STREAMING INFLUENCE MAXIMIZATION IN SOCIAL NETWORKS BASED ON MULTI-ACTION CREDIT DISTRIBUTION

Qilian Yu[⋆], *Hang Li*[⋆], *Yun Liao*[†], *and Shuguang Cui*[⋆]

[⋆]Dept. of ECE, University of California, Davis, CA 95616, USA
[†]Dept. of ECE, University of California, San Diego, CA 92093, USA

ABSTRACT

In a social network, influence maximization is the problem of identifying a set of users that own the maximum *influence ability* across the network. In this paper, a novel credit distribution (CD) based model, termed as the multi-action CD (mCD) model, is introduced to quantify the influence ability of each user. Compared to existing models, the new model can work with practical datasets where one type of action is recorded for multiple times. Based on this model, influence maximization is formulated as a submodular maximization problem under a knapsack constraint, which is NP-hard. An efficient streaming algorithm is developed to achieve $(\frac{1}{3} - \epsilon)$ approximation of the optimality. Experiments conducted on real Twitter dataset demonstrate that the mCD model enjoys high accuracy compared to the conventional CD model in estimating the total number of people who get influenced in a social network. Furthermore, compared to the greedy algorithm, the proposed single-pass streaming algorithm achieves similar performance in terms of influence maximization, while running several orders of magnitude faster.

***Index Terms*—** Social Networks, Influence Maximization, Credit Distribution, Submodularity, Streaming Algorithm.

1. INTRODUCTION

As information technology advances, information now spreads at a speed faster than ever before. In particular, people are ubiquitously connected by online social networks nowadays and one person's behavior may quickly affect other people's actions. For example, after a celebrity posts a new message on Twitter, many followers read it and then retweet. It may follow that the friends of these followers keep repeating such actions. Consequently, the same tweet let more and more people get involved. This phenomenon in social networks is referred to as *influence propagation*. Here, such a celebrity could be called the *influencer*. Note that, in general, there may be more than one influencers for the same event.

It is easy to see that influencers may have significant impacts on the dynamics in social networks, and thus the problem of influencer identification has drawn great attention in both academia and industry [1–4]. The influencer identification problem is commonly formulated as an **influence maximization problem** [5, 6]: Given an influence propagation model, find k "seed" nodes such that the expected number of nodes that eventually get "influenced" is maximized. There are two commonly used influence propagation models, namely the Independent Cascade (IC) model and the Linear Threshold (LT) model, in both of which one of the most important parameters is the edge weight. In existing works [5–9], the weight of each edge is usually determined by one of the following methods: 1) assigning a constant (e.g., 0.01); 2) assigning a random number from a small set (e.g., $\{0.1, 0.01, 0.001\}$); 3) assigning the reciprocal of a node's in-degree; or 4) assigning a value learnt from real data.

Although accelerated greedy algorithms have been developed [10, 11] to mitigate the high computation cost in influence maximization, all works mentioned above [5–11] need a significant number of Monte-Carlo simulations to calculate the expected number of influenced nodes, which prevents their implementation from being applied in analyzing large-scale social networks. To bypass the need of edge weights, a Credit Distribution (CD) model [12] was proposed to measure the influence only based on the history of user behaviors. Following [12], some extended versions have also been proposed [13, 14].

The datasets used in existing CD model based works [12–14] usually have a simplified structure such that they only record one timestamp of a certain action for each user, where they implicitly assumed that each user takes the same action for at most once. It is obvious that such a setup is oversimplified, since a user may take the same action multiple times. Moreover, the user who repeatedly performs a certain action would potentially influence more people than a user who just performs such an action once. This issue can be easily verified in social networks like Twitter or Facebook, where users may participate in the discussion of a topic more than once.

In this paper, we propose a novel multi-action credit distribution model (mCD) to perform data analysis on multi-action event logs, where the same action for one particular user could be recorded for multiple times if the user performs this action repeatedly. The proposed mCD model uses the timing-dependent "credit" to quantify the influence ability of each user. Based on this model, we formulate a budgeted influence maximization problem, which aims to identify a set of users with the maximum influence ability. In this problem, the objective function, i.e., the total influence ability, is submodular; and a knapsack constraint is added to regulate the cost for user selection. This problem is NP-hard; by utilizing submodularity, we develop an efficient streaming algorithm to solve the this problem, which can guarantee $(\frac{1}{3}-\epsilon)$ of the optimality.

