Estimating Network Parameters for Approximating the Influence Maximization Problem

Eyvind Niklasson
Cornell University
een7@cornell.edu

Michael Luo
Cornell University
mrl233@cornell.edu

Abstract

The spread of information or influence through a large social network can be reasonably thought of as a cascading process occurring on top of the network structure. As more individuals in the network become aware or accepting of the trend, they in turn affect whether others, too, join the movement. Much previous work has been done to model these phenomena as games, and we consider one of the most well-studied of these games, the influence maximization problem. This is the optimization problem of selecting the most influential size-restricted subset of the vertices, so as to maximize the reach of the resultant cascade.

Previous methods to solving the influence maximization problem assume that the exact threshold functions are known, but the threshold values are unknown and uniformly random. This paradoxical framework is incompatible with the problems faced by modern advertisers, who can learn noisy but non-uniformly random models of their audience from datasets of their historical behavior.

Motivated by the possibility of using data-driven approaches to solving the influence maximization problem, we provide a new framework in which the threshold functions and threshold values are inherent to the network and must be learned from logs of historical cascades. We empirically show that these network parameters can be accurately estimated from limited historical data, for a specific type of network. Additionally, we empirically show that the subset selected when given perturbed network parameters has influence similar to that of the subset selected when given the true network parameters.

1 Introduction

A network is a graphical representation of the relationships and interactions between a group of individuals. Networks can be used to model the friendships among a group of schoolchildren, the citations among a group of academic papers, or the relative travel times between a group of cities. The flexibility and applicability of these models have made network theory a widely-studied topic in sociology, game theory, computer science, and other major fields.

A particularly interesting application for networks is the study of how information, innovation, or influence spread through a social network. One can consider an election campaign which hopes to garner support via a grassroots movement. Each day, prospective voters review the political leanings of their trusted friends as well as that of the general populace, and then update their own stance accordingly. As more individuals in the network become are or accepting of the campaign, they in turn affect whether others, too, join the movement. Though each individual is acting according to their own rules and criteria, the spread of support can be thought of as occurring on the network itself. The process by which the support is successively passed on is a cascading process.

In the election campaign example above, one natural question to ask is which individuals the electoral candidate should convince in-person, in order to maximize the support garnered by the resultant

Research project for CS 6850, The Structure of Information Networks, with Professor Jon Kleinberg.

grassroots movement. Because the candidate is limited on time and resources, she can only afford to convince a fixed number of principal supports, who will go on to campaign on her behalf. Thus, it is of critical importance to the candidate to select the subset of individuals who will go on to start a maximally influential grassroots movement. This optimization problem is exactly the influence maximization problem, one of the most well-studied games on networks.

Previous methods to solving the influence maximization problem assume that the exact threshold functions are known, but the threshold values are unknown and uniformly random. In other words, this framework imposes the assumptions that each individual in the network has some known fixed valuation of her neighbor's opinions but the individuals' propensities to join the movement is uniformly random. Though this framework is highly expressive in that it can encode any submodular threshold function, it is incompatible with the problems faced by many real-world practitioners.

Consider the challenge faced by modern advertisers. Because they only have limited budgets, and because their audience is being inundated with competitor's ads, advertisers have come to rely on viral marketing – ads which are shared and redistributed by the audience itself. Advertisers are faced with the challenge of selecting the best audience members to target, so as to maximize the eventual reach of their ad campaign. This is exactly the influence maximization problem.

The previous framework for solving the influence maximization problem is incompatible with the problem faced by these modern advertisers. The framework assumes that the advertisers know exactly how each individual values the opinions of her neighbors, but then also assumes that each individual's propensity to redistribute the ad is uniformly random. Though this framework might make sense for maximizing influence of one-off cascades, it does not accurately capture what knowledge is available to modern data-backed advertisers. These advertisers can learn noisy but non-uniformly random models of their audience from datasets of their historical behavior. By looking at who had shared with whom in the past, advertisers can learn which individuals to target in order to create a successful, wide-reaching ad campaign.

