DiffuGreedy: An Influence Maximization Algorithm based on Diffusion Cascades

George Panagopoulos, Fragkiskos Malliaros, Michalis Vazirgiannis

DiffuGreedy: An Influence Maximization Algorithm based on Diffusion Cascades

George Panagopoulos, Fragkiskos D. Malliaros, and Michalis Vazirgiannis

Abstract Finding a set of nodes that maximizes the spread in a network, known as the influence maximization problem, has been addressed from multiple angles throughout the literature. Traditional solutions focus on the algorithmic aspect of the problem and are based solely on static networks. However, with the emergence of several complementary data, such as the network's temporal changes and the diffusion cascades taking place over it, novel methods have been proposed with promising results. Here, we introduce a simple yet effective algorithm that combines the algorithmic methodology with the diffusion cascades. We compare it with four different prevalent influence maximization approaches, on a large scale Chinese microblogging dataset. More specifically, for comparison, we employ methods that derive the seed set using the static network, the temporal network, the diffusion cascades, and their combination. A set of diffusion cascades from the latter part of the dataset is set aside for evaluation. Our method outperforms the rest in both quality of the seed set and computational efficiency.

1 Introduction

Albeit the massive amount of work over this problem over the past 15 years, influence maximization (IM) remains the holy grail of social network analysis. The problem, at its core, is to find a set of nodes that would infect the largest possible part of the network, if a spreading process started from them. It is proven to be NP-hard, but a greedy algorithm [1] can get at least as close as (1-1/e) to the optimum,

George Panagopoulos
École Polytechnique, France e-mail: george.panagopoulos@polytechnique.edu
Fragkiskos D. Malliaros
CentraleSupélec and Inria Saclay, France, e-mail: fragkiskos.malliaros@centralesupelec.fr
Michalis Vazirgiannis
École Polytechnique, France, e-mail: mvazirg@lix.polutechnique.fr

under the two most prevalent spreading models, the independent cascade (IC) and the linear threshold (LT). The edge weights represent the probability of influence in IC and the amount of influence in LT. Typically, IM algorithms work with uniform or degree-based edge weights. This renders their spreading estimation wildly divergent from the actual spreading [2], due to the complexity and diversity that governs spreading processes [3]. To this end, some methods focus on learning the transmission probabilities between nodes using real diffusion cascades [14]. This research branch is divided into models that use the cascades to define an inferred network [7] or to adjust the edge weights of an existing underlying network [5]. An example of the former is to infer a transmission probability between two news blogs based on how often and fast one copies the other, while an example of the latter is defining the strength of a follow relationship in twitter based on how many times one node retweeted the other. IM algorithms can then run on such inferred or weighted [2] networks to derive the seed set. Although these methods tend to approximate real spreading better, they suffer from issues of scalability and overfitting.

In this work, we propose DIFFUGREEDY, an algorithm based on SIMUGREEDY [1] that utilizes the real diffusion cascades instead of simulations over the network. Particularly, it follows the same hill climbing manner to construct the seed set iteratively, but the computation of the marginal gain is based on a candidate seed's most suitable diffusion cascade. This function is submodular, which allows us to retain the theoretical guarantees. To showcase the effectiveness of our method, we employ the temporal Sina Weibo follower network (1.7m nodes, 400m edges), that spans 32 days and is accompanied with the retweet cascades of that time span. We keep the diffusion cascades and the network of the first 25 days as the train set and the last week as the test set. We use four different IM approaches in the train set for comparison with DIFFUGREEDY. The first is ranking the nodes by k-core decomposition on the follower network, which has indicated strong correlations with influence [23]. Secondly, we employ IMM [6], one of the fastest IM algorithms, to extract a seed set efficiently from the follower network as is formed at the end of the training set. The third approach utilizes NETRATE [7] to infer a network from the diffusion cascades in the train set, and applies PMIA [8] to perform IM on it. In the final method, we use the diffusion cascades to weigh the follow edges proportionally to the nodes' activity and apply SIMPATH [9] on the resulting network.

