2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)

Forecasting Transactional Amount in Bitcoin Network Using Temporal GNN Approach

Shakshi Sharma
Institute of Computer Science
University of Tartu, Estonia
Email: shakshi.sharma@ut.ee

Rajesh Sharma
Institute of Computer Science
University of Tartu, Estonia
Email: rajesh.sharma@ut.ee

***Abstract*—Financial institutions such as banks regularly forecast the amount of finances an individual will have in his/her account in the near future. This can help banks in categorizing their customers so that banks can recommend financial products that matches the needs of their customers. In this work, we explored the historical financial transactions for predicting the amount a customer will receive through his/her transacting partners at a specific time. In particular, we use the Bitcoin transactional dataset, which has two main characteristics: i) network, and ii) temporal. This paper contributes by exploiting a specific kind of Graph Neural Network approach called Temporal-Graph Convolutional Network (T-GCN) for predicting the amount of Bitcoins received by a customer at a particular timestamp. The lower errors obtained using T-GCN approach compared to 11 baseline approaches (such as Support Vector Regression (SVR), Random Forest Regression (RFR), Vector Auto-Regressive (VAR), Long Short-Term Memory (LSTM), etc.) clearly demonstrate the effectiveness of T-GCN approach. In addition, our findings reveal that time is an important feature for such kind of predictive tasks.**

***Index Terms*—Temporal Graph Neural Network, Graph Convolutional Network, Financial Transactional Network, Cryptocurrency, Regression.**

I. Introduction

Financial transactions have come a long way from exchanging goods (commodity) to precious metals (coins) to paper money to online transactions. Most of the customers prefer online platforms for day to day transactions in these days due to various factors like i) ease of transactions: availability of various online mediums such as mobile apps and websites, ii) security: individuals do not have to carry cash with them, and iii) privacy: only the transacting individuals are aware of it.

Various works have been performed on online financial transactions in the past. One line of research focuses on analyzing the financial transactional networks using network theory such as understanding the users' behavior in a static network [1], [2], interpreting the network evolution [3], [4], [5], exploiting the network with the third party data like social media data, call records data [6], [7], [8], etc. Another line of research focuses on prediction of financial transactions using various machine learning approaches such as financial distress of the companies [9], [10], stock market prediction [11], credit card risk prediction [12], identifying the lifetime value of the customers [13], to name a few.

IEEE/ACM ASONAM 2020, December 7-10, 2020
978-1-7281-1056-1/20/$31.00 © 2020 IEEE

This work deals with the prediction of online financial transactions. In particular, our goal is to predict the total amount a user will have in his/her account (from all the transacting users) at a particular time window. The problem of predicting online financial transactions is important as the amount of money not only reflects the financial status of a person in society but also helps financial institutions such as banks in classifying customers in different financial categories. By leveraging this information, banks can recommend financial products or offer designated loans to their customers based on their income.

As the online financial transactions are not publicly available and neither easily accessible due to privacy and business reasons [14], we use the Bitcoin dataset, *BTC Blockchain processed data* [15] containing over one billion transactions with over one hundred thousand unique users (See Section III for further details). In particular, we focus on Bitcoin compared to other publicly available cryptocurrencies datasets as Bitcoin is the dominant cryptocurrency over other cryptocurrencies in the market [16].

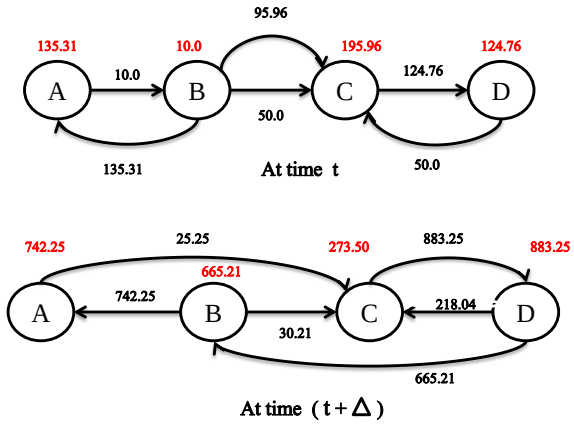


Fig. 1: Bitcoin Temporal Network (best seen in color)

Figure 1 illustrates typical financial transactions from a Bitcoin dataset at time interval t and later at $t+\Delta$, where Δ can represent any specific time value such as a week, a

month or even a year. The nodes in the graph correspond to the individuals (representing bank customers for example) who are involved in the transactions, while an edge corresponds to the transaction that happened between any two individuals. The values on the nodes indicate the amount of Bitcoins received by an individual at a particular time window, and the value on the edges represents the amount being transferred in a particular transaction. With time, the dynamics of the network can also evolve. For example, at time t, a particular bank customer C receives a collective amount of 195.96 from B and D, and at time $t+\Delta$, C receives the total amount of 273.50, collectively from A along with B and D.

Thus, typical financial transactions features (which are also present in Bitcoin dataset) can be categorized into the following:

1) **Network Features:** These includes features that aids in forming a network such as *source user*, *target user*, *amount of Bitcoins transferred*. Furthermore, the network features have the following characteristics:
   a) **Directed edges:** The direction signifies the receiver and initiator of the transaction. For example, consider a transaction in Figure 1 at time t, A transfers an amount of 10 to B.
   b) **Multiple edges:** While executing financial transactions, there is a high possibility that a transacting user could involve in more than one transaction at a specific time window. As it can be seen in Figure 1, at timestamp t, B transacts with C two times. This property of the graph is known as multiple edges in the graph.
2) **Temporal Features:** This feature denotes the time aspect, that is particular timestamp at which a particular transaction has happened. Also, the structure of the network changes with time. For instance, at time t, A is transacting only with B, whereas at time $t+\Delta$, A is transacting with B and C.

