# ANOMALOUS BEHAVIOR ANALYSIS IN SOCIAL NETWORKS AND CONSUMER REVIEW WEBSITES

A Dissertation Presented to

the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

By

Santosh K C

December 2018

# ANOMALOUS BEHAVIOR ANALYSIS IN SOCIAL NETWORKS AND CONSUMER REVIEW WEBSITES

Santosh K C

APPROVED:

Arjun Mukherjee, Advisor
Department of Computer Science

Rakesh M. Verma
Department of Computer Science

Omprakash Gnawali
Department of Computer Science

Robert Bronk
Department of Computer and Information Systems

Dean, College of Natural Sciences and Mathematics

# Acknowledgements

I am very much grateful to Dr. Arjun Mukherjee for providing me this precious opportunity of pursuing doctoral studies under his guidance. I am very much thankful towards him in nurturing, directing and guiding me on each and every steps of PhD studies. It would have been impossible for me to come this far without his constant support. I appreciate all his contributions of time, ideas, funding that helped me to have a productive PhD experience.

I would also like to thank Dr. Omprakash Gnawali for his constant guidance on both social and research issues. He has always been a source of inspiration in the tough times of PhD. I would like to thank Dr. Rakesh M. Verma for his important feedbacks and time that helped me hone my research skills. I would like to thank Dr. Robert Bronk for his guidance and assistance that helped me learn more about the current problems of anomalies in the web.

My sincere gratitude goes to all my collaborators and friends: Fan Yang, Marjan Hosseinia, Prasha Shrestha, Sohan De Sarkar, Suman Kalyan Maity, Suraj Maharjan, and Yifan Zhang.

My greatest thanks goes to my family for the endless support throughout this process. Your love and care has always guided me to succeed.

# ANOMALOUS BEHAVIOR ANALYSIS IN SOCIAL NETWORKS AND CONSUMER REVIEW WEBSITES

---

An Abstract of a Dissertation

Presented to

the Faculty of the Department of Computer Science

University of Houston

---

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

---

By

Santosh K C

December 2018

# Abstract

Web and social media have been influencing every aspect of today's world, rendering a tremendous amount of data that requires new insights to know about the current society. The usage and dependence on these media has led to their active use by a small yet powerful group of users to sway the sentiment of people for selfish gains. To check the infiltration of these anomalous users, we face two challenges: (1) studying opinions to learn their behaviors, and (2) detecting opinion spam to reduce their effects. We study the behaviors of anomalous users in reviews collected on Yelp, Amazon and social data from Twitter. Using Yelp reviews we explore the temporal behaviors of spammers. Social spammers easily penetrate and are difficult to filter as they adapt to changing filtering algorithms. Using a Twitter dataset, we study their behaviors of success rate, fraudulence, and content posting activities. We uncover that successful spammers have a stronger friendship base and post an amalgam of spam and non-spam contents. We exploit the behaviors learned from Yelp and Twitter to generate spam detection algorithms. Our novel temporal features are instrumental in spam detection in consumer reivews, performing better than existing state-of-the-art approaches. We combine the content-based features and graph based approach embodying social relationships for spam detection in Twitter. Biased random walks and language models significantly improve the classification. We further characterize the review system in Amazon, a leading online marketplace. We use verified purchases as a popularity index to evaluate models of popularity prediction. We find that it is indeed possible to analyze behaviors and develop methods that perform well for anomaly detection in the web, even in these challenging situations.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Opinions of people are archived on the web. Various forms of platforms encompassing social networks, review websites, forums, new portals, and blogs are having an increasing impact on shaping the knowledge of people.

Social media are becoming so dominant that traditional media are blending themselves with the wave. For the majority of the population, these social media, such as Facebook, Twitter, etc., are the first sight of information. They are also important for the government or other agencies who want to know the status of society.

With the humongous growth of online marketing, we find consumer opinions as reviews of products and services constantly emerging as influential, and they are among the key factors driving e-commerce. Their increasing share in the market economy has led to a larger influence and importance in purchase and decision making. Reviews of consumer review websites, such as Amazon, Yelp, Tripadvisor, and others, shape the purchase or recommendation of the service or product. Positive reviews increase business while negative reviews adversely impact it.

Since the opinions in social media and consumer review websites are important in e-commerce, they are being exploited for selfish gains. We label these small subsets of users who want to sway the public opinion for personal benefits as *anomalous users*. These vile users are increasing their influence and feeding misleading information. Thus, it is very much important to detect them.

This work focuses on analyzing the behaviors of anomalous users so that they can be detected to minimize their influence. By learning the important behaviors that differ from normal users, we can characterize anomalous users and exploit their characteristics for detection. We explore the temporal aspects of spamming in Yelp, a leading consumer review websites, and detect spammers using time features [20]. We explore the social media behaviors of spammers and detect them using both their local features and their effect on the social network graph [22]. Inspired by demoting reviews on Yelp, we perform popularity prediction in standalone and competing scenarios in Amazon.

## 1.1 Research Challenges

There are two major research challenges to check the infiltration of anomalous users. The first challenge is to study the behaviors by mining the opinions in social media and consumer review websites. The second challenge is to detect them. We use temporal features and network analysis to generate models of spam detection.

### 1.1.1 Temporal Dynamics of Opinion Spamming in Yelp

Recently, the problem of opinion spam has been widespread and has attracted a lot of research attention. While the problem has been approached on a variety of dimensions, the temporal dynamics in which opinion spamming operates is unclear. Are there specific spamming policies that spammers employ? What kind of changes happen, with respect to the dynamics, to the truthful ratings on entities? How do buffered spamming operate for entities that need spamming to retain threshold popularity and reduced spamming for entities making better success? We analyze these questions in the light of time-series analysis on Yelp. Our analyses discover various temporal patterns and their relationships with the rate at which fake reviews are posted. Building on our analyses, we employ vector autoregression to predict the rate of deception across different spamming policies. Next, we explore the effect of filtered reviews on (long-term and imminent) future rating and popularity prediction of entities. Our results discover novel temporal dynamics of spamming which are intuitive, arguable, and also render confidence on Yelps filtering. Lastly, we leverage our discovered temporal patterns in deception detection. Experimental results on large-scale reviews show the effectiveness of our approach, which significantly improves the existing approaches.

We use the truthful and fake (spam) reviews of popular restaurants in Chicago from Yelp to characterize the dynamics of opinion spamming. We start by analyzing the time-series of fake ratings of each restaurant in our data. We notice similar patterns in the time-series of different restaurants that indicate presence of latent spamming policies/trends likely to be used by spammers. To uncover them, we employ spectral clustering [85] on the time series. Our analyses reveal that there exist three dominant trends of spam injection: early, mid, and late spamming across the restaurants in our data. For each restaurant in each of the three early, mid, and late spamming policies, we jointly characterize the time series of cumulative deceptive ratings with the time series of various other modalities (e.g., truthful like ratings, truthful dislike ratings, truthful review count, etc.). This yields interesting inferences that hint that deceptive like ratings (promotion spamming) is linked with different behavioral modalities of truthful reviews over time and the rating dynamics of truthful reviews can potentially determine the future deception rates for each restaurant.

To validate the relationship, we perform time-series correlation analysis. Cross correlation results show statistically significant correlations of time-series of truthful ratings (as covariate) with future deceptive like ratings (as response) confirming the previous result beyond mere coincidence across each of the three polices. It further reveals two interesting spamming trends: buffered and reduced spamming which reveal the adaptive spam injection rates for two kinds of restaurants: i) those that need spamming to retain threshold popularity, ii) others that are more successful and consequently in lesser need of spamming.

Upon characterizing the spamming patterns, we predict future deceptive like ratings on a restaurant using vector auto regression. The predictions, being decent, lead us to explore the question, "How well can one predict the future truthful popularity (# of reviews) and average rating of a restaurant in the presence of deceptive reviews?" Working using lasso regression and vector auto regression we develop models capable of long term and imminent future popularity/rating predictions. The analyses also facilitate indirect validation of Yelps filtering. Lastly, we leverage the discovered temporal dynamics to devise a suite of novel time-series features.

### 1.1.2 Anomaly Detection in Social Networks

Social media are increasing their influence tremendously. Twitter is one of the most popular platforms, where people post information in the form of tweets and share the tweets. Twitter is available from a wide range of web-enabled services to all people. The real time reflection of a society can thus be viewed in twitter. Celebrities, governments, politicians, and businesses are active in twitter to provide their updates and to listen to the views of the people. Thus, the bidirectional flow of information is high. The openness of online platforms and reliance on users enables the spammers to easily penetrate the platform and overwhelm the users with malicious intent and content. This work attempts to detect spammers in social networks using a case study of twitter.

Spammers in social networks constantly adapt to avoid detection. Moreover, they follow reflexive reciprocity [17, 82] (users following back when they are followed by someone to show courtesy) to establish social influence and act normal. It is thus becoming difficult for traditional spam detection methods to detect spammers. Such spammers have widespread impacts. There are several reports of an army of fake Twitter accounts[1] being used to troll [2] and promote political agendas[3]. Even US

[1]http://theatln.tc/2m8g3eA
[2]http://bzfd.it/2m8rlja
[3]http://bit.ly/2kJiMKu

President Donald Trump has been accused of having fake followers[4].

We present ENWalk, a framework that uses the content information to bias a random walk of the network and obtain the latent feature embedding of the nodes in the network. ENWalk generates biased random walks and uses them to maximize the likelihood of obtaining similar nodes in the neighborhood of the network. We study the twitter content dynamics that could be important to bias those random walks. We found that there are two types of spammers: follow-flood and vigilant. We found that success rate, activity window, fraudulence, and mentioning behaviors can be used to compare the equivalence of users in twitter. We calculate the network equivalence using these four behavioral features between pairs of nodes and try to bias the random walks with interaction proximity of the pair of nodes. Experimental results on a 17-million-user network from twitter show that the combination of behavioral features with the underlying network structure significantly outperform the existing state-of-the-art approaches for deception detection.

### 1.1.3 Popularity Prediction in Online Marketplace

As products are always competing with their competitors to seize the same target market, we anticipate such rivalry online. Potential rivalry could be one major source of spamming to defame a competing product. We therefore want to explore the dynamics of competition in the market. We used amazon review data and found two dominant trends viz. death (the competitor seizes the previous market leader in sales) and survival (where the previous leader recovers its market sales after being

[4]http://bit.ly/1ViorHd, http://53eig.ht/2kzrhfL

hit by the competitor). Using a labeled dataset of 800 leader-competitor pairs, we evaluate models that can characterize popularity and competition.

## 1.2 Summary of Contributions

For spam detection in consumer reviews, we analyze the temporal behaviors and use them to model spam detection. Experimental results show that the time-series features derived from our analyses significantly outperform the existing state-of-the-art approaches for deception detection, demonstrating a value of our analysis beyond mere characterization of temporal dynamics.

We analyze the node behaviors of spammer nodes in Twitter and employ these behaviors with network structure. Our model based on a biased random walk strategy performs better than the benchmark models.

We further characterize competition in Amazon reviews to learn the effects of competing products based on popularity.

## 1.3 Roadmap

The rest of this document is organized as follows: Chapter 2 discusses prior work related to our research and discusses either how our work builds upon those ideas or stands out from them. Chapter 3 introduces spamming in consumer review website Yelp. We perform various analyses on spamming and develop models based on temporal features for spam detection. In Chapter 4, we discuss the spamming in social network using datasets from Twitter. We analyze the spamming behaviors and exploit them to detect social spammers. Chapter 5 discusses our work on characterizing popularity in Amazon and using various models to predict popularity in standalone and competing environments. Finally, in Chapter 6, we summarize our work and present some potential future directions.

# Chapter 2

# Literature Review

Recent studies have uncovered the importance of opinion-rich contents on the web, such as online posts, reviews, comments, and discussions, are having greater social and economic impacts on consumers and companies alike compared to traditional media [58, 34]. Sentiments expressed in social media are influential in the economy of the country [2]. Due to the enormous economic incentive, spammers try to exploit the dissemination of online content [18].

## 2.1 Spam Detection in Consumer Review Websites

As positive/negative reviews can either enhance/defame products, the essence of truthful opinions is being misused by deceptive opinion spamming. First reported in

[18], opinion spam refers to deliberate attempts (e.g., writing fake reviews, giving unfair ratings) to promote/demote target products/services. Several high-profile cases of fake reviews have been reported in the news. While credit-card fraud is as low as 0.2%, opinion spam is prevalent [55] and it is estimated that up to 20% of online reviews could be fake [81]. The problem has also received significant research attention. Notable works include detecting individual spammers [33], group spammers [48, 49], and satirical news [84] using a variety of approaches such as rating behaviors [33], unexpected association rules [19], linguistic approaches [9, 32, 56], latent variable models [31, 46], semi-supervised methods [27, 28], etc. Other related works include identifying multiple aliases of the same author (sockpuppet) [72, 65, 64, 14], authorship attribution [70, 15], generic deception detection [54], and deceptive content detection in forums [5, 26]. For a comprehensive survey, see [45].

Prior to our work, there were two notable stand-of-the-art approaches for spam detection in consumer reviews. [56] developed and compare three approaches to detecting deceptive opinion spam based on works from psychology and computational linguistics. They ultimately develop a classifier that is nearly 90% accurate on their gold-standard opinion spam dataset. Based on feature analysis of the learned models, they additionally make several theoretical contributions, including revealing a relationship between deceptive opinions and imaginative writing. They used the following three n-gram feature sets, with the corresponding features lowercased and unstemmed: UNIGRAMS, BIGRAMS, TRIGRAMS. Another research work done by [52] introduced behavioral features in review spam detection. Behavioral features performed better than the n-gram features. They further performed theoretic analysis to uncover the precise psycholinguistic difference between AMT reviews and Yelp reviews (crowdsourced vs. commercial fake reviews). From the analysis, they postulate that Yelps filtering is reasonable and its filtering algorithm seems to be correlated with abnormal spamming behaviors.

While the above works have made important progresses, we still do not know the temporal dynamics that underpin the problem of opinion spam. How does opinion spamming operate on a daily basis? What are the dominant spamming policies? How do the spam injection rates vary upon increased/reduced popularity of entities? What factors are temporally correlated with opinion spamming? How effectively can we predict the long term/imminent future of popularity and average rating of an entity in the presence of deception and how accurately can future deception be predicted? In addition, we employ these learned behaviors for spam detection.

This work aims to answer these questions in the light of time-series analysis. We use Yelp as a target of our case-study as it is one of the largest online consumer review hosting sites for services (e.g., restaurants, hotels, etc.) in the commercial setting. The closest works to ours were attempted in the following studies. Authors in [8] explored temporal burstiness patterns in product reviews for singular spammer detection. The rationale was spammers writing only one review per id (sockpuppets) could be detected as they tend to appear in product review bursts. In [10], rating distributional divergence was used to identify review spam, and in [51] a hardness analysis of detection was presented based on real and pseudo fake reviews. Xie et al., [83] investigated temporal burstiness patterns in ratings of online stores. While

[50] explored detection using fully unsupervised generative models, in [29], spatio-temporal patterns on the geographical distribution of spammers, were explored using internal data (e.g., IP addresses, cookies, etc.). Although these works have looked into the temporal dimension of spamming, their focuses were mostly on detection as opposed to characterizing the very way opinion spamming works. They also did not explore the behaviors which are temporally correlated with spamming and future deception prediction, which are the core focuses of this work.

## 2.2 Spam Detection in Social Networks

There have been several works on spam detection in general, especially review spam [7], and opinion spam. However, in Twitter there are limited attempts. One of the earliest works was done by [1]. They manually labeled and trained a traditional classifier using the features extracted from user contents and behaviors. Lee et al. leveraged profile-based features and deployed social honeypots to detect new social spammers [25]. Stringhini et al. also studied spam detection using honey profiles [73]. Ghosh et al. studied the problem of link farming in Twitter [11]. They hypothesize acquiring followers not only increases the size of a user's direct audience, but also contributes to the perceived influence of the user, which in turn impacts the ranking of the user's tweets by search engines. They thus introduced a ranking methodology to penalize the link farmers.

Abuse of online social networks was studied in [77]. They identified an emerging marketplace of illegitimate programs operated by spammers that include Twitter account sellers, ad-based URL shorteners, and spam affiliate programs that help enable underground market diversification. Their results show that 77% of spam accounts identified by Twitter are suspended within one day of their first tweet. Because of these pressures, less than 9% of accounts form social relationships with regular Twitter users. Instead, 17% of accounts rely on hijacking trends, while 52% of accounts use unsolicited mentions to reach an audience. In spite of daily account attrition, they show how five spam campaigns controlling 145 thousand accounts combined are able to persist for months at a time, with each campaign enacting a unique spamming strategy. Surprisingly, three of these campaigns send spam directing visitors to reputable store fronts, blurring the line regarding what constitutes spam on social networks.

Campaign spams was studied on [88, 30]. To detect campaign spammers, [30] formulate the problem as a relational classification problem and solve it using typed Markov Random Fields (T-MRF), which is proposed as a generalization of the classic Markov Random Fields. [88] proposed a framework consisting of three steps: firstly linking accounts who post URLs for similar purposes, secondly extracting candidate campaigns which may exist for spam or promoting purpose and finally distinguishing their intents.

Skip-gram model [43] has been popular to learn the features from a large corpus of data. It inspired to establish an analogy for networks by representing a network as a document. Similar to document being an ordered sequence of words, we can create an ordered sequence of nodes from a network using sampling techniques. Our work is inspired by latent representation of vertices in a network: DeepWalk [62], LINE [76], and Node2vec [13]. DeepWalk [62] uses local information obtained from truncated random walks to learn latent representations by treating walks as the equivalent of sentences. They evaluate DeepWalks latent representations on several multi-label network classification tasks for social networks such as BlogCatalog, Flickr, and YouTube. LINE [76] learns the d-dimensional features into two phases: d/2 BFS-style simulations and another d/2 2-hop distant nodes. Node2vec [13] creates the ordered sequence simulating the BFS and DFS approaches. All these feature learning approaches dont use the data associated with node which are important to learn the behaviors of the nodes.

Previous researches in social network spam detection were either just content based or graph-based [36]. However, it is important to note that both the content and social network are equally important for spam detection. Thus we want to combine both pieces of information for better detection methods. We propose a novel method, ENWalk, which combines both the content and network structure.

## 2.3 Popularity Prediction in Consumer Review Websites

Our work employs time-series modeling of popularity using Amazon reviews, and in that regard is related to the works on mining of time-stamped data such as [60, 59, 69, 68, 80] and pattern discovery from time series [4, 57, 61, 79, 44]. Currently, there are two aspects of research: *volume prediction* and *competition modeling.*

### 2.3.1 Volume Prediction

There have been several works on analyzing user generated contents (comments, likes, views, shares, subscriptions) and modeling their volume. [78] analyzed volume of news comments using log-normal and negative binomial distributions. Political blog comments in context with the content was analyzed by [86]. [67] extracted features for involvement analysis in social networks. [71] used neural networks to predict comments in Facebook. While these are outstanding achievements, they fail to employ the effects of competition.

