Influence Maximization on Social Networks

Final Project Report

Xiao Feng
xiao.feng@student-cs.fr

Xinyu Hu
xinyu.hu@student-cs.fr

Vidhi Mahavir Jain
vidhimahavir.jain@student-cs.fr

Miao Wang
miao.wang@student-cs.fr

Abstract

Nowadays, more and more fields can be described as graph data. A pivotal idea in network science and marketing research is that a small group of nodes, called influencers, have the largest impact on social contagion and epidemic processes in networks. For viral marketing, with given budget, we hope to maximize the outreach by word-of-mouth effect and to realize it, we need to target initial 'seed' nodes with biggest impact. In our project, we chose Twitch gamers dataset and the task is to spread a product news with certain budget - the number of initial seed nodes, K. Seed set selection for Information Maximization Problem (IMP) is an NP-hard problem. We implemented centrality-based methods including K-core number, PageRank, ViralRank, and etc.,vanilla greedy algorithm and its variant, CELF algorithm. Meanwhile, we tried degree discount heuristics method. Evaluated by the impact (the simulated number of infected nodes) and time complexity(how many iterations it takes for converge), CELF significantly outperformed all state-of-art centrality measures in simulated impact with slightly more time complexity. In the end, we talked about further work we thought missed in our project.

1 Motivation

Social network influence is the process by which people receive information from others in order to disseminate it. [4] posed the question, influence maximization, that explored the technique of marketing to spread product publication on a large scale through limited contact with influential customers. Further, defined by [7], the influence maximization is the problem of finding the k most influential nodes in the social network that could maximize the number of nodes and the scale of spread. The influence and information would disseminate following certain patterns, termed as information diffusion models. Based on existing information diffusion models, we would like to further explore such optimization by comparing the performance of algorithms and possibly make improvements.

The problem is worth investigating since it has wide applications in the marketing domain, for example, viral marketing [4], rumor monitoring [1], etc. Many companies would be intrigued since the powerful word-of-mouth effect would provide innovative ideas which can effectively facilitate the communication process with lower costs. It could also contribute to the detection of outbreaks, for instance, detecting contaminants in the water distribution network [8], or closer to current affairs, discovering more contagious areas of COVID-19, and so on.

2 Problem Definition

The influence maximization can be further illustrated as follows [9]:

An undirected graph network G(V, E) has been given (V representing the set of nodes in the graph and E showing the set of edges or relationships between nodes). For a set A of k integral of initial nodes (seeds), satisfying $|A| = k$ and $A \subseteq V$, the k nodes follow an information diffusion model, M, influencing other nodes, and finally, the number of nodes influenced is optimized. The mathematical expression of the problem would be:

$$max\{\sigma(A), |A| = k, A \subseteq V\},$$

in which $\sigma(A)$ is the number of nodes that has been eventually influenced.

Under such models, Ms, we will implement algorithms that could best disseminate the influence. However, regarding the enormous social network provided by the internet, the graphs would be hugely scared and complexly structured. Therefore, the solution needs to be efficient and adaptive to large graphs [2].

3 Related Work

The problem of identifying nodes with good spreading properties in networks, can be further split in two subtopics: (i) identification of individual influential nodes and (ii) identification of a group of nodes that, by acting all together, are able to maximize the total spread of influence.To locate individual influencers by gauging the "influencing capability" of these users. At the individual level, a straightforward approach is to consider node centrality criteria and in particular the one of degree centrality. A series of related work were proposed, like PageRank, LeaderRank and ClusterRank. However, it was found that the k-core and k-truss decomposition and their extensions are more effective measures to capture the

spreading properties of nodes. Which we also plan to explore for our usecase.
Collective influence maximization is also formally called Influence Maximization. The Linear Threshold (LT) and the Independent Cascade (IC) model are widely approved and used to build the diffusion model. Influence Maximization has been identified as an NP-hard problem, so the majority of literature provided the approximation algorithms like greedy algorithm and heuristic way rather than exact solution which can prove to be very computationally intensive. Kempe et al. [7] proposed the earliest greedy algorithm, and a series of following algorithms were proposed to improve the computing speed by reducing the number of influence spread evaluations, designing scalable heuristics, etc. Leskovec et al. [8] proposed the CELF algorithm with 700 times faster than previous greedy algorithms. Goyal et al. [5] improved the CELF hat further optimized the aforementioned one by 35 to 55%. Chen et al. [3] proposed the LDAG algorithm for LT model and achieves to produce results by being orders of magnitude faster than the greedy algorithm.

