International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 11 Issue 5 ǁ May 2023 ǁ PP. 744-753

Influence Maximization Algorithm based on Node Similarity and Average Path Distance

Qiang Zhang，Yue Meng

Abstract
Social network is an important information dissemination platform in today's era, and a major channel for people to express their opinions, share their feelings and interact and communicate. In social networks, some users have high influence and can attract more followers, trigger more discussions, and even influence public opinion and social change. Therefore, how to find the influence of a user, organization, brand, event, etc. in social networks has become an important and challenging problem. To address this problem this paper proposes a new solution and conducts the following research, and proposes the CDBIM algorithm.First, the topological similarity of network nodes and the topic similarity are considered simultaneously, and the similarity of node topology and semantic content are taken into account when calculating the node similarity, and then the calculated results are used to update the created social network model for community discovery. Secondly, the intra-community decomposition is performed in the delineated thematic community structure, the core nodes are identified using an improved kernel decomposition algorithm combining similarity and path distance - kpi algorithm, and the node similarity average influence size is compared into the candidate seed set, and finally IC model propagation is performed for the candidate seed set.

Keywords: *Community detection; Node similarity; Influence maximization; IC model*

Date of Submission: 14-05-2023 Date of acceptance: 26-05-2023

I. Introduction

Online social networks have gradually evolved into giant information distribution platforms comparable to or even surpassing mainstream information media such as newspapers and broadcasting, and people involved in them actively or passively receive all kinds of known and unknown information pushed by the platforms and come into contact with all kinds of strangers.

The problem of influence maximization is a profoundly realistic research value in social network analysis and has a wide range of application prospects [1]. The influence maximization problem was proposed by Domimgos et al [2] in 2001, and the solution given by Domimgos et al was to use Markovian prediction to deal with the influence maximization problem. Subsequently, Kempe et al [3] applied the Independent Cascade (IC) and Linear Threshold (LT) models to the influence maximization problem and defined the influence maximization problem as a discrete optimization problem for the first time.The IC and LT propagation models have since become The IC and LT propagation models have since become two of the most classical propagation models in the field of influence maximization problem research. Although the original greedy algorithm can effectively solve the influence maximization problem, its high computational cost cannot be applied to large-scale social networks.New Greedy [4], Mix Greedy [5] and CELF [6] algorithms were developed to solve this problem. However, these algorithms still have significant shortcomings in balancing influence size and time complexity.

In this study, we propose a new influence maximization algorithm based on community node similarity and path distance, called the community discovery-based influence maximization algorithm (CDBIM algorithm). This algorithm achieves the balance of time complexity and influence size through the following three advantages. First, we calculate the topic similarity between nodes in the social network based on the topic semantic information, and fuse the node topological similarity to make the social network model into a topic social network. Then we divide the social network into communities, and use the kpi accounting method to find the candidate seed set within each community of the division result. Finally, the final seed set is found using the celf algorithm on the improved ic model to complete the calculation of the influence maximization algorithm.

The rest of the paper is organized as follows. Section 2 presents the work related to the influence maximization algorithm. Section 3 presents our CDBIM algorithm. Section 4 provides the experimental dataset and evaluation metrics, and presents the experimental results and their analysis. Section 5 describes the conclusions of this paper.

II. Related work

Our work is related to friend recommendation in social networks. Here we briefly review these related works.

Influence maximization algorithms based on greedy strategies and influence maximization algorithms based on heuristic strategies are the most studied influence maximization algorithms by researchers. The respective algorithms under both types of strategies have obvious advantages and disadvantages. For the algorithm with greedy strategy, due to the large number of Monte Carlo simulation calculations in its process, it is impractical to be applied to large-scale social networks due to its high time complexity and low operational efficiency despite the high accuracy of describing the influence propagation values. For algorithms with heuristic strategies, the accuracy of the influence propagation values is often not guaranteed, although a low running time is ensured [7]. The Degree algorithm, which selects seed nodes based on the metrics of basic social network characteristics, was the first heuristic algorithm used to solve the influence maximization problem. The literature [8] proposed the Degree Discount algorithm based on the Degree heuristic algorithm, combined with the independent cascade propagation model. The core idea of this algorithm is that in the whole influence propagation process of a node, if a node u has a node w among its in-degree neighbors that has already been selected as a seed, then there is an influence overlap between these two nodes, so the degree of node u needs to be discounted to eliminate the influence caused by the influence overlap and thus reduce the error.Qiu, Tian et al.[9] proposed a method to reduce the error by performing candidate node The LGIM algorithm, which uses a node filtering strategy and an objective function to achieve a balance between accuracy and computational efficiency, was proposed by Yipeng Chen et al. Combining user semantic information with influence maximization, its essential purpose remains to find a set of user seed sets that maximize influence.

After the above discussion, we believe that we need to find other ways to balance the time complexity and impact size of the algorithm.

III. Algorithm implementation

This section introduces the CDBIM algorithm, the framework of which is shown in Figure 1. The work of CDBIM algorithm is divided into the following three stages. In the first step, we calculate the topic similarity between nodes based on the topic semantics and fuse the topological similarity to construct the topic social network. In the second part, we use the slpa algorithm for community partitioning. In the third step, we use kpi decomposition on the divided communities. In the fourth step, based on the results of kpi decomposition, the influential nodes are selected to join the candidate seed set. In the fifth step, we use the celf algorithm on the IC-S model to obtain the final seed set and calculate its influence size.

