithms assign a score to a potential new edge (u, v) based on the neighbors of its incident vertices, denoted as Γ_u and Γ_v. The basic idea behind these scores is that the two vertices u and v are more likely to connect via an edge if Γ_u and Γ_v are similar, which is intuitive. Considering social networks, two people are likely to be friends if they have a lot of common friends. There are different scores used in the literature, including the common neighbors, Jaccard's coefficient, Adamic/Adar, and Resource Allocation Index (RA). We use RA as part of our approach, since it was found successful on a variety of experimental studies on real-life networks [26]. One could also adopt more advanced prediction algorithms such as [2], in order the increase effectiveness of this approach.

RA is founded on the resource allocation dynamics of complex networks and gives more weight to common neighbors that have low degree. For an edge (u, v) between any two vertices u and v, RA is defined as follows:

$$RA_{u,v} = \sum_{w \in \Gamma_u \bigcap \Gamma_v} \frac{1}{degree(w)}, \quad \text{where } \Gamma_v \text{ is the neighbors of } v \tag{8}$$

The RA score, $RA_{u,v}$ for the edge (u, v), is proportional to the probability of an edge being formed between the vertices u and v in the future. Based on this, we

ALGORITHM 1: Algorithm for Dynamic Network Fetching

Input: G'_{t-1}, IP, $PR'(t-1)$, θ, $\beta \in [0,1]$, k
Output: G'_t
// Fetch network
for all $v \in V_t$ **do**
  $\sigma_{IP_v} = \sqrt{Var(PR'_v)}$
  $Score(v) = (1-\theta)PR'_v(t-1) + \theta \cdot \sigma_{IP_v}$
end for
$U_t \leftarrow \emptyset$
while $|U_t| \leq k \cdot \beta$ **do**
  $v \leftarrow argmax_{v \in V_{t-1}} Score(v)$
  $U_t \leftarrow U_t \cup \{v\}$, $V_{t-1} \leftarrow V_{t-1} \setminus \{v\}$
end while
while $|U_t| \leq k$ **do**
  $v \leftarrow$ randomly choose from V_{t-1}
  $U_t \leftarrow U_t \cup \{v\}$, $V_{t-1} \leftarrow V_{t-1} \setminus \{v\}$
end while
Probe U_t for relationships, Form G'_t
// Infer network
Calculate $RA_{u,v}, \forall (u,v) \in \widetilde{E} = V_t \times V_t$
for E_g times **do**
  $(u,v) \leftarrow argmax_{(u,v) \in E_t} RA_{u,v}$
  $E_t \leftarrow E_t \cup \{(u,v)\}$
end for
Output G'_t

rank all the calculated RA scores. Since the edges in our network are not defined probabilistically and are defined deterministically as existent or non-existent, we need to determine how many of these scored edges should be selected. Therefore, we define a growth rate, E_g, which is the average change in the number of edges ($|E|$) between snapshots of the network after excluding the changes due to U_t. After calculating RA scores for all possible new edges, we choose E_g edges with the highest scores. Using this method, we add new connections to the current graph, to finally have the estimated graph G'_t. The pseudo code of the network inference based probing algorithm we use to select k vertices to probe is given in Algorithm 1.

ALGORITHM 2: Dynamic tweet fetching via G-WG

Input: $T_{t-1}^{j'}, TIP^j, WPR^{j'}(t-1), \theta, \beta \in [0,1], k$
Output: $T_t^{j'}$
for all C_j **do**
  for all $v \in V_{t-1}^j$ **do**
    $\sigma_{TIP_v} = \sqrt{Var(TPR'_v)}$
    $Score^j(v) = (1-\theta)WPR_v^{j'}(t-1) + \theta \cdot \sigma_{TIP_v^j}$
  end for
  $U_t^j \leftarrow \emptyset$
  while $|U_t^j| \leq k \cdot \beta$ **do**
    $v \leftarrow argmax_{v \in V_{t-1}^j} Score^j(v)$
    $U_t^j \leftarrow U_t^j \cup \{v\}, V_{t-1}^j \leftarrow V_{t-1}^j \setminus \{v\}$
  end while
  while $|U_t^j| \leq k$ **do**
    $v \leftarrow$ randomly choose from V_{t-1}^j
    $U_t^j \leftarrow U_t^j \cup \{v\}, V_{t-1}^j \leftarrow V_{t-1}^j \setminus \{v\}$
  end while
  Probe U_t^j for tweets, Form $T_t^{j'}$
  Output $T_t^{j'}$
end for

4.2 Dynamic Tweet Fetching using Topic-Based Influence Past

Our dynamic tweet fetching solution makes use of the weighted PageRank values and comprises of two steps. First, we infer the evolving relationships of the network using the methods explained earlier in the previous section. This way we can track and estimate the changing relationships. Second, we select a subset of users to fetch their tweet data. Specifically, we aim to probe a subset, U_t, collect their tweets, and update the edge weights for the users in U_t; all in order to form $WG_t^{j'}$ for a given topic C_j. We then compute weighted PageRank values to find $WPR_v^{j'}(t), \forall v \in WG_t^j$ for a given topic C_j. To select the subset of users in U_t, we use a time series of the past weighted PageRank values, named the *topic-based influence past* of v. Formally, we have $TIP_v = [\ldots, WPR_v^{j'}(t-2), WPR_{v_i}^{j'}(t-1)]$. This is performed independently for all topics of interest, $\{C_j\}$.

There are two different approaches we employ to track the topic-based influence scores:

- Use the global network parameters for network fetching and the topic-

sensitive network parameters for tweet fetching. This is named as the G-WG method, where global G_t is used for network fetching, and topic-sensitive WG_t is used for tweet fetching.

- Use the topic-sensitive network parameters for both network and tweet fetching. This is named as the WG-WG method.

The first approach, G-WG, is useful for cases where globally influential users are tracked, but with minimal additional resources, topic-based influential users are to be determined as well. This might be the only viable option if the bandwidth is not enough for selecting and updating the vertices separately for each topic, especially if the number of topics is high. For the second approach, that is WG-WG, we construct separate networks WG^j for each topic and evolve them separately. We update each network at the end of a probing period, using the new tweets fetched to track the most influential vertices for each topic C_j. The high-level algorithm for the G-WG method is given in Algorithm 2. The algorithm for WG-WG is very similar, and is omitted for brevity.

