Community structure-aware fairness and goodness algorithm for link weight prediction

Hiba Rashid Atiya¹, Dr. Huda Naji Nawaf²

¹,² Information Networks Dep. College IT, University of Babylon, Hillah, Babylon, Iraq.

hibaalmamoori@student.uobabylon.edu.iq
halmamory@itnet.uobabylon.edu.iq

Abstract. In this research, the problem of predicting the edge weight in the Bitcoin network has been resolved by utilizing the structure of the community based Fairness and Goodness. Community detection using Newman-Girvan algorithm has been applied to obtain the trusted communities which depends on the features representation generally. The former may represent the trusted transactions that have trust value more that certain threshold. Concerning the missing edge weight, the prediction is based on the fairness and goodness that supported by the above communities. In fairness measure, can capture how fair the node is in rating other nodes’ trust that has transactions (trusted transactions) with. While the goodness of a node shows how much this node is liked or trusted by other nodes that have transactions (trusted transactions) with. In both cases, only the nodes that are within community of the target node contribute with setting the fairness or the goodness of the target. The model was evaluated in practice using two real-world datasets; the Bitcoin-OTC and the Bitcoin-Alpha datasets. The experimental findings explain the effectiveness of the proposed comparable with other methods. The percentage error minimization is 18% for the Alpha dataset and 26% for OTC dataset.

Keywords. Bitcoin network, Girvan-Newman algorithm, edge weight prediction, signed networks, Fairness and Goodness, Trust Network.

1. Introduction

The term Bitcoin is a collection of concepts and technologies which form the basis of an ecosystem of digital money. Bitcoin is a peer-to-peer, fully distributed protocol [1].

Satoshi Nakamoto says “the root problem with conventional currency is all the trust that’s required to make it work. The central bank must be trusted not to debase the currency, but the history of fiat currencies is full of breaches of that trust” [2]. In fact, Bitcoin is one of the most common cryptocurrencies in today's age which is considered as a weighted signed network (WSN) comprising ratings between users [3]. However, the WSN is not limited to Bitcoin but also includes other networks, and consequently the sense of the sign and the strength of the edges is different with different networks. For instance, for airline networks, edge weight means the number of flights [4], the number of available seats [5], or the number of passengers traveling between two airports [6]; while, in some social networks, link weight may capture the strength of friendship [6].

Beyond doubt, WSNs, like any network, is unfulfilled. In other words, there are many relations among users that have not been weighted. Concerning to Bitcoin network, the anonymization nature of it prompted a need to identify untrusted Bitcoin users to avoid risky transactions [7]. In this setting, trust is a fundamental necessity since any user can initiate a transaction, receive money from the other
user, and never send money back in the other currency. To achieve a web of trust, users rate each other after a successful or unsuccessful transaction. The advantage of using these smaller networks is that each edge corresponds to some weight, the rating from user u to user v. Doing so, forms a web of trust between users, and it allows two users who do not know each other to perform a transaction based on the aggregated trust that they each possess [8].

The main challenge is how to predict these weights accurately. Prediction of such weights is meaningful for different reasons. In this work, the community structure has been leveraged to support the predict the weights of relations among nodes.

2. Related Works

The prediction of weights is not limited to Bitcoin networks only, however link weight prediction may be useful to develop traditional tasks in signed networks such as anomaly detection [9], network analysis [10], node ranking [11], etc.

This section is providing the most essential research articles in this field where various approaches have been proposed to predict the link weight. The related work is revised from oldest to latest.