The rest of this paper is organized as follows. In Section 2, we describe the design of the mCD model along with the formulation of the influence maximization problem. In Section 3, we introduce a learning algorithm to train the mCD model and present a streaming algorithm to solve the budgeted influence maximization problem. In Section 4, we use numerical results to demonstrate the performance of the proposed mCD model and the corresponding streaming algorithm over real Twitter dataset. Section 5 concludes the paper.

2. MODEL DESIGN AND PROBLEM FORMULATION

Given an online social network, we model it as an unweighted directed graph $\mathcal{G}=(\mathcal{V},\mathcal{E})$ [12,13], where the node set $\mathcal{V}$ is the set of all users and the edge set $\mathcal{E}$ indicates the social relationship among all the users. Specifically, for any $u,v\in\mathcal{V}$, there is a directed edge (v,u) (from v to u) if v is socially followed by u, which implies that v could potentially "influence" u. The collected data from this social network is a multi-action event log $\mathcal{L}$ with records in the form of (USER, ACTION, TIME), where a corresponding tuple $(u,a,t)\in\mathcal{L}$ indicates that user u performed action a at time t. The action a is from a finite action set $\mathcal{A}$. Here, the action is defined as a user being involved in the same discussion topic.

Given that a user could perform the same action for multiple times, we let $A_u(a)$ denote the number of times that user u performs action a. For some user-action pair (u,a), if $A_u(a)\geq 1$, let $t_i(u,a)$ denote the timestamp when user u performs acton a for the i-th time; otherwise, the timestamp is not needed. Next, we let $\mathcal{A}_u$ be the set of actions that are performed by user u. Note that the conventional CD model is a special case of the proposed mCD model, i.e., the conventional model is equivalent to the case when $A_u(a)\leq 1$ for all $u\in\mathcal{V}$ and $a\in\mathcal{A}$. Based on the directed graph $\mathcal{G}$ and the multi-action event log $\mathcal{L}$, for any action $a\in\mathcal{A}$, we define a directed graph $\mathcal{G}(a)$ that is generated from $\mathcal{G}$ according to the propagation of action a. Specifically, we define $\mathcal{G}(a)=(\mathcal{V}(a),\mathcal{E}(a))$ such that $\mathcal{V}(a)=\{v\in\mathcal{V}|A_v(a)\geq 1\}$ and $\mathcal{E}(a)=\{(v,u)\in\mathcal{E}|t_1(v,a)<t_1(u,a), A_u(a)\cdot A_v(a)\geq 1\}$. Then, for any user u who performs action a, we let $\mathcal{N}_{in}(u,a)=\{v|(v,u)\in\mathcal{E}(a)\}$ denote the set of direct influencers for user u, i.e., the neighbors of user u who perform action a earlier than user u. Next, we denote $\mathcal{N}_{in}(\mathcal{S},a)=\{v|v\in\mathcal{N}_{in}(u,a), u\in\mathcal{S}, v\notin\mathcal{S}\}$ as the neighborhood of a given user set $\mathcal{S}$ with respect to action a.

For a given action a, we define a timestamp set $\mathcal{T}_{v,u}(a)=\{t_i(v,a)|t_i(v,a)<t_1(u,a), 1\leq i\leq A_v(a)\}$ for every pair of users u and v such that $u\in\mathcal{V}(a)$ and $v\in\mathcal{N}_{in}(u,a)$, which is a collection of timestamps of v performing action a before user u. Intuitively, each time when user v performs the action, it causes influence on user u, since v and u have a directed edge (v,u) in $\mathcal{G}(a)$. To take this effect into consideration, we obtain a series of delays that can be expressed by the timestamp differences, i.e., $t_1(u,a)-t$, for all $t\in\mathcal{T}_{v,u}(a)$. Note that the conventional CD model just simply uses one delay to calculate the direct credit. Here, instead, we adopt an effective delay from v to u on action a, which is defined as $\Delta t_{v,u}(a)=1/\sum_{t\in\mathcal{T}_{v,u}(a)}(t_1(u,a)-t)^{-1}$.

