Beyond 280 Characters: A Non-Conservational Approach to Capture the Spread of Hate Speech on Twitter

BY

Aatman Vaidya

AU1940108

Discipline of Computer Science and Engineering

Ahmedabad University

SCHOOL OF ENGINEERING & APPLIED SCIENCE
AHMEDABAD UNIVERSITY, AHMEDABAD

May 2023

Beyond 280 Characters: A Non-Conservational Approach to Capture the Spread of Hate Speech on Twitter

A Thesis submitted in partial fulfillment of the requirements for the award of the Degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE and ENGINEERING (CSE)

Submitted by:

Aatman Vaidya

AU1940108

Guided by:

Amit Nanavati

Seema Nagar

Ahmedabad University

May 2023

Declaration

This is to certify that:

1. The thesis titled, *"Beyond 280 Characters: A Non-Conservational Approach to Capture the Spread of Hate Speech on Twitter"* comprises my original work towards the degree of Bachelor of Technology in Computer Science and Engineering (CSE) at Ahmedabad University, and it has not been submitted elsewhere for a degree.

2. Due acknowledgment has been made in the text to all other material used.

Signature of the Candidate
Aatman Vaidya,
Ahmedabad University.

Certificate

This is to certify that the thesis titled, *"Beyond 280 Characters: A Non-Conservational Approach to Capture the Spread of Hate Speech on Twitter"* is the bonafide work carried out by Aatman Vaidya, a student of BTech (CSE) of School of Engineering and Applied Science at Ahmedabad University during the academic year 2022-2023, in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering and that the thesis has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar title.

Signature of the Guide
Dr Amit Nanavati,
Professor,
Ahmedabad University.

Place: Ahmedabad
Date: May 2023

To Aai, Baba and Anannya

Abstract

The spread of hate speech on social media platforms has become a rising concern in recent years. Understanding the spread of hate is crucial for mitigating its harmful effects and fostering a healthier online environment.

In this work, we aim to propose a novel model to capture the spread of Hate Speech on Twitter. The model effectively captures how toxic a social network will become when a tweet with a certain toxicity (hatefulness) is posted. We compute a toxicity score for each tweet which indicates the extent of the hatefulness of that tweet. Previous work, such as SIR (Susceptible-Infected-Recovered) models and their variants, use a threshold-based binary approach and therefore are not suitable for modelling this situation as toxicity exists on a spectrum. Spread-activation models (SPA) often treat hate as energy and use energy-conservation principles to model its spread. By analysing a dataset of $19.58M$ tweets from January 2017 to October 2017, we observe that the total toxicity, as well as the average toxicity per user, is not conserved but rather increases over time. The previous works do not address the question of how much hate a single tweet can generate or the extent to which a social network becomes more hateful over time.

We, therefore, propose a novel model to capture the spread of toxicity, where in we divide the users into three categories: Amplifiers (who receive less hateful tweets than they send out), Attenuators (who receive more hateful tweets than they send out) and Copycats (who send what they receive). Our model is able to reproduce the increase in the total and average toxicities per user for simulated Barabási–Albert graphs as well as real-world graphs. Additionally, the proposed model also effectively reproduces the behaviours of the categorised users.

Analysing the user behaviours and the spread of toxicity in our proposed model may also suggest more practical and effective ways of inhibiting the spread of the spectrum of hate on Twitter.

Acknowledgements

I express my deepest gratitude to my advisor, Professor Amit Nanavati, for his invaluable guidance, expertise and unwavering support throughout the thesis. From giving me direction to work on to discussing ideas, to small things like helping me clean the dataset and fix errors in the code and to always pushing me to do what I like and what interests me. This has been instrumental in shaping this work. I sincerely appreciate their insightful feedback and patience, and I am grateful for his mentorship.

I extend my gratitude to my co-advisor, Dr Seema Nagar, for her guidance, expertise and constant support throughout. She played a pivotal role in helping us choose the topic, generating ideas, and being our go-to person for problem-solving and fixing things. I sincerely thank her for their invaluable contributions.

I would also like to extend my heartfelt thanks to my friend, Hirmay, for his invaluable help throughout. From sitting with me to write code, listening to me talk at length about the work and giving his suggestions to fix latex errors, I am truly thankful for all his help.

Lastly, I want to thank Aai for always pushing me to pursue research and my family for their constant support and encouragement.

Contents

List of Figures

List of Tables

Acronyms

SIR	Susceptible-Infected-Recovered
SPA	Spread Activation
BA	Barabási–Albert
API	Application Programming Interface
ODE	Ordinary Differential Equation
SEIR	Susceptible-Exposed-Infected-Recovered
ESIS	Emotional-Spreader-Ignorant-Stifler
S-SEIR	Single Layer-Susceptible-Exposed-Infected-Recovered
SCIR	Susceptible-Contacted-Infected-Removed
irSIR	Infection Recovery-Susceptible-Infected-Recovered
FSIR	Fractional-Susceptible-Infected-Recovered
BERT	Bidirectional Encoder Representations from Transformers
IQR	Interquartile Range

Chapter 1

Introduction

"They deserved it" were the words of a man who brutally murdered worshipers at a Pennsylvania synagogue. He was very active on a white supremacist social media platform and posted hateful content[1]. Social media has become an integral part of our lives; it has changed the way we communicate, express ourselves, share information and interact with each other. Twitter is a popular platform for public discourse, where users on the platform share their thoughts and opinions. With that said, there are adverse effects of social media, such as online harassment, cyber-bullying, hate speech etc. Hate speech has often been categorised as "trolling", but its severity and brutality has grown more alarming in recent years. Hate Speech has led to dreadful scenarios like the anti-Muslim mob violence in Sri Lanka[2], the Pittsburgh synagogue shooting[3], the Rohingya genocide in Myanmar[4], and the shooting at the Sikh temple in Wisconsin[5].

The Observer Research Foundation released a study [1] which, based on statistical mapping of hate speech accounts on online social media, found that there was a strong correlation between online hate speech and offline violence. The report cited several cases where hate speech online led to mob attacks, lynching, riots and several other atrocities in India. Although big tech giants such as Facebook and Twitter have strict rules against hate speech, they fail to moderate it well and take necessary action to curb it[6]. A study found that suicide rates among ethnic immigrant groups strongly correlate to the hate speech directed towards them [2]. Olteanu et al. modelled the

[1] https://www.washingtonpost.com/nation/2018/11/30/how-online-hate-speech-is-fueling-real-life-violence/

[2] https://www.theguardian.com/world/2018/mar/14/facebook-accused-by-sri-lanka-of-failing-to-control-hate-speech

[3] https://www.nytimes.com/2018/10/27/us/active-shooter-pittsburgh-synagogue-shooting.html

[4] https://www.reuters.com/investigates/special-report/myanmar-facebook-hate

[5] https://www.bbc.com/news/world-us-canada-19143281

[6] https://www.technologyreview.com/2018/12/28/1527/facebooks-leaked-moderation-rules-show-why-big-tech-cant-police-hate-speech/

volume and type of hate speech on Twitter and Reddit and analysed its effects. They found that extremist violence caused a rise in online hate speech [3].

This brings attention to a dire need to understand and map what is happening to hate speech; How is it spreading? Can we understand user behaviour and dynamics facilitating the spread of hate? Through our work, we aim to look at the spread of hate speech on Twitter, analyse the factors contributing to its spread and understand user behaviour.

Before we begin our work, we first have to understand and establish the meaning of "Hate". *““What is Hate””* is a complex social sciences problem, but the spread of hate on a given network becomes a computer science problem and can be addressed with machine learning and social network analysis. Countries worldwide have different approaches to addressing hate speech with their respective legislative frameworks and legal systems[7]. ElSherief et al. performed a linguistic analysis of hate speech in social media. They defined hate as a “direct and serious attack on any protected category of people based on their race, ethnicity, national origin, religion, sex, gender, sexual orientation, disability or disease.” [4]. The Twitter policy of Hateful Conduct defines hate similarly[8]. We have used a machine learning algorithm called *Perspective*[9] and have assigned a toxicity score in the range [0-1] to each tweet; this suggests how toxic or hateful a tweet is. For our research, we are bound by the definition of hate as that of perspective API. We shall go by the following definition of hate in our work.

“We define ’Toxicity’ as a rude, disrespectful, unreasonable comment that is likely to make someone leave a conversation.”

Previous work looked at various important aspects of hate spread but do not address the question of *how much hate* a tweet could generate, or *how hateful (toxic)* a social network would become over a period of time. Our analysis shows that capturing hate speech using Spread Activation Modelling (SPA) and SIR (Susceptible - Infected - Recovered) models is insufficient for our approach. SIR models assume that any user is in one of a few sets of states at any given time, such as: “infected (hateful), “exposed", “recovered"(not hateful), etc. In our case, each user may send and receive messages with various toxicities in the range [0-1]. Further, we observed that toxicity in the social network is increasing over a period of time. Since toxicity

[7]https://www.cfr.org/backgrounder/hate-speech-social-media-global-comparisons
[8]https://help.twitter.com/en/rules-and-policies/hateful-conduct-policy
[9]https://perspectiveapi.com/

is not conserved, the SPA model and its variants are not suitable for modelling the toxicity of a network. Therefore, we need a new model to examine toxicity spread on social networks.

We, therefore, propose a new model to capture the spread of the *spectrum* of hate speech. Our model is based on user behaviour and captures the two important factors, a) Toxicity exists as a spectrum, and b) Toxicity is not conserved. We divide the set of users can be into 3 categories: “amplifiers", who spread more hate than they receive; “attenuators", who spread less hate than they receive; and “copycats", who spread as much hate as they receive. We empirically establish the extent of spread; we call it the shift of the users into three different categories can have. We empirically validate the proposed model on both the simulated Barabási–Albert (BA) graphs of various sizes as well as samples of real worlds graphs from the data we studied. For BA graphs, we borrow the shifts and the distribution of users in three categories from the empirical observation. Our model is able to reproduce for both types of graphs the expected increase in toxicity, as well as the impact of the position of the three categories of users in the social network.

Chapter 2 of the thesis provides a comprehensive literature review, exploring previous work, such as using Spread Activation(SPA) modelling, Belief propagation, SIR models and their variants etc., to study the spread of hate. Chapter 3 outlines the research methodology, including information about the dataset, pre-processing, analysis of the dataset and our conclusion of the analysis. Given our learnings from the analysis, Chapter 4 presents a new approach to mapping the spread of hate speech. We propose a new model. In Chapter 5, we run the model on random graphs and test it out thoroughly. Finally, Chapter 6 provides a conclusion and a summary of the research findings and discusses some future work.

Chapter 2

Review of Literature

In this section, we comprehensively review some of the related literature. Our primary focus is to look at literature about the spread of hate speech on Twitter and how it propagates on a network. We have primarily cited two research studies in our work. They are spread Activation Modelling (SPA) and SIR (Susceptible - Infected - Recovered) models and their variants.

Spread Activation (SPA) is a method for modelling how specific energy (influence or information) spreads over a network of connected nodes (individuals/ groups). SPA captures the dynamics of information diffusion and the effects of the diffusion in the network [5]. Two things happen in the process: node activation and the spread amount. In each iterative step, there are a set of activation nodes. An active node will transfer its energy to its neighbour in the process of activating it. Each set of active nodes will start with an initial energy E, and at each successive step, a portion of the energy is transferred to its neighbours while retaining the remaining for itself. The total amount of energy in the network does not change at each iteration step. Hence the energy in SPA is conserved [6]. Ziegler and Lausen used SPA for trust propagation and propagated an algorithm through a network of agents based on the similarity between their interests and expertise. The authors found that this algorithm effectively evaluated local group trust relationships between individuals on the Semantic Web. They found that the algorithm performed well under different scenarios and provided reliable [7]. Nagar et al. presented a new approach to capturing the spread of hate on Twitter using SPA. The work shows how SPA and TopSPA perform better than baseline models for capturing the spread of hate. The study's empirical findings show that TopSPA and SPA perform better than standard approaches to detecting the spread of hate speech on Twitter. They specifically demonstrate that

TopSPA outperforms SPA in capturing the spread of various types of hate speech, as it incorporates topic-specific energy dissipation variables for each vertex. They use latent topic modelling to detect hate forms. The authors also discuss the characteristics of various types of hate speech and their network propagation methods [8].