It should be noted that a comparison between all these approaches has not been attempted in the literature before, to the best of our knowledge. This can be due to the lack of a common realistic evaluation methodology. Methods that work on static networks are evaluated based on their computational time and the estimated influence spread. On the other hand, diffusion learning methods are evaluated based on the behavior of the chosen seed set in unseen cascades. We deem the latter more realistic, and we choose to validate our seed set using the number of distinct nodes influenced by it in the test set [10]. The results indicate that Diffusion Greedy and its CELF counterpart clearly outperform the rest in terms of the seed set's influence spread in the test set. In addition, the CELF approach is an order of magnitude faster to the second fastest method. Finally, we notice that the methods based on k-core and NETRATE perform adequately, unlike the IMM. The paper is organized

as follows. Section 2 is a brief literature review on influence maximization. Section 3 delineates some state of the art influence maximization methods we employed for our comparative analysis. Section 4 describes the new algorithm we propose. Section 5 presents the dataset and the results of our experiments, along with insights and justifications. The paper concludes in Section 6 with a contribution synopsis and suggestions for future work.

2 Related Work

The basis of most IM algorithms is SIMUGREEDY [1]. Starting with an empty seed set, the algorithm adds in the set the node that provides the best marginal gain i.e. the increase of the set's influence spread, in each iteration. Due to the monotonicity and submodularity of the influence spread function under the two diffusion models, the algorithm is guaranteed to reach a near optimal solution. The most time-consuming part of SIMUGREEDY is the influence spread estimation, which has proven to be P-hard [8]. Since the diffusion models are stochastic, all possible paths of influence need to be taken into account proportionally to their probability, which is not feasible, thus Monte Carlo simulations are employed. Most attempts to improve the algorithm focus on that part. CELF [11] capitalizes on the submodularity of a node's marginal gain, which can only diminish as the seed set grows. This means that if a candidate seed's marginal gain is higher then what the rest candidates had in the previous iteration, it is higher at the current iteration as well, hence removing the need to recompute the marginal gain of all candidates. PMIA [8] is a heuristic approach developed for the IC model, and it is based on the maximum influence in and out arborescence (MIIA and MIOA) of every node. This is the union of all maximum influence paths that end up or start from that node. The key part of the algorithm is that paths with probability under a certain threshold are removed, assuming that influence is mostly local. SIMPATH [9] is also based on the path-pruning idea, but for the LT, computing the influence spread of a node by summing the probabilities of paths that start from it. Although providing a substantial speedup, those heuristic methods do not retain theoretical guarantees, in contrast to sketch-based algorithms. The idea of sketch-based approaches is to create several instances of the network that represent varying outcomes of the edge probabilities beforehand and use them to estimate influence spread of a seed set. SKIM is an example of sketch-based IM[12], which alleviates the need for Monte Carlo simulations, achieving extreme acceleration with $(1 - \frac{1}{e} - \varepsilon)$ approximate guarantees, where ε is a trade-off between accuracy and efficiency. An alternative and faster sketch based methodology with the same theoretical guarantees is based on Reverse Reachable (RR) sets [13]. An RR set of a node consists of other nodes that can influence it. After generating a sufficient number of RR sets for random nodes, the optimum seed set can be derived by selecting the nodes that cover their majority. The intuition is that the frequency of a node's appearance in the RR sets is analogous to its influence.

