Graph Attention Network for Traded Commodity Price Prediction

Yifei LIU
Department of Computer Science
University of Auckland
Auckland, New Zealand
yliu523@aucklanduni.ac.nz

Kwong Chun WU
Department of Computer Science
University of Auckland
Auckland, New Zealand
kwu764@aucklanduni.ac.nz

Kam Leong AO
Department of Computer Science
University of Auckland
Auckland, New Zealand
kao323@aucklanduni.ac.nz

Yong YU
Department of Computer Science
University of Auckland
Auckland, New Zealand
yyu482@aucklanduni.ac.nz

Mark CHEN
Department of Computer Science
University of Auckland
Auckland, New Zealand
zche677@aucklanduni.ac.nz

Abstract* – Market prediction - an ever-important topic in the broad world of machine learning and finance alike. With a stellar reputation and solid track record in recent years, Graph-based Neural Networks (GNN) have been all the buzz in the world of Machine Learning. Combined with a never-before-seen surge of interest in both mainstream and alternative investments since the inception of COVID-19, we have conducted an extensive study fusing two of the hottest topics from the respective domains. In this paper, we propose a novel approach to complement existing financial modelling algorithms with the addition of bleeding-edge Graph Attention Network (GAT) architecture proposed in 2018 by Petar Veličković et al.[1] Our proposed model GAT+LSTM has been designed to accept price data over a broad range of mainstream and alternative commodities - including cryptocurrencies, stocks, indexes, and even gold & oil prices. The results of our experiments indicate that the integration of an additional GAT module significantly increases the accuracy of trend prediction when compared with current mainstream methods.

***Keywords* – Price prediction, cryptocurrency, stocks, index, commodities, graph neural network, graph attention network**

I. INTRODUCTION

Traded Commodity Price Prediction - a decade-old topic that seems to ceaselessly demand attention from financial analysts and data scientists alike, with a well-documented existence even before the term 'machine learning' was coined, traders and brokers have long used mathematical models such as the Fama and French 3 factor model[2] and Black–Scholes model[3] within their respective industries to compete for the leading spot of highest investment returns.

With the meteoric rise of machine learning in recent years, every industry that deals with numbers has been reshaped by some form of algorithmic pervasion regardless of whether the change was welcomed or not. With numerous proven successes to reference - the meteoric rise of Amazon, Netflix, and Uber[4] driven by big data algorithms, it has become very clear that the only options are either to lead the market with optimized algorithms or be left behind in the dust. Our team of researchers were brought together by a shared passion for forward-thinking approaches to solving mundane problems, which sparked the idea of our original study - we were curious to find out if bleeding-edge machine learning algorithms are able to assist in the decade-old problem of stock price prediction.

Due to the recent nature of GNN's debut into the ML limelight, the unexplored landscape of this domain is still unfolding rapidly in real-time, as such, the availability of relevant articles that are available for direct reference were limited. Due to the inaccessibility of useful material in the context of GNN and financial price prediction, our team resorted to referencing content from the course *CS224W:Machine Learning with Graphs* hosted online by Stanford, and complemented those fundamental ideas by participating in online ML journal reading clubs such as the *Graph Representation Learning* journal club. After a basic understanding of GNN was achieved, our original objective was naively defined as '*To use bleeding-edge graph neural networks to predict the price movement of Cryptocurrencies*'.

As the group explored deeper into GNN literature, some flaws in our understanding started surfacing very quickly. The following three main points were identified during the early stages:

Firstly, the term Graph Neural Network is an umbrella term that covers a broad range of algorithms that can range from Graph Convolutional Networks (GCN) to various spatial and spectral methods. Therefore we had to clearly re-define and justify our GNN of choice. The team unanimously agreed to have a main focus on Graph Attention Networks(GAT) as it was often the best-performing flavour of GNN within the context of commodity price prediction.

Secondly, our team identified a major shortcoming of our chosen algorithm - GAT by itself was not capable of handling time-series data elegantly, therefore our entire workflow was re-evaluated and a decision was made to implement additional Recurrent Neural Network(RNN) modules to enable GAT to work seamlessly with time-series price data.

Finally, after the basic architecture had been successfully actualized as a working proof-of-concept, our preliminary results indicated that restricting our scope to only cryptocurrency was not a feasible option, as Figure 1 has shown, almost all cryptocurrencies are highly positively correlated with each other, even more so than traditional stocks of the same sector (Tech, medicine, etc..). Therefore we decided to introduce more diversity to our dataset by introducing various stocks, traded indices, and even gold & oil

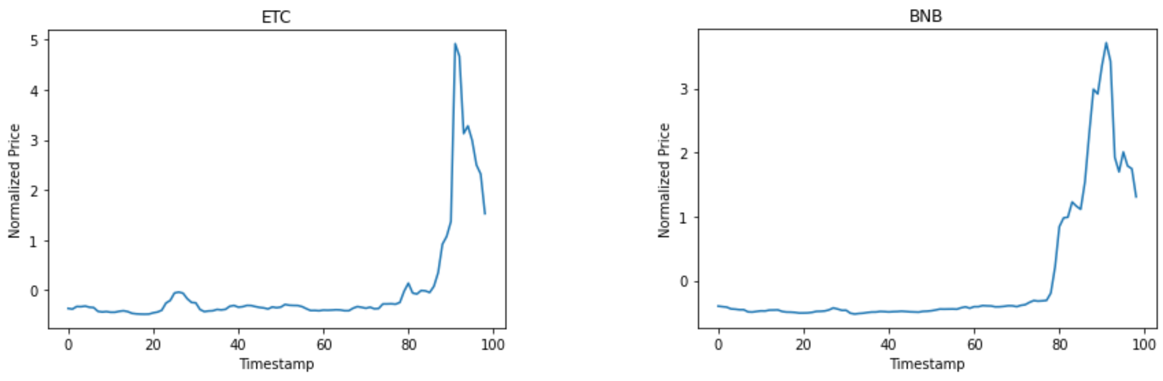

Fig 1: Cryptocurrencies are highly positively correlated with each other.

prices - modifying our architecture to take as input a variety of traded commodity historical prices.

With the three roadblocks above identified and resolved, we clearly re-define our final research objective as '***Optimizing GAT to thrive when working in a time-series context of commonly-traded commodity price prediction.***'

The above points summarize our research group's journey to reach the final proposed architecture of **GAT+LSTM** model, which is the main contribution of our paper - a novel architecture that utilizes one of the latest GNN algorithms to complement existing mainstream time series forecasting algorithms, offering a significant increase in accuracy when predicting the direction of traded commodity price movement.

