Influence Maximization on Twitter Accounts

Ikbal Singh Gurdev Singh Dhanjal
SJSU ID: 017418804
ikbalsinghgurdevsingh.dhanjal@sjsu.edu

Soham Choudhury
SJSU ID: 014181557
soham.choudhury@sjsu.edu

1. Introduction

Influence maximization was first introduced in data mining as a viral marketing problem. In recent years, there has been an increase in interest in influence maximization problem especially in social network analysis due to its real-valued applications in fields such as marketing, political campaigns, outbreak detection, etc. The boom of social networks has brought a lot of attention to *information diffusion* where a piece of information can quickly spread through a social network. A recent example of this is Donald Trump's 2016 election, where twitter was used on a daily basis for political campaign. In this project, we solve the influence maximization problem for twitter social network, where we figure out the optimal seed set and model the spread of influence using the seed set. We take two different approaches to the problem. In the first approach, we use the network structure to find the seed sets and model the spread, while in the second approach, we use an activity log of twitter users for a specific amount of time along with network structure to solve the problem. In Section 2, we go over the past work that has been done to solve the problem of influence maximization and give background about the problem including formally defining it with notations. In Section 3, we detail the algorithms used for influence maximization and models used for calculating diffusion spread. In Section 4, we start with the visualization of the network and then follow up with the results of both approaches and their visualization of spread, including metrics that measure the amount of spread. Section 5 concludes, with the challenges and future scope of this project. We use the *Twitter Higgs dataset** for this work. Our results achieve a spread that is close to the spread of the best approximation algorithm and it does that efficiently, making our solution scalable to large to moderately large networks. While we use machine learning techniques like Maximum likelihood estimation to use the Twitter activity to model the spread better, future works can focus on using deep learning to not only model the spread but also solve the influence maximization problem.

2. Background and Related Work

To understand the problem of influence maximization, it would be useful to divide the problem into two parts.

*https://snap.stanford.edu/data/higgs-twitter.html

- Influence Maximization problem
- Diffusion Modelling problem

2.1 Diffusion Modelling Problem

Let us consider a graph G(V, E, W) where V are nodes, E are the edges and W are the weights associated with the edges. A *seed* node set S of k nodes is a set of nodes in graph G that is the source of information diffusion. When modeling the diffusion, the information diffusion starts from these source nodes and then spreads to other nodes in the network. Diffusion modelling is a continuous process that is often discretized into a set of time steps T. The nodes that have the information at a time step t are called *activated* or *influenced* nodes. After running for a large number of time steps, the spread of the influence converges, which means that no new nodes can be activated. The maximum amount of activated nodes over a large amount of timesteps T given a seed set S is called the *spread* of the seed set S over a diffusion model M. There are two diffusion models that are popularly used for diffusion modeling.

2.1.1 Independent Cascade Model (ICM)

Independent Cascade Model (ICM) is a probabilistic model for modeling information diffusion i.e. contagion of information through a network. In ICM, we consider a user *v* that gets activated by each of its incoming neighbors independently by a probability that is often associated with the weight of the edge. This probability is called *influence probability* $p_{u,v}$, or *propagation probability.* Based on the probabilities and given a seed set S and starting with time step 0, the model iterates through discrete timesteps. Each node u in timestep t will activate each of its outgoing neighbors v that is inactive in timestep t-1 with probability $p_{u,v.}$ The stochastic activation process is generally simulated using a Monte Carlo simulation. Once a node u has been activated, it stays activated. The *spread* $\sigma_{G,M}(S)$ of seed set S is the expected number of activated nodes when S is the seed set and the above process converges over many timesteps [Li et al., 2018].