Motivated by the possibility of using data-driven approaches to solving the influence maximization problem, we provide a new framework for approaching the problem. In this new framework, we treat the threshold functions and threshold values as network parameters – fixed characteristics inherent to the network itself. Additionally, these network parameters are unknown and must be learned from logs of historical cascades. In other words, one must first estimate the network parameters by looking at how previous cascades had spread through the network, in order to then use these estimates to approximate the subset of maximal influence.

We empirically show that network parameters can be accurately estimated from limited historical data, for a specific type of network. Additionally, we empirically show that the quality of the selected subset does not degrade greatly. That is, the subset selected when given perturbed network parameters has influence similar to that of the subset selected when given the true network parameters.

2 Background

2.1 Social Networks

The structures of relationships and interactions among groups of people have long been of fundamental interest to the social sciences. This includes fields ranging as broadly as history and sociology and economics. In order to study the complex interactions among large groups of people, scientists needed to characterize them using simplifying models. One particularly powerful, simple model is that of the network. Each individual is represented as a vertex on a graph, and the relationships and interations among the individuals are encoded as edges on the graph.

More formally, a network is defined as a graph $G = (V, E)$ which consists of a set of vertices V and a set of edges between them E. Each edge has exactly two endpoints, which are vertices in V. The edges may define a start vertex and end vertex, which impose a direction to the edge. The graph may allow for self loops, where both endpoints of an edge are the same vertex. However, duplicate edges are not allowed in the graph.

An example of a network is a social network. In a social network, each vertex represents an individual person in a group – such as the set of all people in the United States. Each edge then can represent a friendship between two individuals. The existence of an edge between two vertices indicates a

friendship between the two people, while the nonexistence of an edge between two vertices indicates that they are not friends.

Even after omitting the complex dynamics of an ever-changing social landscape, the network model has enough expressiveness to highlight many interesting characteristics and phenomena. For example, one could count the number of friends an individual has by looking at the degree of its vertex, i.e. the number of edges which has that vertex as an endpoint. In another example, one could determine whether a message carried from person to person could ever reach a target from a given source by examining whether there exists a path between their two vertices. Even further, one could partition a group of people by splitting their social network into connected components or strongly connected components. Clearly, there are many issues which can be expressed well in terms of the network model.

2.2 Cascading Processes

There are many processes in which some property is passed on successively from individuals to individuals. One such example is the spread of diseases through a community, as each infected individual has some probability of infecting her nearby neighbors, who in turn continue the process with their neighbors. Another example is the spread of an opinion which is repeatedly shared and possibly passed on to others in the population. Finally, an additional example is the spread of a viral video online, where users of a given social network site may share the video with their friends, who then may do the same.

All of these examples have in common that there is some structure on top of which the cascading process spreads. With the disease there is the physical distances between community members; with the opinion there is the influence individuals in the community hold over each other; and with the viral video there is the friendship graph of users of the social network site. This underlying structure is what motivates modeling these processes as occuring on top of a network structure.

In addition to the network structure itself, a cascade consists of a start set, the threshold functions, and the threshold values. In order to model a cascading process, a few steps are taken:

1. Each of the individuals to which the process may spread is modeled as a vertex in the graph.
2. The relationships between individuals are modeled as edges between the vertices.
3. A cascade begins with a start set, the set of vertices which automatically join the the cascade at time zero.
4. The criteria by which an individual decides whether or not to join the cascade are encoded in the threshold functions and the threshold values. If the value of a given vertex's threshold function exceeds its threshold value, then the vertex joins the cascade.
5. The cascade runs until no new vertices are willing and able to join.

Thus, a cascading process is defined by the network structure it runs on top of, the set of vertices which join at the start, the threshold functions for each vertex, and the threshold values for each vertex. Other properties of a cascading process, such as run time and influence, are secondary properties which are defined implicitly by (or computed explicitly from) those primary properties.

