International Journal of Database Theory and Application
Vol.8, No.2 (2015), pp.193-204
http://dx.doi.org/10.14257/ijdta.2015.8.2.18

Exploiting Historical Diffusion Data to Maximize Information Spread in Social Networks

Donghao Zhou[1, 2], Wenbao Han[2] and Yongjun Wang[1]

[1] *College of Computers, National University of Defense Technology, Changsha, China*
[2] *State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou, China*
dhzhou2084@163.com;wb.han@netease.com;wwyyjj1971@sina.com

Abstract

Information spread maximization is to find a small subset of nodes in social network such that they can maximize the expected spread of information. In this paper, we attempt harnessing historical information cascades data to learn how information propagates in social networks and how to maximize its spread. In particular, we proposed a voting algorithm to learn diffusion probabilities of edges from cascades data. Then a pruning method is developed to remove trivial edges whose weights are smaller than a threshold. Moreover, motivated by the social influence locality, we propose a Local Influence Model to evaluate node's influence within a local area instead of the whole network, which can effectively reduce the computational complexity. Based on Local Influence Model, we use greedy algorithm to find an approximate optimal solution. Experimental results show that our method significantly outperforms state-of-the-art models both in terms of information spread and algorithm runtime.

Keywords: *information spread, influence maximization, local influence model, greedy algorithm, pruning method*

1. Introduction

Diffusions over networks, such as the spread of information, ideas, or influence are a pervasive phenomenon in many networks. One of the fundamental issues is to find a small subset of influential spreaders such that they can spread information to the largest number of nodes in the network. Kempe *et. al.* [1] first formalized the problem as influence maximization problem and proved that it is NP-hard. For such problem, a greedy algorithm [1] can be used to find a target nodes set, which is an approximate optimal solution with a theoretical guarantee of (1-1/*e*). Extensive researches show that the greedy algorithm significantly outperforms the approach using degree and centrality-based heuristics [2].

For greedy algorithm, two factors are crucially important, 1) how to determine the diffusion probabilities of each edge in network G, and 2) because Monte Carlo (MC) simulation used in the greedy algorithm is computationally expensive, how to reduce the computation complexity so it can be used in large-scale social networks.

For the first factor, existing methods either simply treat the probabilities as constant (*e.g.* 0.1) or drawing values uniformly from a small set of constants, *e.g.*, {0.1, 0.01, 0.001}, which obviously can't satisfy the real-world scenarios and may lead to a poor result. For the second factor, some improved algorithms are proposed to speed up the computing process, such as PMIA [3], CELF [4], SPM [5], and LDAG [6]. But this is still an open field need further improvements.

In this paper, we propose a data-driven approach to maximize information spread in target network. The data we use is the historical information cascades data, *i.e.*, the trace

ISSN: 2005-4270 IJDTA

of a piece of information spreading across a network. We propose a novel voting algorithm to learn diffusion probabilities of each edge in network using the observed information cascades dataset. Then, node's influence is defined based on the network structure and diffusion probabilities of edges. To reduce the computation complexity, we develop a pruning method to remove trivial edges whose diffusion probabilities are smaller than a thread hold value, and restrict a node's influence within a local area around of the node. Last, a greedy algorithm is exploited to iteratively find the top-k influential spreaders in the diffusion network.

Experimental results indicate that our method is consistently better than that of repost algorithms in terms of information spread coverage and runtime.

The rest of the paper is organized as follows. In section 2 we briefly surveys related work. In Section 3, we describe the data-driven approach in detail which can maximize information spread in social networks. We proceed by describing experimental evaluation in Section 4 and conclude in Section 5.

2. Related Work

The issue of finding influential spreaders to maximize influence spread has drawn much research attention over recent years. There are two research lines in this field.

