


Research and Analysis of Influence Maximization Techniques in Online Network Communities Based on Social Big Data

Jun Hou, Nanjing Vocational University of Industry Technology, China

 <https://orcid.org/0000-0002-6986-4961>

Shiyu Chen, Nanjing University of Science and Technology, China*

Huaqiu Long, Wuyi University, China

Qianmu Li, Nanjing University of Science and Technology, China

ABSTRACT

Many online network communities, such as Facebook, Twitter, Tik Tok, Weibo, etc., have developed rapidly and become the bridge connecting the physical social world and virtual cyberspace. Online network communities store a large number of social relationships and interactions between users. How to analyze diffusion of influence from this massive social data has become a research hotspot in the applications of big data mining in online network communities. A core issue in the study of influence diffusion is influence maximization. Influence maximization refers to selecting a few nodes in a social network as seeds, so as to maximize influence spread of seed nodes under a specific diffusion model. Focusing on two core aspects of influence maximization (i.e., models and algorithms), this paper summarizes the main achievements of research on influence maximization in the computer field in recent years. Finally, this paper briefly discusses issues, challenges, and future research directions in the research and application of influence maximization.

KEYWORDS

Approximation Algorithms, Influence Maximization, Information Diffusion, Social Networks

INTRODUCTION

Online social networks have become important platforms for people to communicate, share knowledge, and disseminate information. Moreover, online social networks are also widely used to spread news, create trends. Due to the widespread use of online social networks, individual thoughts, preferences, and behaviors are often influenced by peers or friends via social networks. From listening to a song, watching a movie, reading a new book, and choosing a restaurant, to buying a property, choosing a career, choosing a city to live in, and determining political opinions, traces of influence can be found in user's sharing behaviors on social media. Analyzing and understanding how users in online social networks interact with each other can help researchers better understand, control, and effectively utilize the diffusion process and is conducive to efficiently making public opinion analyses, information predictions, and commercial recommendations.

DOI: 10.4018/JOEUC.308466

*Corresponding Author

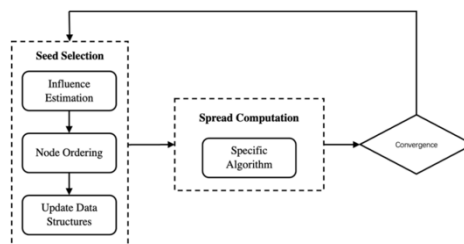
This article published as an Open Access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0/>) which permits unrestricted use, distribution, and production in any medium, provided the author of the original work and original publication source are properly credited.

A core issue of influence diffusion research is influence maximization, which aims to optimize the spread of influence. One of the important applications is viral marketing (Domingos & Richardson, 2001; Qi et al., 2019; Xu et al., 2022; Yuan et al., 2021). Specifically, a company looks for the most influential initial customers in a virtual market and provides them with the opportunity to try a product for free. The company expects these initial customers to actively promote the product in the circle of their friends after using it. Finally, these influential customers can influence their friends, friends of their friends, to accept and purchase the product. This is the goal of influence maximization. Besides viral marketing, there are many other applications of influence maximization, including network monitoring (Leskovec et al., 2007; Wang, Wang, et al., 2021; Qi et al., 2020), rumor control (Budak et al., 2011; He et al., 2012; Xu, Tian, et al., 2021; Peng et al., 2021; Hou et al., 2021), and social recommendation (Ye et al., 2012; Wu et al., 2021; Liao et al., 2021; Sun et al., 2021; Nitu et al., 2021).

The ubiquitous social networks generate massive amounts of social big data with large scale, dynamics, and heterogeneity. Different from many big data analyses, the big data analysis of influence diffusion requires analyzing the influence strength between any two associated users. This is much more difficult than analyzing only the characteristics of a user or the characteristics of a group. In addition, influence diffusion involves the analysis of people's complex behaviors. These behavioral data are not easy to be mined in social media data. Therefore, it is very important to design efficient and effective influence maximization techniques. In the big data environment, the performance of an influence maximization algorithm can be evaluated from three aspects: quality, computational efficiency, and scalability. Firstly, the influential seed users identified by a good influence maximization algorithm should have a superior expected influence spread. Secondly, a good influence maximization algorithm is superior in the running time. Thirdly, a good influence maximization algorithm should have little memory consumption as the number of required seed users increases.

In summary, this paper mainly summarizes research achievements of influence maximization. Specifically, this paper focuses on two aspects of influence maximization, i.e., models and algorithms. This paper first introduces basic models of influence diffusion and then introduces optimization problems and their algorithms under basic models. Because algorithms need to be suitable for large-scale networks in a big data environment, this paper introduces the design and analysis of efficient and scalable optimization algorithms in detail. Figure 1 shows the common workflow of influence maximization (Arora et al., 2017).

Figure 1. The framework for influence maximization



The next section of this paper introduces related definitions of influence maximization, followed by an introduction to the basic models of influence diffusion, including an independent cascade model and linear threshold model, and then introduces monotonicity and submodularity of diffusion models closely related to algorithm designs. This is followed by a discussion of the computational hardness of influence maximization and various efficient and scalable approximation algorithms of it, which in turn is followed by a summary and brief discussion of the direction of developments in this field.

PRELIMINARIES

The spread of information and influence in social networks is complex and diverse, but there are still rules to follow. In the remainder of this paper, influence diffusion is used to summarize the spread of information, concepts, ideas, innovations, and products in social networks.

Definition 1 (social networks): A social network is described as a directed graph $G = (V, E)$, where V is a set of nodes, $E \subseteq V \times V$ is a set of directed edges. Each node $v \in V$ represents a person in a social network. Each directed edge $(u, v) \in E$ represents the direction of influence, i.e., Node u has an influence on Node v , but Node v may have no influence on Node u .

The variable $n = |V|$ is used to represent the number of nodes in a graph, and $m = |E|$ is used to represent the number of directed edges in a graph. In the process of specific modeling, each edge usually has a weight indicating the strength of influence. For a directed edge $(u, v) \in E$, it is called the outgoing edge of Node u and the incoming edge of Node v . Node v is an outgoing neighbor of Node u , and Node u is an incoming neighbor of Node v . The set of all outgoing neighbors of a node v is represented by $N_{\text{out}}(v)$, and the set of all incoming neighbors is represented by $N_{\text{in}}(v)$.

Definition 2 (active nodes): In the process of influence diffusion, each node in a graph is described as two possible states: active and inactive. An *inactive state* indicates that a node does not accept some information, while an *active state* indicates that a node accepts or disseminates some information. When a node changes from an inactive state to an active state, it means that this node accepts some information. In other words, this node is activated.

Definition 3 (influence diffusion models): Diffusion models are used to describe how the state of a node in a social network influences the state of its neighbors, which further causes some state (usually referred to as an active state) to diffusion in the network.

Definition 4 (seeds): Under diffusion models, some nodes are active states at the beginning of the diffusion process. These nodes are called *seeds* and are denoted by S .

Definition 5 (influence spread): The (final) influence spread of a seed set S is the expected number of nodes activated at the end of the diffusion process. Influence spread is shown as $\tilde{A}(S)$.

Definition 6 (influence maximization): Given a social network $G = (V, E)$, an influence diffusion model and its parameters (such as independent cascade model and probabilities on edges). Given a budget k , influence maximization finds a seed set S^* with at most k nodes, so that influence spread generated by S^* is maximized. That is,

$$S^* \in \operatorname{argmax}_{S \subseteq V, |S|=k} \sigma(S). \quad (1)$$

In the aspect of social network analysis platforms and systems, many well-known scientific research institutions are devoted to the research of social network analysis and have established their own analysis tools for social networks, such as the SNAP system from Stanford University (see Table 1).

Table 1. Existing social network analysis platforms and systems

System Development Institutions	Systematic Name	Source of System
Stanford University	SNAP	https://snap.stanford.edu/data/
Tsinghua University	AMiner	https://www.aminer.cn/data-sna
Arizona State University	Social Computing Data Repository	http://datasets.syr.edu/pages/home.html
University of Koblenz–Landau	KONECT	http://konect.cc

For ease of reference, Table 2 lists the notations that are frequently used in this paper.