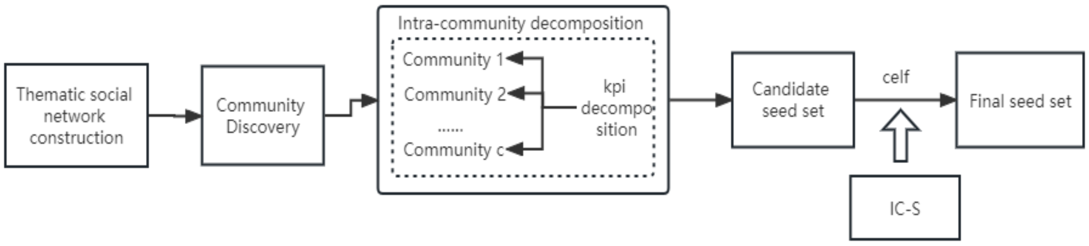


Fig 1. The framework of the proposed approach CDBIM

3.1. Thematic social network construction

Definition 1 (Topological Network) Input network structure G (V, E, Pt), which consists of nodes V, edges E, and weights Pt. The weight Pt's are calculated from the degree fraction of the nodes, which is calculated by Equation 1..

$$Pt = 1/Degree \tag{1}$$

where Degree is the degree of the node.

After the topological network is constructed, we obtain the semantic vector by the Bert model and calculate the topic similarity between nodes using the cosine similarity formula. In recent years, with the research on semantic information, many deep learning algorithms can be used to analyze the semantics of text, such as word2vec [10], LSTM [11] and Bert [12].The Bert model is a pre-trained language model based on Transformer's bi-directional encoder representation, which was proposed by Google in 2018[13] and has attracted wide attention and application in the field of natural language processing. Cosine similarity is a formula commonly used to calculate the similarity between nodes. The similarity of nodes u, v based on topic i can be calculated from Equation 2.

$$S(u,v)=\frac{(V_{iu}*V_{iv})}{\sqrt{V_{iu}^{2}}\sqrt{V_{iv}^{2}}} \qquad (2)$$

where *Viu* and *Viv* denote the vectors of node *u* and node *v* under topic *i*, respectively.The cosine similarity is found by the cosine of the angle between the vectors, and the closer the value is to 0, the further the two nodes are from each other, and conversely, the closer it is to 1, the more similar they are.

We combine the above two probabilities to construct a thematic social network, and for the influence probability P between nodes in the thematic social network, it is calculated by the topological probability Pt and the semantic probability Pst according to Equation (3).

$$P=\theta P_{st}+(1-\theta)P_t \qquad (3)$$

Where, θ is the parameter that balances the topological weights *Pt* and thematic weights *Pst* in the proportion of total weights *P*.

3.2. Community Discover

With the above sections, we construct a social network containing semantic information. In this subsection, we utilize currently known community discovery algorithms for community discovery.Xie et al. proposed the SLPA (Speaker-listener Label Propagation Algorithm) algorithm [14] to address the shortcomings of the label propagation LPA algorithm [45] which is only applicable to mining non-overlapping communities.The SLPA algorithm can The SLPA algorithm can efficiently mine overlapping and non-overlapping communities by adjusting the parameters. Besides, the SLPA algorithm is widely used in overlapping community discovery because of its better performance and effectiveness.

The process of this algorithm can be summarized in the following four steps: (i) Assigning labels: The topic of each node in the network is considered as its label. Nodes belonging to the same label can be considered as the same community. (ii) Label propagation: A node is randomly selected as a "listener". Then, the neighboring nodes of this "listener" will propagate their respective labels to them with influence probability P. (iii) Statistics: The tags received by the "listener" from its neighbors are counted. Then, the most received tags are added to its tags. (iv) Result processing: Nodes with the same label are classified into the same community. The pseudo code for the SLPA algorithm is shown below.

```
Algorithm 1：SLPA
Input：Social Network $G(V, E, Pt, Ps)$
Output：Community
//Assigning labels
1:for $i$ in $V$ :
2:    Consider each node i's topic as its label
//Label propagation
3: Iterate T times：
4:     for $i$ in $V$ :
5:         Node i as "listener"
6:         Node i's neighbor node j as "speaker"
7:         for $j$ in range(1,len($N(i)$)):
8:             "Speaker" transmits its label to the speaker according to the rules of the speech
9:             "Listener" receives labels from its neighbors according to the listening rules
//Data statistics
10:    Count the tags received by i from its neighbors
// Result processing
11:for $i$ in $V$ :
12:     Remove tags received by i with probability of occurrence less than r. Remain as their final labels
```

Suppose that the number of nodes and edges in the social network G are m and n. In the label assignment phase, its time complexity for assigning labels to n nodes is O(n). Then, at the label propagation nodes, its time complexity is O(Tn). Then, both data statistics and result processing consume O(n). Therefore, by the above analysis, we can find the time complexity of the SLPA algorithm as $O(n)+O(Tn)+O(n)+O(n)\approx O(Tn)$.