The first class methods are centrality-based. These methods include degree centrality, between's centrality, closeness centrality and eigenvector centrality [2]. Degree centrality is a simple local measure, which neglects the global structure of the network. While some well-known global metrics like between's centrality and closeness centrality can yield better results, they suffer expensive computational costs and can't be applied to large-scale networks. Recently, Kitsak *et al.* [7] found that the most efficient spreaders are those located within the core of the network as identified by the k-shell decomposition analysis. Some random walk methods are also proposed to rank node's spread ability, such as TwitterRank [8], LeaderRank [9], etc. All these methods only focus on network structures without take into consideration the information diffusion mechanism, *e.g.* independent cascade model (IC model), and the influence of nodes chosen by these methods may overlap seriously.

The second class methods are greedy algorithms. Kempe *et al.* [1] first formulated the problem as an optimization problem and proved that it is NP-hard. For such a NP-hard problem, a greedy algorithm can be used to get an approximate optimal result. Leskovec *et al.* [4] exploited the submodularity property of influence function to propose an efficient greedy algorithm called CELF, which used "lazy-forward" method to select seeds iteratively. Kimura and Saito [5] considered shortest diffusion path to reduce the number of evaluations in Monte Carlo simulation process of the influence spread. Chen *et al.* [3] instead considers Maximum influence Paths (MIP) to reduce the computation time under the IC model. Chen *et al.* [6] also proposed a scalable heuristic called LDAG for the linear threshold model (LT model). All of these models assume the weights of the network are known in advance which is not realistic under many situations.

Our work is distinct from those existing works in two aspects: 1) we exploit information diffusion historical data to directly evaluate node's influence without the Monte Carlo simulations, and 2) we harness social influence locality to evaluate a node's influence within a local area and develop a pruning method to remove trivial edge in the diffusion network to reduce computation complexity.

3. Information Spread Maximization Approach

In this section we first give a formal definition of the information spread maximization problem. Then we develop a voting algorithm to learn diffusion probability for each edge in the network from information cascade data. Next, by exploiting social influence locality we propose a heuristic method to evaluate node's influence approximately within

a local area. Based on this, we use standard greedy algorithm to find the final influential nodes in *G*. We call this data-driven model ***Local Influence with Voting algorithm* (LIV)**.

3.1. Problem Definition

Given a directed graph $G=(V,E)$ where V represents nodes set, and E represents edges set. Besides, we also get some traces of information spreading within the network, which is called **information cascade**. A cascade is defined as $c:=\{(u,v,t_u^c,t_v^c),...,(w,z,t_w^c,t_z^c)\}$, where (u,v,t_u^c,t_v^c) means node u spreads information c to node v, and their infected time is t_u^c and t_v^c, respectively. We call (u,v,t_u^c,t_v^c) a diffusion step, and a cascade may have many diffusion steps. Notation $\sigma(S)$ denotes the spread of information if we choose nodes set S to publish the information.

The information spread maximization problem can be formulated as: finding a seed set S, $|S|=k$, which can maximize $\sigma(S)$. Table 1 gives a description of the notations used in this paper.

Table 1. Notations

Notations	Descriptions
$G=(V,E)$	A graph G with nodes set V and edges set E
k	Number of seeds to be selected
(u,v,t_u^c,t_v^c)	A diffusion step that means node u spreads information c to node v, and their infected time is t_u^c and t_v^c respectively
c	An information cascade in form of $c:=\{(u,v,t_u^c,t_v^c),...,(w,z,t_w^c,t_z^c)\}$
C	Information cascades set that contains different information cascade c
$p_{u,v}$	Diffusion probability of each edge (u,v) in G
S	The selected seeds set which maximize the spread of information diffusion
$\sigma(S)$	Influence of nodes set S
$\varphi_{u,v}$	Influence of node u on node v

This problem has been proved to be NP-hard [1]. It is not realistic to find an optimal solution. However, the influence functions of $\sigma(S)$ under IC model and LT model is monotone and sub modular. A function $\sigma(\cdot)$ is sub modular if it satisfies a natural "diminishing returns" property, *i.e.*, if $S\subseteq T$, then we have $\sigma(S\cup\{v\})-\sigma(S)\geq\sigma(T\cup\{v\})-\sigma(T)$. For sub modular and monotone function $\sigma(\cdot)$ with $\sigma(\varnothing)=0$, the problem of finding a set S of size *k* that maximizes $\sigma(S)$ can be approximated by a simple greedy algorithm which is shown in Algorithm 1.