Table 2. Frequently used notations

Notation	Description
$G = (V, E)$	a social network G with a node set V and an edge set E
n, m	the numbers of nodes and edges in G , respectively
$N_{\text{out}}(v)$	the set of all outgoing neighbors of a node v
$N_{\text{in}}(v)$	the set of all incoming neighbors of a node v
$\tilde{A}(S)$	the influence spread of a node set S on G
$p(u, v)$	the influence probability in the IC model
$w(u, v)$	the influence of weight in the LT model

INFLUENCE DIFFUSION MODELS

There are many types of influence diffusion models. The mainstream of current research is random diffusion model, because it more directly reflects the uncertainty in influence diffusion. The random propagation model can be divided into discrete-time and continuous-time models (Aral & Walker, 2012) and progressive and non-progressive models (e.g., the election model; Kempe et al., 2003). Among numerous models, the discrete-time progressive model is studied the most. In this model, the state transition of each node and influence diffusion are specified to occur at discrete time steps. Moreover, the model assumes that any node will remain active state once it changes from inactive to active and will not return to an inactive state again. This paper mainly introduces two classical models of the discrete-time progressive diffusion model (Kempe et al., 2003), namely the independent cascade model and linear threshold model.

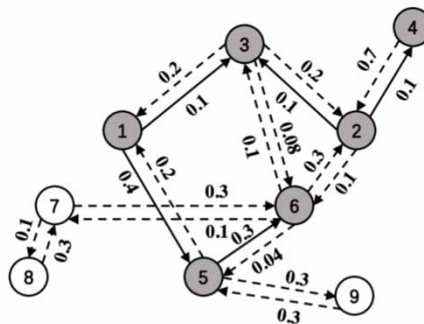
INDEPENDENT CASCADE (IC) MODEL

In the independent cascade model, every directed edge $(u, v) \in E$ in a graph has a corresponding probability $p(u, v) \in [0, 1]$. The probability $p(u, v)$ represents the probability that when node u is activated, u independently activates a node v through the edge (u, v) . The probability $p(u, v)$ is called the *influence probability*.

The dynamic diffusion process under the independent cascade model is accomplished at discrete time steps as follows: In step $t = 0$, seed set S is first activated, while other nodes are inactive. In any step $t \geq 1$, for any node u that was just activated in step $t - 1$, u will try to activate each outgoing neighbor $v \in N_{\text{out}}(u)$ that has not been activated once, and the probability of this attempt succeeding is $p(u, v)$. Furthermore, this activation attempt is independent of all other activation attempts. If the attempt is successful, Node v is activated in Step t . If the attempt is unsuccessful and other incoming neighbors of Node v do not successfully activate v in Step t , then Node v is still inactive in Step t . When no new nodes are activated in some step, the diffusion process stops.

Figure 2 (Chen et al., 2013) shows an example of the diffusion process of the IC model. Filled nodes represent active nodes. Empty nodes represent inactive nodes. Each edge of solid arcs indicates that the influence successfully propagates on the edge, and each edge of dotted arcs indicates that the influence does not propagate on the edge. The number on an edge represents the probability of influence propagation on that edge. In Step $t = 0$, Nodes 1 and 2 are selected as seed nodes and activated. In Step $t = 1$, Nodes 1 and 2 activate Nodes 5 and 4, respectively, and activate Node 3 simultaneously, but Node 2 fails to activate Node 6 successfully. In Step $t = 2$, all outgoing neighbors of Node 5 are Nodes 1, 6, and 9. Since Node 5's outgoing neighbor Node 1 is activated, Node 5 can no longer attempt to activate it. For outgoing neighbor Nodes 6 and 9 that are not activated, Node 5 successfully activates Node 6 but fails to activate Node 9. Similarly, the outgoing neighbor of Node 4 is Node 2. Since Node 2 is activated, Node 4 can no longer try to activate it. The outgoing neighbor of Node 3 that is not activated is Node 6, but the attempt of Node 3 to activate Node 6 is unsuccessful. In Step $t = 3$, the outgoing neighbor of Node 6 that is not activated is Node 7, and Node 6 tries to activate Node 7 but fails. This diffusion process ends, and Nodes 7, 8, and 9 are not activated during the diffusion.

Figure 2. An example of the diffusion process of the IC model



In real life, phenomena such as the spread of new news in online networks or the spread of a new virus among people are in line with characteristics of independent diffusion. Therefore, the independent cascade model is currently the most widely studied model.

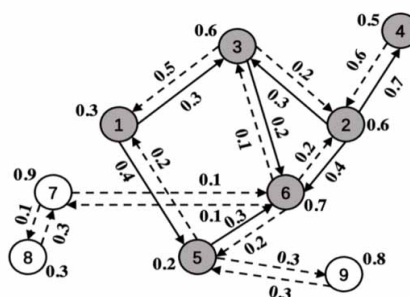
3.2 Linear Threshold (LT) Model

In linear threshold model, each directed edge $(u, v) \in E$ has a weight of $w(u, v) \in [0, 1]$, which is called the *influence weight*. The weight $w(u, v)$ reflects the proportion of the importance of Node u 's influence in all incoming neighbors of Node v . Require $\sum_{u \in N_{in}(v)} w(u, v) \leq 1$. Moreover, each node v has an influenced threshold, $\theta_v \in [0, 1]$, which is uniformly and randomly selected in the range of 0 to 1. This threshold will not change during diffusion once it is determined.

Like the independent cascade model, only nodes in the seed set S are activated in Step $t = 0$. In any step $t \geq 1$, each inactive node, v , needs to determine whether it is activated or not according to whether the linear weighted sum of all v 's activated incoming neighbors to it has reached v 's influenced threshold. If so, Node v is activated in Step t . Otherwise, Node v remains inactive. When no new nodes are activated in some step, the diffusion process stops.

Figure 3 (Chen et al., 2013) shows an example of the diffusion process of the LT model. Filled nodes represent active nodes. Empty nodes represent inactive nodes. The edges of solid arcs indicate that the weights on these edges together exceed the node thresholds, thus successfully activating the pointing nodes. Each edge of dotted arcs indicates that the influence does not propagate on the edge. The number on an edge represents an influence weight on that edge. The number next to each node is a randomly generated threshold for that node in the diffusion process. In Step $t = 0$, the threshold of each node is determined by sampling and displayed next to each node. Moreover, Nodes 1 and 2 are selected as seed nodes and activated. In Step $t = 1$, the edge weight 0.4 of Node 1 pointing to Node 5 is bigger than the threshold 0.2 of Node 5, so Node 5 is activated. The edge weight 0.7 of Node 2 pointing to Node 4 is bigger than the threshold 0.5 of Node 4, so Node 4 is also activated. The sum of the edge weights of Nodes 1 and 2 pointing to Node 3 is 0.6, which reaches the threshold 0.6 of Node 3, so Nodes 1 and 2 jointly activate Node 3. However, in Step $t = 1$, node 6 has only one incoming neighbor, Node 2, that was an active node in the previous timestep, and the edge weight 0.4 from Node 2 to Node 6 is less than the threshold 0.7 of Node 6, so Node 6 is not activated in Step $t = 1$. In Step $t = 2$, the sum of the edge weights of Nodes 2, 3, and 5 pointing to Node 6 is 0.9, which is bigger than the threshold 0.7 of Node 6, so Node 6 is activated. However, Node 5 does not activate Node 9 because the edge weight 0.3 from Node 5 to Node 9 is less than the threshold 0.8 of node 9. In Step $t = 3$, Node 6 fails to activate Node 7, so the diffusion process ends. Nodes 7, 8, and 9 are not activated during the diffusion process.

Figure 3. An example of the diffusion process of the LT model



COMPARISON OF DIFFUSION MODELS

As mentioned above, when modeling the influence diffusion, each edge $(u, v) \in E$ is often weighted to represent the strength of influence diffusion. Ideally, the influence strength should be learned from the actual data. However, most existing influence maximization work directly assigns the weights to each edge, instead of learning them. Table 3 summarizes various methods for assigning edge weights under the IC and LT models. In the IC model, there are three ways of assigning edge weights. Firstly, each edge is assigned a constant weight. For example, the value of $p(u, v)$ is set to 0.01 or 0.1, or $p(u, v)$ is a value in the range of $[0.01, 0.1]$. Secondly, each edge is assigned an equal weight. For example, $p(u, v) = 1 / |N_{in}(v)|$, where $N_{in}(v)$ is a set of all incoming neighbors of v . Thirdly, each edge is randomly assigned a weight from a set of weights. For example, the weights can be randomly chosen from the set $\{0.001, 0.01, 0.1\}$. The LT model also includes three methods to assign weights to each edge. Firstly, the weight on each edge is assigned to $w(u, v) = 1 / |N_{in}(v)|$. This is similar to the second approach in the IC model. Secondly, each edge weight is randomly chosen in the range $[0, 1]$. The third approach considers the case of multi-graphs. In real-world social networks, there are usually multiple edges between users. For example, in a co-author network, each node corresponds to a user, and an edge (u, v) simulates u and v co-authoring an article. Since u may co-author multiple articles with v , there may be parallel edges between these users. Therefore, in the

third approach, each edge weight is assigned to $w(u, v) = \frac{c(u, v)}{\sum_{u' \in N_{in}(v)} c(u', v)}$, where $c(u, v)$ is the number of parallel edges between u and v .