2.1.2 Linear Threshold Model (LTM)

Linear Threshold was introduced by Granovetter and Schelling [Li et al., 2018]. The basic idea of the linear threshold model is that a user can switch its status from inactive to active if a *threshold* number of its neighbors is active. In the LT model, each node has a random threshold value $\theta_v \sim U[0,1]$. A node v is influenced by each neighbor w according to a weight b_{vw} such that $\Sigma_{u \in N(v)} b_{u,v} \leq 1$ and it activates if $\Sigma_{u \in N(v)} b_{u,v} \geq \theta_v$ [Li et al., 2018]. LTM proceeds in discrete timesteps. It starts with seed set S as activated nodes in the first timestep and at time t, a node gets activated if the amount of influence from its neighbors $b_{u,v}$ increases the threshold of the node and then it remains active. The iterative process stops when there is no new user that gets activated. The *spread* $\sigma_{G,M}(S)$ of the seed set S under the LT model is the expected number

of nodes that will get activated after running the diffusion process before converging when no new nodes get activated.

2.2 Influence Maximization Problem (IM)

Influence maximization is a combinatorial optimization (CO) problem where the task is to find the optimal seed set S for which the expected spread is the maximum under a diffusion model M. It can be formally defined as the following.

Definition (Influence Maximization (IM)) [Li et al., 2018]**:-** Given a social graph G, a diffusion model M, and a positive integer k, IM selects a seed set S of k users from V as the seed set to maximize the influence spread $\sigma_{G,M}(S)$ i.e $\sigma_{G,M}(S) = \text{argmax}_{S \epsilon V \wedge |S| \leq k} \sigma_{G,M}(S)$.

Influence Maximization is an NP-hard problem and it can be proved using set-cover reduction. However, there is a greedy algorithm that gives a $1 - 1/e$ approximate solution and it can be shown that it achieves a spread not less than 63% of the optimal solution [Trivedi et al., 2020].

2.3 Past Work and evolution in the field of IM

Influence maximization was first explored by Richardson et al., 2006 in a viral marketing problem to select the best viral marketing strategy by the use of network structure and probabilistic models. Later, [Kempe et al., 2003] proved that the IM problem is an NP-hard problem and formulated the problem as an optimization problem under the ICM and LTM. They also proved the submodularity and monotonicity of the spread function and gave a greedy algorithm that provided the solution close to the optimal solution. In the greedy algorithm, we start with an empty seed set and then at each time step, we add the node which increases the influence spread to the maximum. So it uses the spread function and uses it as a heuristic for the selection of nodes for the seed set. This causes the algorithm to run the diffusion process an exponential number of times, which makes the algorithm not scalable for moderate to large graphs. Hence, the greedy algorithm is generally used to compare the results of other algorithms as a benchmark to evaluate new algorithms. The greedy algorithm is considered state-of-the-art. There has been significant research done to optimize the greedy algorithm even further. One of the widely used optimizations is the Cost Effective Lazy Forward (CELF) algorithm which exploits the sub-modularity property of the greedy algorithm to reduce the complexity of the algorithm. This was further optimized by Goyal et. al., 2011 as CELF++. That said, these algorithms still are inefficient for large graphs. The bottleneck here is the calculation of spread and the use of it as a heuristic. The efficiency issue can be tackled if we turn our way towards centrality heuristics. Alshahrani et al., 2018 propose an algorithm **PrKatz** which first pre-processes the network based on the degree of the nodes and then calculates the katz centrality of nodes and selects the seed nodes based on the katz centrality values. Chen et. al.,

2009 propose an algorithm based on degree centrality as a heuristic algorithm, yet they fail to provide a strong reason behind considering the active neighbors of the node. Trivedi et al., 2020 propose a degree heuristic algorithm named DHICM (Degree Heuristic for Independent Cascade Model) which uses a different degree discount and gives a solution that is close to the optimal (Greedy).

2.4 Past Work and Evolution in the Field of Influence/Diffusion Learning

In a diffusion model, we have edge weights that correspond to the probabilities that get used in the stochastic diffusion process. There are different ways to initialize these edge probabilities and there are model-specific methods to initialize the weights.LTM edge-weight assignment models include Uniform, Random, and Parallel Edges. The uniform model assigns edge-weight as $1/|In(v)|$ where $|In(v)|$ denotes the in-degree of node v. The Random model initializes the weights randomly and the parallel edges model assigns the weights to parallel edges in a multi-graph. On the other hand, ICM edge-weight assignment models include Constant, weighted cascade, and tri-valency models. In the Constant model, edge weights are constant probability values. Each edge has the same probability value. The most stable probability value is 0.01 whereas other values are possible depending on the network. Weighted Cascade edge-weight assignment is similar to the uniform model in LTM. Again weights are assigned as $1/|In(v)|$, where $|In(v)|$ denotes the in-degree of node v. This model states that each incoming neighbor of v exerts equal influence on v hence it is easier to influence nodes with fewer degree [18, 19].
In the Tri-valency Model, edge weights are randomly chosen from the set of probability [0.001, 0.01, 0.1].