Traditionally, classical machine learning models have been explored for financial forecasting. Examples include predicting the price of cryptocurrencies, predicting individuals' behavior in retail banking [17], customer lifetime value [13], etc. Although network-based approaches have been used for understanding user-to-user [18], companies [19], banks [20] financial networks, to the best of our knowledge, these approaches have not been used for predicting the amount of financial transactions. That is, how much a particular user will receive in his or her account.

In this work, we utilize state-of-the-art Temporal-Graph Convolutional Network (T-GCN) for predicting the amount of Bitcoins an individual will receive from his/her transacting partners at a specific time. Since Bitcoins represent the amount (which we are interested in), hence, we use the terms *Bitcoins* and *the amount of transactions* interchangeably. T-GCN model captures both the spatial (topological structure of the network) and temporal features simultaneously from a time-series graph by employing both Graph Convolutional Networks (GCN) and Gated Recurrent Units (GRU). We compared T-GCN with 11 traditional algorithms such as Support Vector Regression (SVR), Random Forest Regression (RFR), Vector Auto-Regressive (VAR), Long Short-Term Memory (LSTM), etc. (See Section V-A). The T-GCN model shows a definite improvement over baseline methods demonstrated using evaluation metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE).

This paper is organized in the following way. Section II covers related work. Section III presents a detailed description and the transformation of the dataset which is used in our experiments. Section IV covers the methodology proposed in this paper. Section V discuss the various models used and experimental results obtained. Finally, Section VI presents the conclusion and future work of the paper.

II. Related Work

Broadly, financial transactions can be divided into two categories: one belonging to traditional financial institutions and the other falls under cryptocurrencies. Thus, we cover works related to both these domain types. We first discuss works related to financial transactions (Section II-A) and next we discuss works related to predictive analysis in the financial domain (Section II-B).

A. Financial transactions

There has been a surge in the interest related to cryptocurrency research [21]. In particular, we present works related to Bitcoin [22], Ethereum [23], and Namecoin [4]. These digital currencies are based on secured blockchain technology [24]. However, the concerns are raised with the use of cryptocurrencies with respect to anti-money laundering approaches [25], taxation and managerial issues [26]. To overcome some problems, mainly due to the lack of governments' involvement of cryptocurrencies, a reliable transaction tracking approach was proposed in [27]. Nonetheless, it has been shown that these alternative currencies can complement traditional currencies [28]. In addition, researchers have also analyzed the transacting patterns on static [1], [2] and dynamic networks [3], [4], [5].

Cryptocurrencies have also been analyzed using network theory for Bitcoin [1], [2], [3] Namecoin [4], [29], Ethereum [5]. Some of the studies analyzed the transacting patterns to understand the behavior of users [1], [2] on a static snapshot of the networks. While others have analyzed the network evolution of these networks [3], [4], [5] and reported that the network densification occurred only in the initial years of the conception of these currencies. In comparison, we performed non-evolutionary study on a financial transactional network. In traditional financial institutions, the network theory has been employed for analyzing user-to-user [18], companies [19], banks [20] networks, to name a few. Besides, few researchers have also analyzed financial data by complementing it with the third-party data such as call data records [6], [7] and social media data [8].

B. Predictive analysis

In this section, we discuss past literature related to various aspects of financial forecasting. One line of works has tried to predict the financial distress of companies [9], [10]. However, the majority of works have dealt with stock market prediction in various forms such as forecast price variation [30], [31], price variation in the stock rates [30], [32], stock market return [11]. These works have performed textual analysis, such as by using news [33] or especially breaking news [34], survey, Twitter, and search engine data [35].

In terms of digital currencies, there are past researches which have studied network-based features [36] and regression models [37] to predict the price of cryptocurrencies such as Bitcoin, Litecoin, Ethereum, Digital Cash, and Ripple by employing Deep Learning models such as Recurrent Neural Networks (RNNs) [38]. In comparison, in traditional financial domains, there are works which developed a framework to forecast the price volatility [39].

There have been few works related to predicting financial behavior of individuals using machine learning techniques [40], particularly in retail banking [17]. Some works have also exploited external data sources such as phone-based call features to predict a person's credit risk [12]. In addition, predicting analytics has been performed on customer lifetime value [13], customer potential value in the insurance industry [41], credit losses [42] of the financial domain.

Our work lies at the intersection of these works. To be specific, we will be performing the regression task of analyzing the Bitcoin transaction network. In this paper, we employed T-GCN approach in order to forecast the transactional amount received by an individual at a specific timeline.

III. Dataset Description

In this section, we first discuss the publicly available Bitcoin transactional dataset (Section III-A). Next, we describe the data transformation steps before applying various predictive models (Section III-B).

A. Bitcoin Dataset

In this paper, we utilized a publicly available Bitcoin dataset, that is, *BTC Blockchain processed data*[1] of size 3 GB [15]. This dataset consists of a processed Bitcoin blockchain transaction network containing over one billion transactions with over 100,000 unique users from the year 2009 to 2017. This dataset is provided in the form of a text file and contains four columns. Specifically, the four columns are, *source id* that represents source of the user that initiates the transaction, *target id* represents target of the user that receives the transaction amount, *timestamp* represents the Unix transaction time (seconds since 1970-01-01) and *amount* represents the amount of bitcoins transferred from source node id to target node id at a particular timestamp. In this paper, we use the term *user* and *id* interchangeably. Table I shows the descriptive statistics of the Bitcoin dataset on per year basis.