Table 3. A comparison of classical diffusion models

Model	$p(u, v)$ or $w(u, v)$	
IC	Constant	$p(u, v)$ is a constant probability. For example, $p(u, v) = 0.01$ or 0.01, or $p \in [0.01, 0.1]$.
	Weighted cascade	$p(u, v) = \frac{1}{ N_{in}(v) }$
	Tri-valency model	$p(u, v)$ is chosen randomly from a set of probabilities, such as the set $\{0.001, 0.01, 0.1\}$.

Table 3 continued on next page

Table 3 continued on next page

Model	$p(u, v)$ or $w(u, v)$	
LT	Uniform	$w(u, v) = \frac{1}{ N_{in}(v) }$
	Random	$w(u, v)$ is chosen uniformly at random from $[0, 1]$.
	Parallel edges	$w(u, v) = \frac{c(u, v)}{\sum_{\forall u' \in N_{in}(v)} c(u', v)}$

SUBMODULARITY OF DIFFUSION MODELS

One of the most important properties of diffusion models for influence maximization is its submodularity. This section mainly introduces the submodularity of basic models mentioned above. Firstly, the definition of submodularity is given below.

Definition 7 (submodularity of diffusion models): For a set function $f : 2^V \rightarrow \mathbb{R}$. For any subset $S \subseteq V$, any of its supersets $S' (S \subseteq S' \subseteq V)$ and any element $u \in V \setminus S'$, if f satisfies $f(S) \geq f(S' \cup \{u\}) - f(S')$, then function f is submodular. *Submodularity* reflects that the incremental effect of Element u on Set S decreases with the increase of S . This is the phenomenon of diminishing marginal gains. Furthermore, the monotonicity of set functions is often used with submodularity.

Definition 8 (monotonicity of diffusion models): For any subset $S \subseteq V$ and any of its supersets $S' (S \subseteq S' \subseteq V)$, if a set function f satisfies $f(S) \leq f(S')$, then function f is monotonic. As a function of a seed set S , the influence spread $\tilde{A}(S)$ is proved to satisfy submodularity under independent cascade and linear threshold models and many of their extension models. Therefore, submodularity of influence spread can be understood as the marginal influence of a node on some set decreases as the set increases. The definition of marginal influence gain is given below.

Definition 9 (marginal influence gain): Assume that the first seed is Node u , the second candidate seed is Node v . Marginal influence gain of Node v on Node u represents increased marginal influence after adding Node v . Marginal influence gain of Node v on current seed set S is denoted as $\text{gain}(v|S)$. Therefore,

$$\text{gain}(v|S) = \tilde{A}(S \cup \{v\}) - \tilde{A}(S) \quad (2)$$

where $\tilde{A}(S)$ represents influence spread that seed set S can generate.

The existing research proves submodularity and monotonicity of influence spread under diffusion models introduced above.

Theorem 1 (Monotonicity and sub-modularity of models such as independent cascade model):

In the independent cascade model, linear threshold model, and trigger model, influence spread $\tilde{A}(S)$ is monotonic and submodular (Kempe et al., 2003).

INFLUENCE MAXIMIZATION AND ITS ALGORITHMS

An important purpose of modeling influence diffusion is to control and optimize the spread of influence, and a core issue that is widely studied is influence maximization.

NP-HARDNESS OF INFLUENCE MAXIMIZATION

As defined in Definition 6, influence maximization refers to selecting a few nodes in a social network as seeds, so as to maximize the influence spread of these seeds under a specific diffusion model. Because influence maximization belongs to the combinatorial optimization problem, it is NP-hard to maximize influence spread under classical independent cascade linear threshold models. The following theorem gives this conclusion.

Theorem 2 (The Hardness of Influence Maximization): Influence maximization is NP-hard under both independent cascade and linear threshold models (Kempe et al., 2003).

GREEDY ALGORITHM FOR INFLUENCE MAXIMIZATION BASED ON SUBMODULARITY

Influence maximization is NP-hard under basic independent cascade and linear threshold models. In order to solve NP-hard optimization problem, effective approximation algorithms are needed. Approximate algorithms of influence maximization depend on the submodularity property of the influence spread function and the greedy algorithm technique it brings.

Regarding submodularity of diffusion models, as stated in Theorem 1, under independent cascade and linear threshold models, and even under the triggering model including them both, influence spread function $\sigma(S)$ is monotonic and submodular. Therefore, a greedy algorithm can be used to find an approximate optimal solution for influence maximization. This simple greedy algorithm can guarantee that the obtained solution is at least $1 - 1/e$ (i.e., about 63%) of the optimal solution (Even-Dar & Shapira, 2007). The greedy approximation algorithm for influence maximization is shown in Algorithm 1.

Algorithm 1: Greedy(σ, k)

Input: influence spread function $\sigma(\cdot)$, budget k

Output: seed set S of size k

```

1.       $S = \emptyset$ 
2.      for  $i = 1$  to  $k$  do
3.           $u^* = [\text{argmax}]_{(u \in V \setminus S)} (\sigma(S \cup \{u\}) - \sigma(S))$ 
4.           $S = S \cup \{u^*\}$ 
5.      end for
6.      return  $S$ 

```

Greedy approximation algorithm for influence maximization is divided into k iterations. Initially, the output seed set S is empty (line 1). At each iteration, the greedy algorithm needs to find Node u^*

that is not in S . Based on the current seed set S , u^* can maximize the marginal influence gain of σ (line 3). Then, add Node u^* to the seed set S (line 4).

The hardness of exactly computing influence spread greedy approximation algorithm of influence maximization given by Algorithm 1 cannot be directly used to compute the influence spread function A , because the key step of Algorithm 1 is to compute the influence spread $\tilde{A}(S \cup \{u\})$ of the seed set S (line 3). For an influence diffusion model and a given seed set S , for a general directed graph, it is #P-hard to accurately compute influence spread $\tilde{A}(S \cup \{u\})$ under the simplest independent cascade and linear threshold models.

Theorem 3: It is #P-hard to accurately compute the influence spread of a set under independent cascade and linear threshold models. Under the independent cascade model, even if a directed graph is a directed acyclic graph (DAG), it is also #P-hard to accurately compute influence spread (Yildiz et al., 2011).

Therefore, many publications have proposed various scalable influence maximization algorithms.

SCALABLE INFLUENCE MAXIMIZATION ALGORITHMS

In this paper, existing scalable influence maximization algorithms are divided into four categories, i.e., Monte-Carlo-based greedy approximation, heuristic-based, reverse influence sampling, and machine learning algorithms. A theoretical analysis of existing influence maximization algorithms is shown in Table 4.