While the previous algorithms work with a static network, there's a growing literature dedicated to IM based on diffusion cascades. A diffusion cascade is a series of events that takes place over the network and indicates how information spreads in it, e.g. a tweet and its list of retweets. The first model that utilized real diffusion cascades for IM, attempted to learn the transmission probabilities between nodes in IC [14]. The model uses survival analysis to express the probability of a node getting influenced in the course of a cascade, and it is solved using an EM algorithm. The same group proposed learning an extension of IC in continuous time (CTIC) [4]. CTIC extends traditional IC, by defining the diffusion probability between two nodes analogous to the number of times one node was influenced by the other, as well as the time it took for the latter to get influenced by the former. The intuition behind this is that the longer it takes for a node to copy an action the less likely it is to copy it. NETRATE [7] is a seminal algorithm that steps on both aforementioned works. It is based on similar modeling as [14], but it learns the transmission delays between nodes. NETRATE can be used to infer how the nodes of the cascades are connected when the underlying network is not available. Subsequently, time-constrained IM can run on that inferred network, to give an estimate of the most influential users [15, 10], solely based on the cascades. A similar line of work, but with a different purpose, is diffusion cascade learning [16]. These machine learning models use cascades to predict whether a node will get infected or not [21], or the size of the cascade [22] when a cascade has already started. In the intersection of the two aforementioned approaches, lie methods that use both, the follower network and the diffusion cascades. A characteristic example is the credit distribution model [2]. Whenever a node u copies a node v in the diffusion cascades, credits are given to v and to the nodes that v copied. The influence spread of a seed set is the total influence credit of its seeds and is estimated efficiently, by alternating between credit estimations from the action logs and CELF. A simpler approach is to weigh the edges of the graph analogously to the activity of the nodes, e.g. how many times v has copied u in the cascades [5]. Subsequently, an IM algorithm can run on this weighted network.

All aforementioned approaches, though differing methodologically, address the same problem. However, their evaluation methods are quite deviant. IM algorithms on static networks are evaluated based on their estimated spreading and time efficiency. These methods totally overlook the real spreading dynamics of the network, as they focus on the problem from a more algorithmic than data-driven perspective. In some cases, epidemic simulations like SIR and SIS are utilized to give an estimate of a seed set's spreading in the network [23]. However, these models suffer from oversimplifying assumptions [18] and overlook several important characteristics of real diffusion cascades [19]. Moreover, their spreading estimate has proved inaccurate compared to actual diffusions that take place over the network [17]. Hence, although epidemic models might be a valid choice in the absence of diffusion data, in our case, we can form a more realistic ground truth based on the diffusions. Even in this case, however, evaluation is not straightforward. An erroneous example is representing the spread of a seed set in the test set by the sum of the average size of each seed's test cascades [25]. This is inherently problematic because large cascades

from individual seeds do not guarantee a large combined spread. A similar fault occurs when evaluating a seed set based on each individual seed's follows, mentions, retweets, and tweets [26]. Instead, our evaluation tactic is based on the number of distinct nodes influenced in the test set, by the seed set [10]. Although not devoid of assumptions, it is the closest and most objective measure of a seed set's influence over a network at a given time span.

3 Influence Maximization Analysis

In this section, we describe the analytical framework we followed to apply different approaches of IM on the same dataset, which is comprised of a temporal network and diffusion cascades. As mentioned above, we split the dataset in the train and test set. The train set corresponds to the diffusion cascades and the follow relationships that took place during the first 25 days, as well as the initial follower graph which is formed before the first day of crawling. The test set consists of the last week's diffusion cascades. In the train set, we utilize four different IM techniques to derive seed sets for comparison with the seed set derived by the proposed DIFFUGREEDY (described in Sec. 4). A general overview of the methodology can be seen in Figure 1. Below we analyze each technique and how we applied it.

3.1 Ranking by K-core decomposition

K-core decomposition has proven a strong reliable predictor of influence in previous studies [20, 23]. The K-core of a network is the maximal subgraph such that each vertex has at least K degree. As a proxy for IM, we can rank the nodes based on the maximum K-core they belong to, i.e. their coreness, and take the top as a seed set.

3.2 Influence Maximization via Martingales

As a representative to classic IM approaches, we use IMM, an algorithm based on the aforementioned RR sets, to derive the seed set from the follower network. This is a network of more than 95 million edges, so efficiency is of utmost importance. The most important advantage of IMM is that, in contrast with the rest of RR -based algorithms, it derives RR sets that depend on one another. As a result, the number of RR sets is diminished dramatically. In addition, it achieves a theoretical guarantee of $((1-1/e)/(1+\varepsilon)^2)$ using martingale analysis. The network is weighted using weighted cascade [1] and the parameter ε, which governs the trade-off between speed and accuracy, is set to 0.1.