5 Experiments and Results

In this section, we present the experimental setup and the results of our evaluation of the proposed algorithms. We also present experiments analyzing the sensitivity of the parameters used.

5.1 Data Sets

We collected data using the public Twitter API, as described in Section 3. These API calls are restricted by rate limit windows. These windows represent 15 minute intervals and the allowed number of calls within each window can vary with respect to the call type. Our system makes two different calls, *a*) "GET followers/ids", which returns a cursored collection of user IDs for every user following the specified user, and *b*) "GET statuses/user_timeline", which returns a collection of the most recent Tweets posted by the specified user.. For the first call type, we are allowed to make 15 calls per window. Every call can return up to 5K followers. For the users who have more than 5K followers, we have to make multiple calls, accordingly. For the second type, we are allowed to make 1804 calls per window. Each call can return 3.2K tweets of the queried user. Details of the calls are also presented in Section 2 with the accompanying analysis.

We collected the network between the end of August 2014 and the beginning of January 2015, with a period of 15-20 days. As a result, we have obtained

11 snapshots of the Turkish users' network with progressing timestamps. We collected the relations of 2.8 million users, which amounts to a total of 310 million edges on average. We took the first snapshot as the initial network to calculate the probing scores (see Eq. 7) and the rest of the snapshots were used as ground truth for the evaluation of the probing algorithms. For the topic-based influence estimation, we also collected the tweets of our seed users in the same period. We constructed a dataset formed of 11 snapshots containing 5.5 billion tweets in total. We take the first snapshot as the initial tweet set as in the case of the relationship network analysis. From this data, we built up the topic weighted networks and calculated probing scores (see Eq. 7), accordingly.

In our probe simulation module, we fetch the connections of the users we have selected for probing, from the real network G_t at time t. We then update these connections (adding new ones and deleting old ones) on the previously observed network G'_{t-1} at time $t-1$, in order to obtain the estimated network G'_t at time t. Finally, we compare the influence estimation results from the observed network G'_t with the ones from the real network G_t. Same procedure is also applied for the tweet sets.

In order to include extensive number of experiments in our evaluation, we focused on the top $250K$ influential users and restricted the network on which the scores are computed to the network formed by these users.

Figure 4 shows the in-edge distribution of the original and the pruned network. Both follow a power-law distribution. Impact of the pruning process on the network structure seems to be minimal and has not created any anomalies in the analysis. We also pruned the tweet list according to the same top $250K$ influential users, which reduced the total size of the tweet sets to $200M$.

5.2 Evaluation of Dynamic Network Fetching

We have implemented several algorithms to compare the performance of the proposed techniques. The details of the algorithms used are given as follows:

NoProbe and **Random Probing**. These are two baseline algorithms. *NoProbe* algorithm assumes that the network does not change over time and uses the fully observed network at time $t = 0$ for all time points without performing any probing. It represents the worst case scenario for dynamic network fetching. The second baseline algorithm is *Random Probing* algorithm which randomly chooses k users to probe with uniform probability.

MaxG. As described in [51], users are probed with a probability proportional to the "performance gap", which is defined as the predicted difference between the

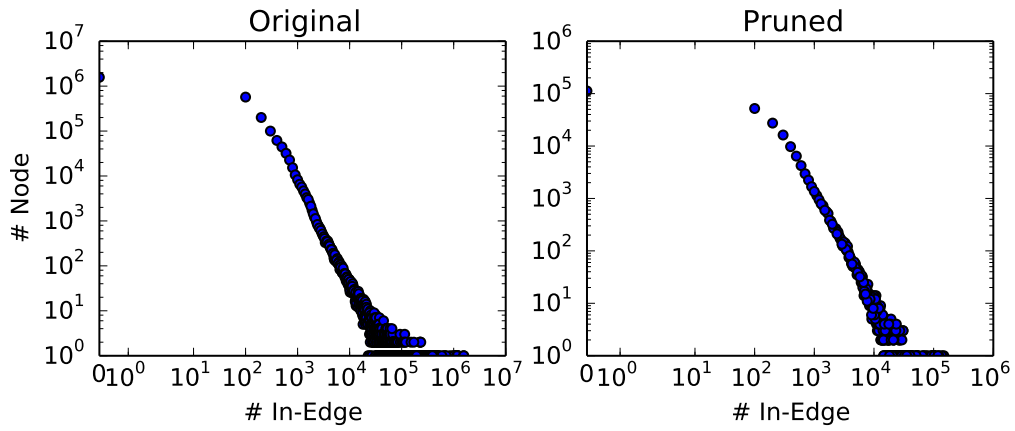


Figure 4: In-edge distributions of the original network (on the left) and the pruned network (on the right).

results of the approximate solution and the real solution. Briefly, the method incrementally probes users which will bring the largest difference in the results. It assumes that the influence of a specific user is related to the output of the *degree discount heuristic*. Although their influence determination function is different than ours, we use the MaxG algorithm for performance evaluation of our proposed algorithms.

Priority Probing. As described in [3], this algorithm chooses users to probe according to a value proportional to their priorities. Priority of a node is defined as the value of its PageRank score. For every iteration of the method, if a node is not probed, the current PageRank value is added to its priority and if the node is probed, its priority is reset to 0.

Change Probing. This is our first proposed method, which chooses k users to probe with value proportional to their scores, as computed by Eq. 7. The network is then constructed via Alg. 1.

Round-Robin & Change Probing. This is our second proposed method, which chooses $\beta \cdot k$ users to probe with Change Probing and $(1-\beta) \cdot k$ users with Round-Robin Probing. When $\theta = 0$ in Eq. 7 for the Change Probing part, the method becomes similar to [3]. The difference is that Priority Probing increases the probe possibility of a node by its PageRank value in every step if it is not probed, so that at some point the probe possibility becomes 1.

We evaluate performance by comparing the quality of the influential users found by each approach with that of the ideal case. For this purpose, we use two

different evaluation measures:

- Jaccard similarity between the correct and estimated top-k most influential users lists.
- The mean squared error (MSE) of the PageRank scores.

5.3 Evaluation of Dynamic Tweet Fetching

We evaluate the performance of the proposed tweet fetching technique with two baselines algorithms, namely *NoProbe* and *Random Probing*. The details of these baselines are given below:

NoProbe. This algorithm assumes that the tweet set does not change over time and use the fully observed tweet set at time $t = 0$ for all time points without any probing. This method represents the worst case scenario for the dynamic tweet fetching problem.