In the weight prediction proposed in signed networks [12] has been considered as the baseline method, the authors used 11 special properties which include triadic balance, deserve, and bias, along with their proposed new characteristics of node behavior called fairness and goodness measures. In principle, supervised regression has been implemented to predict the weights on the edges. The node's goodness conceptually measures how much this node is trusted/liked by other nodes, while the node's fairness measures how it is fair in rating the possibility or trust level of other nodes. The standard definitions of goodness and fairness are as follows,

\[
g(v) = \frac{1}{|\text{in}(v)|} \sum_{u \in \text{in}(v)} f(u) \times W(u, v) \quad \ldots (1)
\]

\[
f(u) = 1 - \frac{1}{|\text{out}(u)|} \sum_{v \in \text{out}(u)} \frac{|W(u,v) - g(v)|}{R} \quad \ldots (2)
\]

Fairness values are usually in the interval [0,1] and goodness values are in the interval of [-1, 1]. The highest possible difference in the scale of the score is between edge weight and goodness value differential. Where \(W(u, v)\) is the weight of link \(u \rightarrow v\), and in \((v)\), out \((u)\) are the set of incoming and outgoing neighbors and \(R=2\). The weight of the link \((u, v)\) can be predicted by-product the fairness \(f(u)\) and goodness \(g(v)\). \(W(u, v) = f(u) \times g(v)\). This approach is efficient and reliable but they assume it can be limited to using goodness and fairness (with only 2 ways of good or bad) for weight prediction.

The main idea of [7] is mapping node properties into vectors and using the inner product to predict the weights of the link. Influenced by collaborative filtering, the weight matrix of the network is approximated by matrix decomposition, and optimized by gradient descent. Predict weights of the link using vectors for node \(u\). The inner product of the vectors \(p_u^T q_v\) has been used to predict the \(W_{uv}\) and to model their similarity using cosine distance, where \(p_u \in \mathbb{R}^k\), and \(q_u \in \mathbb{R}^k\). In fact, \(P = F \in \mathbb{R}^{n \times 1}\), \(Q = G \in \mathbb{R}^{n \times 1}\), follows the same concept as the Fairness and Goodness algorithm, where \(P\) represents the fairness of each node (how fair is a node in rating others), and \(Q\) denotes the goodness of each node (how probably is the node to be trusted by other nodes).

\[
P = (p_1, p_2, \ldots, p_n)^T \in \mathbb{R}^{n \times K} \quad \ldots (3)
\]

\[
Q = (q_1, q_2, \ldots, q_n)^T \in \mathbb{R}^{K \times n} \quad \ldots (4)
\]

The goal is to find \(P, Q\)

\[
W = PQ \quad \ldots (5)
\]

Where \(W \in \mathbb{R}^{n \times n}\) denotes the weight matrix of a WSN, and \(W_{uv}\) is the weight of link \(u \rightarrow v\) in the network. In the case of \((f(u)g(v)) \neq \text{sign}(W_{uv})\) or minimizing \(|f(u)g(v) - W_{uv}|\), the model metrics can be enhanced by correcting the wrong predicted signs. Consequently, apart from finding the fairness and goodness metrics \(P\) and \(Q\), the model allows the weights of the known links to be corrected to the fairness and goodness scores. The parameters in their model are initialized first with the result of the Fairness-Goodness method (rather than initializing all parameters to 0), and gradient descent is carried out.

The authors in [13] consider two sets of features; edge-to-vertex dual graph features and fairness goodness scores in order to suggest a supervised framework for weight prediction that uses fewer features than those used previously in literature. Based on the edge-to-vertex dual (line graph) that has
been used by [14], they extend their approach to directed weighted signed networks to provide the first group of features. In their proposed work, the node centrality indices based on path has been chosen such as page rank and betweenness centrality as features on the line graph that are relevant for directed graphs. As for the second group of features, they find that the unsupervised measures such as Fairness and Goodness measures are good candidates for graph features. Given those two groups, they propose a supervised regression approach. The work done by [12] has been considered as the baseline method. Particularly, the proposed model achieved superior weight prediction scores with a significantly lower number of features.