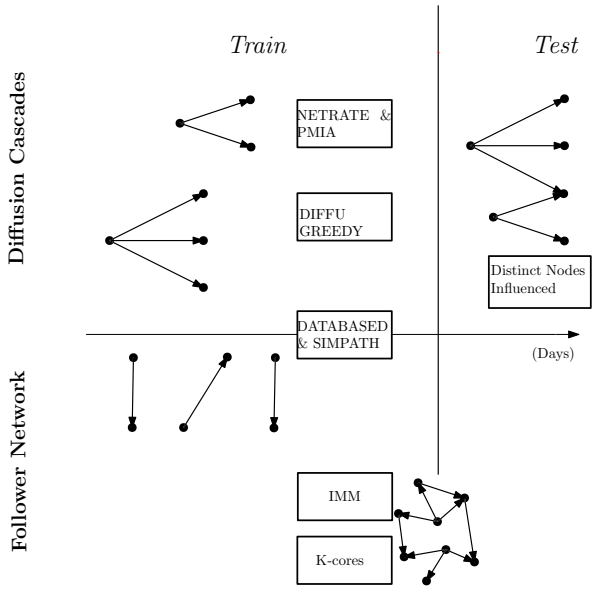

Fig. 1 An overview of the models we employed and which data they utilize. The upper quadrants correspond to diffusion cascades and the lower to the follow relationships established in time. The left quadrants belong to the train set and the right to the test set. Each network represents a different type of data. The static follower network is formed at the last day of the training set, so it is under the vertical axis. Each rectangle represents a different method, and its position indicates which type of data it is based on.

3.3 PMIA on the Diffusion-based network

To perform IM based exclusively on the diffusion cascades, we employ NETRATE [7] to infer the transmission rates between users in the train set. This can, in turn, define a new network with edge weights proportional to the inferred rates. The size of the inferred network needs to be very limited to satisfy the computational demands of NETRATE. Hence we follow the literature [25] and filter the cascades to keep only the most important nodes. In our experiments, degree proved to produce the most effective diffusion network. We filter all the cascades to remove nodes, either starting or participating in the cascade, that do not belong to the top 3000 nodes. Having computed the network and its transmission rates, we tried to use InfluMax, which is the archetype algorithm for continuous-time IM. However, NETRATE's inferred transmission rates that exceeded 10^{-6} were a mere 50 out of the 37937 inferred edges. Instead, we used PMIA [8] on the diffusion network with weights defined by weighted cascade and pruning parameter $\theta = 1/320$. Although inferior to InfluMax, it has served as a method of comparison [15] and provides a more then descent approximation to SIMUGREEDY. Moreover, PMIA is based on IC, which is closer to NETRATE's continuous-time IC then LT. Finally, since the diffusion network is small the computational requirements were minuscule.

3.4 SIMPATH *on the Data-based weighted network*

To combine the information of the temporal network with the diffusion cascades, we employed an edge weighting technique [5], which belongs to the data-based approaches [2]. We assume that a node u copies a node v, whenever u appears after v in a diffusion cascade and the time that u started following v is before the cascade's initiation. The edge weight is defined as:

$$E_{v,u} = \frac{A_{v2u}}{A_v} \times e^{-\frac{\bar{D}t_{v,u}}{\delta}}, \tag{1}$$

where A_{v2u} is the number of times u copied v, A_v is the total number of tweets and retweets of v, and $\bar{D}t_{v,u}$ is the average time that takes for u to copy v.
The first term captures the relationship's strength while the second is analogous to its speed and depicts the exponential decay of influence in time [4, 5]. The parameter δ facilitates containing the second term over 0 and is set empirically to 1000. The resulting network is in the scale of 1 million edges, because their overwhelming majority had zero weight. We perform IM using SIMPATH with pruning parameter $\eta = 0.01$.

