Cryptocurrency Prices Forecast Using GNNs

LY NGUYEN*, EDWIN ONYANGO*, CHAHAK GOYAL*, and UTTAM RAO*, Dartmouth College, USA

Abstract

Cryptocurrencies have become an extremely popular and volatile market, delivering massive returns (as well as losses) to investors. Cryptocurrency returns prediction remains an open and extremely challenging forecasting task. Using the time series of historical prices as training data, we want to predict if prices will go up or down, and by how much, namely the asset returns given the extreme volatility of the assets, the non-stationary nature of the data, the market and meme manipulation, the correlation between assets and the very fast-changing market conditions. In this paper, we attempt to predict the returns of 14 popular cryptocurrencies, in the time scale of minutes to hours. As changes in prices between different cryptocurrencies are highly interconnected, this article proposes using a spatiotemporal graph neural network strategy (STGNN) to capture both temporal and spatial features to forecast the price of cryptocurrencies. The main results show that the STGNN seems to have relatively better performance than LSTM and LGBM in forward forecasting.

Additional Key Words and Phrases: datasets, neural networks, cryptocurrency, GNNs, STGNN, LGBM, LSTM

ACM Reference Format:

Ly Nguyen, Edwin Onyango, Chahak Goyal, and Uttam Rao. 2023. Cryptocurrency Prices Forecast Using GNNs. *ACM Trans. Graph.* 37, 4, Article 111 (February 2023), 8 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Markets for cryptocurrencies have grown significantly in recent years. Businesses such as Tesla and Microsoft have begun integrating cryptocurrencies into their daily operations as a strategy to save costs, improve efficiency, and attract more consumers interested in utilizing digital currency. This poses a problem for investors who want to choose the best time to purchase, sell, or keep crypto. Thousands of cryptocurrencies have been created with a few major ones that many of you will have heard of including Bitcoin (BTC), Ether (ETH) or Dogecoin (DOGE). Cryptocurrencies are traded extensively across crypto exchanges, with an average volume of $41 billion traded daily over the last year, according to CryptoCompare (as of 25th July 2021).

Changes in prices between different cryptocurrencies are highly interconnected. For example, Bitcoin has historically been a major driver of price changes across cryptocurrencies but other coins also impact the market. The state-of-the-art in crypto forecasting currently depends on conventional time series analysis and machine learning models, but these methods frequently have drawbacks such as a lack of interpretability and subpar performance on extremely non-linear and complicated data.

We suggest using graph neural networks (GNNs) to represent the intricate connections between cryptocurrencies and other pertinent aspects, primarily volatility of the underlying product, in order to solve these issues. GNNs can produce forecasts about future bitcoin values that are more precise and understandable by utilizing the structure of the cryptocurrency network and combining more sources of data. Additionally, GNNs may be utilized to find hidden connections and market insights that standard models can overlook. In addition to contributing to the creation of more sophisticated and understandable models for this purpose, the goal of this study is to investigate the potential of GNNs for crypto forecasting. The findings of this study may have important impacts on traders and investors as well as for companies and organizations planning to use cryptocurrencies in their daily operations.

*All authors contributed equally to this research.

Authors' address: Ly Nguyen, ly.h.nguyen.25@dartmouth.edu; Edwin Onyango, edwin.o.onyango.jr.25@dartmouth.edu; Chahak Goyal, chahak.goyal.24@dartmouth.edu; Uttam Rao, uttam.rao.gr@dartmouth.edu, Dartmouth College, Hanover, New Hampshire, USA, 03755.

0730-0301/2023/2-ART111 $15.00
https://doi.org/XXXXXXX.XXXXXXX

2 DATA

2.1 Dataset Description

We are using a dataset that we obtained from Kaggle published by G-Research, an Europe's leading quantitative finance research firm. It contains information on historic trades for several crypto assets, such as Bitcoin and Ethereum. This data has 7 files including train.csv, example_test.csv, asset_details.csv, supplemental_train.csv, and two files in a folder called gresearch_crypto. Most versions of this data might be self-explanatory except the gresearch_crypto. The gresearch_crypto is an unoptimized version of a time series API file for offline work that needs Python 3.7 and a Linux environment to run. The data has 37 columns: 24 of which are in decimal, one in integer, one id column, and an 'other' data column.