[1]https://sites.google.com/site/ninoantulovfantulin/code-data

TABLE I: Descriptive Statistics of Bitcoin dataset

Year\Stats	Total Rows	Total Sources	Total Targets
2009	1	1	1
2010	8330	4	8
2011	333738	59	500
2012	3524408	798	1651
2013	6607083	2085	2885
2014	14794955	9646	76583
2015	24302198	9482	83559
2016	21282910	8321	69213
2017	5450089	1062	1421

B. Transformation of Bitcoin Dataset into Target-Specific Dataset

In the Bitcoin dataset, each row represents a transaction from a *source* to *target* at a specific *timestamp*. However, we are interested in discovering the total *amount* of Bitcoins received by the *target*. To be specific, we have considered only frequent transactions ('frequent' term will be explained in the following steps). In light of this, the following are the transformation steps from the Bitcoin dataset into the Target-Specific dataset.

Step 1: Splitting the dataset: The dataset is first sorted according to the *timestamp* (in this case, the *timestamp* is *the year of transaction*). Then, this sorted data is split temporally into nine different CSV files on a yearly basis. To be specific, each file represents data for one particular year.

Step 2: Finding frequent ids in the dataset: We first identified frequent ids, that is, ones who have transacted at least 100 times in a particular year. We remove those ids from the dataset, which have less than 100 transactions. This limit of 100 is in line with [15].

Step 3: Addition of new features: For each file, we transform the data on per *target id*. That is, for each *target*, we extracted i) total number of sources, ii) total number of transactions, and iii) the total amount at a specific year. In addition, for each file, we added two new features (or columns), that is, i) *number of sources* which represents the total number of sources that have transacted with a particular *target*, and ii) *number of transactions* which represents the total number of transactions in which a particular *target* is involved. Thus, each row contains information about one *target* only and have five features in total, that is, *year of transaction*, *number of sources*, *number of transactions*, and *total amount of bitcoins* received by a particular *target*.

Step 4: Merging the dataset: Next, all the CSV files are then merged into a single Target-Specific dataset file.

Step 5: Handling outliers: After merging the data, we use 1.5 * Inter Quartile Range (IQR) method for filtering out the outliers in the Target-Specific dataset.

See Table II for the descriptive statistics of this Target-Specific dataset (after removing the outliers). Here, 2009 and 2010 year data have been removed as part of the outlier process.

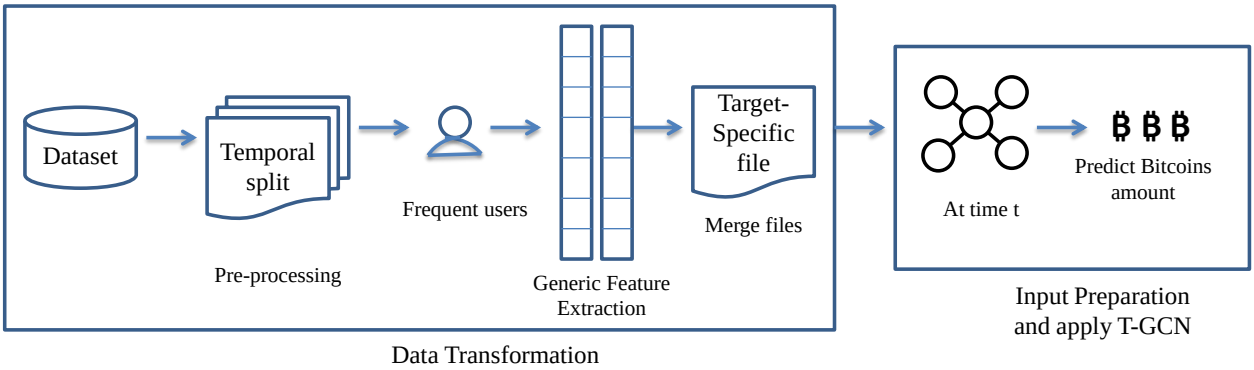


Fig. 2: Methodology

TABLE II: Descriptive Statistics of Target-Specific dataset

Year\Stats	Total Transactions	Total Sources	Total Targets
2011	237	7	300
2012	685	14	1650
2013	1056	27	1885
2014	1537	231	4520
2015	1976	260	12580
2016	1698	198	6439
2017	561	30	1421

IV. Methodology

In this section, we discuss the proposed methodology and its various components that have been employed. The proposed methodology comprises of the following components (Figure 2):

Step 1: Data Pre-processing: We employed numerous steps in the data pre-processing and data transformation process. Like any data analysis step, most of the time consuming task is the data pre-processing and transformation process. The Bitcoin dataset, *BTC Blockchain processed data* is transformed into a Target-Specific dataset in order to find the total amount of Bitcoins received by a frequent user at a specific year (See Section III for a detailed description of the dataset). In particular, in this step, the whole Bitcoin dataset file is split into multiple files based on the years of transactions. Next, we filtered the frequent users from each file separately (Figure 2, *Pre-processing* of Data Transformation step).