4 The DIFFUGREEDY Algorithm

In this section, we propose a new influence maximization algorithm that utilizes the diffusion cascades in the train set to extract a seed set. The basic idea is to use the standard SIMUGREEDY algorithm [1], but substitute the candidate seed's influence spread estimation, with a summary of the seed's diffusion cascades. The algorithm can iteratively build an influence spread network, using the most suitable seed in each iteration and its final size represents the cumulative influence spread of the seed set. The main difference with SIMUGREEDY lies in the calculation of a seed's influence spread. In the original algorithm, it is computed using Monte Carlo simulations of a diffusion process that starts from that seed. In our case, we substitute this with an estimate from the list of diffusion cascades that the node has initiated in the train set, for brevity's sake the node's train cascades. We define a node's influence spread as the node's train cascade that provides the highest marginal gain. Initially we experimented by using the train cascade with the median marginal gain,as a more objective estimate, or the number of distinct nodes in the node's train cascades. These approaches performed worse in our experiments, hence we kept the definition based on the cascade with the highest marginal gain. If our definition of influence spread is submodular, we can retain the $(1 - 1/e)$ theoretical guarantee [1]. Below we provide the proof of submodularity and the algorithm.

Theorem 1. *Computing the influence spread of a candidate seed using the diffusion cascade that maximizes the set's marginal gain, is a submodular function.*

Algorithm 1 FIND THE SEED'S CASCADE WITH THE MAXIMUM MARGINAL GAIN

```
    procedure MARGINALGAIN(final_spread,seed_cascades)
2:      set max_gain ← −1, casc_idx ← −1
        for casc ← 0; casc < size(seed_cascades); casc++ do
4:          marginal_gain ← size(final_spread ∪ seed_cascades[casc]))
            if marginal_gain > max_gain then
6:              max_gain ← marginal_gain
                casc_idx ← casc
8:      return max_gain, casc_idx
```

Algorithm 2 INFLUENCE MAXIMIZATION USING NODES' DIFFUSION CASCADES

```
    procedure DIFFUGREEDY(node_cascades,seed_set_size)
2:  seed_sed ← [], final_spread ← ∅
        while size(seed_set) < seed_set_size do
4:          set max_seed = −1, max_gain = 0, max_cascade = −1
            for seed = 0; seed < size(node_cascades); seed++ do
6:              marginal_gain, cascade_idx = MarginalGain(final_spread,node_cascades[seed])
                if marginal_gain > max_gain then
8:                  max_gain ← marginal_gain
                    max_seed ← seed
10:                 max_cascade ← cascade_idx
            final_spread ← final_spread ∪ node_cascades[max_seed][max_cascade])
12:         seed_sed.insert(max_seed)
            delete node_cascades[max_seed]
14:     return size(influence_spread)
```

Proof. The influence spread of a node u at step t is represented by its train diffusion cascade with the highest marginal gain at that step, c_u^t. If another node v is added to the seed set at step t, the influence spread of c_u^t at $t+1$ can only be diminished, due to overlaps with v's influence spread. If c_u^t has a high overlap with v, then another one of u's cascades will be used, let c_u^{t+1}, in order to maximize marginal gain at step $t+1$. Since we always choose the cascade with the maximal marginal gain, c_u^t had larger marginal gain than c_u^{t+1} at step t. In addition, by definition, a cascade's marginal gain can only diminish or stay the same as the seed set grows. Thus, c_u^{t+1} at $t+1$ will always have smaller marginal gain then what c_u^t had at t, whether it is the same cascade or not, which is to be shown.

The complexity of DIFFUGREEDY is $O(KVC)$, where K is the size of the seed set, V is the number of nodes that initiated a cascade and C is the average size of cascades. Since the marginal gain estimation is submodular, we can utilize CELF, which does not change the worst case complexity, but has proven to accelerate greedy [11]. We do not add the Diffusion CELF here due to space limitations, but its derivation is similar to the way CELF is derived from SIMUGREEDY [11]. It should be noted here, that although SIMUGREEDY and CELF estimate the same marginal gains, the final seed set may differ, because of multiple nodes having the same gain and each algorithm choosing based on different seed orders. This results in Diffusion CELF having a slightly inferior performance in our experiments.