- Timestamp: All timestamps are returned as second Unix timestamps (the number of seconds elapsed since 1970-01-01 00:00:00 UTC). Timestamps in this dataset are multiple of 60, indicating minute-by-minute data.
- Asset_ID: The asset ID corresponding to one of the crytocurrencies (e.g. Asset_ID = 1 for Bitcoin). The mapping from Asset_ID to crypto asset is contained in asset_details.csv.
- Count: Total number of trades in the time interval (last minute). Open: Opening price of the time interval (in USD).
- High: Highest price reached during time interval (in USD). Low: Lowest price reached during time interval (in USD).
- Close: Closing price of the time interval (in USD).
- Volume: The number of cryptoasset units traded during the minute.
- VWAP: The average price of the asset over time, weighted by volume. VWAP is an aggregated form of trade data.
- Target: Residual log-returns for the asset over 15 minutes.

2.2 Exploratory Data Analysis

We first gain an insight into the crypto market by Exploratory Data Analysis (EDA) to understand the structure and patterns of the data. It helps identify any potential issues with the data, such as missing values or outliers, and provides insights into relationships between variables. EDA also helps determine the most relevant features for the prediction model, improving the accuracy of the forecast.

Data Visualization

The trading data format is an aggregated version of market data that includes Open, High, Low, and Close values. This data can be presented using a popular charting method called the candlestick bar chart, which is useful for technical analysis of intraday values. The length of the bar's body indicates the price range between the open and close of the trading day, with a red bar indicating a close lower than the open and a green bar indicating the opposite. These bars are also known as bullish and bearish candlesticks. Additionally, the wicks extending above and below the bars represent the highest and lowest prices of the trading interval.

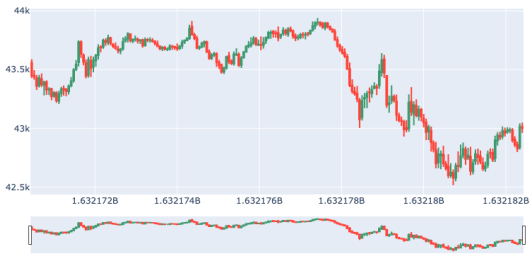

Fig. 1. Candlestick charts of the train data

We visualize the Close prices for the two assets BTC and ETH.

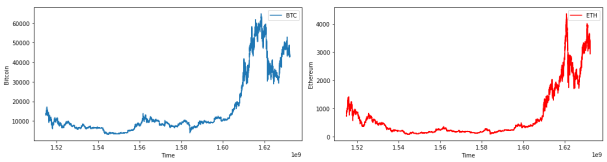

Fig. 2. Close prices for the two assets BTC and ETH

The two assets have different histories, it may still be useful to check if they are correlated in recent times. This is because market conditions can change over time, and correlations between assets can also change. By checking for correlations in recent times, we can get a better idea of whether there is a relationship between the two assets that could be useful for forecasting.

Log returns

While it may be possible to visually observe some potential correlation between two assets on shorter intervals with ups and downs,

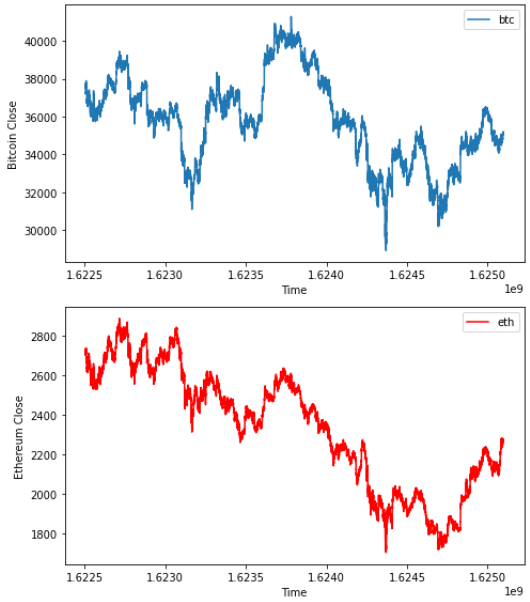

Fig. 3. Close prices for the two assets BTC and ETH

it's generally better to analyze the movements of the assets by calculating their returns. When we plot the log returns, we can see that the resulting signal looks more like white noise, with less drift than the time series for prices. This is because the log returns have removed the underlying trend and level of the original time series, and are only capturing the short-term fluctuations in the asset prices.

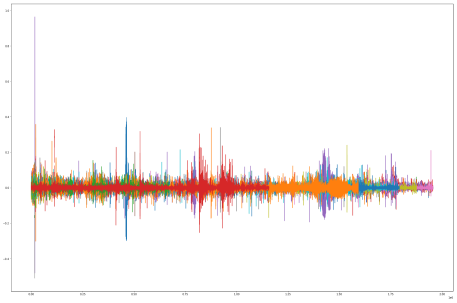

Fig. 4. Log returns for 14 assets

Correlation between assets

Previously, we had an assumption that there could be some correlation between the returns of different crypto assets. To investigate this further, we can specifically examine how the correlation between Bitcoin and Ethereum changes over time during the 2021 period that we selected for our analysis.