II. Related works

A. Graph Attention Network (GAT)

Graph Attention Network enables certain self-attention based structured data to transform to a graph-like data to represent the entire data structure[5]. GAT model is based on the algorithm mechanism of the graph neural networks and the attention mechanisms, also the model enables the graph to implicitly allocate the weights to different edges that the entire process so called correlation coefficient. In other words, number of nodes and edges connecting to formulate a graph, if the weight is higher between nodes (i.e., higher value of the edge), this indicates a greater relevance between the two nodes. In the proposed project, nodes represent the commodities; edges represent the relationships between two commodities (further introduced in 'methodology' section), the goal of implementing the GAT model is to extract the relationships between commodities, we assume that if two nodes have the higher weight in the graph structure, the prices changes are more likely to have the similar patterns. Therefore, the GAT model is deployed different real-life applications, such as recommender system, pedestrian walking trajectory prediction [5], and speaker verification [6] these continuous movement tasks. In terms of stock patterns prediction, One study conducts a model called Financial Graph Neural Network (FinGAT) to predict specific stock prices patterns, as well as to rank the stocks which are profitable; the model applies numbers of techniques such as Attentive Graph Recurrent Units to generate short-term and long term features and GAT to generate a fully connected graph.

B. Long Short-Term Memory (LTSM)

LSTM has become one of the most frequently used methods to solve time series-related issues [7]. The LSTM model is to mitigate one of the drawbacks of GAT as mentioned earlier in the introduction – time insensitivity. A literature review of LSTM [8] demonstrates that the model and its variant models are optimal for solving time series-related prediction, text recognition, natural language processing, and generating video subtitles, alongside a variety of other continuous information tasks.

The chain form of model mechanism is able to learn new input data from the previous memory. In other words, the forget gate layer is capable to decide the input information whether reserving it or abandoning it, which means that the information is to be kept in memory if it comes from the previous timestamp data. Otherwise, it is going to be forgotten. Seemingly, the model does not require to learn a series of data from stretch every time. Therefore, roughly dissecting the LSTM model, the entire model comes with number of blocks connecting together, for each block, it comprises forget gates, input gates, and output gates that there are splitting three parts when the data input and output into the model, the forget gate as introduced earlier; followed by the input gate is to learn and update information when the data is inserted into the second part; the final part is the output gate, which serves the main purpose of passing the updated information to the next block. Greff et al. [9] discovered that in terms of LSTM architecture, the forget gate and the output activation function are the two most important parts to develop LSTM blocks after they initiated a series of experiments. The research provides an insight to the proposed project that these two components are required to adjust appropriate values in order to obtain decent results. A study [10] initiate a project called graph-based stock price prediction, in which the graph

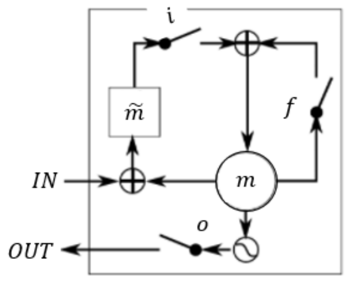

Fig 2: The structure of LSTM. i, f and o are the input, forget and output gates, respectively. m and $\widetilde{m}$ denote the memory cell and the new memory cell content.

is produced by graph convolutional network, which is then fed into an LSTM module to predict the stocks investment return ratio.

C. *Gated Recurrent Units*

GRU was introduced by Cho et al. [11] in 2014. It has a similar design as LSTM that every recurrent unit inside could obtain dependent relationships of various time scales. Both LSTM and GRU are good at handling the vanishing gradient problem. In the Bitcoin price prediction study conducted by A. Dutta et al. [12] in 2019, Gated Recurrent Units (GRU) outperformed other existing models including LSTM. Moreover, M. Rahman et el. [13] obtained an impressive result in the stock price prediction study.

Different from the LSTM, there are only two special units inside GRU, namely update gate and reset gate. The update gate is a combination of the input and forget gate of the LSTM while it also combines the hidden state and cell state of LSTM. Both gate units regulate the information flow and makes decisions on whether an information could be passed to output. An important feature of these two gates is their trainability on keeping information. The GRU could retain information obtained from extended periods of time, and also eliminate information which is unrelated after training.

Compared with LSTM, the advantage of GRU is that there is less redundant information retained inside the model. GRU also uses less parameters, hence training time would be shorter. With the insights from A. Dutta et al. [12] and M. Rahman et al. [13], we were also interested to apply GRU as an alternative to LSTM, to observe effects on performance.

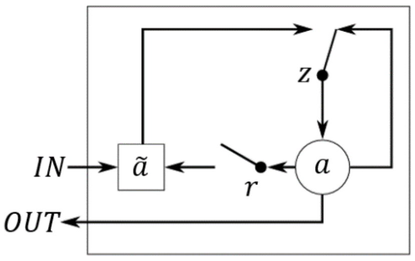

Fig 3: The structure of GRU. z and r are the update gate and forget gate respectively, while a and $\tilde{a}$ are the activation and the candidate activation.

III. METHODOLOGY

The GAT+LSTM model architecture is shown in Figure 4. The model consists of three main steps: 1. Feature Extraction, 2. Relation Graph Modeling 3. Long Short-term Memory. Firstly, we extract the daily commodities trading price in week i to the weekly average price A_i and calculate the return ratio R_i. Secondly, the A_i and R_i will use as a feature set v_i to create the graph structure data. Then use GAT to learn the relationships between the target stock S_t with other stocks, in order to get a graph-based representation $g_i^{S_t}$ for the stock S_t in week i. Lastly, we use the graph set U_i^G , which is the graph-based representation graphs, in the last d weeks of week i as input in LSTM to predict the close price in week i.

A. *Data set*

Unlike most of the common price prediction research which have a main focus on stocks, we are more interested in the price movement of cryptocurrencies as it has become a rising star in the investment field. We selected a wide variety of asset classes around the cryptocurrencies in our study, including cryptocurrencies related stocks, commonly traded indices, currency price and commodities. Our assets used are listed in the Table 1.

