Review Spam Detection and Prediction by Modeling Dynamic Behavior in a Graph

Sheetal Mangesh Pandrekar
Stony Brook University
sheetalmangesh.pandrekar@stonybrook.edu

Shashank Jain
Stony Brook University
shashank.jain@stonybrook.edu

ABSTRACT

We propose a novel approach for Review Spam Detection by modeling the behavior of the reviewers in a dynamic graph. All previous methods have made use of Static Graphs for Review Spam Detection. By using Non-Negative Matrix Factorization, we decide the degree by which a reviewer behaves as a Spammer, Possible Spammer and Normal User. We observe the role transition of the reviewer across different snapshots of the graph. Based on their behavior, reviewers are assigned a Spamicity Score. Further, we exploit the connectivity of the Graph to predict if a new review is spam or not. For exploiting the information from the graph connectivity, we propose an algorithm developed by modifying the Personalized Page Rank algorithm.

1 Introduction

Online reviews play a significant role in the success of a product or service. However, review systems are often targeted by opinion spammers who seek to distort the perceived quality of a product by creating fraudulent reviews. Groups of fraudsters operate collaboratively on targeted products and take control of the sentiments of the potential customers.

The work in the area of Review Spam Detection has evolved over different strategies which include Context-based techniques, Behavior Analysis and Graph based approaches. Graph-based approaches which capture the interrelationships between the nodes have proven effective in detecting Spam. However, these graph-based techniques are based on modeling the data as a static graph. We are aiming to solve the problem of Opinion Spam Detection by formulating the problem as a dynamic graph. As described in [1],[2] not much research have been done in detecting anomalies in dynamic graphs. Moreover, there has been no prior work on Opinion Spam Detection which deals with Dynamic Graphs.

Modeling the problem as a dynamic graph allows us to analyse the changing behaviour of the reviewers over time. Based on the changing behaviours, we can classify the reviewers as Spammers, Non-Spammers and Possible Spammers. We introduce the concept of a Possible Spammer to avoid making early judgements about a reviewer being a Spammer. Observing the changing behaviour of the node in a dynamic graph helps us to make a better judgment about the reviewer being a Spammer. In a dynamic graph, new nodes and new edges will continue to be added to the existing graph structure. Using the modeled graph, we predict how likely it is for the new user to be involved in Spamming activity by considering its connectivity to other users in the graph. For this prediction task, we introduce an algorithm based on Personalized Page Rank score of the nodes.

Our basic graph structure will be a bipartite graph of Users and Products where an edge represents the review given by the user for the product. Bipartite graph representation have proven an effective strategy for representing the inter-connectivity underlying in Fraud review detection problem as described in [3].

In the next section, we briefly describe the prior work done in the area of Opinion Spam Detection. In section 3, we describe in detail the methods that we used to model the dynamic graph and how we used this model for the prediction task. We show how this method can be used for detecting group spamming activity. The dataset that we used for this work is described in Section 4. We present our evaluation in Section 5. The conclusion and suggestions for further work are given in Section 6.

2 Prior work

Graph based methods for Deception Detection

Graph-based approach helps to get subtle details of the activities of reviewers which provide important insights about spamming behavior. Previous work in the area of Opinion Spam detection which makes use of Graphs have all viewed the problem as a static graph. [3] describes the FRAUDEAGLE scoring algorithm which employs a signed Inference Algorithm (sIA) which is an extension of the LBP in order to handle signed networks. The sIA converges to assign labels to the users: fake/honest, review: good/bad, edge: real/fake. [6] exploits review bursts to find spammers who write fake reviews in bursts, and models the reviewers and their co-occurrence in bursts as a Markov Random Field (MRF) and employs the Loopy Belief Propagation (LBP) for feature induced message passing. The MRF consists of two nodes - observed nodes(values that are actually observed in the data) and hidden nodes(represent the true state underlying the observed data). The state of the node depends on the value of its corresponding observed nodes as well as the states of the neighbouring hidden nodes. The LBP updates the state of each node by sending messages from its neighbours and depending on the Local observation of the node.