It is worth noting that the correlation between the assets is high but also variable over time. This changing dynamic is crucial in the context of time series challenge because it presents a highly non-stationary environment. A system or process that is stationary maintains consistent statistical properties over time, such as

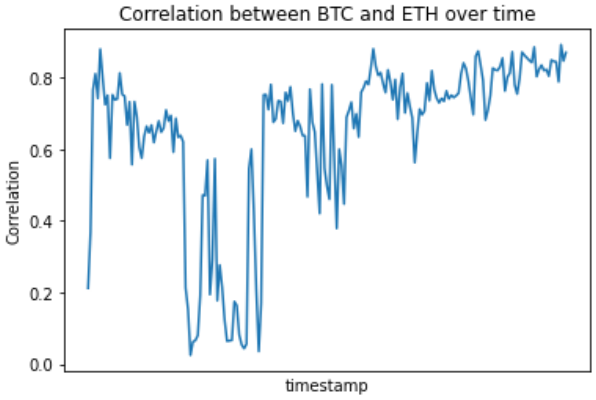

Fig. 5. Correlation between BTC and ETH in 2021

mean, variance, and autocorrelation. In contrast, a non-stationary system exhibits statistical properties that change continuously over time. Stationarity is important because many useful analytical tools, statistical tests, and models rely on it.

To further investigate the correlation between all assets, we can visualize the correlation matrix. This allows us to see that certain assets have significantly higher pairwise correlations than others (Figure 5). With the exception of Ethereum Classic and Dogecoin, most coins exhibit a high pairwise correlation, which confirms our hypothesis and supports our rationale for building a GNN model prediction for these 14 coins.

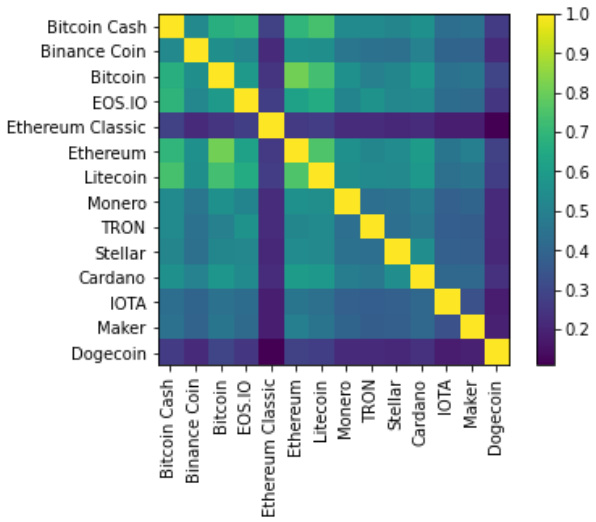

Fig. 6. Correlation between BTC and ETH in 2021

2.3 Data Pre-processing

Check for missing data

There are 340 missed "Target" values. If there is missing data for a given minute, the row representing that minute would be absent from the dataset, rather than containing NaN values. To detect missing data, we can check the timestamp difference between consecutive rows. If the difference between two consecutive timestamps is greater than the expected time interval (e.g., 1 minute), then we know that there is missing data between those two timestamps.

We check the time range for Bitcoin and Ethereum data by using the conversion from timestamp to datetime:

```
BTC data goes from 2018-01-01T00:01:00 to 2021-09-21T00
    :00:00
Ethereum data goes from 2018-01-01T00:01:00 to 2021-09-21T00
    :00:00
```

Time gaps

Time series models require that the data be continuous without any gaps, so it's important to fill in missing data before using these models. One way to do this is by using the '.reindex()' method to forward-fill missing data with the previous valid value. This means that the missing values will be replaced with the value from the previous row, so that the time series remains continuous.

Loader

Our dataset is not given in a convenient form to load a graph, so we wrote a custom loader to format our data to input into PyG modules. It first parses through the dataset (.csv files) and outputs the node IDs, edges, and node features (by timestamp) in JSON format. Then we follow an example from Pytorch Geometric Temporal to load the data as a static graph with a temporal signal.

3 PROPOSED METHOD

3.1 Baseline methods (non-graph based)

Traditional time series methods like ARIMA (Autoregressive Integrated Moving Average) have been widely used in financial data time-series forecasting. However, with the advent of deep learning, methods like LSTM (Long Short-Term Memory) and LGBM (Light Gradient Boosting Machine) have shown promising results in improving the accuracy of financial time-series forecasting.

