ETH Zurich

Applied Network Science: Social Media Networks

Social Platform Virality

Own work Report

Dec 10, 2021

Authors:
Juyi Zhang
Andri Bernhardsgrutter

Coach:
Wei Zhang

Contents

List of Figures

1 Introduction

In studying the paper, one idea came to our mind. We could use the vast dataset to simulate an account and study the growth. And that is how we decided to come up with the Social App Simulator.
The Social App Simulator is a way to simulate a virtual account and study the growth of the virtual account. We have added several variables in order to fully simulate the reality. In the app, you can modify these parameters with a virtual day and the growth will be reflected. The prototype of the social app simulator is shown in fig. 1

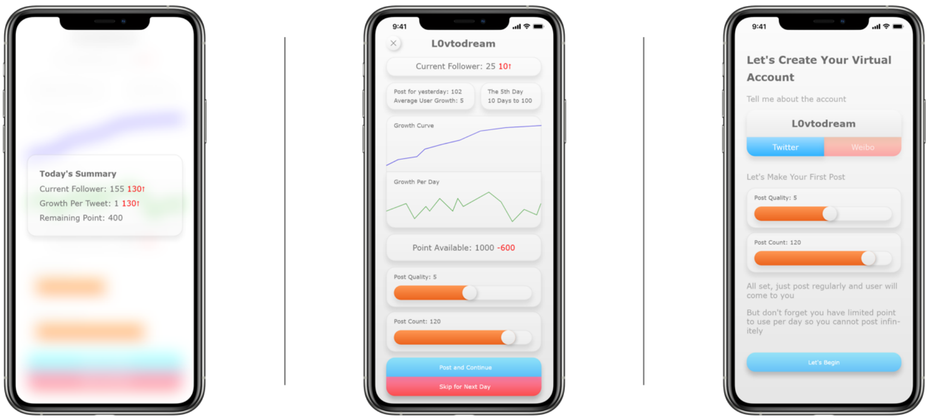

Figure 1. Prototype of the social app simulator

When creating the application and processing the data, we have also found some interesting data patterns we would also like to share in this document.

2 Implementation

The app is completely built on the previously mentioned vast database and we used our algorithm to extract the statistical data from the dataset. As the paper mentioned, we have utilized the heterogeneous percolation model to take the current size of follower into consideration. In the implementation, we stratified the data provided by the amount of followers and later also with the tweet count. And we reach a possibility of growth in each user group, we then construct the model based on this possibility. Our data processing is split into two parts. One part is with matlab where we process the data and generate the stratified model for us to import later in python. In the python part, we implement the model by calculating the growth of the user each day and generate a user friendly interface to operate on.

2.1 Matlab

Our matlab part is mainly focused on creating a stratified model complying with the heterogeneous percolation model and process the raw data from the paper to generate plots which have more readable patterns. Such patterns will lead to the conclusion we draw in the next section.

2.1.1 Split Array

The function uses the given step to split up the raw data by follower count. We have utilized an logarithmic scale to match up with the network structure and in the model we demonstrated the step is set to 50. With more data available we are able to expand the step.

2.1.2 Error Plot Log / Error Data Log

The functions will generate three plots or corresponding data discussed later: x with respect to y, distribution of x, as well as distribution of y with stratified distribution of x. By splitting the value of x according to the step (variable num), minimum value (variable min) and maximum value (variable max). The generated data will be further processed and converted into a model.

2.1.3 Normalize

The function normalizes the data by calculating the sum of all columns and divide each element by this amount. The normalization will result in a probability distribution and is later utilized to estimate the probability of reposting.

2.1.4 Main

The Main function will read the raw data from the txt file provided from the paper and call the corresponding function (Plotter for plotting the data and Data Interpreter for generating the model) twice (for Twitter and Weibo). The end result is 12 graphs for both Weibo and Twitter and a 50*10*10 model that is used in python part.

2.2 Python

Our python part mainly deals with implementing the model and presenting it with a user interface.

2.2.1 Initialization and GUI

The initialization part will read the output of data.mat (which is our model) and assign the corresponding variable of the model. The result variable and helper variable is also initialized, as well as importing the required library (mainly pyqt5, pyqtgraph, scipy, numpy). The GUI part is the most coding intensive part which will firstly take the user's input and pass it to the data processing part. Then it will display the value specified by the data generated by this part.