Apart from the above methods which use the network structure to assign the edge-weights,. Goyal et. al., 2011 proposed different models and algorithms to learn the influence from an activity log. There are also a few temporal neural network methods proposed for learning the influence from the activity. However, the representation for these models is not inductive and cannot be adapted to different models, in a straightforward manner. We use the methods proposed by Goyal et. al., 2011 to learn the probabilities from an activity log.

3. Algorithms

In this section, we propose the algorithms that we use and the modifications that we perform on those algorithms to solve the IM problem for the Twitter dataset.

3.1 IM based on network structure

3.1.1 Degree Heuristic for Independent Cascade Model

Since Degree Centrality performs relatively well to the CELF algorithm, the motivation behind the developers of the Degree Heuristic for Independent Cascade Model (DHICM) algorithm is justified. In their algorithm, they utilized the degree centrality for every node in the network [Trivedi et al., 2020].

Algorithm 3 DHICM

Input: G as social network graph, k is the number of top influential nodes, model of diffusion M
Output: Set S of seed nodes

1: *initialize* : $S = \emptyset$
2: **for** every $v \in V$ **do**
3: compute the degree of v as d_v
4: $dd_v = d_v$
5: **end for**
6: **for** $i \leftarrow 1$ to k **do**
7: $u = argmax_v\{dd_v | v \in V \setminus S\}$
8: $S \cup \{u\}$
9: **for** each neighbour v of u and $v \in V \setminus S$ **do**
10: $dd_v = d_v - 1 - (d_u - 1)p$
11: **end for**
12: **end for**
13: Output S

However, they added a penalization technique for the immediate neighbors of a node once that particular node was selected to be in the seed set. Suppose a node u has just been added into the seed set. Let's call the n neighbors of u to be $\{v_1, v_2, \ldots, v_n\}$. Not only did they make sure to account for the fact that any neighbor v_i, with i being a real number greater than or equal to 1, can no longer influence u, but they also took into account the fact that the spread of u would remain part of the attributed "score" for v_i. Hence, they made sure to further subtract the expected influence of node u, as denoted by the term $(d_u - 1)p$, where p is the propagation probability put into use by the Independent Cascade Model (ICM). Fig. 1 shows that DHICM archives influence close to CELF, which is an optimization over state-of-the-art Greedy algorithm, in a significantly less time compared to CELF.

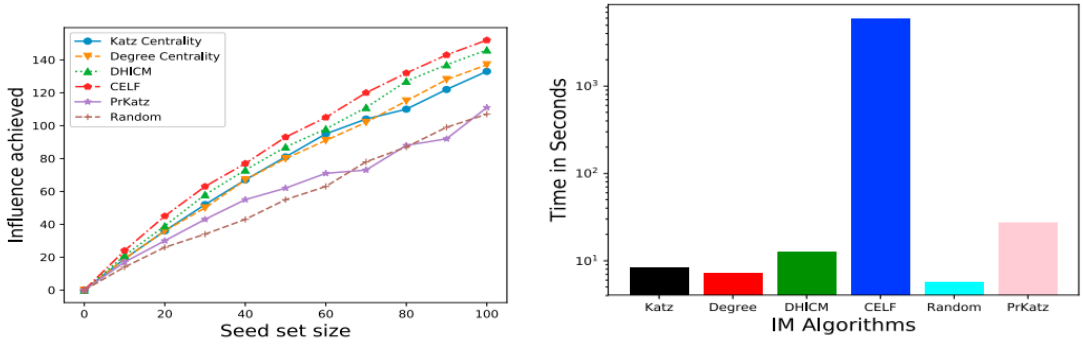

Fig. 1:- Comparison of running time and influence achieved by DHICM with other algorithms on NetHept database