LSTM is a type of recurrent neural network that can capture long-term dependencies in sequential data and has shown success in modeling complex patterns in financial time series data. LGBM, on the other hand, is a gradient boosting framework that uses exclusive feature bundling and gradient-based one-side sampling to improve efficiency and scalability, making it suitable for handling high-dimensional input features and large datasets.

In this paper on cryptocurrency forecasting, we will utilize LGBM and LSTM as the baseline methods for our analysis as these traditional time series methods have shown significant success in modeling complex patterns in financial time series data. By using LGBM and LSTM as our baseline methods, we can establish a benchmark for evaluating the performance of our proposed forecasting approach.

3.2 Constructing the graph

Our dataset contains timestamped pricing and trading information for a number of cryptocurrencies. Similar to previous works in the

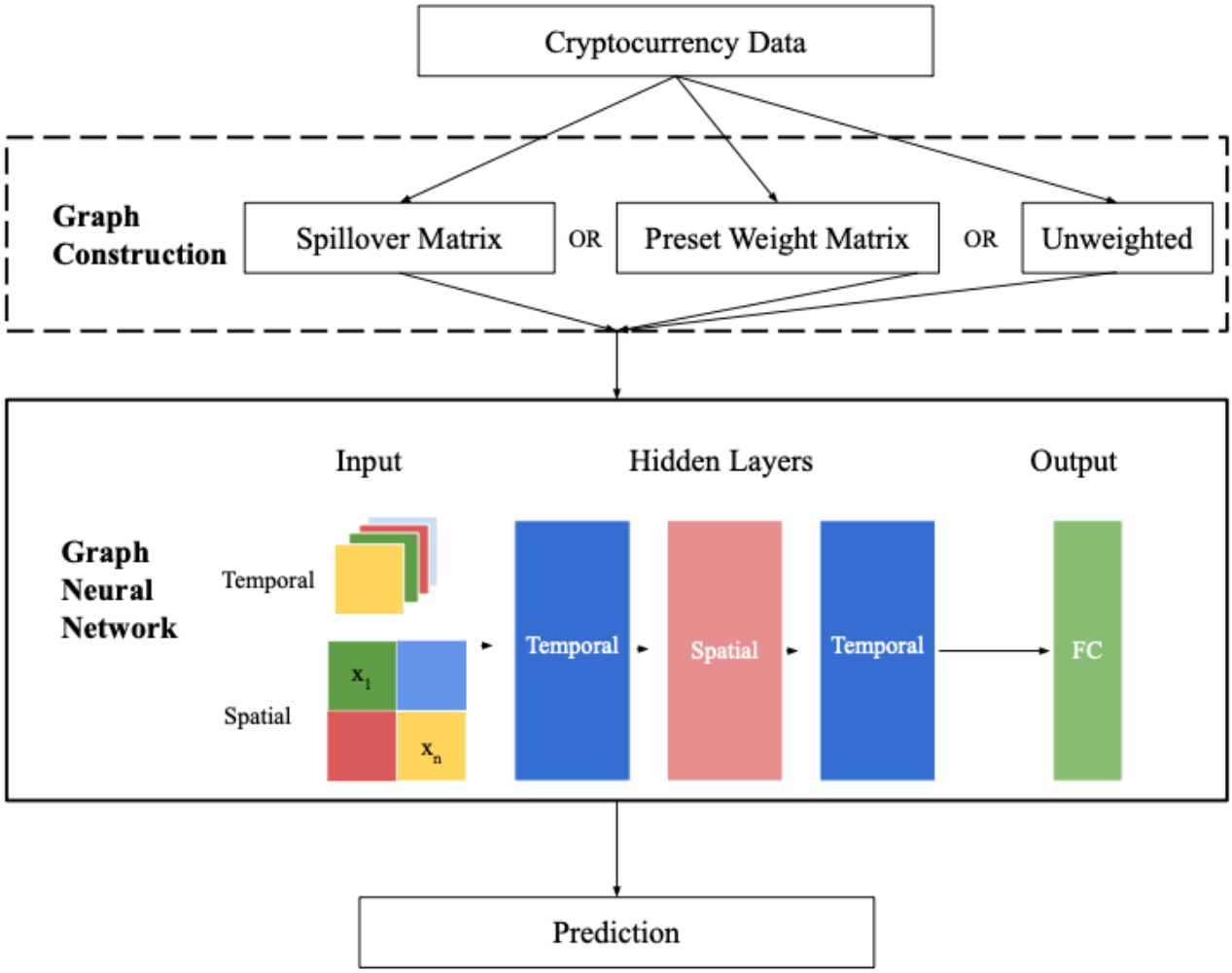

Fig. 7. Framework of the model

finance industry concerning traditional stocks, we can represent our data as a static graph (fixed number of nodes and edges) that evolves over time (node features change over time). In our approach, we view each cryptocurrency as a node and create edges between each of them, which leads to a 14-node graph. We can either leave the edges unweighted (as is done in some previous works) or we can assign them weights. To weight the edges of our graph, we can either calculate the spillover index between cryptocurrencies (using methods similar to Moratis' or Diebold et al.'s mentioned previously) or use set weights based on each cryptocurrencies market cap at a fixed point in time, which are provided by the publishers of our dataset. In our approach, we create both unweighted and weighted (based on market cap) versions of our graph for our various experiments. Figure 8 shows the static structure of our prediction graph.