3.3. Intra-community decomposition

After the community discovery using the SLPA algorithm, the next main task is to find the nodes with higher influence. To solve this problem, this subsection is implemented by the following two steps: find the core nodes within the community using k-core decomposition. Then, using the community core as the base point, the search for nodes with high influence is expanded outward in layers.

The classical k-core algorithm, which relies only on the network topology for the decomposition process, is not precise enough in the granularity of the result metrics it calculates. In order to avoid the coarse-grained metrics of the k-core decomposition algorithm, this paper improves the accounting method and proposes a new algorithm kpi (k-path-influence)-core method based on the k-core method.

In order to better explain the kpi accounting method, we first introduce the concept of path influence degree here. The classical accounting method is only based on the centrality concept of the network topology, and the node with the largest degree is selected as the core node, and then the propagation spreads with this node as the center of the circle. However, this method considers the network topology too much and underestimates the importance of the propagation probability in this process, and also does not consider the path position of this node in the association. For example, on a dating site there is a user a who has friendships with several users, but the path distance of a from the rest of the core nodes in the community is far (low-value customers tend to attach to high-value customers to form groups, which leads to a lengthened path among high-value customers), and user a seldom outputs views although he has established more friendships, in this case if we still rely on the size of the node degree to measure influence, it will lead to a large error in the evaluation of influence. In subsection 3.1, we have updated the similarity network that combines topic similarity and topological similarity in social networks, based on which we propose the concept of Path Influence Degree (*PId*). The *PId* of node *u* is calculated as shown in Equation 4.

$$PId(u) = \frac{d(u) + \sum_{v \in N(u)} p(u,v) * d(v)}{PD_a(u)} \qquad (4)$$

where d(u) and d(v) are the degrees of node u and v, respectively, denoting the probability of influence of the node, which is calculated by Equation 3, N(u) denotes the set of neighboring nodes of node u, and PDa(u) denotes the average path length from node u to the remaining nodes, which is calculated by Equation 5.

$$PD_a(u) = \frac{\sum_{i \in N} PD(u,i)}{|N|} \qquad (5)$$

where *PD(u,i)* is denoted as the path distance from node *u* to node *i* and *N* is the number of community nodes.
To more clearly define the concept of path impact degree, we introduce the process of calculating *PId* through a simple community example, such as the network shown in Figure 2.

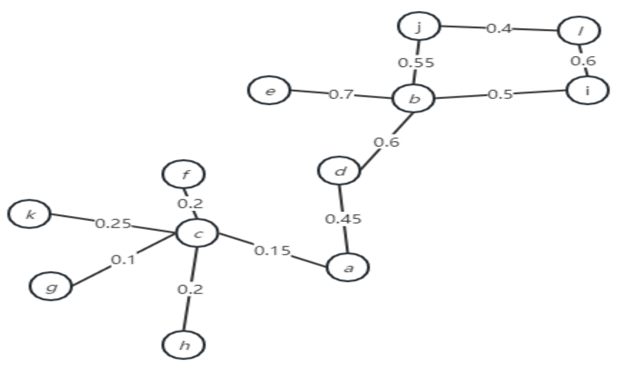


Fig 2. Example network for path impact degree calculation

The *PId* of each node can be calculated by Equation 4, for example, for node *b*, its *PId* = (4+0.7*1+0.55*2+0.5*2+0.6*2)/2.1=3.8, and the *PId* of all 12 nodes can be calculated by repeating the calculation, and then the value of kpi is calculated according to the *PId* using the rounding principle, and the final results of *PId* and kpi values are shown in Table 1.

Table 1. Results of node PId and kpi values

Node	Degree	PId	kpi
a	2	1.72	2
b	4	3.8	4
c	5	2.25	2
d	2	2.65	3
e	1	1.03	1
f	1	0.6	1

g	1	0.5	1
h	1	0.6	1
i	2	1.79	2
j	2	1.72	2
k	2	0.72	1
l	2	1.23	1

3.4. Candidate seed set

Through the above calculation of path impact degree, we can summarize the process of kpi-core method as follows. (i) For a given social network G, first calculate the path impact degree of nodes in the network based on Equation 4, and update this index into the network. (ii) Traverse the network G, put all nodes in the network with path impact degree less than 1.5 into the hierarchy with point kpi=1, and remove the node from the network. After completing this step, form a new node G1 from the remaining nodes in the network. (iii)Repeat the process of (ii) for G1 all nodes are assigned kpi values.

The number of community candidate seed nodes can be assigned according to the ratio of the community size to the overall size of the social network. For a community Ci, the number of candidate seed nodes assigned to it is calculated by Equation 6.

$$C_i = k * \frac{|N_i|}{N^-} \qquad (6)$$

where *k* denotes the number of the final seed node set. *Ni* denotes the number of nodes in the community *Ci* and *N* denotes the total number of nodes in the social network.

For node u, its Similarity Average Influence (SAI) is calculated by Equation 7.

$$SAI(u) = \sum_{i \in k_{pi}[u]} S(u,i) \qquad (7)$$

where kpi[u] denotes all nodes in the kpi layer where node u is located. s(u,i) is the semantic similarity between node u and node i, and its calculation formula is shown in Equation 2.