2.3 Influence Maximization Problem

When considering cascading processes, a very natural question to ask is how widely the cascade can spread. If the graph is infinite, is is possible for the cascade to spread to the entire graph, or at least spread infinitely? If half of the vertices have joined the cascade, will the entire graph join? How small can a start set be, such that the entire graph may still join the cascade?

All of these problems concern the influence of a cascading process. The influence is defined at the number of vertices that join the cascade by the time the cascading process finishes running, whether that be infinite or not. Intuitively, the influence is the eventual reach of the cascade. Because threshold functions and threshold values may be randomized, the influence is not inherent to a given cascading process, but rather the result of a single run. Thus, a given cascade with a fixed network, fixed start set, fixed threshold functions, and fixed threshold values may produce a different influence values on different runs.

One interesting problem is that of influence maximization. Given the network and the thresholds, what starting set should we pick such that the expected influence of the resultant cascading process is maximized? Obviously, if the start set is the entire graph, then the influence is trivially maximal. Instead consider the situation where the size of the start set is restricted, perhaps due to budgetary or physical restraints. The question becomes an optimization problem: what size-restricted starting set should one pick, in order to maximize the influence of the resultant cascading process?

More formally, we define the influence function $\sigma(A)$ to be the expected influence of a cascading process starting from the set A, where the network and thresholds are fixed. The influence maximization problem is thus to select the start set $A \in V$ such that $|A| \leq k$ and $\sigma(A)$ is maximal, for some fixed size parameter k.

2.4 Previous Work on Influence Maximization

In order to discuss the previous approaches to approximating the influence maximization problem in Kempe et al. [3], we must first describe the framework for modeling casacding processes defined in Kleinberg [4]. This framework is critical to deriving the results obtained in Kempe et al. [3]. It is also where we make our fundamental modification in order to approach the influence maximization problem from a data-driven perspective.

The General Threshold Model. Thus far, we have discussed threshold functions and threshold values generically, without defining what values they can take on or what inputs the threshold functions can receive. Though the threshold functions can be arbitrarily complex in real-world applications, the General Threshold Model make simplifying assumptions to make the problem space more manageable. In particular, the authors designed the General Threshold Model in order to allow for restrictions which force the influence functions to be submodular.

As described in Kleinberg [4], the General Threshold Model defines the thresholds as such:

- For a given cascading process, each vertex v has an arbitrary monotone threshold function $g_v(\cdot)$ defined on subsets of its neighbor set $N(v)$, such that for every $X \subseteq N(v)$ the output $g_v(X)$ is between 0 and 1.
- For a given cascading process, each vertex v has a threshold value which is chosen uniformly at random in $[0, 1]$ for each run of the process.

Thus, the influence function $\sigma(\cdot)$ for a given cascading process is a random variable whose value is determined by the threshold values for a given run of the process. After fixing the network structure and the threshold functions, the influence maximization problem becomes a question of selecting the start set which defines the cascading process with maximal expected influence.

It is important to note that in the General Threshold Model, the threshold functions and starting set are properties of the cascading process, and the threshold values are uniformly random variables whose values are fixed for a given run of the cascading process. Thus many cascading processes may share a given network, but the threshold functions may vary greatly from process to process.

A primary benefit of defining the General Threshold Model in this manner is that if every threshold function is submodular, then the influence function itself is submodular [4]. The authors define submodularity as the property of a function where adding an element to a set Y causes a smaller marginal increase than adding the same element to a strict subset of Y. More formally, a function f is submodular if for all sets $X \subseteq Y$ and all elements $v \notin Y$, it is true that

$$f(X \cup \{v\}) - f(X) \geq f(Y \cup \{v\}) - f(Y).$$

The submodularity of the influence function is critical for the following approximation algorithms for influence maximization.