3.3 Graph-based methods

Since our data is represented as a static graph which evolves over time, we need to use both spatial and temporal components in our model. This lends itself to a class of models called Spatio-Temporal Graph Neural Networks (STGNN), in which standard GNNs are used in combination with various traditional temporal blocks to learn over sequences ("time slices") of graphs. These "time slices" act like a sliding window over which all the timestamps will be processed. Similar to traditional GNN models, a fully connected layer is added at the end to act as a "decoder" to get output with the right dimensions. In our approach, we apply various STGNN methods to make node-level predictions for our graph. Figure 7 shows the basic framework of the models we use in our experiments.

We first implemented a basic STGNN with LSTM as the temporal block and GCN as the spatial block. This configuration is sometimes called a Recurrent Graph Convolution Network (RGCN) in the literature. We also tried a few different configurations of spatial

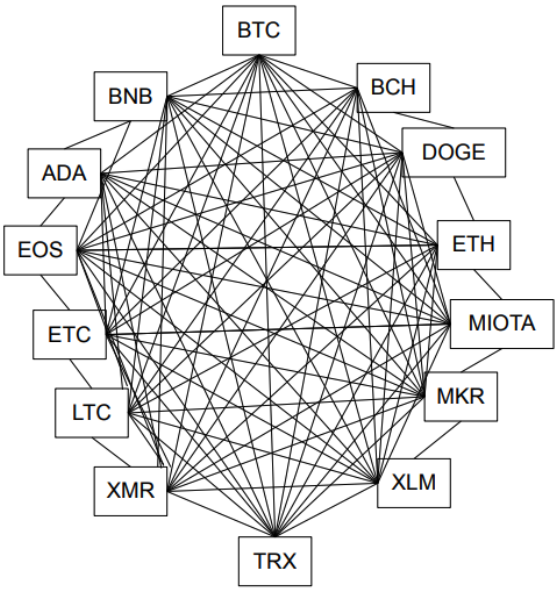

Fig. 8. Static graph structure

and temporal blocks that have been proposed in other papers. Instead of implementing these ourselves, we used an extension library of PyTorch called PyTorch Geometric Temporal. The models we used from this library include DCRNN, GConvLSTM, STConv, and GConvGRU.

3.4 Evaluation plan and metrics

As the evaluation plan relies on the objectives, reemphasizing our objective of obtaining better forecasts of bitcoin values compared to other graph-free methods. The data is already divided into training and test files. This implies that after training on the train_csv data set, we have data to run tests on a weighted version of the Pearson correlation coefficient. Additionally, there is a supplementary training data set given that will help with additional training. As for the evaluation metrics, we evaluated our method using traditional metrics such as MAE and RMSE and comparing our results to the graph-free methods discussed above. The equation for these metrics are as follows:

$$MAE = \frac{\Sigma_{i=1}^{n} |\hat{y}_i - y_i|}{n}$$

$$RMSE = \sqrt{\frac{\Sigma_{i=1}^{n} |\hat{y}_i - y_i|^2}{n}}$$

where n denotes the number of samples, $\hat{y}_i$ is the predicted value of the model and y_i means the true value of the response. We aim to get a minimum MAE and RMSE to ensure that our model works well in forecasting the Crypto prices.

4 EXPERIMENTS

4.1 Light Gradient Boosting Machine (LGBM)

The Light Gradient Boosting Machine (LGBM) is a novel framework for gradient boosting, introduced in 2017 by Ke et al. [2], that aims to solve the issues of efficiency and scalability faced by GBDT and XGB in handling high-dimensional input features and large datasets. Wen et al. [7] suggest that LGBM surpasses other gradient enhancement methods in terms of both training speed and prediction accuracy by utilizing gradient-based one-side sampling (GOSS) and exclusive feature bundling (EFB). The estimation function of LGBM is defined as follows:

$$y_t = \Sigma_{h=1}^{T} f_t(x)$$

where $f_t(x)$ is the regression tree and T denotes the number of regression trees.