SIR (Susceptible - Infected - Recovered) models are primarily used in epidemiological studies. These models divide the population into three groups; Susceptible (S), Infected (I) and Recovered (R). These models use ordinary differential equations (ODE) to show how an infectious disease spreads over time in a closed population. These models could also be modified to include other factors. A standard simple SIR model with its ODE is defined as the following[1]:-

$$\begin{aligned}\frac{dS}{dt} &= -\frac{\beta SI}{N}\\ \frac{dI}{dt} &= \frac{\beta SI}{N} - \gamma I\\ \frac{dR}{dt} &= \gamma I\end{aligned} \tag{2.1}$$

where $N = S + I + R$ in equation 2.1.

A study established a model for how information spreads through social networks. The simulation experiment results demonstrate that the proposed SEIR-based model can describe the information diffusion process over a social network [9]. Wang et al. showed that the emotional content of tweets serves as a crucial signpost for the information being forwarded by the recipient. They proposed the ESIS model, which is based on the SIS model and uses the proportion of information sent with an emotional quality as an edge weight. It outperformed the SIS model [10]. Other similar works, such as S-SEIR(Single Layer - SEIR) [11], SCIR (Susceptible Contacted Infected Removed) [12], irSIR (Infection Recovery SIR) [13], FSIR (Fractional SIR) [14] models also account for nuances of information spreading on social media platforms. Caldera et al., using time series, found out that the impact of hate speech is more and spreads faster [15].

Ribeiro et al. characterised users on Twitter; they compared hateful users with normal users. Contrary to popular belief, they find out that hateful users are actually at the centre of their social network [16]. This method, along with belief propagation, was used by Mathew et al. [17] to capture the spread of hate speech on Gab. Mathew et al. found that when compared to content created by normal users, hateful users

[1]`https://docs.idmod.org/projects/emod-generic/en/2.20_a/model-sir.html`

content tends to spread faster, farther, and to a larger audience. They generate a belief diffusion network where they initialise hateful users with energy 1 and all other users with 0. Then they use the DeGroots learning model to propagate beliefs through the network. The diffusion model identifies the users who are not hateful but have a high potential to be hateful using homophily. Hateful users are 0.67% of the total users and are highly connected to one another and produce close to 1/4th of the content on Gab.

Hate speech is a very complex problem; it brings attention to several crucial other concepts. Maity et al. studied incivility on Twitter and found a strong correlation between the acts of incivility and the difference of opinion between users involved [18]. Chandrasekharan et al. used Bag of Communities (BoC) to detect abusive content [19]. Gunasekara and Nejadgholi 2018 used an SVM classifier along with word embeddings, recurrent neural networks, and attention mechanisms to identify toxic language [20]. Sood et al. showed the flaws of profanity detection systems and how the systems don't consider the specific context and domain [21]. Chau et al. did a network topology analysis on blog hyperlinks (web pages) to identify hate communities [22]. Zhou et al. used the multidimensional scaling (MDS) algorithm to illustrate the proximity of hate websites and express the degree of similarity between them [23]. Zhang et al. used a deep neural network with graph convolutional networks; they found that this method outperformed 6 out of 7 datasets [24].

In recent works, Saha et al. looked at the spread of "fear speech" on Gab. They found that fear speech users gain more followers and occupy central positions in the network. Additionally, they can more effectively engage with normal users than hate-speech users [25]. Maarouf et al. found out that hateful content written by verified users is disproportionately more likely to go viral than content written by non-verified users [26]. Uyheng and Carley looked at Twitter conversations related to COVID-19 and proposed a dynamic network framework to characterise hate communities. They find that higher levels of community hate are consistently associated with smaller, more isolated, and highly hierarchical network clusters across both contexts [27]. Pérez et al. improved the performance of a transformer-based machine learning model by incorporating additional contextual information, leading to enhanced results [28]. Mnassri et al. implemented transformer-based language models such as BERT to improve classification results for hate detection [29]. By reviewing various relevant studies, we are provided with a certain foundation to work upon. Chapter 3 looks at hate speech on Twitter.

Chapter 3

Hate Speech on Twitter

3.1 Dataset and Preprocessing

We have used the Dataset published by Ribeiro et al. [30]. This Twitter dataset[1] contains 100,386 users and 19.58 million tweets. The majority of the tweets in the dataset are from the months of January 2017 to October 2017. The dataset also has a directed retweet graph $G = (V, E)$, where each node $u \in V$ represents a Twitter user. Each edge $(a, b) \in E$ represents a retweet in the network; there will be an edge from a to b if b retweets a. The retweet graph has $2,286,592$ edges. A study also shows that retweets graphs are better than follower-following graphs in analysing the influence of users [31]. Every tweet in the dataset is categorised into an original tweet, retweet or quoted tweet (retweet with additional text/ comment). The tweets are only text data, there is no image, video and audio data in the dataset. Using a random crawling algorithm, this dataset contains the 200 most recent tweets of 100,386 users. The creators of the dataset also label some users as hateful or normal; the labels are available for 4,972 users, out of which 544 users are labelled as hateful and the other as normal [30].

Before beginning our analysis, we assigned each tweet a toxicity score using the Perspective API[2]. All the 19.58 million tweets in our dataset have a toxicity score. *Perspective* is a machine learning algorithm that predicts the impact that a comment (text) could have on a conversation. By evaluating several attributes such as toxicity, severe toxicity, profanity, identity attack, threat, sexually explicit and insult, a score between the range of 0 to 1 is returned. The toxicity score helps us understand the extent of hatefulness in a tweet. This score is a probability score which reflects how

[1]`https://www.kaggle.com/datasets/manoelribeiro/hateful-users-on-twitter`
[2]`https://perspectiveapi.com/`

likely a reader perceives the comment. If the toxicity score is 0.8, this means that 8 out of 10 people will find this comment toxic. After assigning each tweet a toxicity score, we filtered the dataset to include the tweets from January 2017 to October 2017. We decided to study the spread of toxicity in this timeline. Table 3.1 shows the shape of the dataset before and after pre-processing.

Count	Before Preprocessing	After Preprocessing
Rows	19.58 M	17.22 M
Columns	24	26
Nodes (users) in graph	100,386	99,980
Edges in graph	2,286,592	2,272,251

Table 3.1: Shape of the Dataset and Graph; Before and After Pre-Processing

We analyzed the network properties of the graph to enhance our understanding of the structure of the graph. This gave us insight into how well and to what extent the nodes were connected with each other.

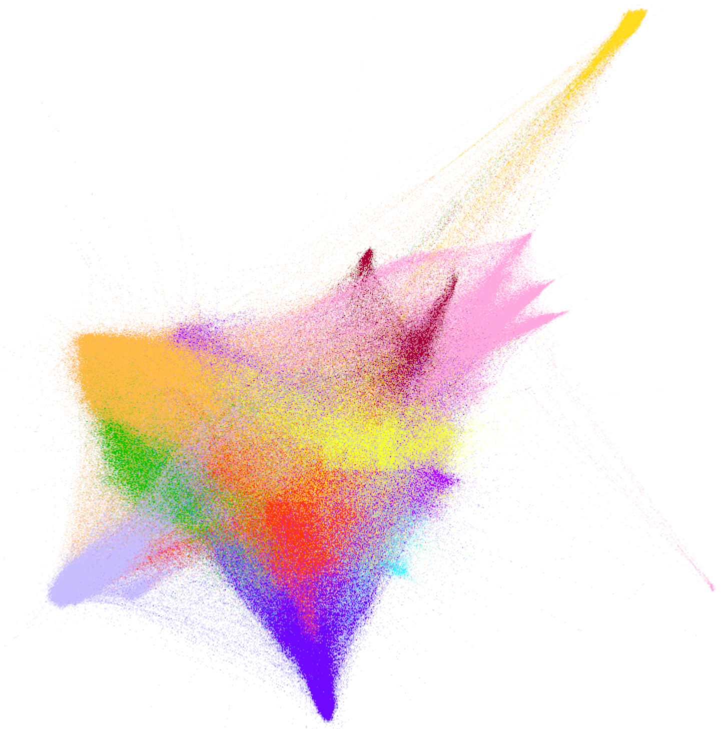


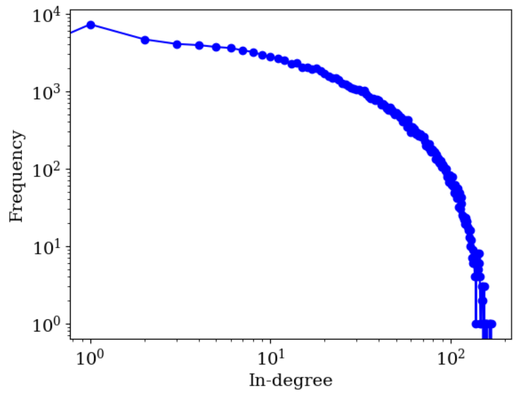
Figure 3-1: Twitter Re-Tweet Graph - Communities

Table 3.2 lists some of the important network properties of the graph. These network properties provide insights into the structural characteristics of the graph.

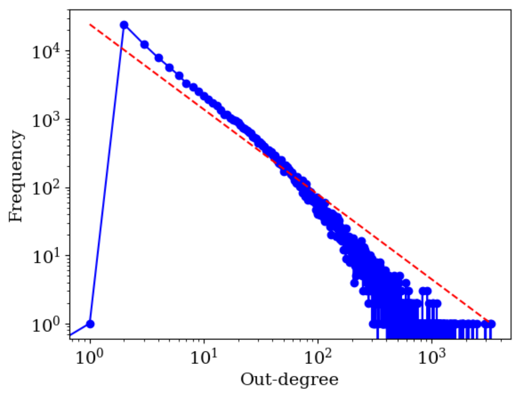
Network Properties	**Value**
Clustering Co-efficient	0.056
Modularity	0.62
No: of Communities	18
Assortativity	0.104
Bow-Tie Structure	S: 91,914, IN: 8,471, OUT: 1

Table 3.2: Network Properties of the Twitter Re-Tweet Graph

The clustering coefficient measures the degree to which nodes in a graph tend to cluster together. A value of 0.056 suggests that the graph exhibits a relatively low level of clustering, indicating that nodes are not strongly interconnected within local clusters. Modularity is a measure of the division of a graph into communities or modules. A value of 0.62 indicates relatively high modularity, suggesting that the graph has distinct communities or groups of nodes that are more densely connected within themselves. This was calculated using the Louvain community detection method in Gephi [32], and the number of communities is 18 in the graph. The preference of nodes to connect with other nodes with similar degrees is measured by Assortativity. 0.104 as a value indicates a slight positive assortativity. The bow-tie structure is a conceptual model used to understand the structure of large-scale networks. It consists of four components: the strongly connected component (S), the input component (IN), the output component (OUT), and tendrils. In this case, the strongly connected component (S) contains 91,914 nodes, the input component (IN) contains 8,471 nodes, and the output component (OUT) contains 1 node.



(a) In-Degree Distribution



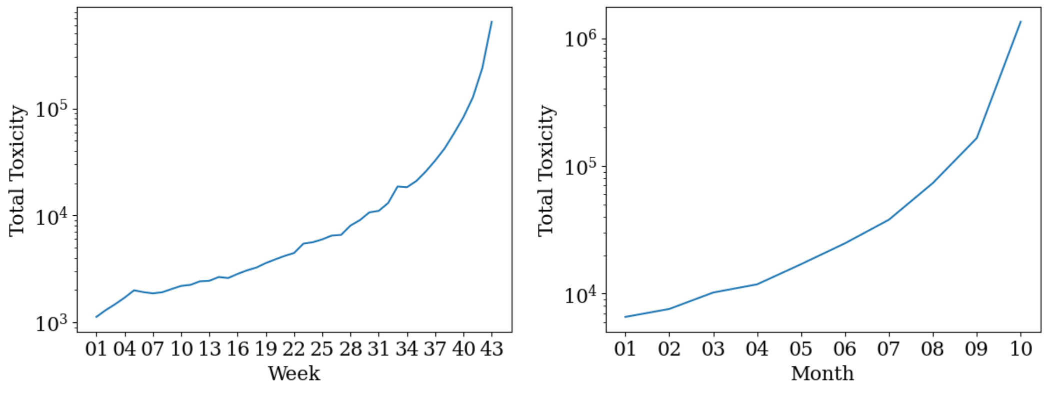
(b) Out-Degree Distribution

Figure 3-2: Degree Distribution of the Twitter Retweet Graph

Figure 3-2 shows the In-Degree and the Out-Degree distribution of the retweet graph plotted on a log-log scale. The in-degree distribution refers to the frequency of incoming edges to each node in the graph. In other words, it quantifies the number of connections directed towards a specific node. On the other hand, the out-degree distribution represents the frequency of outgoing edges from each node in the graph. It illustrates how many connections a particular node establishes with other nodes.