Table 4. A comparison of influence maximization algorithms

Category	Algorithm	Model	Time Complexity	Pros and Cons
MC-based greedy approximation	Greedy	IC, LT	$O(kmnr)$	It provides a theoretical guarantee, but its computational efficiency is low.
	CELF	IC, LT	$O(kmnr)$	
	CELF++	IC, LT	$O(kmnr)$	

Table 4 continued on next page

Table 4 continued

Category	Algorithm	Model	Time Complexity	Pros and Cons
Heuristic-based	Degree-discount	IC, LT	$O(k \log n + m)$	It has high speed and good practical efficiency, but it lacks theoretical guarantees.
	PMIA	IC	$O(nt_i + kn_o, n_i (n_i + \log n))$	
	LDAG	LT	$O(n\bar{t} + km, n (m + \log n))$	
	SimPath	LT	$O(k\ell n\mathcal{P})$	
	IRIE	IC	$O(k(n_o, k + m))$	
RIS-based	RIS	IC, LT	$O(k\ell^2 (n + m) \log^2 n / \varepsilon^3)$	It provides a theoretical guarantee, and its time complexity is low.
	TIM/TIM+	IC, LT	$O((k + \ell)(n + m) \log n / \varepsilon^2)$	
	IMM	IC, LT	$O((k + \ell)(n + m) \log n / \varepsilon^2)$	
	SSA/D-SSA	IC, LT	-	
	OPIM	IC, LT	-	
	SKIM	IC	$O(n\ell + \sum_i E^{(i)} + m\epsilon^{-2} \log^2 n)$	
Machine learning-based	IMINFECTOR	Model-free	$O(\ell \cdot I(I \log I) \cdot (N \log N))$	It provides good scalability, prediction accuracy, computational efficiency, and influence quality.
	DISCO	IC, LT	$O(n \cdot N(v))$	

MONTE-CARLO-BASED GREEDY APPROXIMATION ALGORITHMS

The computational hardness pointed by Theorem 3 refers to the accurate computation of influence spread. Therefore, Kempe et al. (2003) pioneered the use of the Monte Carlo method to approximately compute influence spread. This approach is an extension of the greedy algorithm (i.e., Algorithm 1). The MC-greedy algorithm also iteratively selects Node u^* with the maximum marginal influence gain and adds it to the seed set S . The difference between MC-greedy algorithm and greedy algorithm is that for each node set $S \cup \{u\}$, MC-greedy algorithm uses the Monte Carlo method to estimate its influence spread $\tilde{A}(S \cup \{u\})$. Specifically, the workflow of the Monte Carlo simulation is to repeatedly simulate the diffusion process of influence on a seed set S , count the average number of activated nodes in these simulation processes, and finally use this average as the estimated value of influence spread of S . Therefore,

$$\tilde{A}(S \cup \{u\}) = \frac{1}{r} \sum_{u \in V \cup S} \mathbf{1}(u). \quad (3)$$

In Equation 3, r is the number of Monte Carlo simulations, which is usually 10,000. If Node u is activated, the indicator function $\mathbf{1}(u) = 1$; otherwise, $\mathbf{1}(u) = 0$. The solution of MC-greedy algorithm can obtain a $1 - 1/e - \mu$ approximation of the optimal solution, where ε is a number greater than zero, and it corresponds to the accuracy of $\tilde{A}(S)$ estimation. However, simply using the Monte Carlo method for influence maximization is time-consuming. It is an $O(k^2 \ell n^2 m \log n / \mu^2)$ algorithm. For slightly larger-scale social networks, the MC-greedy algorithm is challenging to scale and computationally expensive. Therefore, many researchers further propose many algorithms for the scalability problem of MC-greedy to improve computational performance.

Leskovec et al. (2007) proposed the CELF (cost-effective lazy-forward) algorithm based on the submodularity of the influence spread function. CELF uses a lazy evaluation method to greatly reduce the number of times to evaluate the influence spread function of a seed set. The difference from the MC-greedy algorithm is that the CELF algorithm does not simply estimate the influence spread $\tilde{A}(S \cup \{u\})$ of all node sets but estimates an upper bound of $\tilde{A}(S \cup \{u\})$ to avoid unnecessary recomputation of marginal influence gain at each iteration. According to Definition 7, for a node $u \in V$ and a set $S_k \subseteq S_{k+1} \subseteq V$:

$$\tilde{A}(S_k \cup \{u\}) - \tilde{A}(S_k) \geq \tilde{A}(S_{k+1} \cup \{u\}) - \tilde{A}(S_{k+1}). \quad (4)$$

where S_k is the seed set after the k th iteration. It can be seen from Equation 4 that for any $S_k \subseteq S_{k+1}$, $\text{gain}_u(S_k) = \tilde{A}(S_k \cup \{u\}) - \tilde{A}(S_k)$ is an upper bound for any $\text{gain}_u(S_{k+1}) = \tilde{A}(S_{k+1} \cup \{u\}) - \tilde{A}(S_{k+1})$. Therefore, the CELF algorithm can be divided into two parts. The first part is similar to the MC-GREEDY algorithm. Specifically, it iterates on each node in a graph and calculates marginal influence gain $\text{gain}_u(\emptyset)$ for each node $u \in V$. Then it selects the node with the maximum marginal influence gain and adds it to the seed set. Unlike the MC-greedy algorithm, the CELF algorithm needs to store marginal influence gain $\text{gain}_u(\emptyset)$ of each node in a sorted list and use it as an upper bound for the second part. The second part of the CELF algorithm iteratively finds the remaining $k - 1$ seeds. At each iteration, the CELF algorithm only re-estimates the marginal influence gain of the node at the top of the list. If this node is still at the top of the list after reordering, it must be the node with the maximum marginal influence gain among all nodes. Because according to the submodularity of influence spread function, if marginal influence gains of all other nodes in the list are recalculated, their values must be smaller than the current marginal influence gains in the list. Therefore, the CELF algorithm adds the node that is still at the top of the list after reordering to the seed set. If the node is not at the top of the list after reordering, then CELF estimates the marginal influence gain of the new top node in the list, this continues until k seeds are found.

Experiments by Leskovec et al. (2007) show that the CELF algorithm not only retains the performance of the $1 - 1/e - \mu$ approximation ratio but also has a speed increase of nearly 700 times compared with the MC-greedy algorithm. Although the performance of the CELF algorithm is significantly improved, it still takes hours to find the top 50 nodes in a network with tens of thousands of nodes (Chen et al., 2009). Subsequently, Goyal, Lu, and Lakshmanan (2011a) proposed an optimization algorithm CELF++ based on the CELF algorithm by further mining the sub-modularity.

The difference between the CELF and CELF++ algorithms is that when computing marginal influence gain of a node $\text{gain}_u(S_k)$ at each iteration, CELF++ considers the marginal influence of the node with the maximum marginal influence gain in the previous iteration to achieve a better pruning effect. Although CELF++ maintains more information than CELF, it has a performance improvement of 17% ~ 61% compared to CELF.

HEURISTIC-BASED ALGORITHMS

Although the lazy estimation method mentioned above can improve algorithm efficiency hundreds of times, it only reduces the running time from a few days to a few hours in a relatively small graph with tens of thousands of nodes and edges. When a graph size is hundreds of thousands or millions, it becomes difficult to run Monte-Carlo-based greedy algorithms in a reasonable amount of time. Therefore, some researchers propose heuristic-based algorithms that do not depend on Monte Carlo simulation. Therefore, heuristic-based algorithms run much more efficiently and are more scalable in larger network graphs (Qi et al., 2021; Xu, Fang, et al., 2021; Wang, Zhu, et al., 2021).

First, Chen et al. (2009) optimized the greedy algorithm proposed by Kempe et al.; however, the results show that it is difficult to greatly improve the performance by improving the greedy algorithm. Therefore, Chen et al. (2009) proposed a degree-discount algorithm, which selects seed nodes by discounting the out-degree of candidate nodes based on their connections to selected nodes. Specifically, when node v is selected as the seed node, the expected increment of the number of active nodes brought by it is:

$$1 + \left(d_v - 2t_v - (d_v - t_v)t_v p + o(t_v) \right) \times p \quad (5)$$

where d_v represents the degree of Node v , and t_v represents the number of v 's neighbors that are selected as seed nodes. It can be seen from Equation 5 that greater t_v means greater discount to d_v . Although degree-discount can improve computational performance, it assumes that influence probabilities on all edges are the same under the independent cascade model (i.e., uniform independent cascade model). This assumption is obviously inconsistent with actual requirements.

Chen, Wang, and Wang (2010) proposed the PMIA algorithm by extending the idea of the degree-discount algorithm under the general independent cascade model and proposed the LDAG algorithm (Chen, Yuan, & Zhang, 2010) under the linear threshold model. The PMIA algorithm assumes that influence diffusion between users is propagated through the maximum influence paths and considers influence probabilities between users. The maximum influence path $MIP(u, v)$ refers to the path with the maximum influence probability among all paths from Node u to Node v . The workflow of the PMIA algorithm is as follows: (1) First, it uses Dijkstra shortest path algorithm to compute the maximum influence paths MIPs between each pair of nodes in a network, and filters unimportant nodes by setting the threshold θ of the maximum influence paths to reduce the size of candidate nodes. (2) Then, for each node $v \in V$, it uses Dijkstra shortest path algorithm to develop the maximum influence in-arborescence $MIIA(v, \theta)$ and the maximum influence out-arborescence $MIOA(v, \theta)$ based on MIPs. $MIIA(v, \theta)$ contains all MIPs ending at Node v , and influence probabilities of all paths are at least θ . $MIOA(v, \theta)$ contains all MIPs starting at Node v , and influence probabilities of all paths are at least θ . (3) By using MIIAs and MIOAs, it can efficiently compute the influence spread of each node and marginal influence $\text{gain}_u(S_k)$ for adding any node $u \in V$ to the seed set S .