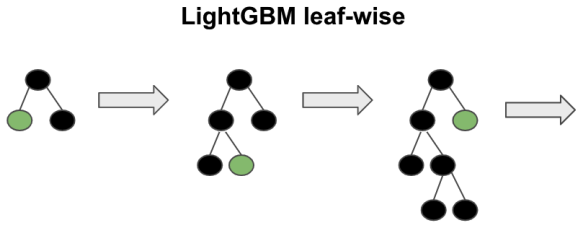

Fig. 9. LGBM Model

Due to LGBM being a decision tree-based model, it has the added benefit of being resistant to multicollinearity. Consequently, including correlated independent variables, which are frequently encountered in economics data, is not a significant concern when utilizing LGBM [2]. This attribute enables greater flexibility in selecting relevant features to incorporate into the model, which ultimately improves its accuracy [7].

Overall, the efficiency and robustness of LGBM make it a valuable tool for machine learning tasks, particularly in the domain of economics and finance where multicollinearity is often present in datasets.

We evaluate LGBM model with MAE and got the results below in Table 1.

Cryptocurrency	15 minutes
Etherium	0.001273
Bitcoin cash	0.001074
Binance Coin	0.001626
Bitcoin	0.001074
Cardano	0.001926

Table 1. LGBM results, MAE

4.2 Long Short-Term Memory (LSTM)

Long Short-Term Memory is a type of recurrent neural network (RNN) architecture that is specifically designed to handle the problem of vanishing gradients in traditional RNNs. The ability of LSTM networks to handle long-term dependencies and capture complex patterns in sequential data has made them one of the most popular and effective types of neural networks for sequence modeling tasks like this one.

For our LSTM neural network, we use a 10-unit LSTM layer followed by a 5-unit LSTM layer before adding the drop-out layer and the dense final layer. The important hyperparameters for this neural network are summarized in the table below:

Hyperparameters	
Units	15
Kernel Initializer	Uniform
Return sequences	True
Optimizer	adam
Batch size	20
Epochs	3
Activation	relu

Table 2. LSTM hyperparameters

Make note that the return sequence is False for last hidden layer as it does not have to return a 3D output to the final Dense layer. As for the kernel initializer, when the Neurons start their computation, some algorithm has to decide the value for each weight and this parameter specifies that. Additionally, the input expected by our LSTM is in 3D format. Our training data has a shape of (a, 15, 1) in the form of (number of samples, time steps, number of features). This means we have 25000 (a) examples to learn in training data, each example looks back 15-steps in time of residualized returns which are given for every minute but represent returns over past 15 minutes. The last number '1' represents the number of features. Here we are using just one column 'Target' hence it is equal to '1'.

As done through this project, we use mean square error (MSE) in the evaluation. With LSTMs, it is more relevant to organize our data such that we are doing predictions on each coin and then evaluate our accuracies for each. Just to get an idea of how this might look like, the table below gives the MSE on various assets for 15 minute prediction, 1 day prediction, and 15 days prediction:

Cryptocurrency	**15 minutes**	**1 day**	**15 days**
Etherium	3.8167e-04	8.5361e-04	6.6209e-04
Bitcoin cash	2.1449e-04	4.9743e-04	3.2002e-04
Binance Coin	5.9046e-04	14e-04	15e-04
Bitcoin	2.8418e-04	4.7447e-04	4.9270e-04
Cardano	4.5230e-04	6.4308e-04	4.3316e-04

Table 3. LSTM results, MAE

We go on to use the trends in these results when making comparisons with our graph models and other experimented models. Based on this, it is important to note that due to LSTMs' ability to incorporate information from past time steps and adjust predictions based on new input, it should be deemed quite competitive among other methods and therefore great for this experimentation.

4.3 STGNN

We first load the data using the custom loader mentioned in section 2.3. We first used max-min normalization on our original input data (node features) to remove dimensional differences. The input data was linearly transformed between 0 and 1. To ultimately get the prediction from the final results, we transform the data back using the reverse of our normalization process. For our node features, we initially tried using all of the features mentioned in the dataset description section, but eventually settled on just using the fifteen-minute residualized returns of each cryptocurrency at each timestamp. This decision was inspired by Yin et al. [8] approach, as they seem to have better results with just their target feature.

Although we tried the different STGNN models listed in the proposed method section, we report results for the RGCN model mentioned above. To train our STGNN model we randomly split our data into training and test sets according to an 80/20 percent split. Our training optimizer was the Adam algorithm and our loss function was binary cross entropy. Table 4 shows the hyperparameters of our STGNN model

Hyperparameters	
Epochs	200
Learning rate	0.01
Batch size	64
STConv blocks	1
Filter size	3
LSTM hidden size	32
Activation	relu
Sliding window size	15
Dropout	0

Table 4. STGNN hyperparameters

4.4 Results

The results of different prediction steps (from +15 minutes to +15 days) are given in Table 5. It can be observed that the STGNN strategy gives a better prediction result to LSTM and LGBM in each of the prediction steps.