Random Probing. This algorithm randomly chooses k users to collect tweets with uniform probability at each time step.

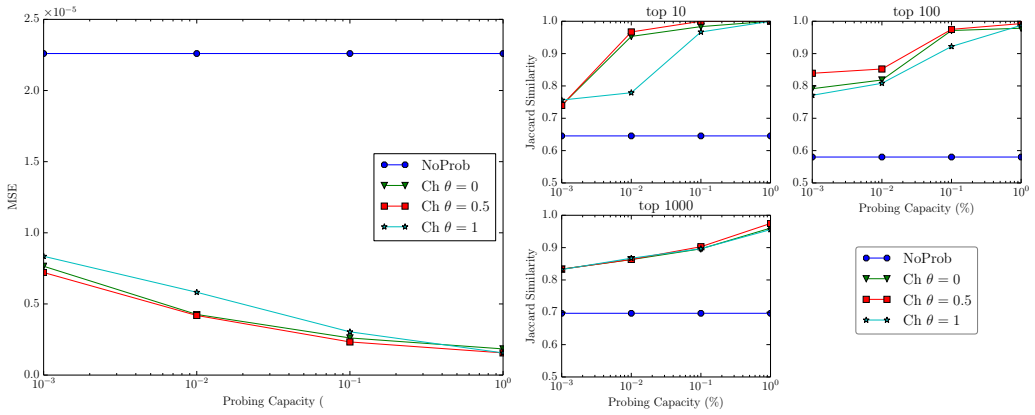
Round-Robin & Topic Change Proportional Probing. This is the algorithm we proposed, which greedily chooses k users to collect tweets with value proportional to their scores describe in Eq. 7. Differently from the network fetching method, scores are calculated by using WPR_v^j for the topic C_j, instead of PR_v.

5.4 Experimental Results and Discussion

This section compares and discusses the performance of the proposed network and tweet probing methods with the state-of-the-art and baseline methods using experiments executed on real datasets. We also provide an empirical interpretation of the calculated topic-based influence scores.

5.4.1 Experimental Setup

As indicated by Eqs. 1 and 2, given the resource limits permitted by the service providers, one cannot probe a significant portion of the network. We have executed our experiments with different probing capacities and used 0.001%, 0.01%, 0.1% and 1% of the network as the size of the probe set. For the analysis of the effect of the θ parameter used in Change Probing, we set: *a*) $\theta = 0$, meaning PageRank proportional scores are used; *b*) $\theta = 0.5$, meaning equally weighted PageRank and influence past scores are used; *c*) $\theta = 1$, meaning only influence past scores are used. For the Round-Robin Change algorithm we tested the ratio parameter β with three values, which control the fraction of vertices proved via random selection: 0.4, 0.6, and 0.8.



(a) MSE (b) Jaccard similarity

Figure 5: Performance of Change Probing.

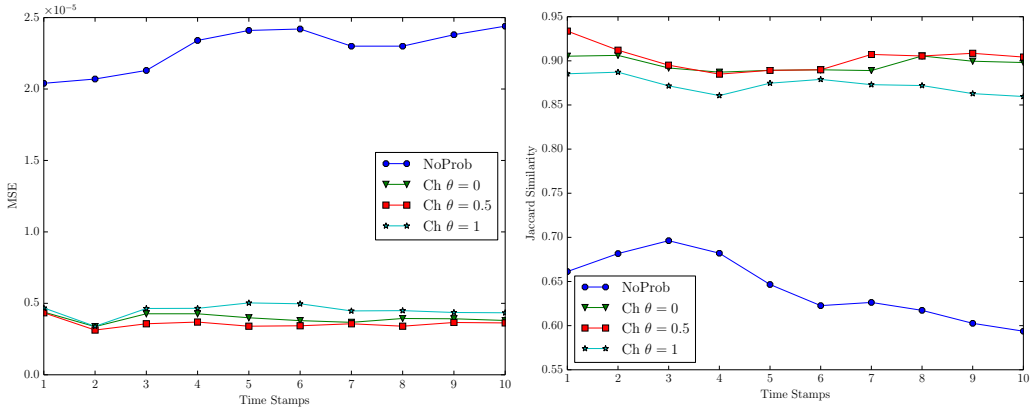
5.4.2 Change Probing Performance w.r.t. θ

Figure 5 depicts the performance of Change Probing algorithm for the Jaccard similarity measure. As expected, Change Probing algorithm significantly outperforms NoProb algorithm. For the optimization of the θ parameter, we test Change Probing algorithm under three different θ configurations:

- Using the MSE measure, $\theta = 0.5$ setting performs 8% better than $\theta = 0$ setting and 19% better than $\theta = 1$ setting. Overall, it performs 83% better than NoProbing.

- Using the Jaccard distance measure, $\theta = 0.5$ setting is 3% better than $\theta = 0$ setting and 5% better than $\theta = 1$ setting. In the overall case, $\theta = 0.5$ outperforms NoProbe by 43%. We also note that as the probing capacity increases, performance of the Change Probing algorithm becomes less dependent on the setting of θ.

We also illustrate the change in error as the network evolves, in order to see how the performance of different algorithms are affected as the seed network data ages. Figures 6a and 6b show the performance of Change Probing as a function of time for the mean squared error (MSE) and Jaccard similarity measures, respectively. We observe that NoProb has an increasing error as time passes. Change Probing gives a more robust and stable performance with respect to time. This is mainly because as the number of past influence points increases, the algorithm



(a) MSE

(b) Jaccard similarity

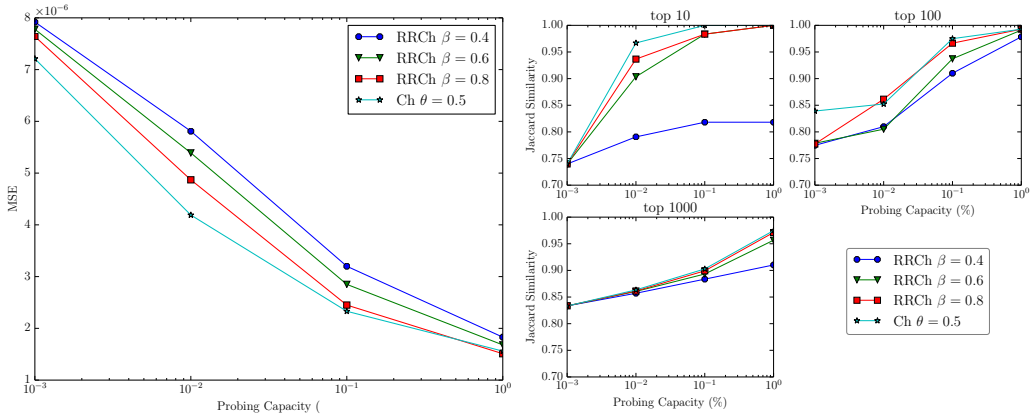
Figure 6: Performance of Change Probing as a function of time.