Note that $\Delta t_{v,u}(a)$ equals the harmonic mean of $\{(t_1(u,a)-t)\}$ devided by $|\mathcal{T}_{v,u}(a)|$. Observing $\Delta t_{v,u}(a)$, we obtain some useful properties: 1) $\Delta t_{v,u}(a)\leq\min\{(t_1(u,a)-t)\}$ for $t\in\mathcal{T}_{v,u}(a)$, and 2) $\Delta t_{v,u}(a)$ decreases as $|\mathcal{T}_{v,u}(a)|$ increases.

The definition of $\Delta t_{v,u}(a)$ is inspired by the calculation of parallel resistance, where the equivalent resistance of multiple parallel resistors is mainly determined by the smallest one, and whenever a new resistor is added in parallel, the equivalent resistance decreases. Similarly, whenever user v taking action a poses some influence on user u, it is sensible to assume that the most recent action induces the most significant influence. Thus, it is desired that the value of the effective delay $\Delta t_{v,u}(a)$ is dominated by $\min\{t_1(u,a)-t|t\in\mathcal{T}_{v,u}(a)\}$. In addition, if user u takes action a after its neighbor v takes it for multiple times, the influence that v poses on u is stronger than those who have only taken the action once. We next define direct credit and indirect credit.

Direct Credit. This credit is what user u assigns to user v when u takes the same action a after v. The direct credit $\gamma_{v,u}(a)$ is defined as $\gamma_{v,u}(a)=\exp\left(-\Delta t_{v,u}(a)/\tau_{v,u}\right)\cdot R_{u,a}^{-1}$, where $\tau_{v,u}$ and $R_{u,a}$ are normalization factors. Note that the direct credit decays exponentially over the effective delay $\Delta t_{v,u}(a)$. Such an exponential expression follows from the original definition of the CD model [12]. Here, $\tau_{v,u}$ is the mathematical average of the time delays between v and u over all the actions: $\tau_{v,u}=\frac{1}{A_{v2u}}\cdot\sum_{a\in\mathcal{A}}\sum_{t\in\mathcal{T}_{v,u}(a)}(t_1(u,a)-t)/|\mathcal{T}_{v,u}(a)|$, where A_{v2u} denotes the number of actions that v takes prior to u. In addition, $R_{u,a}$ is given by

$$R_{u,a}=\sum_{v\in\mathcal{N}_{in}(u,a)}\exp\left(-\Delta t_{v,u}(a)/\tau_{v,u}\right),$$

which ensures that the sum of direct credits assigned to all the neighbors of u for action a is 1.

To summarize, for $u,v\in\mathcal{V}$, the direct credit given to v by u with respect to action a is given as

$$\gamma_{v,u}(a)=\begin{cases}\exp\left(-\frac{\Delta t_{v,u}(a)}{\tau_{v,u}}\right)\cdot R_{u,a}^{-1}, & (v,u)\in\mathcal{E}(a);\\ 0, & \text{otherwise.}\end{cases}$$

Indirect Credit. Suppose that (v,w) and (w,u) are in $\mathcal{E}(a)$ such that v and u are connected indirectly. Then, user u may assign all indirect credit to v via w as $\gamma_{v,w}(a)\cdot\gamma_{w,u}(a)$. As such, the total credits given to v by u on action a can be defined iteratively as $\Gamma_{v,u}(a) = \sum_{w \in \mathcal{N}_{in}(u,a)} \Gamma_{v,w}(a) \cdot \gamma_{w,u}(a)$, where $\Gamma_{v,v}(a) = 1$. Then, the average credit given to v by u with respect to all actions is defined as:

$$\kappa_{v,u} = \begin{cases} 0, & |\mathcal{A}_u| = 0; \\ \frac{1}{|\mathcal{A}_u|} \sum_{a \in \mathcal{A}} \Gamma_{v,u}(a), & \text{otherwise.} \end{cases}$$