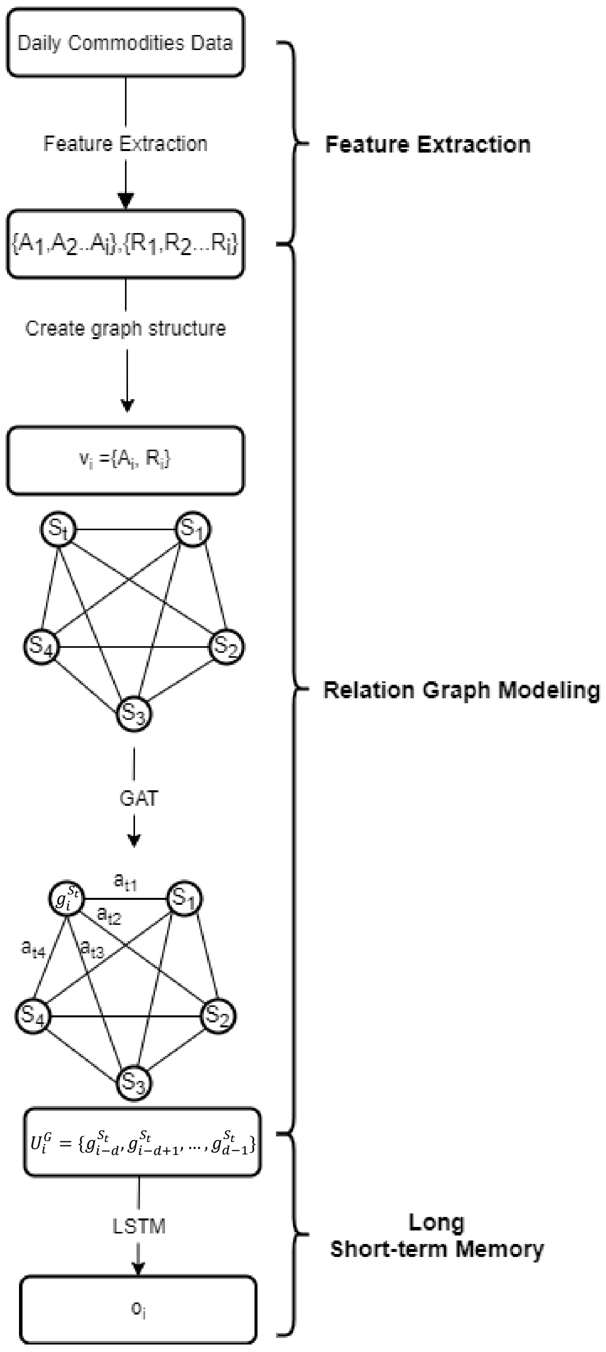

Fig 4: GAT+LSTM model

The raw data we used in our study was all time series data in daily basis, starting from 2019-Jul-01 to 2021-Jun-30. We collected dates and four features of the price, they were Open, High, Low and Close. All the non-cryptocurrencies and cryptocurrencies data by scrapping from https://finance.yahoo.com and https://www.binance.com respectively.

Cryptocurrencies are traded 24x7 non-stop in the market while stocks and other assets have regular trading time, there is missing data arisen from the weekends and market holidays if we use daily data to build our model. Therefore, we took an average weekly price to build graph structures for in GAT model. Apart from that, we conducted a standard normalization on our average weekly price in order to diminish the effects from numerical value difference between the prices of the assets. After the preprocessing steps, we had 104 points of data per feature of an asset.

Table 1: List of the 37 assets used in the paper

Ticker	Company/ Underlying Asset	Asset Class
AAPL	Apple Inc.	Stock – IT
AMZN	Amazon.com Inc	
FB	Facebook, Inc.	
GOOGL	Alphabet Inc	
BAC	Bank of America Corporation	Stocks – Financial
JPM	JP Morgan Chase & Co	
WFC	Wells Fargo & Company	
ARKK	ARK Innovation ETF	Stock – Firms buy Cryptocurrencies
MSTR	MicroStrategy Incorporated	
SQ	Square Inc	
TSLA	Tesla Inc	
VYGR	Voyager Therapeutics Inc	
MA	Mastercard Inc	Stocks – Payment
PYPL	Paypal Holdings Inc	
V	Visa Inc	
F	Ford Motor Company	Stocks – Others
GE	General Electric Company	
HD	Home Depot Inc	
T	AT&T Inc	
UNH	UnitedHealth Group Incorporated (DE)	
WMT	Walmart	
XOM	Exxon Mobil Corporation	
ADA	Cardano	Cryptocurrencies
BNB	Binance Coin	
BTC	Bitcoin	
DOGE	Dogecoin	
EOS	EOS.IO	
ETC	Ethereum Classic	
ETH	Ethereum	
LTC	Litecoin	
XRP	Ripple	
^DJI	Dow Jones Index	Commonly Traded Indices
^GSPC	S&P 500 index	
^TNX	10-year Treasury Yield Index	Indices Reflecting USD Strength
DX-Y.NYB	US Dollar Index	
BZ=F	Brent Crude Oil Price	Commonly Traded Commodities
GC=F	Gold Futures	

B. *Feature Extraction*

The first step is to set the length of days for averaging. Stock trading is based on the day, but in general, the price fluctuation range of each day is minimal, and it fluctuates around a value in a short time. Considering the computational efficiency of the model and the real-world stock trading situation, our model uses the week as the length of days for averaging. Therefore, all data are the average of all trading days of the current week. For example, if there are five trading days in this week, the data for this week is the average of the five trading days. If there are only three trading days in a week, the average of these three days will be used as the data for the week. The formula is expressed as follows:

$$A_i = \frac{\sum_{j=1}^{n} P_{i,j}}{n}$$

where A_i is the average close price of the week i. $P_{i,j}$ is the close price of day j in the week i, and n is the trading days in the week i.

After obtaining the weekly average data, we calculated the return ratio between the week and the week. The formula is as follows:

$$R_i = \frac{A_i - A_{i-1}}{A_i}$$

where R_i is the return ratio value in week i, and A_i is the weekly average price value, A_{i-1} is the average price of the previous week i.

Since the prices of different stocks are significantly different, we have performed standard normalization for the close price data. As the result of this step, we get the normalized weekly average close price and weekly return ratio as the feature.

C. *Relation Graph Modeling*

The price changes of each commodity may be related to the price changes of other commodities. However, we do not know the specific relationship between them. In this step process, weekly data will generate a graph structure data and model the relationship between different commodities through GAT. After this process, the weekly time series data will become the time series data of the graph with relationships. This step includes two processes.

1. Create graph structure data

After obtaining the weekly average data of each stock, use the weekly average data as a feature to establish a graph model between each stock. First, each commodity is a node, and each node contains its data as a feature. In our final model, the features that we used are close price and return ratio. The graph is fully connected, which means every node connects to each other, including itself. Moreover, the connects are bidirectional, so if a graph with M vertices will have M^2 bidirectional edges.

2. GAT process

In the previous process, we established graph structure data. However, the relationship between each node is unknown at this stage. This process uses GAT to model the weight of each edge and extract the relationship between commodities. The graph attention network learns attention weights between nodes and extract the weight information of edges from the neighboring nodes. The formula can be represented as follows:

$$GAT(S_t) = ReLU(\sum_{S_n \in U_t} \alpha_{tn} W v_i^{S_n})$$

where S_t is the target stock, which we want to find the relationship with others. $ReLU$ is a non-linear activation function, and α_{tn} is the attention weight from neighbour stock S_n to target stock S_t, U_t is a set of all the neighbour nodes of S_t, and W is a learnable weight matrix, the $v_i^{S_n}$ is the feature vector of stock S_n in the week i. The attention weight α_{tn} can be expressed as follows:

$$\alpha_{tn} = \frac{\exp\left(LeakyReLU\left(a^T\left[\boldsymbol{W}v_i^{S_q} \parallel \boldsymbol{W}v_i^{S_n}\right]\right)\right)}{\sum_{S_n \in U_t}\exp\left(LeakyReLU\left(a^T\left[\boldsymbol{W}v_i^{S_q} \parallel \boldsymbol{W}v_i^{S_n}\right]\right)\right)}$$

where a is a single-layer feedforward neural network, parameterized by a weight vector, and T means transposition and || is the concatenation operation.