can estimate the influence variability of the users more accurately, which compensates the deteriorating effect of aging of the baseline network data. Since $\theta = 0.5$ outperforms the other cases, we use $\theta = 0.5$ configuration in the subsequent experiments with other algorithms. We also note that y-axis contains relatively small values because the PageRank values are normalized. We have assumed NoProb algorithm as the reference point for normalization.

5.4.3 RR Change Probing Performance w.r.t. β

Figure 7 shows the performance results for the Round-Robin Change (RRCh) Probing algorithm under different round-robin ratios. We use the Change Probing algorithm (with $\theta = 0.5$ setting) as the baseline reference point.

We observe that the RRCh algorithm performs poorly for small probing capacities, such as 0.001% and 0.01%. Randomness impacts the performance more with smaller number of probed users, since we are not able to probe the influential users with great influential power, thus lowering the performance. For MSE, $\beta = 0.8$ configuration performs 7% better than $\beta = 0.6$ and 12% better than $\beta = 0.4$. For the Jaccard similarity measure, it is 2% better than $\beta = 0.6$ and 7% better than $\beta = 0.4$. Although, it performs worse than Change Probing in the short term, it reaches the performance of Change Probing in the long term, as show in in Figures 8a and 8b. Moreover, it guarantees the probing of every node within a time frame, preventing the system to focus on only a limited section of the network and missing other regional changes that might accumulate and start to affect the network in the global sense. We would have seen this phenomenon more



(a) MSE

(b) Jaccard similarity

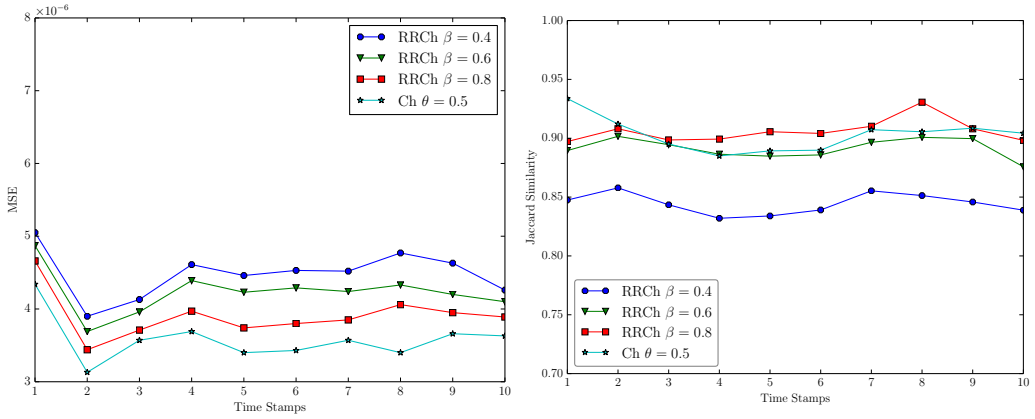
Figure 7: Performance of Round-Robin Change Probing.

explicitly if the number of snapshots were larger, which was the case in [51]. The results are slightly better when the ratio is set to $\beta = 0.8$. Therefore, we choose to use this algorithm (with $\theta = 0.5$ and $\beta = 0.8$ configurations) instead of Change Probing for the comparison with others in the following sections.

5.4.4 Comparison with the State-of-the-Art

Figure 9 compares the performance of RR Change method (with $\theta = 0.5$ and $\beta = 0.8$ settings) against the baselines and the state-of-the-art methods from the literature. PR Change achieves better results for all performance measures used for comparison in our report. It reduces MSE by 21% (see Figure 9a) when compared to Priority Probing and 49% when compared to the MaxG method. Priority Probing suffers especially for low probing capacities, since the priority of a user is set to 0 after probing. A probed user can regain its priority very late in the process, which prevents it to track quick changes in the scores of the highly influential users. Therefore, after probing an important user in terms of influence, that user is not being probed for some time, even if the influence of the user is changing very fast. RR Change always probes β portion of the users according to their influence impact and change over time, so that the important users are in the probe set at each step.

In Priority Probing, only a single user's connection is updated at a time, after which PageRanks are re-calculated and the next iteration is started. However, in real life applications fetching periods are longer, such as one week, thus determining a subset of users and updating their connections in the next fetching period



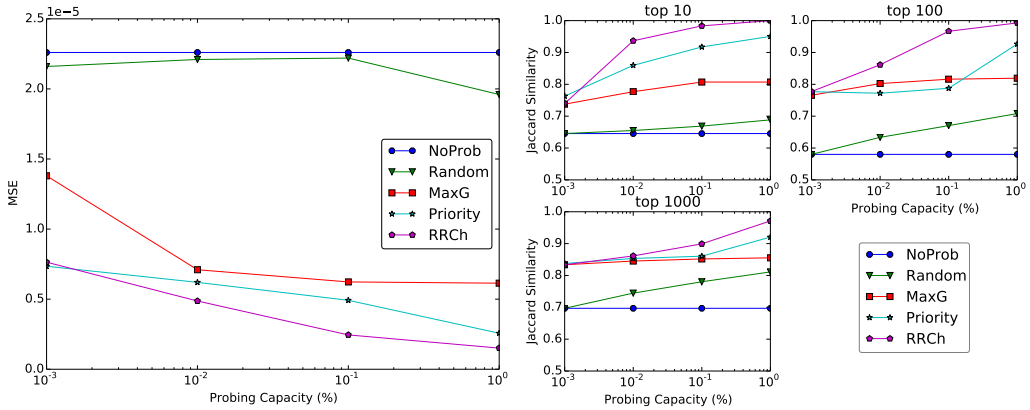
(a) MSE

(b) Jaccard similarity

Figure 8: Performance of Round-Robin Change Probing as a function of time.

is a more applicable approach. Furthermore, in Priority Probing, the priority of the newly fetched user is reset and other users' priorities are increased proportional to their PageRank scores, so that every user will be eventually fetched. When the fetching period is long and there are users who have large number of quickly changing followers, resetting the priority of such users could significantly decrease the accuracy of the results as such users won't be fetched. In our algorithm, we ensure the selection of these users. When we compare the methods in terms of the freshness of the local data they maintain, longer fetching periods cause our approach to use the same results during the fetching interval. On the other hand, Priority Probing updates the network at the end of each individual probe and recalculate the PageRank. Although, the local data is more up-to-date, this technique increases the computational overhead for real life applications. In our setup, influence score calculation overheads are significantly less.