Behavioral Analysis-based Deception Detection
Behavioral analysis aims at capturing behavioral cues that differ for Spammers. According to [5], a number of supervised learning methods to detect opinion spam fail due to the lack of reliable ground truth data needed for model building. Instead the authors proposed an unsupervised method using a fully Bayesian approach, and formulating opinion spam as a clustering problem. The idea is that opinion spammers have different behavioural distribution than non-spammers which helps in creating a distributional divergence between latent population of two clusters used here:-spam, non-spam. This model facilitates the characterization of various spamming activities using estimated latent variables and posterior. It models spamicity of a reviewer, review as latent variables. This model learns the latent population distribution of two clusters using various behavioural dimensions, and handles the cluster assignment of reviews in unsupervised setting based on probabilistic model clustering. Some of the features used in the model for learning spam and non-spam clusters are as follows:- Content similarity, maximum number of reviews, reviewing burstiness, early time frame, rating deviation etc. Author features are modelled as Beta distribution, while review features are captured as binary variables using a Bernoulli distribution. It then learns the latent behaviour distribution for spam and non-spam clusters using a generative process.

Group Spam Detection
The first work on detecting group spamming activity is introduced in [7]. Group Spamming is an important problem as an individual Spammer may not necessarily affect the quality of the product. However, spammers operating collaboratively in a group can take control of the perception of the product. In the work done in [7], they perform frequent pattern mining to extract candidate groups. They compute spam indicator value for each group by considering features like Group Rating Deviation, Group Content Similarity, Group Size and Early Time Frame. With these features, they develop three different ranking features to detect different flavours of Group Spamming Activities.

Review Graph based Spam
One of the first techniques to take advantage of the obvious product, review, user network embedded in opinion spam problem was [9]. This paper proposed a heterogeneous graph model named as review graph which consisted of 3 types of nodes: - users, reviewers, stores. It devised equations to formulate a scoring mechanism, thereby assigning trustiness score to the reviewers, honesty score to the reviews, and reliability score to the stores. After initializing the scores, the algorithm updates them iteratively until a convergence is reached.

3 Spam Detection and Prediction

3.1 Modeling Dynamic Behaviour for Spam Detection in a Review Network

We model the network of reviewers and products as a bipartite graph with edges representing the review written by the reviewer for the respective product. To model the dynamic behavior of the reviewer nodes, we consider a snapshot of the graph at each timestamp. The graph will evolve with time as new reviewers and products get added to the network and new edges are formed. This leads to a new snapshot at a later timestamp. For each snapshot, we extract the features necessary for detecting spamming activity of the reviewer nodes.

3.1.1 Feature Extraction

From the literature reviewed[10],we have observed that Rate Deviation, Content Similarity and Burstiness are the features that have led to good results in detecting spam. Thus, we extracted these features for each reviewer node. The basic idea behind using these features and the formula used to determine the features are as follows:

Rate Deviation (RD)- It measures how much the rating of this review deviates from the average ratings of all reviews of the product. A spammer will have a high rate deviation as he/she will try to promote or demote the product.

RD = |Rating of the review - Average Ratings of all reviews for the Product|

Content Similarity (CS) - It measures how similar the content of the review is to the contents of other reviews written by the reviewer. A spammer does not take much effort to change the content of each review. Spammer will have a high content similarity score. We are using the bag-of-words model to represent each review text and the cosine similarity between two reviews is the content similarity.[1]

If V is a set of all the reviews written by a reviewer, $CS = avg_{i,j}(cosine(V_i, V_j))$

Burstiness Score (BS) - It is the number of reviews written by the reviewer within the current snapshot. A normal user does not write many reviews in a short timespan. Thus, having a high burstiness score is indication of being a Spammer.

BS = Number of reviews written by a reviewer in the current snapshot

First Time Reviewer (FT) - A spammer will try to be the first reviewer for the product in order to deviate the customers. If a reviewer is constantly being the first reviewer, then it is an indication of a suspicious activity.

FT - Number of times the review written by a reviewer is the first review for the product

3.1.2 *Role Assignments and Analysis of Changing Behaviours*

After extracting the features, we need to determine if these features indicate spamming behaviour or not. We view this problem as that of assigning roles to each node depending upon their behavior which is determined from their features. For each node we decide how well it fits into the roles that we wish to assign to it. We chose three roles, namely, Spammer, Possible Spammer and Normal. We used the Non-Negative Matrix Factorization approach described in [3],[4] which have proven very efficient for role-extraction in time evolving graphs.

Consider a matrix U_t which is of the form nxf where n are the number of users, and f are the features collected above.
$U_t(i,j)$= Average of a feature j for a user i over all the products reviewed by him
, where U_t corresponds to a given timestamp t.

Similarly, let G_t be matrix of the form nxr where n are users, r is the possibility of each user being in role r. Finally, let F be the matrix of form rxf where r are roles, f are features, where each cell contributes the weightage of each feature in a given role r. These matrices can simply be represented in the below form:-

$U_t = G_t F$

Formally, we can say that given a non-negative matrix U_t, we have to find G_t and F which minimizes the functional,

$f(G_t, F)$ = argmin ||U - $G_t F$|| where || is the frobenius norm. We are using the functional as mentioned in [10].