Therefore, after this step, we can get the graph-based representation $g_i^{S_t} = GAT(S_t)$, which encoded the relationship between S_t and other stocks in week i.

3. Long short-term memory

The stock data is time series, and our goal is to predict the stock price in week $N+1$ through N weeks of data. Here, we input the encoded relationship graph data $g_i^{S_t}$ into LSTM to get the final predicted price. Assume that past d weeks are used to learn in this step, then we need to get a sequence from the last step Relation Graph Modeling:

$$U_i^G(S_t) = \{g_{i-d}^{S_t}, g_{i-d+1}^{S_t}, \ldots, g_{d-1}^{S_t}\}$$

which is a set of relation graphs from the week $i-d$ to the week $i-1$. Then Long Short-term Memory (LSTM) is adopted, and its last output price $o_i^{S_t}$ in the week is given by:

$$o_i^{S_t} = LSTM(g_i^{S_t}, h_{i-1}^{S_t}, c_{i-1}^{S_t})$$

where $h_{i-1}^{S_t}$ is the hidden-state vector and $c_{i-1}^{S_t}$ is cell-state for the target stock in week $i-1$. The LSTM is based on the attention mechanism to learn the dynamic weights depending on the importance of each week.

4. Loss Function

The loss function is Mean Squared Error is given by:

$$Loss = \frac{1}{n}\sum_{i=1}^{n}(o_i^{S_t} - y_i^{S_t})^2$$

where $y_i^{S_t}$ is the ground-truth price of stock S_t in week i.

The code repository for our methodology design can be found here:https://github.com/ProfiterolePuff/DL_Stock_Prediction

IV. Evaluation

A. *Train/Test Split for Time Series Data*

Since we are dealing with time series data, the order of time is essential which means we cannot randomly sample the data during evaluation stage. We have 104 records at weekly intervals in total, we retain the first 84 weeks (80%) as training data and uses the last 20 weeks for testing.

We also have a baseline model which is pure LSTM which is used as comparison for our final model performance. We train both the proposed hybrid model and the baseline model on the first 84 weeks and use the model to make the predictions for the last 20 weeks. We collect the statistics for evaluations only based on those 20 predictions for the last 20 weeks.

B. *Statistics for Evaluation*

In order to evaluate the performance of the model, one statistic we must include would be Mean Squared Error. However, in terms of real-world applications of our model, MSE has a major drawback since MSE can only measure the average of the squared errors. A simple example here can illustrate this drawback. In the Figure 5, black points indicate the actual price of the asset. Point P_a and P_b indicates the prediction of the model A and model B respectively. The squared errors d^2 for those two model predictions are exactly the same while if we trust Model A's prediction and buy that asset at t_0 the result would be a loss and if we trust model B's prediction and short sell on that asset at t_0, the result would be a gain.

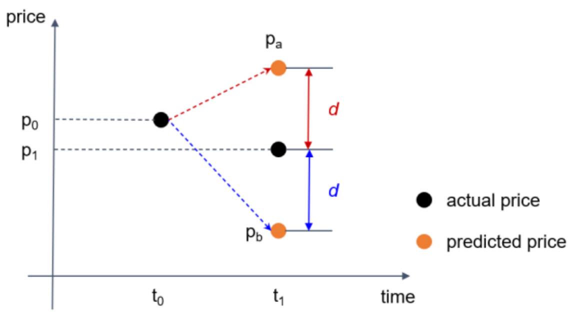

Fig. 5: Drawback of MSE

Thus, we introduce another statistic, accuracy of prediction trend, to measure the model performance. In the Figure 5, P_a would be classified as wrong prediction trend while P_b would be classified as correct prediction trend as from t_0 to t_1 the actual price of the asset is going down. This new statistic overcome the drawback of the MSE. In the following result section, we would like not only to minimize the MSE but also maximize the accuracy of trend prediction.

V. Experiments and Results

For all statistic figures (Figure 6 – 11), the blue, orange, green, red bars are the model statistic of performance evaluation for overall, cryptocurrency, stock asset and the other asset respectively. Since we have five parameters exclude learning rate and number of epochs and the training time is almost one hour for each complete run, we decided it was time-infeasible to perform grid search on every parameter. Thus, we make the assumption that all five parameters are independent which means in the following sections we change one parameter at a time while retaining all the others.

A. *General ML Hyperparameters*

Our model involves two general ML hyperparameters which are learning rate and number of epochs. In this work, we use learning rate of 0.001 for both proposed and baseline models and 500 epochs for proposed model and 200 epochs for baseline model. However, for our proposed GAT+LSTM, lowering the learning rate and increase the number of epochs could leads to better performance not only in terms of MSE but also accuracy of trend prediction. However, it will also increase the running time dramatically. With the existing setting of 0.001 learning rate and 500 epochs for our proposed GAT+LSTM model, it takes 45 minutes with a Nvidia

GeForce RTX 3080 desktop for one run on CUDA. Due to time restraints, we are not able to reduce the learning rate further or increase the number of epochs any further.

B. *General parameters for Time Series Forecasting Model*

There are two general parameters for time series forecasting model. They are $\boldsymbol{N}$ which represents the number of previous observations (weeks) used to predict the next observation and $\boldsymbol{F}$ which represents the features inputted into the model. Even through LSTM is designed for sequence data, it's still not suitable to use a very larger $\boldsymbol{N}$ number which results in very long inputted sequence and causes a reduced performance of the model. We reported the model performance for those two parameters in Figure 6 and Figure 3. It is noticeable that Figure 7 shows the inputted features have a huge impact on the accuracy of trend prediction of the model.

C. *GAT Hyperparameter*

There are two hyperparameters for GAT component of the hybrid model. $\boldsymbol{O}$ represents the output feature size before last GAT layer and $\boldsymbol{H}$ is the attention mechanism number since we are using multi-head attention mechanism. As the Figure 8 and Figure 9 have shown, both $\boldsymbol{O}$ and $\boldsymbol{H}$ has quite a large effect on both MSE and accuracy of trend prediction of the model.