Overall, our proposed method gives 80% higher performance than the baseline algorithms for the MSE measure. As seen in Figure 9b, RR Change shows better results for the top-k set similarities as well. It is 5% better than Priority Probing and 11% better than MaxG method on average. RR Change performs 35% better against baselines when Jaccard similarity is considered. Since it also considers the change in the influence over time, it is also able to preserve its accuracy while the performance of other methods degrade over time (see Figures 10a and 10b).



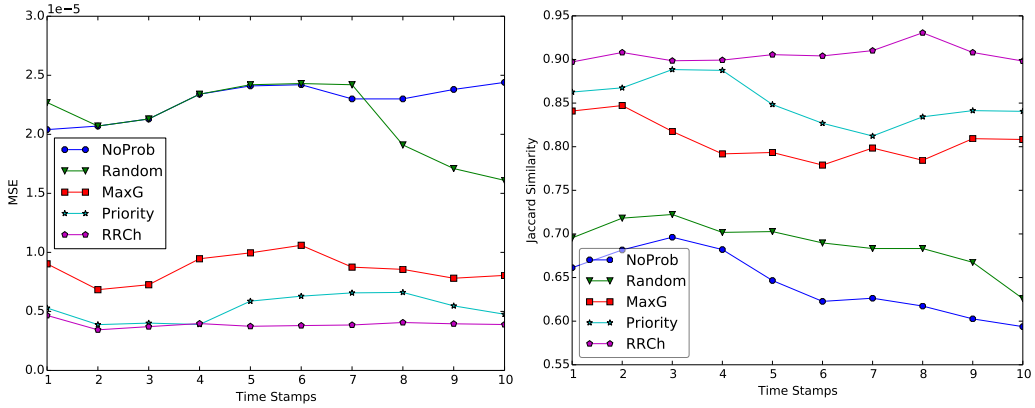
(a) MSE

(b) Jaccard similarity

Figure 9: Comparison of the probing strategies.

5.4.5 Evaluation of the Network Inference Method

To assess the prediction quality of the link prediction algorithm, we plotted the histogram of the edges proposed by RA index that has really occurred in the real network. This is shown in Figure 11. The histogram indicates the accuracy of the RA index used for network inference. The edges that were determined by the prediction algorithm as more likely to happen were found to be existent in the future network with a higher probability. However, when we analyzed the incorrectly predicted edges, we have observed that the algorithm predicts links between users who are unlikely to follow each other in real life. For example, the algorithms predict an edge between two pop stars since they have many common neighbors. However, they would not follow each other because they are main competitors. Furthermore, some of these users are not willing to follow anybody at all. This is the same issue studied in [22]. Link prediction algorithms typically do not consider these facts in social networks. In addition to indexes which they use to calculate similarities between users, they should also consider the tendency of the users to make new connections. Therefore, we apply a filtering process such that we only consider users who follow more than a threshold number of users in order to determine users who are likely to follow somebody. We add the predicted edges only to these selected users. As a result, we improve the RR Change method by 3% for MSE and 2% for the set similarities on average. Here, adaptation of more advanced (like mentioned in 4) prediction algorithms could potentially increase the accuracy of this technique.



(a) MSE over time

(b) Jaccard similarity over time

Figure 10: Comparison of the Probing strategies with respect to time.

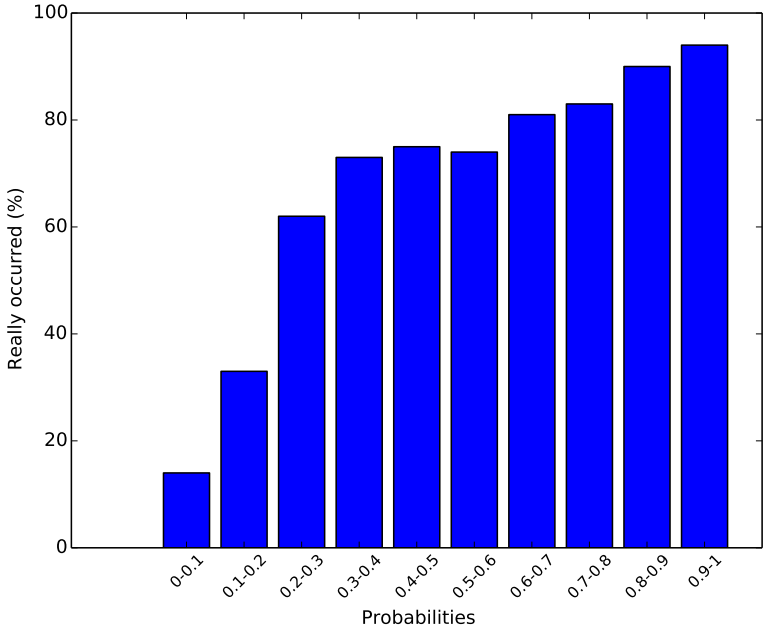


Figure 11: Accuracy of the link prediction algorithm.

5.4.6 Evaluation of the Topic Influence Estimation

We evaluated the influence of users with respect to four different topics: *a*) Politics, *b*) Sport, *c*) Health, and *d*) Cultural and Art Activities. This section provides a qualitative discussion about the accounts which were found to be influential by

the proposed methods. Table 1 shows the accuracy of topic relevance of the top-10 users found by the system for the specific topics.

Topics	**Topic Relevance**
Politics	*10 out of 10*
Sport	*8.5 out of 10*
Health	*4 out of 10*
Cultural and Art Activities	*9 out of 10*

Table 1: Estimated influential accounts.