3.2 Analysing the Dataset

We start by looking at what is happening to toxicity over the timeline. The dataset consists of tweets collected across ten months (or forty-four weeks). Figure 3-3b shows the total toxicity across all the months, and Figure 3-3a shows the total toxicity across all the weeks. Both figures are plotted on a logarithmic scale on the y-axis for better visualisation.



(a) Total Toxicity Over the Weeks **(b) Total Toxicity Over the Months**

Figure 3-3: Total Toxicity Distribution over the Timeline

We can see that there is a rise in toxicity over time. This could happen because the number of users and tweets over time is increasing. Not all the users in the dataset are present in all months and weeks. To understand this further and eliminate the effect of users entering/leaving the system, we look into what is happening to the average toxicity of original tweets and re-tweets.

Figure 3-4 shows us what is happening to the average toxicity of all original tweets over all the weeks. We see an evident rise in toxicity. One thing from this is clear that the net toxicity after normalisation (in this case, the average) is increasing. We do the same analysis for retweets.

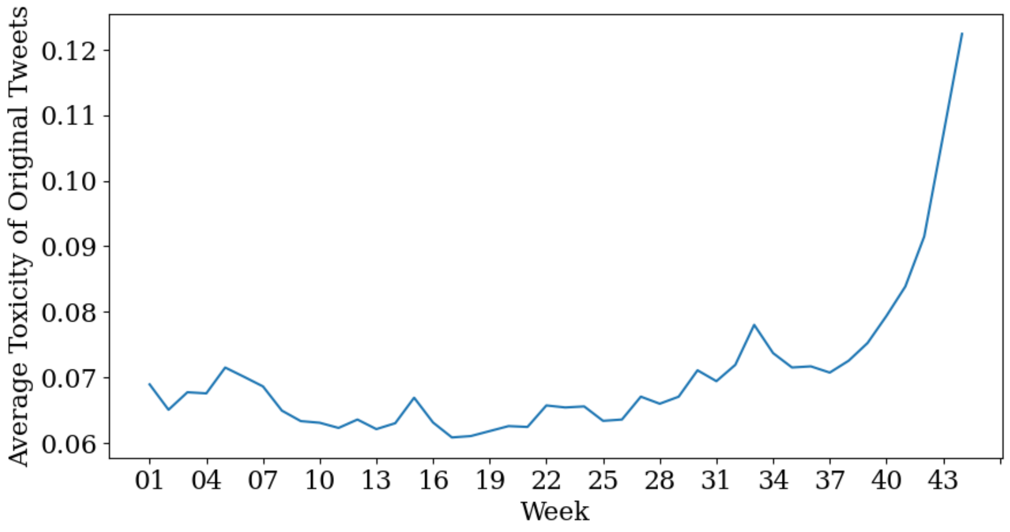


Figure 3-4: Average Toxicity of Original Tweets over the weeks

Figure 3-5 shows what is happening to the average toxicity of retweets over all the weeks. Here also we see an apparent increase in toxicity over time. We see that the highest value of average toxicity in retweets is more than that of original tweets. We see that in our dataset, retweets spread more hate. Evkoski et al. show that hateful tweets are retweeted more significantly than non-hateful tweets [33]. The analysis from our dataset can also back this. Figure 3-6 shows that tweets with higher toxicity values are retweeted significantly more than the tweets with low-toxicity values. While the count of high-toxicity tweets is also very less as compared to low-toxicity tweets

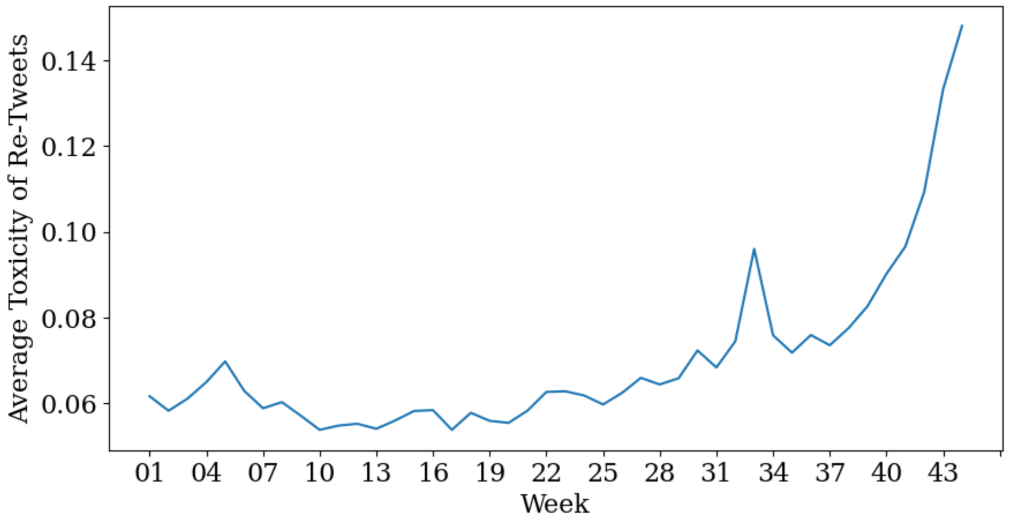


Figure 3-5: Average Toxicity of ReTweet over the weeks

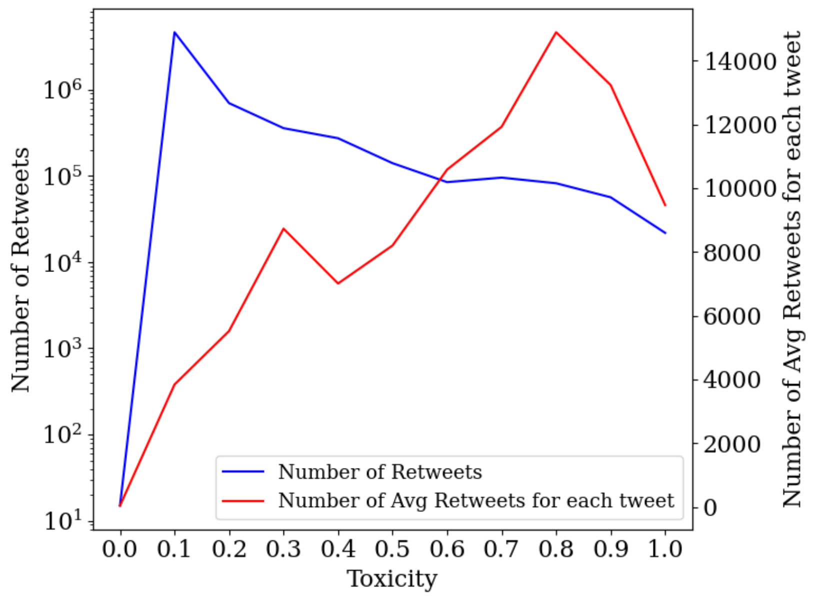


Figure 3-6: Number of Retweets and Average Retweets for each tweet as per Toxicity

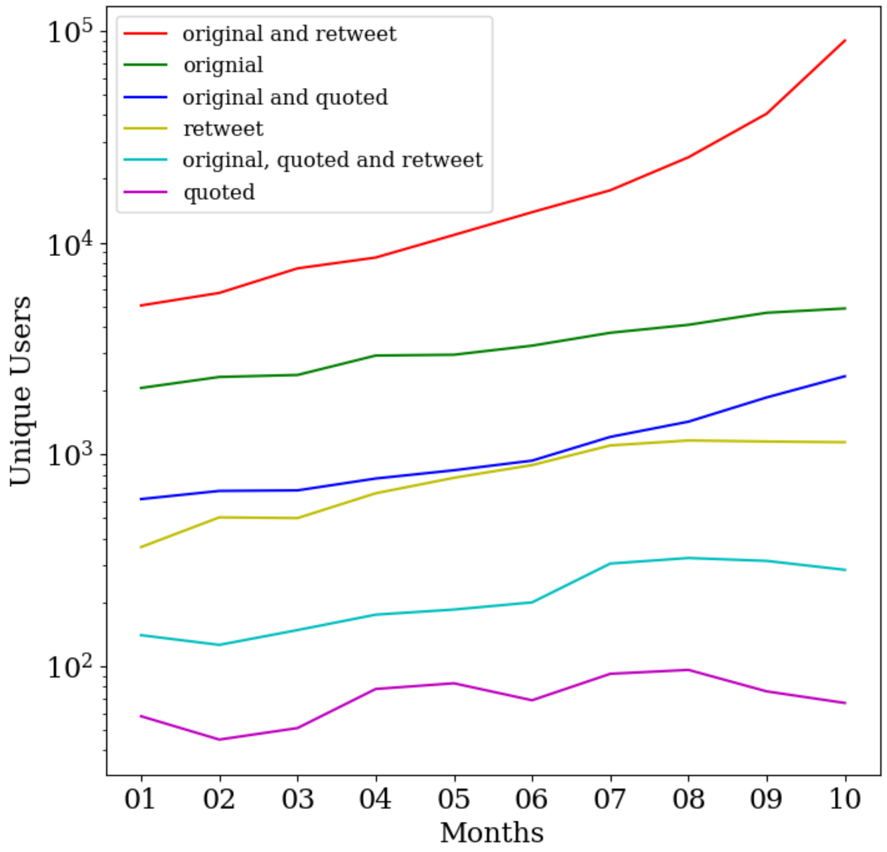


Figure 3-7: Number of Unique users in each Tweet Category

We now wanted to look at the distribution of users and the type of tweets (original, retweet and quoted) users tweeted. Figure 3-7 shows the number of unique users in a particular tweet category over all the months. By tweet category, we mean the various combinations of original tweets, retweets, and quoted tweets. For example, if a user has five tweets consisting of three original and two retweets, they would be counted in the "original and retweet" category. Similarly, if a user has ten tweets with five originals, four retweets, and one quoted tweet, they would fall into the "original, quoted, and retweet" category. Upon analysing Figure 3-7, we observed a significant increase over time in the number of users who shared a combination of original and retweeted tweets. The y-axis of the plot is displayed on a semilog scale to provide a clear visual representation of this trend.

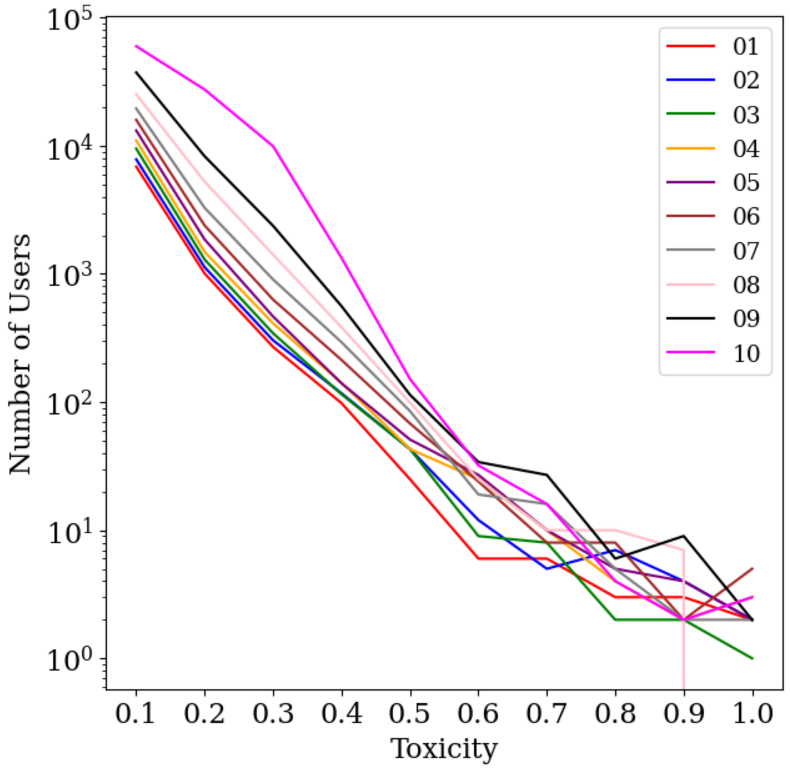


Figure 3-8: Number of Users and their Tweets Toxicity Levels

3.3 Conclusion of the Analysis

From our analysis of the data in section 3.2, we come to two important conclusions. They are as follows: -

1. **Energy (Hate) is not conserved.**

It is evident that energy (hate) is not conserved. The total toxicity and average toxicity per tweet consistently increase with time (weeks and months). This observation aligns with the findings depicted in Figures 3-3a,3-3b, 3-4, and 3-5, demonstrating an upward toxicity trend over time. Plots 3-4 and 3-5 provide further insight by revealing that the average toxicity increases over time. This shows that the SPA model is insufficient to capture the spread of hate.