Approximization Algorithms for Influence Maximization. The influence maximization problem is generally a hard computationally intractable problem. In fact, in many cases the influence function itself is NP-hard to even approximate. This means that for broad classes of influence maximization problems, the influence function is intractable to even approximate – let alone optimize over.

So what hope is there to solving the influence maximization problem? One strategy is to focus on subclasses of the problem for which the optimal solution can be approximated. The approximation algorithms for influence maximization described in [3] are designed for situations in which the influence function is submodular. This includes widely studied models such as the Linear Threshold Model and the Independent Cascade Model. Both of these classes are subclasses of the General Threshold Model, specifically subclasses in which the influence function is submodular.

The reason for focusing on submodular influence functions is that there exists an efficient algorithm for approximating the optimal size-restricted input for submodular functions. This algorithm is the greedy hill-climbing algorithm which iteratively selects elements to add to its chosen set, greedily maximizing the value of the current set. In the context of influence functions, the algorithm:

1. Selects the vertex which individually results in the greatest influence, and adds it as the first element of the selected set.
2. Selects the vertex which – when added to the current selected set – results in the greatest increase in influence.
3. Repeats the greedy selection process in step 2 until k vertices are selected.

When the function in question is submodular, the hill-climbing algorithm selects a k-sized subset whose influence is within a factor of $(1 - 1/e)$ of that of the optimal k-sized subset [3].

The approximation bound above can be achieved when the optimized function can be computed exactly. However, this is not typically true for the influence function in the General Threshold Model. Because the influence depends on the specific assignment of threshold values for each particular run of the cascading process, the influence function varies from run to run and thus cannot be computed exactly.

Instead, one can use the hill-climbing algorithm to approximate the optimal *expected* influence. By running many simulations of the runs of the cascading process, one can find arbitrarily close estimations of the expected influence. Thus, one can run the hill-climbing algorithm using estimations of the expected influence and get approximate approximations of the optimal expected influence. Essentially we run the hill-climbing algorithm as per usual, but every time the algorithm calls to evaluate the function, we return an estimate of the expected influence. This generalization of the hill-climbing algorithm provides approximation guarantees arbitrarily close to $(1 - 1/e)$ [3].

3 Model

Advertisers and Motivation. Our work is motivated by the need to more accurately model the problems faced by – for example – modern, data-backed advertisers. Consider the challenge faced by modern advertisers. Because they only have limited budgets, they can only afford to target a small subset of their intended audience. Thus, they have come to rely on viral marketing – ads which are shared and redistributed by the audience itself. The targeted subset is the start set from which the cascading process of ad redistribution begins. In order to increase the effectiveness of their ad campaigns, advertisers must strategically select their target subset so as to maximize the influence of the resultant cascading process.

The previous framework for solving the influence maximization problem is incompatible with the problem faced by these modern advertisers. In the old framework, the advertiser is assumed to know the threshold functions exactly while the threshold values are uniformly random. In other words, the framework assumes:

- The advertisers know exactly how each individual values the opinions of her neighbors.
- Each individual's propensity to redistribute the ad is uniformly random.

Though this framework might make sense for maximizing influence of one-off cascades – where the threshold functions can be assumed to be of a fixed class, but the threshold values are completely unknown – it does not accurately capture what knowledge is available to modern, data-backed advertisers. These advertisers can learn the thresholds by using the logs of how previous ad campaigns spread through the network. By observing who shared ads with whom in the past, advertisers can effectively build a model of their audience. Thus modern advertisers require a

framework in which they can utilize this learned knowledge.

A New Framework. In this section, we introduce a new framework for approaching the influence maximization problem. The primary difference betwen our approach and that in Kempe et al. [3] is how we formalize cascading processes. Though the mechanism by which a cascading process spreads across a network remains the same, we make a critical change in how thresholds are modeled.