For the evaluation of the results, we performed a small survey containing 10 people. We asked participants to evaluate the users with respect to their topic relevance and their influence on the topic. In order to identify influence of a user, we asked participants to mark one of the following categories: *a*) very influential (1), *b*) influential (.5), *c*) not influential (0). We used the results of the survey to provide an evaluation of the selected users for the Turkish Twitter network, on a per-topic basis.

For the topic Politics, the results are very accurate for top-10. We have observed that the dictionaries constructed for each topic has a big impact on the results. For example, we observe that the dictionary constructed for Politics topic contains many keywords that are related only with politics without any ambiguity. These keywords have increased the performance of the semantic analysis, which in turn increased the accuracy of the topic-based network influence analysis. Top-10 list contains the president of Turkish Republic (RT_Erdogan), the chairman of one of the opposition parties (kilicdarogluk), and the mayor of the capital city (06melikgokcek). It is fair to assume that these users, who give political messages in their tweets and who have lots of followers, should be in the top-10 influential list on Turkish Politics topic.

The influential accounts for the Sport topic were the biggest sport clubs of Turkey (GalatasarаySK, Fenerbahce) and one of the highest rating sport channel (ntvspor). Their tweets were mostly related about the sport competitions, news from clubs, etc. They have a lot of followers who actively pay attention to what they tweet. Thus, they achieve high RT and Fav statistics, which shows that they have a big impact on their followers. It is very reasonable that they are top influential accounts on this topic.

As intuitively expected, the influential accounts for the Health topic are mostly

doctor associations and governmental authorities. One of the accounts is Republic of Turkey Ministry of Health (saglikbakanligi), which mainly tweets about hospitals, doctors, and health regulations. Its follower numbers can be considered as relatively high and is followed by other influential accounts. Since its tweets have critical news potential, it has considerable number of RTs about the health topic. The other two are doctor associations (YYD_tr, istabip). They are followed by many doctors, which also have some potential impact on the Health topic. In this topic, accurate relevance ratio is relatively low because the constructed dictionary for this topic is not specific enough, causing errors in semantic analyses that propagates to the latter phase of influence estimation.

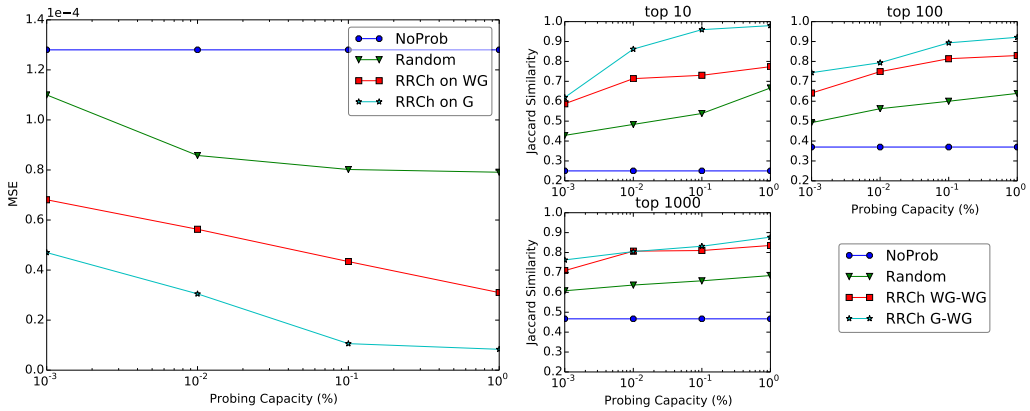
The Cultural and Art Activities topic includes users which tweet about movies, art, books, history, etc. The top-10 influential users are perfectly matched with the keywords. CMYLMZ is very famous Turkish comedian, actor and producer. He also has one of the highest follower numbers in the Turkish Twitter network. AtlasTarihDergi is a history magazine tweeting mainly about historical events and information which has considerable amount of followers and RTs. The third user (Siirler_sokakta) shares street poems and mottos, and it's posts receive many RTs and Favs.

5.4.7 Evaluation of Dynamic Tweet Fetching

We have used the same default parameter settings from the network fetching experiments to evaluate our proposed tweet fetching methods.

Figure 12 shows the performance of the RR Change method for dynamic tweet fetching. For the MSE measure, global network based G-WG method performs 78% better, and topic network based WG-WG method performs 40% better than the baselines, on average, respectively. In Figure 12b, we see that as the probing capacities increase, G-WG method achieves almost perfect similarity against the results obtained using the original network, for the top-10 influential users. For the top-1000 influential users experiment, it reaches close to 0.9 similarity. Together with WG-WG method, they quickly reach close to their top performance at around 1% capacity, except for the top-10 case. For the latter, WG-WG method does not enjoy the quality increase that the $G-WG$ method enjoys with increasing capacities. When we look at the Jaccard similarity based results, G-WG achieves 77% better and WG-WG achieves 65% better results than the baselines. Overall, the results show us that using the globally maintained network is more advantageous.

Although G-WG method outperforms WG-WG method when we compare the top-10 results for the two methods, they are similar in terms of the topic rel-



(a) MSE (b) Jaccard similarity

Figure 12: Performance of Change Probing for dynamic tweet fetching.

evance of their top influential users. Table 2 shows the topic relevance ratios for the two methods. Top-10 selected users are found to be related with the topics of interest and are popular accounts in the topic area.

Topics	**Topic Relevance**	
Politics	*10 out of 10*	*10 out of 10*
Sport	*8 out of 10*	*9 out of 10*
Health	*5 out of 10*	*4 out of 10*
Cultural and Art Activities	*9 out of 10*	*9 out of 10*
	$G-WG$ **method**	$WG-WG$ **method**

Table 2: Top-10 topic relevance ratios for G-WG and WG-WG for dynamic tweet fetching.

6 Related Work

Increases in the popularity of social networks and the availability of public data acquisition tools for them have put social networks on the spotlight of both academic and industrial research. Influential user estimation problem is studied by many researchers following a wide variety of different methodologies. Within this context, some studies introduce centrality measures in order to reflect influence of users. [44] introduces several definitions, such as *degree centrality*, *betweenness centrality*, and *closeness centrality*. For viral marketing applications,

[11] develops methods for computing network influence from collaborative filtering databases by using heuristics in a general descriptive probabilistic model of influence propagation. [18] addresses a similar problem by studying the linear threshold and independent cascade models, and [19] presents a simple greedy algorithm for maximizing the spread of influence using a general model of social influence, termed the decreasing cascade model.