Using the above formulation we can compute a node-role membership Gt for each snapshot t. Each row of G_t indicates the degree of membership of the node to the respective role ie. D_{spam}, $D_{possible-spam}$, D_{normal}.

We observe the role transitions of a node at various timestamps, which prove to be useful in detecting spamming activity as described in further sections.

3.1.3 *Assigning Spamicity Score*

A spamicity score is assigned to each reviewer by taking a weighted average of the roles that it exhibits. A general formulation for the Spamicity Score is as given by the equation,

$SpamicityScore = avg((a)(D_{spam})$
$+ (b)(D_{possible-spam}) - (c)(D_{normal}))$

Where $b < a$ as the degree of Possible Spam must not contribute more than the degree of Spam. Also, we subtract Dnormal, since degree of Normal Behaviour should reduce the spamicity score for the node.

This Spamicity score is determined from the features extracted from the current snapshot of the graph as determined in Section 3.1.2. Hence, it is not indicative of the past behavior of the node. In order to obtain a spamicity score that considers the past behavior of the node, which we call the Total Spamicity Score, we can use the Summary transition model as described in [8].We can assign weights a_{t-k}, a_{t-1}, a_t which dictates for each of the previous snapshots. A node that was spamming previously. and continues to spam, will get a higher spamicity score. Formally we can assign a collective spamming score, S to a node as follows:-

$S = a_{t-k}G_{t-k} + a_{t-k-1}G_{t-k-1} + a_t G_t$

3.2 Using Model for Spam Prediction of new Reviews

We have already determined the collective spamicity score for each node up until present. Now we compute the likelihood of whether the reviews in the new snapshot are possible cases of spam,or not. To help us with providing the probability of the new reviews in the next timestamp as being spam or not, we are going to use an adaption of Personalized PageRank algorithm.Before going into the details, let us be clear about a few assumptions:-

1. We choose a set of seed users nodes which are the users who have provided their reviews in the given snapshot.We add an user to the seed node list only once,inspite of the number of reviews given by him in the current snapshot.
2. Essentially we are not labelling the review as being spam or not, instead we would provide the likelihood of the user being spam.
3. We examine the k-distance neighbourhood around all the seed nodes.

Our basic Personalized page rank equation is as follows:-
$R = \alpha MR + (1-\alpha)S$
Our adapted PPR would prove very useful in finding possible spammer groups. Similar approaches for Graph Partitioning using PageRank have been described in [11] before. In the below sections, we shall describe the intuition behind initialization of above variables.

3.2.1 *Neighbourhood of seed nodes*

The idea behind neighbourhood is to find a set of users close to each other which could be possible candidates for being involved in group spam. Users far apart are not likely to be involved in group spam, therefore we've used the maximum value of k=2 in our approach
Matrix R is supposed to contain the set of seed nodes along with the k-distance neighbourhood involving user nodes. K distance neighbour is defined as follows:-

1-distance is defined as all the user nodes who have reviewed a product in common.
2-distance neighbouris defined as user nodes who

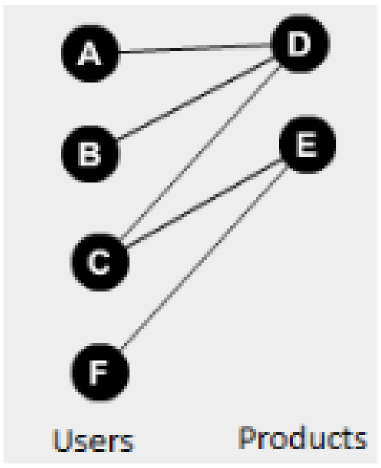

Figure 1: Sample portion of a bipartite graph

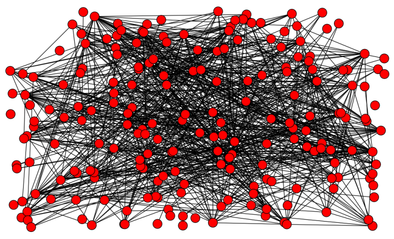

Figure 2: Neighbourhood Graph at a snapshot t

are 1distance neighbours of 1distance neighbours of initial user node.Similarly, k-neighbours can be defined.