5 Experiments

5.1 Data

We apply our methodology in the Sina Weibo dataset [27], a network consisting of more than 1.7 million nodes and 0.4 billion edges, accompanied by a set of 300,000 retweet cascades. The actual expanding follower network is given for a time span of 32 days (2012.9.28 to 2012.10.29), throughout which, almost 10 million new follow relationships occurred. The diffusion cascades are gathered by the most popular of the past 1,000 tweets of each node in the network, and they date back since the year 2009. Since our methodology relies on the intersection of the follower network and the retweet cascades, we cannot utilize the cascades before 2012.9.28, as we do not know the structure of the follower network at that time. Concurrently, we can not use the nodes in the network that are not present in the retweet cascades, because we have no information about their interactions. We thus extract the diffusion cascades of those 32 days and remove nodes from the network that are not present in these cascades. That results in a network of 641,575 nodes, with 95,272,167 edges and 18,652 cascades. We split the cascades into training (14,555) and testing (4,097).

5.2 Results

As mentioned above, our evaluation method is based on the number of distinct nodes influenced (DNI) by the seed set in the test set. We consider influenced, every node that participates in a test set diffusion initiated from one of the predicted seeds. Since we measure the size of the distinct set, potential overlaps between diffusions of different seeds are taken into account. The DNI of each method are shown in Figure 2 and Table 1 shows the average DNI of each method throughout all seed set sizes. DIFFUGREEDY and DIFFUCELF clearly outperform the other approaches by a considerable gap. In addition, DIFFUCELF takes only 16 seconds, which is almost 40 times faster then k-core decomposition, and 1000 times faster then DIFFUGREEDY.

One important observation is the failure of the classic IM approach. Although IMM derived the seed set fast, its quality seems questionable. Only 15 out of the 100 seeds selected had started at least one cascade in the test set, and their spread was scant. This can be attributed to a lot of follow relationships not translating into retweets. This phenomenon is well-known [28] and plays a vital role in social influence analysis. Regarding the data-based weighted network, its failure might stem from the data itself. More specifically, we observed that the follower networks of retweet cascades are extremely sparse i.e. the ratio between the number of edges and the number of nodes is 0.84. This happens because, during crawling, the dataset's authors chose 100 users at random and crawled their follower ego-network up to three hops. Subsequently, they kept the last 1000 tweets of each user, each one containing a list of retweets. This list of retweets is filtered to contain only nodes that

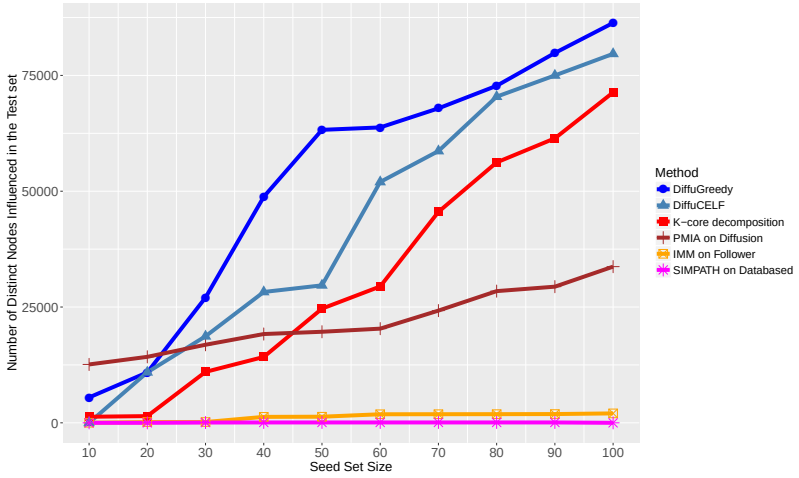

Fig. 2 Number of distinct nodes influenced (DNI) in the test set by the seed set of each method. Method labels are ordered based on their average DNI.