### 2.3.2 Competition Modeling

Competition is evident in almost every part of the web. Companies want popularity for business advantage. People want popularity to increase their social influence in social networks. Thus, either explicitly or implicitly, there is competition all the time. Hence, researchers are modeling competition. While linear approaches are successful for time series mining such as auto-regression, autoregressive integrated moving average (ARIMA), Kalman filters, linear dynamical systems, they are not very suitable to characterize popularity in the online world that is inherently non-linear in nature. Non-linear methods [38, 39, 12, 37, 41, 66, 7, 63, 53, 74, 40, 23] are more popular as they can model the non-linear behaviors in the nature and are closer to our approach. However, these works did not explore the latent features embeddings of time series based on running window in standalone and competition environment which is the focus of this work.

# Chapter 3

# Temporal Dynamics of Opinion Spamming in Yelp

S. KC, and A. Mukherjee. On the temporal dynamics of opinion spamming: Case studies on yelp. *Proceedings of the 25th International Conference on World Wide Web*, pages 369-379, 2016. [20]

In this chapter, we study the temporal behaviors of spamming in Yelp. We explore the spamming patterns and allied temporal dynamics. We evaluate prediction capabilities of temporal features on popularity and ratings. Lastly, we leverage these time-series features to devise a novel suite of features for spam detection which outperform the state-of-the-art approaches.

## 3.1 Yelp as a Reference Dataset

Despite opinion spamming being prevalent [55], there are not many commercial websites that filter fake/deceptive reviews. Yelp is an exception and implements review filtering on a commercial scale. The filter has been in place for over a decade now and maintained by its dedicated anti-fraud team [51]. Although Yelp's filter may not be perfect, it is important to note that unlike other forms of generic Web spam (e.g., link[72], email[6], blog[26], etc.) that are relatively easier to detect, opinion spam is harder and usually requires a lot of internal signals [29, 48], and thus industrial opinion spam filters (e.g., Yelp) have a unique advantage. Thus, unlike previous small scale studies in [32, 48, 56], it is not possible to do large-scale analysis upon relying on data tagged by human experts or solicited ground truths fake reviews using Amazon Mechanical Turk. Even expert-annotation cannot fully eliminate the possibility of any noise. Also obtaining ground truths in the given domain and commercial setting is only possible upon spammer confessions or sting operations [35] which again cannot be performed at large scale. Thus, Yelp's filter although may not be perfect, nevertheless it provides us a unique opportunity to understand the dynamics of spamming at large-scale in the commercial setting. In fact, there have been studies that put Yelps filtering methods to test and have found it to be reasonably reliable [35, 52]. Hence, we choose Yelp as a reference dataset for characterizing the dynamics of opinion spamming.

As demonstrated by our experiments, we will see that the spamming patterns discovered are arguable, intuitive, and further pave the way for indirectly validating

Table 3.1: Yelp Dataset Statistics.

| | Deceptive | Truthful |
|---|---:|---:|
| # of dislike (1-3) reviews | 1630 | 10042 |
| # of like (4-5) reviews | 4465 | 30652 |
| # of reviews | 6095 | 40694 |
| % of reviews | 13.03% | 86.97% |
| # of reviewers | 5359 | 21761 |

Yelp's filtering.

We use the Yelp dataset in [52] of 70 popular Chicago restaurants over a 5 year time span (see Table 3.1). The reviews filtered by Yelp are considered deceptive (fake/spam) while others as truthful. We refer to reviews with 1-3 ratings as exhibiting dislike whereas reviews with 4-5 ratings exhibiting like connotations. Although opinion spamming can take both promotion/demotion flavors (by injecting deceptive like/dislike reviews), our pilot studies revealed that majority ( 75%) of the spam is focused on promotion as opposed to demotion. Hence we focus on promotion spamming. The next section lays the foundation for analyzing the dynamics of promotion spamming.

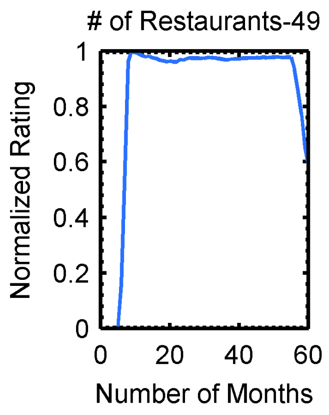
## 3.2 Determining Dominant Spamming Policies

Although opinion spamming can be interleaved throughout the entire lifespan of an entity, characterizing the dominant spamming patterns over time is the first step in understanding the dynamics of spamming. To find the number of promotion spamming policies, for each restaurant, we compute its time series of average cumulative rating of deceptive like (positive fake) reviews. The cumulative rating was computed for each time step by averaging the like fake ratings on that restaurant from start till that time step. The time series was further normalized and scaled to the range [0, 1] to gauge the relative promotion dynamics and facilitate time-series clustering on shape dynamics. The rationale here is that such a cumulative deceptive rating time series can quantify how spamming grew and faded over time for that restaurant. For each restaurant, its time-series starts at the date of the first review and continues until 60 months from the start. Time-series of all restaurants were aligned by pivoting on their starting time-step.

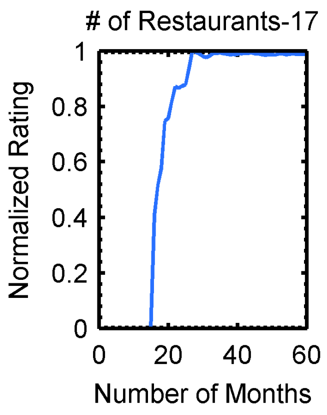
We hypothesize that there exist commonalities in spamming trends (policies) that exist across different restaurants. To characterize these spamming policies, we employed time-series clustering that can discover similar shapes in the deceptive rating time series of restaurants. We used the K-spectral Centroid (K-SC) time-series clustering algorithm in [85]. The distance function of K-SC is invariant to scaling and translation, which is particularly suited to our domain in capturing similarities in spamming across restaurants with varying popularity (review volume) and different launch dates. Its distance measure, $d(x, y)$ for two time-series $x$, $y$ is calculated as:

$$d(x, y) = min_{\alpha,q} \frac{||x - \alpha y_{(q)}||}{||x||} \tag{3.1}$$

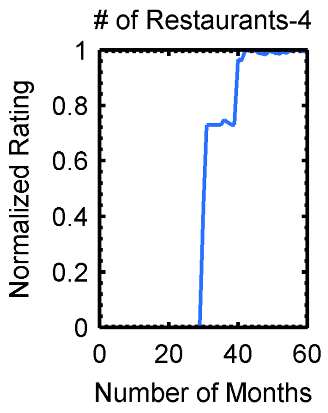
where $y_(q)$ is the result of shifting time series $y$ by $q$ time units, $||.||$ is the $L_2$ norm and $\alpha$ is the scaling coefficient to match the shape of two time series. Apart from clustering time-series having similar temporal patterns, it also yields the cluster



(a) Cluster 1

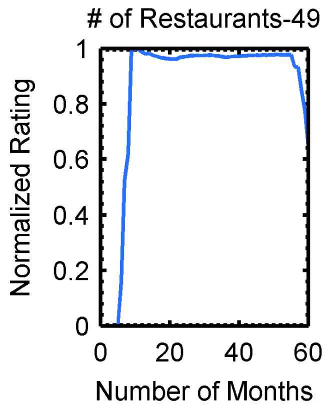


(b) Cluster 2

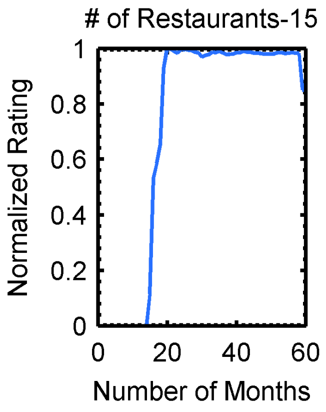


(c) Cluster 3

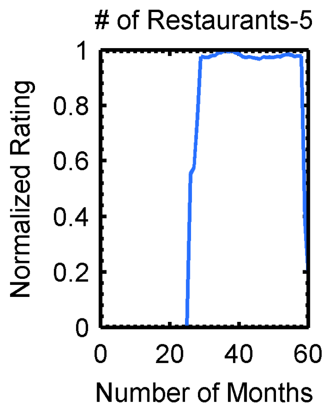
Figure 3.1: Time series cluster centroids with K=3 for cumulative rating of deceptive like reviews



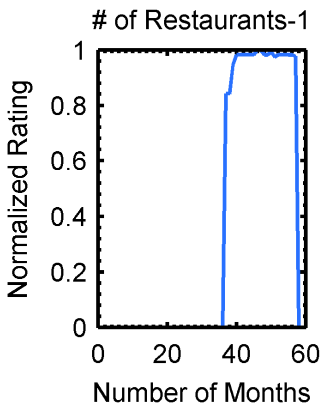
(a) Cluster 1



(b) Cluster 2



(c) Cluster 3



(d) Cluster 4

Figure 3.2: Time series cluster centroids with K=4 for cumulative rating of deceptive like reviews

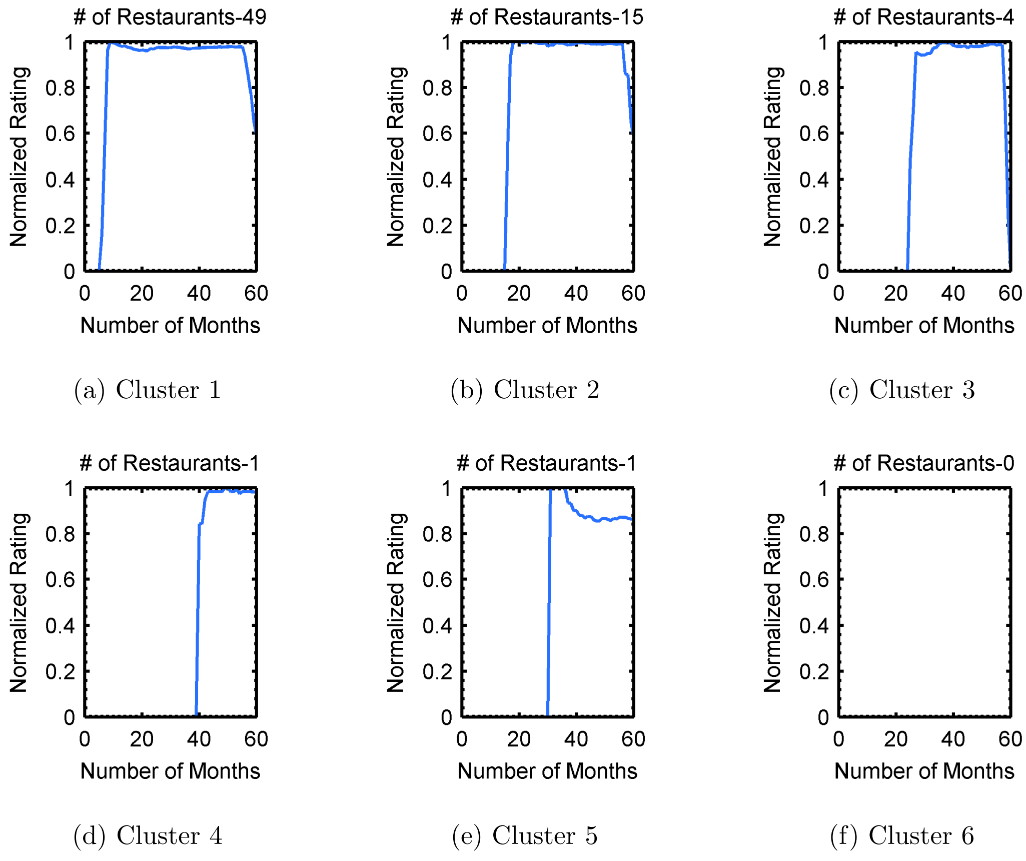


Figure 3.3: Time series cluster centroids with K=6 for cumulative rating of deceptive like reviews

centroid time-series for each cluster that is representative of that cluster.

We clustered the deceptive like rating time-series of restaurants in our data using the K-SC algorithm. As the dominant number of spamming policies are unknown, we explored different values for K. Fig. 3.1, 3.2 and 3.3 show the time-series centroid plots for K=3,4,6 respectively. The centroid plot header also reports the number of restaurants for that cluster. We note that for K=6, cluster 6 (Fig. 3.3.f) is empty. Out of the remaining five clusters for K=6, cluster 1 (Fig. 3.3.a) has similar shape to cluster 1 (Fig. 3.2.a) in K=4 and cluster 1 (Fig. 3.1.a) in K=3 and all three have the same 49 restaurants. Cluster 2 (Fig. 3.3.b) in K=6 and cluster 2 (Fig. 3.2.b) in K=4 are quite similar due to their starting spikes and profile. Cluster 3 (Fig. 3.3.c) in K=6 is similar to cluster 3 (Fig. 3.2.c) as both plummet in the right end and have similar starting spikes. Further, cluster 5 (Fig 3.3.e) in K=6 and cluster 4 in K=4 (Fig. 3.2.d) also have similar profiles. We see that the similar profiles are getting merged as K is reduced. Cluster 1 is same across all three values of K. Fifteen cluster 2 and two cluster 3 restaurants in K=6 and K=4 are clustered in cluster 2 in K=3. Remaining restaurants from K=6 (two from cluster 3 and one each from cluster 4, 5) and K=4 (three from cluster 3 and one from cluster 4) are merged to cluster 3 in K=3. Thus, we clearly see that there are three dominant promotion spamming policies corresponding to representative cluster centroids. We now explain each spamming policy using the plots for K=3.

Cluster 1 of K=3 (Fig. 3.1.a) refers to early spamming where the representative centroid shows steady spamming beyond the 5 months of launch. Although centroid has a zero till the fifth month, the deceptive like reviews of restaurants in the early spam cluster gradually build up a momentum from their inception. They tend to maintain continuous spamming until the end as depicted by the profile of cluster 1. The second cluster centroid (Fig. 3.1.b) refers to mid spamming policy where spamming is a bit delayed and starts rising only after the 14th month (after more than a year). On average, it takes about 10 more months to have the peak rating of 1 after gradual improvement in spamming. The third spamming policy (Fig. 3.1.c) starts rather late, stalls for 10 months before attaining the peak in deceptive like ratings. Only few restaurants exhibited such late spamming. Thus, we find three dominant spamming policies prevalent in restaurant promotion. The next section evaluates each spamming policy by assessing it in tandem with other behavioral modalities.

## 3.3 Dynamics of Spamming Policies

To study the dynamics of the three promotion spamming policies, we generated the time series of the ten modalities as follows:

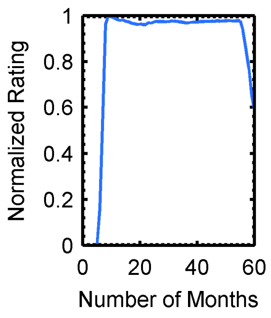
1. Number of fake dislike reviews
2. Cumulative rating of fake dislike reviews
3. Number of non-fake dislike reviews
4. Cumulative rating of non-fake dislike reviews
5. Number of fake like reviews

6. Cumulative rating of fake like reviews
7. Number of non-fake like reviews
8. Cumulative rating of non-fake like reviews
9. Cumulative rating of n-fake reviews
10. Cumulative rating of non-fake reviews

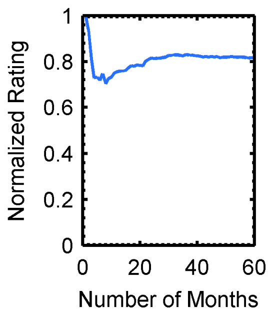
For each of the three spamming policies, we grouped all restaurants belonging to a policy and computed an additional set of normalized time-series on the ten modalities. For each behavioral modality, we further employed time-series clustering of restaurants in a given policy (cluster) and chose the dominant sub-cluster of that modality in a given policy. We now explain the three spamming policies based on the centroids of the dominant sub-clusters of relevant behavioral modalities for a policy.

### 3.3.1 Early Spamming

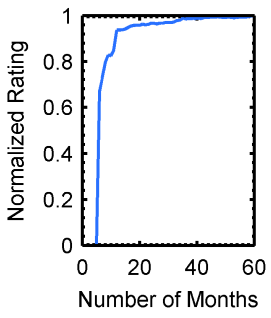
Fig. 3.4.a shows the reference centroid plot for this pattern (Fig. 3.1.a) where 49 out of 70 restaurants employ this policy. The restaurants employing early spamming wait for the truthful reviews for the initial period of around five months. Then they start spamming as shown in Fig. 3.4.a. It is interesting to note that the average truthful rating is seen rapidly dropping in the initial months (Fig. 3.4.b.) There is also a rapid increase in the truthful dislike rating (Fig. 3.4.c) and increase in the count of such dislike reviews (Fig. 3.4.d) till the fifth month. Though, the truthful like



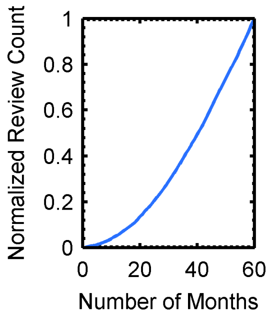
(a) Early Spamming



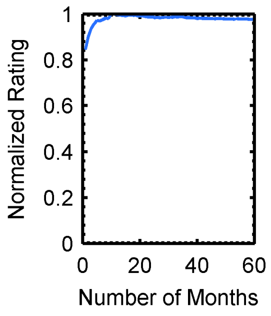
(b) Truthful avg. rating



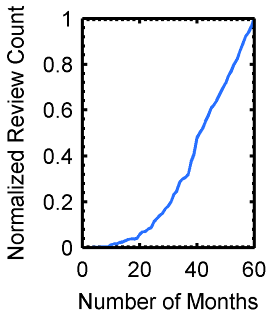
(c) Truthful dislike rating



(d) # of truthful dislike

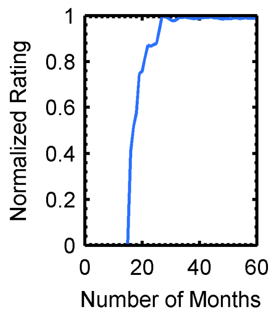


(e) Truthful like rating

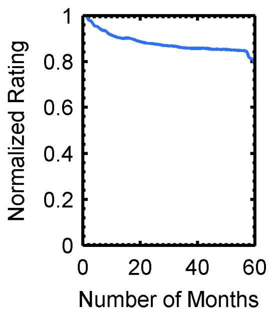


(f) # of deceptive like

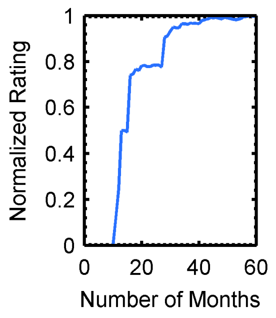
Figure 3.4: Normalized average cumulative rating and review count (#) of early spamming policy



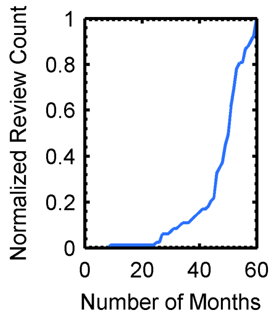
(a) Mid Spamming



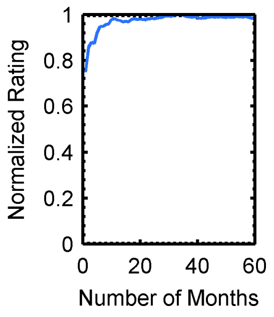
(b) Truthful avg. rating



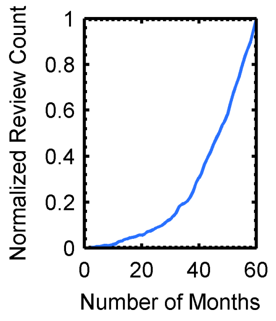
(c) Truthful dislike rating



(d) # of truthful dislike



(e) Truthful like rating



(f) # of deceptive like

Figure 3.5: Normalized average cumulative rating and review count (#) of early mid policy

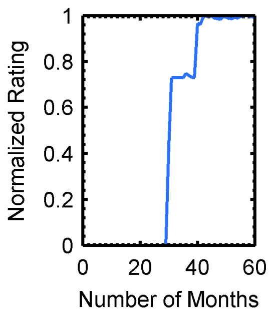
rating is constant as shown in Fig. 3.4.e, the restaurants inject more spam to check the influence of the truthful dislikes. This explains the sustained rate of deceptive like fake ratings (Fig. 3.4.a) with increase in the count of deceptive like reviews (Fig. 3.4.f). Thus, in early spamming, the influx of deceptive reviews starts early and maintains a steady promotion spamming rate to balance the truthful dislike influence.

