Artificial Intelligence (CSE3013)

Anomaly Detection in Ego-Centric Networks

Sourish Banerjee | Souradeep Bhattacharya | Suvansh Dutta | Debdeep Mohanty

Submitted To

Prof. Annapurna J.

Project Documentation

Problem Statement

Anomalies in social networks arise due to irregular activities or sudden changes in interaction by individuals or a group of individuals in the network. These irregularities make such individuals different from the rest of their nodes in the network. Detection of such anomalies can be used to identify fraudulent and malicious individuals. Anomalies have a significant impact on the network structure, as several malicious entities often have distinct patterns of interaction. These patterns can be learned via several techniques in an attempt to detect such anomalies.

Introduction

As indicated by our problem statement, we intend to detect anomalous interactions or behavior in social networks, which may be indicative of even larger problems in the network. Depending on the specific domain or context, these interactions could indicate the presence of fraudulent individuals, spammers, or subversive activities like interactions of terrorist groups on social networks. Evidence of anomalous behavior with a high degree of accuracy can allow us to analyze and maintain social networks in a manner that keeps in check the above-mentioned individuals and activities.

The major types of anomalies in social networks are:

- Point anomalies
- Contextual anomaly – Unusual behavior given a certain context; might be normal in some other context
- Collective anomaly – A collection of related data instances is anomalous with respect to the entire set

Some common methods of detecting these anomalies are:

- Eyeballing metric visualizations – Effective, but does not scale
- Statistical Machine Learning
- Correlation of data (e.g. Pearson's)

There are various forms of anomalous behavior; a major one that we are interested in being outlier detection where abnormal behavior or characteristics in a dataset might often indicate that the person perpetrates suspicious activities. These could be of two types:

- Data Set Level – Where the behavior of a person/instance does not comply with the overall behavior.
- Data Item Level – Behavior of a person/instance does not comply with normal behavior of that person/instance.

If our system is setup specifically to detect fraudulent activity, it is fruitful to discuss the types of fraud that we might encounter. These are, broadly:

- Uncommon - <1% in volume. These affect the learning phase
- Well-considered
- Imperceptibly concealed
- Time-evolving – Incremental updates required
- Carefully organized

Finally, some of the required properties of a real-time fraud detection model are:

- High accuracy
- Operational efficiency – E.g. 6 second rule in credit card fraud
- Economic feasibility
- Interpretability

Literature Review

In their paper on anomalous user behavior in social networks, Bimal Vishwanath et. Al. try to use unsupervised anomaly detection techniques to search for suspicious behavior patterns, thus lowering the dependence on labeled examples. The technique used is principal component analysis (PCA), which searches for deviation from a normal standard of behavior on social networks. This use case for detecting anomalous behavior is a focus we adopt in our paper as well, though we think trained examples give us greater accuracy in our learning model. This paper also explicates the usage of anomalous behavior to take advantage of the crowd-sourced nature of interactions on social networks, which is a scenario where ego-centric networks could be formed which our model would be particularly well-placed to detect.

Mohammadreza Ebrahimi, Ching Y. Suen et. al. use semi-supervised anomaly detection to recognize predatory chat documents or for sexual predator identification (SPI). The problem they identify with the current approach is that a large number of predatory and non-predatory examples are needed to train a model for this purpose. Their system seeks to mitigate this by using a one-class support vector machine that doesn't need non-predatory examples for classification. In analysis of their system, they are able to show that though their system uses only one class label for classification, it is able to achieve a higher degree of accuracy in predictions than the two-class Naïve Bayes classifier, which was their baseline, as well as achieve a performance comparable to that of the binary SVM classifier. This provides a model for us in the type of behavior that we can target and our system also uses semi-supervised anomaly detection as we have information about some nodes but not all of them, thus helping us tailor our system to an important real-world application.