Table 1 Average evaluation metrics and computational time for each method.

Method	DNI	Computational Time (sec)
DIFFUGREEDY	52,600	16,504
DIFFUCELF	42,325	16
K-cores	31,657	632
NETRATE PMIA	21,863	27,966[a]
IMM	1,248	104,078[b]
DATABASED SIMPATH	56	96,908[c]

[a] Preprocessing took 999, NETRATE 26398 and PMIA 569
[b] Extraction and weighing took 103898 and IMM 180
[c] Extraction took 2981, weighing 93898 and SIMPATH 29

are in the crawled set. However, the follow relationships that indicates how the tweet reached a node that retweets can be lost somewhere after the three-hops of the original poster. As a result, the retweet cascades are comprised of mostly unconnected users. Therefore, weighing the follow relationships based on the diffusion cascades results in very few edges being retained, and most of the actual crucial follows being overlooked. Finally, the lack of success of the diffusion network approach could be attributed to the substantial mismatch with the actual follower network. Less than 1% of the inferred edges were actual follow edges. Even though as mentioned above, there were a lot of follow relationships in the diffusion cascades missing, it is safe to assume that a large part of the diffusion network was not based on direct follow edges, but rather on higher order relationships. These relationships although useful, are not as stable as direct ones i.e. an active follower is more likely to retweet than a follower's active follower. The retweets in the test set might consist of mostly fol-

lowers, which caused this approach's deficiency. The code, along with instructions to reproduce the experiments can be found on github[1].

6 Conclusion

As network science drifts towards data-driven approaches and increasingly more networks are accompanied by diffusion cascades, we have to reconsider our view of many important problems. In this study, we address influence maximization on a large scale social network. We employ multiple state-of-the-art methods, each exploiting a different aspect of the dataset, and propose an algorithm that outperforms them. In addition, we utilize an evaluation methodology based on actual diffusion cascades, as a more realistic alternative to epidemic simulation models. For future work, we plan to examine methods based on machine learning to derive the seed set. More specifically, while numerous neural network algorithms have been developed recently for influence or outbreak prediction [21, 22], the problem of influence maximization remains unaddressed. This is a promising path that we would like to explore further in subsequent steps.

References

1. Kempe, D., Kleinberg, J., Tardos, É. (2003, August). Maximizing the spread of influence through a social network. In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 137-146). ACM.
2. Goyal, A., Bonchi, F., Lakshmanan, L. V. (2011). A data-based approach to social influence maximization. Proceedings of the VLDB Endowment, 5(1), 73-84.
3. Vespignani, A. (2012). Modelling dynamical processes in complex socio-technical systems. Nature physics, 8(1), 32.
4. Saito, K., Kimura, M., Ohara, K., Motoda, H. (2009, November). Learning continuous-time information diffusion model for social behavioral data analysis. In Asian Conference on Machine Learning (pp. 322-337). Springer, Berlin, Heidelberg.
5. Goyal, A., Bonchi, F., Lakshmanan, L. V. (2010, February). Learning influence probabilities in social networks. In Proceedings of the third ACM international conference on Web search and data mining (pp. 241-250). ACM.
6. Tang, Y., Shi, Y., Xiao, X. (2015, May). Influence maximization in near-linear time: A martingale approach. In Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data (pp. 1539-1554). ACM.
7. Rodriguez, M. G., Balduzzi, D., Schölkopf, B. (2011). Uncovering the temporal dynamics of diffusion networks. arXiv preprint arXiv:1105.0697.
8. Chen, W., Wang, C., Wang, Y. (2010, July). Scalable influence maximization for prevalent viral marketing in large-scale social networks. In Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 1029-1038). ACM.
9. Goyal, A., Lu, W., Lakshmanan, L. V. (2011, December). Simpath: An efficient algorithm for influence maximization under the linear threshold model. In Data Mining (ICDM), 2011 IEEE 11th International Conference on (pp. 211-220). IEEE.