### 3.3.2 Mid Spamming

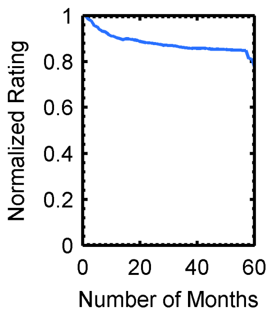
We find 17 restaurants employing mid spamming as shown by the reference centroid plot in Fig. 3.5.a. These restaurants dont exhibit spam injection until the 14th month. However, truthful average rating keeps on dropping rapidly till about 11th month (Fig. 3.5.b). It is worth noting that spamming picks up momentum after 14th month (Fig. 3.5.a). In the same time, truthful dislike rating rapidly increases from 10th month onward (Fig. 3.5.c) along with the increase in the number of truthful dislike reviews (Fig. 3.5.d). The truthful like rating is not affected (Fig. 3.5.e). So, the number of deceptive like ratings increases after 14th month (Fig. 3.5.f) and as if counteracts the increase in the truthful dislike ratings (Fig. 3.5.c). This clearly shows that deceptive like rating (spam injection) is almost in tandem with truthful average and dislike ratings.

### 3.3.3 Late Spamming

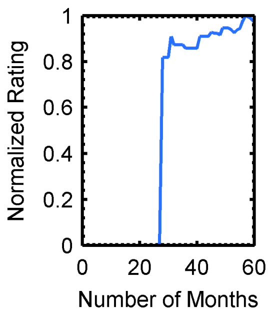
The late spamming pattern was found in 4 restaurants only. Fig. 3.6.a shows the reference centroid plot. These restaurants start promotion spamming only after the 30th month (Fig 3.6.a). Interestingly, truthful average rating (Fig 3.6.b) is seen monotonically decreasing. After 28th month, there is rapid increase in the truthful dislike rating (Fig. 3.6.c) caused by the soaring of truthful dislike reviews (Fig. 3.6.d). Since, the truthful like rating rate does not differ much (Fig 3.6.e), promotion spamming seems to be carried out after 30th month (Fig 3.6.f) to check the influx of the truthful dislike reviews. We also note that after a slight decrease in truthful



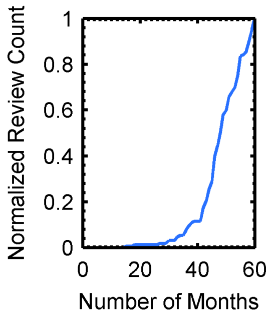
(a) Late Spamming



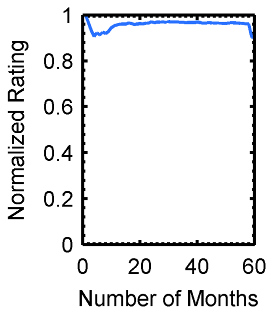
(b) Truthful avg. rating



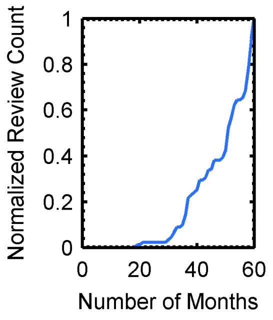
(c) Truthful dislike rating



(d) # of truthful dislike



(e) Truthful like rating



(f) # of deceptive like

Figure 3.6: Normalized average cumulative rating and review count (#) of early late policy

dislike rating, it increases again after 40th month (Fig. 3.6.c). Interestingly enough, we find the restaurants increase spamming after 40th month (Fig. 3.6.a) as if to bring an equilibrium with the dislike ratings spike. This phenomenon of increasing spam injection being tightly connected with the dynamics of dislike ratings across all policies can be inferred as buffered spamming.

## 3.4 Causal Modeling of Deceptive Ratings

In this section, we aim to characterize the plausible causes of spamming by comparing the time series of deceptive like ratings with the truthful like/dislike ratings. We use week as time interval for the time series in this section. We first explore the potential causes that are forerunners of deceptive ratings using cross correlation. Next, we encode the potential causal time-series in a vector autoregressive framework to forecast future deceptive ratings.

### 3.4.1 Time Series Causal Analysis Framework

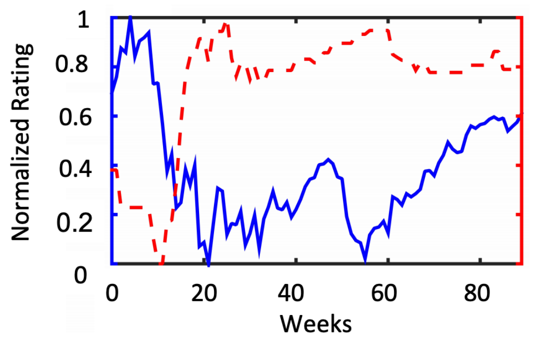
From the spamming dynamics explored in previous sections, it reflects the intuition that variations in truthful review ratings has a certain influence in the dynamics of deceptive like ratings. To discover the relationship between truthful ratings and deceptive like ratings (promotion spams) of restaurants, we consider their respective cumulative time-series. To validate the relationship, we extracted three segments of truthful ratings time-series (overall truthful average, truthful like, and truthful dislike) and compared them against the deceptive like rating time series of individual restaurants. To discover potential causality, we analyzed their cross-correlation (CCF) at different time lags. CCF at lag $k$ estimates the relationship between a response $Y(t)$, and a covariate $X(t)$ time-series at different time-steps shifted by $k$ time units and is given by:

$$CCF(k) = \frac{\sum_i ((X(i) - \mu_X)(Y(i+k) - \mu_Y))}{\sqrt{\sum_i (X(i) - \mu_X)^2}\sqrt{\sum_i (Y(i+k) - \mu_Y)^2}} \tag{3.2}$$

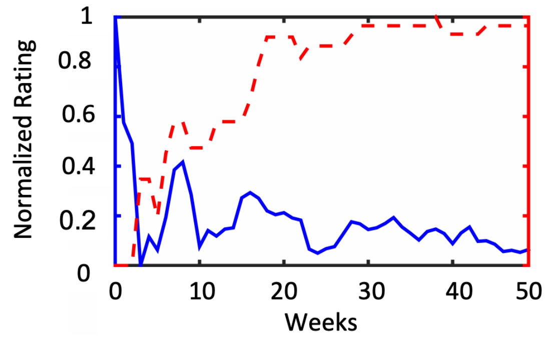
Correlation at a positive lags implies that $X$ is a good predictor of $Y$ and positive/negative CCF values indicate the changes in the series $X$ and $Y$ are in the same/opposite directions respectively.

### 3.4.2 Buffered Spamming

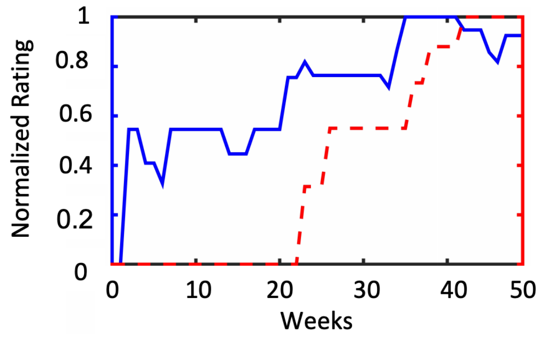
How do restaurants deal with their weaning popularity and growth of dislike ratings? Do they proactively inject deceptive reviews to maintain threshold average rating/popularity or to lessen the impact of truthful dislike reviews? To analyze the impact of truthful ratings on the rate of deceptive like ratings, we compare the time-series of truthful average rating, truthful like rating and truthful dislike rating (as covariates) with the deceptive like time-series (as the response). From Fig. 3.7, we note that the time-series for deceptive like ratings is shifted in future and increases with decrease in truthful average rating (Fig. 3.7.a) and truthful like rating (Fig. 3.7.b). Further, the increase in truthful dislike ratings tends to also cause increase in deceptive like ratings (Fig. 3.7.c). It is also interesting to note that the changes in the response time-series ($Y \sim$ deceptive like ratings) are tightly connected with



(a) Truthful average rating

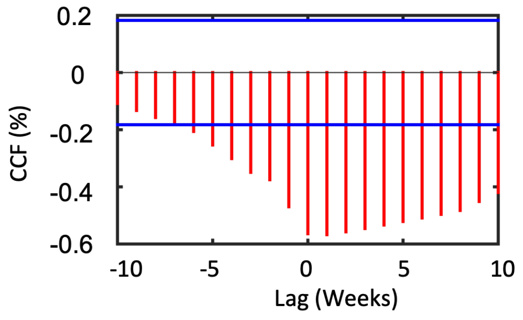


(b) Truthful like rating

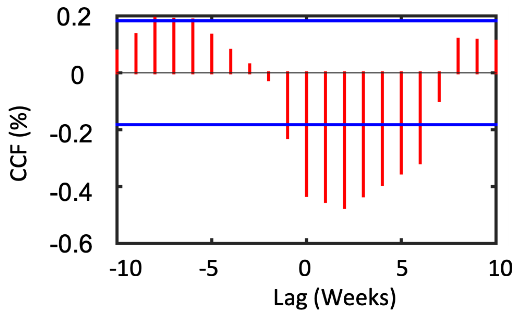


(c) Truthful dislike rating

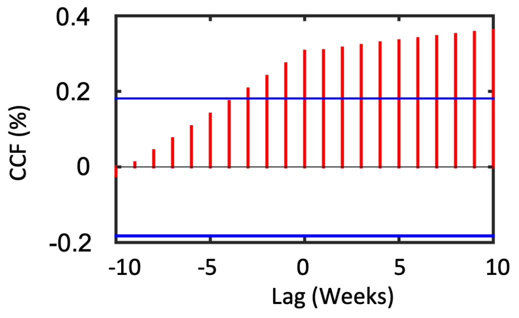
Figure 3.7: Buffered Spamming - Time series of truthful ratings (solid blue) vs deceptive like rating (dashed red) for different representative restaurants. Representative restaurants refers to the ones where the behavior was most prominent.



(a) Truthful average rating



(b) Truthful like rating

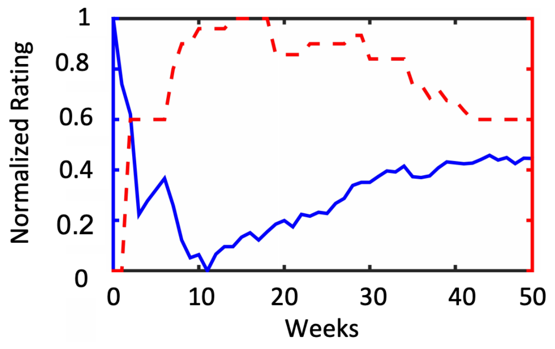


(c) Truthful dislike rating

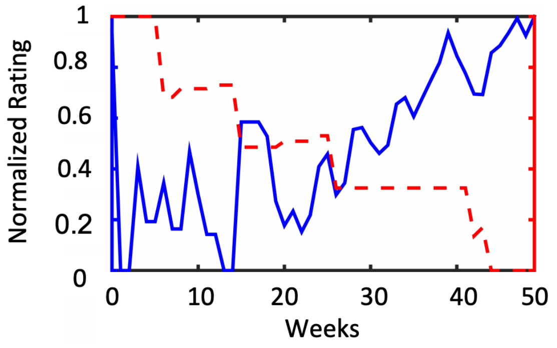
Figure 3.8: Buffered Spamming CCF plots for respective time-series in 3.7 for representative restaurants. Red lines indicate the CCF value and blue lines indicate the confidence interval bounds at 99% ($p < 0.01$) confidence.

the covariate time-series (e.g., steep drops in truthful avg. ratings from week 9 follows a rise in deceptive like ratings from week 11 in Fig 3.7.a, gradual decrease of truthful like ratings from week 15 almost co-occur with steady increase in deceptive like ratings in Fig 3.7.b, increase in truthful dislike in the first 20 weeks follow an increase of deceptive like in Fig 3.7.c). These patterns tend to indicate a causality beyond mere coincidence. It is as if there is a buffer action at work which adjusts the spamming rate by injecting deceptive like reviews as the truthful average and like ratings decrease or truthful dislike ratings increase. Hence, we refer to this spamming pattern as buffered spamming.

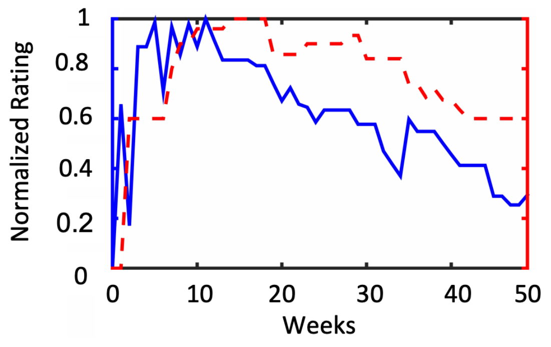
To further confirm and quantify the strength of the correlation, we computed the CCF plots (Fig. 3.8) of the covariate and response time-series corresponding to Fig 3.7. We note that for all potential causalities, the CCF values exceed the 99% confidence interval bounds indicating statistically significant correlations. Further, all the correlations exhibit positive lags (in the range of [0, 5] weeks) indicating that truthful rating influences the future rate of deceptive like ratings. Negative correlations in Fig. 3.8.a, 3.8.b explain the fact that the spamming increases when the average truthful rating and like truthful rating decrease. Positive correlations in Fig. 3.8.c explain the increase in deceptive like ratings with increase in truthful dislikes. These results tend to confirm the buffered nature of spamming in Fig. 3.7 which is an attempt of self-promotion via deceptive like reviews when the truthful reviews are not in their favor.



(a) Truthful average rating

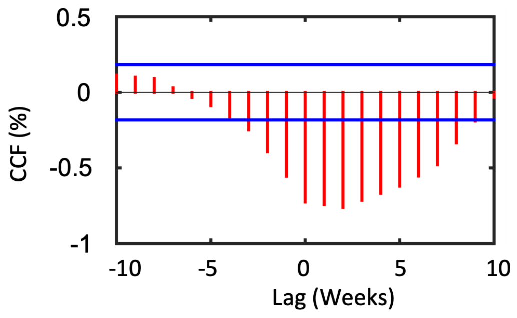


(b) Truthful like rating

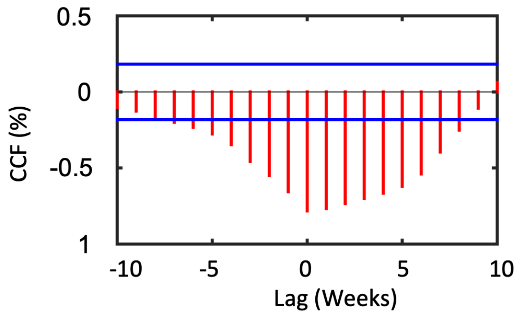


(c) Truthful dislike rating

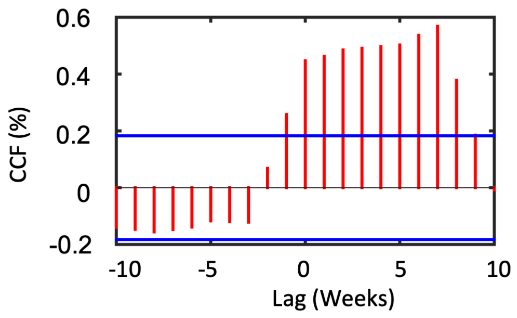
Figure 3.9: Reduced Spamming - Time series of truthful ratings (solid blue) vs deceptive like rating (dashed red) for different representative restaurants. Representative restaurants refers to the ones where the behavior was most prominent.



(a) Truthful average rating



(b) Truthful like rating



(c) Truthful dislike rating

Figure 3.10: Reduced Spamming - CCF plots for respective time-series in 3.9 for representative restaurants. Red lines indicate the CCF value and blue lines indicate the confidence interval bounds at 99% ($p < 0.01$) confidence.

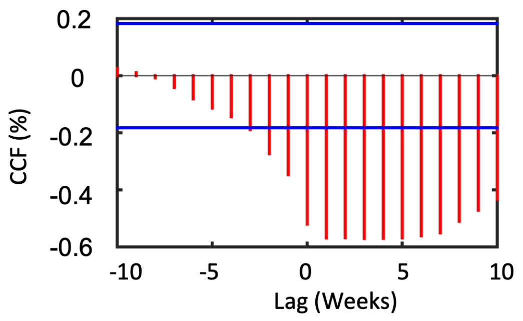
### 3.4.3 Reduced Spamming

We now explore the case when restaurant maintain decent popularity and rating. Is there a reduction in the spam injection rate, as they have a better standing already? We again analyze the impact of truthful ratings on deceptive like ratings. In Fig. 3.9, we see that the time series for deceptive like rating is shifted in future and now decreases with the increase in truthful average rating (Fig. 3.9.a) and truthful like rating (Fig. 3.9.b). Moreover, the decrease in the truthful dislike rating causes the deceptive like rating to decrease (Fig 3.9.c). Interestingly, here also the changes in the response time series ($Y \sim$ deceptive like ratings) are tightly connected with the co-variate time series (e.g., gradual increase in truthful avg. rating from week 11 is followed by drop in deceptive like rating from week 18 in Fig. 3.9.a, rapid increase in truthful like rating from week 14 to 15 co-occur with rapid decrease in deceptive like rating in Fig. 3.9.b, gradual decrease in truthful dislike rating after week 11 is followed by gradual decrease after week 18 in deceptive like rating in Fig 3.9.c). These trends show a pattern where spam injection rates are reduced when the truthful reviews are favorable. Thus, we refer this temporal dynamics as reduced spamming.

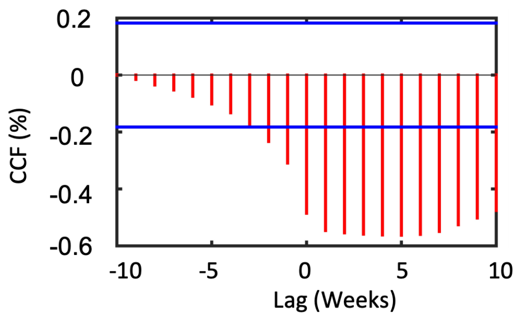
For significance testing, we computed the CCF plots (Fig. 3.10) of the covariate and response time series in Fig. 3.9. The CCF values exceed the 99% confidence interval bounds for all potential causalities indicating statistically significant correlations. Further, the positive lags (in the range of [0, 5] weeks) indicate that truthful ratings influence the future rate of deceptive like rating. Negative correlations in Fig. 3.10.a, 3.10.b explain the fact that the spamming decreases when the average truthful rating and like truthful rating increase. Positive correlations in Fig. 3.10.c explain the decrease in deceptive like ratings with decrease in truthful dislikes.