2. **Users respond differently to tweets of different toxicities**

   Categorising a user as hateful or non-hateful involves assigning them a discrete state. This classification is typically determined by applying a threshold to the user's toxicity levels. If a user's toxicity surpasses the threshold, they are considered hateful; otherwise, they are categorised as non-hateful. Using discrete states in SIR models to represent the spread of hate proves insufficient to us, as it fails to consider non-hateful users' toxicity adequately. Furthermore, determining an appropriate threshold for classification is context-dependent and may vary across research studies, heavily relying on how toxicity classifications are assigned to tweets. From Figure 3-6, we see that users respond differently to tweets of different toxicities.

Based on these findings, the upcoming chapter will introduce a novel approach to effectively model the spread of hate speech on Twitter.

Chapter 4

A Non-Conservational Model for Hate Spread

With our learnings from Section 3.3, we proceed towards building a new model that captures the spread of hate speech. This chapter focuses on developing the model and provides simulations of the model using small, directed graphs as illustrative examples.

4.1 User Classification

We look at user behaviour and in order to understand the influence of the social network upon a user and vice-versa, we compare a user's average toxicity with the average toxicity of the user's *indegree* neighbours. Figure 4-1 shows us a scatter plot of the average toxicity of the user versus its neighbourhood. Since we are considering only the predecessors of every node, this plot consists of $92,847$ users. For visual clarity, we divide the plot into four quadrants as shown in 4-2.

The top right is quadrant I, and the remaining quadrants are sequentially in a clockwise direction. Users in quadrant III (bottom left) show similar average toxicity levels as their neighbours. We call such users as *copycats*, since their outgoing average toxicity closely mimics their incoming average toxicity. Users in quadrant II have higher average toxicity levels as compared to their incoming neighbours, so we call such users *amplifiers* since they amplify the toxicity they receive. Note that these users may also be the generators of toxic content. Users in quadrant IV have lower average toxicity compared to their incoming neighbours. We call these users *attenuators*.

Therefore we classify our users into three categories: amplifiers, attenuators, and copycats.

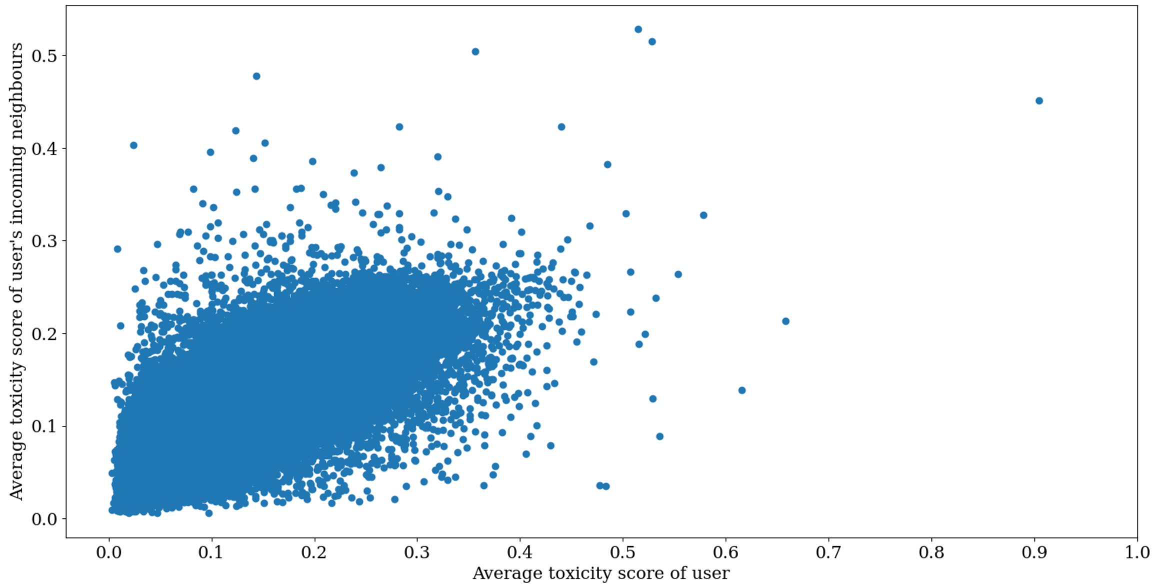


Figure 4-1: Average Toxicity of a User versus Average Toxicity of its In-Degree Neighbourhood

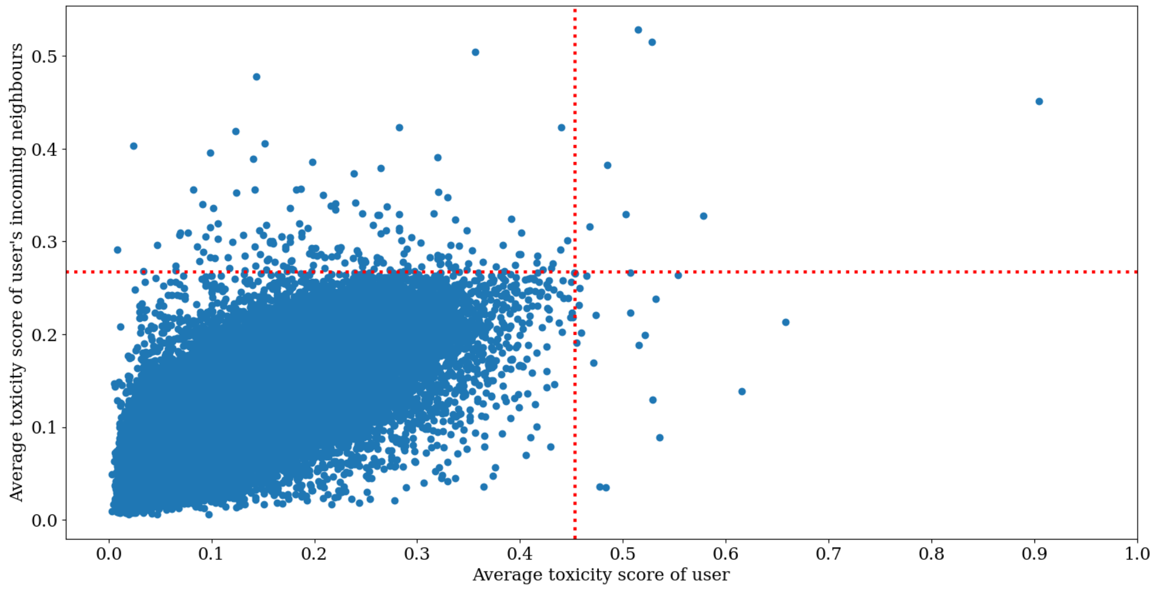


Figure 4-2: Average Toxicity of a User versus Average Toxicity of its In-Degree Neighbourhood - with quadrants

We then analyse the distribution of the difference between a user's average toxicity and the average toxicity of their neighbourhood. Figure 4-3 displays a plot representing the distribution of users based on the difference between their average toxicity and the average toxicity of their in-degree neighbourhoods. However, visual examination alone does not provide conclusive evidence regarding the normality of the distribution. Therefore, we employ statistical tests to determine its normality.

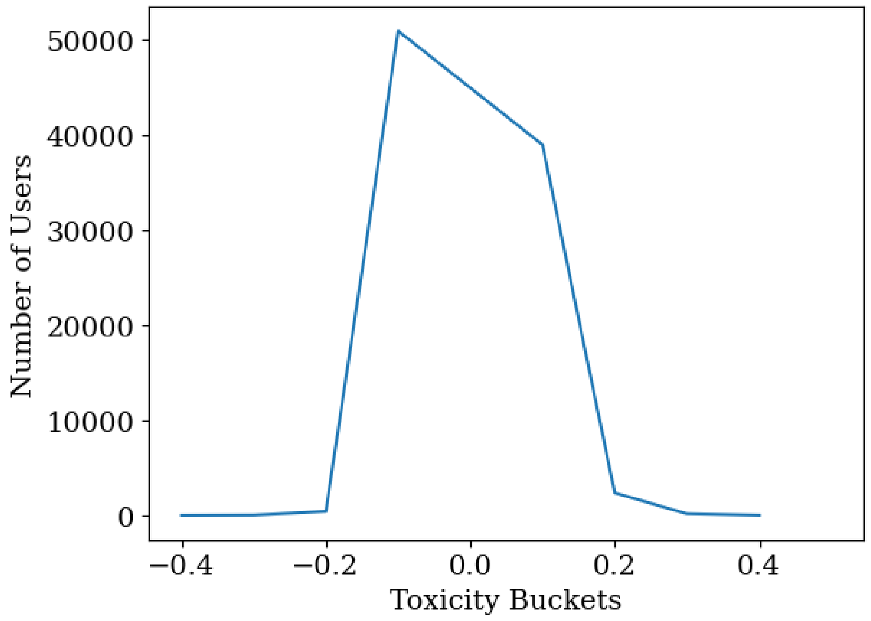


Figure 4-3: Distribution of the difference between a user's average toxicity and the average toxicity of their In-Degree Neighbourhood

First, we implemented the Shapiro-Wilk test [34] of normality, which showed that the distribution did not pass the test, suggesting it is not normally distributed. Then, we conducted the Kolmogorov-Smirnov test [35], which also yielded a negative result, further indicating that the distribution does not follow a normal distribution. In addition, we explored further divisions within the toxicity buckets. Currently, Figure 4-3 displays the toxicity intervals as 0.1. However, we also examined intervals of 0.01 and 0.001 and conducted normality tests on each of them. Despite the finer divisions, the distributions failed the normality tests, indicating that they did not follow a normal distribution.

In cases where a distribution deviates from normality, the Interquartile Range (IQR) proves to be a valuable measure. IQR is used when a distribution is not normally distributed as it is a reliable measure of variability unaffected by outliers or extreme values. It does so by dividing the data into equal parts or quartiles. The quartiles Q1, Q2 and Q3 are calculated by sorting the data in ascending order. Before beginning to model, we wanted to categorise all the users in the graph, i.e. find the amplifiers, attenuators and copycats. With our understanding from Figure 4-2, we tried to find out the outliers in the distribution of users and the difference between their average toxicity and the average toxicity of their in-degree neighbourhoods. To find outliers in data, IQR has a lower limit and an upper limit; all the values above the upper limit and below the lower limit are considered outliers. The lower and upper limits are defined as follows.

$$\begin{aligned} LowerLimit &= Q1 - 1.5 \times IQR \\ UpperLimit &= Q3 + 1.5 \times IQR \end{aligned} \tag{4.1}$$

By calculating the IQR, we obtained a value of 0.03809. This allowed us to determine the lower limit value, i.e. -0.07507, and the upper limit values, i.e. 0.07731, for identifying outliers within the dataset. With these limits established, we examined the dataset and found that out of the 99,860 users, a total of 6,252 users were identified as outliers. After this, we use the lower and upper limits to categorise amplifiers and attenuators. Of the 6,252 outliers, 4,954 were amplifiers, and 1,298 were attenuators. This means amplifiers account for 5.33% of the total users, while attenuators comprise 1.39% of the user population. All the remaining users were categorised as copycats.