Instead of being properties of a given cascading process, the threshold functions and threshold values are inherent to the network itself. In other words, a given network is defined in part by the threshold functions and thershold values of its vertices. These network parameters remain fixed for a given network and do not change as different cascading processes run on top of the network. This is a reasonable assumption when dealing with individuals whose behavior is approximately fixed over multiple cascades. For example, if multiple advertisers were to run similar viral ad campaigns on a social network site, users' propensities to share the content may be reasonably fixed.

More formally, we define a network to consist of:

- A graph $G = (V, E)$ which consists of a set of vertices V and a set of edges E.
- For each $v \in V$, an arbitrary monotone threshold function $g_v(\cdot)$ defined on subsets of its neighbor set $N(v)$, such that for every $X \subseteq N(v)$ the output $g_v(X)$ is between 0 and 1.
- For each vertex $v \in V$, an arbitrary threshold value which is chosen from $[0, 1]$.

Futher, we define a cascading process to consist of:

- The network on which the process runs.
- The set of vertices which join at the start, A.

Again, the steps by which the process runs remains the same; each vertex iteratively joins the movement if-and-only-if the value of their threshold function surpasses their threshold value. Clearly, differing cascades running on the same network differ only in their choice of start set. Thus the influence maximization remains a question of selecting the subset of vertices whose resultant cascade has maximal influence.

Data-driven Approach to Influence Maximization. The motivation behind designing out new framework was to allow practitioners to learn from logs of previous cascades. Given our new framework for modeling cascading processes, we can now describe our data-driven approach to solving the influence maximization problem. Using our intuitive, two-step approach, one may use logs of historical cascades to closely approximate the optimal starting set.

The first step is to estimate the network parameters from the logs of historical cascades. This is essentially a regression problem on the historical data. One must determine the set of network parameters which best fit the logs of how the previous cascades had spread across the vertices. This can be using a priori or a posteriori likelihoods. Additionally, the network parameters may be modeled as fixed values or fixed distributions. If the learned threshold values are modeled as distributions, then in the second step, one must use simulations to estimate the expected influence function, rather than calculate the influence function directly.

The second step is to apply the estimated network parameters and approximate the subset of maximal influence. In the case where every threshold function is submodular, the influence function is submodular, and one can run the hill-climbing algorithm. In this situation, the second step is very similar to the approximation algorithm for influence maximization described in Kempe et al. [3], but the threshold values come from distributions other than the uniform random distribution. In other other words, when one estimates the influence function through simulations, the threshold values take on values from distributions that are different than – and possibly tighter than – the uniformly random distribution.

This improved tightness is a key benefit of our framework when compared to a framework which models the threshold values as uniformly random. By utilizing tighter bounds on the threshold values, one can gain much better approximations of the true influence function. This in turn can lead to improvements in the quality of the approximations of the optimal subset.

4 Results

The two main objectives for the empirical investigation were:

1. Produce usably tight estimates of threshold values for fixed-degree graphs with linear thresholds.
2. Show decently small degradation of subset quality (in terms of true influence function) when given perturbed network parameters. Also for fixed-degree graph and linear threshold model.

4.1 Deriving bounds on threshold values for fixed-degree graphs with linear threshold model

4.1.1 Method

Observing any time-series of vertex activations during a cascade provides upper and lower bounds for the threshold values of any given vertex which happens to activate, or does not activate despite having activated neighbours. This straightforward phenomenon was used to calculate bounds on the threshold values for individual vertices in a graph when given a number of random cascade timeseries.
The timeseries were produced by randomly sampling a seed set to activate in the graph, and initiating a cascade from that seed set.