4 Methodology

Under the topic of influence maximization, we focused on a social network where we are given the task to maximize the influenced nodes with a certain budget on how many nodes to touch on. The data set we are using is Twitch user-user networks, an undirected social network of online streamers. Nodes are the users themselves and the links are mutual friendships between them. Twitch is a streaming service where users can broadcast live streams of playing computer games and users can follow each other there. There are 1,912 nodes and 31,299 edges in our dataset. Apart from basic node and edge information, we are also given detailed attributes of each single node like 'days', how long a user has joined Twitch, and 'views', number of views on the channels, which can thus add weights to our graph. In this case, influencers are those individuals who have a loyal following of users, and they achieve a high level of engagement on their content.
Surrounding the main task of influence maximization, our pipeline was implemented as below:

4.1 Data pre-processing

Based on the magnitude of the selected dataset, calculating directly on the original dataset will result in large amounts of unnecessary computations while the pruned graph chips out the excessive calculation. Therefore, the first step is to prune the graph by removing low betweenness centrality nodes to keep denseness and save computation expenses.

4.2 Information diffusion model selection

Influence spread is simulated based on information diffusion models. In these models, various states are considered for nodes where each state may change based on some defined transition conditions. We implemented three well-known diffusion models, including Susceptible-Infected-Recovered (SIR) model, Independent Cascade(IC) Models and Linear Threshold(LT) Model, the implement details are shown below.

4.2.1 Susceptible-Infected-Recovered (SIR) Model. The classic susceptible-infected-recovered (SIR) model came as a disease spread model, in which all nodes in the network are in one of three states: susceptible (able to be infected), infected, or recovered (no longer able to infect or be infected). At each time step, only nodes infected in the last time step can infect any of its neighbors who are in a susceptible state with a probability. In SIR, parameter gamma() represents recovery rate that a contagious person has probability of becoming non-contagious while beta represents the effective contact rate of the disease: an infected individual comes into contact with N other individuals per unit time. Based on the specific background of our problem, the gamma is set as 0.1 while beta is set as the inverse of the eigenvalue of the adjacency matrix of the graph plus 0.2. Considering a network structure, a key problem relating to the SIR model is how to identify the nodes that, if initially infected, will result in the greatest expected infected population. Plot below shows how the number of susceptible,infected and recovered nodes involved along with iterations in SIR model.

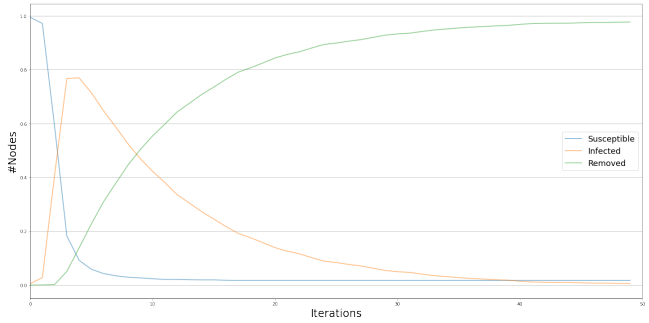

Figure 1. SIR-The progression of the number of susceptible, infected and the recovered nodes

4.2.2 Independent Cascade(IC) Models. Independent cascade (IC) model, which is a generalized of SIR model, is a well-known cascade-based progressive information diffusion model where nodes are considered active or inactive. The IC model begins with the assumption that all the nodes of a social network are inactive except the initial set of active nodes. Activation process for inactive nodes proceeds in discrete steps as follows: when the node v becomes active in

step t, it tries only once to activate each one of its inactive neighbor nodes based on an influence probability which is set as 0.5 in our model. If an inactive node has numerous newly activated neighbors who are trying to activate it, they try to do so in a random order. If v succeeds, then w would be active in step t+1. The activation process continues until no node can be activated. Plot below shows how the number of active and inactive nodes involved along with iterations.

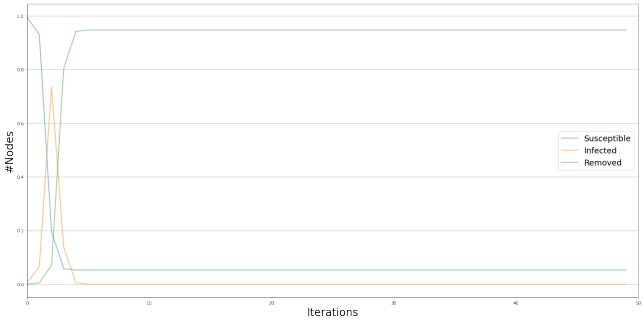

Figure 2. Independent Cascade-The progression of the number of susceptible, infected and the recovered nodes

4.2.3 Linear Threshold(LT) Model. Linear threshold (LT) is another well-known progressive information diffusion model as a threshold-based model, where nodes are also considered active or inactive. This model consists of: 1) the desire of each node v to be influenced by its neighbors presented by a threshold value in the range of 0 to 1, 2) the influence weight of each node w on its neighbor node v. After several trials, the threshold is set as 0.1 in our model. The LT model begins with the assumption that all the nodes of a social network are inactive except the initial set of active nodes. Activation process for inactive nodes proceeds in discrete steps as follows: in each step t, every inactive node v subject to its activation formula will become active and become added to the active nodes of previous steps. The activation process continues until no node can be activated. Plot below shows how the number of susceptible and removed nodes involved along with iterations.