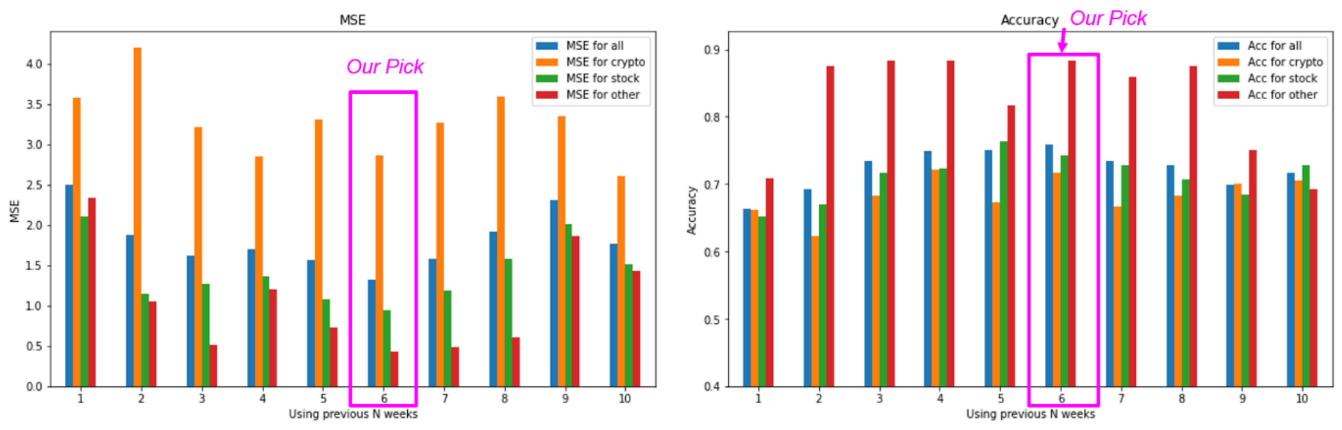

Fig. 6. Effect of General Parameter $\boldsymbol{N}$. $\boldsymbol{N}$ = 6 is picked in final proposed model.

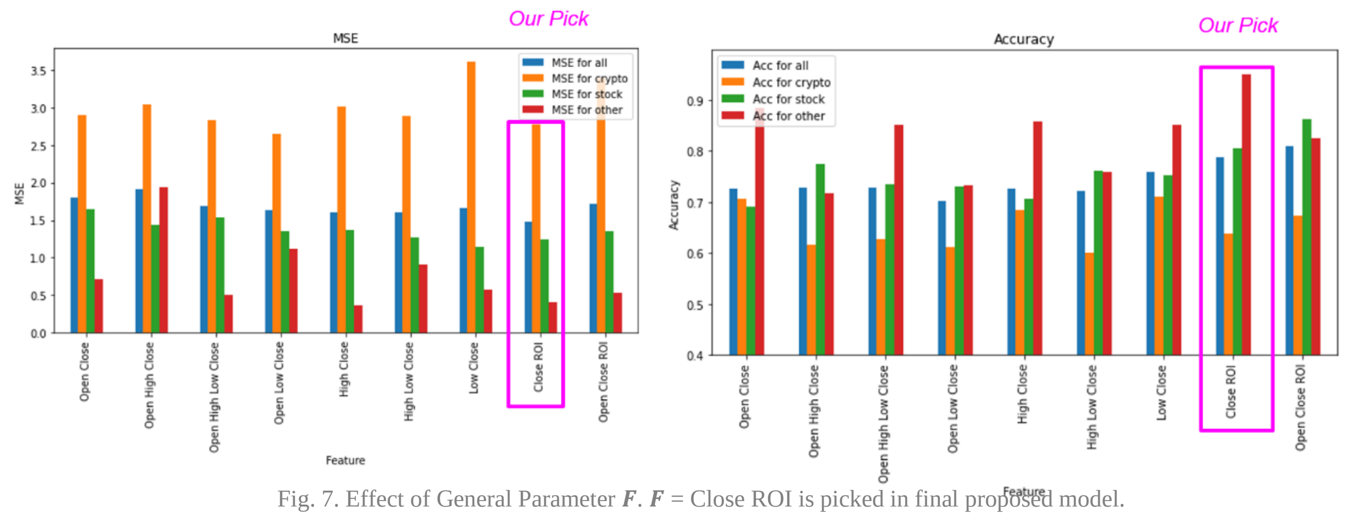

Fig. 7. Effect of General Parameter $\boldsymbol{F}$. $\boldsymbol{F}$ = Close ROI is picked in final proposed model.

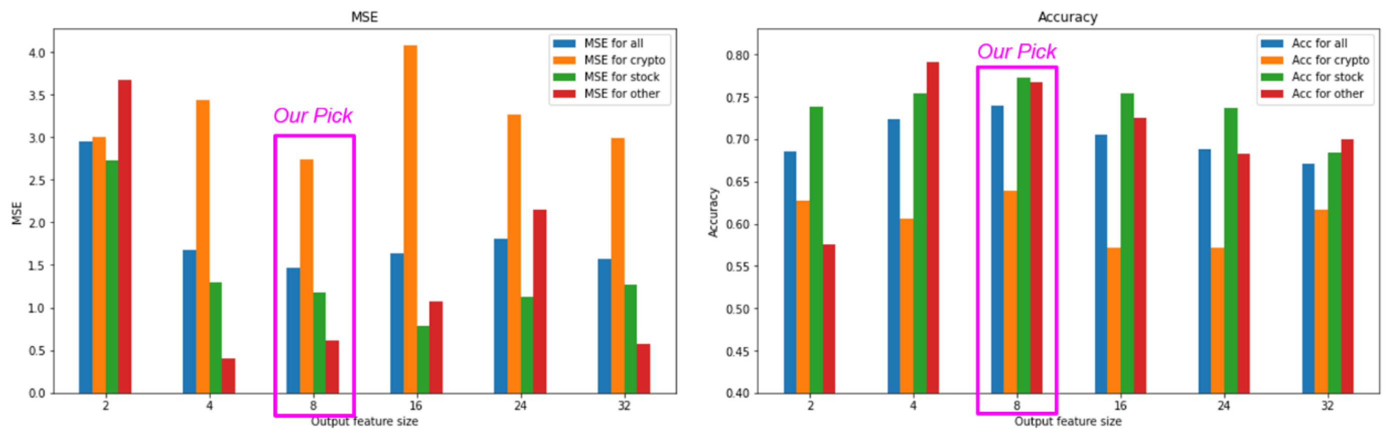

Fig. 8. Effect of GAT Hyperparameter $\boldsymbol{O}$. $\boldsymbol{O}$ = 8 is picked in final proposed model.