The basic idea of the LDAG algorithm (Chen, Yuan, & Zhang, 2010) is similar to the PMIA algorithm, but it is designed for the linear threshold model. The workflow of the LDAG algorithm is as follows. First, it uses Dijkstra shortest path algorithm to construct $LDAG(v, \theta)$ of Node v . In $LDAG(v, \theta)$, influence probability of each node on v is at least θ . Then, it can efficiently compute the influence spread of any node and marginal influence gain, $gain_u(S_k)$, for adding any node $u \in V$ to the seed set S based on the constructed LDAGs. Experiments on several actual networks show that PMIA and LDAG algorithms can achieve influence quality close to Monte-Carlo-based greedy algorithms, and their running time is about 1000 times faster than that of Monte-Carlo-based greedy algorithms.

In addition to the above algorithms, there are several heuristic-based algorithms to further improve efficiency. For instance, the SimPath algorithm proposed by Goyal, Lu, and Lakshmanan (2011b) is a heuristic-based algorithm for the linear threshold model. SimPath is an optimization method for CELF. It also iteratively selects seed nodes in a lazy forward fashion. However, unlike CELF, SimPath does not use expensive Monte Carlo simulations to estimate influence spread but approximately computes the influence spread of seed nodes by enumerating simple paths from seed nodes to other nodes near seed nodes. A simple path means that there are no duplicate nodes in a path. It is #P-hard to enumerate all simple paths. However, most of the influences propagate within a small neighborhood, because propagation probabilities in a path decrease rapidly with the increase of path length. Therefore, the SimPath algorithm restricts enumerations to a small neighborhood by removing paths with propagation probabilities less than a given threshold N . In the SimPath algorithm, influence spread of a seed set S is the sum of the influence spread of each node $u \in S$ on the subgraph $(V \setminus S) \cup \{u\}$. The influence spread of a node can be computed by enumerating all simple paths starting from the node with propagation probabilities of at least N and then summing the propagation probabilities of these simple paths. Experiments show that compared to the LDAG algorithm, the SimPath algorithm is more efficient, occupies less memory, and generates a larger influence spread of a seed set.

Jung, Heo, and Chen (2012) proposed the IRIE algorithm for the independent cascade model. IRIE speeds up seed node selection by utilizing the belief propagation method and the characteristics of approximate iteration. In addition, IRIE also solves the problem that PMIA needs more memory.

REVERSE INFLUENCE SAMPLING ALGORITHMS

Among the two types of algorithms introduced above, greedy approximation algorithms based on the Monte Carlo method have a theoretical guarantee, but their time efficiency is low. Therefore, subsequent research mainly focused on improving computational efficiency through heuristic-based algorithms. The heuristic-based algorithms are fast, and their influence quality outcomes are close to those of greedy algorithms. However, heuristic-based algorithms lack a theoretical guarantee (Kim et al., 2013).

Recently, the reverse influence sampling (RIS) method under the independent cascade model, pioneered by Borgs, Brautbar, Chayes, and Lucier (2014), changed this situation. The RIS-based algorithms are both fast and theoretically guaranteed. Instead of simulating and estimating the influence of seed nodes from them, the core idea of RIS is to randomly select a node and use the Monte Carlo method to simulate influence diffusion from this random node in the opposite direction of all edges. In this fashion, the set reachable by reverse influence diffusion in this fashion is called the reverse reachable set (RR-set).

Algorithm 2 gives the basic framework of the RIS-based algorithm. The framework is divided into two steps. The first step is to estimate the number of required RR-sets and generate these RR-sets (i.e., Sampling subfunction), and then store them in a set \mathcal{R} . The second step is to use the greedy algorithm in \mathcal{R} to find k nodes so that they cover as many RR-sets as possible (i.e., NodeSelection

subfunction). The number of RR-sets generated in the first step directly determines the running time of an algorithm. Therefore, it is the focus of different algorithms to improve their efficiency (Tang et al., 2014; Tang et al., 2015). For the second step, various algorithms use the same method to pick k nodes from the generated RR-sets. Therefore, this paper focuses on comparing the improvement for the first step of different algorithms.

Algorithm 2: RR-set-based influence maximization algorithm

Input: Directed graph $G = (V, E)$, budget k , approximation ratio ε , error probability l

Output: Seed set S of size k

```

1.      /* Estimate the number of required RR-sets and generate
these RR-sets.  $\mathcal{R}$  is a set of these RR-sets. */
2.       $\mathcal{R} = \text{Sampling}(G, k, \varepsilon, l)$ 
3.      /* Find  $k$  nodes with greedy algorithm on RR-Set in  $\mathcal{R}$ 
*/
4.       $S_k^* = \text{NodeSelection}(\mathcal{R}, k)$ 
5.      return  $S_k^*$ 

```

The algorithm proposed by Borgs et al. (2014) uses a threshold A -based on computational cost to indirectly control the number of required random RR-sets. Moreover, an average time complexity of their algorithm is $O(k\ell^2(m+n)\log^2 n / \mu^2)$. It is not improved enough, and no experimental verification is given. Therefore, Tang, Shi, and Xiao further proposed the TIM/TIM+ (Tang et al., 2014) and IMM algorithms (Tang et al., 2015) and performed simulation experiments to verify them.

The theoretical average complexity of TIM/TIM+ (Tang et al., 2014) and IMM algorithms is $O((k+\ell)(m+n)\log n / \mu^2)$. It is better than the algorithm proposed by Borgs et al. The number of RR-sets generated in TIM/TIM+ and IMM depends on $\tilde{A}(S) - \epsilon$, k and errors in various other

internal parameters. Specifically, TIM requires $O\left(\mu^{-2} \cdot n \cdot \left(\log n + \log \binom{n}{k}\right) / \text{OPT}\right)$ RR-sets, where

OPT is the influence spread of the optimal seed set. Since OPT is unknown, the authors proposed a series of bootstrap estimation methods for OPT to compute the number of required RR-sets. TIM+ improves on TIM by adding an intermediate step in the influence estimation process. This intermediate step heuristically adjusts the number of required RR-sets to a tighter lower bound. Therefore, it ensures that TIM+ exhibits better experimental performance while having the same worst-case complexity as TIM.

IMM (Tang et al., 2015) uses the centralized property of a stochastic process called martingales to estimate the number of required RR-sets. Specifically, the IMM algorithm uses a binary-guessing method to estimate the lower bound of OPT. It not only reduces the number of generated RR-sets but also speeds up the running time. Thus, it ensures that IMM can be performed in nearly linear time. The process of estimating the lower bound of OPT by IMM is as follows: In the i th guess, the lower bound is $n / 2^i$. First, it uses the current lower bound to generate θ_i RR-sets, which are used as the input of the NodeSelection function. Second, it calls the NodeSelection function to find a set S_i , and verifies whether the estimated influence spread is greater than $n / 2^i$. If so, the guess is successful. If not, the lower bound is halved, and it continues to guess. In this way, the lower bound of OPT can always be found within $\log_2 n$ guesses. In the experimental results (Tang et al., 2015), when $\mu = 0.5$ and $\ell=1$, IMM algorithm is 100 times faster than the heuristic-based algorithm, such as IRIE and SimPath, on some graphs. Moreover, IMM can finish running on graphs with over 1 billion edges in 100 seconds. Although the theoretical guarantee when $\mu = 0.5$ is not strong, IMM can simultaneously satisfy a $1 - 1/e - \mu$ approximation ratio and $O((k+\ell)(m+n)\log n / \mu^2)$ running time. The

performance of IMM can also reach or even exceed efficient heuristic-based algorithms in the simulation experiments. Therefore, IMM is a state-of-the-art algorithm among influence maximization algorithms.