In their paper by Jing Gao, Feng Liang, etc. They study the problem of finding contextual outliers in an "information network". They regard each node in a network as an object, and there usually exist large amounts of information describing each object. They have developed an algorithm called community outlier detection algorithm (CODA). They propose a probabilistic model for community outlier detection in information networks. It provides a unified framework for outlier detection and community discovery, integrating information from both the objects and the network. It provides a unified framework for outlier

detection and community discovery, integrating information from both the objects and the network. The object information is described by some generative mixture model, and network information is encoded as spatial constraints on the hidden variables via a HMRF model. They validate the proposed algorithm on both synthetic and real data sets, and the results demonstrate the advantages of the proposed approach in finding community outliers.

In their paper on moving towards fully supervised anomaly detection, Nico Gornitz, Marius Kloft et Al. demonstrate how unsupervised anomaly detection often fails to meet the required detection rates and thus demonstrate the need for labeled data to guide model generation. They say that though anomaly detection is seen to be an unsupervised task as these anomalies are unpredictable and have unknown distributions, they say that a semi-supervised algorithm can be grounded on unsupervised anomaly detection methods and devised to overcome this problem. Additionally, they also propose an active learning strategy to label candidates. They are able to observe that this methodology requires much less examples than other state-of-the-art techniques, while providing higher detection rates. As our solution also attempts to implement semi-supervised anomaly detection, we used this paper to understand the development of such a model.

In their paper on evaluating Github's network using weighted community detection, Joseph Marrama and Tiffany Low used Girvan-Newman's community detection algorithm on a different type of social network, GitHub. They performed the community detection on unweighted graphs by converting them to weighted ones based on conductance, betweenness and modularity. This gave us the inspiration to do the same on Facebook's dataset of unweighted edges, based on common characteristics of users.

Leman Akoglu et. Al. have proposed the OddBall algorithm which spots anomalies in weighted graphs. They discuss the power laws based on density, weights, ranks and eigenvalues and how they can be used as rules to identify neighborhoods and anomalies. New features are developed which are applied to further detect anomalies. We have taken inspiration from this paper in our approach towards detecting anomalies and developing new features.

Methodology

Algorithm: Fraudulent Community Detection Approach

Our proposed solution is loosely based on the Gotch'All methodology for fraud detection in social networks. The algorithm calculates scores for the individual nodes as well as the detected cliques and uses them to form a model that can be used to detect anomalies in a social network. The algorithm is divided into three steps.

i. Individual Scoring

Initially, the edges are not weighted. We assign each edge a weight based on the number of common characteristics between the nodes connected by the edge. This helps us find ego-centers. We discretize the features of these nodes and use it to calculate the individual scores of the nodes.

ii. Clique Detection

In order to detect complete cliques, we search for the exact match for the clique in the community. Then, we delete the original clique. To detect partial cliques, we find partial overlaps and keep the original clique.

iii. Clique Scoring

Clique scoring is done using three parameters:

- Intrinsic Features - Intrinsic features are features which pertain to a particular node itself like age, gender, activity etc.
- Relational Features - Relational features indicate the interaction between two individuals.
- Clique Based Features

The features for the clique are the common characteristics among the nodes of that clique.