3.1.2 Modified Independent Cascade Model

The value parametrized by p is, as mentioned before, the propagation probability. This value is obviously between 0 and 1 since it denotes the chance that one node (twitter user) successfully spreads a piece of information to another node. The standard is to use a scalable static value for p. We see the usage of p = 0.1 and p = 0.01 [Wang et al., 2012]. When p = 0.1, maximal influence, or very close to it, is achieved even with a small seed set. This means that as the seed set size increases, the improvement in spread reported by the ICM converges to 0. So, we try an even lower probability, p = 0.01; this leads to more stable results. Nevertheless, the spread achieved by this choice is minimal and does not align with real-world diffusion dynamics. To solve this, usage of a dynamic implementation for p is proposed and it allows us to define a more personalized relationship between two twitter users [Trivedi et al., 2020].

$$p_{ij} = .01 + \frac{d_i + d_j}{n} + \frac{CN(i, j)}{n}$$

The above equation defines p for the edge between nodes i and j. The first term, "0.01", simply defines the base propagation probability. "CN(i, j)" denotes the number of common neighbors of the nodes i and j, "$d_i + d_j$" denotes the combined degree centralities of the two nodes, and n is the number of nodes in the network. Examining the third term, "CN(i, j) / n", we can attribute this to judging the similarity between the two nodes because this directly compares the similarity of the crowd they interact with and normalizes it by dividing by the maximum possible number of common neighbors, which is, technically, n-1, since twitter users cannot follow themselves. However, on networks with a size of hundreds of thousands of nodes, a difference of 1 is negligible. The second term, "$(d_i + d_j) / n$" is included to symbolize the strength of the current edge; since our dataset is a directed graph, it assesses the nodes' influential power by assessing the number of other twitter users (i.e. their followers) they can possibly spread a piece of information to. However, the decision to use only n as the denominator, when that value could have a maximum of 2(n-1), was a puzzling attempt to normalize the numerator, $(d_i + d_j)$. Hence, we incorporated both the proposed version and a fix in our implementation of the Modified ICM.

Additionally, for our Retweet network, which is directed and weighted, we also experimented with multiplying p_{ij} by $weight_{ij}$ to suggest that a Twitter user who retweets an account m times is m times more likely to retain a piece of information sent out by the retweeted account.

It should also be noted that the Modified ICM takes longer to compute because of the expensive set arithmetic that is done in the third term, "CN(i, j)".

3.2 IM based on network structure and temporal activity

In this approach, we use a linear threshold model for model diffusion and learn the edge weights using a maximum likelihood estimation using the activity of action logs [Goyal et. al., 2011].
$T : E \rightarrow N$ is a function labeling each edge with the timestamp at which the social tie was created. We have an *action log*, a relation *Actions(User, Action, Time)*, which contains a tuble (u, a, t_u) indicating that a user u performed action a at time t_u.

The approach can be divided into the following steps.

1. Create an Action Propagation graph using the *Actions* and G(V,E,T) where each edge (u,v) indicates that both u and v perform the same action in order with a certain time interval t.
2. Calculate the propagation probability for each edge $p_{u,v} = A_{v2u} / A_v$ where A_{v2u} is the number of actions propagated from v to u in the propagation graph and A_v is the total number of actions performed by v in the *Actions*.
3. Use the Linear Threshold Model with a random threshold model with values ~ [0,1] and the propagation probabilities learned from the propagation graph.

4. Experiments

The dataset used for this project is the Snap Twitter Higgs dataset. The dataset contains 4 different graphs and an activity log for the tweets made for a time interval.

1) Friends/follower graph (directed): 450k nodes and 14 million edges
2) Graph of who retweets whom (directed and weighted): 250k nodes and 300k edges
3) Graph of who replies to who (directed and weighted): 38k nodes and 35k edges
4) Graph of who mentions whom (directed and weighted): 116k nodes and 150k edges
5) The dataset provides information about activity on Twitter during the discovery of Higgs boson

The experiments are performed on Google Collaboratory and we use Gephi along with matplotlib for visualization of the graphs. Fig. 2 shows the visualization of a sampled network of the Friends/Followers graph.