After the IMM algorithm, there have been several new algorithms trying to improve IMM further. Nguyen, Thai, and Dinh (2016) proposed the SSA/D-SSA algorithm. SSA algorithm adds a stop-and-stare strategy based on the framework of RIS. In each iteration, SSA doubly generates new RR-sets and extracts seed nodes based on currently generated RR-sets. Then, SSA stops and verifies whether the estimated influence spread of the current seed set is close to the estimated influence spread of the seed set in the previous iteration. If so, SSA stops generating RR-sets and returns the current seed set. Authors use this approach to find an approximation of the minimum number of RR-sets. D-SSA is an SSA algorithm that dynamically adjusts parameters. However, K. Huang et al. (2017) pointed out that SSA/D-SSA has issues in algorithm efficiency analysis, and they corrected and improved these issues. Based on the experimental results in the original paper and K. Huang et al. (2017), it can be concluded that the efficiency of SSA/D-SSA is generally higher than that of IMM, but the efficiency of IMM is still higher than that of SSA/D-SSA when the number of seed nodes is small (such as several to dozens).

After that, Tang, Tang, Xiao, and Yuan (2018) proposed a new idea of online processing for influence maximization. Specifically, this approach does not need to input the accuracy requirement μ of the approximation ratio. During the implementation of the algorithm, when a user pauses it, this method uses half of the generated RR-sets to give the seed set at the current time step (using the NodeSelection subfunction) and uses the other half to estimate the accuracy guarantee μ of approximation ratio of this seed set. The approximation ratio is equal to the ratio of the upper bound of the optimal solution of estimation to the lower bound of the greedy solution of estimation. The approximation ratio is continuously estimated until the given requirement is satisfied. The experimental results show that the algorithm outperforms all existing methods, including IMM and D-SSA.

Although the above RIS-based algorithms achieve high efficiency in running time, they have a common issue that memory consumption during computation is large. This is because RR-sets used during computation need to be stored so that they can be used to select seed nodes in the final step. Although reducing the number of sampled RR-sets can reduce memory usage, when the average size of RR-sets is large, memory consumption is still a problem. SKIM algorithm proposed by Cohen et al. (2014) addressed these issues. SKIM algorithm efficiently computes the influence spread of nodes and selects seed nodes by constructing a reachability sketch of nodes in a stochastic context. Specifically, SKIM uses the bottom-K2 minHash method to speed up the estimation of constructed sketches' influence spread. The key idea is to perform a reverse BFS walk on the sketch and simultaneously update bottom-K minHash values of several candidate seed sets. Theoretically, SKIM has no guarantee of near-linear time, but it has a guarantee of approximation ratio. Experimentally, SKIM is comparable to TIM/TIM+.

MACHINE LEARNING ALGORITHMS

In addition to greedy-based, heuristic-based, and RIS-based algorithms introduced above, there are now some other studies using machine learning or data mining techniques to make some practical improvements for influence maximization. Panagopoulos, Malliaros, and Vazirgiannis (2020) pioneered the application of representation learning to influence maximization and proposed the IMINFECTOR method. Different from scalable algorithms of influence maximization introduced above, the IMINFECTOR method does not use time-consuming diffusion models to simulate the influence propagation process for computing the influence spread of seed sets but instead utilizes representations learned from diffusion cascades to maximize influence. The IMINFECTOR method consists of two parts. The first part is INFECTOR (influencer vectors), which is a multi-task learning neural network. It can simultaneously capture the influence relationship between nodes and the ability

of a node to create a large-scale diffusion cascade. Specifically, INFECTOR uses the logs of diffusion cascades to learn the embedding of a node initiating cascades (influencer vector) and embeddings of nodes participating in these cascades (influenced vectors). The norm of the learned influencer vector can be used to capture the ability of a node to initiate a large-scale cascade and can be used to reduce the number of candidate seed nodes. A propagation probability between an influencer and an influenced can be obtained by taking the dot product of their embeddings. The second part is the IMINFECTOR method, which is a scalable greedy algorithm. Specifically, firstly, the IMINFECTOR algorithm reformulates influence maximization as a weighted binary matching problem using propagation probabilities output by INFECTOR. Secondly, it computes a seed set using a submodular influence function to preserve a theoretical guarantee of $1 - 1/e$. Experimental results show that the IMINFECTOR algorithm outperforms the state-of-the-art algorithm, i.e., IMM, in terms of scalability, prediction accuracy, and quality of seed set.

Li et al. (2019) proposed a deep learning-based influence maximization algorithm called DISCO. The main idea of DISCO is to approximate the influence spread function $\tilde{A}(v, S)$ as $y = \tilde{A}(v, S; \Theta)$, and then use some machine learning methods to learn the value of parameter Θ , so as to predict the expected influence spread of node v . Specifically, the DISCO algorithm combines network embedding and deep reinforcement learning techniques. Firstly, it employs a network embedding technique to represent the network topology as vector-based features, which serve as the input for deep reinforcement learning. Then, during the learning phase, it uses a deep reinforcement learning technique to approximate $\tilde{A}(v, S)$ as $y = \tilde{A}(v, S; \tilde{\cdot})$. For each candidate seed node, the DISCO algorithm does not need to estimate its influence spread by sampling diffusion paths like the IMM algorithm but directly predicts the influence spread of each candidate seed node through the learned mapping function $\tilde{A}(v, S; \tilde{\cdot})$. Furthermore, DISCO can select all seed nodes simultaneously without iteratively selecting k seed nodes. The experimental results show that the DISCO algorithm outperforms current state-of-the-art non-machine learning-based influence maximization algorithms in terms of computational efficiency and influence quality. For example, the running time of DISCO is 36 times faster than that of SSA (K. Huang et al., 2017).

APPLICATIONS OF INFLUENCE MAXIMIZATION ALGORITHMS

In recent years, some researchers have combined the influence maximization techniques introduced in the previous sections with real-world scenarios (such as topic, time, and location) for specific applications. In topic-aware influence maximization, the definition of the influence is to find a seed set that maximizes the expected influence over users who are relevant to a given topic. Topic-aware influence maximization considers the topics of an item being propagated in the classical influence maximization techniques. Specifically, topic-aware influence maximization introduces the topics to represent item characteristics and user's interests and considers that the influence spread $\tilde{A}(S)$ depends not only on the seed set S but also on the topics. For example, Li, Zhang, and Tan (2015) and Nguyen, Dinh, and Thai (2016) combined temporal features with the TIM algorithm (see the Reverse Influence Sampling Algorithms section) and proposed topic-aware influence maximization. Guo et al. (2013) adopted the MC-based greedy approximation and heuristic-based methods for topic-aware influence maximization. Topic-aware influence maximization can be directly applied to online advertising.

In time-aware influence maximization, the definition of the influence is to find a seed set such that the expected number of nodes is influenced by the seed set within a time constraint. Time-aware influence maximization integrates classical influence maximization with temporal features. The classical influence maximization algorithms assume that each influence diffusion process stops only when there are no more new nodes that can be influenced. This assumption is unreasonable. In many

real-world viral marketing applications, people only care about how widely the influence is spread before a fixed time. Therefore, some researchers propose to impose a time constraint on the influence diffusion process. For example, discrete time-aware diffusion models (Chen et al., 2012; Kim et al., 2014; Liu et al., 2012; Liu et al., 2013) treat the discrete diffusion step as the time measure and limit the maximal step of the influence diffusion process. Since discrete time-aware diffusion models are the extension of the IC model, the influence maximization algorithms under these models are based on the MIA method (see Heuristic-Based Algorithms). For the continuous-time independent cascade model, Rodriguez et al. (2012) used a greedy framework with lazy forward optimization under this model to solve time-aware influence maximization. Xie et al. proposed a CELF-optimized greedy method (2015; see Monte-Carlo-Based Greedy Approximation Algorithms) for seed node selection.

Location-aware influence maximization is inspired by location-based social networks (such as Weibo and Twitter). In location-aware influence maximization, the definition of the influence is to maximize the influence spread on nodes within a given query region. The key idea of location-aware influence maximization is to maximize the influence of location-relevant users, rather than any users in the traditional influence maximization settings. For example, Li, Chen, et al. (2015) and Wang et al. (2016) adopted the standard IC model and used the MIA/PMIA model (see Heuristic-Based Algorithms) combined with spatial features to calculate the influence spread. Song et al. (2016) adopted the RIS-based approach (see Reverse Influence Sampling Algorithms) to develop a sampling-based approximation algorithm for location-aware influence maximization.

CONCLUSION

This paper systematically summarizes the mathematical model of the information and influence diffusion and its corresponding algorithms. After more than ten years of development, the study of influence diffusion has made great progress, which gives researchers a deeper understanding of the mode of influence diffusion and its optimization. However, to further develop its research and application, there are still many problems to be solved.