Implementation

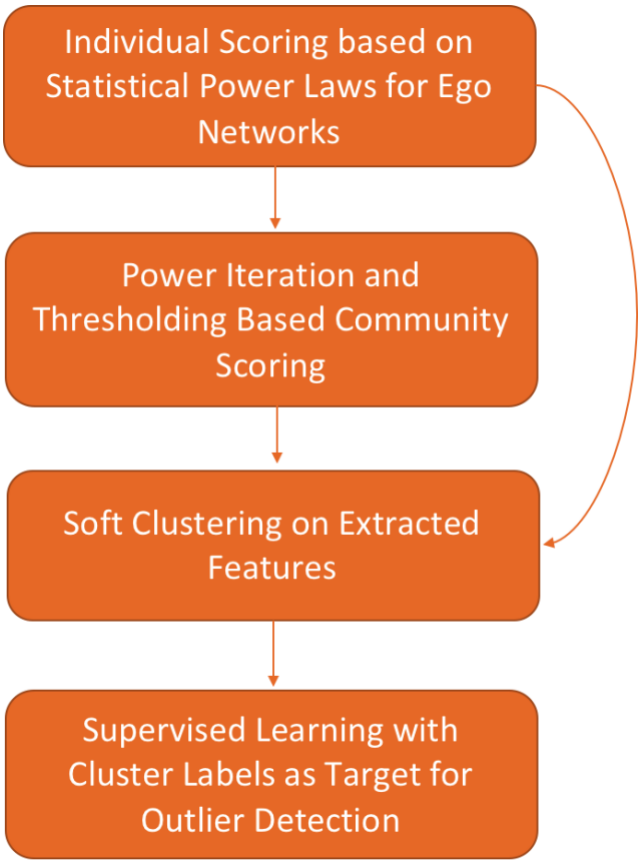
We have implemented this algorithm using Python on Jupyter Notebook. Our implementation attempts to extract features from the dataset, which will then be used to cluster the data into a model which will give us information about possible outliers, in addition to the nodes that are already known as definite outliers and then train a semi-supervised model based on these predictions for future outliers.

We have initially used three power laws, the edge weight power law, the edge density power law and the Eigen value power law, to form three features that allow us to test for particular types of anomalies (bursty edges, almost star sub-networks). We then iterated these similar to the Page Rank algorithm with separate initial weights as the initial outlierness scores, to determine whether these converge to a certain level. We eyeballed certain thresholds for all these features to determine outliers on graphs.

We then formed thresholded features for each of these, which would have not one but two thresholds, thus leading to three sets, with the idea being that we would be able to separate the nodes into safe, possibly outlier and definitely outlier sets. The final set of features was formed by calculating a score for each node from the weights of its edges raised to the power of the thresholded normal outlierness score for the edge target. What this would return is a higher value for those nodes near many outliers, which is an important characteristic for prediction.

This gave us our list of features which we used to create a dataset for analysis. At first, from a heat map of the different features, some interesting properties were observed. The iterated outlierness scores from the modified PageRank algorithm were all perfectly correlated even though they corresponded to different metrics, thus implying that they resulted in the same data for us, allowing us to discard 2 of them. Other sets also displayed high degrees of correlation, though these could be disregarded as the minute change in values represented high amounts of information in an essentially sparse dataset.

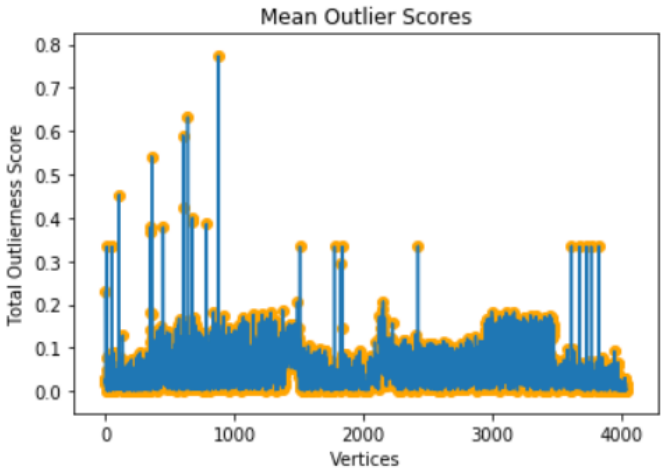
We then formed a Gaussian Mixture Model cluster using Principal Component Analysis to find the 2 components with maximum variance for easier visualization. The 2-centroid model proved to have the best returns (calculated using the silhouette score) as the 3-centroid model wasn't as clear, thus affecting our goal of 3 easily separable sets. We then performed semi-supervised learning on this dataset and compared the performance of several classifiers on test data, with SVC model showing peculiarly poor returns, but AdaBoost and random forest showing very promising returns. We then printed the possible outliers from the dataset with these predictions.