The LookLike algorithm [15] was introduced to expect trust between 2 users with similarity measures. The outcomes have shown that there is a significant association between the similarity of users and their opinions on trust networks. Firstly, an algorithm has been defined to identify the most similar trustor node between the trustee neighbors, next returns the trust value between the trustor and the trustee. Secondly running the first step of the algorithm for all data with different similarity measures (Jaccard’s Coefficient, Common neighbor, Preferential attachment, Adamic/Adar) and save them as new features. Lastly, check diverse supervised classifiers on these new features to predict that trust or distrust is the relationship between the two nodes.

The Weight Perturbation and Latent Factor (WPLF) algorithm [16] is primarily made of two parts, the perturbation of the observed weight information and the abstraction of the potential factors. The proposed model is best explained in three phases. Firstly, a link weight matrix is made by perturbing the structure of an observed weighted network. Secondly, a link weight matrix is reconstructed with the factorized latent factors of the observed network. Lastly, the two created matrices are consolidated to acquire the predicted weight of each missing link. Enlivened by the weight consistency and latent features of the weighted networks, they propose an edge weight prediction model on the basis of weight perturbation and matrix completion, which is along these edges called the WPLF algorithm. Analytic and simulation results propose that joining the weight consistency of the network and the link weight-related latent factors of the nodes is a precise real approach to resolve the link weight prediction difficulty.

The Directed Edge Weight Prediction Model (DEWP) [17] using decision tree ensembles is proposed for Industrial Internet of Things(IIoT). It needs to extend the metrics of local similarity to Directed Weighted Networks (DWNs) and summaries a series of metrics of similarity as features of each edge between nodes. These characteristics are used as the final predictor of edge weights to construct a four mixed regression model of random forest, gradient boost decision tree, eXtreme gradient boosting, and light gradient boosting machine, which can significantly reduce overfitting. Without relying on the nodes’ private attribute information, it uses network topology. They layout a set of similarity metrics in detail that incorporate directional factors with the classic local similarity indices, then use them as link characteristics to train the prediction model of edge weights. DEWP calculates indices of similarity between node pairs and then utilizes supervised learning to build a high precision edge weight model. Finally, all weighted edges in the network G have been scanned, then the indices of local similarity calculated. x represents the edge generator for each edge in E, and y is the receiver for the edge.

3. Newman-Girvan Algorithm
Historically, the Girvan-Newman algorithm is significant because it is used in the area of community detection as the beginning of a new era, a top-down hierarchical community detection algorithm suggested by Girvan and Newman, but it requires a time $O(n^3)$ on a sparse graph. It is a divisive hierarchical clustering method works on the opposite way of Hierarchical Agglomerative Clustering (HAC) where the number of shortest paths passing through the edge. The former is called edge betweenness and is a generalize of the central betweenness of the vertex which specifies the impact of the vertex on other vertices of the network [17]. In other words, vertex betweenness is the number of shortest paths that pass through the vertex, while the edge betweenness is the number of shortest paths that go through the edge endpoints [18]. When 2 geodesic paths (shortest) cross a given pair of vertexes, then each one counts as half of a path, and likewise for 3 or more [19].

In practical applications, this algorithm with betweenness centrality provides better outcomes than other centrality measures [20], although the Girvan-Newman algorithm cannot discovery overlap communities, as each node is assigned to a single cluster.
In practice, the Girvan-Newman algorithm starts by computing the edge betweenness of all the edges in weighted graph by applying the betweenness algorithm mentioned in [20]. It is important to say, that when two or more edges with the same highest betweenness, they have to be removed both. Then, the betweenness of all edges is recalculated on the remaining network and the process is repeated. In fact, the recalculation is necessary for the proper operation of the algorithm [17], as it allows for the common situation in which there is more than one edge between a given pair of community.

The main steps of the algorithm as illustrated below [17]:
1) Calculate edge betweenness for every edge in the graph.
2) Remove all edges with the highest edge betweenness.
3) Recalculate edge betweenness for remaining edges.
4) Repeat 2-4 until the graph becomes empty.