Step 2: Extracting generic features: We extracted generic features (Figure 2, *Generic Feature Extraction and Merge Files* of Data Transformation step) such as the *number of sources*, *number of transactions*, *year of transaction*, *target* which are given as inputs to all the algorithms (baselines as well as T-GCN (see Section V-A)). The two new features, (*number of sources* and *number of transactions*) along with remaining features are transformed such that one row corresponds to one *target id*. Lastly, all files are then merged into one file containing the Target-Specific information for the combined years. The less number of features is one of the limitations of our financial transactional dataset.

Step 3: Preparing the inputs for T-GCN: Graph Neural Networks takes the inputs in the form of the adjacency matrix and (or) feature matrix. Thus, this step involves preparing the inputs in order to be fed to the T-GCN model (See Figure 2 *Input Preparation and apply T-GCN* step).

The transactional network that we considered is dynamic in nature, which can be represented using a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \tau)$, where τ represents the set of transacting years, $\mathcal{V}$ denotes the set of transacting individuals and an edge $e_{st} \in \mathcal{E}$ corresponds to the transaction link from a *source* (v_s) to a *target* (v_t), where both v_s, $v_t \in \mathcal{V}$, and $t \in \tau$, denotes the year of transaction. *Source* users, are those users who are sending Bitcoins to others, whereas *target* users are those users who are receiving the amount of Bitcoins from the *source* users. In addition, each *t* corresponds to the *year of transaction*. In these settings, this model is used for the node regression problem. In particular, we regress the node's value, that is, the amount of Bitcoins received by a user.

Specifically, the inputs of the T-GCN model are feature matrix, and adjacency matrix. Feature matrix corresponds to the historical *amount of Bitcoins* (features) received by each node, that is, each row in the matrix corresponds to the features of all the nodes $\mathcal{V}$ at a particular timestamp *t*. The dimensions of a feature matrix are FX$\mathcal{V}$, where F be the total number of features, and $\mathcal{V}$ is the total number of nodes (number of users in our case). The adjacency matrix simply stores the relationships between the nodes. Here, the graph is directed containing multiple edges. The dimensions of an adjacency matrix are $\mathcal{V}$X$\mathcal{V}$. See Table III for graph properties.

TABLE III: Graph Properties

Number of Nodes	28795
Number of Edges	323679
Average Node Degree	19.04
Adjacency Matrix	28795 X 28795
Feature Matrix	37300 X 28795

Step 4: Applying T-GCN model: It is clear from the *Input Preparation and apply T-GCN* step of Figure 2 that the Bitcoin time-dependent transactional data inherently forms a network.

Thus, T-GCN [43] model is a perfect fit for this dataset. In this work, the inputs (the feature matrix and adjacency matrix) are passed to T-GCN model in order to predict the *amount of Bitcoins* received by a frequent user in the next time window. Besides, this model combines both GCN and GRU. Specifically, GCN is used to capture spatial dependencies and GRU is used to address temporal aspects of the network (See Section V-A for technical details of T-GCN).

V. Experimental Results and Analysis

In this section, we first mention the baseline models and discuss the main model we have employed for the regression task (Section V-A). Then, we discuss few of the main hyper-parameter tuning of the models and optimization techniques (Section V-B), and lastly, we discuss the results of our approaches (Section V-C).

A. Models

Baseline Methods: To show the effectiveness of T-GCN model, we used various baseline approaches for regression task such as Support Vector Regression (SVR), Random Forest Regression (RFR), Vector Auto-Regressive (VAR), Long Short-Term Memory (LSTM), Decision Tree Regression (DTR), Polynomial Regression (PR), Ada Boost Regression (ABR), XG Boost Regression (XGBR), Gradient Boost Regression (GBR), Gated Recurrent Unit (GRU), Graph Convolutional Network (GCN). We did not use Linear Regression model as it was violating the assumptions of linear model.

T-GCN: We employed T-GCN [43] model for our problem as the dataset is dynamic. That is the structure of the network in our dataset changes over time. In particular, T-GCN is a specific class of the Graph Neural Networks (GNN) that can handle the time-series graph-structured data. Specifically, T-GCN predicts downstream tasks such as regression, classification by capturing both spatial and temporal properties based on historical data. This model is composed of T-GCN units and each T-GCN unit is a combination of two models, namely, GCN and GRU. GCN is a two-layer neural network that extracts the topological structure of the network using adjacency and (or) feature matrix. Specifically, the following is the propagation rule for every layer in GCN. This rule is inspired from first-order approximation of localized spectral filters on graphs [44].

$$\boldsymbol{H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}}\,\tilde{A}\,\tilde{D}^{-\frac{1}{2}}\,H^{(l)}\,W^{(l)})}, \qquad (1)$$

where, $H^{(l+1)}$ represents the current hidden layer, σ is the non-linear activation function, $\tilde{D}^{-\frac{1}{2}}$ is the normalized degree matrix, $\tilde{A}$ represents adjacency matrix, and $W^{(l)}$ is the learnable weight matrix parameter of previous hidden layer.