### 3.4.4 Average Cross Correlation

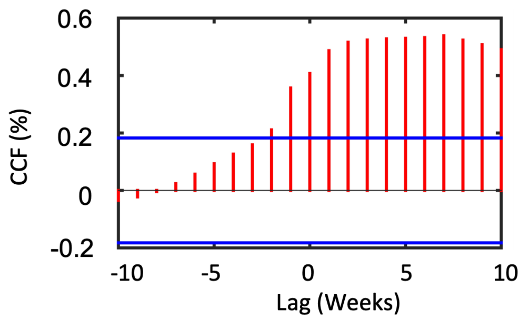
The above results although establish a decent confidence in causality, they were based on individual restaurants. To ascertain whether these patterns are prevalent, we evaluated their trend in all restaurants. Time series of all the restaurants cannot be shown as average as different restaurants have the buffered and spamming trend at different instance of time. However, it is important to note that since the lags and directions of correlations of buffered and reduced spamming share the same trend (see Fig. 3.8, 3.10), it is sufficient to explore the average CCF values over all the restaurants. The average CCF plots have been shown in Fig. 3.11 which strengthen our conclusion that the three segments of truthful rating time series are good predictors of deceptive like rating time-series as there exist statistically significant correlations at positive lags. We also see that the average CCF over all restaurants have small yet significant correlation at lag of -1, -2 weeks. This may be due to prognostic behavior of the restaurants where they can sense an imminent consumer dissatisfaction and thus begin spamming beforehand to maintain their ratings.



(a) Truthful average rating



(b) Truthful like rating



(c) Truthful dislike rating

Figure 3.11: Average CCF plot. Red lines indicate the CCF value and blue lines indicate the confidence interval bounds at 99% ($p < 0.01$) confidence.

## 3.5 Predicting Dynamics of Deceptive Ratings

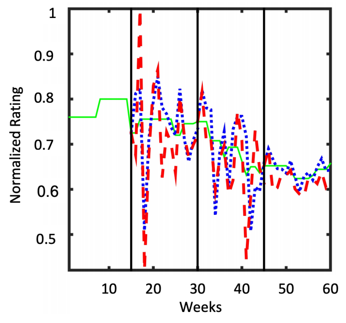
The preceding analysis shows that the dynamics of truthful ratings are harbingers of deceptive like rating. Naturally this raises the research question: Can we predict the dynamics of deceptive like ratings? This section employs vector auto regression to predict deceptive like ratings on restaurants using the time-series of the three truthful ratings.

Let $y_t$ denote a $n \times 1$ vector of n time series variables. A $p$-lag vector auto regression $VAR(p)$ model takes the form:

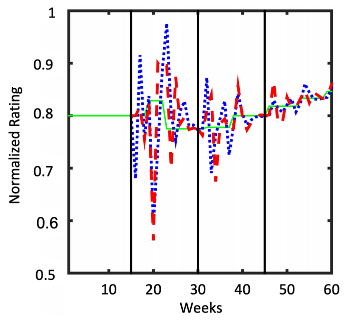
$$y_t = a + \sum_{i=1}^{p} A_i y_{t-i} + \epsilon_t \tag{3.3}$$

where $a$ is a bias vector of offsets with $n$ elements, $A_i$ are $n \times n$ autoregressive matrices and $\epsilon_t$ is an $n \times 1$ vector of serially uncorrelated innovations (error terms). Training a VAR model entails fitting multiple time-series and parameter estimation via maximum likelihood estimators. Upon parameter learning, values in $y_{t+1}$ are predicted using values of $y_{t-W+1}$ where $W$ is the width of the moving window.

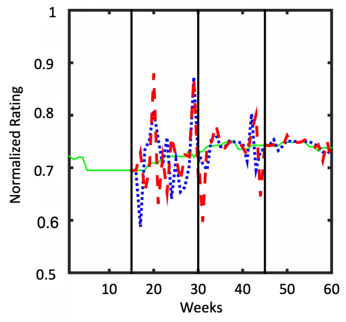
We employed a 4 dimensional VAR with the deceptive like rating as the response time-series and truthful average rating, truthful like rating and truthful dislike rating as covariate time-series. For this analysis, we consider moving window based forecasting of deceptive like ratings for the first 60 weeks (1 year approximately). We trained the model for each restaurant at lags 1 and 2, and predicted the next week's deceptive like rating. We experimented with three different training window widths $W$=15,30,45 weeks. For the rating prediction from 16th week to 30th week, model with training window of 15 weeks was used. For the rating prediction (see



(a) Early Spamming

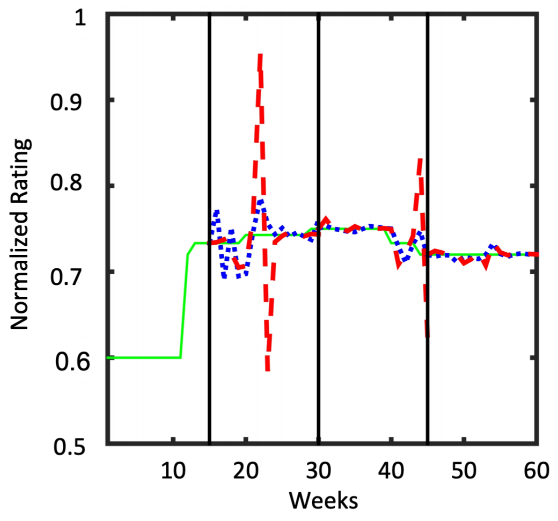


(b) Mid Spamming

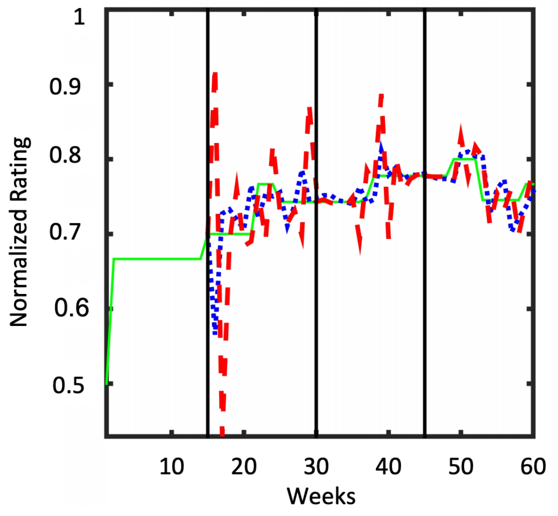


(c) Late Spamming

Figure 3.12: Spamming Policies - Deceptive like rating prediction of the next week using VAR Model. Forecasting was done using two $p$-lag VARs: dotted blue refers to $p$=1 week lag model; red dashed refers to $p$ = 2 week lag predictor. Solid green line represents the actual rating.



(a) Buffered Spamming



(b) Reduced Spamming

Figure 3.13: Causal spamming - Deceptive like rating prediction of the next week using VAR Model. Forecasting was done using two $p$-lag VARs: dotted blue refers to $p$=1 week lag model; red dashed refers to $p$ = 2 week lag predictor. Solid green line represents the actual rating.

Fig. 3.12) from 31st to 45th weeks, the model with training window of 30 weeks was used and for the rating prediction from 46th to 60th weeks, the model with 45 weeks training window was used. So, for example to predict the rating of 21st week, the training data of 6th to 20 weeks was used whereas to predict the rating of 41st week, the training data of 11th to 40th week was used. Thus, the window is moved each time to include the number of weeks specified by the window length. Fig. 3.12, 3.13 show the deceptive like time series forecast using $p$-lag VARs ($p$=1,2 weeks) across three training windows for early; mid; late spamming policies, buffered spamming and reduced spamming for the representative restaurants in each policy (based on closeness to cluster centroid). Table 3.2 reports the respective MAEs averaged over all restaurants in each policy.

From Fig. 3.12, 3.13 and Table 3.2, we note the following observations. Across all policies, the forecasts are decent improve with longer training windows. Consequently, MAEs are higher for 15 week training windows than 45 week windows across all spamming policies as longer training windows helps learn the complexities of deceptive rating dynamics. Also, the VAR model with lag = 2 weeks performs better than lag 1 as it can leverage the context of an additional previous time unit. Comparing the MAE across the three policies, we see that early spamming has the highest while late spamming has the lowest (see Table 3.2 and Fig. 3.12) indicating predicting deception in early spamming is more difficult compared to mid and late spamming. One reason for this could be that early spamming starts earlier, but their spams are also caught earlier. This can prompt spammers to devise newer and more complex ways of deception and altering their deception rate to avoid being filtered.

Table 3.2: Mean Absolute Error for different training windows and different spamming policies and trends for deceptive rating prediction.

| Spamming | Lag | Training Window | | | |
|---|---|---|---|---|---|
| | | 15 weeks | 30 weeks | 45 weeks | Avg. |
| Early | 1 | 0.55 | 0.17 | 0.11 | 0.28 |
| | 2 | 0.42 | 0.16 | 0.11 | 0.23 |
| Mid | 1 | 0.28 | 0.12 | 0.06 | 0.15 |
| | 2 | 0.22 | 0.1 | 0.05 | 0.12 |
| Late | 1 | 0.22 | 0.12 | 0.03 | 0.12 |
| | 2 | 0.12 | 0.04 | 0.03 | 0.06 |
| Buffered | 1 | 0.33 | 0.16 | 0.14 | 0.21 |
| | 2 | 0.2 | 0.15 | 0.12 | 0.16 |
| Reduced | 1 | 0.23 | 0.08 | 0.06 | 0.12 |
| | 2 | 0.18 | 0.07 | 0.04 | 0.1 |

This can result in a higher change rate in the deceptive ratings in early spamming (e.g., see Fig. 3.12.a) making it harder to predict.

We found that buffered/reduced spamming trends percolated across all early, mid and late spamming policies. So, the MAE reported in Table 3.2 for buffered and reduced spamming have been contributed by all types of restaurants. This is the reason why the average values of MAE for buffered (0.21, 0.16) and reduced spamming (0.12, 0.10) are those between the MAE of late (0.12, 0.06) and early (0.28, 0.23) spamming policies for both lag = 1, 2 VARs (see Table 3.2 last row). Upon VAR forecasting, we see that predicting buffered is harder than reduced (Fig. 3.13). This is because in buffered spamming, the reviews are injected via a counter buffer action (Fig. 3.7) making the changes in cumulative deceptive like rates higher (Fig. 3.13.a) thereby making forecasting harder as the spam injection rates are changing frequently. On the other hand, the deceptive reviews are decreasing in reduced spamming, resulting in relatively easier forecasts.

## 3.6 Predicting Truthful Popularity and Rating

Do deceptive reviews affect a restaurants popularity and average ratings? In order to answer this, we need a prediction framework that can forecast the popularity and average rating of a restaurant gained from truthful reviews. Popularity here refers to the total number of reviews in a time period. We studied factors that govern the truthful average rating and popularity of the restaurants 6 months in future and trained lasso regression models for future popularity/rating predictors. The response variables for popularity and rating predictors used the truthful average review rating and total number of reviews beyond 6 months of the start date for model building. Regression models were trained on truthful reviews of the first 10 weeks from the start date and used the following four feature families:

1. OL: Opinion lexicon of positive/negative words
2. NG: Word n-grams (n=1,2)
3. ASL: Restaurant domain specific Aspect/Sentiment Lexicon obtained by fitting the model in [47] to our data.
4. NTF: We computed the non-text features list below:
    - Weekly rating of each week of the training period
    - Overall rating of the training period
    - Friend count of the top 10 reviewer
    - Total Friend count of all the reviewers
    - Average review per user
    - Average review length
    - Rating deviation in each week of the training period
    - Overall rating deviation in the training period
    - Funny count in the reviews
    - Cool count in the reviews

Table 3.3: MAE for truthful popularity regression.

| | Early | Mid | Late |
|---|---|---|---|
| NTF | 3.94 | 2.02 | 1.52 |
| NTF+OL | 3.88 | 2.01 | 1.49 |
| NTF+OL+NG | 3.78 | 1.99 | 1.29 |
| NTF+OL+NG+ASL | 3.27 | 1.80 | 0.92 |

- Parking type Boolean features (street, private lot, garage, valet, validated, on-site)
- Attire type Boolean features (casual, dressy, formal)
- Ambience Boolean features (casual, intimate, classy, touristy, trendy, upmarket, hipster, upscale, divey, romantic)
- Restaurant specific Boolean features (good for kids, accepts credit cards, good for groups, price range, reservations, delivery, takeout, waiter service, outdoor seating, Wi-Fi available, good for lunch, dinner, desert, late night or breakfast, alcohol and wine, bar, noise level, cuisine available and wheelchair accessibility)
- Miscellaneous characters like (?, smileys, !)

The start date for this experiment was set to the month where fake reviews started accumulating for each policy (see Fig. 4, 5, 6) for setting a comparison reference for the subsequent experiments. Table 3.3, 3.4 report the MAEs of prediction of

Table 3.4: MAE for truthful average rating regression.

| | Early | Mid | Late |
|---|---|---|---|
| NTF | 0.47 | 0.38 | 0.16 |
| NTF+OL | 0.44 | 0.30 | 0.15 |
| NTF+OL+NG | 0.36 | 0.29 | 0.14 |
| NTF+OL+NG+ASL | 0.30 | 0.28 | 0.13 |

popularity and average rating for restaurants in each policy using 10-fold cross-validation (over all restaurants in each policy).

We employed forward feature selection of four feature families incrementally adding each family in the order NTF, OL, NG and ASL. From Table 3.3, 3.4 , we note that across all three policies, we that the regression model performs better as we continue adding the features OL, NG and ASL respectively in both popularity and rating prediction showing that natural language signals were helpful. This could be argued by the fact that truthful review contents invariably leave sentiment signals that eventually contribute to the truthful popularity and average ratings of a restaurant. It is important to note that the MAEs for this task is higher in early spamming than mid/late spamming policies. One reason for this could be that the effect of early spamming altered the actual popularity/average rating response for a restaurant in the policy that the regression model could not pick (as it was trained using truthful reviews).

Table 3.5: MAE comparison for popularity regressor.

| | Early | Mid | Late |
|---|---|---|---|
| Truthful only | 3.27 | 1.80 | 0.92 |
| All reviews | 3.97 | 2.38 | 1.16 |

Table 3.6: MAE comparison for rating regressor.

| | Early | Mid | Late |
|---|---|---|---|
| Truthful only | 0.30 | 0.28 | 0.13 |
| All reviews | 0.37 | 0.35 | 0.24 |

### 3.6.1 How reliable are Yelp's filtered reviews?

Ideally, deceptive reviews are injected for spamming and are not grounded on true experience, thereby regarded as fake. Hence, the information contained in fake reviews should be detrimental in predicting the future popularity or average rating of a restaurant. In other words, if we have a hypothetical regression oracle (a perfect guesser/ideal solver) for predicting the future truthful popularity and average rating of a restaurant trained on truthful reviews alone, then upon adding fake reviews to the training data for the regression oracle, the error on predicting the response should increase because the oracle has a noise in its training (imparted by fake reviews). This result can be a basis to indirectly validate Yelps filtering as follows.

Although we don't have a regression oracle for rating/popularity predictor, the regression models trained in Table 3.3, 3.4 are of high quality as the MAE of popularity is in the range of roughly 0,4 reviews (given median popularity as 65 reviews) and the MAE of the average rating lies in the range of [0.13,0.4] on a normalized average rating scale of [0, 1]. Hence, the regressors can be used as basis for testing the quality of Yelps filtering. We again trained popularity/rating regressors (using the same settings as in Section 3.6) with the full feature set NTF+OL+NG+ASL but also added Yelps filtered reviews along with truthful reviews in the training set (i.e., used all reviews in training). Table 3.5, 3.6, report the MAEs for popularity/rating regressors across all three policies. We note a statistically significant ($p < 0.03$) increase in MAE upon adding review filtered by Yelp across all policies (see the All reviews row in Table 3.5, 3.6).

Statistically significant increase in the MAEs of the regression models upon changing the training set renders a high confidence that the altered training set imparts a considerable noise, i.e., the result implies that the reviews filtered by Yelp imparted noise and were actually harmful to popularity/rating prediction. In other words, the reviews which imparted the noise in the popularity/rating models were not representative of truthful experiences (or potentially fake) and Yelp filtered those reviews. This indirectly raises confidence that Yelp's filter although may not be perfect is reasonably reliable.

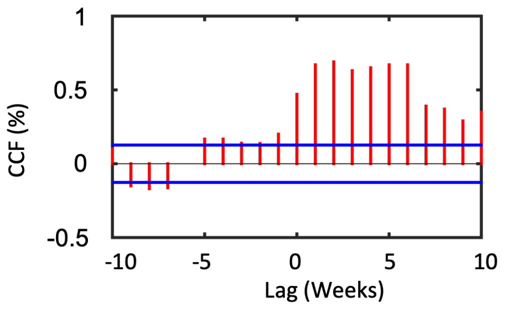
## 3.7 Predicting Imminent Future

The previous section provided us insights on long term (6 month) future prediction of a restaurant's popularity and rating. However, in the restaurant business, imminent prediction (e.g., next weeks popularity/rating) based on the review data till current time is more useful as it can help assess the recent impact of fake reviews or even help restaurants devise their plans easily. This section employs VARs on popularity and cumulative rating time-series to predict the imminent future performance of a restaurant. We found that the restaurant businesses have different dynamics for different days of the week. So, instead of using week or month as a time unit, we devised a novel time unit based on pooling multiple days of a week. For each week, Mon/Tue jointly formed the first time-step, Wed/Thu the second, and Fri/Sat/Sun the third followed by next weeks Mon/Tue as the fourth time-step.

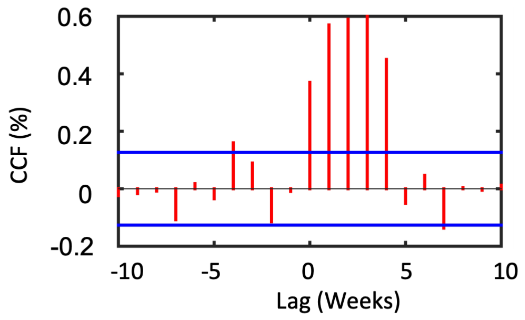
### 3.7.1 Popularity/Rating Time Series VARs

To build good VAR predictors of time-series of popularity and cumulative average rating, we investigated various time-series features as covariates and generated their respective CCF plots with the time-series of truthful popularity and truthful average rating as response variables. Then for time-series feature selection, we sampled covariates that had significant CCF values, at positive lags. The time-series features (covariates, $X(t)$) for each restaurant are tabulated in Table 3.7.