Firstly, the algorithm calculates the edge betweenness in a network based on the betweenness theory as in equation (6).

\[ B_{ui} = \sum_{u_j \neq u_i \neq u_k} \frac{\sigma_{u_j u_k(u_i)}}{\sigma_{u_j u_k}} \]  \[ \ldots (6) \]

where \( B_{ui} \) is the betweenness centrality value at node \( u_i \), \( \sigma_{u_j u_k(u_i)} \) is the number of the shortest path between node \( u_i \) and \( u_k \) that pass through the node \( u_i \). As for \( \sigma_{u_j u_k} \), it is the number of the shortest path between node \( u_j \) and \( u_k \). Therefore, if the edges with high betweenness scores have been removed that leads to eliminate the edges of the intercommunity and leave only the communities themselves [19].

Girvan-Newman algorithm's main problem is that it's not truly an algorithm that has a graph as input and a community structure as output. In other words, if the labeling of the graph vertices is reconfigured, then the outcome of the Girvan-Newman algorithm may change. Therefore, a given input (which would be a result of any algorithm) does not have a unique output. New modification modifies this drawback of the Girvan-Newman algorithm so that the community structure does not depend on the vertices of the graph being labeled. Additionally, this alteration can decrease the number of operations of the Girvan-Newman algorithm. But, it relies greatly on the graph found, where there are graphs for which no improvements can be made [18].

Modularity is described as the difference in the value gained by the fraction of the edges inside the module to the predicted fraction if the edges are distributed randomly. The numerous modularity optimization approaches include greedy strategies (hierarchical clustering), spectral optimization, extreme optimization, and simulated annealing. Newman-Girvan is the most common modularity method used, in which each network partition is divided into n disjoint modules, the value \( Q \) is computed in each step of the divisible algorithms. The highest value \( Q \) gives us the graph's best partition of the graph [21].

Newman has generalized the modularity maximization equation [22]:

\[ Q = \sum_{k} \sum_{(i,j) \in E_k} A_{ij} - \frac{d_i d_j}{2m} \]  \[ \ldots (7) \]

Given a network with \( m \) edges, the expected number of edges between two nodes \( i \) and \( j \) with degrees \( d_i \) and \( d_j \) respectively is \( d_i d_j / 2m \). As for \( A_{ij} \), it is the link weight between node \( i \) and \( j \). Basically, when, the value of the modularity \( Q \) larger, the partition is better.

4. How Bitcoin Works

Bitcoin works like an electronic cash kind where the people can buy Bitcoins on special websites. To issue Bitcoins for the given account, one needs to know the public key of the person. On the other hand, it is essential to keep the private key that can encrypt transactions. When a person sends and receives digital currency, its identity is not identified. Nevertheless, achieving acceptable anonymity with Bitcoin can be challenging [23].

These transactions are defined by: M outputs, each of which is correlated with: the amount of Bitcoin, the public key (address), N inputs, each referring to the output of a previous transaction (thus implicitly to the amount of Bitcoin and public key), a timestamp of approximate value [25].

Every Bitcoin transaction comes with its transaction ID, which is expressed as a string of letters and numbers. The transaction input refers to the Bitcoin address from which the funds are sent. While, the
transaction output refers to the Bitcoin address to which the funds are sent. Finally, the transaction value refers to the quantity of Bitcoins that were sent in a transaction. To maintain the integrity of accounting, the input value to a transaction must equal the output value [24]. In the Bitcoin system, if anyone wants to make a transaction then, each owner transfers the Bitcoin to another by digitally signing a hash of previous transaction and the next owner's public key and adding these to the end of the coin. A payee can verify signatures to verify ownership chain as shown in figure 1. It uses (Elliptic Curve Digital Signature Algorithm) ECDSA, which is based primarily on mathematics [25].

![Figure 1: Bitcoin Transaction [26].](image)

Thousands of computers-named "miners" are working together to secure the system. To motivate people to enter their computer into the network, the system offers opportunities by giving bonuses. A reward is given approximately every 10 minutes, and its value is halved after the creation of a certain amount of blocks ±210,000-. This ensures Bitcoins supply increases at a given (not linear) rate until a total of 21 million Bitcoins have been generated. When the maximum is reached, the reward of the miners consists of only transaction fees [27].