Moreover, for a set of influencers $\mathcal{S} \subseteq \mathcal{V}(a)$ on action a, we have

$$\Gamma_{\mathcal{S},u}(a) = \begin{cases} 1, & u \in \mathcal{S}; \\ \sum_{w \in \mathcal{N}_{in}(u,a)} \Gamma_{\mathcal{S},w}(a) \cdot \gamma_{w,u}(a), & \text{otherwise.} \end{cases}$$

Similarly, we define the average credit given to $\mathcal{S}$ by u with respect to all the actions as:

$$\kappa_{\mathcal{S},u} = \begin{cases} 0, & |\mathcal{A}_u| = 0; \\ \frac{1}{|\mathcal{A}_u|} \sum_{a \in \mathcal{A}} \Gamma_{\mathcal{S},u}(a), & \text{otherwise.} \end{cases}$$

Note that the average credit $\kappa_{\mathcal{S},u}$ can also be interpreted as the "influence ability" of the set $\mathcal{S}$ on a particular user u, and the value of $\kappa_{\mathcal{S},u}$ indicates how influential $\mathcal{S}$ is. Finally, we define $\sigma_{mCD}(\mathcal{S})$ as the influence ability of $\mathcal{S}$ over the whole network, which is given as $\sigma_{mCD}(\mathcal{S}) = \sum_{u \in \mathcal{V}} \kappa_{\mathcal{S},u}$.

Budgeted Influence Maximization Problem. While aiming to maximize the influence ability over the network, we consider the budget of selecting users into the influencer set $\mathcal{S}$ as the major constraint. Suppose there are n users in the dataset. Denote a positive $n \times 1$ weight vector $g = (g_1, g_2, \ldots, g_n)^T$ as the cost for selecting each user, and $I_{\mathcal{S}} = (I_1, I_2, \cdots, I_n)^T$ as an $n \times 1$ characteristic vector of $\mathcal{S}$, where $I_i = 1$ if $i \in \mathcal{S}$; $I_i = 0$, otherwise. Let b be the total available budget on the cost for selecting users into $\mathcal{S}$. Then, the budgeted influence maximization problem could be cast as

$$\begin{aligned} \underset{\mathcal{S} \subseteq \mathcal{V}}{\text{maximize}} \quad & \sigma_{mCD}(\mathcal{S}) \\ \text{subject to} \quad & g^T I_{\mathcal{S}} \leq b, \end{aligned} \tag{1}$$

where we can normalize problem (1) such that each entry in g is no less than 1 and the number of selected users will not exceed b. For the rest of this paper, we only consider the standardized problem. Note that $\sigma_{mCD}(\mathcal{S})$ is a lower bound of the total number of users that finally get influenced over all actions, as shown in Proposition 1. Due to the page limit, the proof is skipped, which could be referred to our Arxiv version [15].

Proposition 1. $\sigma_{mCD}(\mathcal{S}) \leq |\cup_{a \in \mathcal{A}} \mathcal{V}(a)|$.

Therefore, problem (1) is to find a subset $\mathcal{S}$ from the ground set $\mathcal{V}$ to maximize a lower bound of the total number of users that finally get influenced over all actions.

Similar to the argument in [12], it is easy to show that the objective function of problem (1) is monotone and submodular. Therefore, problem (1) is a submodular maximization problem under a knapsack constraint, which has been proved to be NP-hard [16]. In general, such a problem can be approximately solved by greedy algorithms [10,16]. However, due to the large volume of online social network datasets, the implementation of greedy algorithms is not practical. In the next section, we develop an efficient streaming algorithm to solve the budgeted influence maximization problem under the mCD model.

3. ALGORITHM

The proposed algorithm is divided into the following modules. The "Model Learner" is designed to learn the parameters $\{\tau_{v,u}\}$, the mathematical average time delay between each pair of v and u over all actions, and $\{A_u(a)\}$, the frequency of u taking action a, from the training dataset before solving the optimization problem, such that the algorithm can deal with a newly arriving dataset or test set much more efficiently. Then, for the new or test set of data, we start with the preprocessing module "Log Scanner", which scans the dataset to calculate the total credit $\Gamma_{v,u}(a)$ assigned to user v by u for action a by using the already learned $\{\tau_{v,u}\}$ and $\{A_u(a)\}$ from the training set. The last but the most important module "Problem Solver" solves the influence maximization problem (1) based on $\{\Gamma_{v,u}(a)\}$ and outputs the seed set.