Algorithm 1: Greedy algorithm $GA(k,\sigma(\cdot))$

1 Initialize $S=\varnothing$
2 **for** *i*=1 to *k* **do**
3 $v=\text{argmax}_{u\in V\setminus S}(\sigma(S\cup\{u\})-\sigma(S))$

4 $S = S \cup \{v\}$
5 **end for**
6 Output S

3.2. Learn Information Diffusion Probability of Edge

To compute the spread of a seeds set $\sigma(S)$, a probability attached to the network edge is necessary and important. We propose to exploit information cascades data to learn diffusion probability $p_{u,v}$ for each edge (u,v) in G. We borrow the concept of "vote" from political election and use it to learn diffusion probability $p_{u,v}$. The proposed algorithm is called Voting algorithm, where each edge in G is viewed as a candidate, and each diffusion step (u,v,t_u^c,t_v^c) is viewed as a vote to edge (u,v). Let $VOT_{u,v}$ denotes the votes that edge (u,v) gets. If there exists a diffusion step (u,v,t_u^c,t_v^c), then $VOT_{u,v}$ will be updated as

$$VOT_{u,v} = VOT_{u,v} + 1 \tag{3.1}$$

By scanning all the diffusion steps in C and updating the votes of edges, we finally get a total votes count for each edge in G. Note that, the more times node u spreading information to node v, the more votes the edge (u,v) gets. A higher value of votes for edge (u, v) means a larger diffusion probability assigned to it. Furthermore, we note that for a node v, all its inbound neighbors $N_{in}(v)$ share the influence on him, so a normalization method is used to make the sum of the total influence from $N_{in}(v)$ being exactly 1.

In the above model, we may suffer from a scarcity issue, *i.e.*, because the data we collect is not sufficient enough, there may be some edges which get no votes at all, but we should not naively set their diffusion probability as zero. To solve this problem, we give each edge an initial vote and in particular we set this value as 1. Then we get a smooth diffusion graph $G = (V, E, p)$, where p indicates the edge's diffusion probability.

Note that, in the basic voting algorithm, we ignore the time factor in diffusion step (u,v,t_u^c,t_v^c). Previous work [10] shows that influence decays over time in an exponential fashion, so diffusion steps with different time delays $\Delta_{u,v}^t = t_v^c - t_u^c$ should have different weights in the voting algorithm. Motivated by these ideas, we improve the vote updating process in algorithm 1 and rewrite it as

$$VOT_{u,v} = VOT_{u,v} + \exp\{-\frac{t_v^c - t_u^c}{\tau}\} \tag{3.2}$$

which captures temporal feature of diffusion step. In equation (3.2), τ is a parameter which controls the scale of time delay.

The voting process is shown in Algorithm 2.

Algorithm 2. Voting algorithm $VA(G,C)$: Learning diffusion probability $p_{u,v}$ from information cascades

1 Initialize $VOT_{u,v} = 1$ for each edge (u,v) in E
2 **for** each cascade c in C **do** /*scanning*/

for each diffusion step (u,v,t_u^c,t_v^c) in cascade c **do**

$VOT_{u,v} = VOT_{u,v} + \exp\{-\frac{t_v^c - t_u^c}{\tau}\}$ /*updating*/

end for

end for

for each $(u,v) \in E$ **do** /*regularization*/

$p_{u,v} = \frac{VOT_{u,v}}{\sum_{u \in N_{in}(v)} VOT_{u,v}}$

end for

Output $G = (V, E, p)$

3.3. Local Influence Computation

The key step in greedy algorithm is to evaluate the influence $\sigma(S)$ for target seed set S. Because there are no effective approaches to compute $\sigma(S)$ exactly, Kempe [1] proposed to simulate the random diffusion process by generating different diffusion cascades enough times. However, this approach is computationally expensive, which makes it not applicable to large scale network.

