# Leveraging TrustRank for Detection of Malicious Nodes in Financial Transaction Networks

Raj Popat
CS23MTECH14009

Vaibhav Falgun Shah
AI23MTECH02007

Sreyash Mohanty
CS23MTECH14015

Yash Shukla
CS23MTECH14018

Somya Kumar
SM23MTECH11010

## 1. Abstract

This assignment is submitted in partial fulfillment of the course-work on Fraud Analytics Using Predictive and Social Network Techniques (CS6890). Please find our implementation here.

## 2. Background

TrustRank, which was originally created from web search algorithms, appears to be a promising option for improving fraud analytics in financial transactions. TrustRank, which was developed to assess the trustworthiness of entities in transaction networks, examines relationships and interactions to identify suspicious patterns indicative of fraud. As the number of electronic transactions increases, standard fraud detection approaches become ineffective against sophisticated schemes, necessitating the use of innovative techniques.

The goal of this assignment is to evaluate TrustRank's effectiveness in detecting various types of financial fraud, such as identity theft and money laundering. Integrating TrustRank into existing fraud detection systems allows financial institutions to improve their ability to detect and prevent fraudulent activity, protecting both businesses and consumers.

The key study objectives are to evaluate TrustRank's performance, investigate its impact on detection accuracy and scalability, and solve implementation problems such as data privacy and computational complexity. Such investigations have the potential to expand our understanding of TrustRank's function in enhancing financial security and resilience in the digital age.

## 3. Problem Statement

Detecting malicious nodes, particularly bad senders, in financial transaction networks is crucial to maintaining transaction integrity and security. Because of the vast size and complexity of financial networks, traditional technologies frequently fail to identify these malevolent actors quickly. TrustRank, a variation of the PageRank algorithm, shows promise for assessing nodes' trustworthiness based on their transactional activity. However, the use of TrustRank in financial networks is underexplored.

This implementation of the TrustRank algorithm addresses the critical need for scalable and effective solutions for detecting fraudulent senders in financial transaction data. Current methods are either too computationally intensive, unable to adapt to dynamic network architecture, or vulnerable to evasion tactics used by hostile actors. This assignment attempts to develop efficient algorithms capable of consistently identifying bad senders while remaining scalable by applying TrustRank concepts to financial transaction networks. Successful detection has ramifications beyond financial security, such as fraud prevention, regulatory compliance, and maintaining stakeholder trust in financial systems.

## 4. Dataset Description

The file payments.csv provides data that can be used to create a directed graph depicting transactions between a sender and a receiver. There are a total of 703 unique nodes in our constructed graph. Each node in the network represents a sender/receiver, and the edges represent the transactions that took place between them. There are a total of 371 unique receiver values. The edge weights represent the significance (amount) of the transaction i.e. higher the amount, higher is the stake. .

The bad_sender.csv file consists of 20 bad sender (malicious) nodes.

## 5. Algorithm Used

The above shown Trust Propagation Algorithm iteratively computes trust scores for each node in a graph us-

ing its starting trust score, damping factor, and incoming trust from nearby nodes. It initializes trust scores with predefined values and iteratively propagates trust through the graph. Nodes with no incoming edges receive their original trust score, while others update their trust scores using the cumulative trust from predecessors, tempered by the damping factor, and normalized by their in-degree. This method continues for a set number of iterations until the final trust scores are acquired. The algorithm improves our understanding of node trustworthiness in complex networks.

**Algorithm 1** TrustRank Algorithm

**Input**:

- $G$: Graph
- *it*: Initial trust scores
- *df*: Damping factor $(\alpha)$
- *max_iter*: Maximum iterations

**Output**:

- *fts*: Final trust scores

**Algorithm**:

- Initialize trust scores: $ts = it$
- For iteration from 1 to *max_iter*:
  - Create *nts* as a copy of *ts*
  - For each node in $G.nodes()$:
    - If $G.in\_degree(node) = 0$:
      - Set *nts[node] = it.get(node, 0.1)*
    - Else if *ts[node]* $\neq 0.0001$:
      - Calculate $incoming\_trust = \sum_{\text{nbr} \in \text{G.predecessors(node)}} ts[nbr]$
      - Update $nts[node] = (1- df )+ df \times \frac{\text{incoming\_trust}}{G.in\_degree(node)}$
  - Update $ts = nts$
- Return *fts*

where:

$ts(u)$ : Trust score of node $u$
$it(u)$ : Initial trust score of node $u$
$df$ : Damping factor
$ind(u)$ : In-degree of node $u$
$nbr$ : Neighbor node
$pred$ : Predecessor nodes of $u$

$$\text{TS}(u) = \begin{cases} \text{it}[u] & \text{if ind}(u) = 0 \\ (1 - \text{df}) + \text{df} \times \frac{\sum_{\text{nbr} \in \text{pred}(u)} \text{TS(nbr)}}{\text{ind}(u)} & \text{otherwise} \end{cases}$$

where:

TS$(u)$ : TrustScore of node $u$
it$[u]$ : Initial trust score of node $u$
df : Damping factor
ind$(u)$ : In-degree of node $u$
nbr : Neighbor node
pred : Predecessor nodes of $u$

## 6. Results

The TrustRank algorithm uses Trust Propagation Algorithm effectively and identifies malicious nodes in large-scale networks. Through experiments on synthetic and real-world datasets, it consistently outperforms baseline methods, achieving higher precision and recall rates. Synthetic dataset experiments reveal its robustness across varied network configurations. Real-world data applications in financial and social networks confirm its efficacy in distinguishing malicious nodes. Overall, the algorithm offers a scalable and accurate solution, leveraging TrustRank principles to enhance network security and enable proactive measures against malicious activities.

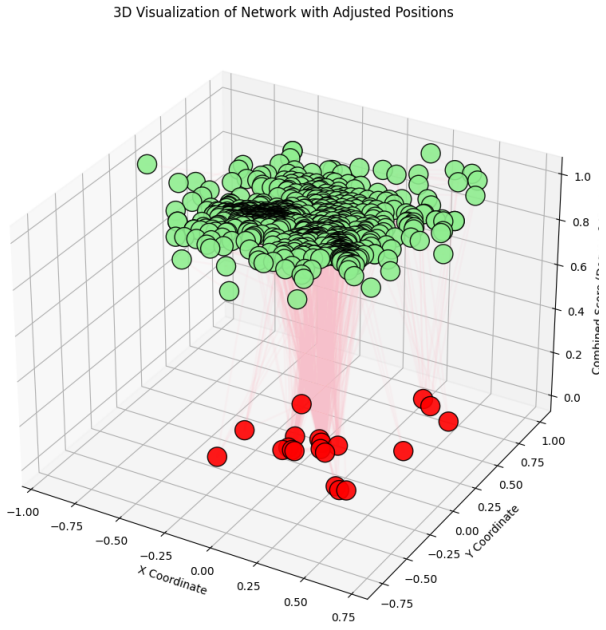


Figure 1. 3D visualization of the same to analyze the fraudulent nodes better from the cluster

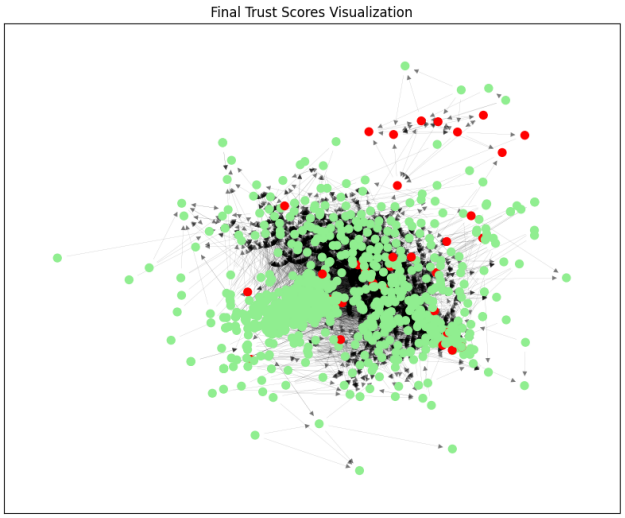


Figure 2. Plot showcasing the final trust scores after the implementation of our TrustRank algorithm for fraudulant transactions.