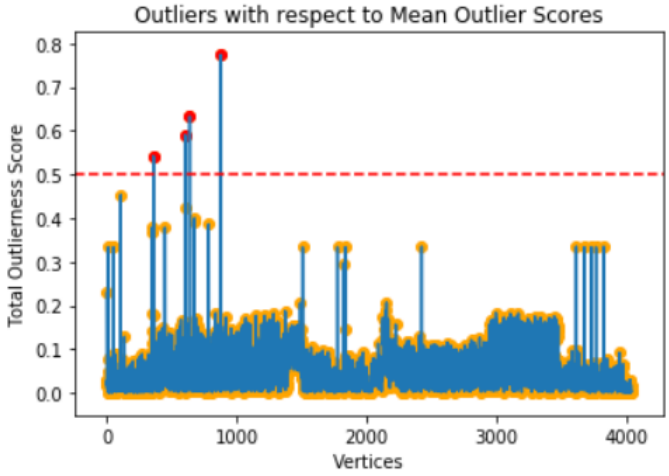


Flowchart Representation of Implementation Workflow

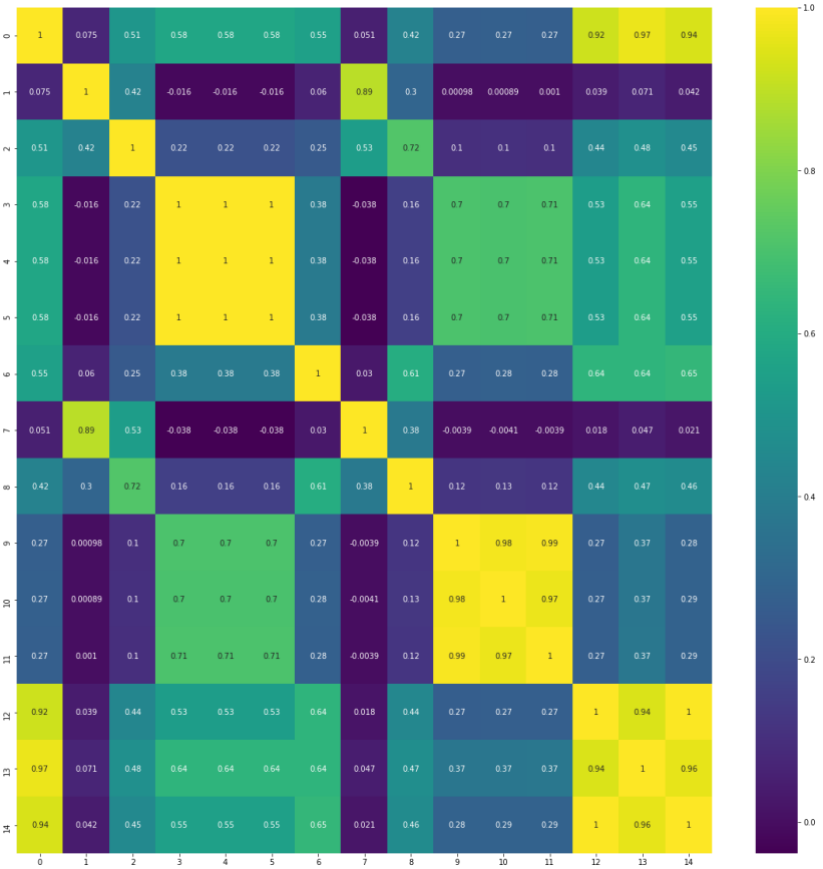
Results and Discussion

Following are some of the visualizations from our experiments. Each visualization has been clearly annotated and the corresponding significances has been explored in brief.