We show the CCF plots for selected covariates in Fig. 3.14, 3.15. We see that the features: (i) total number of friends of reviewers, and (ii) number of reviewers,

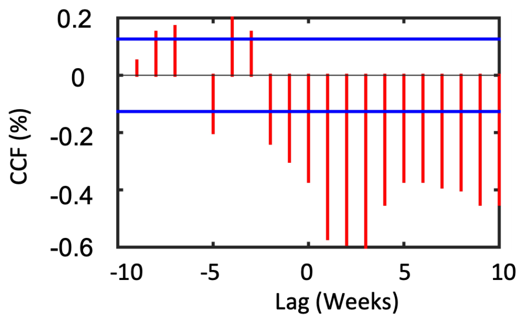


(a) # of friends of reviewers

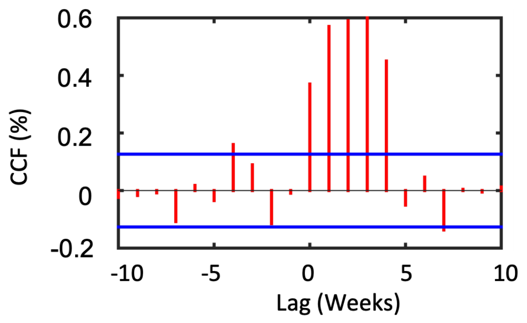


(b) # of reviewers

Figure 3.14: Truthful Popularity CCF Plot. Red lines indicate the CCF value and blue lines indicate the confidence interval bounds obtained 99% confidence ($p < 0.01$).



(a) Dislike Count



(b) # of reviewers with 5+ reviews

Figure 3.15: Truthful Average Rating CCF Plot. Red lines indicate the CCF value and blue lines indicate the confidence interval bounds obtained 99% confidence ($p < 0.01$).

| |
|---|
| Typed day index of time-step t : -1 for Mon/Tue, 0 for Wed/Thu, and +1 for Fri/Sat/Sun. |
| Total # of friends of all the reviewers in time-step $t$ |
| # of distinct reviewers in time-step $t$ |
| # of reviewers posting  5 reviews within [-$\infty$,$t$] |
| # of reviewers outside of the Chicago area in time-step $t$ |
| Standard deviation of the rating of the reviews in time-step $t$ |
| # of dislike reviews in time-step $t$ |
| # of like reviews in time-step $t$ |
| # of +ve lexicon words normalized by review length in time-step $t$ |
| # of -ve lexicon words normalized by review length in time-step $t$ |
| # of words (lol, !, ?, etc) normalized by review length in time-step $t$ |
| # of restaurant specific aspect sentiment words normalized by review length in time-step $t$ |

Table 3.7: Time series features for VAR Model

Table 3.8: Mean Absolute Error for next time-step's truthful popularity (# of reviews) prediction using VAR model.

| Training Window | Early | | Mid | | Late | |
|---|---|---|---|---|---|---|
| | lag1 | lag2 | lag1 | lag2 | lag1 | lag2 |
| 50 time-steps | 0.161 | 0.149 | 0.143 | 0.140 | 0.108 | 0.102 |
| 100 time-steps | 0.144 | 0.138 | 0.136 | 0.130 | 0.092 | 0.089 |
| 150 time-steps | 0.130 | 0.124 | 0.122 | 0.120 | 0.088 | 0.082 |
| Average | 0.145 | 0.137 | 0.134 | 0.130 | 0.096 | 0.091 |

Table 3.9: Mean Absolute Error for next time-step's truthful rating (# of reviews) prediction using VAR model.

| Training Window | Early | | Mid | | Late | |
|---|---|---|---|---|---|---|
| | lag1 | lag2 | lag1 | lag2 | lag1 | lag2 |
| 50 time-steps | 0.161 | 0.149 | 0.143 | 0.140 | 0.108 | 0.102 |
| 100 time-steps | 0.144 | 0.138 | 0.136 | 0.130 | 0.092 | 0.089 |
| 150 time-steps | 0.130 | 0.124 | 0.122 | 0.120 | 0.088 | 0.082 |
| Average | 0.145 | 0.137 | 0.134 | 0.130 | 0.096 | 0.091 |

have significant positive correlation at positive lags, indicating that they are good predictors of restaurant popularity in next time step (Fig 3.14.a, b). Similarly, for the case of next time step's average rating, the time-series of dislike count of reviews (Fig. 3.15.a) shows a significant negative correlation which is arguable as having more dislike reviews in previous time-steps can impact the cumulative average rating in future time-steps. Other significantly correlated features include, number of reviewers with 5 or more reviews (Fig 3.15.b) which is quite intuitive. In fact all the 12 features listed above had a significant CCF value at lags 1 and 2 periods. All twelve have thus been used as time series in the VAR model.

We trained VARs with the time-series of the 12 covariates (Table 3.7) and 2 response variables (popularity/average rating) for each restaurant at lags 1 and 2. We predicted the next time-step truthful popularity and truthful average and experimented with 3 moving training window widths, W=50,100,150 time-steps. Similar to Section 3.5, we use moving window based forecasting. Table 3.8, 3.9 show the corresponding MAEs averaged over all restaurants in each policy.

We note that across all policies, longer training windows improve imminent popularity and rating predictions as seen by the tightness of fit in Fig 16, 17 and respective MAEs in Table 3.8, 3.9. VAR models with lag 2 are better as they yield one more time step to regress on. Comparing the MAE across the three policies, we see that for early spamming, it is harder to predict the next time-steps popularity and rating than mid and late spamming. This is because early spamming having higher change rate of deceptive review injection (see Section 3.5) makes the prediction of imminent popularity and rating more difficult. It is also important to note that the trained models are good predictors of next time-step popularity and average rating as MAEs are quite low. The MAE of popularity is in the range of [0.53, 1.02] reviews (given median popularity per time-step is 19) and the MAE of rating lies in the range [0.082, 0.161] on a normalized scale [0, 1].

### 3.7.2 Modeling Deceptive Noise via Exogenous Variables

How do fake reviews filtered by Yelp affect the imminent future predictions of a restaurant's popularity and rating? Answering this can render insights into the robustness of Yelp's filtering on affecting the imminent future popularity/rating of restaurant. It can also improve the confidence on the reliability of Yelp's filtering (strengthening the conclusions Section 3.6.1). Similar to the analysis in Section 3.6.1, we include the filtered reviews in the VARs trained in Section 3.7.1 in predicting the imminent truthful popularity and rating. However, since our response variable are time-series, we cannot directly add the filtered reviews in the training set. Hence, we modeled the filtered reviews as exogenous variables in our VAR models. A VAR with exogenous variables takes the following form:

$$y_t = a + X_t.b + \sum_{i=1}^{p} A_i y_{t-i} + \epsilon_t \tag{3.4}$$

where $X_t$ is an $n \times r$ matrix representing the $r$ exogenous values for each of the $n$ elements in $y_t$. The other terms are similar to the traditional VARs detailed in Section 3.5. Exogenous variables can be seen as additional signal for each time-series. In our setting, we only used $r = 1$ exogenous value for each time-series. Specifically, the exogenous variables took values of all the 14 features used in the preceding analysis

Table 3.10: Mean Absolute Error for next time-step's truthful popularity (# of reviews) prediction using VAR model with the deceptive review series as exogenous inputs.

| Training Window | Early | | Mid | | Late | |
|---|---|---|---|---|---|---|
| | lag1 | lag2 | lag1 | lag2 | lag1 | lag2 |
| 50 time-steps | 1.44 | 1.28 | 1.36 | 1.32 | 1.18 | 1.08 |
| 100 time-steps | 1.32 | 1.14 | 1.28 | 1.27 | 1.09 | 1.07 |
| 150 time-steps | 1.26 | 1.12 | 1.22 | 1.20 | 0.98 | 0.92 |
| Average | 1.34 | 1.18 | 1.29 | 1.26 | 1.08 | 1.02 |

(Section 3.7.1) with the exception that those features were calculated only on the filtered reviews.

We repeated the experiments in Table 3.8, 3.9 with filtered reviews included in the training of VAR models as exogenous variables. The MAEs have been reported in Table 3.10 and 3.11. We note that across all policies and both rating and popularity predictors the MAE of VARs with filtered reviews as exogenous variables is higher. The increase in MAEs for all policies and across both popularity and rating predictors were statistically significant at 98% confidence levels (using a paired *t*-test). All other trends between the relative errors across policies and prediction performance based on lags remain the same as in Table 3.8, 3.9.

Thus, additional knowledge gained upon using the filtered reviews in VARs for popularity and rating predictions is actually harmful indicating the filtered reviews as being noisy and non-informative in predicting the imminent truthful popularity

Table 3.11: Mean Absolute Error for next time-step's truthful rating prediction using VAR model with the deceptive review series as exogenous inputs.

| Training Window | Early | | Mid | | Late | |
|---|---|---|---|---|---|---|
| | lag1 | lag2 | lag1 | lag2 | lag1 | lag2 |
| 50 time-steps | 0.344 | 0.325 | 0.312 | 0.302 | 0.288 | 0.253 |
| 100 time-steps | 0.323 | 0.298 | 0.276 | 0.270 | 0.245 | 0.217 |
| 150 time-steps | 0.283 | 0.212 | 0.244 | 0.200 | 0.196 | 0.194 |
| Average | 0.317 | 0.278 | 0.277 | 0.257 | 0.243 | 0.221 |

and ratings for a restaurant. In other words, those filtered reviews were likely to be untrue experiences as using them in the exogenous variables of 12 modalities in Table 3.7 increased the error against using the same 12 modalities on truthful reviews which had significant cross correlations with the target responses (see Fig. 3.14, 3.15). Yelp was able to detect those reviews as fake and filter them which not only shows that Yelp's filter is decent but also indicates its resilience that allowed it to pass our tests on imminent future predictions of rating and popularity.

## 3.8 Spam Detection in Yelp

Having characterized the temporal dynamics of opinion spamming, can we improve deception prediction beyond the existing state-of-the-art approaches leveraging the knowledge of temporal dynamics? To answer this we used our Yelp data and its filtering labels (filtered: fake; non-filtered-truthful) to set up the fake review detection as a classification problem. We use two state-of-the-art approaches as our baselines: Ott et al., (2011) [56] which employed linguistic n-grams (NG) and Mukherjee et al. (2013) [52] which uses a set of 8 anomalous behavioral features (BF) (e.g., reviewer deviation, percentage of positive reviews, etc). Classification settings were same as in [52], except that we partitioned the Yelp data by the spamming policies. We trained linear kernel SVMs with 5-fold cross validation on balanced data (using under-sampling). The soft margin parameter was tuned using cross validation and set to $C = 1.5$. From our analyses in this work, we derived a set of Time-Series Features (TSF) listed below:

- The truthful average rating of the previous week reviews only
- The truthful like rating of the previous week reviews only
- The truthful dislike rating of the previous week reviews only
- The truthful review count of the previous week
- The truthful like review count of the previous week
- The truthful dislike review count of the previous week
- The truthful average rating till the review date
- The truthful like rating till the review date
- The truthful dislike rating till the review date
- The standard deviation of the ratings of the previous week deceptive reviews

Table 3.12: SVM 5-fold CV classification results across time series features (TSF), behavioral features (BF), and n-gram features (NG) on Restaurants employing Early Spamming.

| Feature Setting | P | R | F1 | A |
|---|---|---|---|---|
| Ngrams (NG) | 63.5 | 77.1 | 69.6 | 65.0 |
| Behavior (BF) | 82.1 | 85.3 | 83.7 | 84.4 |
| TSF | 65.2 | 92.7 | 76.5 | 73.1 |
| NG+BF+TSF | 84.9 | 94.8 | 89.6 | 89.0 |

- The deceptive average rating of the previous week reviews only
- The deceptive like rating of the previous week reviews only
- The deceptive dislike rating of the previous week reviews only
- The deceptive review count of the previous week
- The deceptive like review count of the previous week
- The deceptive dislike review count of the previous week
- The deceptive average rating till the review date
- The deceptive like rating till the review date
- The deceptive dislike rating till the review date

We compare TSF and TSF+NG+BF against the baselines in Table 3.12, 3.13 and 3.14. We note that TSF alone does significantly better than linguistic n-grams,

Table 3.13: SVM 5-fold CV classification results across time series features (TSF), behavioral features (BF), and n-gram features (NG) on Restaurants employing Mid Spamming.

| Feature Setting | P | R | F1 | A |
|---|---|---|---|---|
| Ngrams (NG) | 64.2 | 77.7 | 70.3 | 67.5 |
| Behavior (BF) | 83.3 | 86.5 | 84.9 | 84.7 |
| TSF | 67.6 | 93.1 | 78.3 | 75.1 |
| NG+BF+TSF | 85.9 | 94.9 | 90.2 | 89.6 |

Table 3.14: SVM 5-fold CV classification results across time series features (TSF), behavioral features (BF), and n-gram features (NG) on Restaurants employing Late Spamming.

| Feature Setting | Precision | Recall | F1-score | Accuracy |
|---|---|---|---|---|
| Ngrams (NG) | 64.8 | 78.4 | 71.0 | 69.3 |
| Behavior (BF) | 83.9 | 87.2 | 85.5 | 86.2 |
| TSF | 68.5 | 93.9 | 79.2 | 76.4 |
| NG+BF+TSF | 86.3 | 95.3 | 90.6 | 90.1 |
| One class SVM | 61.3 | 68.4 | 64.6 | 66.5 |

but is weaker than the 8 anomalous behaviors proposed in [52]. However, upon combining TSF with NG and BF feature sets, we obtain the highest F1-scores which are significantly better than both linguistic and behavioral features demonstrating that the discovered temporal dynamics have a value in improving deception detection beyond just characterization. We also performed feature ablation (not shown due to space constraints) and found deceptive review count, like and dislike ratings to be among the most discriminative features. Since we have a imbalanced dataset, we want to see if one class classifier performs better. We performed trained using the positive class using all the three sets of features. However, the classification performance decreased implying the inability of one class model's separation of classes.

## 3.9 Summary

We performed an in-depth analyses on the temporal dynamics of opinion spamming. It used a large-set of reviews from Yelp restaurants and its filtered reviews to characterize the way opinion spamming operates in a commercial setting. Experiments using time-series analyses showed that there exist three dominant spamming policies: early, mid, and late across various restaurant. Our analyses showed that the deception rating time-series for each restaurant had statistically significant correlations with the dynamics of truthful ratings time-series indicating that spam injection may potentially be coordinated by the restaurants/spammers to counter the effect of unfavorable ratings over time. Causal time-series analysis of deceptive like rating timeseries as response with different covariates time-series (e.g., average truthful ratings, truthful like and truthful dislike ratings) established the presence to two additional trends of spam injection: buffered and reduced spamming. The covariate time-series along with various other features were then used to predict future deceptive ratings, long term/imminent future popularity and rating of a restaurant in the presence of deception using vector auto regression. The framework further allowed us to indirectly validate Yelp's filter which was shown to be reasonable. We also derived a novel suite of time-series features from our discovered temporal dynamics. Experiments on fake review detection showed the effectiveness of our features that significantly outperformed relevant baselines for the task of opinion spam detection establishing a value of the temporal dynamics in spam detection beyond characterization.

# Chapter 4

# Spam Detection in Social Networks

S. KC, S. Maity, and A. Mukherjee. Enwalk: Learning network features for spam detection in twitter. *Proceedings of International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*, pages 90-101,2017. [22]

In this chapter, we study the social network behaviors in Twitter. Twitter is one of the most powerful social media extensively being used. The behaviors of anomalous users are important to characterize and detect them. Further, the social relationship pattern created by anomalous users is also important as they characterize their effectiveness to spread the message to the public. We therefore exploit both the local node feature and the social network to model the system. We use random walks biased by nodes' properties to model their behavior. Results on real-world data verifies the effectiveness of our model.

## 4.1 Twitter Dataset

For this work, we use the Twitter dataset used in [85]. It contains 17 million users having 467 million Twitter posts covering a seven month period from June 1 2009 to December 31 2009. To extract the network graph for those 17 million users, we extracted the follower-following topology of Twitter from [24], which contains all the entire twitter user profiles and their social relationships till July 2009. We pruned the users so that they have social relationship in [24] and tweets in [85] and are left with 4,405,698 users.

Twitter suspends the accounts involved in the malicious activity[1]. To obtain the suspend status of accounts, we re-crawled the profile pages of all the 17 million users. This yielded a total of 100,758 accounts that had been suspended (the profile page redirects to the page *https://twitter.com/account/suspended*). We use this suspension signal as the primary signal for evaluating our models as the primary reason for account suspension is the involvement in the spam activity. However, there might be other reasons like inactivity. So, to ensure the suspended accounts are spammers, we further checked for malicious activities for those users. For this, we examined various URLs from the accounts timeline and checked them against a list of blacklisted URLs. We use three blacklists: Google Safebrowsing (*http://code.google.com/apis/safebrowsing/*), URIBL (*http://uribl.com/*) and Joewein

[1]https://support.twitter.com/articles/18311

(*http://www.joewein.net/*). We found that 75% of suspended accounts posted at least one shortened URL blacklisted. We also looked for duplicate tweets enforced for promotion. After applying these additional criteria, our final data comprised of 86,652 spammers and 4,319,046 non-spammers, which was used for evaluating our model.

## 4.2 Spam Analysis

Characterizing the dominant spammer types is important as it is the first step in understanding the dynamics of spamming. We studied the follower-following network creation strategies of the spammers. We found that there are two main types of spamming based on the follow-following strategies: (1) *follow-flood spammers* and (2) *vigilant spammers*. So, the question arises why some spammers are more successful? In this section, we study the behavioral aspects of tweet dynamics of spammers. We later leverage them in model building.

### 4.2.1 Spammer Type

To analyze the strategies of follower-following, we calculated the number of followers (users that are following the current user) and the number of followings (users that the current user is following) for each spammer. Fig. 4.1 shows the plot in log scale count. It shows that the follower and following count differ for each spammers. The users with more followers than followings tend to be more successful as they have been able to earn a lot of users who are following them. So, we define success rate

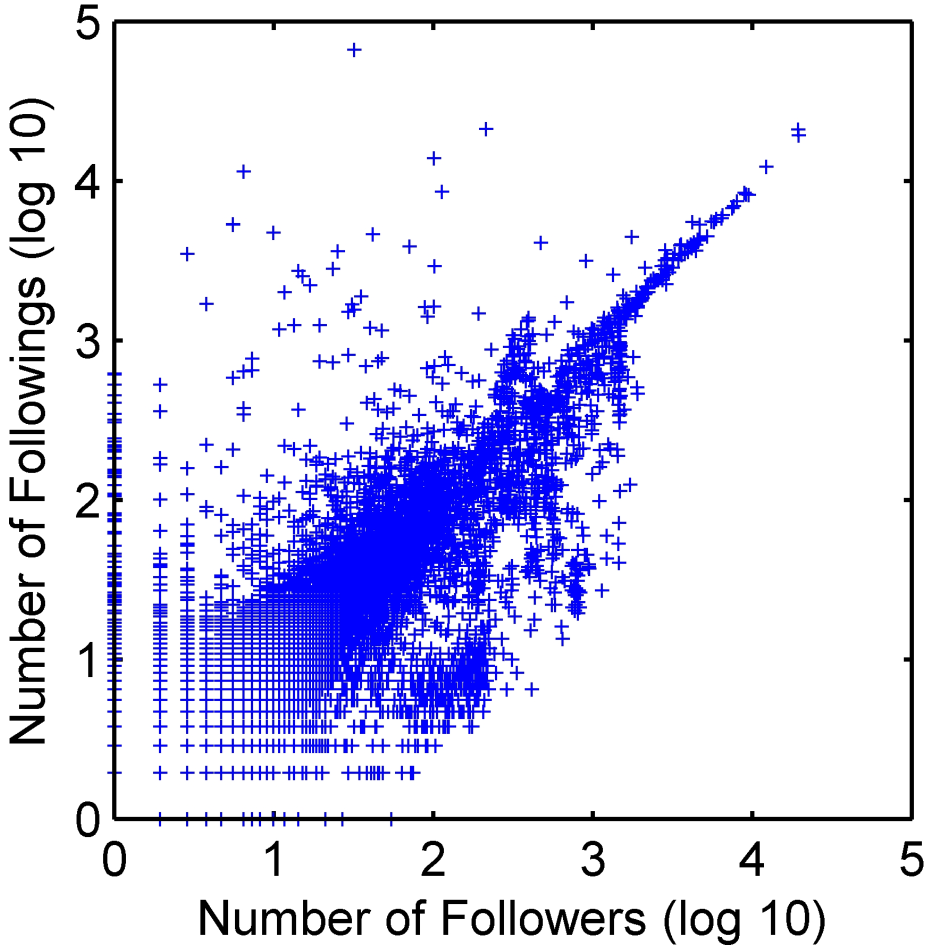


Figure 4.1: Follower-Following Count of Spammers. Each Blue dot represents a spammer.

as:

$$sr_u = \frac{\#offollowersofu}{\#offollowingsofu} \tag{4.1}$$

Based on the network expansion success rate, we find that there are two dominant spamming strategies:

- *Follow-flood Spammers*: These are less successful spammers who just flood the network with friendship initiation so as to get followers who they can influence. We categorize the spammers with success rate ($sr_u$) less than 1 in this type.