Taking the network in Figure 2 as an example, the similar vacation set among the 12 nodes in the figure is shown in Table 2. Combining Table 1 and Table 2, we can see that nodes b and d are located in the most central position in this community structure, and their kpi values are 4 and 3 respectively, so the influence of both b and d is the largest. For nodes a, c, i and j, their kpi values are all 2, which indicates that they belong to the same layer and have similar influence. Next we calculate the similar influence degree of the nodes according to Equation 7. As shown in Table 2, we can conclude that SAI(i) > SAI(j) > SAI(a) > SAI(c). Similarly, nodes e, f, g, h, k and l all have kpi value of 1. According to the SAI values, their influence in descending order is: k, g, e, l, h.

Table 2. Similar average impact degree of nodes

	a	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*	*l*	*SAI*
a	1	0.4	0.15	0.45	0.3	0.2	0.25	0.35	0.41	0.48	0.56	0.6	1.04
b	0.4	1	0.23	0.6	0.7	0.1	0.25	0.66	0.5	0.55	0.05	0.45	\
c	0.15	0.23	1	0.23	0.19	0.2	0.1	0.2	0.39	0.27	0.25	0.4	0.81
d	0.45	0.6	0.23	1	0.42	0.22	0.38	0.08	0.44	0.16	0.34	0.31	\
e	0.3	0.7	0.19	0.42	1	0.28	0.25	0.44	0.56	0.21	0.12	0.35	1.44
f	0.2	0.1	0.2	0.22	0.28	1	0.1	0.24	0.3	0.51	0.43	0.16	1.21
g	0.25	0.25	0.1	0.38	0.25	0.1	1	0.23	0.24	0.3	0.51	0.43	1.52
h	0.35	0.66	0.2	0.08	0.44	0.24	0.23	1	0.56	0.41	0.21	0.06	1.18
i	0.41	0.5	0.39	0.44	0.56	0.3	0.24	0.56	1	0.69	0.43	0.6	1.66
j	0.48	0.55	0.27	0.16	0.21	0.51	0.3	0.41	0.69	1	0.21	0.4	1.44
k	0.56	0.05	0.25	0.34	0.12	0.43	0.51	0.21	0.43	0.21	1	0.26	1.53
l	0.6	0.45	0.4	0.31	0.35	0.16	0.43	0.06	0.6	0.4	0.26	1	1.26

Algorithm 2 describes the entire flow of subsections 3.1-3.4. First, the topic similarity is updated for social network G to construct a topic-based social network (row 1). Then the community division is performed using the SLPA community discovery algorithm (line 2). Next, a kpi-core decomposition is performed for each community to divide the nodes according to their kpi values. The number of candidate seed nodes that should be assigned to each community is calculated according to Equation 6, (lines 3-5). Then, the seed nodes are found by sequentially expanding outward from the core layer using the metric of similar average influence, and the nodes with higher influence are added to the candidate seed set (rows 6-8). Finally, the candidate seed set obtained above is returned (line 9).

```
Algorithm 2: Candidate seed set selection(CSS)

Input: Social Network $G(V, E)$
Output: Candidate seed set (CS)
// Thematic social network construction
1: Update the topology network by Equation 2
// Community discovery with SLPA algorithm
2: C=SLPA(G)
// Intra-community decomposition
3:for $c$ in $C$:
4:   $kpi(c)$
5:    $num$(c) by Equation 6
6:for $i$ in $num(C)$:
7:        for $j$ in $num(c)$:
8:             $CS$.add(max $SAI$) // Nodes with similar average influence within the community join the candidate seed set
9: return $CS$
```

3.5. Candidate seed set

Users in realistic social networks often like to get along with like-minded people, and the more two users have common topics to chat with each other, the higher the degree of fit, and the greater the possibility of influencing each other. Therefore, the traditional IC propagation model only considers the topological similarity of the social network distortion propagation process is will make the experimental results have a large error, in order to overcome this shortcoming, we use the node similarity to improve the IC propagation model, using the CELF algorithm for the final seed set determination. The main process is as follows:
(i)Improve the IC propagation model by adding the node similarity (including attribute similarity and topic similarity) to the calculation of the propagation probability, so that the whole propagation process is more realistic. And the CELF algorithm is used to select the most influential node from the candidate seed set as the seed node.
(ii)When activating the next target node, the existing seed node is compared with its neighbor nodes for similarity, and it is observed whether the activation probability between the neighbor node and the node exceeds a given threshold λ. If it exceeds λ, the node is included in the seed node set. Otherwise, the node is added to the final seed set.
(iii)The above process is repeated until the number of seed sets reaches k.

3.6. CDBIM algorithm

Based on the above analysis of subsections 3.1, 3. 2, 3. 3, 3.4 and 3.5, we propose the CDBIM influence maximization algorithm. Algorithm 3 illustrates the basic idea of CDBIM as follows.

```
Algorithm 3: CDBIM

Input: Social Network $G(V,E,P,T)$
Output: seed set S
// Find the candidate seed set according to Algorithm 2
1: CSS(G)→CS
// Final seed set selection
2: while u in CS
3:       S.add(u)
4:       if len(S)<k:
5:       v=CELF(CS)
6:            if Neighbors.S(u,v) <λ:
7:            S.add(v)
8:       if len(S)>k:
9:       break
10: return S
```

The time complexity of the CDBIM algorithm consists of five main components: Thematic social network construction, community discovery, intra-community decomposition, candidate seed set selection and final seed set selection. The time complexity of theme network construction is O(nm), n is the number of nodes and m is the number of edges; the time complexity of community discovery is O(Tn), where T is the number of iterations of SLPA algorithm; the time complexity of kpi-core algorithm for intra-community decomposition is O(n); then the candidate seed set selection requires O(hn), h is the number of candidate seed sets, the number of which is much smaller than the number of nodes n ; finally, the time complexity of the CELF algorithm is O(h2). In summary, the time complexity of the CDBIM algorithm is $O(mn)+O(n)+O(hn)+O(h^2)\approx O(hn+mn)$.