In the Figure 1,
one distance neighbours of A are:-B,C
while 2-distance neighbour of A is F
Similarly,
1distance neighbour of B=A,C
2distance neighbour of B=F
1distance neighbour of c=A,B,F
2distance neighbour of c=
1distance neighbour of F=C
2distance neighbour of F=A,B

In Figure-2 we show a neighbour graph formed from this step which depicts the interconnectivity of various users

After calculation of neighbourhood, we initialize the R matrix by taking the collective spamming score from Section 3.1.3, i.e, we use the collective spamicity score for each user over k previous time snapshots

3.2.2 Relationship Matrix M

The purpose of Relationship matrix is to potray the amount each of the neighbourhood nodes add to the spamicity score of a given user node.We also know that M is supposed to be a Stochastic Matrix where sum of each elements in a column is normalized. Therefore we are assigning the weights to each node contribution in the following manner:

Sum of contribution of all one distance nodes is :0.75
Sum of contribution of all two distance nodes is :0.25

Let A,B be two one-distance nodes to a particular node X.
Number of products A has in common with X=N_A
Number of products B has in common with X= N_B

Spamicity contribution of X to A,B be X_A, X_B respectively
Therefore with respect to all one distance nodes from X, we can form the equation as:-

$N_A X_A + N_B X_B = 0.75$
where $M_{ij} = X_A$ when i=A,J=X

Similarly, contribution of nodes in relationship Matrix at distance 2 away from each other can be calculated.

3.2.3 Seed Matrix

Seed Matrix is defined to be consisting of nodes which could be visited with the probability of $1 - \alpha$. Some of the important points while considering the initial set of seed matrix are :

1. Seed matrix nodes are supposed to be chosen among the set of user nodes(seed nodes) which have reviewed in this snapshot.
2. We find out k random nodes from the seed-nodes set with the lowest Spamicity score from Section 3.1.3
3. We assign each of them negative values while keeping the column stoichastic. The intuition behind giving negative value is that during Power iteration of personalized Page rank, these k nodes spamming score would decrease by a probability of $1 - \alpha$. Therefore, at the end of power iteration, if these nodes still have high PPR value then something is seriously wrong about them.

3.2.4 Observations from PPR

We have applied PPR using the value of α as 0.85.Following observations were made from its results:-

1. Nodes with high PPR score are supposed to be in similar group with nodes with high spamming score.
2. Similarly, nodes with low PPR score are supposed to be in similar group with nodes with low spamming score.

Figure3 displays the result of PPR at a particular snapshot. Value at x co-ordinate does not matter, and have been chosen as such to keep these nodes together(it just means nodes in same snapshot). However, value of y shows which user nodes are given relatively high spamming score after PPR convergence. We can easily figure out nodes involved in group spam by looking for nodes with similar high

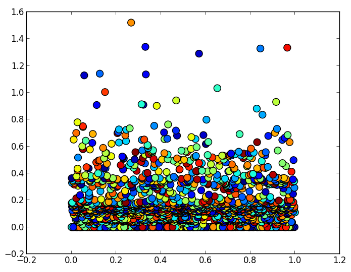

Figure 3: After PPR results at some snashot

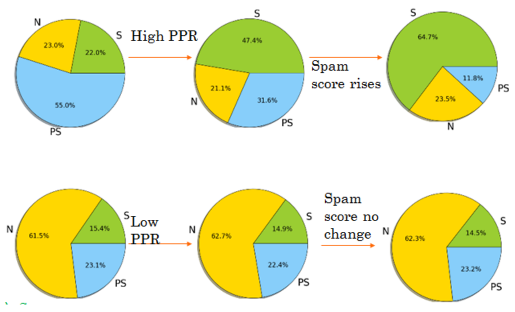

Figure 4: Role Transitions across various snapshots. S:Spam,PS:Possible Spam,N:not spam

Table 1: Precision vs Recall

Method	Precision	Recall
Static	49%	62%
Dynamic(after NMF)	45%	56%
Dynamic(after PPR)	56%	71%

PPR score

Figure4 also shows the role transitions of user nodes with high PPR vs low PPR.

3.3 Time Complexity

Let n be the number of nodes, f be the number of features, r be the number of roles and t be the number of timesteps. As described in [8] timecomplexity of NMF with multiplicative update is O(tnrf)
Feature extraction, summary statistics calulcation can be done in linear time. Neighbourhood of seed nodes can also be found in linear time by using either of breadth first, or depth first search.
PPR although it is quadriatic, but since number of nodes in each snapshots are less when compared to the whole graph. Therefore, applying PPR in each snapshot wouldnt be so harmful because of the benefit of predicition output it provides us.