2.2.2 Data Processing

This part will take the user input from the GUI and calculate the growth of followers with the model. In order to convert the raw data from the GUI to the model data, several helper functions are included to process the data and we will determine the interval that the input value lies in. Additionally, the amount of time for each data set is different and we make sure to remove the effect of such time scale. Lastly, we also convert the post quality input from the user to the repost possibility by randomizing the parameters. Note that the approach is not verified by the actual model and we could work on this part a bit more given more time.

2.3 Difference between our simulation and model from the paper

Our implementation has utilized the concept in the paper and was converted into a simulation software. However, in the paper, the model focuses on the information cascading process on one special event, while our model mainly studies the growth of the account which is not tied to a specific event. Even though the two subjects can seem different, both of them is in fact the representation for the core of the heterogeneity model.
The simulation also has an internal variable that controls the repost possibility β the same as the model illustrated by the paper. However, instead of predicting the amount of accounts one post can reach (percolation), we predict the average amount of accounts such posts can reach from the data of the paper (available at: https://github.com/Jia-Rong-Xie/data-DMRP). Such a prediction is constructed based on stratifying different layers of followers and tweet counts with respect to the follower change.

3 Result

The result will focus on discussing the plots generated by the errorplotlog function and includes recreations of the paper's graphs as well as the creation of our own graph.

3.1 Follower Count versus Tweet Count

The first two plots shown in fig.2 are reproductions of the paper's results. However, instead of stopping at 10^7, we have extended the graph as far as possible, and we have found an interesting downward trend. The trend can be explained by the accounts to act pretty conservatively and reduce the tweet count when the followers gets saturated. Additionally, the account with more followers has a higher possibility of having a professional operation team instead of being a purely personal account, making its tweeting pattern more regulated.

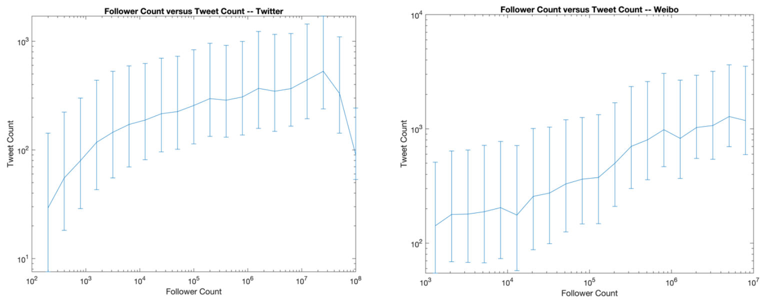

Figure 2. Follower Count versus Tweet Count

3.2 Tweet Count versus Follower Change

In addition to the follower vs tweet change, we have also studied the tweet count versus the follower change and the plot is shown in fig.??. It is well expected that with a higher tweet count, the more follower change there should be. However, we also see that Twitter has a relatively flat curve especially for a medium amount of tweet count while Weibo is seeing the flattening at a later stage. It is possible that such a difference is due to the social impact, where the change of the behavior of the individual is the result of interaction of other individuals.

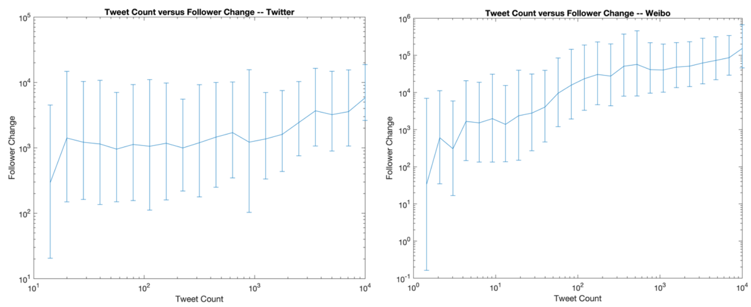

Figure 3. Tweet Count versus Follower Change

3.3 Network Structure

We also discover an interesting effect of network structure in Twitter and Weibo and their user distribution is shown in fig.4 The two graphs looks similar with the amount of users first seeing a decrease and then an increase. We are proposing that such an effect can possibly be due to the personal network (or offline network) that people have brought to their online network. The fact that people have a limited offline network results in the sharp decrease of the curve. We also found some spikes in the Weibo graph indicating that this platform might have a higher tendency to buy bot accounts which follow the user at one thousand, ten thousand and hundred thousand followers.