GRU is a variant of RNN models that captures the temporal dependency in the data. RNN is a basic choice to handle sequential data. However, it is not able to capture long-term dependencies. Thus, LSTM and GRU comes to the rescue. Here, GRU is chosen over LSTM as it has a shorter training time. Furthermore, GRU has a gating mechanism through which it learns the long term dependencies. To be specific, there are three gates, reset gate, that controls how much information to ignore from the previous timestamp, update gate that controls how much information to bring into the current timestamp and memory gate that stores the memory in the current timestamp. Following equations represents the T-GCN unit. $f(A, X_t)$ represents the graph convolution process (explained in equation 1), h_{t-1} is the output at time t-1, u_t, r_t, c_t, h_t denotes the update gate, reset gate, cell state (memory gate), and output of hidden state at time t respectively.

$$u_t = \sigma(W_u[f(A, X_t), h_{t-1}] + b_u) \qquad (2)$$

$$r_t = \sigma(W_r[f(A, X_t), h_{t-1}] + b_r) \qquad (3)$$

$$c_t = tanh(W_c[f(A, X_t), (r_t * h_{t-1})] + b_c) \qquad (4)$$

$$h_t = u_t * h_{t-1} + (1 - u_t) * c_t \qquad (5)$$

For each unit, in order to capture both the spatial and temporal dependencies simultaneously, the T-GCN model first extracts the topological features from the data using GCN and then extracts the temporal features at the current timestamp along with information of previous timestamps using GRU in order to predict the output at the next timestamp.

B. Hyper-Parameter Tuning and Optimization Techniques

Hyper-parameters play a significant role in machine learning models. In this section, we describe the tuning of some of the hyper-parameters of the models. Specifically, the hyper-parameters of the T-GCN model include batch size, learning rate, training epochs, and the number of hidden units. Table IV describes various hyper-parameters values exploited while evaluating the T-GCN model. The optimal values of the hyper-parameters of T-GCN model are; batch size is 32, learning rate is 0.001, number of training epochs are 100 and the number of hidden units are 64. We used Adam optimizer and a fully connected layer at the end. In addition, we tried different settings of hyper-parameters for all the models and then chose the optimal values. For instance, for SVR, we use radial basis function (rbf) kernel; for RFR, DTR, XGBR, the total number of trees kept are 1000; for VAR, we verified all the assumptions required to perform VAR model such as Cointegration test, Granger's causality test, etc; for LSTM, GRU and GCN, we used 64 as the number of hidden units and one dense layer at the end.

There are two optimization techniques that have been used before feeding the data to the model. First is, *standardization* which is the most important step before training the model. This step rescales the features so that it has a Gaussian distribution with mean zero and standard deviation value of one. It helps the model in better prediction. The second is *feature importance*. In order to select the features that are helpful in predicting the model, few feature importance techniques have been employed, such as F-regression, Mutual Information regression, etc. In all these techniques, the importance scores

of all the features are positive. Thus, we kept all the features and concluded that they all are important in predicting the regression problem.

TABLE IV: Hyper-Parameter Tuning of T-GCN model

Hyper-parameters	Values			
Batch Size	32	64	128	256
Learning Rate	0.0001	0.001	0.01	0.1
Training Epochs	10	50	100	500
Hidden Units	8	16	32	64

C. Results

First, the Target-Specific data is split into train, validation and test set temporally. That is, the training set is from the year 2011 to 2015, the validation set is from the year 2016 and the year 2017 is kept for the test set. Then, the data is fed to the model. As hyper-parameters plays a major role in order to efficiently train the models, we first explore the various hyper-parameters in each of the models. Apart from T-GCN, for all other baseline models, we present only the best results after using various hyper-parameter combinations. See Section V-B for the hyper-parameters discussion. In particular, for T-GCN, we tune the number of hidden units, which is a crucial hyper-parameter. Table V shows the results for various hidden units' values being selected for the T-GCN model. One can notice that with an increase in the number of hidden units, the error decreases. However, the errors start to increase when the number of hidden units has increased to 128. Therefore, we used 64 as the optimal number of hidden units in the T-GCN model as the scores for all the metrics are lowest.

TABLE V: Hidden Units of T-GCN model

Hidden Units\Metrics	MAE	RMSE	MAPE
8	5.05	6.12	1004.21
16	4.55	6.22	904.45
32	1.60	3.62	807.21
64	**1.13**	**3.24**	**669.46**
128	2.63	4.85	750.59

We tune the values of various hyper-parameters of T-GCN model. Specifically, in Figure 3 (a), we increased the number of epochs in order to decrease the error values and continue until the error values start to increase further. As it can be seen that after 100 epochs, the error again start to increase. Thus, we kept the number of epochs as 100. Similarly, in Figure 3 (b), we also tune the number of hidden units. In this case, the optimal number of hidden units are 64 as all the three errors are at its lowest.

Figure 4 displays the training and validation loss obtained while training the T-GCN model. It can be observed that the loss values start to decrease as the number of epochs increases (which is the desired scenario). In addition, it is clear that there is no overfitting in the model, as validation loss is close to the training loss.

Table VI, Columns "w/t" (signifying that time was considered as one of the features) shows errors for various models for three different types of metrics (that is, MAE, RMSE, and

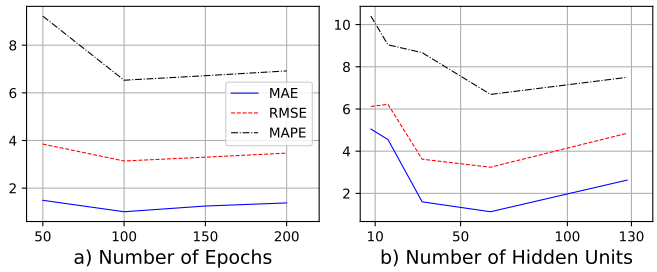


Fig. 3: Tuning hyper-parameters in T-GCN model a) x-axis represents number of epochs and y-axis represents error values; b) x-axis represents number of hidden units and y-axis represents error values