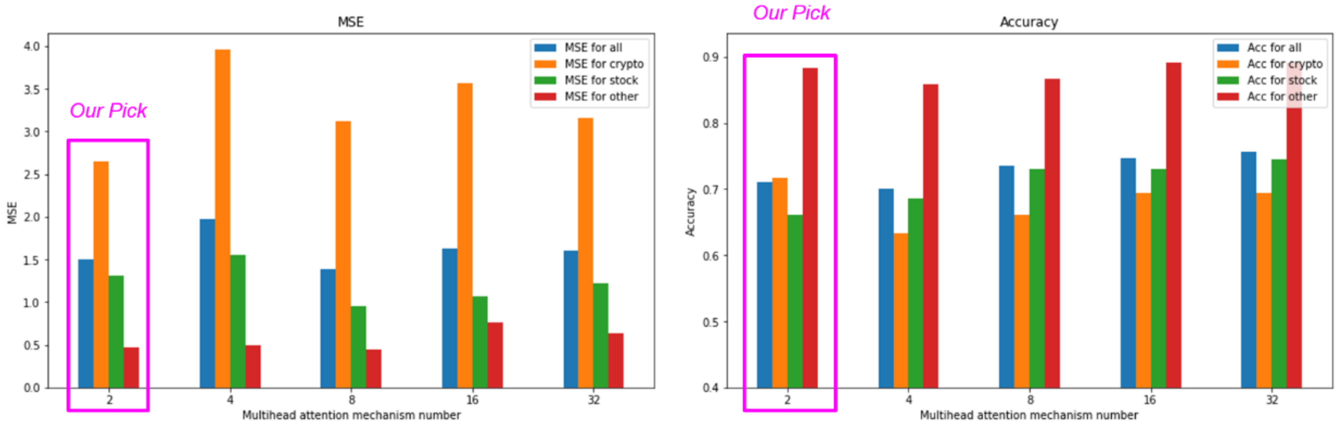

Fig. 9. Effect of GAT Hyperparameter $\boldsymbol{H}$. $\boldsymbol{H}$ = 2 is picked in final proposed model.

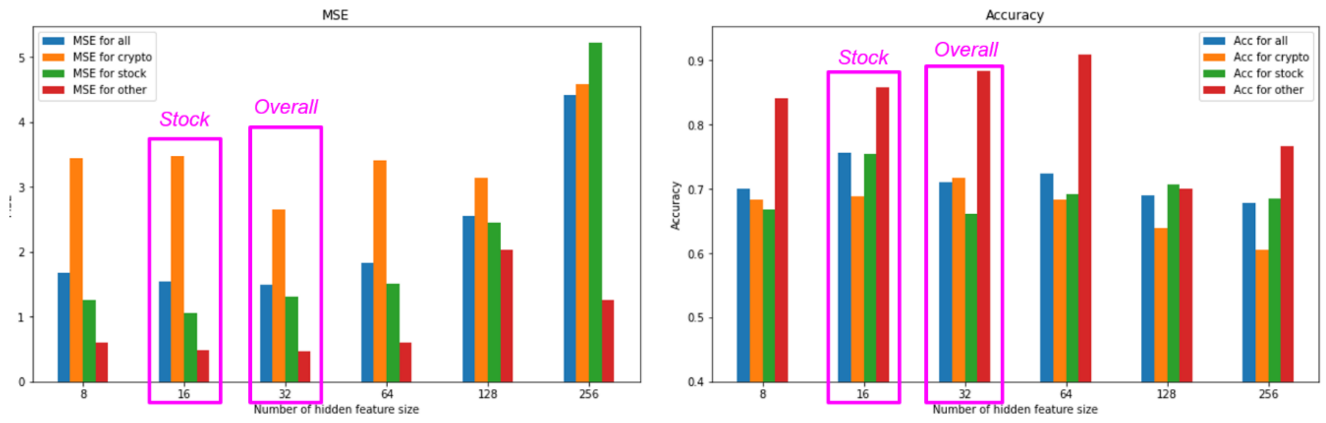

Fig. 10. Effect of LSTM Hyperparameter $\boldsymbol{L}$. $\boldsymbol{L}$ = 16 is for stock asset and $\boldsymbol{L}$ = 32 is for crypto asset and the other assets in final proposed model.

D. LSTM Hyperparameter

There is one hyperparameter $\boldsymbol{L}$ in LSTM component of the hybrid model which represents the number of hidden feature size in LSTM. As our final model uses different LSTM for different types of assets, we need to determine which $\boldsymbol{L}$ is suitable for each type of asset. Here in Figure 10 also reveals it is true that using different $\boldsymbol{L}$ for different asset could improve the model performance.

E. Final Proposed Model

Figure 11 shows the performance comparison of our final purposed model ($\boldsymbol{N}$ = 6, $\boldsymbol{F}$ = Close ROI, $\boldsymbol{O}$ = 8, $\boldsymbol{H}$ = 2, $\boldsymbol{L}$ for stocks = 16; $\boldsymbol{L}$ for crypto and others = 32) with baseline model ($\boldsymbol{N}$ = 6, $\boldsymbol{F}$ = Close, $\boldsymbol{L}$ = 32). Regardless of the model, cryptocurrency always has a significantly greater amount of MSE compare with the other two types of assets. Although the overall MSE for our proposed final GAT+LSTM model is still higher than the baseline pure LSTM model, adding GAT significantly increases the model's accuracy of prediction trend. The overall average accuracy for baseline model is 46% while our final model is 80%. For the asset other than crypto and stocks, we gained a 98% of accuracy.

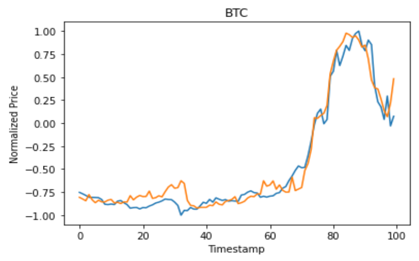

Fig. 11.1 Visualization of Bitcoin Actual price (blue) vs GAT+LSTM Predicted price (orange)

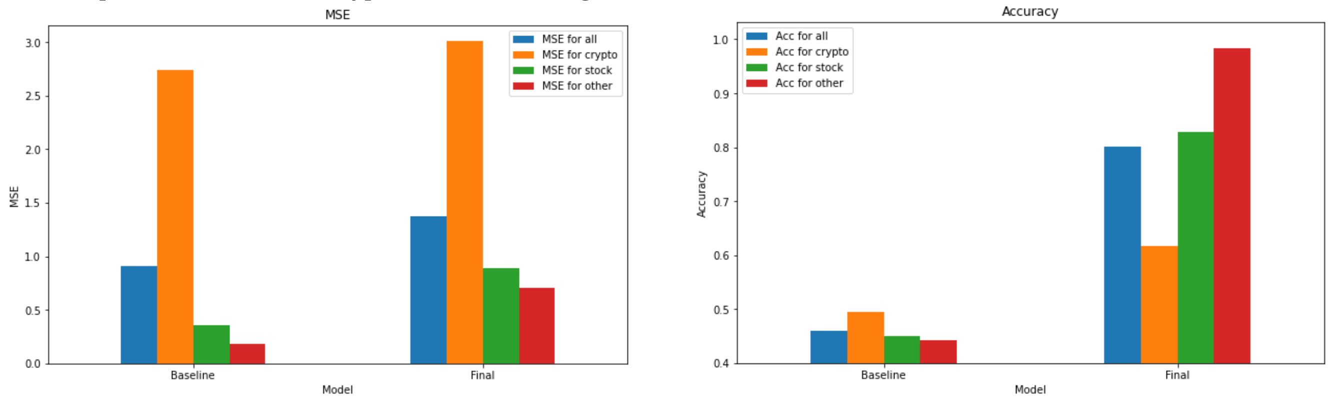

Fig. 11. MSE and Accuracy comparison of baseline model and the final proposed model

F. *Attention Coefficient*

In our proposed model, an attention coefficient matrix is learned for each time point(week). This attention coefficient matrix represents the weight GAT learned for each edge of the graph which tells us the relationship between assets. However, only when the hybrid model has a very good performance can the attention coefficient matrices become meaningful.