Recently, researchers have studied extracting textual information associated with social networks. [27] studies topic modeling in social networks and proposes a solution for text mining on the network structure. [37] introduces the topic-based social influence problem. Their proposed model takes the result of any predefined topic modeling of a social network and constructs a network representing topic-based influence propagation. Distributed learning algorithms are used for this purpose, which leverage the Map-Reduce concept. Thus, their methodology scales to large networks. [25] combines heterogeneous links and textual content for each user in order to mine topic-based influence.

Another recent study [45] uses a PageRank-like measure to find influential accounts on Twitter. They extend PageRank by using topic-specific probabilities in the random surfer model. Although their method is similar to ours, their influence measure utilizes the number of posts made on a specific topic. However, this is an indirect measure that cannot reliably capture influence. Therefore, we use topic distributions of user posts along with their sharing statistics (re-tweets and favorites in Twitter), which provides robust results, as it takes into account the real impact of posts. [16] conducts an empirical study of different topic modeling strategies based on standard *Latent Dirichlet Allocation* (LDA) [4]. [24] proposes joint probabilistic models of influence and topics. Their methodology performs a topic sampling over textual contents and tracks the topic snapshots over time. [15] uses re-tweets in measuring popularity and proposes machine learning techniques to predict popularity of Twitter posts. [36, 50, 7] propose solutions for predicting popularity of online content. [6] studies the topic-aware influence maximization problem. Within this context, in this work we introduce a new method that combines topic-based analyses of posts with their sharing popularity for the purpose of topic-based influential user estimation.

Dynamic graph analysis has also attracted a lot of attention recently. In order to maintain dynamic networks, [46, 8, 9, 33, 31] propose algorithms for determining web crawling schedules. [21] studies the microscopic evolution of social networks. [10] studies incremental PageRank on evolving graphs. Researches have also investigated probing strategies for analyzing evolving social networks. [3] proposes influence proportional probing strategies for the com-

putation of PageRank on evolving networks and [51] uses a probing strategy to capture observed image of the network by maximizing a performance gap function. [41, 30] study sampling over social networks. However, these studies only focus on current image of a network in their probing strategies. In contrast, we propose a method which also considers evolution of the probing metrics, so that the network could be probed more effectively.

In the context of network inference, [12] proposes representations for structural uncertainty and use directed graphical models and probabilistic relational models for link structure learning. However, their methodologies are not scalable. [13, 35, 20] use time evolving graph models for social network estimation. They apply time-varying dynamic Bayesian networks for modeling evolving network structures. [5] shows that third-parties can reach a user's information by searching a few friends. [14] develops a scalable algorithm to infer influence and diffusion network based on an assumption that all users in the network influence their neighbors with equal probability. [28] removes this assumption and addresses the more general problem by formulating a maximum likelihood problem and guarantee the optimality of the solution. [48] proposes a linear model to predict how diffusion unfolds over time and [17] proposes the notion of diffusion centrality. [49, 34] studies a different problem related to network inference. Different from these works, we use friendship weighting method in order to infer link structures, similar to [38, 42, 23]. However, we use friendship weights only to infer edges between users. Moreover, one can also use more informative features such as content-based influential effects. [43] studies diffusion of tweets throughout the Twitter network. This kind of technique could also be used in order to estimate impact of posts.

7 Conclusion

The rate restrictions enforced by social network service providers have a negative impact on the third-party evolving network analysis tasks. Therefore, we proposed probing algorithms to dynamically fetch network topology and text data from social networks under limited probing capacities. Our proposed solutions use the past influence trends of the users, as well as their current influences, in order to determine the best users to probe, with the aim of maximizing the influence estimation accuracy. In particular, we observed that highly influential users and users with strong influence trends affect the overall influence estimations the most. We have leveraged these two metrics across our probing algorithms. Experimental results have shown that considering past trends in the probing strategy increases the

overall accuracy of influence prediction. Furthermore, we improved our probing strategies by inferring possible relations between users via link prediction algorithms. We also developed techniques for estimating topic-based user influence in dynamic social networks. For computing topic-based influence, we proposed methods that consider both the place of the user in the network topology, as well as the topic analysis performed on the user posts and the sharing statistics of these posts. Our experimental results performed on Twitter network data has shown improved accuracy compared to state-of-the-art methods from the literature.

References

[1] P. Analytics. Twitter study–august 2009. *San Antonio, TX: Pear Analytics. Available at: www. pearanalytics. com/blog/wp-content/uploads/2010/05/Twitter-Study-August-2009. pdf*, 2009.

[2] L. Backstrom and J. Leskovec. Supervised random walks: predicting and recommending links in social networks. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 635–644. ACM, 2011.

[3] B. Bahmani, R. Kumar, M. Mahdian, and E. Upfal. Pagerank on an evolving graph. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 24–32. ACM, 2012.

[4] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[5] J. Bonneau, J. Anderson, R. Anderson, and F. Stajano. Eight friends are enough: social graph approximation via public listings. In *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems*, pages 13–18. ACM, 2009.

[6] S. Chen, J. Fan, G. Li, J. Feng, K.-l. Tan, and J. Tang. Online topic-aware influence maximization. *Proceedings of the VLDB Endowment*, 8(6):666–677, 2015.

[7] J. Cheng, L. Adamic, P. A. Dow, J. M. Kleinberg, and J. Leskovec. Can cascades be predicted? In *Proceedings of the 23rd international conference on World wide web*, pages 925–936. International World Wide Web Conferences Steering Committee, 2014.

[8] J. Cho and H. Garcia-Molina. Effective page refresh policies for web crawlers. *ACM Trans. Database Syst.*, 28(4):390–426, Dec. 2003.

[9] J. Cho and H. Garcia-Molina. Estimating frequency of change. *ACM Transactions on Internet Technology (TOIT)*, 3(3):256–290, 2003.

[10] P. Desikan and N. Pathak. Incremental pagerank computation on evolving graphs. In *in WWW*, pages 10–14, 2005.

[11] P. Domingos and M. Richardson. Mining the network value of customers. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 57–66. ACM, 2001.

[12] L. Getoor, N. Friedman, D. Koller, and B. Taskar. Learning probabilistic models of link structure. *The Journal of Machine Learning Research*, 3:679–707, 2003.