[1] https://github.com/GiorgosPanagopoulos/DiffuGreedy-Influence-Maximization

10. Du, N., Song, L., Rodriguez, M. G., Zha, H. (2013). Scalable influence estimation in continuous-time diffusion networks. In Advances in neural information processing systems (pp. 3147-3155).
11. Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J., Glance, N. (2007, August). Cost-effective outbreak detection in networks. In Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 420-429). ACM.
12. Cohen, E., Delling, D., Pajor, T., Werneck, R. F. (2014, November). Sketch-based influence maximization and computation: Scaling up with guarantees. In Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (pp. 629-638). ACM.
13. Tang, Y., Xiao, X., Shi, Y. (2014, June). Influence maximization: Near-optimal time complexity meets practical efficiency. In Proceedings of the 2014 ACM SIGMOD international conference on Management of data (pp. 75-86). ACM.
14. Saito, K., Nakano, R., Kimura, M. (2008, September). Prediction of information diffusion probabilities for independent cascade model. In International Conference on Knowledge-Based and Intelligent Information and Engineering Systems (pp. 67-75). Springer, Berlin, Heidelberg.
15. Rodriguez, M. G., Schölkopf, B. (2012). Influence maximization in continuous time diffusion networks. arXiv preprint arXiv:1205.1682.
16. Bourigault, S., Lamprier, S., Gallinari, P. (2016, February). Representation learning for information diffusion through social networks: an embedded cascade model. In Proceedings of the Ninth ACM International Conference on Web Search and Data Mining (pp. 573-582). ACM.
17. Rossi, M. E. G., Vazirgiannis, M. (2016, June). Exploring Network Centralities in Spreading Processes. In International Symposium on Web AlGorithms (iSWAG).
18. Pei, S., Morone, F., Makse, H. A. (2018). Theories for influencer identification in complex networks. In Complex Spreading Phenomena in Social Systems (pp. 125-148). Springer, Cham.
19. Gallos, L. K., Song, C., Makse, H. A. (2008). Scaling of degree correlations and its influence on diffusion in scale-free networks. Physical review letters, 100(24), 248701.
20. Kitsak, M., Gallos, L. K., Havlin, S., Liljeros, F., Muchnik, L., Stanley, H. E., Makse, H. A. (2010). Identification of influential spreaders in complex networks. Nat Phys, 6(11), 888–893.
21. Qiu, J. Z., Tang, J., Ma, H., Dong, Y. X., Wang, K. S., Tang, J. (2018). DeepInf: Modeling influence locality in large social networks. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'18).
22. Li, C., Ma, J., Guo, X., Mei, Q. (2017, April). DeepCas: An end-to-end predictor of information cascades. In Proceedings of the 26th International Conference on World Wide Web (pp. 577-586). International World Wide Web Conferences Steering Committee.
23. Malliaros, F. D., Rossi, M. E. G., Vazirgiannis, M. (2016). Locating influential nodes in complex networks. Scientific reports, 6, 19307.
24. Csardi, G., Nepusz, T. (2006). The igraph software package for complex network research. InterJournal, Complex Systems, 1695(5), 1-9.
25. Xie, M., Yang, Q., Wang, Q., Cong, G., De Melo, G. (2015, January). DynaDiffuse: A Dynamic Diffusion Model for Continuous Time Constrained Influence Maximization. In AAAI (pp. 346-352).
26. Jendoubi, S., Martin, A., Liétard, L., Hadji, H. B., Yaghlane, B. B. (2017). Two evidential data based models for influence maximization in twitter. Knowledge-Based Systems, 121, 58-70.
27. Zhang, J., Liu, B., Tang, J., Chen, T., Li, J. (2013, August). Social Influence Locality for Modeling Retweeting Behaviors. In IJCAI (Vol. 13, pp. 2761-2767).
28. Cha, M., Haddadi, H., Benevenuto, F., Gummadi, P. K. (2010). Measuring user influence in twitter: The million follower fallacy. Icwsm, 10(10-17), 30.