IV. Experiment Results and Analysis

In this section, we conduct the experiments to evaluate the proposed algorithm over four real-world networks. At the same time, we compare the CDBIM algorithm with the other three algorithms.

4.1. Experiment setup

Dataset set. Extensive experiments are carried out on the following networks.

- The Oregon dataset: This dataset, from the Physical Sciences Research Collaborative Network, collects traffic movement views and point-to-point information for the state of California.
- The Email-Enron dataset: The dataset is a dataset consisting of emails exchanged between company executives. The exchange of emails between user a and b is considered as an edge in the network graph, and the title of the email can be used as a content feature.
- The Wiki-vote dataset: Consists of Wikipedia site user data. Users, voting relationships, and voting topics are used as node, edge, and content features, respectively.
- The Ca_AstroPh dataset: A network consisting of articles submitted to e-print Arxiv and the authors, with authors as nodes, partnerships as edges, and article titles as content features.

The statistics of the context datasets are shown in Table 3.

Table3: Statistics of the datasets

Data	Oregon	Email-Enron	Wiki-vote	AstroPh
#Node	11011	35782	7145	18762
#Edge	22678	183939	102987	197915
max degree	2278	1237	1325	627
Average Weighted	2.66	4.15	26.12	5.23
Average clustering coefficient	0.35	0.52	0.17	0.71

Algorithms Compared: To verify the performance of the CDBIM algorithm in terms of both runtime and impact, this paper compares the CDBIM algorithm with three more classical algorithms on the six different types of data sets mentioned above, namely the CCA, CoFIM and GRIS-SIM algorithms.

- CCA algorithm: This algorithm is a heuristic algorithm, and the CCA algorithm is based on the k-core decomposition theory, which finds the nodes with large outer centrality from the core nodes layer by layer as seed nodes.
- CoFIM algorithm: This algorithm is a simple and fast solution to the problem of maximizing the total influence influence in a network based on submodularity theory.
- GRIS-SIM algorithm: This algorithm introduces a semantic-aware influence maximization problem (SIM) in order to consider the semantics of nodes. The problem finds a set of seed sets that maximize influence propagation by evaluating influence propagation based on semantic values under a given model that integrates semantic information of users with maximized influence.