	STGNN	**LSTM**	**LGBM**
+15 min RMSE	0.00204	0.0195	0.00436
+15 min MAE	0.00181		0.00233
+1 day RMSE	0.00341	0.0292	
+1 day MAE	0.00288		
+15 days RMSE	0.0205	0.0257	
+15 days MAE	0.0192		

Table 5. Comparison of methods

Table 5 shows the RMSE and MAE for predictions made by our 3 models for +15 minutes, +1 day, and +15 days. Due to both the nature of the model and lack of time, we are yet able to get all the results from the LSTM and the LGBM models. As the table shows, the STGNN model seems to do better than the other models for each prediction step, though it is very close for +15 mins between LGBM and STGNN. In general, the RMSE and MAE get larger for the STGNN as the prediction step gets farther in the future, which suggests that the robustness of the longer-term predictions is not as strong as short-term predictions. This result is similar to that found in Yin et al. [8]. The results are promising, but a full table is needed to make stronger conclusions.

5 RELATED WORK

Several researchers have studied graph-based methods in the traditional finance industry. Wang et al. [6] survey the use of GNN methods in financial applications and review their successes and failures in the recent financial context. They suggest that GNNs are uniquely able to handle the complex graph structures of heterogeneous and time-varying financial data, which result from the inherent volatility and intricacy of the financial market. They categorize the most commonly used financial graphs and outline both the feature preprocessing and GNN methodology for each graph category. This survey is a great place to start for those unfamiliar with the use of GNNs in the financial context.

Matsunaga et al. [3] explore the effectiveness of GNNs with regard to individual stock price predictions in the Japanese Nikkei 225 market over a period of two decades. They build a knowledge graph from the Nikkei Value Search data (a dataset showing supplier relations between foreign and Japanese companies) directly into their predictive model and backtest using rolling window analysis. Their results showed a 2.2-fold increase in the Sharpe ratio (which measures the return on an investment accounting for risk) against the market benchmark and a 1.3-fold increase against a baseline LSTM model. Matsunaga et al. show that GNNs may significantly outperform graph-free machine learning methods for market predictions.

Theurer et al. [5] investigate using relational learning to predict stock prices from the New York Stock Exchange and the Nasdaq Stock Market. Essentially replicating Matsunaga et al., they use a rolling window analysis to ensure that their model performs well across a large range of time slices. Unlike Matsunaga et al. they use Autoregressive Integrated Moving Average (ARIMA) models and Graph Convolutional Networks (GCNs) in conjunction to formulate their model. Similar to Matsunaga et al., they found that their approach outperformed the market standard.

Although several prior works have investigated using GNNs for traditional stock market predictions, surprisingly few have applied similar methods to the cryptocurrency market (though a plethora of work on graph-free methods exists). Yin et al. [8] present a method for using GNNs to forecast the prices of cryptocurrencies with the financial stress index (FSI). Focusing on four specific cryptocurrencies (BTC, LTC, DASH, and ETH), they create a graph with the cryptocurrencies as nodes and the spillover index (using FSI) as edges. They then apply a Spatio-Temporal GNN (STGNN) comprised of LSTMs as their temporal component and GCNs as their spatial component. Their results show that their model outperforms traditional LSTM models. An important takeaway from this paper that may aid us in our project is that the spillover index is an effective way to construct edges in their graph. In this paper, they use a method introduced by Diebold et al. [1] to get the spillover between the cryptocurrency market and the traditional market. In our project, we can use the spillover index between cryptocurrencies themselves. Moratis [4] proposes a method to quantify the spillover effect in the cryptocurrency market at the pairwise directional level. We can apply this method to calculate spillover indices for our data, though further investigation is needed.

6 CONCLUSION

6.1 Summary of what we learned

Some of us have never taken any ML course, so the traditional ML methods were new to us. We learned that STGNN is particularly useful for analyzing spatiotemporal data, such as those generated by social networks and traffic sensors. Traditional ML methods are often designed for analyzing static, spatial data, but spatiotemporal data is dynamic and constantly changing. STGNN is specifically designed to analyze spatiotemporal data and can capture both the spatial and temporal dependencies within the data. STGNNs may also lead to better performance due to their ability to capture explicit relations (edges) that may be missed in traditional models. Overall, STGNN is a powerful machine learning tool that is well-suited for analyzing spatiotemporal data represented as graphs. Its ability to capture both the spatial and temporal dependencies within the data, as well as the relationships between entities in the graph, make it a valuable tool for not only predicting Crypto prices but also for other wide range of applications.