Worth to mention, miners are a class of unique network workstations that require substantial computational power and energy resources. They are responsible for block creation and for having valid transactions in them. In exchange, Bitcoins are awarded to the miners and they also earn transaction fees for all of those transactions included in the block that is created. Furthermore, miners compete in a race to be the first to create a block and receive the corresponding reward. If a block is created, a message is broadcasted to the miners and nodes network, which verifies the block's validity. If found to be valid, the nodes incorporate the block into their blockchain copy and the miners also. The miners will abandon their work on the current set of transactions and start working with transactions from the pool to create the next block, waiting to be integrated into a block [1].

5. **Community Structure Based Weight Link Prediction**

Mainly, the proposed algorithm involves two main stages.

5.1. **Newman-Girvan Algorithm**

Firstly, the Newman-Girvan algorithm is used for detecting Bitcoin communities. Essentially, the quality of the detection algorithm depends on how the features of nodes are represented. Thus, the features of Bitcoin nodes have been modeled in different ways.

Weight matrix or adjacency matrix is considering as a core of the Newman-Girvan algorithm. Generally, the matrix $A$ (see eq. 7) based on the representation of nodes.

5.1.1. **Similarity Measures.**

To quantify the extent to which users in the Bitcoin network resemble each other in terms of a rating to others, it is necessary to use the similarity measures which rely on the data type. In our scenario, the
features have been represented as categorical and numerical ways. As a result, Jaccard and cosine similarity have been applied by data type. In the case of the categorical representation of features, the Jaccard similarity has been applied, however the Cosine similarity has been applied in the case of numerical representation of the features.

All the outgoing edges have been extracted to represent the profile for each user. Doing so generates communities of the shared transactions among users in Bitcoin networks. To get more trust communities in terms of transactions, only outgoing edges that have been rated (>1) once and (>3) again are kept for each user to represent the profile in another way. Worthy to mentioned the attributes of nodes in both ways above have been represented as categorical form.

Besides, the profile can be created in terms of weights value of the edges, the outgoing edges rated (>1) are considered in an experiment. In other words, the node is represented as a numeric form.

Let u and v are two users, where u and v have n and m relations transactions respectively:

\[
u = \{w_1, w_2, w_3, \ldots, w_n\}
\]
\[
v = \{w_1, w_2, w_3, \ldots, w_m\}
\]

Pearson correlation (r) coefficient can be used to compute the similarity among users as follows:

\[
r(u, v) = \frac{N(\sum u - \overline{u})(\sum v - \overline{v})}{\sqrt{\left[N(\sum u^2 - (\sum u)^2)\right]\left[N(\sum v^2 - (\sum v)^2)\right]}}
\]  

where N represents the length of the profile. As well as, the Cosine similarity has been used to calculate the similarity matrix as below:

\[
cos(u, v) = \frac{u \cdot v}{\|u\| \times \|v\|} = \frac{\sum_{i=1}^{N} u_i \times v_i}{\sqrt{\sum_{i=1}^{N} u_i^2 \times \sum_{i=1}^{N} v_i^2}}
\]

Pearson and Cosine similarity have been used with the numeric representation of node.

Finally, Jaccard index is also applied, however for categorical representation of attributes:

\[
J(u, v) = \frac{|u \cap v|}{|u \cup v|}
\]

In practice, the similarity matrix \(T\) that obtained from the previous step can be used in this stage rather than \(A\). Actually, \(T_{trans}\) is representing the transactions among users in networks whether they are categorical or numerical form.