4.2.4 Model Comparison. Here we use the speed of convergence of different propagation models, i.e., the number of iterations required, to initially select a simulation propagation model. It is obvious that the IC model converges the fastest, so we will use the IC model to simulate the propagation of nodes.

4.3 Centrality Measures

Centrality is one of the most widely used indicator based on network data. Before to maximize the influence, the choice

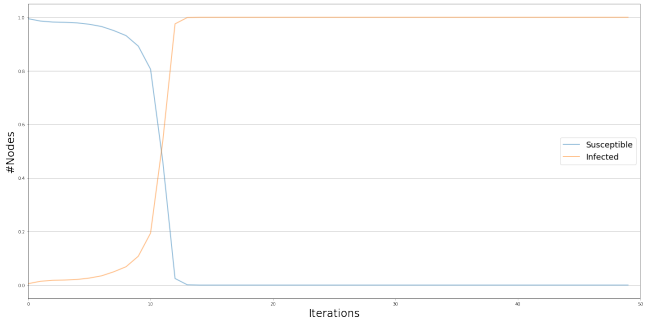

Figure 3. Linear Threshold-The progression of the number of susceptible, infected and the recovered nodes

of centrality is essential to first identify the influencers. Here, we implemented 7 methods to measure the centrality in order to detect as many influencers as possible. The following section describes in detail how the centrality measures are used.

4.3.1 Degree Centrality. Degree centrality is the simplest way to calculate centrality. Recall that the degree of a node is simply a calculation of how many social connections (i.e., edges) it has. A node's degree centrality is its degree.

4.3.2 PageRank Centrality. Before diving into PageRank centrality, we first introduce EigenCentrality. EigenCentrality measures a node's influence by measuring each nodes 'degree' score, while EigenCentrality goes a step further than degree centrality which assesses beyond the first-degree connections to count how many links their connections have, and so on through the network. PageRank centrality is a variant of EigenCentrality using hyperlinks between pages as a measure of importance. PageRank's main difference from EigenCentrality is that it accounts for link direction. Each node in a network is assigned a score based on its number of incoming links. These links are also weighted depending on the relative score of its originating node. The result is that nodes with many incoming links are influential, and nodes to which they are connected share some of that influence.

4.3.3 k-core Centrality. Based on k-core centrality, the centrality of a node is determined by decomposing a network into subcomponents consisting of nodes connected with a degree of at least k or lower, where degree is measured in terms of single ties.

4.3.4 Neighborhood Coreness. Neighborhood centrality is one of the hybrid approaches where the entire graph is first traversed, and the k-shell centrality values of nodes are determined. The influence factor of node v is then specified through considering the k-shell values of node v and its neighbors. Neighborhood coreness centrality is a hybrid approach that calculates that calculate the influence centrality as the accumulated sum of the k-shell values of node v.

4.3.5 Hypertext Induced Topic Selection.

The HITS algorithm measures the "hubness" and the "authority" of each node in the graph. "Hubness" of a node is a defined in terms of the authorities of the outgoing nodes and "authority" is defined in terms of the hubness of the incoming nodes. Therefore, a node has a high hubness value if it has many outgoing edges leading to nodes with high authority values and a node has a high authority value if the hubness values of its incoming nodes are high.

4.3.6 Viral Rank.

ViralRank is a centrality measure that identifies the most influential nodes in complex networks, defined in [6].
ViralRank is defined as:

$$v(\lambda) = -\sum_{j}(D_{ij} + D_{ji})/N$$

where D_{ij} is the random-walk effective distance from i to j and λ is a parameter that plays the role of the inverse temperature (see next). λ is defined as:

$$\lambda = log[(\beta - \mu)/\alpha] - \gamma$$

where β, μ and α are the infection, recovery and diffusion rates, respectively, while γ is the Euler constant. For contact networks its value is set equal to (approximately) zero corresponding to a high temperature expansion, while it can be specifically tuned for metapopulation networks using the above definition.

4.4 Locate individual influencers

In terms of measures for 'influencing capability', we explored the basic centrality measure methods and some variants. Apart from this, we also implement several popular and advanced techniques with higher robustness to locate individual influential nodes under the IC model, which are Pagerank, k-core, neighborhood coreness, HITS and authority methods. The number of required iterations to identify the infected nodes of various algorithms are compared and illustrated in the evaluation part. As a result, PageRank, viral rank and authority targeted the most influencers compared with the other measures.

4.5 Influence maximization

Influence maximization problem can be viewed as a general case of an NP-complete Set Cover problem under the IC model. Solving this problem turns out to be extremely computationally burdensome. Approximate solutions like greedy algorithms were proved to give the near optimal solution. We implemented both the simple greedy algorithm and the greedy algorithm with Cost Effective Lazy Propagation(CELF) to compare the computation efficiency and accuracy of these two approaches.