Upon categorising the users, we determined the toxicity shifts for each category. These shifts indicate the amount of change in incoming toxicity that a user will generate and subsequently transmit as outgoing toxicity. The shifts are calculated by averaging the toxicity levels of all users within the respective category. We found that the shift caused by copycats is -0.000497, by amplifiers is $+0.1133$, and by attenuators is -0.1022. Table 4.1 presents the shifts and user proportions for each user category.

User Categories	**Toxicity Shift**	**User Proportion**
Amplifiers	+0.1133	5.33%
Attenuators	-0.1022	1.39%
Copycats	-0.000497	93.28%

Table 4.1: Toxicity Shifts and User Proportions for each User Category

4.2 Theoretical Formulation of the Proposed Model

We model the flow of toxicity per tweet. It takes into account all incoming toxicity (tweets) and applies the corresponding shift based on the user's category. The model then forwards the adjusted toxicity ahead. We created two versions of the model: a sum variant and an average variant. We proceeded with the sum variant due to its more conclusive results. In the sum variant, the model aggregates the toxicities of the tweets a user receives from its predecessors, applies its respective shift and subsequently sends it to its successors.

We make a few assumptions in order to define our model:

1. We consider the network to be static - users do not enter or leave the system.
2. Rather than compute each user's toxicity shift for each range of toxicity, we consider only averages.
3. A user's classification does not change with time.
4. A tweet is forwarded endlessly. The simulation stops if it reaches a node that has no further successors.

Given the assumptions, our model can still factor in the user and tweet fluctuation.

We consider a directed graph $G = (V, E)$ with N nodes and E edges, representing the network structure of our study. Each node in the graph $u \in V$ represents a user on the social media platform. The set $T = T_1, T_2, \ldots, T_K$ represents the timestamps at which we observe the spread of hate speech in the network; these could be days, weeks or months.

At the initial timestamp T_0, a specific node v posts a tweet with a toxicity value denoted as *tox* (a node could also post multiple tweets with different toxicity values). This toxicity value represents the degree of hatefulness of the tweet. The shift applied by each node in the network for the toxicity received will depend on its user category and can be denoted as *shift*(v). These shift values capture the individual propensity of nodes to amplify or suppress the toxicity of the content they receive. The total toxicity of the network at T_0 is *toxicity*(T_0) = *tox*. Each neighbour of node v will have a specific shift based on its user category denoted by *shift(nbr(v))*. This shift will be added to *tox* and the updated value of toxicity for that tweet will be *tox* = *tox* + *shift(nbr(v))*.

To compute the toxicity in the network at each subsequent timestamp, we employ the following algorithm, as depicted in Algorithm 1.

Algorithm 1 Compute Toxicity in a Twitter Re-Tweet Network

Input:

- G: A directed graph.
- $copyCatList, attenList, ampList$: Lists of nodes of respective user categories.
- $timestamp$: The number of time steps

Output:

- A dictionary of the toxicity levels of each node at each timestamp.

```
1: $values$: A dictionary mapping nodes to their toxicity values
2: $values[v] = tox$
3: for each timestamp $t$ in $T$ do
4:   $toxicity(t) \leftarrow \sum_{i=1}^{n} values[v_i],\ n \in$ number of nodes {Total toxicity at timestamp $t$}
5:   for node $v$ in $values$ that are not empty i.e. in the current timestamp do
6:     for successors of node $v$ do
7:       Let $v_{nbr}$ be the receiving neighbor of $v$
8:       if $v_{nbr}$ in a certain user category then
9:         $shift(nbr(v)) \leftarrow shift(v)$ {Check for user category; attenuators, amplifiers and copycats have their respective shifts}
10:      end if
11:      $tox \leftarrow tox + shift(nbr(v))$ {A neighbour who has received a tweet applies the shift to the received tweet and further forwards a tweet with updated toxicity}
12:      if $tox > 1$ then
13:        $tox \leftarrow 1$
14:      else if $tox < 0$ then
15:        $tox \leftarrow 0$
16:      end if
17:      $values[v_{nbr}] \leftarrow tox$
18:    end for
19:  end for
20:  $values[v] \leftarrow clear$ {clear the values of node v}
21: end for
```

4.3 Demonstration on an Example Graph

To gain a better understanding of the model described in algorithm 1, we demonstrate the model using a small graph. Let's look at Figure 4-4, which demonstrates the model using a small example of a directed graph.

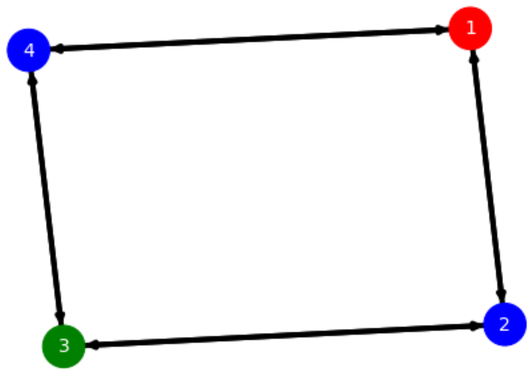


Figure 4-4: A small example graph.

In Figure 4-4, we just demonstrate the model on a small graph with four nodes. To simulate the model, we begin with Node 1 and initialise it with two tweets having a toxicity value of 0.9 and one tweet with a toxicity value of 1. The format for denoting the tweets toxicity and their count is *"toxicity value: count of tweets"*. As this is a demo example, we manually assign Node 1 as an amplifier, Node 2 and Node 4 as copycats, and Node 3 as an attenuator. Additionally, we define the toxicity shifts for amplifiers to be $+0.1$, for attenuators to be -0.2, and for copycats to be -0.1. Lets begin with the simulation, Node 1's outgoing toxicity is initialised as $0.9 : 1, 1 : 1$. This toxicity is subsequently received by Node 2 and Node 4, the successors of Node 1. These nodes receive the toxicity, apply their respective shifts, and pass it on to their successors.

Table 4.2 demonstrates the full simulation in detail. The final two columns show the sum of toxicity and average toxicity per user at each timestamp. We can easily observe that the total and average toxicity in the network is not conserved. The changes in the total toxicity also depend on the configuration of the copycats, attenuators and amplifiers.

Time	Node 1			Node 2			Node 3			Node 4			Total Toxicity	Average Toxicity per User
	In-Coming	Shift	Out-Going	In-Coming	Shift	Out-Going	In-Coming	Shift	Out-Going	In-Coming	Shift	Out-Going		
0			0.9:2, 1:1	0.9:2, 1:1						0.9:2, 1:1			2.8	0.933
					0.8:2, 0.9:1						0.8:2, 0.9:1			
1						0.8:2, 0.9:1	0.8:4, 0.9:2					0.8:2, 0.9:1	5	1.25
								0.6:4, 0.7:2						
2				0.6:4, 0.7:2					0.6:4, 0.7:2	0.6:4, 0.7:2			3.8	0.95
					0.5:4, 0.6:2						0.5:4, 0.6:2			
3	0.5:8, 0.6:4					0.5:4, 0.6:2						0.5:4, 0.6:2	6.4	1.6
		0.6:8, 0.7:4												
4			0.6:8, 0.7:4										7.2	1.8

Table 4.2: Demonstration of the model on an example graph as shown in Figure 4-4

Let's look at another such example to illustrate the model, and for simplicity, let's just focus on the outgoing toxicity. The small example graph is shown in Figure 4-5. We use the same parameters and shifts as used in Table 4.2. Node 1's outgoing toxicity is initialised as $0.9 : 1, 0.7 : 2$. Here too, we clearly see a rise in total toxicity and the average toxicity per user is not conserved. The simulation is demonstrated in Table 4.3.

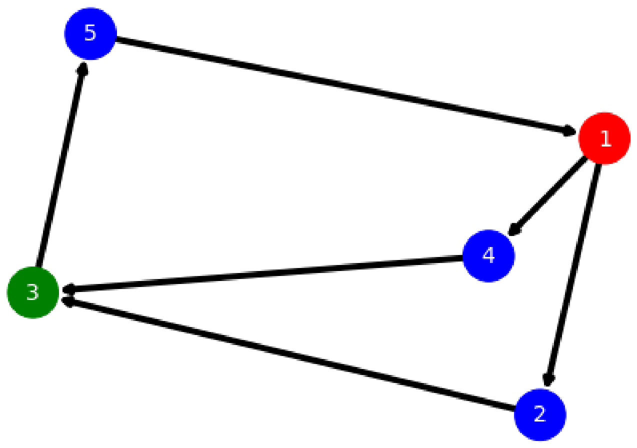


Figure 4-5: A small example graph. The blue nodes are the copycats, the green node is an attenuator, and the red node is an amplifier.

Time	Node 1	Node 2	Node 3	Node 4	Node 5	Total Toxicity	Average Toxicity Per User
0	0.9: 1, 0.7: 2					2.3	0.575
1		0.8: 1, 0.6: 2		0.8: 1, 0.6: 2		4	1
2			0.6: 2, 0.4: 4			2.8	0.7
3					0.5: 2, 0.3: 4	2.2	0.55
4	0.6: 2, 0.4					2.8	0.7

Table 4.3: Demonstration of the model on an example graph as shown in Figure 4-5

4.4 Amplifier, Attenuator and Copycat Characteristics

In this section, we analyse the characteristics of amplifiers, attenuators and copycats, both in terms of their network properties and their behaviour. Several natural questions arise. For example, *How does the average toxicity of an amplifier vary with its outdegree?, Do users display any kind of consistent behaviour?*

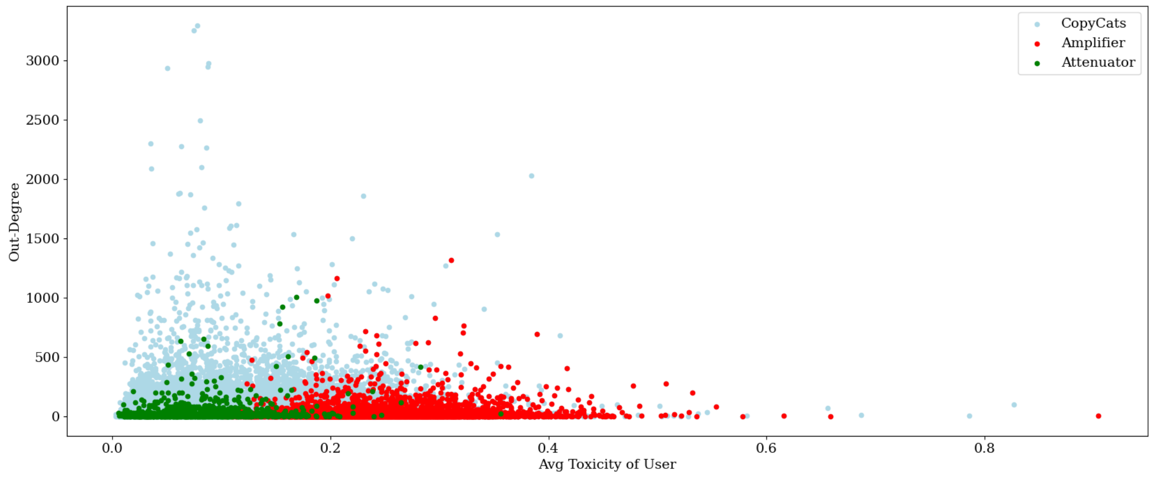


Figure 4-6: Average Toxicity of a User versus its Out-Degree

Figure 4-6 shows the scatterplot of the average toxicity of each node plotted against its outdegree for amplifiers, attenuators and copycats. We observe that the outdegrees of the copycats are among the highest. The average toxicities of the attenuators are less than those of the amplifiers, but for a given outdegree, there are more amplifiers than attenuators. There are quite a few copycats with large average toxicity but low outdegree.