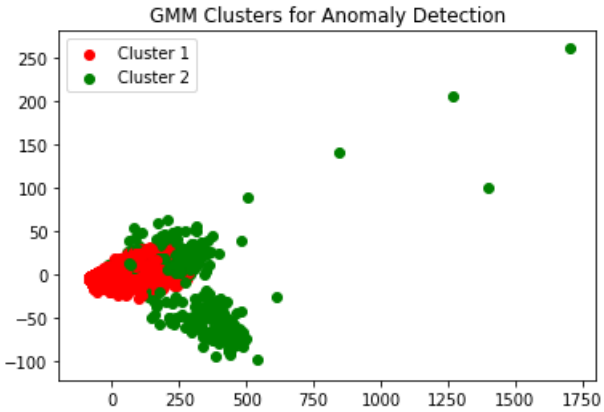




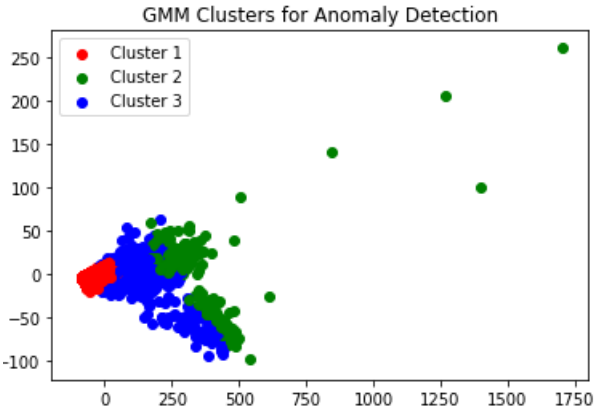
Eyeballed threshold and consequent outliers for the mean outlierness feature (an average of the outlierness due to the three power laws)



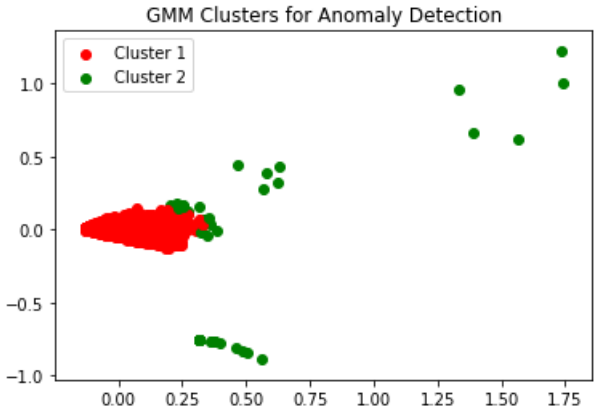
Heat map of the features formed from the dataset, exhibiting interesting properties (perfect correlation between the first 3 iterated features) and very high correlation between other sets