Here, based on the learned graph $G = (V, E, p)$, we develop an approximate approach to compute node's influence. Our approach focuses on a node's local influence in the network. Intuitively, if node u publishes a piece of information, it may spread to node v through all possible paths, which is represented as $path(u,v)$. All the paths form path set $PATH(u,v)$. The influence of node u on node v along $path(u,v)$ is represented as $\Pr(path(u,v))$. Then, the total influence of node u on v is the sum of $\Pr(path(u,v))$ over all possible paths, *i.e.*,

$$\varphi_{u,v} = \sum_{path(u,v) \in PATH_{u,v}} \Pr(path(u,v)) \tag{3.3}$$

Let $d(u,v)$ denote the length of $path(u,v)$, which equals to the hops from u to v. Previous work [11] shows that people's behaviors in social network are mainly influenced by close friends in their ego networks. As $d(u,v)$ increases, the influences along the path reduce fast. Motivated by these ideas, we consider the path whose length is shorter than a threshold L, and particularly in this paper, we set the value as 2. That is to say, from perspective of influencer node u, we only consider its influence on its neighbors and influence on neighbors of its neighbors. Then we have

$$\varphi_{u,v} = \begin{cases} p_{u,v} + \sum_{w \in N_{in}(v) \cap N_{out}(u), w \neq v} p_{u,w} p_{w,v}, & if\ d(u,v) = 1 \\ \sum_{w \in N_{in}(v) \cap N_{out}(u)} p_{u,w} p_{w,v}, & if\ d(u,v) = 2 \\ 0, & else \end{cases} \tag{3.4}$$

We take Figure 1 as an example to describe the computation of $\varphi_{u,v}$ in Equation (3.4). In Figure 1, the path length from u to a is 2. Node v and node *w* are inbound neighbors a and outbound neighbors of u, so the influence of u on a is

$\varphi_{u,a} = p_{u,v} \cdot p_{v,a} + p_{u,w} \cdot p_{w,a} = 0.5 \cdot 0.1 + 0.3 \cdot 0.3 = 0.14$.

So as to the influence of node v on node a, a is inbound neighbor of v. Node b and node *w* are inbound neighbors a and outbound neighbors of v, so we get

$\varphi_{v,a} = p_{v,a} + p_{v,b} \cdot p_{b,a} + p_{v,w} \cdot p_{w,a} = 0.1 + 0.5 \cdot 0.3 + 0.2 \cdot 0.3 = 0.31$.

For the influence of node u on node d, because the path length from u to d is larger than $L = 2$, we set $\varphi_{u,d} = 0$.

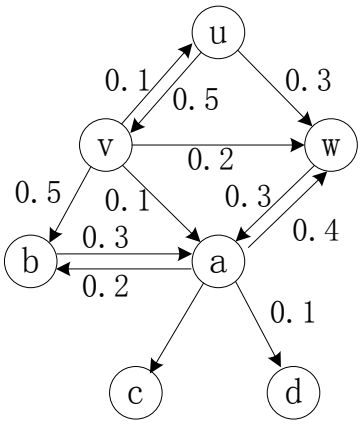

Figure 1. A Part of Information Diffusion Graph, with each Edge Assigned with a Diffusion Probability

For a subset $S \subseteq V$, assuming the nodes in S influence node v independently, then we can define its total influence on v as

$$\varphi_{S,v} = \begin{cases} 1, & if \quad v \in S \\ 1 - \prod_{w \in S} (1 - \varphi_{w,v}), & else \end{cases} \tag{3.5}$$

Then the total influence of S in the whole network G can be represented as the sum of its influence over all nodes within G, *i.e.*

$$\sigma_{LIM}(S) = \sum_{v \in V} \varphi_{S,v} \tag{3.6}$$

Pruning Edges to Reduce Computation Complexity: Note that in Equation (3.4), to compute $\varphi_{u,v}$ we need to scan all the paths from u to v, which will consume expensive runtime, even after we add a restrain to the path length. In order to reduce the computation complexity, we develop a pruning method to remove trivial edges whose diffusion probabilities are smaller than threshold β. For edge (u,v), if diffusion probability assigned to it is very small, then its contribution to $\varphi_{u,v}$ will be very small too, according to Equation (3.4). So we can remove it from the network without affecting the final results too much. The value of β controls the extent to which we will omit the trivial edges when computing $\varphi_{u,v}$. By removing trivial edges from network G, we can largely reduce the computation cost.