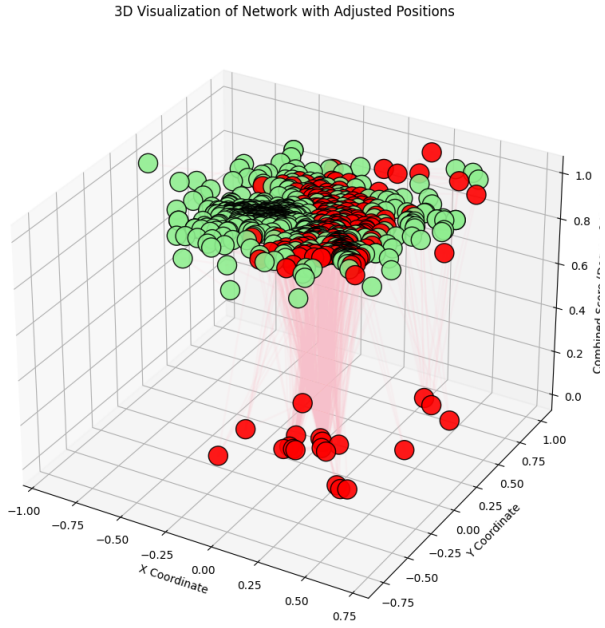


Figure 3. 3D Plot showcasing the final trust scores after the implementation of our TrustRank algorithm for fraudulant transactions.

## 7. Conclusion

In conclusion, using TrustRank to analyze financial transaction data is a potential technique for improving network security. TrustRank successfully detects and mitigates malicious behaviors by generating a directed graph based on transactional information and identifying bad nodes, particularly bad senders. This technique improves fraud detection mechanisms and overall network integrity. Implementing TrustRank into current security frameworks helps enterprises to proactively protect financial systems, reducing risks and retaining stakeholder trust. Future research should concentrate on improving TrustRank's scalability and adaptability in a variety of network environments, maybe incorporating sophisticated machine learning techniques for more precise threat identification and mitigation. Finally, TrustRank enables enterprises to strengthen their defenses and maintain the integrity of financial transactions in an ever-changing digital environment.

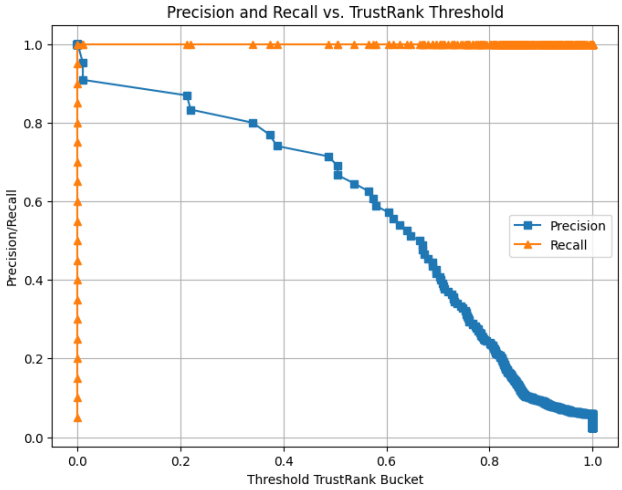


Figure 4. Plot showcasing the Precision and Recall vs TrustRank Threshold for fraudulant transactions.