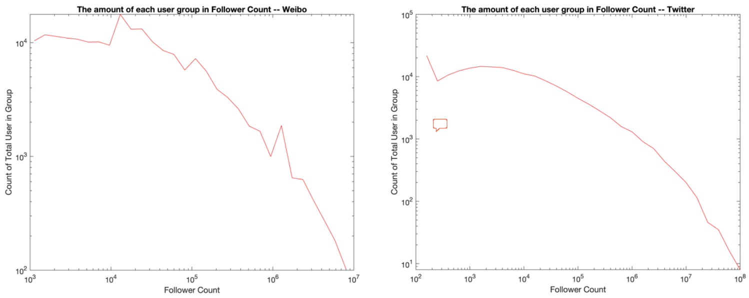

Figure 4. The amount of each user group in follower count

3.4 Tweet Count versus Sample Portion with Stratified Follower Count

From fig.5 we can clearly see that as follower count increases, the amount of tweets increases as the line corresponding to more follower count is above. Therefore, the willingness to tweet increases, or they are having a more proper account operation. Twitter and Weibo share a similar curve but they do have a shape difference where Twitter has a curve more concentrated on the top. Indicating a more healthy user tweeting pattern.

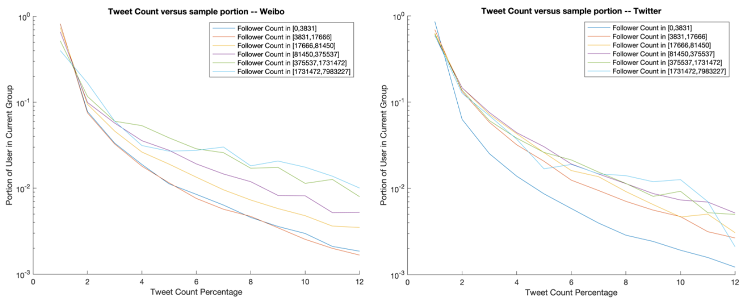

Figure 5. Tweet Count versus Sample Portion

3.5 Follower Change versus Sample Portion with Stratified Follower Count

Lastly, we also plotted the graph with respect to follower change with respect to tweet count in fig.6. We can see that a higher tweet count will generally lead to a higher follower change. As the line corresponding to a higher tweet count is higher. Additionally, we have also noticed an exponential effect within the same amount of tweet count. The follower count will change exponentially indicating that better repost possibility is only available to a limited amount of tweets.

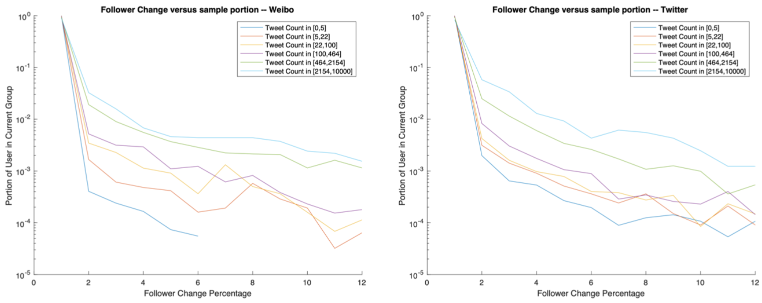

Figure 6. Follower Change versus Sample Portion

4 Conclusion

4.1 Current Work

We have achieved a lot in our current work. We firstly created the social app simulator by extracting the model from the data provided by the paper. During the data processing, we have also generated some plots which lead to some pretty interesting conclusion on follower distribution spikes and dips. We have also found the tendency of tweeting with respect to follower count, as well as the follower change in respect to tweet count.

4.2 Further Work

Given the time being more sufficient, we would like to firstly analyze the full dataset. Due to the time and computing power constraint, we are not able to process the raw data for each user. Additionally, the data utilized by the paper is relatively old and it would be interesting to study the impact of AI based recommendation algorithms instead of growth with respect to retweeting. Lastly, we would also like to expand the application so that it gets a better user interface as well as a consideration of the multiple day impact of one post.

4.3 Code Availability

The code for the report is available at https://github.com/HarunaIppai/TSAS