4.5.1 Greedy Algorithm.

Greedy Algorithm is to find the node with the biggest spread, adds it to the seed set and then finds the node with the next biggest marginal spread over and above the spread of the original and so on until seed nodes are found. The main steps of the greedy algorithm to maximize the influence are described as below.

Algorithm 2 Greedy Algorithm

```
1: procedure GreedyApproximation(V, k)
2:     Set 𝒳 = ∅
3:     while |𝒳| ≤ k do
4:         Pick s ∈ V where σ(𝒳 ∪ s) − σ(𝒳) is the greatest
5:         Add s to 𝒳
6:     end while
7:     return 𝒳
8: end procedure
```

Figure 4. Pseudo code for the Greedy algorithm

Though greedy algorithm has no guarantee to find the optimal solution but it is proved that the algorithm is theoretically guaranteed to choose a seed set whose spread will be at least 63% of the spread of the optimal seed set. No better theoretically proved approximation has been found yet.
In our model, the greedy() function does the same for all iterations of the outer for loop. Specifically, it computes the marginal spread of all remaining candidate nodes, and then selects the node with the largest spread. The calculation of the marginal spread of all nodes occurs in an inner for loop that iterates over the nodes in the rangelist by subtracting the nodes of the current seed set from all nodes g.vcount(). In each iteration, the marginal diffusion is computed by calling the IC() function with the input seed set equal to the union of S and the current node j. The marginal spread of j is then compared to the largest spread so far in the loop (best_spread), and if it is greater, the node becomes the current "leader".
The function not only returns the optimal seed set S but also the average spread of that seed set along with a list showing the cumulative time taken to complete each iteration.

4.5.2 Greedy Algorithm with CELF.

Cost Effective Lazy Propagation(CELF) exploits the sub-modularity property of the spread function, which implies that the marginal spread of a given node in one iteration of the Greedy algorithm cannot be any larger than its marginal spread in the previous iteration. This helps us to choose the nodes for which we evaluate the spread function in a more sophisticated manner, rather than simply evaluating the spread for all nodes. The main steps of the greedy algorithm with CELF to maximize the influence are described as below.

In our model, the implement of the algorithm using celf() function can be split into two components. The first component iterates over each node in the graph and selects the node with the highest spread into the seed set while it also stores the spreads of each node in the sorted list for use in

Algorithm 5 CELF

```
Input:  K size of seed set, f submodular function, graph G
Output:  Seed set of influencers
procedure CELF Seed Set Detection(G)
    Initialization: S ← φ, Priority Queue Q ← φ, iteration ← 1
    for vertex u in vertices do
        u.mg ← f(u|φ)
        u.iteration ← 1
        Insert vertex u in Q
    while iteration <= K do
        Delete top element v from Q
        if v.iteration equals iteration then
            S <- S ∪ {v}
            iteration <- iteration + 1
        else
            v.mg ← f(v|S)
            v.iteration ← iteration
            Insert v in Q
    return S
```

Figure 5. pseudo code for CELF algorithm

Algorithm 4 DegreeDiscountIC(G, k)

```
initialize S = ∅
for each vertex v do
    compute its degree d_v
    dd_v = d_v
    initialize t_v to 0
end for
for i = 1 to k do
    select u = arg max_v{dd_v | v ∈ V \ S}
    S = S ∪ {u}
    for each neighbor v of u and v ∈ V \ S do
        t_v = t_v + 1
        dd_v = d_v − 2t_v − (d_v − t_v)t_v p
    end for
end for
output S
```

Figure 6. Pseudo code for the Degree Discount Heuristic algorithm

the second component.
The second component iterates to find the remaining k-1 seed nodes. Within each iteration, the algorithm evaluates the marginal spread of the top node in the list and replaces it within the list. If the top node stays in place then that node is selected as the next seed node. If not, then the marginal spread of the new top node is evaluated and so on.
Like greedy(), the function returns the optimal seed set, the resulting spread and the time taken to compute each iteration. In addition, it also returns the list LOOKUPS, which keeps track of how many spread calculations were performed at each iteration. We didn't bother doing this for greedy() because we know the number of spread calculations in iteration is .

4.5.3 Degree Discount Heuristics.

Even with CELF, the running time will still be large and may not be suitable for large social network graphs. A possible alternative is to use heuristics. With the combination of degree centrality measure and greedy algorithm, degree discount heuristics is proposed, which nearly match the performance of the greedy algorithms for the IC model, while also improve upon the pure degree heuristics discussed above.

The general idea is that ,let v be a neighbor of vertex u. If u has been selecteed as a seed, then when considering selecting v as a new seed based on its degree, we should not count the edge $\overline{vu}$ towards its degree. Thus we should discount v's degree by one due to the presence of u in the seed set, and we do the same discount on v's degree for every neighbor of v that is already in the seed set.[2]

Using Fibonacci heap, the running time of Degree Discount Heuristics is $O(klogn + m)$. In theory, it should be much faster than vanilla greedy algorithm with $O(KNnm)$.