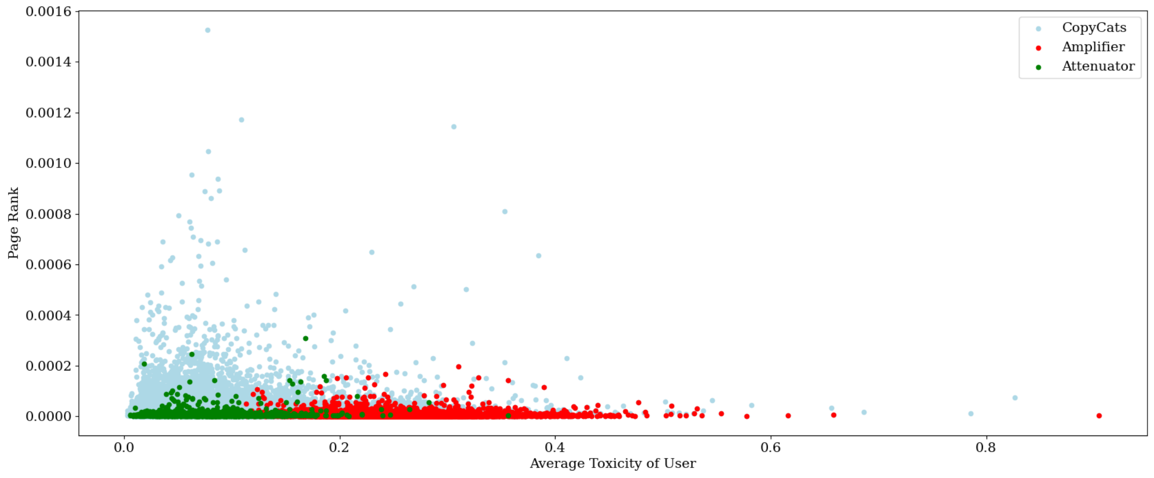


Figure 4-7: Average Toxicity of a User versus its Hub Value (PageRank on outdegrees)

Figure 4-7 shows a similar plot with hub values on the Y-axis. The highest hub values belong to the copycats, clearly highlighting their dominant role in the spread of toxicity.

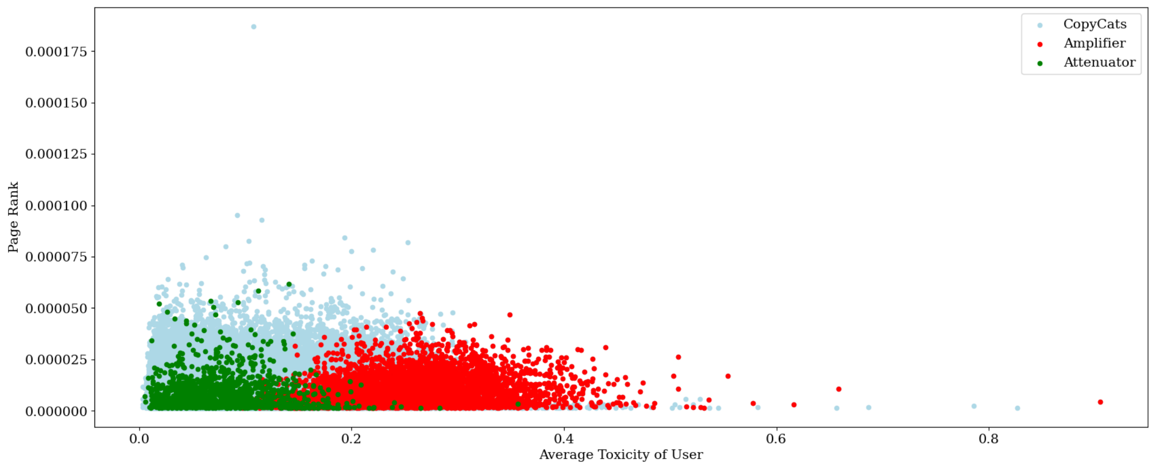


Figure 4-8: Average Toxicity of a User versus its PageRank - InDegree

Figure 4-8 shows a similar plot with PageRank on the Y-axis. The highest pagerank values belong to the copycats, indicating their importance as recipients of links.

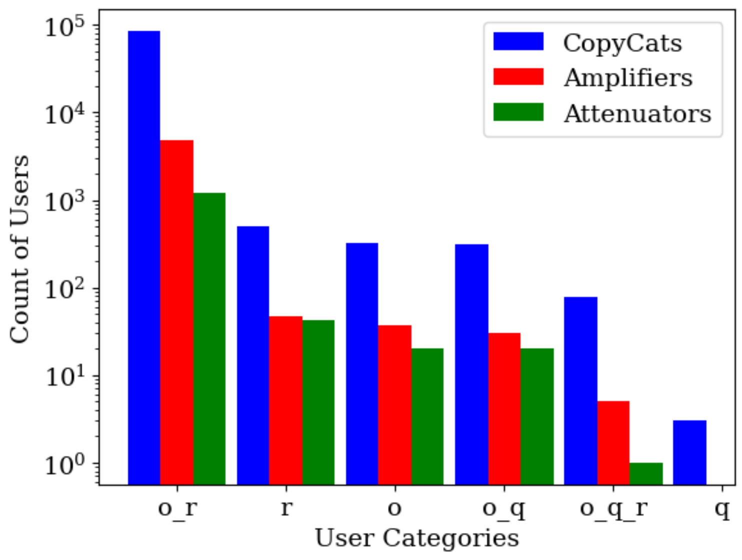


Figure 4-9: Tweet Distribution of each User Category

Figure 4-9 shows the number of users of each category active across the combinations of the actions of sending original tweets, retweets and quoted tweets. In each bucket, once again, we find that the copycats have an order of magnitude more number of users participating in the set of activities represented by the bucket. Further, note that the number of amplifiers who are sending original tweets and retweets is almost 5 times the number of attenuators indulging in the same two activities. The same is true of the amplifiers and attenuators doing all the 3 activities.

User Category	**Attenuator**	**Amplifier**	**CopyCats**
Attenuator	7.26e-18	0.02068	0.02844
Amplifier	0.02068	0	0.10937
CopyCats	0.02844	0.10937	0

Table 4.4: Attribute Assortativity Co-efficient for User Category

How are the amplifiers, attenuators and copycats distributed in the network? Are the amplifiers(attenuators) well-connected among themselves? Table **??** shows a lack of evidence for both homophily and inverse homophily for all types of users since all the values in the Table are close to 0. This suggests that the amplifiers, attenuators and copycats are almost randomly connected, without any preference or prejudice among each other.

Chapter 5

Results

To validate and test the model, we use Barabási–Albert (BA) graphs [36] as well as the sub-graphs sampled from our dataset.

BA graphs are used as testing grounds for network models due to multiple reasons, the most important being that they represent real-world networks closely. They are scale-free, meaning they have a characteristic of self-similarity across different scales. They also follow the power law of degree distribution, which means that there are few nodes with a high degree (more number of connections), and there are many nodes with a low degree (fewer connections). This property shows us the *rich-gets-richer* phenomenon seen in many real-world networks[1]. BA graphs are generated through a mechanism of growth and preferential attachment. Growth is a process of adding nodes to the network, and Preferential attachment is the probability that a link of the new node connects to an existing node i depending on the degree k_i as,

$$\Pi(k_i) = \frac{k_i}{\sum\limits_j k_j} \tag{5.1}$$

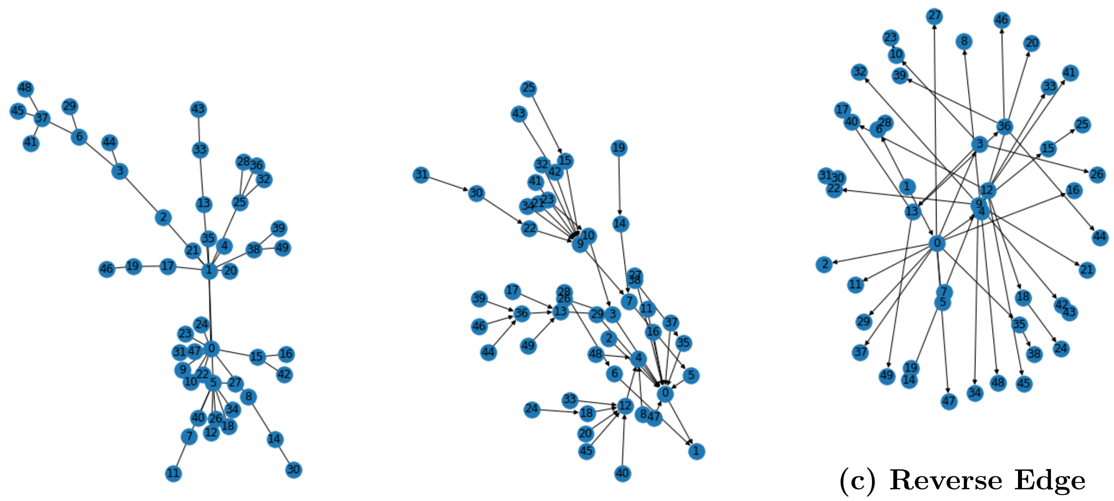
A new node is free to connect to any node in the network, whether it is a hub or has a single link, but as shown in equation 5.1, if given a choice between a node with a higher degree or a lower degree, it is twice likely to prefer the node with the higher degree [37].

Apart from this, we also experimented with Erdős–Rényi graphs [38, 39] and Watts–Strogatz graphs [40]. We could not get the desired results from these models, that's why we decided to proceed with BA graphs for testing the model. As we were

[1] `http://networksciencebook.com/chapter/5#barabasi-model`

looking for a random graph that closely mimics the Twitter re-tweet graph as well, providing a suitable random graph for our purposes.

We use the Python library NetwrokX[2] to generate the BA graphs. NetowrkX, by default, generates BA as undirected as seen in Figure 5-1a, but by nature, BA graphs should be directed. We tweak the NetworkX function to make the generation directed as shown in Figure 5-1b. And to make them resemble as close as the Twitter re-tweet network, we reverse the direction of the edges as displayed in Figure 5-1c.



(a) Undirected Graph **(b) Directed Graph** **(c) Reverse Edge Direction of Directed Graph**

Figure 5-1: Changes to the Barabási-Albert Graph (node size-50)

After this, to begin with testing, we use the toxicity shifts and user proportions (percentage of amplifiers, attenuators and copycats) we calculated from our dataset in Chapter 4 and as described in Table 4.1.

Through the experiments and testing, we aimed to address the following questions:

1. Can the proposed model be used to map the spread of toxicity on Barabási-Albert Graphs?
2. How does the proposed model behave when applied to the studied data and Barabási-Albert Graphs? Do the categorised users (attenuators, amplifiers, and copycats) behaviours studied from real data align with findings from Barabási-Albert Graphs and Re-Tweet graph?
3. Is there a temporal evolution of toxicity values?

[2]`https://networkx.org/`

In an attempt to answer the above questions, We tested the model on different sizes of the BA Graphs i.e. 5,000, 10,000, 25,000, and 50,000 nodes. In each case, we used an "m" value of 5, which determines the number of edges connecting a new node to the existing nodes. These graphs allowed us to test the model on a range of different networks that sufficient variation in size and scale. We kept the value of m constant to maintain consistency. To establish a starting point for simulating the model, we observed that the initial nodes created during BA graph generation often exhibited higher eccentricity. Hence, we selected the most suitable initial node for the simulation of the model on BA graphs. We began the simulation by initialising a node with one tweet of toxicity 0.0985. This value is the average toxicity of all the tweets in the dataset.

Subsequently, we simulated the model on the retweet graph present in the dataset. For these simulations, we focused on the sub-graphs that were similar to the node sizes used in the BA graphs. These sub-graphs were extracted from certain timelines (weeks and months) of the dataset. After that, for each respective sub-graph, we recalculate the toxicity shifts and identified the amplifiers, attenuators and copycats by the method we followed in chapter 4. To find a starting point for the simulation, we find the largest strongly connected component in the graph and selected a node with the highest eccentricity as our initial node. And for each subgraph, we calculate the average toxicity separately, and we initialise the starting node with one tweet of toxicity value as the average toxicity in that timeline.

5.1 Results on Barabási–Albert Graphs

With the changes and setting up of the BA graphs, we initially just simulated our model on different sizes of BA graphs. This didn't give us data to compare and validate the efficacy and effectiveness of the model. Therefore, we devised five scenarios of testing to help us assess better. In these scenarios, nodes are assigned as amplifiers, attenuators, and copycats to nodes of high and low degrees to see the effect of the user behaviour and model better. The distribution of these categories is maintained to match what is seen in the real data. The five scenarios are as follows:

1. **Case 1** - All nodes are randomly assigned a user category.
2. **Case 2** - Nodes with High Out-Degree are assigned as Attenuators, and the remaining nodes are randomly assigned.

3. **Case 3** - Nodes with High Out-Degree are assigned as Amplifiers, and the remaining nodes are randomly assigned.