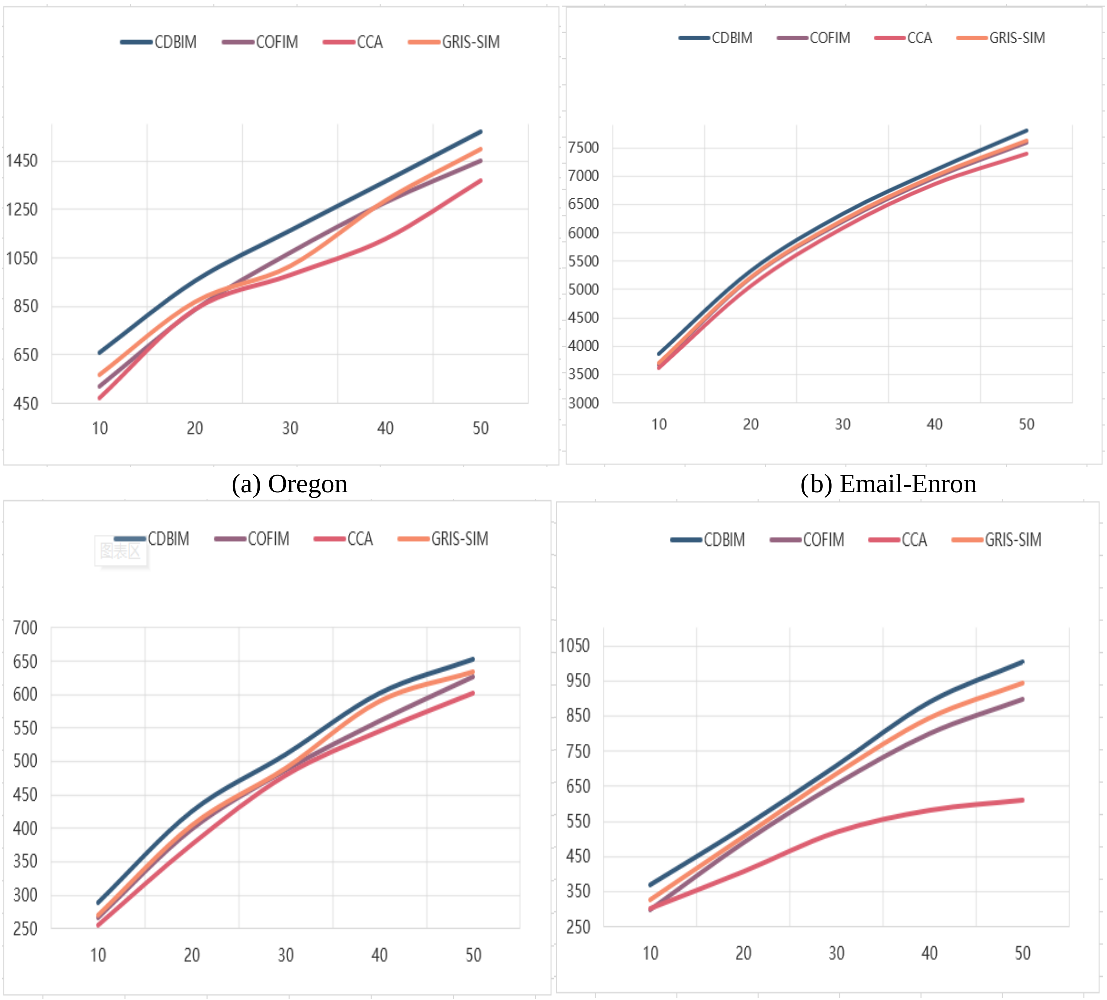
4.2. Experiment results

In the social network influence maximization problem, the algorithm is mainly judged by the influence size. The higher the influence, the higher the gain of the algorithm is indicated. In this paper, the CDBIM algorithm is compared with three comparison algorithms, CCA, CoFIM and GRIS-SIM, on four datasets, Oregon, Email-Enron, Wiki-vote, and Ca_AstroPh, for the experimental results. In addition, the number of seed nodes k is selected as 10, 20, 30, 40 and 50 for the experiments.

Size of influence. Figures 3(a), (b), (c) and (d) depict the performance of four algorithms, CDBIM, CCA, CoFIM and GRIS-SIM, on four datasets, Oregon, Email-Enron, Wiki-vote and Ca_AstroPh, in terms of influence size, respectively. By comparing Figure 3, it can be seen that the performance of the CCA algorithm is relatively poor overall. For example, for a seed set of 50, the CCA algorithm is about 71.29 % and 46.29 % lower than CDBIM in Figure 3(c) and Figure 3(d). Compared with CoFIM algorithm, the impact of CoFIM algorithm is 1.59 %, 3.13 %, 51.92 % and 31.94 % higher than CCA on four datasets such as Email-Enron, Wiki-vote, etc. Similarly, when the seed set is 50, GRIS-SIM outperforms CCA by 2.76%, 3.72%, 59.78%, and 31.58% on the above four datasets, respectively. Overall, the CDBIM and GRIS-SIM algorithms outperform the remaining two algorithms. For example, on the Email-Enron and Wiki-vote datasets, the CDBIM algorithm outperforms the CCA and COFIM algorithms by 23.18% and 15.86%. Similarly, the GRIS-SIM algorithm is higher than the CCA and

COFIM algorithms by 13.06% and 6.48% on both datasets. The above phenomenon may arise because both CDBIM and GRIS-SIM algorithms take into account the topic similarity between nodes. As can be seen from Figure 3, the CDBIM algorithm outperforms the remaining three algorithms on all six datasets, and on the Ca_AstroPh dataset, CDBIM is 11.73%, 43.25% and 7.95% higher than the COFIM, CCA and GRIS-SIM algorithms, respectively. Similarly, on the Ca-condmat dataset, CDBIM is 14.67%, 25.13%, and 10.34% higher than the COFIM, CCA, and GRIS-SIM algorithms, respectively. On the Oregon dataset, the CDBIM algorithm is 12.93%, 7.95%, 15.11%, 3.23% and 6.17% higher than the GRIS-SIM algorithm on five different number of seed sets. While on the Ca-condmat dataset, the CDBIM algorithm is 6.94%, 10.63%, 8.19%, 7.08% and 10.03% higher than the GRIS-SIM algorithm on five different number of seed sets. This comparison leads to the conclusion that the CDBIM algorithm can effectively handle the influence maximization aspect and improve the influence through the kpi-core algorithm, the introduction of topic similarity, and the improved IC model.

Considered together, the experimental results of CDBIM outperform the heuristic algorithm as well as the algorithm with semantic information considered, which can improve the impact of the experiment to a certain extent and also prove the effectiveness of CDBIM.