5 Evaluation

Apart from the comparative study of the algorithms to reach the best influencer for marketing reasons, we also plan to analyze the time complexity of each model as to understand the production deployability value and the attempt to achieve high score (performance guarantee) with computational cost trade-off. Thus evaluating accuracy and efficiency:

- The time of convergence (time complexity)
- The time of computation (computation burden)
- Number of infected nodes (influence spread)
- Locations of seed sets (core VS periphery)

6 Experiment Results

6.1 Information Diffusion Model Comparison

From the following chart, we discovered the superiority of the independent cascade model in terms of the least time it took to converge. The model is based on random seeds of 100 nodes from the data set.

Regarding SIR, we adopted the gamma (0.1) and obtained a beta of 0.21 based on eigenvector centrality (the one that outperforms other nodal measures). This means that the probability of infection is 0.21. Whereas within the independent cascade model, instead of a single probability infection, we assigned a probability of infection associated with each edge (threshold = 0.5). As for LTM, we first gave a threshold of 0.1. Node's individual decision depends on the active status of its neighbors and the percentage of which that have made the same choice, thus imposing a threshold. Therefore, the convergence speed at first was quite flat, and once the activeness was spread, it was soon elevated to a high level, finally affecting all the nodes given the high density of the graph.

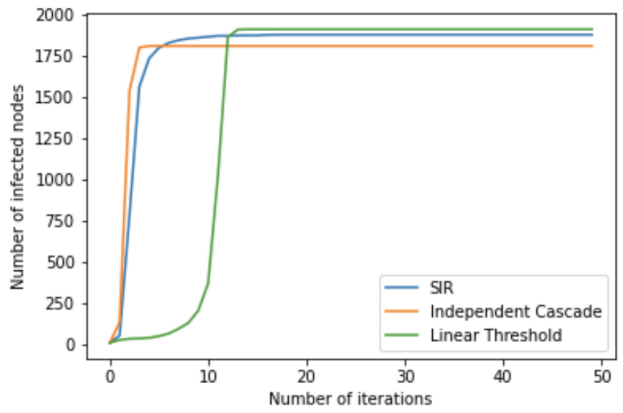

Figure 7. The number of Iterations of different diffusion models

6.2 The Effect of Seed Sets (Comparison of Centrality)

Since the IC model efficiently converged, we adopted IC as the diffusion model to investigate the effect of seed sets given different centrality measures. We selected top 10 nodes under the centrality as seed sets. And it can be observed that k-core and viral rank lay under the other measures despite they finally reached similar performance of the number of infected nodes.

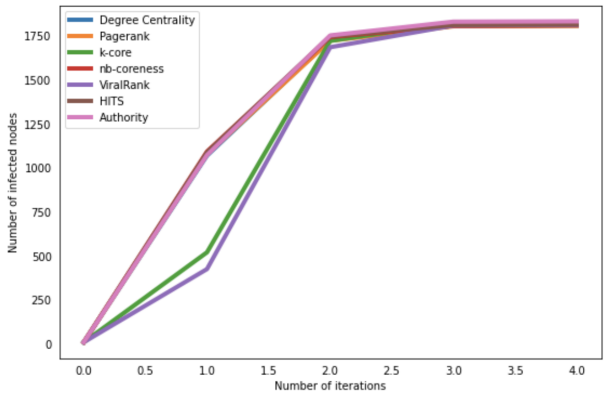

Figure 8. The number of infected nodes of different centrality measures under the IC model

Similar results have been yielded under the SIR and LT models. It also more clearly shows degree centrality may provide a better seed set for IC propagation.

To compare the measures, we visualized the identified seed sets based on the kcore, Viral Rank and degree centrality. The seed sets are all in dense and centralized clustering of the graph, whereas no overlap has been found between

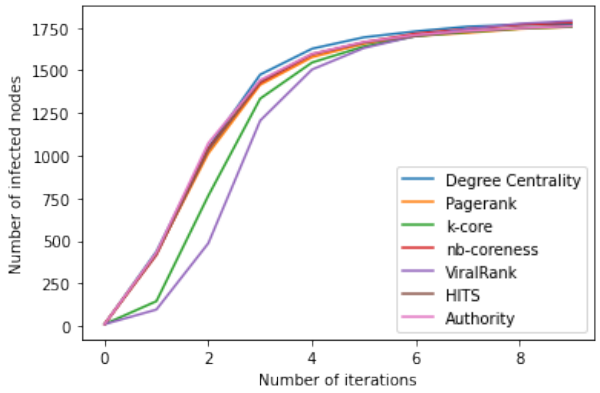

Figure 9. The number of infected nodes of different centrality measures under the SIR model

kcore or Viral Rank with the degree centrality (see attached visualizations in the appendix). Therefore, the degree centrality may give a better seed set for IC to diffuse the network.