- *Vigilant Spammers*: These are successful spammers who take a cautious approach of friendship creation and content posting. Spammers with success rate ($sr_u$) greater or equal to 1 are categorized as vigilant.

To learn the dynamics of each spammer type, we further analyzed the success rate of spammers with other behavioral aspects  activity window, usage of promotion words or blacklist words and hashtag mentioning.

### 4.2.2 Activity Window

We compute the activity window as the number of days a user is active in the twitter network. Since, we dont have the exact time when a user was suspended, we approximate the time of suspension as the date of the last tweet tweeted by the user. We found that the average activity window of a vigilant spammer is 138 days with a standard deviation of 19 days compared to the average of 35 days and standard deviation of 12 days for follow-flood spammers. Although, the basic strategy of any spammer is to inject itself into the network and emit the spam contents, the success rate also depends how long it can remain undetected in the network. Vigilant spammers therefore have a higher success rate.

### 4.2.3 Fraudulence

One of the primary reason to spam is to inject constant fraudulence information. So, we analyzed the fraudulence behavior of the two types of spammers. We labeled the tweets containing promotional, adult words or the blacklisted urls as fraud tweets. So, we compute fraudulence as:

$$fr_u = \frac{\#of fraudtweetsofu}{total\#oftweetsofu} \tag{4.2}$$

We found that the average fraudulence of vigilant spammers is 0.34 compared to 0.86 for follow-flood spammers. The follow-flood spammers are thus more involved in spam.

### 4.2.4 Mentioning Celebrities and Popular Hashtags

Mentioning the popular celebrities or hashtags empowers a tweet. So, one of the common strategies of spammers is to include the popular ones in their tweets. We studied mentioning phenomenon and found that vigilant spammers mention half the celebrities per tweets compared to the follow-flood spammers.

## 4.3 Learning Latent Features for Spam Detection

Having characterized the dynamics of spamming in Twitter, can we improve spam detection beyond the existing state-of-the-art approaches? To answer this we used our Twitter data to setup a latent feature learning problem in networks. Our analysis is general and can be used to any social network.

### 4.3.1 Overview

As discussed in the previous section, the dynamics of Twitter are interesting and can be leveraged to catch the spammers. So, we use the spam dynamics to formulate the latent feature learning in social networks. Let $G = (V, E, X)$ be a given network with vertices, edges and the social network data of users in the social network. We aim to learn a mapping function $f\colon V \rightarrow R^d$ from nodes to a $d$-dimensional feature representations which can be used for prediction. The parameter $d$ specifies the number of dimensions of the latent features such that the size of $f$ is $|V| \times d$.

We present a novel sampling strategy that samples nodes in network exploiting the spam dynamics such that the *equivalent neighborhood* $EN(u) \subset V$ contains the node having similar tweeting behaviors with the node $u$. We generate $EN(u)$ for each nodes in the network and predict which nodes are the members of $u$s equivalent neighbors based on the learnt latent features $f$. The basic rationale is that we wish to learn latent feature representations for nodes that respect equivalent neighborhoods (which are based on the spamming dynamics) so that classification/ranking using the learned representation yields results that leverage the spamming dynamics.

## 4.3.2 The Optimization Problem

As our goal is to learn the latent features $f$ that best describe the equivalent neighborhood $EN(u)$ of node $u$, we define the optimization problem as follows:

$$\max_{f} \sum_{u \epsilon V} \log Pr(EN(u)|f(u)) \tag{4.3}$$

To solve the optimization problem, we extend the SkipGram architecture [13, 62, 76] which approximates the conditional probability using an independence assumption that the likelihood of observing an equivalent neighborhood node is independent of observing any other equivalent neighborhood given the latent features of the source node.

$$Pr(EN(u)|f(u)) = \Pi_{v \epsilon EN(u)} Pr(v|f(u)) \tag{4.4}$$

Since the source node and the equivalent neighborhood node have symmetric equivalence, the conditional likelihood can be modeled as softmax unit parameterized by a dot product of their features.

$$Pr(v|f(u)) = \frac{exp(f(v).f(u))}{\sum_{t \epsilon V} exp(f(t).f(u))} \tag{4.5}$$

The optimization problem now becomes:

$$\max_{f} \sum_{u \epsilon V} \Big[ -\log Z_u + \sum_{t \epsilon EN(u)} exp(f(t).f(u)) \Big] \tag{4.6}$$

For large networks, the partition function $Z_u = \sum_{t \epsilon V} exp(f(t).f(u))$ is expensive to compute. We thus use negative sampling [43] to approximate it. We use stochastic gradient descent over the model parameters defining the features $f$. Feature learning methods based on Skip-gram architecture are developed for natural language [42]. Since natural language texts are linear, the notion of a neighborhood can be naturally defined using a sliding window over consecutive words in sentences. Networks are not linear, and thus a richer notion of a neighborhood is needed. To mitigate this problem, we use multiple biased random walks each one in principle exploring a different neighborhood [13].

### 4.3.3 Equivalent Neighborhood Generation

The analyses of spam dynamics leads to an important inference that the nodes are similar if they have similar spam dynamics. We thus want to exploit those dynamics to generate the equivalent neighborhood $EN(u)$ for the node $u$. Nodes in a network are equivalent if they share similar behaviors. We use the random walk procedure which can be biased to generate the equivalent neighborhood.

We bias the random walks based on the four dynamics: common time of activity ($ct_{tv}$), success rate difference ($sr_{tv}$), fraudulence commonalities ($fr_{tv}$) and common mentioning in tweets ($me_{tv}$). We calculate each dynamics as follows:

$$ct_{tv} = \frac{\#\ of\ days\ with\ common\ activity}{\#\ of\ days\ either\ t\ or\ v\ is\ active} \tag{4.7}$$

$$sr_{tv} = 1 - \left| max\left(1, \frac{\#\ of\ t's\ followers}{\#\ of\ t's\ followings}\right) - max\left(1, \frac{\#\ of\ v's\ followers}{\#\ of\ v's\ followings}\right) \right| \tag{4.8}$$

$$fr_{tv} = 1 - \left| max\left(1, \frac{\#\ of\ t's\ fraud\ tweets}{\#\ of\ t's\ tweets}\right) - max\left(1, \frac{\#\ of\ v's\ fraud\ tweets}{\#\ of\ v's\ tweets}\right) \right| \tag{4.9}$$

$$me_{tv} = \frac{\#\ of\ common\ mentions\ between\ t\ and\ v}{total\ \#\ of\ mentions\ between\ t\ and\ v} \tag{4.10}$$

For all the above four features, a higher value represents a closer connection between the pair of nodes. For a source node $u$, we generate a random walk of fixed length $k$. The $i^{th}$ node $c_i$ of a random walk starting at node $c_0$ is generated with the distribution:

$$P(c_i = tc_{i-1} = v) = \begin{cases} \beta_{vt} & if(v,t)\epsilon E \\ 0 & Otherwise \end{cases} \tag{4.11}$$

where $B_{vt}$ is the normalized transition probability between nodes $v$ and $t$. The transition probability are computed based on the spam dynamics so that the source node has equivalent spam dynamics with its neighborhood nodes. We define four parameters which guide the random walk. Consider that a random walk just traversed edge $(t, v)$ to now reside at node $v$. The walk now needs to decide on the next step so it evaluates the transition probabilities on edges $(v, x)$ leading from $v$. We set the transition probability to $B_{vx} = \alpha_{pqrs}(t, v, x).w_{vx}$, where

$$\alpha_{pqrs}(t,v,x) = p.(ct_{tv}+ct_{vx})+q.(sr_{tv}+sr_{vx})+r.(fr_{tv}+fr_{vx})+s.(me_{tv}+me_{vx}) \tag{4.12}$$

where the parameters $p, q, r, s$ are used to prioritize the tweet dynamics. To select the next node, the random walk is biased towards the nodes which have similar tweet dynamics to both the current node and the previous node in the random walk.

## 4.3.4 Algorithm:ENWalk

Algorithm 1 details our entire scheme. We start with $\lambda$ fixed length random walks at each node $l$ times. To obtain each walk, we use *GetEquivalentNeighbor*, the random sampler that samples the node based on the transition probabilities computed in equation 4.12. It is worth noting that the tweet dynamics between the

Algorithm 1: ENWalk $(G, d, \lambda, l, k, [p, q, r, s])$

```
Input: graph $G(V, E, W, X)$
embedding dimensions $d$
walks per node $\lambda$
walk length $l$
context size $k$
tweet parameters $p, q, r, s$
Output: matrix of latent features $F$
1. $(CT, SR, FR, ME)$ = Preprocess$(G, p, q, r, s)$
2. Initialize $walks$ to empty
3. for $i = 1$ to $\lambda$ do
4.     for each $v_i \epsilon V$ do
5.         Initialize walk to $v_i$
6.         for $j = 1$ to $l$ do
7.             $x$ = GetEquivalentNeighbor$(G, CT, SR, FR, ME, walk[j], W)$
8.             Append $x$ to $walk$
9.         Append $walk$ to $walks$
10. $F$ = StochasticGradientDescent$(k, d, walks)$
```

nodes $(CT, SR, FR, ME)$ defined in equation 4.7, 4.8, 4.9, 4.10 respectively can be pre-computed. Once, we have random walks we can obtain $d$ dimensional numeric features using the optimization function in equation 4.6 with a window size of $k$. The three phases preprocessing, random sampling and optimization are asynchronous so that ENWalk is scalable.

## 4.4 Experiment

We applied ENWalk to twitter dataset to evaluate its effectiveness. In this section, we discuss the baseline methods and compare with ENWalk for classification and ranking.

### 4.4.1 Baseline Methods

For classification, we compare our model with two graph embedding methods: Deepwalk and node2vec. We use PageRank and Markov Random Field (MRF) approaches for ranking of spam nodes. We did not use feature extraction techniques like [1] as they only use the node features without using the graph structure.

**Deepwalk** [62]. It is the first approach to integrate the language modeling for network feature representation. It generates uniform random walks equivalent to sentences in the language model.

**Node2vec** [13]. It is another representation learning for nodes in the network. It extends the language model of random walks employing a flexible notion of neighborhood. It designs a biased random walk using BFS and DFS neighborhood discovery.

**PageRank Models**. PageRank is a popular ranking algorithm that exploits the link-based structure of a network graph to rank the nodes of the graph.

$$PR = (1 - \alpha) * M * PR + \alpha * p \tag{4.13}$$

where $M$ is transition probability matrix, $p$ represents the prior probability with which a random surfer surfs to a random page and $\alpha$ is damping factor. For variations of PageRank, we vary the values of $M$ and $p$ using trustworthiness of a user. Trustworthiness ($f_{Trust}$) is using a set of features (# of Blacklist URL, # of tweets, # of mentions, # of duplicate tweets, # of tweets containing adult/bad words, # of tweets containing violent words, # of tweets containing promotional words and the total time of activity for the user). We manually labeled $f_{Trust}$ of 800 users (400 non-suspended and 400 suspended). We gave a real-valued trustworthiness score between 0 and 1. A value closer to 0 means the user is most likely a spammer. We then obtain the weight of the features by learning linear regression model on the users.

*Traditional PageRank*: We use the default PageRank settings for $M$ and $p$.

*Trust Induced and Trust Prior*: Transition matrix $M$ is modified as $M_{uv} = M_{uv} * f_{Trust}(v), \forall u, \forall v$ and $f_{Trust}(v)$ is used as prior probability.

**Markov Random Field Models**. Markov Random Fields are undirected graphs (and can be cyclic) that satisfy the three conditional independence properties (Pairwise, Local, and Global). For the inference, we use the Loopy Belief

Table 4.1: Propagation matrix.

| | Spammer | Mixed | Non-spammer |
|---|---|---|---|
| Spammer | 0.80 | 0.40 | 0.025 |
| Mixed | 0.15 | 0.50 | 0.125 |
| Non-spammer | 0.05 | 0.10 | 0.850 |

Propagation algorithm. Inspired by spam detection in [8], we define three hidden states Spammer, Mixed, Non-Spammer and the Propagation Matrix is used as in Table 4.1. Logically, spammers follow other spammers more (hence 0.8 probability) and non-spammers tend to follow other non- spammers. We also include the mixed state to include those users who are difficult to categorize spammers or non-spammers.

### 4.4.2 Node Classification

We obtained the feature representations from three different algorithms: ENWalk, node2vec and DeepWalk using the settings used in node2vec and DeepWalk. All the feature learnings are unsupervised. Similar to node2vec and DeepWalk, we used $d = 128,\ \lambda = 10,\ l = 80,\ k = 10$. We found that the parameters $d, \lambda, l, k$ are sensitive in a similar style to node2vec and DeepWalk. We used each feature representation as an example for standard SVM classifier. We used 10-fold cross-validation using balanced data obtained from sub-sampling of the negative class. From the classification results in Table 4.2, ENWalk performs better. It has higher precision, recall, F1-score and accuracy due to the biased random walks.

Table 4.2: Classification Results: Precision, Recall, F1-score and Accuracy.

| Model | Precision | Recall | F1-score | Accuracy |
|---|---|---|---|---|
| DeepWalk | 0.44 | 0.49 | 0.46 | 0.51 |
| Node2vec | 0.46 | 0.53 | 0.49 | 0.57 |
| ENWalk | 0.59 | 0.66 | 0.62 | 0.71 |

### 4.4.3 Node Ranking

We use two metrics to evaluate the ranking results: Cumulative Distribution Function of Suspended Users and Precision@n. We rank all the nodes in the graph and provide a node rank percentile. For each node rank percentile, we compute the number of suspended users in that percentile. We plot the cumulative distribution function for those suspended users. We also calculate the Area Under Curve (AUC) for the CDF. The higher the area the better the model. Precision@n of Suspended Users evaluates how many top n nodes suggested by a model are actually the suspended users. This is effective to screen the nodes that are probable being spammers. To evaluate the ranking performance of ENWalk, we use Logistic Regression on the features obtained from the model. We compare our model with PageRank and Markov Random Field models. We present the CDF in Fig 4.2. We can see that ENWalk outperforms all the baseline models. We also computed the AUC and precision@100 (Table 3). A higher AUC and precision@100 signifies the ability to profile the top spammers.

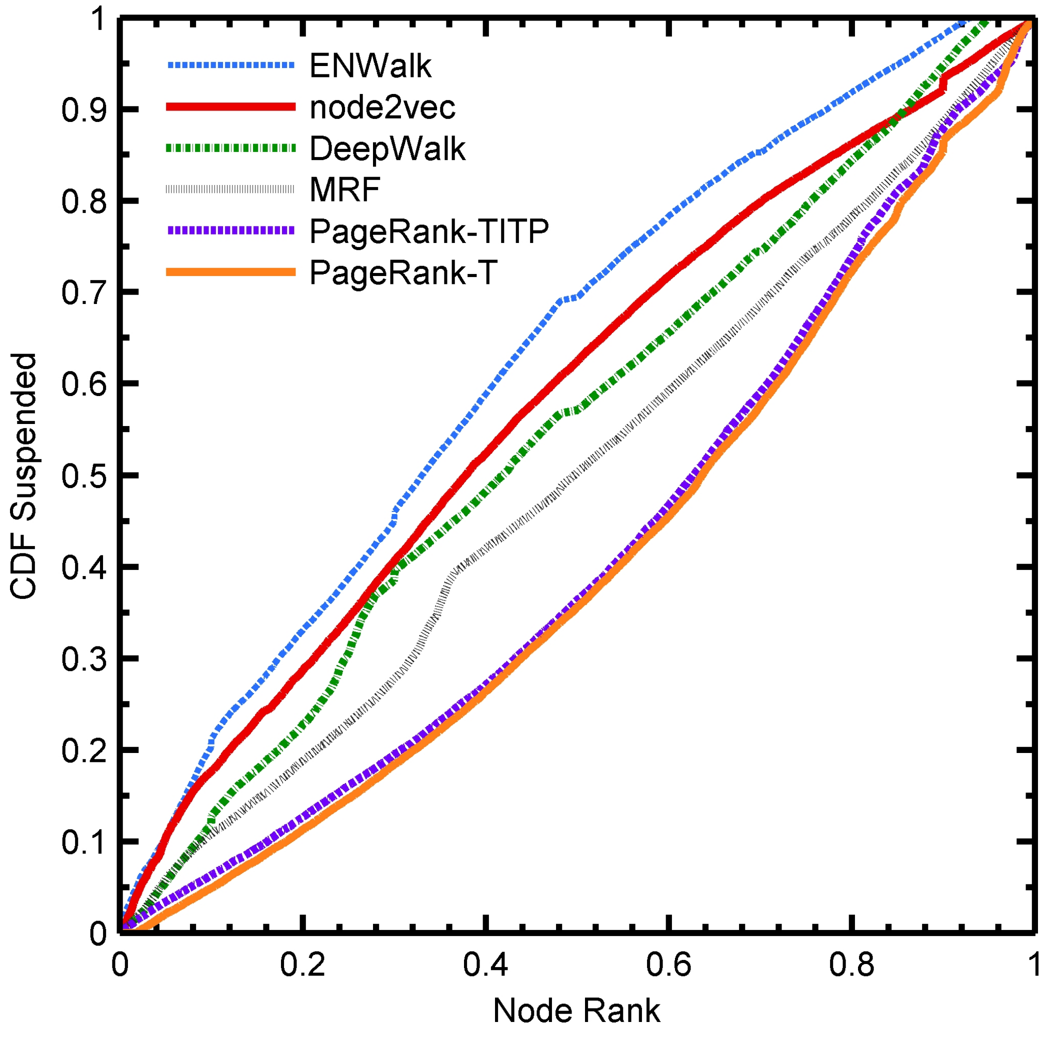


Figure 4.2: Cumulative Distribution Function of Suspended Nodes

Table 4.3: Ranking Results: Area Under CDF Curve and Precision@100

| Model | Area Under CDF Curve | Precision@100 |
|---|---|---|
| PR-T | 0.4059 | 0.02 |
| PR-TITP | 0.4181 | 0.03 |
| MRF | 0.4944 | 0.02 |
| DeepWalk | 0.5502 | 0.05 |
| node2vec | 0.5836 | 0.05 |
| ENWalk | 0.6335 | 0.12 |

## 4.5 Summary

We studied the problem of identifying spammers in Twitter who are involved in malicious attacks. This is very much important as it has many practical applications in todays world where almost everyone is actively social online. We proposed a method of spam detection in Twitter that makes use of the online network structure and information shared. This data driven approach is important as there is a lot of data of social medias online these days. We demonstrated the helpfulness of biased random walks in learning node embedding that can be used for classification and ranking tasks.