There are 37 commodities in our experiment result. The GAT initialized attention coefficient for each edge in the graph with 1/37 = 0.027. Because the graph is fully connected, the threshold is set to 0.04 to display the main relationship, which means the edges with a weight less than 0.04 will be deleted in the graph.

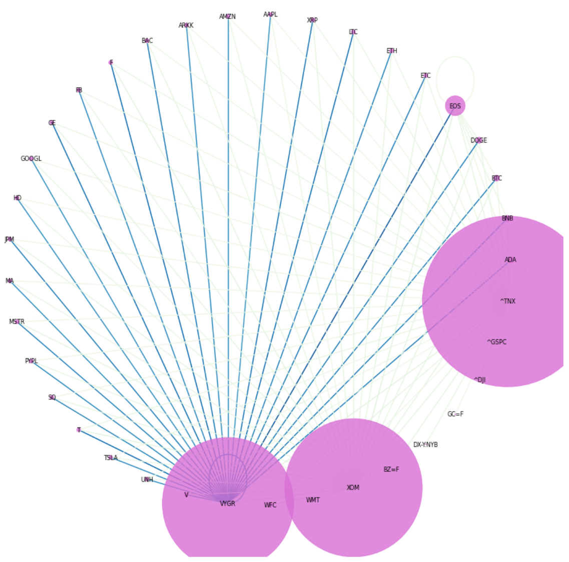

Fig. 12. The first week in the training set with GAT learning

After GAT training is completed, the graph in the first week of the training set is shown as the Figure 12, the size of the pink circle indicates that how many edges the commodity has. The depth of the colour of the edges shows the weight of the edges. In the first week, many stocks and cryptocurrencies are highly related to VYGR. The attention weight is different between each week, but adjacent weeks have little to no difference. The Figure 13 depicts the 34th week in the training set, most stocks and cryptocurrencies are highly related to T this time.

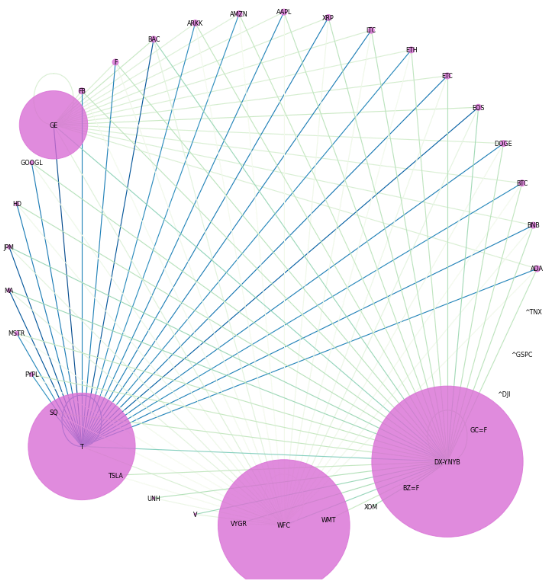

Fig. 13 The 34th week in the training set with GAT learning

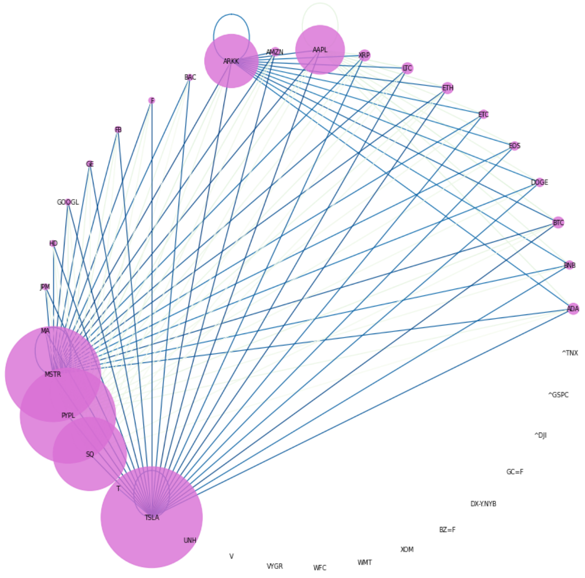

Fig. 14. The 84th week in the training set with GAT learning

As Figure 14 shows that the big pink circles are the technology and information companies, and they are highly related to each other. The Figure 15 shows the 18th week in the test set, all the cryptocurrencies connect to each other this week.

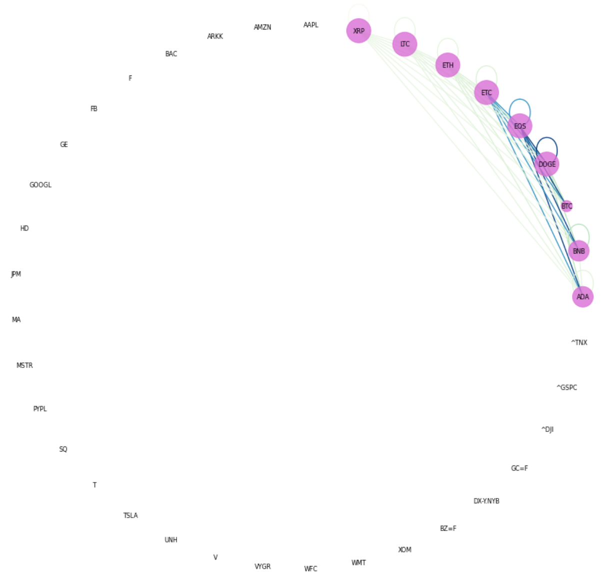

Fig. 15. The 18th week in the testing set with all cryptocurrencies

VI. FUTURE WORK AND DISCUSSION

A. Loss Function

During the training of the model, our loss function is MSE alone $Loss = \frac{1}{n}\sum_{i=1}^{n}(\widehat{y_i} - y_i)^2$. However, just as we have stated in the evaluation section, accuracy of trend prediction is also very important in terms of model performance. We could extend the Loss function $Loss = \frac{1}{n}\sum_{i=1}^{n}(\widehat{y_i} - y_i)^2 + \alpha \times Acc$ where $\alpha \in (0,1)$ and Acc is accuracy of trend prediction. The extended loss could further improve the model's accuracy of trend prediction performance.