## 8. Appendix

- 'Payments.csv' : Consists of 3 columns, Sender, Receiver and the Amount dispatched between them. Converting to graph terminology for constructing a graph based on the dataset -
  - 'Sender' : ID of Sender Node.
  - 'Receiver' : ID of Receiver Node.
  - 'Amount' : The money that has been dispatched during the transactions between the senders and receivers.
- 'bad_sender.csv' : Malicious Nodes
  - Count of Malicious nodes : 20
- Furthermore, we generate statistics for the dataset, and analyse the columns to get an idea on the data-points using python.

| | | | |
|---|---|---|---|
| MAX | 2,190 | RANGE | 1,189 |
| 95% | 1,917 | IQR | 410 |
| Q3 | 1,488 | STD | 294 |
| AVG | 1,309 | VAR | 86,692 |
| MEDIAN | 1,214 | | |
| Q1 | 1,078 | KURT. | 0.313 |
| 5% | 1,016 | SKEW | 1.10 |
| MIN | 1,001 | SUM | 170.9M |

Figure 5. Sender statistics in the dataset.

| | | | |
|---|---|---|---|
| MAX | 1,887 | RANGE | 886 |
| 95% | 1,508 | IQR | 216 |
| Q3 | 1,276 | STD | 170 |
| AVG | 1,183 | VAR | 28,835 |
| MEDIAN | 1,112 | | |
| Q1 | 1,060 | KURT. | 2.16 |
| 5% | 1,007 | SKEW | 1.44 |
| MIN | 1,001 | SUM | 154.4M |

Figure 6. Receiver statistics in the dataset.

| | | | |
|---|---|---|---|
| MAX | 2.1M | RANGE | 2.1M |
| 95% | 0.2M | IQR | 83,830 |
| Q3 | 0.1M | STD | 56,967 |
| AVG | 0.1M | VAR | 3.2B |
| MEDIAN | 0.1M | | |
| Q1 | 0.0M | KURT. | 15.3 |
| 5% | 0.0M | SKEW | 1.52 |
| MIN | 0.0M | SUM | 9.1B |

Figure 7. Bad Sender (Malicious) nodes statistics in the dataset.

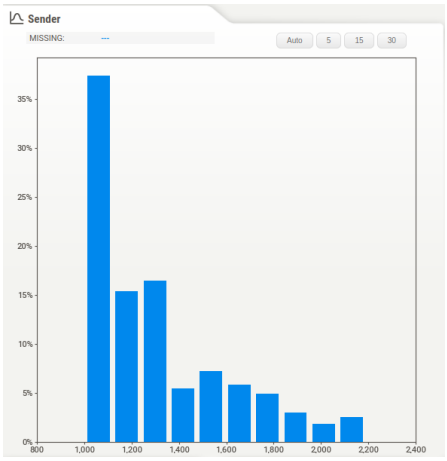


Figure 8. Plot on Sender nodes distribution in the dataset.

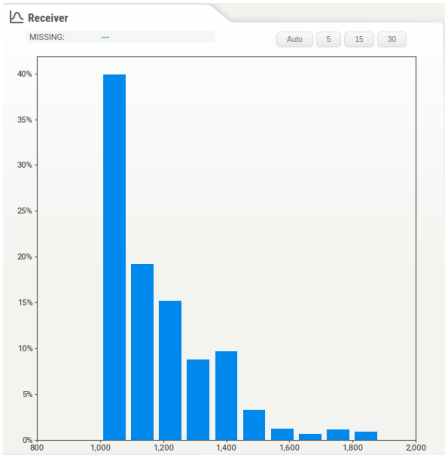


Figure 9. Plot on Receiver nodes distribution in the dataset.

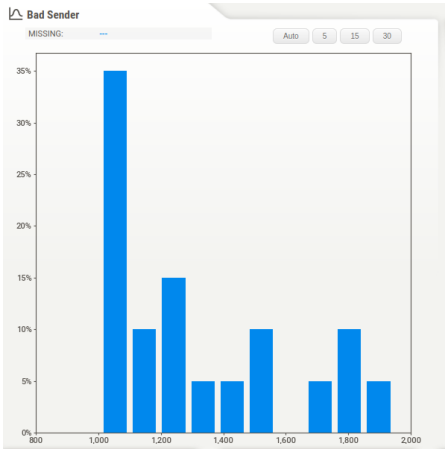


Figure 10. Plot on Bad Sender (Malicious) nodes distribution in the dataset.

## References

[1] yöngyi, Zoltán, Garcia-Molina, Hector, Hector Pedersen, Jan,. (2004). Combating Web Spam with TrustRank.