3.4. Information Diffusion Maximization

We now describe the greedy algorithm under our local influence model. The key step is to compute the margin gain of $\sigma_{LIM}(S)$, *i.e.*, $\sigma_{LIM}(S \cup \{u\}) - \sigma_{LIM}(S)$. According to Equation (3.6), we get

$$\sigma_{LIM}(S \cup \{u\}) - \sigma_{LIM}(S) = \sum_{v \in V} (\varphi_{S \cup \{u\},v} - \varphi_{S,v}) \tag{3.7}$$

Incorporating Equation (3.4), we get

$$\begin{aligned} \varphi_{S \cup \{u\},v} - \varphi_{S,v} &= (1 - \prod_{w \in S \cup \{u\}} (1 - \varphi_{w,v})) - (1 - \prod_{w \in S} (1 - \varphi_{w,v})) \\ &= \prod_{w \in S} (1 - \varphi_{w,v}) - \prod_{w \in S \cup \{u\}} (1 - \varphi_{w,v}) \\ &= \varphi_{u,v} \cdot \prod_{w \in S} (1 - \varphi_{w,v}) \\ &= \varphi_{u,v} \cdot (1 - \varphi_{S,v}) \geq 0 \end{aligned} \tag{3.8}$$

So, function $\varphi_{S,v}$ is monotone. Because $\sigma_{LIM}(S \cup \{u\}) - \sigma_{LIM}(S) = \sum_{v \in V} (\varphi_{S \cup \{u\},v} - \varphi_{S,v}) \geq 0$, then function $\sigma_{LIM}(\cdot)$ is also proved monotone.

We proceed to prove that function $\sigma_{LIM}(\cdot)$ is sub modular. Assuming $S \subseteq T \subseteq V$, then we get

$$\begin{aligned} &(\sigma_{LIM}(S \cup \{u\}) - \sigma_{LIM}(S)) - (\sigma_{LIM}(T \cup \{u\}) - \sigma_{LIM}(T)) \\ &= \sum_{v \in V} (\varphi_{u,v} \cdot (1 - \varphi_{S,v}) - \varphi_{u,v} \cdot (1 - \varphi_{T,v})) \\ &= \sum_{v \in V} \varphi_{u,v} \cdot (\varphi_{T,v} - \varphi_{S,v}) \geq 0 \end{aligned} \tag{3.9}$$

So, the submodularity of $\sigma_{LIM}(\cdot)$ is proved. Therefore, it is straight forward to get the following result.

Theorem 3. Function $\sigma_{LIM}(S) = \sum_{v \in V} \varphi_{S,v}$ is sub modular and monotone and $\sigma_{LIM}(\varnothing) = 0$. Therefore, greedy algorithm $GA(k, \sigma_{LIM})$ can achieve $(1 - 1/e)$ an approximation of the optimal result with Local Influence Model.

The result in Theorem 3 means that we can exploit an approximation algorithm to solve the information maximization problem. According Equation (3.9), we can rewrite the seed selection step as

$$\begin{aligned} v &= \operatorname{argmax}_{u \in V \setminus S} (\sigma_{LIM}(S \cup \{u\}) - \sigma_{LIM}(S)) \\ &= \arg\max_{u \in V \setminus S} \sum_{w \in V \setminus \sigma_{LIM}(S)} (\varphi_{S \cup \{u\},w} - \varphi_{S,w}) \\ &= \arg\max_{u \in V \setminus S} \sum_{w \in V \setminus \sigma_{LIM}(S)} \varphi_{u,w} \cdot (1 - \varphi_{S,w}) \end{aligned} \tag{3.10}$$

The greedy algorithm under local influence model is shown in Algorithm 3.