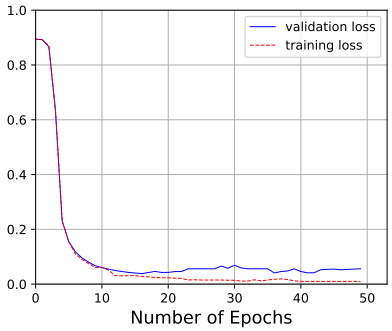


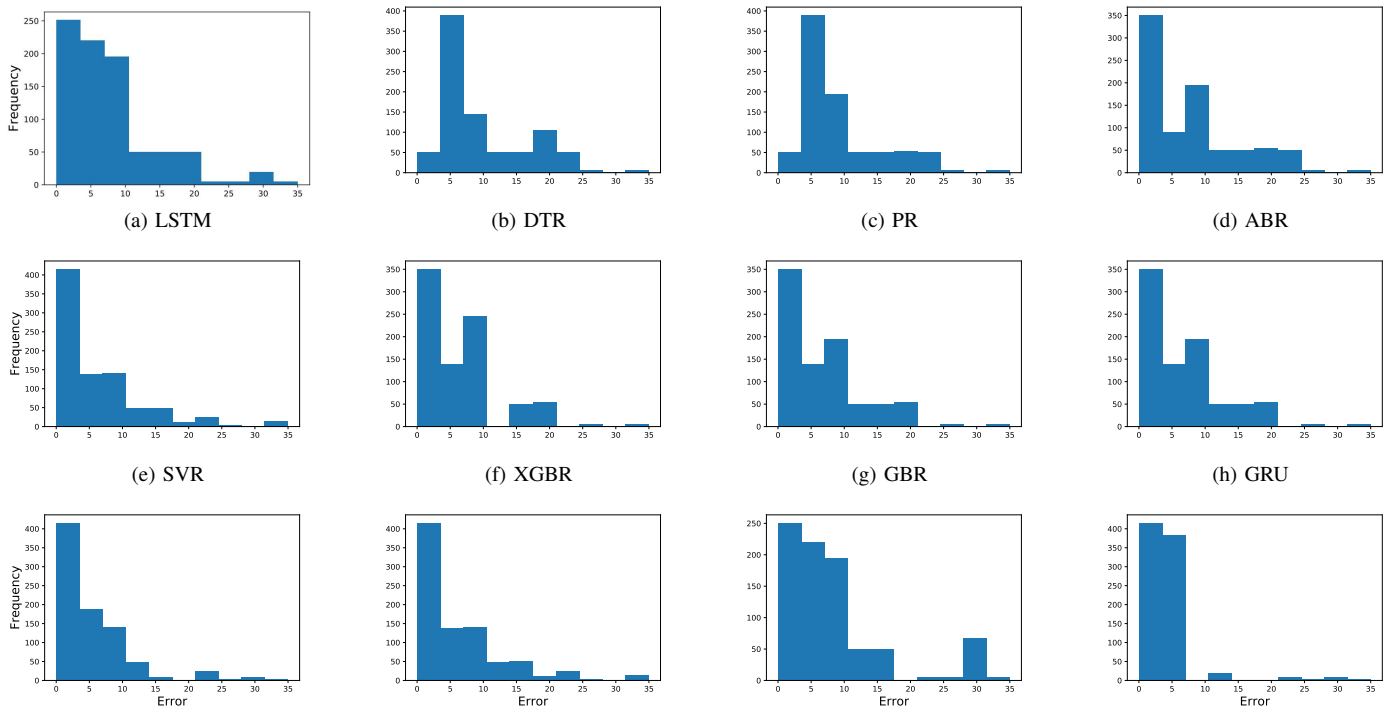
Fig. 4: Training and Validation loss in T-GCN model where x-axis represents number of epochs and y-axis represents loss values

MAPE). It is clear that the T-GCN model outperforms other traditional baseline algorithms. In addition, even in the worst-case, T-GCN values (see Table V) for each metric has a lower error than other baseline models. This is possibly due to the fact that T-GCN is able to exploit both the spatial and temporal properties of the dataset as compared to baseline models.

TABLE VI: Results of Target-Specific dataset on test data with all the features (including time t) and without the time t feature

Models	Metrics					
	MAE		RMSE		MAPE	
	w/ t	w/o t	w/ t	w/o t	w/ t	w/o t
SVR	5.93	6.03	9.28	9.49	1022.49	1047.02
RFR	6.49	6.58	9.22	9.45	1755.82	1758.21
VAR	6.12	6.46	9.49	9.57	1452.21	1464.52
LSTM	6.06	6.45	8.56	8.69	1656.72	1674.12
DTR	8.94	9.04	9.44	9.64	1880.49	1897.62
PR	7.06	7.47	10.04	11.65	1512.27	1524.36
ABR	6.04	6.36	8.01	8.15	1722.19	1745.21
XGBR	5.45	5.55	6.88	7.06	1452.26	1466.82
GBR	6.45	7.02	9.15	9.95	1745.52	1752.63
GRU	6.36	7.25	8.24	9.25	1644.44	1752.22
GCN	4.15	5.15	5.41	5.65	912.01	945.63
T-GCN	**1.13**	**2.15**	**3.24**	**4.26**	**669.46**	**676.21**

In order to verify that time is an important input feature in the Bitcoin dataset, we excluded the time feature in all the



(a) LSTM (b) DTR (c) PR (d) ABR

(e) SVR (f) XGBR (g) GBR (h) GRU

(i) RF (j) GCN (k) VAR (l) GNN

Fig. 5: Error distribution of various predictive models; y-axis represents frequencies and x-axis corresponds to error values of the models

baselines as well as in the T-GCN model. The dataset is now shuffled randomly before feeding it to the model. Here, we used 5-fold cross-validation. The results of the models without considering the time feature is shown in Table VI, columns "w/o t" (representing without time feature) for all the three metrics. From the results, we can observe that the error values in all the models have increased when we do not take the time feature into consideration. However, it is to be noted that T-GCN is still performing better than all the other baseline methods. This concludes that time is an important input feature for the model.