B. Multilayer Perceptron

Our team also experimented with latent feature transformation via the use of Multi-Layer Perceptron. The idea behind this addition to our model was to compare performance between our model's learning capabilities between Euclidean and Latent space. We conducted a series of experiments by implementing a basic version of PyTorch MLP layer prior to the GAT module so that the input features would be transformed to exist in a tightly compressed latent space, however even after extensive experimentation with PyTorch MLP hyperparameters we were unable to see a significant increase in performance.

As a side effect of this change, our training time increased by upwards of 60 minutes to a total of 100+ minutes, therefore we deemed the latent space transformation out of scope for the main objective of our study, and instead decided to better allocate our time and effort on more tangible increases to overall performance. With that said, now that the basis for MLP latent transform has been integrated into our model, it is within our intention to revisit this concept and conduct a thorough experimentation of latent vs Euclidean input feature performance review, as we know this is one of the core strengths of neural network based algorithms [14].

C. Gated Recurrent Unit

We have already implemented GAT+GRU version which replace LSTM component with GRU. However, each run of the GAT+GRU model costs almost the same training time as our proposed GAT+LSTM. In the future, we could carry out the exact same experiments on GAT+GRU model and compare the obtained results with our proposed GAT+LSTM model.

VII. CONCLUSION

In this work, we presented a novel application of GAT to financial time series data. We overcome the time-unaware problem of GAT by combining GAT with LSTM to construct a hybrid model. By using GAT for feature embedding to every timestamp of the data, our GAT+LSTM model is able to learn the relationship between assets and make use of the learned relationship for forecasting. Our work could also be a guideline for applying graph neural networks on time series forecasting problems. In short, our proposed GAT+LSTM model is performs exceptionally well at trend prediction but with a minor drawback - it performs reasonably poorer than pure LSTM model in terms of prediction error.

In summary, our project was sparked by a shared curiosity to apply bleeding-edge GNN algorithms to solve a well-documented classical ML problem. Through this process we explored the different avenues in which GAT, LSTM, GRU and MLP are able to complement each other while conceptualizing our final GAT+LSTM architecture. We hope this report has been helpful for budding academics and domain experts alike in fully understanding how the age-old problem of stock return prediction can be tackled with GNNs. We invite the reader to join us in further contribution to this highly applicable ML problem.

VIII. Author Contribution

Kwong Chun WU contributed to the Related Works and Methodology sections of the report, with additional efforts towards ensuring the consistency between text and figures in the final report. Further responsibilities included data collection and feature engineering throughout the entire project.

Kam Leong AO contributed to the Related Works and Methodology sections of the report, with additional efforts towards ensuring the consistency of references in the final report. Further responsibilities included data collection & preprocessing throughout the entire project.

Mark CHEN contributed to the Evaluation and Results, as well as the Future Work and Conclusion sections of the report, with a main focus on GAT+LSTM architecture coding and optimization throughout the project.

Yong YU contributed to the Methodology and Results sections of the report, with a main focus on the GAT+GRU components as well as attention coefficient extraction and visualization throughout the entire project.

Yifei LIU contributed the Abstract, Introduction, and GAT+MLP sections of the report, with additional efforts towards the fluidity & formatting of the final report. Further responsibilities included overseeing the overall direction of the project and the organization of ideas for presentation and reports. We believe all members have contributed equally as part of this project.

Special thanks to Dr. Kaiqi Zhao, Katerina Taskova, and Dr. Thomas Lacombe from the University of Auckland's Computer Science department for their feedback and guidance in the duration of this 12-week research project.

References

[1] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, & Y. Bengio, "Graph attention networks," 2017, https://arxiv.org/abs/1710.10903.

[2] E. F. Fama and K. R. French, "Common risk factors in the returns on stocks and Bonds," *Journal of Financial Economics*, vol. 33, no. 1, pp. 3–56, 1993, https://doi.org/10.7208/9780226426983-018.

[3] F. Black and M. Scholes, "The pricing of options and corporate liabilities," *Journal of Political Economy*, vol. 81, no. 3, pp. 637–654, 1973, https://doi.org/10.1142/9789814759588_0001.

[4] Y. Fu and C. Soman, "Real-time data infrastructure at uber," Proceedings of the 2021 International Conference on Management of Data, 2021, https://doi.org/10.1145/3448016.3457552.

[5] V. Kosaraju, A. Sadeghian, R Martín-Martín, I. Reid, S. H. Rezatofighi, and S. Savarese, "Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks," 2019, https://arxiv.org/abs/1907.03395.

[6] J.-weon Jung, H.-S. Heo, H.-J. Yu, and J. S. Chung, "Graph attention networks for speaker verification," *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, https://doi.org/10.1109/ICASSP39728.2021.9414057.

[7] P. Ganesh and P. Rakheja, "Deep reinforcement learning in high frequency trading," 2018, https://arxiv.org/abs/1809.01506.

[8] G. Van Houdt, C. Mosquera, and G. Nápoles, "A review on the long short-term memory model," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5929–5955, 2020, https://doi.org/10.1007/s10462-020-09838-1.

[9] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2017, https://doi.org/10.1109/TNNLS.2016.2582924.

[10] J. Gao, X. Ying, C. Xu, J. Wang, S. Zhang, and Z. Li, "Graph-based stock recommendation by Time-aware relational attention network," *ACM Transactions on Knowledge Discovery from Data*, vol. 16, no. 1, pp. 1–21, 2021, https://doi.org/10.1145/3451397.

[11] K. Cho, B.Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation, " 2014, https://arxiv.org/abs/1406.1078.

[12] A. Dutta, S. Kumar, and M. Basu, "A gated recurrent unit approach to bitcoin price prediction," *Journal of Risk and Financial Management*, vol. 13, no. 2, p. 23, 2020, https://doi.org/10.3390/jrfm13020023.

[13] M. O. Rahman, M. S. Hossain, T. S. Junaid, M. S. A. Forhad,and M. K. Hossen, "Predicting prices of stock market using gated recurrent units (GRUs) neural networks," *Int. J. Comput. Sci. Netw. Secur*, vol. 13, no. 1, p. 213-222, 2019. https://www.researchgate.net/publication/331385031_Predicting_Prices_of_Stock_Market_using_Gated_Recurrent_Units_GRUs_Neural_Networks

[14] S. Kim, N. Winovich, H.-G. Chi, G. Lin, and K. Ramani, "Latent transformations neural network for object view synthesis," *The Visual Computer*, vol. 36, no. 8, pp. 1663–1677, 2019, https://doi.org/10.1007/s00371-019-01755-x.

Our Code Repository:

https://github.com/ProfiterolePuff/DL_Stock_Prediction