Algorithm 3: Greedy algorithm $GA(k, \sigma_{LIM}(\cdot))$

Initialize $S = \varnothing$

for i=1 to k **do**

$v = \arg\max_{u \in V \setminus S} \sum_{w \in V \setminus \sigma_{LIM}(S)} \varphi_{u,w} \cdot (1 - \varphi_{S,w})$

$S = S \cup \{v\}$

end for

4. Experimental Evaluation

In this section, we evaluate our proposed greedy algorithm with LIV model by comparing to baseline models like degree-based model, Page Rank [12], and greedy algorithm with IC model [4].

4.1. Experiment Setup

Dataset. We use real world dataset which is crawled from Sina Weibo1, China's the most popular micro blog to evaluate our proposed model. We crawled a subset of users and their posts from March of 2013 to March of 2014. The network has 61,605 nodes and 1,631,228 edges. We extract 10,600 cascades by tracing the spread of special hash tags (in form of #meme name#). The average size of these cascades is 32.4. Totally 35,065 unique users participate at least one cascade.

Baselines. Degree-based model tends to select top-k nodes that have the largest degrees as seed set. It's a simple and effective approach in some situations. Page Rank is a well known method for identifying authoritative or influential pages in a hyperlink network of web pages. This model has a parameter d that controls the probability a surfer jumps a page picked uniformly at random [12]. In our experiments, we used a typical setting of d =0.15. As for greedy algorithm with IC model (GA-IC in abbreviation), we run Monte Carlo simulations using the IC model 10,000 times and take the average. In the IC model, the probability p is set to 0.01[1]. For greedy algorithm with LIV model (GA-LIV in abbreviation), we vary the pruning threshold from 0.01, 0.05, to 0.1, and evaluate the performance for each value.

Metrics. The metric we choose is information spread of seed set S in cascade dataset C, which is denoted as $IS(S)$. In a cascade $c := \{(u, v, t_u^c, t_v^c), ..., (w, z, t_w^c, t_z^c)\}$, for diffusion step (u, v, t_u^c, t_v^c), we call v a direct child of u. If there also exists (v, w, t_v^c, t_w^c), then w is an undirected child of u. Node v and w are called springs of u, which is denoted as $SP_c(u)$. The information spread of seed u in C is $IS(u) = \sum_{c \in C} SP_c(u)$ and the information spread of seed set S in C is $IS(S) = \sum_{u \in S} IS(u)$.

4.2. Experiment Results

First, we show information spread of the 4 methods as we varying seed set size from 5 to 200 in Figure 2. We can see that degree-based method and Page Rank get similar but poor results. Greedy algorithm with IC model gets a better result. Greedy algorithm with our proposed LIV model achieves the best information spread, which is about 40% higher

[1] http://weibo.com

than GA-IC and 168% higher than degree-based method or Page Rank method. One explanation we think is that by exploiting information cascades data, our model can learn the information diffusion probabilities of edges more accurately, which plays an important role in the following influence computation in greedy algorithm. Note that, here the GA-LIV is exerted on a network without pruning method. The results in Figure 2 also indicates that by considering the local or semi-local network structure and restraining the information spread in a local area, our model can fit the real world information diffusion more effectively.

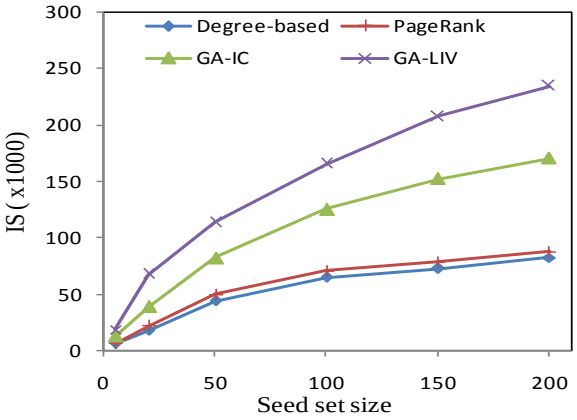

Figure 2. Information Spread Achieved by Various Methods

Then, we consider the information spread achieved by GA-LIV with different pruning threshold value as shown in Figure 3. The pruning threshold values chosen are 0.01, 0.05 and 0.1, and a higher value means a more sparse diffusion network after pruning. From Figure 3, we see that when pruning threshold is 0.01 or 0.05, the information spread achieved shows only a slight reduction comparing to that with no pruning. But when we increase the value to 0.1, the reduction gets broaden largely.