[13] Z. Ghahramani. Learning dynamic bayesian networks. In *Adaptive processing of sequences and data structures*, pages 168–197. Springer, 1998.

[14] M. Gomez Rodriguez, J. Leskovec, and A. Krause. Inferring networks of diffusion and influence. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1019–1028. ACM, 2010.

[15] L. Hong, O. Dan, and B. D. Davison. Predicting popular messages in twitter. In *Proceedings of the 20th international conference companion on World wide web*, pages 57–58. ACM, 2011.

[16] L. Hong and B. D. Davison. Empirical study of topic modeling in twitter. In *Proceedings of the First Workshop on Social Media Analytics*, SOMA '10, pages 80–88, New York, NY, USA, 2010. ACM.

[17] C. Kang, C. Molinaro, S. Kraus, Y. Shavitt, and V. Subrahmanian. Diffusion centrality in social networks. In *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*, pages 558–564. IEEE Computer Society, 2012.

[18] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[19] D. Kempe, J. Kleinberg, and É. Tardos. Influential nodes in a diffusion model for social networks. In *Automata, languages and programming*, pages 1127–1138. Springer, 2005.

[20] J. H. Koskinen and T. A. Snijders. Bayesian inference for dynamic social network data. *Journal of statistical planning and inference*, 137(12):3930–3938, 2007.

[21] J. Leskovec, L. Backstrom, R. Kumar, and A. Tomkins. Microscopic evolution of social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 462–470. ACM, 2008.

[22] H. Li, S. S. Bhowmick, and A. Sun. Casino: towards conformity-aware social influence analysis in online social networks. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1007–1012. ACM, 2011.

[23] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

[24] C. X. Lin, Q. Mei, J. Han, Y. Jiang, and M. Danilevsky. The joint inference of topic diffusion and evolution in social communities. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 378–387. IEEE, 2011.

[25] L. Liu, J. Tang, J. Han, M. Jiang, and S. Yang. Mining topic-level influence in heterogeneous networks. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 199–208. ACM, 2010.

[26] L. Lü and T. Zhou. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150–1170, 2011.

[27] Q. Mei, D. Cai, D. Zhang, and C. Zhai. Topic modeling with network regularization. In *Proceedings of the 17th international conference on World Wide Web*, pages 101–110. ACM, 2008.

[28] S. Myers and J. Leskovec. On the convexity of latent social network inference. In *Advances in Neural Information Processing Systems*, pages 1741–1749, 2010.

[29] M. Naaman, J. Boase, and C.-H. Lai. Is it really about me?: Message content in social awareness streams. In *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work*, CSCW '10, pages 189–192, New York, NY, USA, 2010. ACM.

[30] A. Nazi, Z. Zhou, S. Thirumuruganathan, N. Zhang, and G. Das. Walk, not wait: Faster sampling over online social networks. *arXiv preprint arXiv:1410.7833*, 2014.

[31] C. Olston and S. Pandey. Recrawl scheduling based on information longevity. In *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, pages 437–446, New York, NY, USA, 2008. ACM.

[32] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. 1999.

[33] S. Pandey and C. Olston. User-centric web crawling. In *Proceedings of the 14th International Conference on World Wide Web*, WWW '05, pages 401–411, New York, NY, USA, 2005. ACM.

[34] M. G. Rodriguez, D. Balduzzi, and B. Schölkopf. Uncovering the temporal dynamics of diffusion networks. *arXiv preprint arXiv:1105.0697*, 2011.

[35] L. Song, M. Kolar, and E. P. Xing. Time-varying dynamic bayesian networks. In *Advances in Neural Information Processing Systems*, pages 1732–1740, 2009.

[36] G. Szabo and B. A. Huberman. Predicting the popularity of online content. *Commun. ACM*, 53(8):80–88, Aug. 2010.

[37] J. Tang, J. Sun, C. Wang, and Z. Yang. Social influence analysis in large-scale networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 807–816. ACM, 2009.

[38] B. Taskar, M.-F. Wong, P. Abbeel, and D. Koller. Link prediction in relational data. In *Advances in neural information processing systems*, page None, 2003.

[39] Twitter. https://twitter.com/, 2006.

[40] Twitter API rate limits. https://dev.twitter.com/rest/public/rate-limiting, 2015.

[41] G. Valkanas, I. Katakis, D. Gunopulos, and A. Stefanidis. Mining twitter data with resource constraints. In *Proceedings of the 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT) - Volume 01*, WI-IAT '14, pages 157–164, Washington, DC, USA, 2014. IEEE Computer Society.

[42] J.-P. Vert and Y. Yamanishi. Supervised graph inference. In *Advances in Neural Information Processing Systems*, pages 1433–1440, 2004.

[43] B. Wang, C. Wang, J. Bu, C. Chen, W. V. Zhang, D. Cai, and X. He. Whom to mention: expand the diffusion of tweets by@ recommendation on microblogging systems. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1331–1340. International World Wide Web Conferences Steering Committee, 2013.

[44] S. Wasserman. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[45] J. Weng, E.-P. Lim, J. Jiang, and Q. He. Twitterrank: finding topic-sensitive influential twitterers. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 261–270. ACM, 2010.

[46] J. L. Wolf, M. S. Squillante, P. Yu, J. Sethuraman, and L. Ozsen. Optimal crawling strategies for web search engines. In *Proceedings of the 11th international conference on World Wide Web*, pages 136–147. ACM, 2002.

[47] W. Xing and A. Ghorbani. Weighted pagerank algorithm. In *Communication Networks and Services Research, 2004. Proceedings. Second Annual Conference on*, pages 305–314. IEEE, 2004.

[48] J. Yang and J. Leskovec. Modeling information diffusion in implicit networks. In *Proceedings of the 2010 IEEE International Conference on Data Mining*, ICDM '10, pages 599–608, Washington, DC, USA, 2010. IEEE Computer Society.

[49] J. Yang and J. Leskovec. Modeling information diffusion in implicit networks. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 599–608. IEEE, 2010.

[50] J. Yang and J. Leskovec. Patterns of temporal variation in online media. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 177–186, New York, NY, USA, 2011. ACM.

[51] H. Zhuang, Y. Sun, J. Tang, J. Zhang, and X. Sun. Influence maximization in dynamic social networks. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, pages 1313–1318, Dec 2013.