(a) Oregon (b) Email-Enron

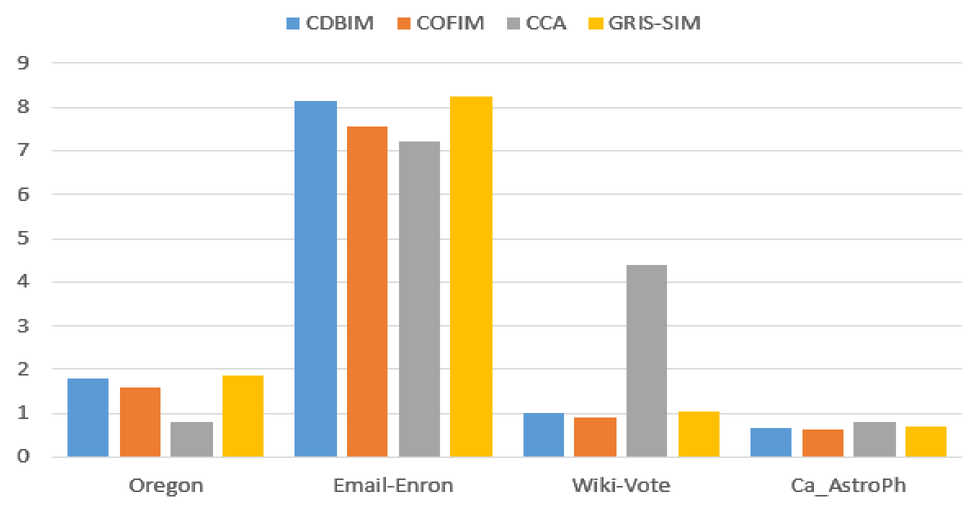
(c) Wiki-vote (d) Ca_AstroPh

Fig 3. Influence size of different algorithms on four datasets

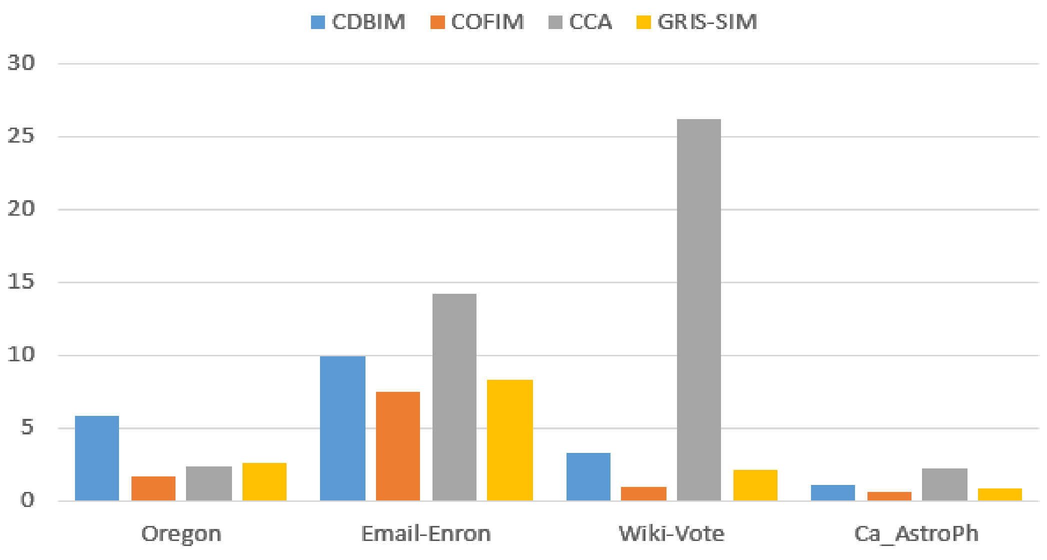
Running time. As shown in Figure 4, for the Email dataset, the CDBIM algorithm has a running time of 10.23 and 8.26 seconds for k=10, 50, respectively. Similarly, for the Wiki-vote dataset, the CDBIM algorithm runs about 86.26 % faster than k=50 for k=10. On the Ca_AstroPh dataset, the CDBIM algorithm runs about 32.91 % faster at k=10 than k=50. On the Oregon dataset, the CDBIM, CCA, COFIM and GRIS-SIM runtimes are about 49.25%, 1.21%, 62.14% and 25.81% faster at k=10 than k=50, respectively. Therefore, comparing k=10 and 50 shows that the running time of the algorithm grows as the number of seed sets increases. Collectively, when k=10, the CDBIM algorithm has the fastest running time, which is 8.09%, 14.26%, -4.97%, and 3.98% faster than the COFIM algorithm on the Oregon, Email, Wiki-vote, and Ca_AstroPh datasets, respectively. When k=50, the running time of COFIM algorithm is the fastest, and its running time on four datasets is 21.34%, 72.02%, -15.13% and 39.02%

faster than CDBIM algorithm, respectively. the reason for the fast running time of CoFIM algorithm is that CoFIM algorithm is a simple and fast method to calculate the total influence based on submodularity. However, as shown by the influence comparison in the previous subsection, however, the COFIM algorithm does not consider the topic semantics of the network, making its influence much lower than CDBIM. both the CDBIM algorithm and the GRIS-SIM algorithm consider the topic semantic information of the network, and as can be seen from Figure 4, when the seed set is 10, the influence of Oregon, Email, Wiki-vote, Ca_ AstroPh dataset the running time of the CDBIM algorithm is 2.33%, 2.66%, -11.28% and 3.19% higher than that of the GRIS-SIM algorithm, respectively. When k=30, the running time of CDBIM algorithm is 36.54%, 4.18%, 7.35% and 9.21% higher than GRIS-SIM algorithm on Oregon, Email, Wiki-vote, and Ca_AstroPh datasets, respectively. When k=50, the running time of CDBIM algorithm is 2.38%, 3.16%, 4.87% and 2.99% higher than GRIS-SIM algorithm on Oregon, Email, Wiki-vote, and Ca_AstroPh datasets, respectively. This illustrates the balance of the CDBIM algorithm in terms of running time.