By applying equation (6), all edge betweenness among nodes can be obtained. Then, the modularity \(Q\) is computed using equation (7)

\[
Q = \frac{1}{4m} \sum_{i=1}^{k} \sum_{u_i \in C_i, \forall u_j \in C_i} \left( T_{trans}(u_i, u_j) - \frac{d_{u_i} d_{u_j}}{2m} \right) s_{u_i} s_{u_j}
\]

Where \(T_{trans}(u_i, u_j)\) represents the similarity in terms of shared trusted transactions or the weight of shared trusted transactions between the two users \(u_i\) and \(u_j\) when the node representation is binary or numeric form, respectively. Practically, most of the resultant communities are small which may not exceed two or three customers, and few number of them comprising hundreds or thousands of customers.

6. Community Structure Based Fairness and Goodness Algorithm

This model is to apply the fairness and goodness algorithm by considering the trusted communities. In this context, can capture how fair the node is in rating other nodes’ trust that has transactions (trusted transactions) with. Concerning the goodness of a node shows how much this node is liked or trusted by other nodes that have transactions (trusted transactions) with. In community context, the fairness and goodness computation relies on the detected communities. In another meaning, only the nodes that are within community of the target node contribute with setting the fairness or the goodness of the target.

Under these conditions, fairness and goodness measures are modeled for each node based on its community. However, the prediction of edge weight based on whether both ends of the edge belong to the same community or not. If so, the former measures are calculated using only the members of the community, otherwise the measures are computed to both ends, each in its community.

As mentioned before, the resultant communities are small inherently. Thus, these communities cannot be used to calculate the fairness or goodness measures of the target node. As a result, the
measures are calculated for the target node that belongs to a community that smaller than a certain threshold by taking into consideration the whole network. In other words, the fairness and goodness computation of the target node that belongs to a small community is not limited by only its edges within that community, however by using its edges within the whole network.

Given community structure of a network as a set of $k$ communities:

$$C_l, l = 1, 2, ..., k$$

Let $u$ and $v$ the source and destination of an edge, knowing that the both $u$ or $v$ belong to a large community size:

$$g(v) = \frac{1}{\left|\text{in}(v)_{\text{ev}}C_l\right|} \sum_{u \in \text{in}(v)} f(u) \times W(u, v) \quad ... (12)$$

$$f(u) = 1 - \frac{1}{2\left|\text{out}(u)_{\text{ev}}C_l\right|} \sum_{v \in \text{out}(u)} |W(u, v) - g(v)| \quad ... \quad (13)$$

Yet, if the either nodes belong to the small size community, then fairness and goodness are calculated taking into account the whole network using equations (1) and (2).

7. Implementation and Results

The datasets (Bitcoin networks) that have been used in this work are available in $^1$, which are the first public weighted signed network. A weighted signed network is a directed weighted graph. Bitcoin Alpha (Alpha for short) and Bitcoin-OTC (OTC for short) are two major Bitcoin rating datasets. Both datasets are a user-to-user trust. In addition, the dataset has been provided with a time of rating, measured as seconds. The dataset starts on August 8th, 2010, and ends on January 25th, 2016, starts on November 8th, 2010, and ends on January 22th, 2016, OTC and Alpha respectively. The table 1 explains the description of the dataset.

| Network | #Nodes | #Edges addresses | Range of edge weight | Percentage of negative edges | Percentage of positive edges | Data collected (dd/mm/yyyy) |
|---------|--------|------------------|---------------------|-----------------------------|-----------------------------|---------------------------|
| OTC     | 5,881  | 35,592           | -10 to +10          | 11%                         | 89%                         | 08/08/2010-25/01/2016     |
| Alpha   | 3,783  | 24,186           | -10 to +10          | 7%                          | 93%                         | 08/11/2010-22/01/2016     |