As a microscopic analysis of our models, we plot the error distribution of all the models (See Figure 5). The error is calculated by taking the absolute value of the intermediate result obtained by subtracting the predicted value from the actual value. The X-axis shows the error, and the Y-axis indicates the number of observations for each particular error. The larger the number of observations is nearer to zero, the better the model is in terms of prediction. Among the baseline models, DTR and PR models (Figures 5 (b) and (c)) has the least number of errors in the range zero to five compared to other baseline models. However, it is clear that the T-GCN model has the highest number of observations among the lowest error range (Figure 5 (l)), which reconfirms that T-GCN is the best fit for the prediction problem.

VI. Conclusion and Future Scope

This paper deals with the prediction of the online financial transactional network. Specifically, the aim is to forecast the transaction amount received by a user in a particular year based on its historical transactional data. Based on our extensive experiments, we found out that T-GCN outperforms the other baseline machine learning algorithms as it is able to capture the network and temporal aspects of the dataset. The future work of this paper includes:

1) **Dynamic GNN approaches:** We would like to apply this data to dynamic GNN approaches such as EvolveGCN [45] to lower the errors further.
2) **Attention-based GNN approaches:** We want to extend this work and applying attention-based models in order to focus on relevant features only [43], thereby, improving the model.
3) **New datasets:** We would also like to perform experiments on additional datasets by including other cryptocurrencies datasets such as Ethereum, Litecoin, etc.

Acknowledgment

This research is financially supported by H2020 SoBigData++ project.

References

[1] Dorit Ron and Adi Shamir. Quantitative analysis of the full bitcoin transaction graph. In Ahmad-Reza Sadeghi, editor, *Financial Cryptography and Data Security*, pages 6–24, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[2] Annika Baumann, Benjamin Fabian, and Matthias Lischke. Exploring the bitcoin network. volume 1, pages 369–374, 04 2014.

[3] Ben Holtz, Julie Fortuna, and Jocelyn Neff. Evolutionary structural analysis of the bitcoin network. 2013.

[4] Tao-Hung Chang and Davor Svetinovic. Data analysis of digital currency networks: Namecoin case study. In Hai Wang and Mounir Mokhtari, editors, *21st International Conference on Engineering of Complex Computer Systems, ICECCS 2016, Dubai, United Arab Emirates, November 6-8, 2016*, pages 122–125. IEEE Computer Society, 2016.

[5] Jiaqi Liang, Linjing Li, and Daniel Zeng. Evolutionary dynamics of cryptocurrency transaction networks: An empirical study. 2018.

[6] Yannick Leo, Márton Karsai, Carlos Sarraute, and Eric Fleury. Correlations of consumption patterns in social-economic networks. In *Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '16, pages 493–500, Piscataway, NJ, USA, 2016. IEEE Press.

[7] Ángel Francisco Agudo-Peregrina, Diego Perez, and Martin Langberg. What banking and phone data tell us about the socioeconomic groups and their consumption patterns? In *IEEE/ACM 2018 International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2018, Barcelona, Spain, August 28-31, 2018*, pages 595–597, 2018.

[8] Lorenzo Cazzoli, Rajesh Sharma, Michele Treccani, and Fabrizio Lillo. A large scale study to understand the relation between twitter and financial market. In *2016 third European network intelligence conference (ENIC)*, pages 98–105. IEEE, 2016.

[9] Edward I Altman. Predicting financial distress of companies: revisiting the z-score and zeta® models. In *Handbook of research methods and applications in empirical finance*. Edward Elgar Publishing, 2013.

[10] Jae Kwon Bae. Predicting financial distress of the south korean manufacturing industries. *Expert Systems with Applications*, 39(10):9159–9165, 2012.

[11] Melek Acar Boyacioglu and Derya Avci. An adaptive network-based fuzzy inference system (anfis) for the prediction of stock market return: the case of the istanbul stock exchange. *Expert Systems with Applications*, 37(12):7908–7912, 2010.

[12] Rishav Raj Agarwal, Chia-Ching Lin, Kuan-Ta Chen, and Vivek Kumar Singh. Predicting financial trouble using call data—on social capital, phone logs, and financial trouble. *PloS one*, 13(2):e0191863, 2018.

[13] Edward C Malthouse and Robert C Blattberg. Can we predict customer lifetime value? *Journal of interactive marketing*, 19(1):2–16, 2005.

[14] Israa Alqassem, Iyad Rahwan, and Davor Svetinovic. The anti-social system properties: Bitcoin network data analysis. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2018.

[15] Nino Antulov-Fantulin, Dijana Tolic, Matija Piskorec, Zhang Ce, and Irena Vodenska. Inferring short-term volatility indicators from the bitcoin blockchain. In *International Conference on Complex Networks and their Applications*, pages 508–520. Springer, 2018.

[16] Billy Bambrough. Forbes fact. [-]https://www.forbes.com/sites/billybambrough/2019/08/22/shock-bitcoin-data-reveals-stark-ethereum-litecoin-and-ripple-xrp-warning/41bd2ba42e37, 2019.