6.3 Influence Maximization under the IC Model

Besides, we also implemented the greedy and CELF algorithm under the independent cascade model. We evaluated the algorithm in terms of both the time and the number of infected nodes. We explored the algorithm given IC threshold of 0.5, the size of seed set equal to 5 in five Monte-Carlo simulations. We found CELF as an improved version of the greedy algorithm, running 66.67% faster than the greedy algorithm and achieving slightly more infected nodes at last. Comparing the two seed sets, we found three overlapping nodes, just within 5 loops of simulations. Different from the nodes engendered from the centrality, the nodes may also in the edged parts of the graph.

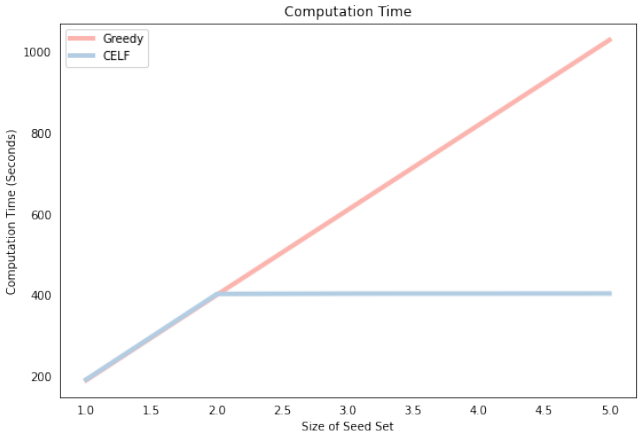

Figure 10. Computation Time (Greedy vs CELF)

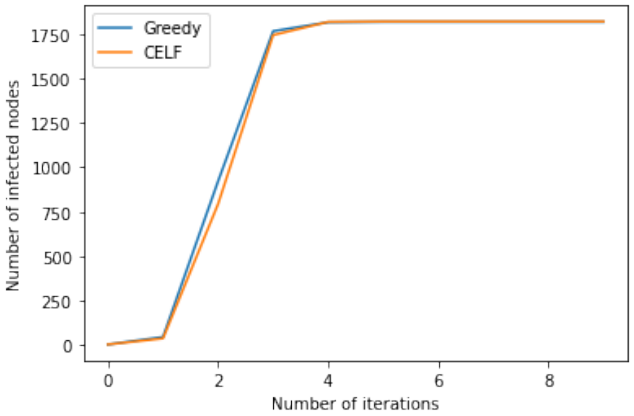

Figure 11. The number of infected nodes (Greedy vs CELF)

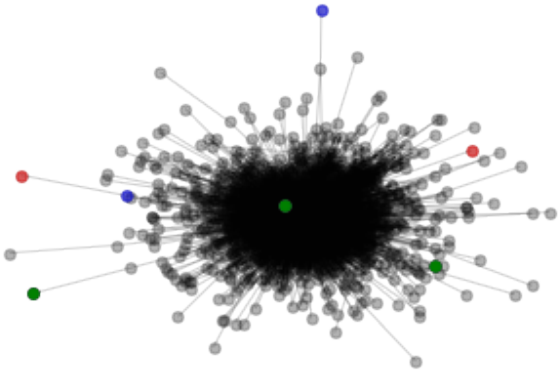

Figure 12. Blue Nodes: Greedy, Red Nodes: CELF, Green Nodes: Intersection

Due to the efficiency of CELF, we chose to employ it to locate seed sets in contrast to the centrality measures. It manifests that the seed set obtained with CELF could reach the largest number of nodes in most cases. Our newly introduced Viral Rank failed to play better than other centrality measures within 50 iterations.

Furthermore, the larger seed set size does not imply a larger range of nodes infected since growing trends have not been detected for each methods with an increasing seed set size. For flatter observations, such as degree and Page Rank, it is highly possible that the first top 10 to 20 nodes and nodes ranked just after those would have similar influence on other nodes.

We still visualized the seed set, comparing the authority measure and CELF at the seed set size of 10 with authority highest amongst all the centrality measures. No nodes have been detected in the intersection sets, which also reached to

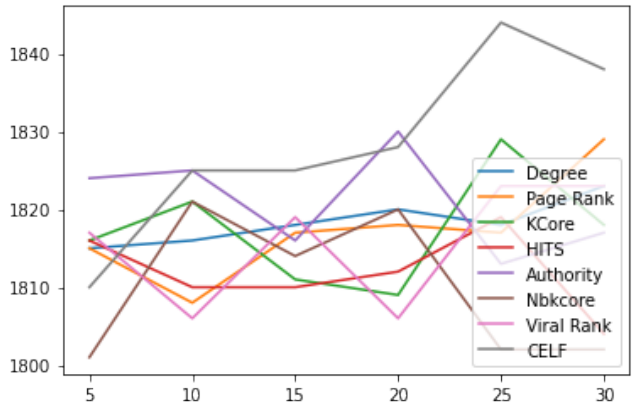

Figure 13. Number of infected nodes with different seed set sizes (CELF vs Centrality Measures)

the conclusion that a better seed set also includes the nodes at the edges in the less dense part of the graph, which could further a larger span of influenced nodes.