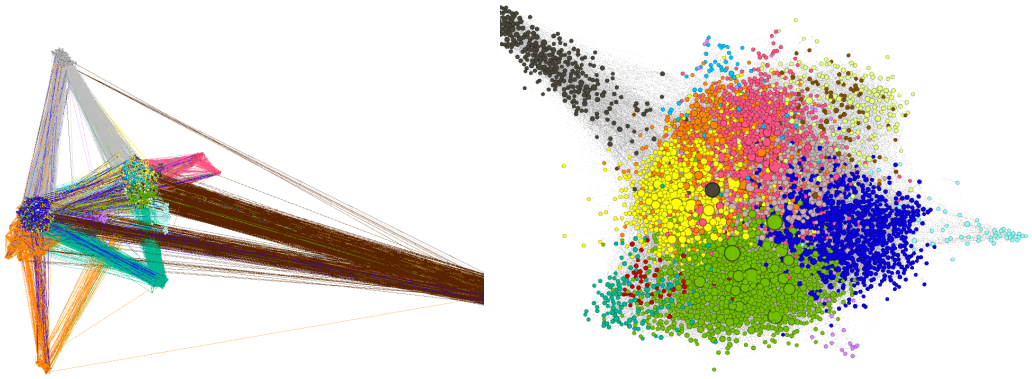

Fig. 2:- Visualization of Friends/Followers graph in Gephi for a sample of 40,000 nodes

4.1 Influence maximization based on network structure

For all the following experiments, we used 10 Monte Carlo simulations due to limited computing power on our machines.

Higgs Social Network

ICM (static p)

K (Seed Set size)	DHICM (p=0.01)	Degree Centrality (p=0.01)	DHICM (p=0.1)	Degree Centrality (p=0.1)
10	28,248.6	28,248.6	302,083.5	301,724.6
100	43,849.6	43,849.6	305,416.6	305,142.3

Higgs Retweet Network

Modified ICM (dynamic p) without accounting for edge weights

K (Seed Set size)	DHICM (p=0.1 + *H**)	Degree Centrality (p=0.1 + *H*)
1,000	22,897.4	22,885.5

Modified ICM (dynamic p) while accounting for edge weights

K (Seed Set size)	DHICM (p=0.01 + *H*)	Degree Centrality (p=0.01 + *H*)	DHICM (p=0.1 + *H*)	Degree Centrality (p=0.1 + *H*)
1,000	3,684.4	3,679.6	17,411.2	17,399.7

*: *H* represents the term $((d_i + d_j) / n) + (CN(i, j) / n)$

Fig. 3 shows the visualization of spread for a 40,000-nodes sampled network computed by DHICM for ICM with static p (a) and dynamic p (b).

(a) (b)

Fig. 3:- Visualization of spread for a 40,000-nodes sampled network computed by DHICM where (a) shows the spread in Higgs social network for static p=0.1 and (b) shows the spread in Higgs social network for dynamic p

4.2 Influence maximization based on activity log and network structure.

4.2.1 Preprocessing and action graph creation

We use the activity log to create the propagation graph where each edge u,v indicates that a user v performed an action (Retweet, Reply, Mention) influenced by u after a certain time interval t. Fig. 3 shows the action propagation graph creation for a 500-sized random sample of the activity.

4.2.2 Propagation probability calculation and LTM

After the propagation graph creation, we calculate the propagation probability using the formula $p_{u,v} = A_{v2u} / A_v$ where A_{v2u} is the number of actions propagated from v to u in the propagation graph and A_v is the total number of actions performed by v in the propagation graph. We use these probabilities to run a linear threshold model to calculate the spread. Fig. 5 shows the spread after we run a linear threshold model and visualization of 1000-sized nodes using the probability calculated from the propagation graph.