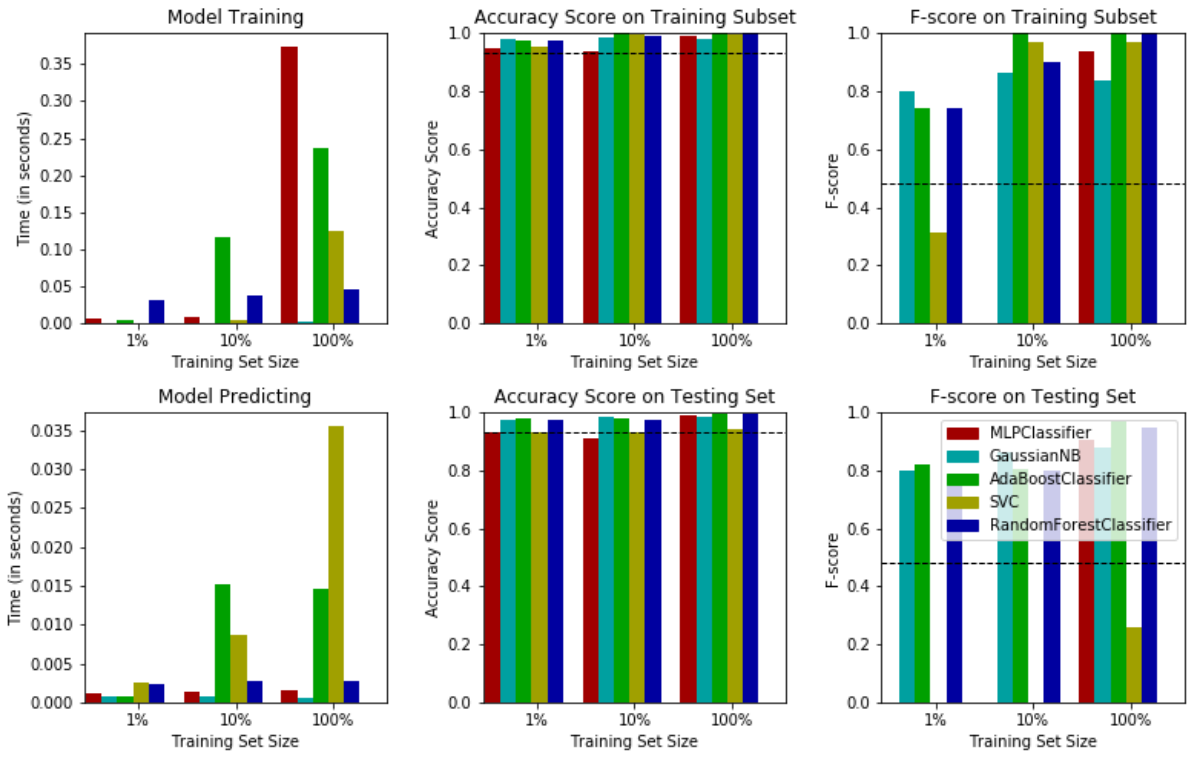
Gaussian Mixture Model clusters with 2 centroids showing good performance (very dense first cluster) and the green cluster showing known outliers and very possible outlier nodes



Performance with 3 centroids which is not very helpful as the blue cluster is very dense and doesn't help in prediction of possible outliers to a great extent



2-centroid model for the statistical features which doesn't give us any additional information over the eyeballed outliers (also validates the importance of the power iterated and thresholded features in the clustering)



A collection of the performance of all the classifiers used, demonstrating that SVC has particularly poor performance but that AdaBoost and Random Forest show particularly promising results

Conclusion

Our aim was to find a method that improves existing anomaly detection results in social networks. We were ultimately able to achieve promising results in this regard. Hence, we conclude that our proposed model is a viable approach for anomaly detection in social networks and highlights the scope for more specific and refined models in the wake of other domain specific datasets.

References

[1] Ebrahimi, M., Suen, C. Y., Ormandjieva, O., & Krzyzak, A. (2016). Recognizing predatory chat documents using semi-supervised anomaly detection. Electronic Imaging, 2016(17), 1-9.

[2] Marrama, J., & Low, T. Social Coding: Evaluating Github's Network using Weighted Community Detection.

[3] Horn, C. (2011). Social Network Analysis: Online Anomaly Detection and Graphical Model Selection.

[4] Viswanath, B., Bashir, M. A., Crovella, M., Guha, S., Gummadi, K. P., Krishnamurthy, B., & Mislove, A. (2014, August). Towards Detecting Anomalous User Behavior in Online Social Networks. In USENIX Security Symposium (pp. 223-238).

[5] Savage, D., Zhang, X., Yu, X., Chou, P., & Wang, Q. (2014). Anomaly detection in online social networks. Social Networks, 39, 62-70.

[6] Gao, J., Liang, F., Fan, W., Wang, C., Sun, Y., & Han, J. (2010, July). On community outliers and their efficient detection in information networks. In Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 813-822). ACM.

[7] Görnitz, N., Kloft, M. M., Rieck, K., & Brefeld, U. (2013). Toward supervised anomaly detection. Journal of Artificial Intelligence Research.

[8] Akoglu, L., McGlohon, M., & Faloutsos, C. (2010, June). Oddball: Spotting anomalies in weighted graphs. In Pacific-Asia Conference on Knowledge Discovery and Data Mining (pp. 410-421). Springer, Berlin, Heidelberg.