6.2 Challenges

Cryptocurrencies are one of the most popular assets for speculation and investment, but trends are not so easy to predict. Cryptocurrencies operate on a different set of fundamentals than the traditional stock market. Price forecasting is difficult due to price volatility and dynamism and the fact that investor sentiment plays a big role. The data is dynamic and is heavily manipulated by the whales of the market. (A whale is an investor who is powerful enough to make decisions that sway the market in their favored direction.)

Potential challenges we might encounter when working on this project are that because the trends are so volatile and unpredictable, hyperparameters will be difficult to tune which might lead to overfitting, and, since interest in cryptocurrencies has skyrocketed, we will have to work with dynamic data which captures volatility.

Another potential challenge is figuring out how to quantify the relationships between different cryptocurrencies (i.e. the edges of the graph). For traditional stocks, the graph structure is relatively natural/straightforward. Each company/stock can be thought of as a node and the edges are the relations between companies on the exchange, their suppliers, and foreign companies. In this way, company knowledge graphs can be incorporated directly into the model (see related work section). Similarly, for fiat currencies, the strength of a country's government and economy and their foreign

relations can be incorporated into the model. For cryptocurrencies, measuring how they connect to one another can be difficult as the whole point is that they are decentralized. However, there are a few proposed methods to effectively quantify the edges between cryptocurrencies which we may try to incorporate into our project (see the following related works section).

6.3 Future Work

As mentioned in the results section, the first future step should be to finish calculating the MAE and RMSE values for the baseline LSTM and LGBM models to fill out Table 5 so that more confident conclusions can be drawn from our results. Although we did try a few STGNN structures, a future work could try different ones and adjust the hyperparameters. Specifically, using more than one temporal+spatial+temporal block should be investigated. In this paper, we only investigate using one such block. Other future works could be to apply this STGNN strategy on new datasets to predict cryptocurrency prices instead of returns.

7 DIVISION OF WORK

Our team's division of work was a stroke of collaboration that brought together the unique skills and expertise of each team member. There were a lot of moving parts in this ranging from the challenges and motivation to experimentation and future work. While different members contributed equally to most parts of the project, each contributing to the writing of the algorithms, some sections involved different individuals at a deeper level. Edwin worked on most of LSTM approach and analysis, Ly contributed a lot to the LGBM implementation, Chahak took on the challenge of identifying and addressing pressing challenges we faced throughout the project and worked on the experiments, and finally, Uttam worked a lot on the STGNNs. Together, besides eventually delivering a project that was both innovative and effective, we learned a lot during the process.

REFERENCES

[1] Francis X. Diebold and Kamil Yilmaz. 2008. Measuring Financial Asset Return and Volatility Spillovers, with Application to Global Equity Markets. *The Economic Journal* 119, 534 (12 2008), 158–171. https://doi.org/10.1111/j.1468-0297.2008.02208.x

[2] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems* 30 (2017).

[3] Daiki Matsunaga, Toyotaro Suzumura, and Toshihiro Takahashi. 2019. Exploring Graph Neural Networks for Stock Market Predictions with Rolling Window Analysis. (2019). https://doi.org/10.48550/ARXIV.1909.10660

[4] George Moratis. 2021. Quantifying the spillover effect in the cryptocurrency market. *Finance Research Letters* 38 (2021), 101534. https://doi.org/10.1016/j.frl.2020.101534

[5] John Theurer, Yugantar Prakash, Utsav Munendra, and Fabrice Harel-Canada. 2020. Predicting Stock Prices with Relational Learning. (2020). https://fabrice.harel-canada.com/project/gnn-stock-prediction/gnn-stock-prediction.pdf

[6] Jianian Wang, Sheng Zhang, Yanghua Xiao, and Rui Song. 2021. A Review on Graph Neural Network Methods in Financial Applications. (2021). https://doi.org/10.48550/ARXIV.2111.15367

[7] Xiao Wen, Yuanchang Xie, Lingtao Wu, and Liming Jiang. 2021. Quantifying and comparing the effects of key risk factors on various types of roadway segment crashes with LightGBM and SHAP. *Accident Analysis Prevention* 159, 534 (12 2021), 106261. https://doi.org/10.1016/j.aap.2021.106261

[8] Wei Yin, Ziling Chen, Xinxin Luo, and Berna Kirkulak-Uludag. 2022. Forecasting cryptocurrencies' price with the financial stress index: a graph neural network prediction strategy. *Applied Economics Letters* 0, 0 (2022), 1–10. https://doi.org/10.1080/13504851.2022.2141436