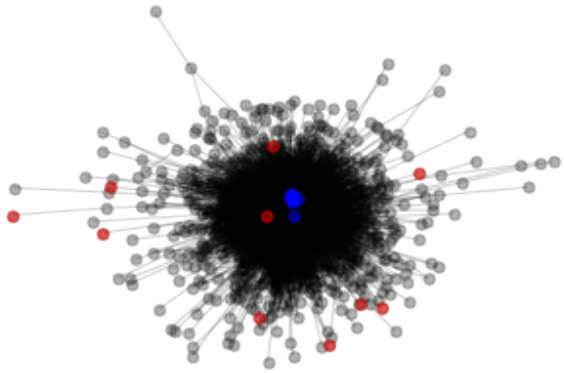

Figure 14. Blue Nodes: Authority, Red Nodes: CELF, Green Nodes: Intersection

6.4 Improving Centrality: Degree Discount Heuristics

Since many of the most central nodes may be clustered, targeting all of them is not at all necessary. We explored the degree discount heuristics in the dense graph and compared it with the algorithms under the IC simulation. Against simulation-based algorithms, the computation time took only 0.1 seconds compared to CELF and greedy algorithms. It also converged faster than these two derived from the following graph with slightly lower performance. Amazingly reducing the running time, it was with subtle degradation in performance.

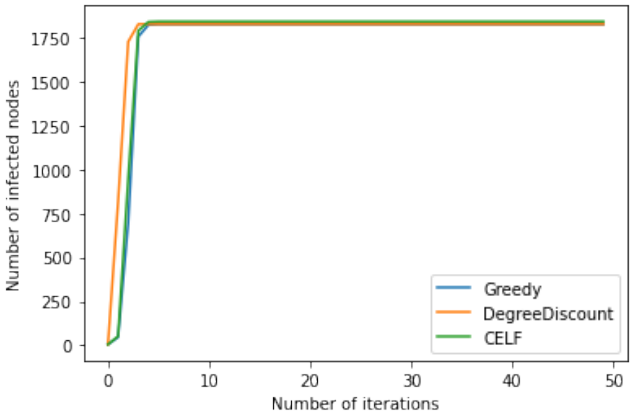

Figure 15. The number of infected nodes (Simulation-based Algorithms vs Degree Discount Heuristics)

Compared to conventional degree/centrality, it did not outperform other centrality measures, and thus would has a smaller number of infected nodes than CELF. However, this may be specific to this particular densely distributed graph.

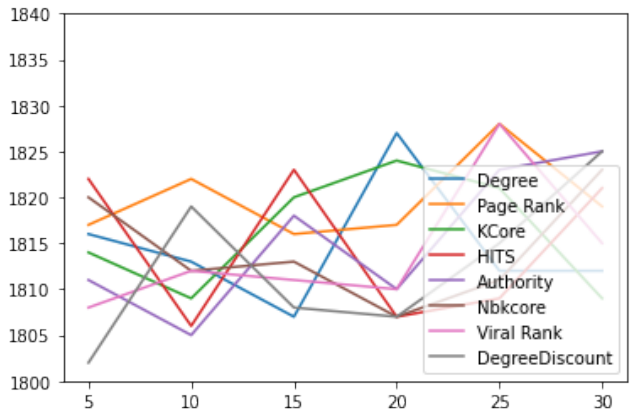

Figure 16. Number of infected nodes with different seed set sizes (Degree Discount Heuristics vs Centrality Measures)

7 Conclusion

In terms of diffusion models, we found that Independent Cascade model works best in terms of time complexity. For centrality measure, actually there still lacks agreement on which metric best quantifies the spreading ability of the nodes. Most proposed centrality measures are based on analytical arguments which are valid for certain types of networks and spreading parameters. So the choice of centrality measures depends on which type of networks we will be dealing with.

To summarize, our highlights of work are mainly twofold. First is on the centrality measurement. Apart from all state-of-art centrality measures, we implemented ViralRank-the first one that builds a centrality measure on analytic estimates of random-walk hitting times[6].

Second is on the selection method. The Influence Maximization problem is a NP-hard problem. The most prevalent method is Greedy algorithm, which is synonym to computation burden, and to tackle this, we implemented CELF('Cost-Effective Lazy Forward'), an optimized version of Greedy algorithm. The CELF optimization uses the submodularity property of the influence maximization objective to greatly reduce the number of evaluations on the influence spread of vertices [2].Our experiment results demonstrate that CELF optimization has almost the same influence spread as the original greedy algorithm but could achieve 66.67% speedup, which is a very impressive result.