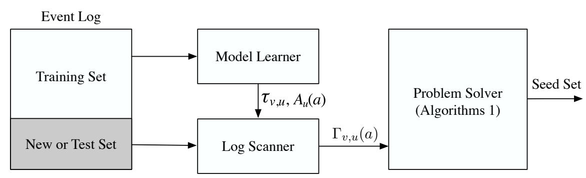

Fig. 1: Overall Structure for Influence Maximization.

We start with a cardinality constraint as a special case of the knapsack constraint (by applying the same weight for every user). Given k as the cardinality limit for $\mathcal{S}$, the simplified problem (also known as the conventional influence maximization problem) is cast as

$$\begin{aligned} \underset{\mathcal{S} \subseteq \mathcal{V}}{\text{maximize}} \quad & \sigma_{mCD}(\mathcal{S}) \\ \text{subject to} \quad & |\mathcal{S}| \leq k. \end{aligned} \tag{2}$$

We then propose a novel streaming algorithm to solve this problem. Let $m := \max_{x \in \mathcal{V}} \sigma_{mCD}(x)$; we construct an optimum value candidate set $\mathcal{O} := \{(1+\epsilon)^i | i \in \mathbb{Z}, m \leq (1+\epsilon)^i \leq k \cdot m\}$, which leads to a useful conclusion given in Lemma 1. The proof could be found in the Arxiv version [15].

Lemma 1. *Let $\mathcal{O} := \{(1+\epsilon)^i | i \in \mathbb{Z}, m \leq (1+\epsilon)^i \leq k \cdot m\}$ for some ϵ with $0 < \epsilon < 1$. Then there exists a value $c \in \mathcal{O}$ such that $(1-\epsilon)OPT \leq c \leq OPT$, with OPT denoting the optimal value for problem (2).*

However, the value of m above usually cannot be obtained in advance. In this case, we may treat m as a variable and update it during the iterations over the user selection. Specifically, we modify $\mathcal{O}$ as $\mathcal{O} = \{(1+\epsilon)^i | i \in \mathbb{Z}, m \leq (1+\epsilon)^i \leq 2k \cdot m\}$, and maintain the maximum marginal value when the algorithm scans over the ground set. Whenever m gets updated, the algorithm updates the set $\mathcal{O}$ accordingly. For each user in the ground set, we scan each element c in set $\mathcal{O}$, and add that user into $\mathcal{S}_c$ as long as the marginal gain is larger than $\frac{c}{2k}$ and $|\mathcal{S}_c| \leq k$. The corresponding pseudo-code is presented in Algorithm 1 and the performance of the algorithm is guaranteed in Theorem 1, whose proof is given in the Arxiv version [15].

Algorithm 1 STREAMING_ALGORITHM(k, UC)

1: **for** each $x \in \mathcal{V}$
2: $\quad m := \max\{m, \sigma_{mCD}(\{x\})\}$
3: $\quad \mathcal{O} := \{(1+\epsilon)^i | i \in \mathbb{Z}, m \leq (1+\epsilon)^i \leq 2k \cdot m\}$.
4: $\quad$ **for** $c \in \mathcal{O}$
5: $\quad\quad$ **if** marginal gain of c is over $\frac{c}{2k}$ and $|\mathcal{S}_c| < k$
6: $\quad\quad\quad \mathcal{S}_c := \mathcal{S}_c \cup \{x\}$.
7: $\quad\quad$ **end if**
8: $\quad$ **end for**
9: **end for**
10: **return** $\mathcal{S} := \text{argmax}_{\mathcal{S}_c, c \in \mathcal{O}} \sigma_{mCD}(\mathcal{S}_c)$.