**Table 1.** Shows data description.

| Methods                        | Bitcoin-OTC (Older transactions (2010-2012)) | Bitcoin-Alpha | Bitcoin-OTC (Recent transactions (2014-2016)) | Bitcoin-Alpha | Bitcoin-OTC | Bitcoin-Alpha |
|-------------------------------|---------------------------------------------|----------------|-----------------------------------------------|----------------|-------------|-------------|
| Reciprocal [28]               | 0.34                                        | 0.28           |                                               |                |             |             |
| Triadic Status [29]           | 0.65                                        | 0.65           |                                               |                |             |             |
| Fairness-
Goodness(Baseline)[12]    | 0.32                                        | 0.29           |                                               |                |             |             |
| Matrix Decomp. [7]            | 0.27                                        | 0.23           |                                               |                |             |             |
| The proposed model            |                                             |                |                                               |                |             |             |
| $Com - str_{tr>1}$ (pear.)    | 0.22                                        | 0.17           | 0.24                                          | 0.19           | 0.26        | 0.23        |
| $Com - str_{tr>1}$ (cos.)     | 0.21                                        | 0.17           | 0.24                                          | 0.19           | 0.25        | 0.23        |
| $Com - str_{tr>3}$ (jacc.)    | 0.25                                        | 0.20           | 0.25                                          | 0.24           | 0.29        | 0.28        |
| $Com - str_{tr>1}$ (jacc.)    | 0.23                                        | 0.16           | 0.25                                          | 0.20           | 0.28        | 0.3         |

$^1$ https://snap.stanford.edu/data/
Table 2 displays the averaged RMSE of each dataset for the proposed model. Further, the table states the comparative with baseline and other methods. Experimentally, 10% to 90% of links have been removed in steps of 30% to examine the robustness of the model in predicting the missing edges.

In fact, the proposed model has been applied in three experiments on both datasets. Firstly, by taking into account only the older transactions, then by taking into account only the recent transactions. Finally, all the transactions have been taken into consideration. It is obviously, the performance of the model is best when considering time factor in model, particularly the old transactions. Beyond doubt, the model superior the performance of the baseline and the recent methods as illustrated in table 2. Concerning to the different methods in same proposed model, the different representation of features which in turn creates different communities, and in fact this resulted to give different RMSE values. Specifically, the model so called Com – str\_tr\_r>1 scores minimum error. The former, uses Pearson similarity once and Cosine similarity again for the features that have been represented numerically, for the three experiments. Generally speaking, the Com – str\_tr\_r>1 (pear.) and Com – str\_tr\_r>1 (cos.) are the best performance horizontally. Vertically, the older links are more influential than newly created ones and also more influential than when the model was applied without taking time factor into account.

Needs to explain one model and the same applies to the rest of the models. For instance, the model Com – str\_tr\_r>1 (pear.) is stand for community structure formed using Pearson similarity measure to find the similarity among trusted transactions value >1. Figures 2 and 3 demonstrate the performance of the proposed model vs. different algorithms including the baseline and recent method for both Alpha and OTC datasets, respectively. Specifically, the figures show the RMSE scores versus the percentage of removing edges that reflect the horizontally between Com – str\_tr\_r>1 (cos.) and different algorithms.

![Figure 2](image)

Figure 2. This figure shows the performance of different algorithms when predicting different percentage of links in the Bitcoin-Alpha.
Figure 3. This figure shows the performance of different algorithms when predicting different percentage of links in the Bitcoin-OTC.

Figures 2 and 3 show that the $\text{Com-str}_{tr>1} \text{(cos.)}$ (the proposed model) gives the best results when compare with baseline (fairness and goodness) and the recent algorithm (Matrix Decom($k=3$)). In other words, the proposed model provides the minimum error as shown in figures.

8. Conclusion
In this work, community structure is shown to have the effectiveness of predicting comparably with baseline methods and to stay competitive with the recent methods. Moreover, the way the node attribute is represented has an effect on the formation of communities. In other words, the formation of trust communities in Bitcoin networks in terms of how each node assesses the other has a significant role to play in minimizing the error, especially considering only the nodes that take advantage of community structure. Unfortunately, the majority of the small communities is not supporting the proposed model. In sum, the model is somewhat sensitive to the node attributes, and may more sensitive for the time factor. In general, the best completion for the model in terms of the error minimization is when considering only the older transactions, as well as when the features have been represented by the transactions weight values. In other words, when represent the features numerically instead of categorically.