The parameters of the model and graph for this experiment are as follows

1. threshold values randomly sampled from [0 , 0.5]
2. 100,000 vertices in graph
3. fixed-degree random graph, every vertex degree=100
4. starting set randomly sampled, size=100

4.1.2 Results

The results were promising in terms of providing useful bounds on the threshold values. The objective function chosen to measure performance of the model was average bound tightness, defined as

$$objective = \frac{\sum_{i \in vertices}(upper_i - lower_i)}{|vertices|}$$

A steep drop could be observed after just a few cascades, as can be seen in Figure 1. The bounds themselves eventually dropped to order of $\tilde{0}.1$. This suggests high precision on a majority of the nodes, suggesting feasibility for the use of these parameter estimates for the process of maximum influence set prediction.

4.2 Investigating degradation in greedy algorithm for maximum influence set estimation when given noisy threshold values

4.2.1 Method

The graph in use is identical to the previous experiment, with the exception of it being two orders of magnitude smaller in order to make the experiment computationally tractable (1000 vertices), as computational power and time was a significant limit. Additionally the threshold values are sourced from [0,1] as opposed to [0 , 0.5], and the fixed degree of the vertices in the random graph is 10 instead of 100.

Initially, the graph was perturbed by modifying the threshold vector with a vector of values sampled *i.i.d* from $\mathcal{N}(0,\, \sigma^2)$. The values of σ tried were 0.1, 0.01.

In order to obtain a smooth estimate of the performance of the influence set generated using the noisy data, several runs with different random seeds for the perturbation were performed and averaged to avoid outliers (which appeared, given the smaller size of the graph).

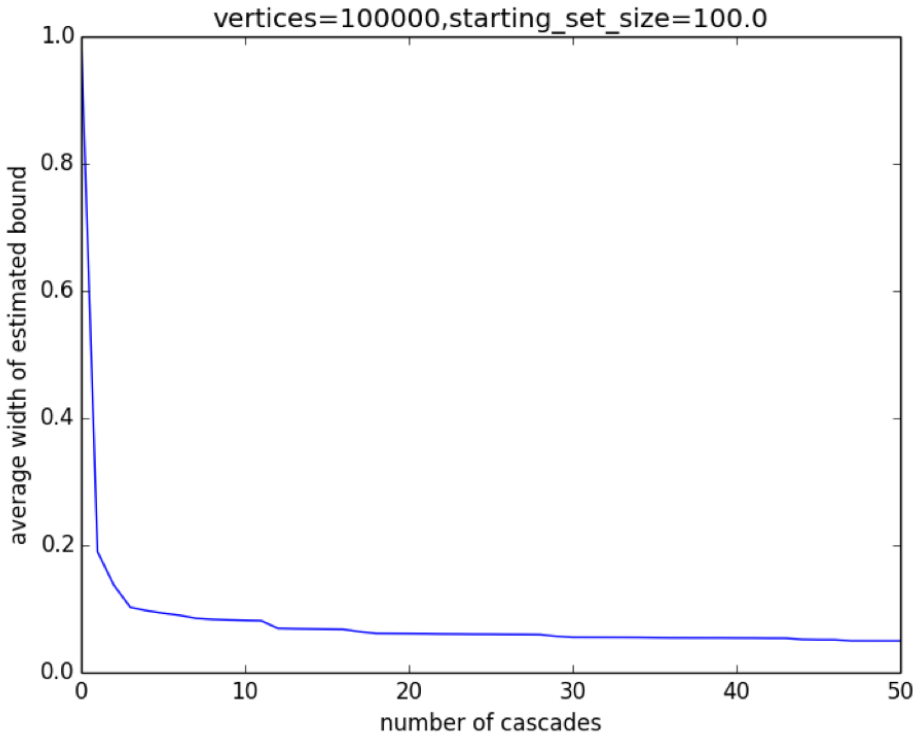

Figure 1: A plot showing the very quick approximation of bounds on threshold values, after observing only a few random cascades.