In the aspect of influence modeling, many models have been developed, among which some models represented by independent cascade models have also been verified to some extent in actual data. However, at present, the threshold model that is more suitable for describing complex propagation behavior still lacks effective verification of actual data. At the same time, the linear threshold model has limitations on the randomness of the threshold, and if a more general threshold model is used, it is likely that the model does not have submodule and other properties, so an effective algorithm cannot be designed. Therefore, there are still many problems to be solved for the threshold model, from data analysis to modeling and optimization.

In addition, the accuracy and effectiveness of influence diffusion learning is still a big challenge at present. Different from many forms of data analysis, big data analysis of influence propagation requires analyzing the influence intensity between any two related users, which is much more difficult than analyzing the characteristics of a user or a group. Further, influence diffusion is also aimed at the analysis of human behavior, and it is a more complex behavior, such as product purchase and accepting new ideas. Such behavior data is not easy to be mined in social media data. As most social media data is meaningless noise, behavioral diffusion such as retweets is too simplistic and very different from true behavioral diffusion for products and ideas. Therefore, the effective analysis of influence diffusion is a major bottleneck in the current research on influence diffusion.

ACKNOWLEDGMENT

This research was supported by the 2022 Jiangsu Province Major Project of Philosophy and Social Science Research in Colleges and Universities “Research on the Construction of Ideological and Political Selective Compulsory Courses in Higher Vocational Colleges” [grant number

2022SJZDSZ011]; the Research Project of Nanjing Vocational University of Industry Technology [grant number 2020SKYJ03]; and the Fundamental Research Fund for the Central Universities [grant number 30920041112].

REFERENCES

- Aral, S., & Walker, D. (2012). Identifying influential and susceptible members of social networks. *Science*, 337(6092), 337–341. doi:10.1126/science.1215842 PMID:22722253
- Arora, A., Galhotra, S., & Ranu, S. (2017, May). Debunking the myths of influence maximization: An in-depth benchmarking study. *Proceedings of the 2017 ACM international conference on management of data*, 651–666. doi:10.1145/3035918.3035924
- Borgs, C., Brautbar, M., Chayes, J., & Lucier, B. (2014, January). Maximizing social influence in nearly optimal time. *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, 946–957. doi:10.1137/1.9781611973402.70
- Budak, C., Agrawal, D., & El Abbadi, A. (2011, March). Limiting the spread of misinformation in social networks. *Proceedings of the 20th International Conference on World Wide Web*, 665–674. doi:10.1145/1963405.1963499
- Chen, W., Lakshmanan, L. V., & Castillo, C. (2013). Information and influence propagation in social networks. *Synthesis Lectures on Data Management*, 5(4), 1–177. doi:10.1007/978-3-031-01850-3
- Chen, W., Lu, W., & Zhang, N. (2012, July). Time-critical influence maximization in social networks with time-delayed diffusion process. *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 592–598.
- Chen, W., Wang, C., & Wang, Y. (2010, July). 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*, 1029–1038. doi:10.1145/1835804.1835934
- Chen, W., Wang, Y., & Yang, S. (2009, June). Efficient influence maximization in social networks. *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 199–208. doi:10.1145/1557019.1557047
- Chen, W., Yuan, Y., & Zhang, L. (2010, December). Scalable influence maximization in social networks under the linear threshold model. *2010 IEEE International Conference on Data Mining*, 88–97. doi:10.1109/ICDM.2010.118
- Chen, W., & Zhang, H. (2019). Complete submodularity characterization in the comparative independent cascade model. *Theoretical Computer Science*, 786, 78–87. doi:10.1016/j.tcs.2018.03.026
- Cohen, E., Delling, D., Pajor, T., & Werneck, R. F. (2014, November). Sketch-based influence maximization and computation: Scaling up with guarantees. *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, 629–638. doi:10.1145/2661829.2662077
- Domingos, P., & Richardson, M. (2001, August). Mining the network value of customers. *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 57–66. doi:10.1145/502512.502525
- Even-Dar, E., & Shapira, A. (2007, December). A note on maximizing the spread of influence in social networks. *International Workshop on Web and Internet Economics*, 281–286. doi:10.1007/978-3-540-77105-0_27
- Goyal, A., Lu, W., & Lakshmanan, L. V. (2011a). CELF++ optimizing the greedy algorithm for influence maximization in social networks. *Proceedings of the 20th International Conference Companion on the World Wide Web*, 47–48. doi:10.1145/1963192.1963217
- Goyal, A., Lu, W., & Lakshmanan, L. V. (2011b). SimPath: An efficient algorithm for influence maximization under the linear threshold model. *2011 IEEE 11th International Conference on Data Mining*, 211–220.
- Guo, J., Zhang, P., Zhou, C., Cao, Y., & Guo, L. (2013, October). Personalized influence maximization on social networks. *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management*, 199–208.
- He, X., Song, G., Chen, W., & Jiang, Q. (2012, April). Influence blocking maximization in social networks under the competitive linear threshold model. *Proceedings of the 2012 SIAM International Conference on Data Mining*, 463–474. doi:10.1137/1.9781611972825.40

- Hou, C., Wu, J., Cao, B., & Fan, J. (2021). A deep-learning prediction model for imbalanced time series data forecasting. *Big Data Mining and Analytics*, 4(4), 266–278. doi:10.26599/BDMA.2021.9020011
- Huang, K., Wang, S., Bevilacqua, G., Xiao, X., & Lakshmanan, L. V. (2017). Revisiting the stop-and-stare algorithms for influence maximization. *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, 10(9), 913–924. doi:10.14778/3099622.3099623
- Huang, W., Li, L., & Chen, W. (2017, February). Partitioned sampling of public opinions based on their social dynamics. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1). Advance online publication. doi:10.1609/aaai.v31i1.10507
- Immorlica, N., Kleinberg, J., Mahdian, M., & Wexler, T. (2007, June). The role of compatibility in the diffusion of technologies through social networks. *Proceedings of the 8th ACM Conference on Electronic Commerce*, 75–83. doi:10.1145/1250910.1250923
- Jung, K., Heo, W., & Chen, W. (2012, December). IRIE: Scalable and robust influence maximization in social networks. *2012 IEEE 12th International Conference on Data Mining*, 918–923.
- Kempe, D., Kleinberg, J., & Tardos, É. (2003, August). Maximizing the spread of influence through a social network. *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 137–146. doi:10.1145/956750.956769
- Kim, J., Kim, S. K., & Yu, H. (2013, April). Scalable and parallelizable processing of influence maximization for large-scale social networks? *2013 IEEE 29th International Conference on Data Engineering*, 266–277.
- Kim, J., Lee, W., & Yu, H. (2014). CT-IC: Continuously activated and time-restricted independent cascade model for viral marketing. *Knowledge-Based Systems*, 62, 57–68. doi:10.1016/j.knosys.2014.02.013
- Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J., & Glance, N. (2007, August). Cost-effective outbreak detection in networks. *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 420–429. doi:10.1145/1281192.1281239
- Li, G., Chen, S., Feng, J., Tan, K. L., & Li, W. S. (2014, June). Efficient location-aware influence maximization. *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, 87–98. doi:10.1145/2588555.2588561
- Li, H., Xu, M., Bhowmick, S. S., Sun, C., Jiang, Z., & Cui, J. (2019). *Disco: Influence Maximization Meets Network Embedding and Deep Learning*. arXiv:1906.07378.
- Li, Y., Chen, W., Wang, Y., & Zhang, Z. L. (2015). Voter model on signed social networks. *Internet Mathematics*, 11(2), 93–133. doi:10.1080/15427951.2013.862884
- Li, Y., Zhang, D., & Tan, K. L. (2015). Real-time targeted influence maximization for online advertisements. *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, 4(8), 1070–1081. doi:10.14778/2794367.2794376
- Liao, X., Zheng, D., & Cao, X. (2021). Coronavirus pandemic analysis through tripartite graph clustering in online social networks. *Big Data Mining and Analytics*, 4(4), 242–251. doi:10.26599/BDMA.2021.9020010
- Liu, B., Cong, G., Xu, D., & Zeng, Y. (2012, December). Time constrained influence maximization in social networks. *2012 IEEE 12th International Conference on Data Mining*, 439–448.
- Liu, B., Cong, G., Zeng, Y., Xu, D., & Chee, Y. M. (2013). Influence spreading path and its application to the time constrained social influence maximization problem and beyond. *IEEE Transactions on Knowledge and Data Engineering*, 26(8), 1904–1917. doi:10.1109/TKDE.2013.106
- Lu, W., Chen, W., & Lakshmanan, L. V. (2015). From competition to complementarity: comparative influence diffusion and maximization. *42nd International Conference on Very Large Data Bases (VLDB)*, 1–44. doi:10.14778/2850578.2850581
- Montanari, A., & Saberi, A. (2009, October). Convergence to equilibrium in local interaction games. *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, 303–312.
- Morris, S. (2006). Review of economic studies. *Contagion*, 67(1), 57–78.