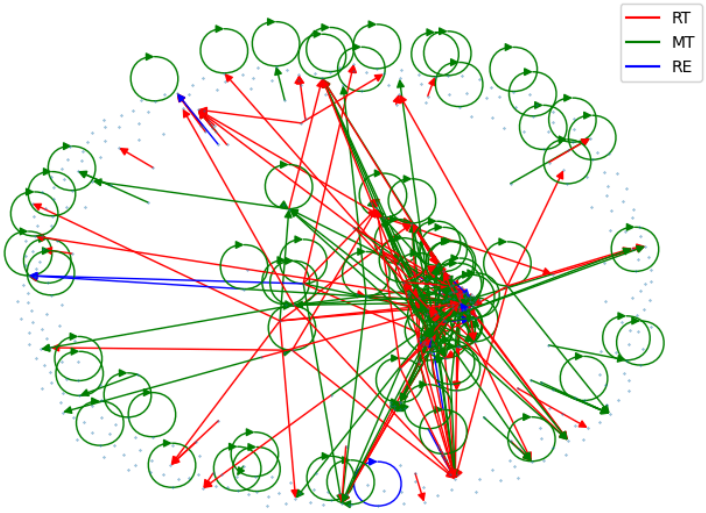

Fig. 3:- Propagation graph for a 500-sized sample network where red edges represent retweet propagation, green edges represent mention propagation and blue edges represent reply propagation

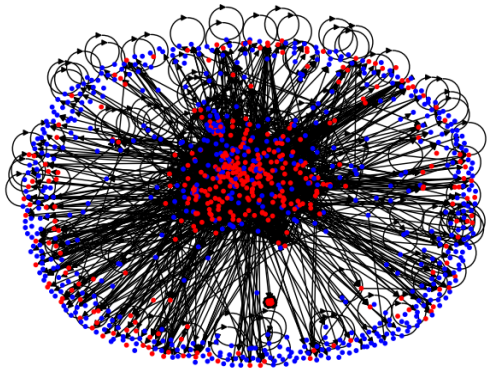

Fig. 5:- Spread visualization for the 1000-sized network *where blue-colored nodes represent non-activated nodes and red-colored nodes represent the activated nodes*

5. Conclusion

In this project, we use two approaches to the influence maximization problem on a Twitter social graph using the network structure and activity propagation graph. We use the DHICM algorithm for influence maximization and achieve a spread under the ICM and LTM close to the CELF algorithm. In future works, we can focus on some of the recurrent neural networks to learn the influence vectors and use them with deep reinforcement learning to solve the IM problem.

References

Alshahrani, M., Zhu, F., Zheng, L. et al. *Selection of top-K influential users based on radius-neighborhood degree, multi-hops distance and selection threshold*. J Big Data 5, 28 (2018). https://doi.org/10.1186/s40537-018-0137-4

Goyal, Amit et al. *CELF++: optimizing the greedy algorithm for influence maximization in social networks.* WWW '11: Proceedings of the 20th international conference companion on World wide web, https://doi.org/10.1145/1963192.1963217

Kempe, David, et al. *Maximizing the Spread of Influence through a Social Network*. Data Mining and Knowledge Discovery, 24 Aug. 2003, https://www.cs.cornell.edu/home/kleinber/kdd03-inf.pdf.

Li, Yuchen, et al. *Influence Maximization on Social Graphs: A Survey*. IEEE Transactions on Knowledge and Data Engineering, 1 Oct. 2018, https://ieeexplore.ieee.org/document/8295265.

Richardson, Matt, et. al. *Mining the network value of customers* KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, https://dl.acm.org/doi/10.1145/502512.502525

Trivedi, Nitesh, and Anurag Singh. "Efficient Influence Maximization in Social-Networks Under Independent Cascade Model." *Science Direct*, Procedia Computer Science, 2020, https://www.sciencedirect.com/science/article/pii/S1877050920315416?ref=pdf_download&fr=RR-2&rr=8329ac0cceb6967f.

Wang, Chi, et al. *Scalable Influence Maximization for Independent Cascade Model in Large-Scale Social Networks*. Data Mining and Knowledge Discovery, Nov. 2012, https://www.researchgate.net/publication/257553299_Scalable_influence_maximization_for_independent_cascade_model_in_large-scale_social_networks.

Wei Chen, Yajun Wang, and Siyu Yang. 2009. *Efficient influence maximization in social networks*. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '09). Association for Computing Machinery, New York, NY, USA, 199–208. https://doi.org/10.1145/1557019.1557047