Theorem 1. *Algorithm 1 produces a solution $\mathcal{S}$ such that $\sigma_{mCD}(\mathcal{S}) \geq \left(\frac{1}{2} - \epsilon\right)$ OPT.*

Next, to solve problem (1), we first modify the threshold in line 5 of Algorithm 1 to $\frac{2qg_x}{3b}$, where $q \in \mathcal{Q} := \{(1+3\epsilon)^i | i \in \mathbb{Z}, \frac{m}{1+3\epsilon} \leq (1+3\epsilon)^i \leq 2b \cdot m\}$, g_x is the weight of user x, and b is the total budget. Moreover, the modified algorithm keeps searching for a particular user who has dominated influences. The property of such a user is described by Theorem 2, whose proof is given in our Arxiv version [15]. At the end of the modified algorithm, we might have two types of sets: one is collected by the modified threshold, and the other exists if a user described in Theorem 2 is found. The set with a higher objective value will be the final algorithm output. Such an algorithm can solve problem (1) with $(\frac{1}{3} - \epsilon)$-approximation to the optimal solution according to Theorem 1 in [17].

Theorem 2. *Assume $q \in [(1-3\epsilon)OPT, OPT]$, x satisfies $g_x \geq \frac{b}{2}$, and its marginal is larger than $\frac{2qg_x}{3b}$. Then, we have $\sigma_{mCD}(x) \geq \left(\frac{1}{3} - \epsilon\right)$ OPT.*

4. EXPERIMENTAL RESULTS

We conduct our experiments on a reduced Twitter dataset [18] containing about 17,000 users and 100 actions to evaluate the mCD model and the corresponding streaming algorithm. Specifically, we are interested in the following performance metrics: 1) the influence ability of the seed set provided by our proposed streaming algorithm; 2) the gap between the output influence ability and the number of people that truly get influenced; and 3) the running time of the algorithm. All experiments are conducted at a server with a 3.50GHz Quad-Core Intel Xeon CPU E3-1245 and 32GB memory.

Influence Ability of the Seed Set. First, we compare the influence ability of different seed sets obtained by our proposed streaming algorithms and the CELF [11] algorithm under the same mCD model. From Fig. 2a, we observe that the seed set provided by our streaming algorithm can achieve utilities close to the CELF algorithm. For instance, when $b = 500$ and the weights of selecting users are positively related to the number of followers, the influence ability of the seed set provided by the streaming algorithm is only 0.1% less than the CELF algorithm. Therefore, we conclude that our proposed streaming algorithm is sufficient to identify seed sets with close influence ability to the CELF algorithm, but with much faster speed as discussed next. later.

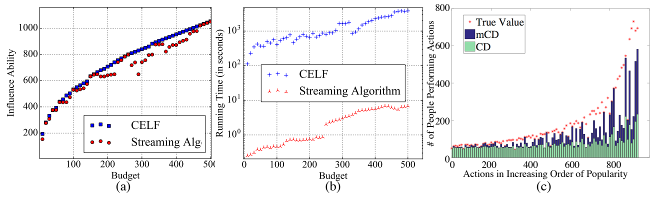

Fig. 2: a) Influence Ability Comparison. b) Running Time Comparison. c) Estimated Influence for Actions in Test Set.

Algorithm Running Time. Unlike the CELF algorithm, the proposed streaming algorithm only requires one scan over the user set. Therefore, the resulting lower computation complexity makes the algorithm more practical and scalable when the number of elements in the ground set is large. In Fig. 2b, for example, when the budget is set to be 500, it takes more than 3,800 seconds to complete the whole process in CELF, while the streaming algorithm only takes 5.3 seconds. Meanwhile, the streaming algorithm achieves almost the same performance as CELF, which implies that our proposed streaming algorithm is both efficient and effective.

Estimation on the Number of Influenced People We also investigate how the mCD model performs in estimating the number of people that get influenced in the network, by picking the most popular 950 actions from the original dataset, with the seed set size fixed as 50. We sort actions with increasing popularity. It can be observed in Fig. 2c that the results obtained by both the CD and the mCD models are smaller than the actual number of users performing the corresponding action, while the mCD model are closer to the true values, which means that the estimation with our model is more accurate for a given seed set size.