4. **Case 4** - Nodes with Low Out-Degree are assigned as Attenuators, and the remaining nodes are randomly assigned.

5. **Case 5** - Nodes with Low Out-Degree are assigned as Amplifiers, and the remaining nodes are randomly assigned.

Table 5.1 shows the results of our model on different node sizes for all five case scenarios. The values are the averages of the highest value of total toxicity in a simulation. We record the average of five simulations for each case; through this, we obtain a more reliable and representative measure of the model's performance. The last time stamp in the table for each node size tells that the simulation has reached the end of the graph.

Nodes	**m**	**Edges**	**Time**	**Case 1**	**Case 2**	**Case 3**	**Case 4**	**Case 5**
5,000	5	24,846	2	2.45×10^2	5.81×10^1	6.3×10^2	3.12×10^2	2.21×10^2
			4	4.75×10^6	1.06×10^6	2.85×10^7	5.74×10^6	2.91×10^6
			6	2.11×10^7	5.83×10^6	1.23×10^8	2.4×10^7	1.14×10^7
			8	4.06×10^7	1.55×10^7	2.66×10^8	4.53×10^7	2.52×10^7
			36	8.26×10^7	2.46×10^7	$\mathbf{3.87 \times 10^8}$	6.66×10^7	3.31×10^7
10,000	5	49,784	2	3.94×10^2	8.2×10^1	1.01×10^3	3.72×10^2	3.59×10^2
			4	1.38×10^7	3.37×10^6	8.23×10^7	1.27×10^7	7.66×10^6
			6	6.89×10^7	1.78×10^7	3.69×10^8	6.34×10^7	3.28×10^7
			8	1.48×10^8	4.21×10^7	8.62×10^8	1.4×10^8	7.47×10^7
			40	5.31×10^8	1.48×10^8	$\mathbf{2.52 \times 10^9}$	4.09×10^8	2.32×10^8
25,000	5	124,812	2	5.81×10^2	8.78×10^1	1.55×10^3	5.7×10^2	4.95×10^2
			4	9.66×10^7	2.34×10^7	5.76×10^8	8.42×10^7	5.39×10^7
			6	6.11×10^8	1.54×10^8	3.56×10^9	7.08×10^8	3.15×10^8
			8	1.69×10^9	4.71×10^8	1.03×10^{10}	1.72×10^9	9.13×10^8
			46	5.81×10^9	1.93×10^9	$\mathbf{3.25 \times 10^{10}}$	5.64×10^9	2.25×10^9
50,000	5	249,772	2	3.49×10^3	3.8×10^2	1.34×10^4	3.48×10^3	3.02×10^3
			4	5.06×10^8	1.11×10^8	2.77×10^9	4.71×10^8	2.47×10^8
			6	3.35×10^9	9.03×10^8	1.92×10^{10}	3.14×10^9	1.7×10^9
			8	1.24×10^{10}	3.1×10^9	6.24×10^{10}	1.12×10^{10}	5.13×10^9
			52	4.85×10^{10}	1.25×10^{10}	$\mathbf{2.15 \times 10^{11}}$	4.86×10^{10}	1.5×10^{10}

Table 5.1: Average of Highest Value of Total Toxicity in all 5 cases when the model is simulated on BA Graphs. Note that Case 3, where the amplifiers have the highest outdegree results in the largest toxicity.

We clearly see the effect of the placement of attenuators and amplifiers on total toxicity. The change in toxicity also depends on the placement of attenuators, amplifiers and copycats. But we see that with time and graph size increasing, the total toxicity also increases. Attenuators exhibit their behaviour by reducing toxicity when assigned to high out-degree nodes, while amplifiers display their behaviour by increasing toxicity when assigned to high out-degree nodes.

Nodes	**m**	**Edges**	**Time**	**Case 1**	**Case 2**	**Case 3**	**Case 4**	**Case 5**
5,000	5	24,846	2	5.48×10^2	1.1×10^1	1.09×10^3	4.29×10^2	3.86×10^2
			4	1.66×10^4	1.10×10^1	6.82×10^4	1.49×10^4	1.28×10^4
			6	1.76×10^4	1.11×10^1	8.38×10^4	1.83×10^4	1.43×10^4
			8	1.81×10^4	1.11×10^1	9.06×10^4	2.12×10^4	1.46×10^4
			10	2.29×10^4	1.09×10^1	8.20×10^4	1.95×10^4	1.44×10^4
10,000	5	49,784	2	8.35×10^2	1.54×10^1	2.53×10^3	1.05×10^3	8.42×10^2
			4	4.54×10^4	1.49×10^1	2.13×10^5	4.71×10^4	3.67×10^4
			6	5.22×10^4	1.48×10^1	2.64×10^5	5.22×10^4	4.06×10^4
			8	6.03×10^4	1.50×10^1	2.59×10^5	5.47×10^4	3.91×10^4
			10	4.76×10^4	1.45×10^1	2.40×10^5	5.12×10^4	3.88×10^4
25,000	5	124,812	2	1.60×10^3	2.65×10^1	4.42×10^3	1.56×10^3	1.35×10^3
			4	1.32×10^5	2.71×10^1	7.01×10^5	1.39×10^5	1.05×10^5
			6	1.91×10^5	2.69×10^1	8.68×10^5	1.70×10^5	1.32×10^5
			8	$\mathbf{1.74 \times 10^5}$	$\mathbf{2.68 \times 10^1}$	$\mathbf{8.26 \times 10^5}$	$\mathbf{1.66 \times 10^5}$	$\mathbf{1.15 \times 10^5}$
			10	$\mathbf{1.72 \times 10^5}$	$\mathbf{2.67 \times 10^1}$	$\mathbf{8.39 \times 10^5}$	$\mathbf{1.55 \times 10^5}$	$\mathbf{1.30 \times 10^5}$
50,000	5	249,772	2	1.13×10^3	2.19×10^1	3.14×10^3	1.13×10^3	1.12×10^3
			4	3.06×10^5	2.20×10^1	1.65×10^6	3.52×10^5	2.36×10^5
			6	3.42×10^5	2.19×10^1	1.92×10^6	3.49×10^5	2.56×10^5
			8	$\mathbf{3.95 \times 10^5}$	$\mathbf{2.19 \times 10^1}$	$\mathbf{1.90 \times 10^6}$	$\mathbf{3.49 \times 10^5}$	$\mathbf{2.68 \times 10^5}$
			10	$\mathbf{3.28 \times 10^5}$	$\mathbf{2.20 \times 10^1}$	$\mathbf{1.85 \times 10^6}$	$\mathbf{3.65 \times 10^5}$	$\mathbf{2.58 \times 10^5}$

Table 5.2: Average of Highest Value of Total Toxicity in all 5 cases when the decay variation of model is simulated on BA Graphs. Note that there is very little difference in the toxicity values between 8 and 10 timesteps.

To make our simulations on the BA graphs even more real world like we make modifications to the existing model. We incorporate an information value for each tweet, ranging from [0-1], that gradually decays over time. Tweets are initialised with an information value of 1 for the simulation. The information value decreases by 0.1 at each time step during the simulation. The information value represents the probability of a tweet being forwarded in the simulation. Once the information value reaches zero, the tweet is removed from the system. It is important to note that, as a probability, a tweet could be deleted from the system before its information value reaches zero.

Table 5.2 shows the results of the modified model for different node sizes, with an average of 5 simulations recorded for each case on BA graphs. Here as well, we see the increase in total toxicity and the effect of the placement of attenuators, amplifiers and copycats in the network. Over time, the total toxicity doesn't increase by much, as tweets are deleted from the system.

Next, we simulate the model on the re-tweet network sub-graphs as described in Table 5.3. Here the five scenarios don't hold valid as the attenuators, amplifiers and copycats will always remain constant in a real-world re-tweet graph.

No: of Nodes	**No: of Edges**	**Time**	**Total Toxicity**
5,278	2,447	2	1.2×10^{-1}
		4	1.38×10^{1}
		6	5.34×10^{2}
		8	2.85×10^{4}
		7	5.38×10^{2}
10,262	8,071	2	5.74×10^{-2}
		4	1.34×10^{-1}
		6	2.71×10^{-1}
		8	3.34×10^{-1}
		9	3.58×10^{-1}
24,118	56,214	2	2.29×10^{-1}
		4	1.85×10^{-1}
		6	8.33×10^{-1}
		8	1.09×10^{1}
		44	5.08×10^{35}
51,358	429,639	2	1.4×10^{0}
		4	2.6×10^{2}
		6	5.18×10^{7}
		8	1.08×10^{14}
		11	2.24×10^{20}

Table 5.3: Highest Value of Total Toxicity when the model is simulated on Twitter Network Sub-Graphs

Here too, we evidently see that for each node size, there is a rise in total toxicity values. Over time we also see a high rise in toxicity; this is due to the presence of a high number of cyclic graphs of sizes 4 and 5. We removed all the self-loops and cyclic graphs of sizes 2 and 3.

Through these results, we clearly see that the proposed model captures the spread of toxicity on Barabási-Albert Graphs. As seen in the real data, we see that over time and with an increase in graph sizes, the total toxicity also increases. It also shows that toxicity values are temporal and change over time. The effect of user behaviours can also be seen in Barabási-Albert Graphs.

The code for the simulation is listed in Chapter 7, Appendix section of the work.

5.2 Discussion

In the dataset we used [30], 543 users out of 4,972 were labelled hateful using crowd-sourcing. *How does the categorisation of hateful vs non-hateful users relate to our approach of user categorisation into amplifiers, attenuators and copycats?*

These two approaches, we believe, are complementary. Of the 543 labelled hateful users, we found that 95 are amplifiers, 5 are attenuators, and 443 are copycats. This is not a contradiction. As seen through various plots in this work, the attenuators, amplifiers and copycats have average toxicity values in a wide range, suggesting that any of them may or may not be hateful. In future work, it may be interesting to consider both these axes as dimensions and analyse the 6 combinations: hateful amplifiers, hateful attenuators, hateful copycats, non-hateful amplifiers, non-hateful attenuators, and non-hateful copycats, and analyse their roles and behaviours in detail. Although the analysis in the current work suggests that there might be few (if any) hateful attenuators.

Chapter 6

Conclusion

We propose a novel model for capturing the spread of hate in a social network, taking into account two important factors: the existence of hatefulness on a spectrum and the non-conservation of hatefulness when treated as energy. Through empirical analysis and observations, we classify users into three distinct categories: Amplifiers, Attenuators, and Copycats. This categorisation allows us to model the spread of toxicity more effectively by considering how users amplify, suppress, or mimic the hatefulness they receive. To validate the efficacy of our proposed model, we conducted experiments on both simulated Barabási–Albert graphs and a real-world dataset, and our model successfully reproduces the increase in total toxicity and average toxicity observed in the empirical data. The proposed model also effectively reproduces the behaviours of the categorised users.

This model also raises many new questions. If more relevant data becomes available, then one might ask:

- What is the effect of users entering and leaving the system? How do we model it?

- How consistent are the shifts applied by the users in the 3 categories? In this paper, we calculated the average shift in each category of users. Do we need a more refined picture? Does an amplifier(/attenuator/copycat) apply the same shift to all levels of toxicity?

- Is this behaviour of a user constant? Over time, can an attenuator become an amplifier?

We hope that future work will address these and related interesting questions.

In summary, we propose a novel model to capture the spread of hate speech on Twitter. This model could help in creating strategies to mitigate hate speech on Twitter, as it offers a fresh perspective to capture its spread.

Chapter 7

Appendix

The Appendix section consists of the code used for the work.