[17] Michael A Lazarus, Larry S Peranich, Frederique Vernhes, AU Mattias Blume, Kenneth B Brown, William R Caid, Ted E Dunning, Gerald R Russell, and Kevin L Sitze. Predictive modeling of consumer financial behavior using supervised segmentation and nearest-neighbor matching, January 16 2007. US Patent 7,165,037.

[18] Rajesh Sharma, Artem Mateush, and Jaan Übi. Tale of three states: Analysis of large person-to-person online financial transactions in three baltic countries. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 1497–1505. IEEE, 2019.

[19] Stephanie Rendón de la Torre, Jaan Kalda, Robert Kitt, and Jüri Engelbrecht. Fractal and multifractal analysis of complex networks: Estonian network of payments. *The European Physical Journal B*, 90(12):234, Dec 2017.

[20] Marco Galbiati and Simone Giansante. Emergence of networks in large value payment systems (LVPSs). Department of Economic Policy, Finance and Development (DEPFID) University of Siena 0110, January 2010.

[21] PICHL Lukáš and KAIZOJI Taisei. Volatility analysis of bitcoin price time series. *Quantitative Financ and Econ*, 1:474–485, 2017.

[22] Satoshi Nakamoto et al. Bitcoin: A peer-to-peer electronic cash system. page 1, 2008.

[23] Gavin Wood et al. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151:1–32, 2014.

[24] Michael Crosby, Pradan Pattanayak, Sanjeev Verma, Vignesh Kalyanaraman, et al. Blockchain technology: Beyond bitcoin. *Applied Innovation*, 2(6-10):71, 2016.

[25] Sagwadi Mabunda. Cryptocurrency: The new face of cyber money laundering. In *2018 International Conference on Advances in Big Data, Computing and Data Communication Systems (icABCD)*, pages 1–6. IEEE, 2018.

[26] Rainer Böhme, Nicolas Christin, Benjamin Edelman, and Tyler Moore. Bitcoin: Economics, technology, and governance. *Journal of Economic Perspectives*, 29(2):213–38, 2015.

[27] Sami Ahmed. Cryptocurreny & robots: How to tax and pay tax on them. *SCL Rev.*, 69:697, 2017.

[28] Jon Carrick. Bitcoin as a complement to emerging market currencies. *Emerging Markets Finance and Trade*, 52(10):2321–2334, 2016.

[29] Harry A. Kalodner, Miles Carlsten, Paul Ellenbogen, Joseph Bonneau, and Arvind Narayanan. An empirical study of namecoin and lessons for decentralized namespace design. In *14th Annual Workshop on the Economics of Information Security, WEIS 2015, Delft, The Netherlands, 22-23 June, 2015*, 2015.

[30] Kyoung-jae Kim. Financial time series forecasting using support vector machines. *Neurocomputing*, 55(1-2):307–319, 2003.

[31] Salim Lahmiri. Minute-ahead stock price forecasting based on singular spectrum analysis and support vector regression. *Applied Mathematics and Computation*, 320:444–451, 2018.

[32] Avraam Tsantekidis, Nikolaos Passalis, Anastasios Tefas, Juho Kanniainen, Moncef Gabbouj, and Alexandros Iosifidis. Using deep learning to detect price change indications in financial markets. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pages 2511–2515. IEEE, 2017.

[33] Manuel R Vargas, Beatriz SLP De Lima, and Alexandre G Evsukoff. Deep learning for stock market prediction from financial news articles. In *2017 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, pages 60–65. IEEE, 2017.

[34] Robert P Schumaker and Hsinchun Chen. Textual analysis of stock market prediction using breaking financial news: The azfin text system. *ACM Transactions on Information Systems (TOIS)*, 27(2):1–19, 2009.

[35] Huina Mao, Scott Counts, and Johan Bollen. Predicting financial markets: Comparing survey, news, twitter and search engine data. *arXiv preprint arXiv:1112.1051*, 2011.

[36] Alex Greaves and Benjamin Au. Using the bitcoin transaction graph to predict the price of bitcoin. *No Data*, 2015.

[37] Suni Ajay Kumar. A regression model for crypto-currency price. 2018.

[38] Salim Lahmiri and Stelios Bekiros. Cryptocurrency forecasting with deep learning chaotic neural networks. *Chaos, Solitons & Fractals*, 118:35–40, 2019.

[39] Werner Kristjanpoller and Marcel C Minutolo. A hybrid volatility forecasting framework integrating garch, artificial neural network, technical analysis and principal components analysis. *Expert Systems with Applications*, 109:1–11, 2018.

[40] Chris Baumann, Suzan Burton, and Gregory Elliott. Predicting consumer behavior in retail banking. *Journal of Business and Management*, 13(1):79–96, 2007.

[41] Peter C Verhoef and Bas Donkers. Predicting customer potential value an application in the insurance industry. *Decision support systems*, 32(2):189–199, 2001.

[42] Brett W Cantrell, John M McInnis, and Christopher G Yust. Predicting credit losses: Loan fair values versus historical costs. *The accounting review*, 89(1):147–176, 2014.

[43] Ling Zhao, Yujiao Song, Chao Zhang, Yu Liu, Pu Wang, Tao Lin, Min Deng, and Haifeng Li. T-gcn: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[44] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[45] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao B Schardl, and Charles E Leiserson. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In *AAAI*, pages 5363–5370, 2020.