One step further, instead of using Greedy algorithm, we experimented with degree discount heuristics.

Still there are several points that we can work further on. In our project, we didn't take node features into consideration but rather focus only on their connectivity, however the features like recency, seniority are informative indicators for gauging a gamer's activity, which should be decisive if we should shortlist a node. We can further develop our project by involving them and adopt UAC Rank algorithm.
Meanwhile, there are plenty of variants of greedy algorithm in this particular topic, we should be able to touch on few more others.
Furthermore, we only look at identifying individual influencers, while in real life, a collection of nodes with lower centrality score can propagate more collectively. Collective influence maximization can be another topic we dig deeper into.

A Appendices

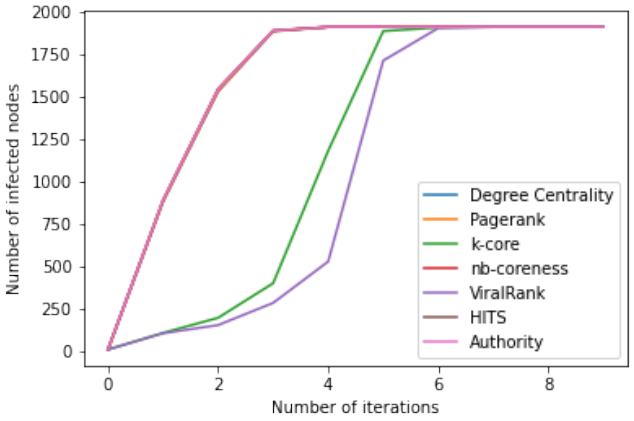

Figure 17. The number of infected nodes of different centrality measures under the LT model

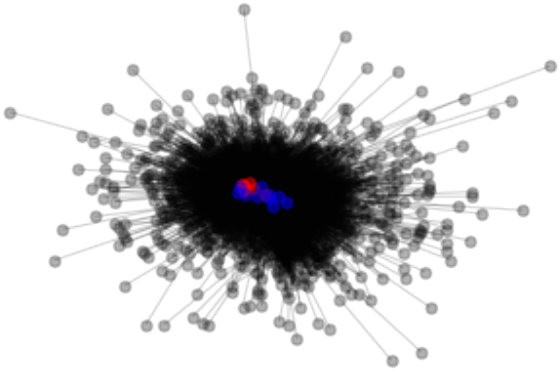

Figure 18. Blue Nodes: Kcore, Red Nodes: Degree Centrality, Green Nodes: Intersection

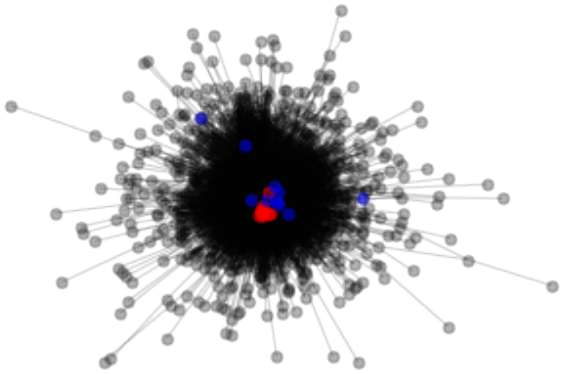

Figure 19. Blue Nodes: Viral Rank, Red Nodes: Degree Centrality, Green Nodes: Intersection

References

[1] Ceren Budak, Divyakant Agrawal, and Amr El Abbadi. 2011. Limiting the spread of misinformation in social networks. In *Proceedings of the 20th international conference on World wide web*. 665–674.

[2] 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*. 199–208.

[3] Wei Chen, Yifei Yuan, and Li Zhang. 2010. Scalable influence maximization in social networks under the linear threshold model. In *2010 IEEE international conference on data mining*. IEEE, 88–97.

[4] Pedro Domingos and Matt Richardson. 2001. Mining the network value of customers. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. 57–66.

[5] Amit Goyal, Wei Lu, and Laks VS Lakshmanan. 2011. Simpath: An efficient algorithm for influence maximization under the linear threshold model. In *2011 IEEE 11th international conference on data mining*. IEEE, 211–220.

[6] Flavio Iannelli, Manuel Sebastian Mariani, and Igor M. Sokolov. 2018. Influencers identification in complex networks through reaction-diffusion dynamics. *Physical Review E* 98 (12 2018), 062302. https://doi.org/10.1103/PhysRevE.98.062302

[7] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. 137–146.

[8] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen, and Natalie Glance. 2007. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. 420–429.

[9] Fei Wang, Zhenfang Zhu, Peiyu Liu, and Peipei Wang. 2019. Influence maximization in social network considering memory effect and social reinforcement effect. *Future internet* 11, 4 (2019), 95.