References
[1] A. Andrea, Mastering BitCoin, vol. 50, no. 4. 2014.
[2] M. Monti and S. Rasmussen, “RAIN: A Bio-Inspired Communication and Data Storage Infrastructure,” Artif. Life, vol. 23, no. 4, pp. 552–557, 2017, doi: 10.1162/ARTL_a_00247.
[3] L. P. Nian and D. L. K. Chuen, “Introduction to Bitcoin,” Handb. Digit. Curr. Bitcoin, Innov. Financ. Instruments, Big Data, no. April, pp. 5–30, 2015, doi: 10.1016/B978-0-12-802117-0.00001-1.
[4] W. Li and X. Cai, “Statistical analysis of airport network of China,” Phys. Rev. E - Stat. Physics, Plasmas, Fluids, Relat. Interdiscip. Top., vol. 69, no. 4, p. 6, 2004, doi: 10.1103/PhysRevE.69.046106.
[5] A. Barrat, M. Barthélemy, R. Pastor-Satorras, and A. Vespignani, “The architecture of complex weighted networks,” Proc. Natl. Acad. Sci. U. S. A., vol. 101, no. 11, pp. 3747–3752, 2004, doi: 10.1073/pnas.0400087101.
[6] N. Eagle, A. Pentland, and D. Lazer, “Inferring friendship network structure by using mobile phone data,” Proc. Natl. Acad. Sci. U. S. A., vol. 106, no. 36, pp. 15274–15278, 2009, doi:
10.1073/pnas.0900282106.

[7] S. N. A, “Link Weight Prediction in Signed Networks,” 2017.
[8] O. Moindrot, “Trust in Bitcoin Exchange Networks Datasets and Representation,” pp. 1–8, 2017.
[9] Z. Wu, “The Troll-Trust Model for Ranking in Signed Networks,” 2016.
[10] S. Kumar, “Structure and Dynamics of Signed Citation Networks.”
[11] M. Shahriari and M. Jallili, “Ranking Nodes in Signed Social Networks,” 2014, doi: 10.1007/s13278-014-0172-x.
[12] S. Kumar, F. Spezzano, V. S. Subrahmanian, and C. Faloutsos, “Edge Weight Prediction in Weighted Signed Networks.”
[13] R. M. B and S. D. Bhavani, Link Weight Prediction for Directed WSN Using Features from Network. Springer International Publishing, 2019.
[14] C. Fu et al., “Link Weight Prediction Using Supervised Learning Methods and Its Application to Yelp Layered Network,” IEEE Trans. Knowl. Data Eng., vol. 30, no. 8, pp. 1507–1518, 2018, doi: 10.1109/TKDE.2018.2801854.
[15] P. T. Naderi and F. Taghiyareh, “LookLike: Similarity-based Trust Prediction in Weighted Signed Networks,” 2020 6th Int. Conf. Web Res. ICWR 2020, pp. 294–298, 2020, doi: 10.1109/ICWR49608.2020.9122293.
[16] Z. Cao et al., “Link Weight Prediction Using Weight Perturbation and Latent Factor,” pp. 1–13, 2020, doi: 10.1109/TCYB.2020.2995595.
[17] M. Girvan and M. E. J. Newman, “Community structure in social and biological networks.” Proc. Natl. Acad. Sci. U. S. A., vol. 99, no. 12, pp. 7821–7826, 2002, doi: 10.1073/pnas.122653799.
[18] L. Despalatović, T. Vojković, and D. Vukičević, “Community structure in networks: Girvan-Newman algorithm improvement,” 2014 37th Int. Conv. Inf. Commun. Technol. Electron. Microelectron. MIPRO 2014 - Proc., no. May, pp. 997–1002, 2014, doi: 10.1109/MIPRO.2014.6859714.