# Chapter 5

# Popularity Prediction in Online Marketplace

S. KC, S. De Sarkar, and A. Mukherjee. Product Popularity Modeling Via Time Series Embedding. *Proceedings of 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 650-653, 2018. [21]

## 5.1 Introduction

With the growth of online marketing, leveraging consumer opinions as reviews are key to business intelligence. In the past decade, several researches have explored different facets of online reviews such as review helpfulness votes [75], sales [89], competition [87] and spam [18, 20, 22]. However, current research discounts the dynamics of *popularity* and *competition*. Further, how does the popularity fare when two products are competing in the same market? How well can these be predicted? In this work, we choose Amazon as our case-study to explore these question.

Evaluating popularity is a subjective matter and has different meanings in different domains. High number of views can be considered more popular in Youtube. More likes and comments can be an indication of popularity in social networks. In online marketplace like Amazon, sales or number of reviews can be attributed to popularity. Amazon does not publicly release sales data but each review in Amazon has a Amazon Verified Purchase (AVP) tag i.e., a signature to attest whether the reviewer indeed purchased the product from Amazon. Review of consumer who did not buy the product on Amazon is tagged non-AVP. AVP reviews are more genuine as consumer purchase is verified. Whereas, anyone claiming to use a product can write non-AVP review. So, we consider only the AVP reviews in our experiment as they are closer to verified sales and review. To generate the popularity time series, we group the number of AVP reviews in a week. Thus, we refer the time series of popularity as *Product Popularity based on Sales Review Volume (PPSRV)*. We will use the term *PPSRV* for popularity in rest of the paper.

In this work we leverage neural networks to generate an embedding of a time-series and explore its effectiveness in future prediction. We feed the past time series to neural networks in order to obtain the time series embeddings. We train our time series models on moving window and predict the next time unit value using the time series embeddings. We further extend our model for a competing environment where two products are competing for the same set of consumers. Experimental results show the time-series embeddings learned outperform the existing state-of-the-art approaches for popularity prediction.

## 5.2 Time Series Embedding Framework

In this section, we formally define the problem and propose time series embedding, a fixed size vector that can be used for predicting the future time series.

We start by assuming that we are given a time series of popularity. This could be a time series of sales, reviews count received in a consumer review website, views count of Youtube video or number of likes, comments and shares in social network. To make a better prediction, allied time series like sentiment and rating can also be employed. Now we want to convert the fixed length time series into a d-dimensional features which we refer as embedding. The embedding can later be used for machine learning tasks like classification, regression and prediction.

We formally define this as a problem of time series prediction and divide into two components: (1) learning embeddings of past time series (2) prediction of future time series. Since, there are various types of online content with different popularity measure and occur at different times, we devise a generic learning framework that generates a set of features for a fixed width time window. We use the learned latent features to predict the future popularity. Next, we extend our model of feature learning in a competing environment. Finally, we perform more experiments in competition dataset using embeddings and evaluate them.

### 5.2.1 Model

We propose the concept of a time series embedding: learning latent features from time series using deep neural networks. Recurrent Neural Networks are popular form of neural networks efficient for temporal history capturing. We use different forms of RNN to obtain the time series embedding. Time series embedding for a fixed sized time series of length ($l$) is fed to a neural network model to get a fixed size vector that represents the latent features of the input time series. We refer to these features as time series embeddings as they embed the knowledge of the time series. Next we employ these embeddings to predict the future time series. Thus, we summarize the history of popularity with embeddings and predict future popularity as shown in Fig. 5.1. We further explain the two phases, namely, *time series embedding generation* and *future time series prediction* in more details.

### 5.2.2 Time Series Embedding Generation

We feed the time series to different variations of sequential and attention layer models. As input, we use the original time series instead of scaled time series as the value of the future time series is affected by the scale at which the current time series is progressing. The dynamics of popular product may be different from less popular product.

From our pilot study, we found that time series prediction using embedding performs better with time series of differences as response variable. The difference time series is obtained from the original time series by finding the difference of consecutive

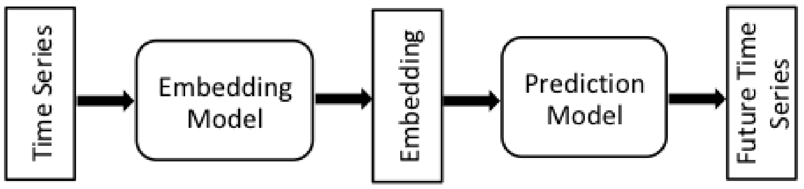


Figure 5.1: Schematic diagram of our approach for future time series prediction

values. Difference time series makes the model more fine-grained. We recover the popularity time series adding predicted difference value to one unit ahead time series value. This accounts to the fact that the popularity scale can be different based on type of product or service. However, the embeddings of the popular products in respective domains should be similar. For example: scale of number of views could be different for music videos compared to a tutorial video in Youtube considering both are popular in their respective domains.

Thus, these models take a vector of fixed size (containing the popularity history over a fixed period of time) as input, and output the future time series using difference value. For all models apart from the Dense model, the popularity time series is divided into fixed sized windows (as shown in Fig.5.2), in order to produce sequential data that can be given as inputs sequentially.

### 5.2.3 Time Series Embedding in Competing Environments

We use the popularity time series of two products under competition to predict the future time series for both products. Here, the model learns the dynamics of the competing products together. This enables it to make better predictions for both the

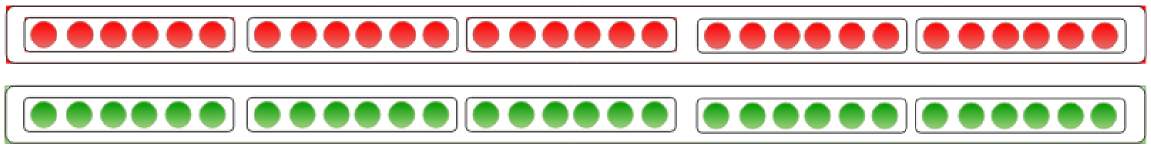


Figure 5.2: Standalone time series prediction. Input Representation with time series value of two competing products shown in red and green.

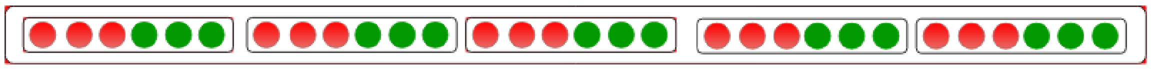


Figure 5.3: Competition time series prediction. Input Representation with time series value of two competing products shown in red and green.

individual products, since it has a bigger picture to look at. These models produce a single time series embedding that is used for the prediction of the popularity difference value of both the products together. Thus, these models expect the fixed length vector inputs, with the popularity history of the two competing products, and outputs the popularity difference value of both the corresponding products. Since the model expects the popularity time series of two products, the input size is twice as compared to the standalone prediction models. The popularity history is divided into $t$ fixed length vectors, each of size (2l/t), with the time series of the products concatenated with each other(as shown in Fig.5.3) in each vector.

### 5.2.4 Future Time Series Prediction

Once we have the time series embeddings that encode the past time series, we can predict the future time series using various machine learning regression techniques. After evaluating various models, we find neural networks perform best. We thus use a single Dense Layer.

### 5.2.5 Time Series Embedding Models

We use various types of embeddings models to evaluate the effectiveness of our approach. We train all our models with $l$ values of the past time series. We further divide the input into $t$ vectors, each of size $l/t$ (as shown in Fig. 5.2). For competing environments, we have two input time series that are trained together as shown in 5.3. We now explain the embedding models we use for our experiments:

- **LSTM**: standard LSTM model followed by ReLU activation to obtain the time series embedding.
- **GRU**: standard GRU model followed by ReLU activation to obtain the time series embedding.
- **BLSTM_LSTM**: LSTM stacked on top of a Bidirectional LSTM followed by ReLU activation to obtain the time series embedding.
- **BGRU_GRU**: GRU stacked on top of Bidirectional GRU followed by ReLU activation to obtain the time series embedding.

- **BLSTM_A**: Bidirectional LSTM followed by ReLU activation and Attention Layer to obtain the weighted average time series embedding
- **BGRU_A**: Bidirectional GRU followed by ReLU activation and Attention Layer to obtain the weighted average time series embedding

## 5.3 Experimental Evaluation

This section describes the dataset, experimental setup, evaluation and findings.

### 5.3.1 Dataset

We crawled a large set of reviews from Amazon consisting a total of 1,091,159 products across 5 domains (see Table 5.1). Our data consists of a snapshot of reviews in Amazon till May 31, 2016. Each review contains the reviewer information, review title, review content, helpfulness votes, number of comments, star rating (ranging from 1 to 5) and the Amazon Verified Purchase (AVP) tag. We generate *PPSRV* for all the products for succeeding experiments and analyses.

If we use day or week as time unit of the time series for *PPSRV*, we often face the problem of sparseness as there are generally a lot of days or weeks when there is no review for a product. On the other hand, using months or higher time unit, will generally not catch the dynamics of temporal signals. So, we first estimate the kernel density of the AVP reviews histogram for each product (with bin=1 week) via kernel density estimation (KDE) using diffusion [3] and then use it as the time-series. We

Table 5.1: Dataset Statistics

| **Domain** | **# of Products** | **# of Reviews** |
|---|---:|---:|
| Books | 455,254 | 17,455,747 |
| Electronics | 84,238 | 4,334,226 |
| Manufactured | 308,636 | 18,248,638 |
| Media | 208,342 | 11,286,938 |
| Software | 34,689 | 1,366,697 |
| Total | 1,091,159 | 52,692,246 |

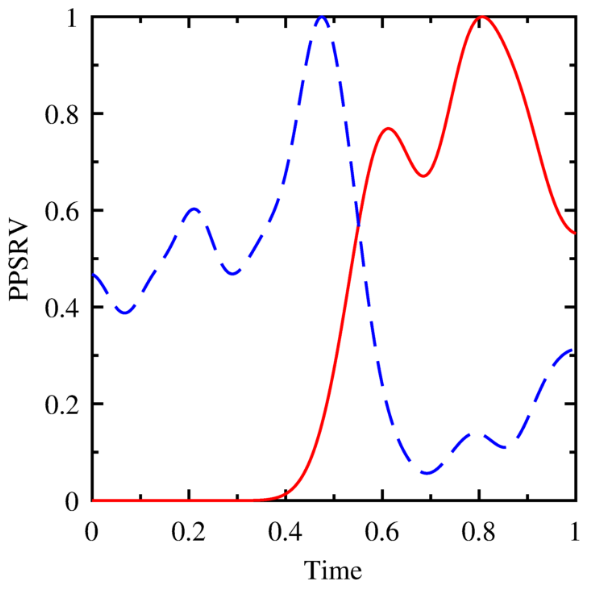
use KDE time series for all the experiments. The dataset can be downloaded from http://bit.ly/2ikkaC4.

In the similar fashion, we create the time series of sentiment and rating for each products. The rating of products range from 1 to 5. However, We normalize it to [0, 1] for our models. To create the time series of sentiment, we count the positive and negative sentiment lexicon in the reviews. We use the lexicon set from [16]. Average sentiment is calculated as normalized difference of positive and negative sentiment lexicon count.

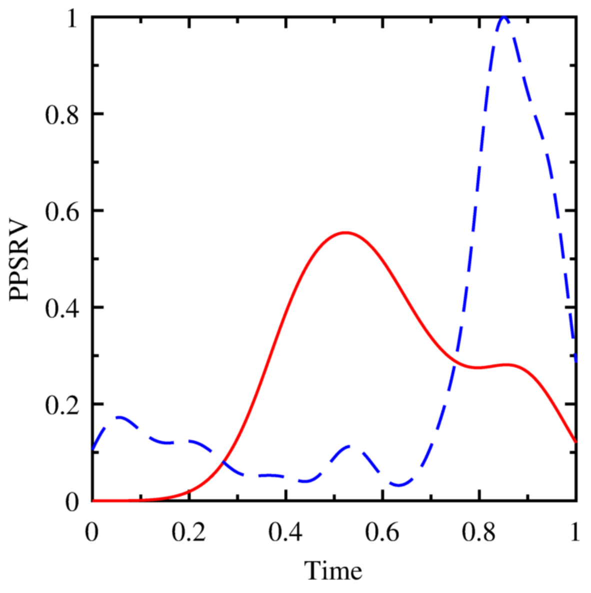
### 5.3.2 Competition in Amazon

There is always competition between similar products in the market. We anticipated similar phenomenon in Amazon and found several cases between similar products of different brands. Amazon recommends to the customer different products based on the search, browsing, and purchase history of several other customers. Some of the recommendation types are *"Sponsored Products Related To This Item", "Customer Who Bought This Item Also Bought", "What Do Customers Buy After Viewing This Item?"* . For each products, we populated such recommendations and generated the *PPSRV* of other related products. However, not all the recommendations from Amazon are potential competitors (i.e., not the same product type but different brand). So, we profiled 800 product pairs manually where we saw potential competition, i.e., the two products were of the same type but different brand, were being sold in the market in the same absolute time (even though one of them could have its reviews earlier than the other), and one product (*competitor*) took over the other product (*leader*) in *PPSRV* that was previously leading the market and show two representative examples in Fig 5.4. The first type is referred to as *death* where the introduction of the competitor completely seizes the market. The second type is *survival* where the previously leading product was hindered by the introduction of the competitor, but altogether regains its market share.

In most competitions in the real world, there are multiple products competing instead of just two. We found some cases in Amazon as well. However, we have very limited data of such multiple competitors. So, we ignore such products. The modeling of competition in online marketplace with more than two products could be an interesting problem for future research.



(a) Death



(b) Survival

Figure 5.4: Competition Examples. Scaled normalized *PPSRV* time-series depicting competition between *leader* (dashed blue) and *competitor* (solid red).

### 5.3.3 Baselines

For performance comparison, we show the best performing state-of-the-art approaches, one from linear method and another from non-linear method. For linear method, ARIMA performs the best while COMP-CUBE [39] is the best performing non-linear method that predicts better than the other state-of-the-art approaches. We only report these two approaches for our analysis as they perform best among the linear and non-linear algorithms. Each baseline model is trained on the same time window and the shifting of training window is performed in the similar fashion. COMP-CUBE learns the compressed model of the large collection of timestamped data. ARIMA is a regression model employing moving average. We also use a single Dense Layer to compare the effectiveness of simple neural networks instead of sequential models that are capable of capturing the histories.

### 5.3.4 Experimental Setup

For all experiments including baselines, we used *PPSRV* history and two normalized allied time series (rating, sentiment). We performed grid search on the parameters and found that training on the previous 250 data points works best for each product. Thus, standalone prediction models take an input vector of size 250, while competition models take an input vector of size 500. For sequential models, we used 5 windows, creating a sequence of 5 vectors as input, i.e., we used a window size of 50 for standalone sequential models, and a window size of 100(50 points of each product) for competition models. From the grid search, we used time series embedding size of 50 and 100 for standalone and competition models respectively. Thus, $l = 250$ and $t = 5$ for all the models. We use 70%-10%-20% of the total number of competing product pairs to prepare train-validation-test data.

We trained all models using Stochastic Gradient Descent (learning rate 0.05 & batch size 32). We optimized models using MAE as loss. All models were trained for 20 epochs.

### 5.3.5 Results

Table. 5.2 shows the average MAE values for the various models explored. All the models predict the future time series values. We see that competition models perform better than all of their standalone counterparts. It highlights that characterizing competition helps popularity better. It also indicates change in dynamics of time series due to competitor's standing. Further, we note that sequential models perform better than baselines. The worse performance of baselines indicate that the dynamics of popularity are inherently complex. Better performance of sequential model indicates superiority of their time series in encoding the history, which later can be used for other machine learning tasks.

## 5.4 Summary

We studied the effect of competition from the perspective of product popularity in online marketplace. We evaluate time series feature embedding using sequential deep

Table 5.2: Result showing the average MAE values for baselines models and Neural network models.

| **Model** | **Standalone ($\times 10^{-3}$)** | **Competition ($\times 10^{-3}$)** |
|---|---|---|
| **Baselines** | | |
| COMP-CUBE | 1.53 | 1.37 |
| ARIMA | 1.62 | 1.04 |
| Dense Layer | 0.82 | 0.79 |
| **Sequential Models** | | |
| LSTM | **0.53** | 0.42 |
| GRU | 0.59 | 0.50 |
| BLSTM_LSTM | 0.57 | 0.48 |
| BGRU_GRU | 0.61 | **0.40** |
| BLSTM_A | 0.56 | 0.52 |
| BGRU_A | 0.55 | 0.46 |

neural network models. We further extend our model for competing environment where two products compete to seize the same market. We found that the effect of competition helps model perform better. We demonstrate the effectiveness of our time series embeddings on large review dataset from Amazon.

# Chapter 6

# Conclusion

We have analyzed, designed and tested ideas based on anomaly detection in opinion media: social media and consumer review websites. We studied the behaviors of the anomalous users and exploit them to model methods to detect them.

Chapter 3 performed an in-depth analyses on the temporal dynamics of opinion spamming. It used a large-set of reviews from Yelp restaurants and its filtered reviews to characterize the way opinion spamming operates in a commercial setting. Experiments using time-series analyses showed that there exist three dominant spamming policies: early, mid, and late across various restaurants. Our analyses showed that the deception rating time-series for each restaurant had statistically significant correlations with the dynamics of truthful ratings time-series indicating that spam injection may potentially be coordinated by the restaurants/spammers to counter the effect of unfavorable ratings over time. Causal time-series analysis of deceptive like rating time-series as response with different covariates time-series (e.g., average truthful ratings, truthful like and truthful dislike ratings) established the presence to two additional trends of spam injection: buffered and reduced spamming. The co-variate time-series along with various other features were then used to predict future deceptive ratings, long term/imminent future popularity and rating of a restaurant in the presence of deception using vector auto regression. The framework further allowed us to indirectly validate Yelp's filter, which was shown to be reasonable. We also derived a novel suite of time-series features from our discovered temporal dynamics. Experiments on fake review detection showed the effectiveness of our features that significantly outperformed relevant baselines for the task of opinion spam detection establishing a value of the temporal dynamics in spam detection beyond characterization.