Listing 7.1: Simulating and Capturing the Spread of Toxicity in a Graph

```
def simulate(G, copyCatList, attenList, ampList, timestamp,
    plotTotalSumofTox, plotAvgToxPerUser, plotTimestamp):
  """
  Simulates the spread of toxicity in a graph.

  Args:
    G: The graph.
    copyCatList: A list of nodes that are copycats.
    attenList: A list of nodes that are attenuators.
    ampList: A list of nodes that are amplifiers.
    timestamp: The number of timestamps to simulate.
    plotTotalSumofTox: A list to store the total sum of toxicity.
    plotAvgToxPerUser: A list to store the average toxicity per user.
    plotTimestamp: A list to store the timestamps.

  Returns:
    A dictionary of the toxicity levels of each node at each timestamp.
  """

  values = {node: {} for node in G.nodes()}
  values[1] = {tox_value: tweet_count}

  for t in range(timestamp):
    total_value = 0
    tweet_count = 0
    for node_values in values.values():
      for key, val in node_values.items():
        total_value += key * val
        tweet_count += val
    avgToxPerUser = total_value / len(values.keys())
    plotTotalSumofTox.append(total_value)
    plotAvgToxPerUser.append(avgToxPerUser)
    plotTimestamp.append(t)

    new_nodeList = []
    for node in G.nodes():
      if values[node] != {}:
          new_nodeList.append(node)

    for node in new_nodeList:
      for succ in G.successors(node):
          shift = None
          if succ in copyCatList:
              shift = toxicity_shifts['copyCatList']
          elif succ in ampList:
              shift = toxicity_shifts['ampList']
          elif succ in attenList:
              shift = toxicity_shifts['attenList']
          else:
              shift = 0

          for key, val in values[node].items():
              key_new = round(key - shift, 8)
              if key_new > 1:
                  key_new = 1
              elif key_new < 0:
                  key_new = 0
              values[succ][key_new] = values.get(succ, {}).get(key_new, 0) +
                  (val)
    values[node] = {}
```

Listing 7.2: Simulating and Capturing the Spread of Toxicity in a Graph - Decay Model

```
def simulate_decay(G, copyCatList, attenList, ampList, timestamp,
        plotTotalSumofTox = [], plotAvgToxPerUser = [], plotAvgToxPerTweet =
        [], plotTimestamp = []):
    values = {node: {} for node in G.nodes()}
    values[1] = {(tox_value, infromation_value) : tweet_count}

    for t in range(timestamp):
```

```
total_value = 0
tweet_count = 0
for node_values in values.values():
    for key, val in node_values.items():
        total_value += key[0] * val
        tweet_count += val
avgToxPerUser = total_value / len(values.keys())
# avgToxPerTweet = total_value / tweet_count

plotTotalSumofTox.append(round(total_value, 2))
plotAvgToxPerUser.append(round(avgToxPerUser, 2))
# plotAvgToxPerTweet.append(round(avgToxPerTweet, 2))
plotTimestamp.append(t)

print(f"{t}:␣{values},␣total␣sum␣of␣tox␣−␣{round(total_value,2)
    },␣avg␣tox␣per␣user␣−␣{round(avgToxPerUser,2)}")

new_nodeList = []
for node in G.nodes():
    if values[node]:
        new_nodeList.append(node)
for node in new_nodeList:
    keys_to_delete = []
    for succ in G.successors(node):
        shift = None
        if succ in copyCatList:
            shift = toxicity_shifts['copyCatList']
        elif succ in ampList:
            shift = toxicity_shifts['ampList']
        elif succ in attenList:
            shift = toxicity_shifts['attenList']
        else:
            shift = 0

        for key, val in values[node].items():
            # print(f"{key}, {val}")
            rand_val = random.random()
            if rand_val <= key[1]:
                key_new = (round(key[0] - shift, 2), round(key
                    [1]-0.1, 2))
                values[succ][key_new] = values.get(succ, {}).get
                    (key_new, 0) + (val)
            else:
                print(f"DELETING VALUE - {key} from {node}")
                keys_to_delete.append(key)

    for key in keys_to_delete:
        if key in values[node]:
            del values[node][key]

    # print(values)
    values[node] = {}
```

List of Algorithms

Bibliography

[1] M. Mirchandani, "Digital hatred, real violence: Majoritarian radicalisation and social media in india," *ORF Occasional Paper*, vol. 167, pp. 1–30, 2018.

[2] B. Mullen and J. M. Smyth, "Immigrant suicide rates as a function of ethnophaulisms: Hate speech predicts death," *Psychosomatic Medicine*, vol. 66, no. 3, pp. 343–348, 2004.

[3] A. Olteanu, C. Castillo, J. Boy, and K. Varshney, "The effect of extremist violence on hateful speech online," in *Proceedings of the international AAAI conference on web and social media*, vol. 12, no. 1, 2018.

[4] M. ElSherief, V. Kulkarni, D. Nguyen, W. Y. Wang, and E. Belding, "Hate lingo: A target-based linguistic analysis of hate speech in social media," in *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 12, no. 1, 2018.

[5] K. Dey, H. Lamba, S. Nagar, S. Gupta, and S. Kaushik, "Modeling topical information diffusion over microblog networks," in *Complex Networks and Their Applications VII: Volume 1 Proceedings The 7th International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2018 7*. Springer, 2019, pp. 353–364.

[6] K. Dasgupta, R. Singh, B. Viswanathan, D. Chakraborty, S. Mukherjea, A. A. Nanavati, and A. Joshi, "Social ties and their relevance to churn in mobile telecom networks," in *Proceedings of the 11th international conference on Extending database technology: Advances in database technology*, 2008, pp. 668–677.

[7] C.-N. Ziegler and G. Lausen, "Spreading activation models for trust propagation," in *IEEE International Conference on e-Technology, e-Commerce and e-Service, 2004. EEE'04. 2004*. IEEE, 2004, pp. 83–97.

[8] S. Nagar, S. Gupta, F. A. Barbhuiya, and K. Dey, "Capturing the spread of hate on twitter using spreading activation models," in *Complex Networks & Their Applications X: Volume 2, Proceedings of the Tenth International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2021 10*. Springer, 2022, pp. 15–27.

[9] C. Wang, X.-y. Yang, K. Xu, and J.-f. MA, "Seir-based model for the information spreading over sns," *Acta Electonica Sinica*, vol. 42, no. 11, p. 2325, 2014.

[10] Q. Wang, Z. Lin, Y. Jin, S. Cheng, and T. Yang, "Esis: emotion-based spreader–ignorant–stifler model for information diffusion," *Knowledge-based systems*, vol. 81, pp. 46–55, 2015.

[11] R. Xu, H. Li, and C. Xing, "Research on information dissemination model for social networking services," *Int. J. Comput. Sci. Appl*, vol. 2, pp. 1–6, 2013.

[12] X. Ding, "Research on propagation model of public opinion topics based on scir in microblogging," *Comput. Eng. Appl*, vol. 51, no. 8, pp. 20–26, 2015.

[13] J. Cannarella and J. A. Spechler, "Epidemiological modeling of online social network dynamics," *arXiv preprint arXiv:1401.4208*, 2014.

[14] L. Feng, Y. Hu, B. Li, H. E. Stanley, S. Havlin, and L. A. Braunstein, "Competing for attention in social media under information overload conditions," *PloS one*, vol. 10, no. 7, p. e0126090, 2015.

[15] H. Caldera, G. Meedin, and I. Perera, "Time series based trend analysis for hate speech in twitter during covid 19 pandemic," in *2020 20th International Conference on Advances in ICT for Emerging Regions (ICTer)*. IEEE, 2020, pp. 1–2.

[16] M. Ribeiro, P. Calais, Y. Santos, V. Almeida, and W. Meira Jr, "Characterizing and detecting hateful users on twitter," in *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 12, no. 1, 2018.

[17] B. Mathew, R. Dutt, P. Goyal, and A. Mukherjee, "Spread of hate speech in online social media," in *Proceedings of the 10th ACM conference on web science*, 2019, pp. 173–182.

[18] S. K. Maity, A. Chakraborty, P. Goyal, and A. Mukherjee, "Opinion conflicts: An effective route to detect incivility in twitter," *Proceedings of the ACM on Human-Computer Interaction*, vol. 2, no. CSCW, pp. 1–27, 2018.

[19] E. Chandrasekharan, M. Samory, A. Srinivasan, and E. Gilbert, "The bag of communities: Identifying abusive behavior online with preexisting internet data," in *Proceedings of the 2017 CHI conference on human factors in computing systems*, 2017, pp. 3175–3187.

[20] I. Gunasekara and I. Nejadgholi, "A review of standard text classification practices for multi-label toxicity identification of online content," in *Proceedings of the 2nd workshop on abusive language online (ALW2)*, 2018, pp. 21–25.

[21] S. Sood, J. Antin, and E. Churchill, "Profanity use in online communities," in *Proceedings of the SIGCHI conference on human factors in computing systems*, 2012, pp. 1481–1490.

[22] M. Chau and J. Xu, “Mining communities and their relationships in blogs: A study of online hate groups,” *International Journal of Human-Computer Studies*, vol. 65, no. 1, pp. 57–70, 2007.

[23] Y. Zhou, E. Reid, J. Qin, H. Chen, and G. Lai, “Us domestic extremist groups on the web: link and content analysis,” *IEEE intelligent systems*, vol. 20, no. 5, pp. 44–51, 2005.

[24] Z. Zhang, D. Robinson, and J. Tepper, “Detecting hate speech on twitter using a convolution-gru based deep neural network,” in *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15.* Springer, 2018, pp. 745–760.

[25] P. Saha, K. Garimella, N. K. Kalyan, S. K. Pandey, P. M. Meher, B. Mathew, and A. Mukherjee, “On the rise of fear speech in online social media,” *Proceedings of the National Academy of Sciences*, vol. 120, no. 11, p. e2212270120, 2023.

[26] A. Maarouf, N. Pröllochs, and S. Feuerriegel, “The virality of hate speech on social media,” *arXiv preprint arXiv:2210.13770*, 2022.

[27] J. Uyheng and K. M. Carley, “Characterizing network dynamics of online hate communities around the covid-19 pandemic,” *Applied Network Science*, vol. 6, pp. 1–21, 2021.

[28] J. M. Pérez, F. M. Luque, D. Zayat, M. Kondratzky, A. Moro, P. S. Serrati, J. Zajac, P. Miguel, N. Debandi, A. Gravano *et al.*, “Assessing the impact of contextual information in hate speech detection,” *IEEE Access*, vol. 11, pp. 30 575–30 590, 2023.

[29] K. Mnassri, P. Rajapaksha, R. Farahbakhsh, and N. Crespi, “Bert-based ensemble approaches for hate speech detection,” in *GLOBECOM 2022-2022 IEEE Global Communications Conference.* IEEE, 2022, pp. 4649–4654.

[30] M. H. Ribeiro, P. H. Calais, Y. A. Santos, V. A. Almeida, and W. Meira Jr, “" like sheep among wolves": Characterizing hateful users on twitter,” *arXiv preprint arXiv:1801.00317*, 2017.

[31] M. Cha, H. Haddadi, F. Benevenuto, and K. Gummadi, “Measuring user influence in twitter: The million follower fallacy,” *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 4, no. 1, pp. 10–17, May 2010. [Online]. Available: https://ojs.aaai.org/index.php/ICWSM/article/view/14033

[32] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, “Fast unfolding of communities in large networks,” *Journal of statistical mechanics: theory and experiment*, vol. 2008, no. 10, p. P10008, 2008.

[33] B. Evkoski, A. Pelicon, I. Mozetič, N. Ljubešić, and P. Kralj Novak, “Retweet communities reveal the main sources of hate speech,” *Plos one*, vol. 17, no. 3, p. e0265602, 2022.

[34] S. S. Shapiro and M. B. Wilk, “An analysis of variance test for normality (complete samples),” *Biometrika*, vol. 52, no. 3/4, pp. 591–611, 1965.

[35] F. J. Massey Jr, “The kolmogorov-smirnov test for goodness of fit,” *Journal of the American statistical Association*, vol. 46, no. 253, pp. 68–78, 1951.

[36] A.-L. Barabási and R. Albert, “Emergence of scaling in random networks,” *science*, vol. 286, no. 5439, pp. 509–512, 1999.

[37] A.-L. Parabasi, “Network science by albert-lászló barabási.”

[38] P. Erdős, A. Rényi *et al.*, “On the evolution of random graphs,” *Publ. Math. Inst. Hung. Acad. Sci*, vol. 5, no. 1, pp. 17–60, 1960.

[39] E. N. Gilbert, “Random graphs,” *The Annals of Mathematical Statistics*, vol. 30, no. 4, pp. 1141–1144, 1959.

[40] D. J. Watts and S. H. Strogatz, “Collective dynamics of ‘small-world’networks,” *nature*, vol. 393, no. 6684, pp. 440–442, 1998.