Thirdly, we show the runtime of various methods in Figure 4. The degree-based method and Page Rank consume lowest runtime, followed by greedy algorithm with LIV model. Considering their poor performances of degree-based method and Page Rank in Figure 2, such advantages in runtime make no sense. The GA-IC shows the worst runtime, which is about 75 minutes when seed set size increase to 200. Note that, here we don't add pruning method to the GA-LIV. It is clearly that by exploiting historical information spread data, instead of using expensive Monte Carlo simulation; our proposed model can effectively reduce the computation cost of greedy algorithm.

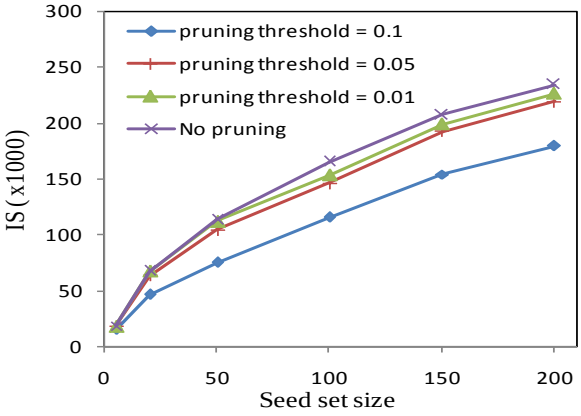

Figure 3. Information Spread Achieved by GA-LIV with Different Pruning Threshold Values

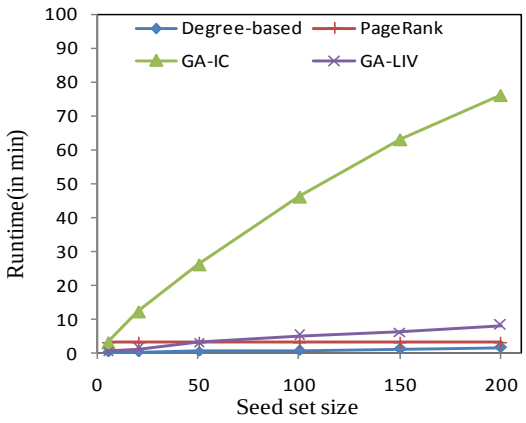

Figure 4. Runtime Comparison for Various Methods

Last, Figure 5 demonstrates the runtime of greedy algorithm with LIV model when we varying pruning threshold value. It is clearly shown that as we increase the pruning threshold value, the runtime is effectively reduced for any seed set size. When the threshold value is 0.05, the runtime can reduced 70% comparing to that with no pruning. So considering the information spread with different pruning threshold value shown in Figure 3, it is wise to choose 0.05 as the pruning threshold value. In such situation, we can largely reduce the runtime with only a slightly performance reduction.