Chapter 4 studied the problem of identifying spammers in Twitter who are involved in malicious attacks. This is very important, as it has many practical applications in today's world, where almost everyone is actively social online. This paper proposed a method of spam detection in Twitter that makes use of the online network structure and information shared. This data driven approach is important as there is a lot of data of social medias online these days. We demonstrated the helpfulness of biased random walks in learning node embedding that can be used for classification and ranking tasks.

Chapter 5 explored the phenomenon of competition in the market where different brands of the same product type compete in sales. We found two cases of competition: (i) Death (the competitor beats the previous leader in the market), and (ii) Survival (the leader survives in the market with increasing its sales after being hit by a competitor). Using a labeled dataset of 800 competing pairs, we further modeled product popularity in standalone and competing environment. Competing models perform better at underlining the evidence of competition, which itself is an anomaly in the web.

This research work is more biased towards the quality of opinions expressed in the web. Our methods work well with a dense dataset, since we are mainly evaluating the temporal aspects. Since we are using weeks as time unit in most cases, our work is applicable only to the opinions which have higher frequencies across time. Sparse datasets will have time series with a lot of zero values. Prediction in such cases fails with our current models.

Our work requires sentiment associated with the opinions. For opinion spam detection in a consumer review website, we need to calculate sentiments such as number of positive words and number of negative words. For social network spam detection, we combine fraudulence scores with a graph to learn the embeddings. We therefore need opinion-rich reviews to evaluate the sentiments expressed.

Current work does not use a lot of metadata related to the opinions. Metadata about the person propagating the opinions, such as IP address, age, race, ethnicity, culture, and geography, could be important in learning the trends across different parts of the globe. Such data could be obtained from the opinion platforms to boost the performance of detecting anomalies.

To extend the findings of this thesis, future work can be directed to model a new rating revision approach using consumer trustworthiness in consumer review websites. This work can also help the researches on fake news detection. Another interesting extension of this work will be studying effects of spamming in e-commerce competition. Deep learning models are becoming dominant. They can be used to better model the text data and the attention based on features we studied.

# Bibliography

[1] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida. Detecting spammers on twitter. *Collaboration, electronic messaging, anti-abuse and spam conference (CEAS)*, 6:12, 2010.

[2] J. Bollen, H. Mao, and X. Zeng. Twitter mood predicts the stock market. *Journal of computational science*, 2(1):1–8, 2011.

[3] A. Z. I. Botev, J. F. Grotowski, and D. P. Kroese. Kernel Density Estimation via Diffusion. *The Annals of Statistics*, 38(5):2916–2957, 2010.

[4] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41(September):1–58, 2009.

[5] Y.-R. Chen and H.-H. Chen. Opinion spammer detection in web forum. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 759–762. ACM, 2015.

[6] P.-A. Chirita, J. Diederich, and W. Nejdl. Mailrank: using ranking for spam detection. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 373–380. ACM, 2005.

[7] T. M. Do, Y. Matsubara, and Y. Sakurai. Automatic and effective mining of coevolving online activities. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 233–246. Springer, 2017.

[8] G. Fei, A. Mukherjee, B. Liu, M. Hsu, M. Castellanos, and R. Ghosh. Exploiting Burstiness in Reviews for Review Spammer Detection. In E. Kiciman, N. B. Ellison, B. Hogan, P. Resnick, and I. Soboroff, editors, *Proceedings of the Seventh International Conference on Weblogs and Social Media, {ICWSM} 2013, Cambridge, Massachusetts, USA, July 8-11, 2013.* The {AAAI} Press, 2013.

[9] S. Feng, R. Banerjee, and Y. Choi. Syntactic stylometry for deception detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 171–175. Association for Computational Linguistics, 2012.

[10] S. Feng, L. Xing, A. Gogar, and Y. Choi. Distributional footprints of deceptive product reviews. *International Conference on Weblogs and Social Media*, 12:98–105, 2012.

[11] S. Ghosh, B. Viswanath, F. Kooti, N. K. Sharma, G. Korlam, F. Benevenuto, N. Ganguly, and K. P. Gummadi. Understanding and combating link farming in the twitter social network. In *Proceedings of the 21st international conference on World Wide Web*, pages 61–70. ACM, 2012.

[12] J. Ginsberg, M. H. Mohebbi, R. S. Patel, L. Brammer, M. S. Smolinski, and L. Brilliant. Detecting influenza epidemics using search engine query data. *Nature*, 457(7232):1012–4, 2009.

[13] A. Grover and J. Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 855–864. ACM, 2016.

[14] M. Hosseinia and A. Mukherjee. Detecting sockpuppets in deceptive opinion spam. *arXiv preprint arXiv:1703.03149*, 2017.

[15] M. Hosseinia and A. Mukherjee. Experiments with neural networks for small and large scale authorship verification. *arXiv preprint arXiv:1803.06456*, 2018.

[16] M. Hu and B. Liu. Mining and summarizing customer reviews. *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining KDD 04*, 04:168, 2004.

[17] X. Hu, J. Tang, Y. Zhang, and H. Liu. Social Spammer Detection in Microblogging. In *IJCAI*, volume 13, pages 2633–2639. Citeseer, 2013.

[18] N. Jindal and B. Liu. Opinion spam and analysis. In M. Najork, A. Z. Broder, and S. Chakrabarti, editors, *Proceedings of the international conference on Web search and web data mining WSDM 08*, page 219. ACM, 2008.

[19] N. Jindal, B. Liu, and E.-P. Lim. Finding unusual review patterns using unexpected rules. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1549–1552. ACM, 2010.

[20] S. K C and A. Mukherjee. On the Temporal Dynamics of Opinion Spamming: Case Studies on Yelp. In *25th International World Wide Web Conference, {WWW} '16, Montréal, Québec, Canada, April 11-15, 2016*, 2016.

[21] S. KC, S. De Sarkar, and A. Mukherjee. Product popularity modeling via time series embedding. In *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 650–653. IEEE, 2018.

[22] S. KC, S. Maity, and A. Mukherjee. Enwalk: Learning network features for spam detection in twitter. In *Social, Cultural, and Behavioral Modeling - 10th International Conference, SBP-BRiMS 2017, Washington, DC, USA, July 5-8, 2017, Proceedings*, pages 90–101, 2017.

[23] S. KC and A. Mukherjee. Characterizing product lifecycle in online marketing: Sales, trust, revenue, and competition modeling. *arXiv preprint arXiv:1704.02993*, 2017.

[24] H. Kwak, C. Lee, H. Park, and S. Moon. What is Twitter , a Social Network or a News Media? *The International World Wide Web Conference Committee (IW3C2)*, pages 1–10, 2010.

[25] K. Lee, J. Caverlee, and S. Webb. Uncovering social spammers: social honeypots+ machine learning. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 435–442. ACM, 2010.

[26] F. Li, Y. Gao, S. Zhou, X. Si, and D. Dai. Deceptive answer prediction with user preference graph. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1723–1732, 2013.

[27] F. Li, M. Huang, Y. Yang, and X. Zhu. Learning to identify review spam. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 2488, 2011.

[28] H. Li, Z. Chen, B. Liu, X. Wei, and J. Shao. Spotting fake reviews via collective positive-unlabeled learning. In *2014 IEEE International Conference on Data Mining, ICDM 2014, Shenzhen, China, December 14-17, 2014*, pages 899–904, 2014.

[29] H. Li, Z. Chen, A. Mukherjee, B. Liu, and J. Shao. Analyzing and Detecting Opinion Spam on a Large-scale Dataset via Temporal and Spatial Patterns. In *Ninth International AAAI Conference on Web and Social Media*, 2015.

[30] H. Li, A. Mukherjee, B. Liu, R. Kornfield, and S. Emery. Detecting Campaign Promoters on Twitter Using Markov Random Fields. In R. Kumar, H. Toivonen, J. Pei, J. Z. Huang, and X. Wu, editors, *2014 {IEEE} International Conference on Data Mining, {ICDM} 2014, Shenzhen, China, December 14-17, 2014*, pages 290–299. IEEE, 2014.

[31] J. Li, C. Cardie, and S. Li. Topicspam: a topic-model based approach for spam detection. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 217–221, 2013.

[32] J. Li, M. Ott, C. Cardie, and E. Hovy. Towards a general rule for identifying deceptive opinion spam. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1566–1576, 2014.

[33] E.-P. Lim, V.-A. Nguyen, N. Jindal, B. Liu, and H. W. Lauw. Detecting Product Review Spammers using Rating Behaviors. *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, pages 939–948, 2010.

[34] B. Liu. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167, 2012.

[35] M. Luca and G. Zervas. Fake it till you make it: Reputation, competition, and yelp review fraud. 2016.

[36] S. K. Maity, S. KC, and A. Mukherjee. Spam2vec: Learning biased embeddings for spam detection in twitter. In *Companion of the The Web Conference 2018 on The Web Conference 2018*, pages 63–64. International World Wide Web Conferences Steering Committee, 2018.

[37] Y. Matsubara and Y. Sakurai. Regime Shifts in Streams: Real-time Forecasting of Co-evolving Time Sequences. In *Proceedings of the 22nd {ACM} {SIGKDD} International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1045–1054, 2016.

[38] Y. Matsubara, Y. Sakurai, and C. Faloutsos. The web as a jungle: Non-linear dynamical systems for co-evolving online activities. *Proceedings of the 24th International Conference on World Wide Web*, pages 721–731, 2015.

[39] Y. Matsubara, Y. Sakurai, and C. Faloutsos. Non-linear mining of competing local activities. In *Proceedings of the 25th International Conference on World*

*Wide Web*, pages 737–747. International World Wide Web Conferences Steering Committee, 2016.

[40] Y. Matsubara, Y. Sakurai, and C. Faloutsos. Ecosystem on the web: non-linear mining and forecasting of co-evolving online activities. *WWW*, 2017.

[41] Y. Matsubara, Y. Sakurai, B. A. Prakash, L. Li, and C. Faloutsos. Rise and fall patterns of information diffusion: model and implications. *the 18th ACM SIGKDD international conference*, pages 6–14, 2012.

[42] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Distributed Representations of Words and Phrases and their Compositionality. *Advances in neural information processing systems*, pages 3111–3119, 2013.

[43] T. Mikolov, G. Corrado, K. Chen, and J. Dean. Efficient Estimation of Word Representations in Vector Space. *Proceedings of the International Conference on Learning Representations (ICLR 2013)*, pages 1–12, 2013.

[44] A. Mueen and E. Keogh. Online discovery and maintenance of time series motifs. In *6th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1089–1098, 2010.

[45] A. Mukherjee. Detecting deceptive opinion spam using linguistics, behavioral and statistical modeling. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Tutorial Abstracts*, pages 21–22, 2015.

[46] A. Mukherjee, A. Kumar, B. Liu, J. Wang, M. Hsu, M. Castellanos, and R. Ghosh. Spotting opinion spammers using behavioral footprints. *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '13*, page 632, 2013.

[47] A. Mukherjee and B. Liu. Aspect extraction through semi-supervised modeling. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 339–348. Association for Computational Linguistics, 2012.

[48] A. Mukherjee, B. Liu, and N. Glance. Spotting fake reviewer groups in consumer reviews. In A. Mille, F. L. Gandon, J. Misselis, M. Rabinovich, and S. Staab, editors, *Proceeding WWW '12 Proceedings of the 21st international conference on World Wide Web*, pages 191–200. ACM, 2012.

[49] A. Mukherjee, B. Liu, J. Wang, N. Glance, and N. Jindal. Detecting group review spam. In *Proceedings of the 20th international conference companion on World wide web*, pages 93–94. ACM, 2011.

[50] A. Mukherjee and V. Venkataraman. Opinion Spam Detection: An Unsupervised Approach using Generative Models. Technical report, UH-CS-TR-2014, 2014.

[51] A. Mukherjee, V. Venkataraman, B. Liu, and N. Glance. Fake review detection: Classification and analysis of real and pseudo reviews. Technical report, UIC-CS-03-2013. Technical Report, 2013.

[52] A. Mukherjee, V. Venkataraman, B. Liu, and N. S. Glance. What yelp fake review filter might be doing? In E. Kiciman, N. B. Ellison, B. Hogan, P. Resnick, and I. Soboroff, editors, *Proceedings of the Seventh International Conference on Weblogs and Social Media, ICWSM 2013, Cambridge, Massachusetts, USA, July 8-11, 2013.* The {AAAI} Press, 2013.

[53] J. D. Murray. *Mathematical Biology II - Spatial Models and Biomedical Appplications.* 2008.

[54] M. L. Newman, J. W. Pennebaker, D. S. Berry, and J. M. Richards. Lying words: Predicting deception from linguistic styles. *Personality and social psychology bulletin*, 29(5):665–675, 2003.

[55] M. Ott, C. Cardie, and J. Hancock. Estimating the prevalence of deception in online review communities. In *Proceedings of the 21st international conference on World Wide Web*, pages 201–210. ACM, 2012.

[56] M. Ott, Y. Choi, C. Cardie, and J. T. Hancock. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 309–319. Association for Computational Linguistics, 2011.

[57] T. Palpanas, M. Vlachos, E. Keogh, and D. Gunopulos. Streaming time series summarization using user-defined amnesic functions. *IEEE Transactions on Knowledge and Data Engineering*, 20(7):992–1006, 2008.

[58] B. Pang, L. Lee, et al. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, 2(1–2):1–135, 2008.

[59] S. Papadimitriou, J. Sun, and C. Faloutsos. Streaming pattern discovery in multiple time-series. In *Proceedings of the 31st international conference on Very large data bases*, pages 697–708, 2005.

[60] S. Papadimitriou and P. S. Yu. Optimal multi-scale patterns in time series streams. *Proceedings of the ACM International Conference on Management of data*, pages 647–658, 2006.

[61] P. Patel, E. Keogh, J. Lin, and S. Lonardi. Mining motifs in massive time series databases. *2002 IEEE International Conference on Data Mining, 2002. Proceedings.*, pages 370–377, 2002.

[62] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014.

[63] B. A. Prakash, A. Beutel, R. Rosenfeld, and C. Faloutsos. Winner takes all: competing viruses or ideas on fair-play networks. *Proceedings of the 21st international conference on World Wide Web - WWW '12*, page 1037, 2012.

[64] T. Qian and B. Liu. Identifying multiple userids of the same author. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1124–1135, 2013.

[65] T. Qian, B. Liu, L. Chen, Z. Peng, M. Zhong, G. He, X. Li, and G. Xu. Tri-Training for authorship attribution with limited training data: a comprehensive study. *Neurocomputing*, 171:798–806, 2016.

[66] B. Ribeiro and C. Faloutsos. Modeling Website Popularity Competition in the Attention-Activity Marketplace. In X. Cheng, H. Li, E. Gabrilovich, and J. Tang, editors, *arXiv preprint arXiv:1403.0600*, pages 389–398. ACM, 2014.

[67] M. Rowe and H. Alani. Mining and comparing engagement dynamics across multiple social media platforms. In *Web science*, 2014.

[68] Y. Sakurai, Y. Matsubara, and C. Faloutsos. Mining and Forecasting of Big Time-series Data. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 929–922, 2015.

[69] Y. Sakurai, S. Papadimitriou, and C. Faloutsos. Braid: Stream mining through group lag correlations. *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, page 610, 2005.

[70] P. Shrestha, S. Sierra, F. Gonzalez, M. Montes, P. Rosso, and T. Solorio. Convolutional neural networks for authorship attribution of short texts. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, volume 2, pages 669–674, 2017.

[71] K. Singh, R. Sandhu, and D. Kumar. Comment volume prediction using neural networks and decision trees. In *UKSim*, 2015.

[72] T. Solorio, R. Hasan, and M. Mizan. A case study of sockpuppet detection in wikipedia. In *Proceedings of the Workshop on Language Analysis in Social Media*, pages 59–68, 2013.

[73] G. Stringhini, C. Kruegel, and G. Vigna. Detecting spammers on social networks. In *Proceedings of the 26th annual computer security applications conference*, pages 1–9. ACM, 2010.

[74] T. Takahashi, B. Hooi, and C. Faloutsos. Autocyclone: automatic mining of cyclic online activities with robust tensor factorization. In *Proceedings of the 26th International Conference on World Wide Web*, pages 213–221. International World Wide Web Conferences Steering Committee, 2017.

[75] J. Tang, H. Gao, X. Hu, and H. Liu. Context-aware review helpfulness rating prediction. In Q. Yang, I. King, Q. Li, P. Pu, and G. Karypis, editors, *Proceedings of the 7th ACM conference on Recommender systems - RecSys '13*, pages 1–8. ACM, 2013.

[76] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1067–1077. ACM, 2015.

[77] K. Thomas, C. Grier, D. Song, and V. Paxson. Suspended accounts in retrospect: an analysis of twitter spam. *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 243–258, 2011.

[78] M. Tsagkias, W. Weerkamp, and M. De Rijke. News comments: Exploring, modeling, and online prediction. *Advances in Information Retrieval*, 2010.

[79] M. Vlachos, G. Kollios, and D. Gunopulos. Discovering similar multidimensional trajectories. *Proceedings 18th International Conference on Data Engineering*, pages 673–684, 2002.

[80] M. Vlachos, G. Kollios, and D. Gunopulos. Elastic translation invariant matching of trajectories. *Machine Learning*, 58(2-3):301–334, 2005.

[81] Z. Wang. Anonymity, social image, and the competition for volunteers: a case study of the online market for reviews. *The BE Journal of Economic Analysis & Policy*, 10(1), 2010.

[82] J. Weng, E. P. Lim, J. Jiang, and Q. He. Twitterrank: Finding topic-sensitive influential twitterers. *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining (WSDM 2010)*, pages 261–270, 2010.

[83] S. Xie, G. Wang, S. Lin, and P. S. Yu. Review spam detection via temporal pattern discovery. In Q. Yang, D. Agarwal, and J. Pei, editors, *The 18th {ACM} {SIGKDD} International Conference on Knowledge Discovery and Data Mining, {KDD} '12, Beijing, China, August 12-16, 2012*, pages 823–831. ACM, 2012.

[84] F. Yang, A. Mukherjee, and E. Dragut. Satirical news detection and analysis using attention mechanism and linguistic features. *arXiv preprint arXiv:1709.01189*, 2017.

[85] J. Yang and J. Leskovec. Patterns of temporal variation in online media. In I. King, W. Nejdl, and H. Li, editors, *Proceedings of the Forth International Conference on Web Search and Web Data Mining, {WSDM} 2011, Hong Kong, China, February 9-12, 2011*, pages 177–186. ACM, 2011.

[86] T. Yano and N. A. Smith. What's worthy of comment? content and comment volume in political blogs. In *International Conference on Weblogs and Social Media*, 2010.

[87] H. Zhang, G. Kim, and E. P. Xing. Dynamic Topic Modeling for Monitoring Market Competition from Online Text and Image Data. In L. Cao, C. Zhang, T. Joachims, G. I. Webb, D. D. Margineantu, and G. Williams, editors, *Proceedings of the 21th {ACM} {SIGKDD} International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pages 1425–1434. ACM, 2015.

[88] X. Zhang, S. Zhu, and W. Liang. Detecting spam and promoting campaigns in the Twitter social network. In *Proceedings - IEEE International Conference on Data Mining, ICDM*, pages 1194–1199, 2012.

[89] F. Zhu. Impact of Online Consumer Reviews on Sales: The Moderating Role of Product and Consumer Characteristics. *Journal of Marketing*, 74(2):133–148, 2010.