- Newman, M. E. J. (2010). *Networks: An introduction*. Oxford University Press. doi:10.1093/acprof:oso/9780199206650.001.0001
- Nguyen, H. T., Dinh, T. N., & Thai, M. T. (2016, April). Cost-aware targeted viral marketing in billion-scale networks. *IEEE INFOCOM 2016: The 35th Annual IEEE International Conference on Computer Communications*, 1–9.
- Nguyen, H. T., Thai, M. T., & Dinh, T. N. (2016, June). Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks. *Proceedings of the 2016 International Conference on Management of Data*, 695–710. doi:10.1145/2882903.2915207
- Nitu, P., Coelho, J., & Madiraju, P. (2021). Improvising personalized travel recommendation system with recency effects. *Big Data Mining and Analytics*, 4(3), 139–154. doi:10.26599/BDMA.2020.9020026
- Ok, J., Jin, Y., Shin, J., & Yi, Y. (2016). On maximizing diffusion speed over social networks with strategic users. *IEEE/ACM Transactions on Networking*, 24(6), 3798–3811. doi:10.1109/TNET.2016.2556719
- Panagopoulos, G., Malliaros, F., & Vazirgiannis, M. (2020). Multi-task learning for influence estimation and maximization. *IEEE Transactions on Knowledge and Data Engineering*, 1. doi:10.1109/TKDE.2020.3040028
- Peng, C., Zhang, C., Xue, X., Gao, J., Liang, H., & Niu, Z. (2021). Cross-modal complementary network with hierarchical fusion for multimodal sentiment classification. *Tsinghua Science and Technology*, 27(4), 664–679. doi:10.26599/TST.2021.9010055
- Qi, L., He, Q., Chen, F., Dou, W., Wan, S., Zhang, X., & Xu, X. (2019). Finding all you need: Web APIs recommendation in web of things through keywords search. *IEEE Transactions on Computational Social Systems*, 6(5), 1063–1072. doi:10.1109/TCSS.2019.2906925
- Qi, L., He, Q., Chen, F., Zhang, X., Dou, W., & Ni, Q. (2020). Data-driven web APIs recommendation for building web applications. *IEEE Transactions on Big Data*, 8(3), 685–698. doi:10.1109/TBDATA.2020.2975587
- Qi, L., Yang, Y., Zhou, X., Rafique, W., & Ma, J. (2021). Fast anomaly identification based on multi-aspect data streams for intelligent intrusion detection toward secure Industry 4.0. *IEEE Transactions on Industrial Informatics*, 18(9), 6503–6511. doi:10.1109/TII.2021.3139363
- Rodriguez, M. G., & Schkopf, B. (2012). Influence maximization in continuous time diffusion networks. *Cement and Concrete Composites*, 34(5), 684–691.
- Song, C., Hsu, W., & Lee, M. L. (2016, October). Targeted influence maximization in social networks. *Proceedings of the 25th ACM International Conference on Information and Knowledge Management*, 1683–1692.
- Sun, L., Ping, G., & Ye, X. (2021). PrivBV: Distance-aware encoding for distributed data with local differential privacy. *Tsinghua Science and Technology*, 27(2), 412–421. doi:10.26599/TST.2021.9010027
- Tang, J., Tang, X., Xiao, X., & Yuan, J. (2018, May). Online processing algorithms for influence maximization. *Proceedings of the 2018 International Conference on Management of Data*, 991–1005. doi:10.1145/3183713.3183749
- Tang, Y., Shi, Y., & Xiao, X. (2015, May). Influence maximization in near-linear time: A martingale approach. *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, 1539–1554. doi:10.1145/2723372.2723734
- Tang, Y., Xiao, X., & Shi, Y. (2014, June). Influence maximization: Near-optimal time complexity meets practical efficiency. *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, 75–86. doi:10.1145/2588555.2593670
- Wang, F., Wang, L., Li, G., Wang, Y., Lv, C., & Qi, L. (2021). Edge-cloud-enabled matrix factorization for diversified APIs recommendation in mashup creation. *World Wide Web (Bussum)*, 1–21. doi:10.1007/s11280-020-00825-8
- Wang, F., Zhu, H., Srivastava, G., Li, S., Khosravi, M. R., & Qi, L. (2021). Robust collaborative filtering recommendation with user-item-trust records. *IEEE Transactions on Computational Social Systems*, 1–11. doi:10.1109/TCSS.2021.3064213

- Wang, X., Zhang, Y., Zhang, W., & Lin, X. (2016). Efficient distance-aware influence maximization in geo-social networks. *IEEE Transactions on Knowledge and Data Engineering*, 29(3), 599–612. doi:10.1109/TKDE.2016.2633472
- Wu, W., Chen, M., Li, J., Liu, B., Wang, X., & Zheng, X. (2021). Visual information based social force model for crowd evacuation. *Tsinghua Science and Technology*, 27(3), 619–629. doi:10.26599/TST.2021.9010023
- Xie, M., Yang, Q., Wang, Q., Cong, G., & De Melo, G. (2015, February). Dynadiffuse: A dynamic diffusion model for continuous time constrained influence maximization. *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 346–352. doi:10.1609/aaai.v29i1.9203
- Xu, X., Fang, Z., Qi, L., Zhang, X., He, Q., & Zhou, X. (2021). Tripres: Traffic flow prediction driven resource reservation for multimedia IoV with edge computing. *ACM Transactions on Multimedia Computing Communications and Applications*, 17(2), 1–21. doi:10.1145/3401979
- Xu, X., Fang, Z., Zhang, J., He, Q., Yu, D., Qi, L., & Dou, W. (2021). Edge content caching with deep spatiotemporal residual network for IoV in smart city. *ACM Transactions on Sensor Networks*, 17(3), 1–33. doi:10.1145/3447032
- Xu, X., Tian, H., Zhang, X., Qi, L., He, Q., & Dou, W. (2022). DisCOV: Distributed COVID-19 Detection on X-Ray Images with Edge-Cloud Collaboration. *IEEE Transactions on Services Computing*, 15(3), 1206–1219. doi:10.1109/TSC.2022.3142265
- Ye, M., Liu, X., & Lee, W. C. (2012, August). Exploring social influence for recommendation: a generative model approach. *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 671–680. doi:10.1145/2348283.2348373
- Yildiz, E., Acemoglu, D., Ozdaglar, A. E., Saberi, A., & Scaglione, A. (2011). *Discrete opinion dynamics with stubborn agents*. Social Science Electronic Publishing. doi:10.2139/ssrn.1744113
- Yuan, L., He, Q., Chen, F., Zhang, J., Qi, L., Xu, X., Xiang, Y., & Yang, Y. (2021). CSEdge: Enabling collaborative edge storage for multi-access edge computing based on blockchain. *IEEE Transactions on Parallel and Distributed Systems*, 33(8), 1873–1887. doi:10.1109/TPDS.2021.3131680

Jun Hou received the Ph.D. degree from the Nanjing University of Science and Technology, China, in 2019. She is currently an Associate Professor with the Nanjing Institute of Industry Technology, China. She has published dozens of articles in prestigious journals and top-tier conferences. Her research interests include ideological education and data mining. She serves a PC member for several international conferences.

Shiyu Chen is a PhD student at Nanjing University of Science and Technology in China. Her main research direction is big data mining.

Huaqiu Long is currently pursuing the master's degree in Intelligent Manufacturing Department, Wuyi University. Now, he is also a teacher in the university laboratory. His research interests include information security, computing system management, and data mining.

Qianmu Li received the BSc and PhD degrees from Nanjing University of Science and Technology, China, in 2001 and 2005, respectively. He is currently a full professor with the School of Computer Science and Engineering, Nanjing University of Science and Technology, China. His research interests include information security.