4.2.2 Results

Once again, this method shows promising results for the global approach of parameter estimation and in turn maximum influence set approximation. There was noticeable, but not drastic, degradation in final influence size when influence set solutions were computed with the random noise present in the thresholds. As can be seen in Figure 2, all three solutions exhibit the same plateau of diminishing returns, but the noisy solutions are consistently worse than the "optimally computed" greedy solution. Noise with a $\sigma = 0.1$ provides approximately a 25% penalty in this configuration, making it non-ideal but not useless.

5 Future Directions

The scope of the work was highly limited by the implementation and computational power available. Additionally, the computationally intractable nature of the influence function in addition to the many iterations required for the greedy hill-climbing algorithm further complicated matters. As the work done was experimental in nature, the model and framework had to be implemented in code. For ease of prototyping, this was done in Python using the "graph-tool" framework [1]. However, lack of access to high-performance computing as well as the Python implementation limited the speed at which the influence function could be computed.

The three main areas which could provide useful results are:

- Rich-get-richer (scale-free) graph analysis
- Threshold functions other than linear threshold
- Immediate approximation of maximal influence set

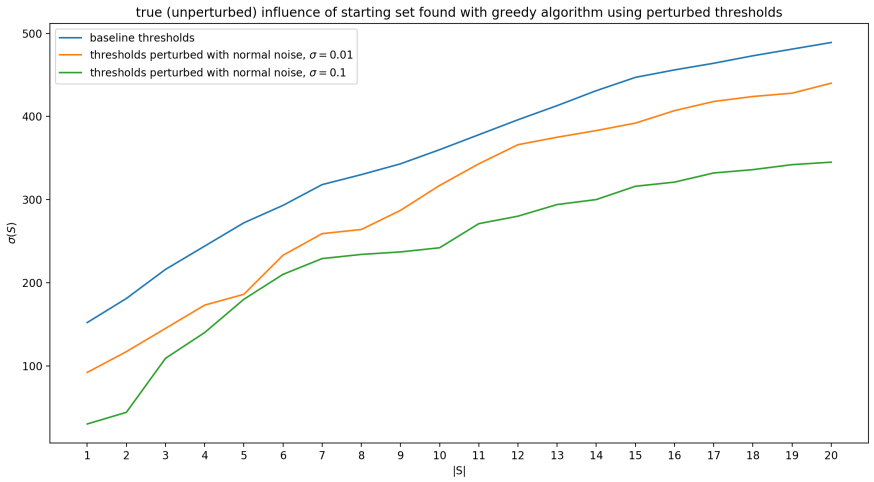

Figure 2: The influence of a starting set determined by the greedy algorithm, run with varying levels of noise.

5.1 Rich-get-richer (scale-free) graph analysis

Rich-get-richer was not implemented in this work due to the difficulty of producing bounds on the threshold functions for a linear threshold model applied to this graph structure. The only scale-free model generator available in the library used for the work was the Barabási-Albert model, which produced an extremely long tail of nodes with one or two vertices.

As the name "rich-get-richer" implies, the distribution of the degrees across the vertices in this graph is extremely unequal - very few of the vertices have almost all of the outgoing edges. Thus, the vast majority of vertices have extremely few outgoing edges (depending on the configuration of the Price model, as few as one or two). Given only two neighbouring vertices, it becomes almost impossible to attempt to estimate thresholds for such a vertex since only two unique bounds can be determined (depending on which of the two neighbours are activated). This problem cannot easily be solved, as any combination of three nodes can only represent a finite set of states, and thus any configuration of cascades can only give a limited amount of information (regardless of thresholds or threshold functions).

One approach to analyzing rich-get-richer models would be to use such a scale-free model more similar to real-life social networks. There have been a number of attempts to produce such graphs, including work such as:

- *Network formation models [2]*
  There exists some research on generating fake social graphs not using simple allocation rules such as Erdos-Renyi, but more complicated rules such as simulating "meeting someone through a friend", etc. These generation models claim to better model social networks by mimicking the processes that supposedly underpin them.