5. CONCLUSION

In this work, we extended the conventional CD model to the mCD model in dealing with the multi-action event logs and analyzing the influence ability of users in social networks. Specifically, we re-designed the credit assignment method in the CD model by utilizing a modified harmonic mean to handle multi-action event logs, which achieves a higher accuracy in estimating the total number of people that get influenced. Based on this new model, an efficient streaming algorithm was developed with $(\frac{1}{3} - \epsilon)$-approximation of the optimal value for the corresponding budgeted influence maximization problem. Experiments showed that the mCD model is more accurate compared to the conventional CD model, and the proposed algorithm can achieve similar performance to the CELF greedy algorithm, but several orders of magnitude faster.

6. REFERENCES

[1] C. Asavathiratham, S. Roy, B. Lesieutre, and G. Verghese, "The influence model," *IEEE Control Systems*, vol. 21, no. 6, pp. 52–64, Decemeber 2001.

[2] C. Asavathiratham, *A tractable representation for the dynamics of networked Markov chain*, Ph.D. thesis, Dept. of ECS, MIT, 2000.

[3] P. Domingos and M. Richardson, "Mining the network value of customers," in *Proceedings of the 7th ACM SIGKDD international conference on Knowledge discovery and data mining*, San Francisco, August 2001, ACM, pp. 57–66.

[4] M. Richardson and P. Domingos, "Mining knowledge-sharing sites for viral marketing," in *Proceedings of the 8th ACM SIGKDD international conference on Knowledge discovery and data mining*, Demonton, Alberta, Canada, July 2002, ACM, pp. 61–70.

[5] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining*, Washington, DC, August 2003, ACM, pp. 137–146.

[6] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, New York City, June 2009, ACM, pp. 199–208.

[7] W. Chen, C. Wang, and Y. Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, Washington, DC, July 2010, ACM, pp. 1029–1038.

[8] A. Goyal, F. Bonchi, and L.V.S. Lakshmanan, "Learning influence probabilities in social networks," in *Proceedings of the third ACM international conference on Web search and data mining*, New York City, February 2010, ACM, pp. 241–250.

[9] K. Saito, R. Nakano, and M. Kimura, "Prediction of information diffusion probabilities for independent cascade model," in *International Conference on Knowledge-based and Intelligent Information and Engineering Systems*, Berlin, Heidelberg, September 2008, ACM, pp. 67–75.

[10] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. Van-Briesen, and N Glance, "Cost-effective outbreak detection in networks," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, San Jose, August 2007, ACM, pp. 420–429.

[11] A. Goyal, W. Lu, and L.V.S. Lakshmanan, "Celf++: optimizing the greedy algorithm for influence maximization in social networks," in *Proceedings of the 20th international conference companion on World Wide Web*, New York, March 2011, ACM, pp. 47–48.

[12] A. Goyal, F. Bonchi, and L.V.S. Lakshmanan, "A data-based approach to social influence maximization," *in* Proceedings of the VLDB Endowment, vol. 5, no. 1, pp. 73–84, September 2011.

[13] X. Deng, Y. Pan, Y. Wu, and J. Gui, "Credit distribution and influence maximization in online social networks using node features," in *Proceedings of the 12th International conference on Fuzzy systems and knowledge discovery*, Zhangjiajie, China, August 2015, IEEE, pp. 2093–2100.

[14] Y. Pan, X. Deng, and H. Shen, "Credit distribution for influence maximization in online social networks with time constraint," in *Proceedings of 2015 IEEE International Conference on Smart City*, Chengdu, China, Decemeber 2015, IEEE, pp. 255–260.

[15] Q. Yu, H. Li, Y. Liao, and S. Cui, "Fast budgeted influence maximization over multi-action event logs," *ArXiv:1710.02141*, 2017.

[16] M. Sviridenko, "A note on maximizing a submodular set function subject to a knapsack constraint," *Operation Research Letters*, vol. 32, no. 1, pp. 41–43, January 2004.

[17] Q. Yu, E. L. Xu, and S. Cui, "Submodular maximization with multi-knapsack constraint and its applications in scientific literature recommendations," in *Proceedings of 2016 IEEE Global Conference on Signal and Information Processing*, Washington D.C., Decemeber 2016, IEEE, pp. 1295–1299.

[18] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," http://snap.stanford.edu/data, June 2014.