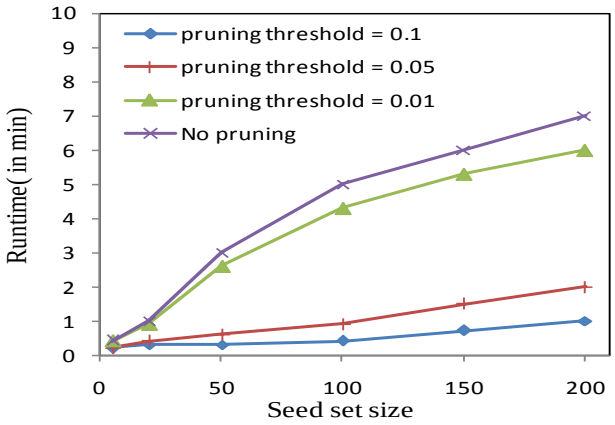

Figure 5. Runtime of GA-LIV with Different Pruning Threshold Values

5. Conclusions

In this paper, we attempt harnessing information cascades dataset to maximize information spread in social networks. We propose a novel voting algorithm to learn diffusion probabilities of edges in social networks using observed cascades dataset. Motivated by the social influence locality, we propose to compute node's influence within a local area instead of the whole network, which can effectively reduce the computational complexity. The maximization problem under our model is NP-hard and the influence function is monotone and sub modular. So we use a greedy algorithm to find an approximate optimal solution. Experimental results show that the proposed method outperforms state-of-the-art models both in terms of information spread and algorithm runtime.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant No.61271252 and No.61202482), the Specialized Research Fund for the Doctoral Program of Higher Education of China (Grant No.20124307110014).

References

[1] D. Kempe, J. Kleinberg and E. Tardos, "Maximizing the Spread of Influence through a Social Network", Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, **(2003)**, pp. 137-146. ACM, New York**.**
[2] D. Koschützki, K. A. Lehmann, L. Peeters, S. Richter, D. Tenfelde-Podehl and O. Zlotowski, "Centrality Indices", Network Analysis, **(2005)**, pp. 16-61, Springer, Berlin Heidelberg.
[3] W. Chen, C. Wang and Y. Wang, "Scalable Influence Maximization for Prevalent Viral Marketing in Large-scale Social Networks", Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, **(2010)**, pp. 1029-1038. ACM, New York.
[4] L. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. Vanbriesen and N. Glance, "Cost-effective Outbreak Detection in Networks", Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, **(2007)**, pp. 420-429, ACM, New York.
[5] M. Kimura and K. Saito, "Tractable Models for Information Diffusion in Social Networks", Knowledge Discovery in Databases, PKDD, **(2006)**, pp. 259-271, Springer, Berlin Heidelberg.
[6] W. Chen, Y. Yuan and L. Zhang, "Scalable Influence Maximization in Social Networks under the Linear Threshold Model", 10th International Conference on Data Mining, **(2010)**, pp. 88-97, IEEE Press, New York.
[7] M. Kitsak, L. K. Gallos, S. Havlin, F. Liljeros, L. Muchnik, H. E. Stanley and H. A. Makse, "Identification of Influential Spreaders in Complex Networks", Nature Physics, **(2010)**, vol. 6, 888-89.
[8] J. Weng, E. P. Lim, J. Jiang and Q. He, "Twitter rank: Finding Topic-sensitive Influential Twitterers", Proceedings of the Third ACM International Conference on Web Search and Data Mining, pp. 261-270. ACM, New York **(2010)**.
[9] L. Lü, Y. C. Zhang, C. H. Yeung and T. Zhou, "Leaders in Social Networks", the Delicious Case, PLoS ONE, vol. 6, no. 21202, **(2011)**.
[10] A. Goyal, F. Bonchi and L. V. Lakshmanan, "Learning Influence Probabilities in Social Networks", Proceedings of the Third ACM International Conference on Web Search and Data Mining, **(2010)**, pp. 241-250, ACM, New York.
[11] J. Zhang, B. Liu, J. Tang, T. Chen and J. Li, "Social Influence Locality for Modeling Retweeting Behaviors", Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence, **(2013)**, pp. 2761-2767, AAAI Press.
[12] S. Brin and L. Page, "The Anatomy of a Large-scale Hyper Textual Web Search Engine", Computer Networks and ISDN Systems, vol. 30, no. 1, **(1998)**, pp. 107-117.

Authors

Donghao Zhou, he was born in 1983. He is now a PhD candidate in School of Computer, National University of Defense Technology (China). His research interests include social networks analysis, data mining and information security (dhzhou2084@163.com).

Wenbao Han, he was born in 1963, He is a Professor in University of Information Engineering (China). His research interests include cryptanalysis, information security and high performance computing (wb.han@netease.com).

Yongjun Wang, he was born in 1971. He is now a Professor in School of Computer, National University of Defense Technology (China). His research interests include information security and networks security (wwyyjj1971@vip.sina.com).