- *Kronecker-graphs [5]*
  Kronecker graphs are an extremely promising field of research. In essence, they provide a framework to compress down some network model (a social network for instance) into a "seed graph", which can then be expanded to produce a random graph with extremely similar properties to the original social network (in terms of degree distribution, triangle participation, etc.).

Using either of these models in the analysis could provide further useful insights into efficient maximization of influence in social networks.

5.2 Threshold functions other than linear threshold

An important and limiting assumption made in this work is the use of a linear threshold model for the cascading processes. As previously justified, this model makes simulations much more computationally tractable, as well as simplifying analysis. However, it is unlikely that linear models are the ones that prevail in real life social networks.

The cascade model we chose to work with would in theory allow for almost any threshold function to be in use - as it would simply be some function on the subset of of neighbours activated. Thus, such a function could be arbitrarily complex. In real life, one would expect people to behave extremely non-linearly - for instance if multiple closely related family members share the same post it is very likely the member in question would also share the post (it could be news about a family member, for instance), however if only a single parent shares an item the same family member might be much less likely to share the same post. Capturing such arbitrary mappings is a difficult task, both in terms of parameterizing a function capable of expressing this, and especially in terms of using historical data to estimate said parameters.

One promising approach in this field would be to in some way incorporate neural networks to aide in this process. Artificial neural networks are widely hailed as being able to encode or simulate almost arbitrary functions, especially functions of naturally occurring phenomena (such as the "internal" fuzzy function in our brains of mapping some input image to a corresponding animal when we see one). Furthermore, they tend to perform well even given noisy data and have shown a remarkable ability to generalize. Possibly training a neural network on the historical cascade data may provide interesting results.

5.3 Immediate approximation of maximal influence set

In this work, we introduce a new framework for approaching the influence maximization problem from a data-driven perspective. The framework allows practitioners to learn from logs of historical cascades in order to select the subset of optimal influence. However, in our approach we simply assume that the best strategy for solving the influence maximization problem using data requires two steps:

1. Estimate the network paramenters for logs of historical cascades.
2. Approximate the optimal starting set using the estimated network parameters.

Though these two steps are intuitive and easy to understand, we provide no evidence that this is the optimal way to approach this the data-backed influence maximization problem. In fact, it may very well be true that a simple heuristic would perform better in real-world scenarios.

Thus we propose investigating alternative, one step strategies to solving the data-back influence maximization problem. Possible heuristics that look at the historical data include:

- Choose the k vertices which were the "tipping point" (i.e. the last neighbor to join the movement before a given vertex joins as well) for the most number of other vertices.
- Choose the k vertices with the highest centrality measures (e.g. betweenness, eigenvector centrality) within the subgraphs induced by the influenced sets.
- Choose k vertices which were present in successful, large influenced sets, but absent from unsuccessful, small influenced sets.

The intuition behind choosing a one step strategy is that less information may be lost or obfuscated if fewer intermediate parameters need to be estimated. A great way to compare these alternatives against our algorithm would be to first run them on simulations similar to the ones above. After fixing the network parameters of a given fixed-degree graph, we could create a dataset of historical cascades by simulating many cascades on the network. Then we could run these heuristics on the datasets and compare the true influence of the resultant starting sets to the true influence of the starting set selected by our algorithm.

References

[1] graph-tool: Efficent network analysis with python. URL `https://graph-tool.skewed.de/`.

[2] Matthew O Jackson and Brian W Rogers. Meeting strangers and friends of friends: How random are social networks? 2004.

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

[4] Jon Kleinberg. Cascading behavior in networks: Algorithmic and economic issues. *Algorithmic game theory*, 24:613–632, 2007.

[5] Jure Leskovec, Deepayan Chakrabarti, Jon Kleinberg, Christos Faloutsos, and Zoubin Ghahramani. Kronecker graphs: An approach to modeling networks. *Journal of Machine Learning Research*, 11(Feb):985–1042, 2010.