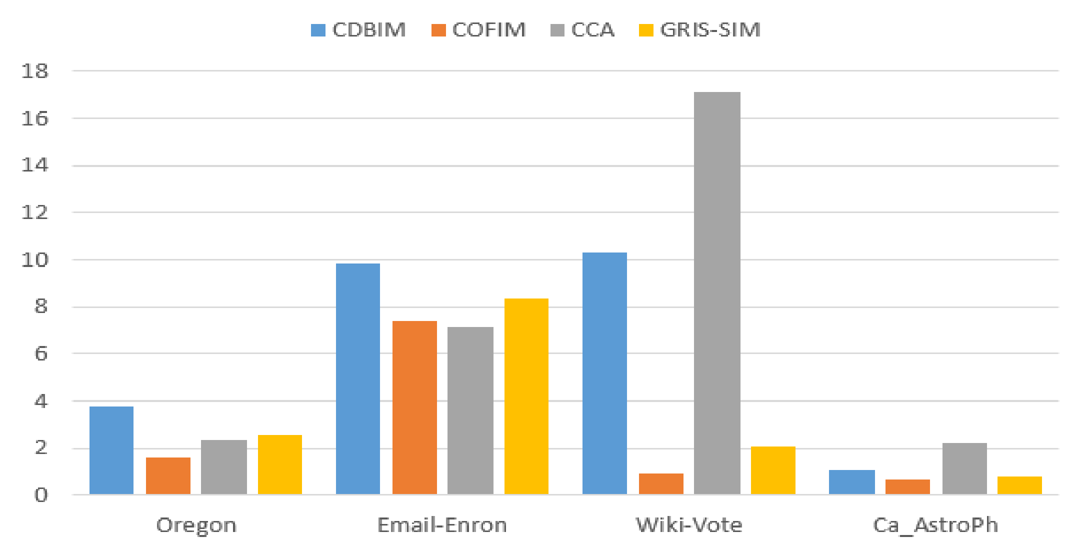
In summary, the CDBIM algorithm is able to balance the running time well while being able to guarantee the impact. Therefore, the CDBIM algorithm considers the subject of nodes and can solve the influence maximization problem more effectively.



(a) Running time of each algorithm for k = 10



(b) Running time of each algorithm for k = 30



（ c ）Running time of each algorithm for k = 50

V. Conclusions

In this paper, we perform community discovery on the constructed topic network, and then use the kpi kernel decomposition algorithm to find out the core nodes for each community, and then calculate the similar average influence among the nodes, based on which the candidate seed set is selected. Then the IC propagation model is improved to improve the propagation fidelity, and the selection of the most seed set is completed using the CELF algorithm on the basis of this propagation model. Finally, the proposed influence maximization algorithm is experimentally compared with comparative algorithms of different levels and ideas, and finally the effectiveness and reliability of the CDBIM algorithm proposed in this paper is verified.

Reference

[1]. Ding Zhaoyun et al. "A Review of Social Network Influence Research." Computer Science (2014).
[2]. Chu Y, Zhao X, Liu S, et al. An Efficient Method for Topic-Aware Influence Maximization[C]// Asia-Pacific Web Conference. Springer International Publishing, 2014.
[3]. Chen S , Fan J , Li G , et al. Online topic-aware influence maximization[J]. Proceedings of the VLDB Endowment, 2015.
[4]. Rui X , Meng F , Wang Z , et al. A reversed node ranking approach for influence maximization in social networks[J]. Applied Intelligence, 2019.
[5]. Hua Y , Chen B L , Yuan Y , et al. An Influence Maximization Algorithm Based On The Mixed Importance Of Nodes[J]. Computers, Materials and Continua, 2019, 58(2):517-531.
[6]. Shang J , Zhou S , Li X , et al. CoFIM: A community-based framework for influence maximization on large-scale networks[J]. Knowledge-Based Systems, 2016, 117(FEB.):88-100.
[7]. Swati S , Hayat K , Shahid Z . A watermarking scheme for High Efficiency Video Coding (HEVC)[J]. PLoS ONE, 2014, 9(8):e105613.
[8]. Bagheri, Esmaeil, Dastghaibyfard, et al. FSIM: A Fast and Scalable Influence Maximization Algorithm Based on Community Detection[J]. International journal of uncertainty, fuzziness and knowledge-based systems: IJUFKS, 2018.
[9]. Qiu L , Tian X , Sai S , et al. LGIM: A Global Selection Algorithm Based on Local Influence for Influence Maximization in Social Networks[J]. IEEE Access, 2020, 8:4318-4328.
[10]. Swati S , Hayat K , Shahid Z . A watermarking scheme for High Efficiency Video Coding (HEVC)[J]. PLoS ONE, 2014, 9(8):e105613.
[11]. Gf A , Schmidhuber J , F Cummins. Learning to Forget: Continual Prediction with LSTM[M]. Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale, 1999.
[12]. Bing L . Sentiment Analysis and Opinion Mining[C]// Synthesis Lectures on Human Language Technologies 5.1 (2012): 1-167. Morgan & Claypool, 2011.
[13]. Devlin J , Chang MW, LeeK , et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding[J],2018.
[14]. Xie J , Szymanski B K , Liu X . SLPA: Uncovering Overlapping Communities in Social Networks via A Speaker-listener Interaction Dynamic Process[J]. IEEE, 2012.