4 Dataset

4.1 Dataset collection

We have written a web crawler using Python Scrapy framework.We crawled Restaurants reviews from yelp, and below are some of our observations:-

1. Yelp provides ground truth because it labels reviews as recommended, and not-recommended.
2. However, there is a problem with non-recommended reviews provided by Yelp. These reviews don't have a user profile connected to them, which makes it very hard to uniquely determine other reviews published by this user
3. We solved the above problem, by parsing various meta-data information embedded in html code of each review like username, number of friends of reviewer, number of reviews by reviewer. Folowing that, we wrote a script to automate searching on google, say strings like: 'Jolie S yelp user details 2 friends 80 reviews' Using above we get various yelp user profile links from google search, and based on the user information collected earlier like username, number of friends of reviewer, number of reviews by reviewer etc; we can successfully single out the corresponding user to the non-recommended review collected earlier.

   The problem with this strategy is that it does not the give us unique users for all the non-recommended reviews filtered by us. But as the dataset collected by us is very large, it still gives us an opportunity to gather a big enough dataset of non-recommended reviews for giving precision and recall of our approach used.

4.2 Dataset Size and Stats

Total Users = 207887
Total Restaurants = 8771
From: 10/12/2004 to 11/18/2014
Reviews : 2043484
Total Timestamps = 131
where one timestamp accounts for each month.

5 Evaluation

As already mentioned above, Yelp provide us a medium for ground truth data. In the past a number of fraud detection problems have been modeled with the help of dynamic graph. However, to our knowledge opinion spam problem has been modeled for the first time using dynamic graph approach.

Therefore, there has been no baseline to compare our results with. However,[9] was one of the first paper to model opinion spam problem in static graphs, henceforth we implemented its algorithm and ran it on our dataset in order to compare it's results against our algorithn. Results obeserved are shown in Table1.

As we can see the results for precision and recall, both of them drastically improved from NMF to PPR. A possible reason may be is that NMF didn't account for relationship between various user nodes i.e, how many common products they have reviewed together,indicator for group spam etc. However, PPR accounted for inter-connectivity so evident in the graph structure.

6 Conclusions

In conclusion, we have discussed about the approach which can be used to model Opinion Spam as a dynamic graph. We developed a method of role transition using Non-Negative Matrix Factorization. To improve the results from NMF, and to leverage the graph connectivity we adapted the Personalized Page Rank algorithm to our cause. Although, there has been no baseline in dynamic graphs, our method proved to have better precision and recall than our implemented method of static Graph technique in [9]. Possible future work could be that random walk in personalized Page rank can be modified in several works by choosing different instances of relationship Matrix, seed matrix in order to find various varieties of Group Spam.

7 References

1. Savage, D.; Zhang, X.; Yu, X.; Chou, P.; Wang, Q. 2014. Anomaly Detection in Online Social Networks. Social Networks.
2. Anamoly detection in Dynamic Networks, Ranshous, Shen, Koutra, Faloutsos, Samatova.
3. Akoglu, L.; Chandy, R.; and Faloutsos, C. 2013. Opinion Fraud Detection in Online Reviews by Network Effects. In Association for the Advancement of Artificial Intelligence.
4. Finding deceptive opinion spam by any stretch of the imagination.- Ott, Choi, Cardie Hancock. HLT '11 Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics
5. Spotting opinion spammers using behavioural footprints- Mukherjee, Kumar, Liu, Wang,Hsu, Castellanos, Ghosh. KDD13 Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining.
6. Fei, G.; Mukherjee, A.; Liu, B.; Hsu, M.; Castellanos, M.; and Ghosh, R. 2013. Exploiting Burstiness in Reviews for Review Spammer Detection. In Association for the Advancement of Artificial Intelligence.
7. Detecting group review spam Proceedings of the 20th international conference companion on World wide web, 2011. Arjun Mukherjee,Bing Liu, Junhui Wang, Natalie Glance, Nitin Jindal
8. Rossi, R.; Neville, J., Gallagher. B; and Henderson, K. 2013. Modelling Dynamic Behavior in Large Evolving Graphs. Association for Computing Machinery.
9. Wang,Xie,Liu,Yu, P. 2011. Review Graph based Online Store Review Spammer Detection. In ICDM
10. Hederson,Gallagher,EliassiRad,Tong,Basu, Akoglu Koutra,Faloutsos, Li, RolX structural role extraction and mining in large graphs. Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining
11. Local Graph Partitioning using PageRank Vectors: Anderson,Chung,Lang :2006 FOCS '06 Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science