Hindawi
Security and Communication Networks
Volume 2023, Article ID 9840831, 1 page
https://doi.org/10.1155/2023/9840831

Retraction

Retracted: Beat Wash-Sale Tax with Multigraph Convolutional Neural Networks Based Trading Strategy

Security and Communication Networks

Received 20 June 2023; Accepted 20 June 2023; Published 21 June 2023

This article has been retracted by Hindawi following an investigation undertaken by the publisher [1]. This investigation has uncovered evidence of one or more of the following indicators of systematic manipulation of the publication process:

(1) Discrepancies in scope
(2) Discrepancies in the description of the research reported
(3) Discrepancies between the availability of data and the research described
(4) Inappropriate citations
(5) Incoherent, meaningless and/or irrelevant content included in the article
(6) Peer-review manipulation

The presence of these indicators undermines our confidence in the integrity of the article's content and we cannot, therefore, vouch for its reliability. Please note that this notice is intended solely to alert readers that the content of this article is unreliable. We have not investigated whether authors were aware of or involved in the systematic manipulation of the publication process.

Wiley and Hindawi regrets that the usual quality checks did not identify these issues before publication and have since put additional measures in place to safeguard research integrity.

We wish to credit our own Research Integrity and Research Publishing teams and anonymous and named external researchers and research integrity experts for contributing to this investigation.

The corresponding author, as the representative of all authors, has been given the opportunity to register their agreement or disagreement to this retraction. We have kept a record of any response received.

References

[1] Q. Wang and W. Jiang, "Beat Wash-Sale Tax with Multigraph Convolutional Neural Networks Based Trading Strategy," *Security and Communication Networks*, vol. 2022, Article ID 3598285, 18 pages, 2022.

Hindawi
Security and Communication Networks
Volume 2022, Article ID 3598285, 18 pages
https://doi.org/10.1155/2022/3598285

Research Article

Beat Wash-Sale Tax with Multigraph Convolutional Neural Networks Based Trading Strategy

Qinan Wang[1] **and Weiwei Jiang**[2]

[1]*Gabelli School of Business, Fordham University, New York, NY 10023, USA*
[2]*Department of Electronic Engineering, Tsinghua University, Beijing 100084, China*

Correspondence should be addressed to Qinan Wang; qwang195@fordham.edu

Received 2 June 2022; Revised 4 July 2022; Accepted 6 July 2022; Published 30 July 2022

Academic Editor: Mohammad Ayoub Khan

Stock forecasting is a method that uses historical data and mathematical models to predict the future movement of stocks. It gives an indication of how much profit or loss an investment can make. The use of machine learning for stock forecasting has been widely. But many studies do not take into account correlations between stocks and likelihood that frequent trading could trigger the wash-sale tax rule. Higher taxes cost could offset positive profits. In this study, we proposed a framework based on graph convolutional network, extracting the interdependencies of stocks to increase the prediction accuracy to 62%. Also, we included tax in the calculation of overall net income in simulated trading and tried different constraints on trades to see whether our new model can generate profits high enough to cover the required taxes. The results with 795.5% net return for two years validated the effectiveness of our model and trading strategy.

1. Introduction

In order to help investors to make better investment decisions, stock forecasting has become a popular tool. Stock forecasting is a method of predicting the future movement direction of a stock. The prediction is made by using historical data and applying mathematical models. It will give a hint about how much profit or loss can be obtained from the investments.

Besides prediction, tax is also an important problem in stock trading. A tax rule called wash-sale rule is an IRS regulation that prevents high-frequency trades from creating artificial loss to deduct tax. It occurs when an investor sells or trades a security at a loss and buys the same one within a month. When this rule is triggered, it makes the initial loss uncountable for tax reduction. Although our intention of trading is not to deduct tax but to make profits, trading at a high frequency might trigger this rule and lead to unnecessary heavy taxes. With the wash-sale rule, investors may pay tax much more than profit when making incorrect trades.

Traditional forecasting methods can be divided into three categories: fundamental, technical, and their combination. Fundamental analysis is concerned with analysing a company's financial statements to forecast its future performance. Technical analysis focuses on analysing stock price past movement patterns. The combination method involves both fundamental and technical analyses.

Newly developed machine learning stock forecasting is a technique that uses artificial intelligence (AI) to predict the future price of stocks. Machine learning stock forecasting can be used for both short-term and long-term forecasts, but it is most commonly used for making short-term predictions. There are two main types of machine learning stock forecasting, namely, regression analysis and neural networks. Neural networks are more accurate than regression analysis, but they are also slower at making predictions because there are many parameters involved in predicting a stock's price movement. Over the last few years, neural networks have become more popular due to their reduced training data. This means that we do not need as much historical data when using them compared with regression analysis or other techniques like technical indicators, which makes them very useful for traders who want to use only a small amount of historical data when making their predictions on any given day.

However, most of the stock forecasting literatures are only based on the historical data of the stock itself, market data, and news. The absence of stocks' correlation factor may bring uncertainty to the result. Therefore, we applied a new machine learning model, graph convolutional neural network (GCN), on stock forecasting to extract the interdependencies between stocks. To take advantage of it, we proposed a multigraph construction method and a GCN-based forecasting framework with multiple graphs as inputs. At last, we proposed a trading simulation system with stock movement predictions as trading signals. The experimental results with a 62% win rate validate the effectiveness of our proposed stock forecasting and trading methods.

Moreover, many existing literatures proposed trades within one month but do not include wash-sale rule. In this paper, we evaluated tax-excluded net income and profit. The comparison of them shows that the wash-sale tax can be a huge cost and offset the profit. But our model is still profitable with a 795.5% return for two years after tax.

The rest of this paper is organized as follows. Section 2 discusses related studies. Section 3 introduces forecasting and trading frameworks. Sections 4 and 5 present trading evaluation and discussion. Section 6 concludes this paper.

2. Related Work

2.1. Machine Learning Methods for Stock Prediction. A number of researchers have explored usefulness of machine learning model on stock prediction. A study was conducted to predict the future values of the stock index using two-stage fusion. The first stage uses Support Vector Regression (SVR), and the second stage uses Artificial Neural Network (ANN) and Random Forest (RF) to create fusion models. Then, the results are compared with those of single-stage models with SVR, ANN, and RF. The first stage predicts statistical parameters in the future that will be input in the second stage. The results showed that the two-stage model was more accurate than the single-stage models [1]. Another study proposed an algorithm that can exploit the temporal correlation of global stock market and financial products and uses SVM model and other regression models to predict the stock trend of next day [2]. Another study used Logistic Regression, Gaussian Discriminant Analysis, Quadratic Discriminant Analysis, and SVM to predict the next day and long-term trend of stock movement and found that although it is hard to predict the next day trend with high accuracy, long-term trend prediction was able to have high accuracies [3]. Another study also showed that predicting stock movement of long term has higher accuracies than predicting the movement of next day. The study tried to predict the direction and strength of stock movement of next day and a week later using Multinomial Logistic Regression, Linear Discriminant Analysis, K-Nearest Neighbours Algorithm, and Multiclass Support Vector Machine [4]. A study tried to predict the stock movement using recurrent reinforcement learning. The result showed that trading with neural network's aid still has high variance during volatile periods [5].

2.2. The Development of GCN and Applications in the Financial Domain. GCN is a state-of-the-art model attracting considerable critical attention [6]. It is a type of deep learning algorithm that uses graph to learn how to recognize objects in images. The main idea behind the GCN is that we can use the structure of an image as a representation for our data and then apply this representation to predict the object present in it [7]. Recently, GCN and subsequent variants have been applied in various areas, including social networks, chemistry, natural language processing, and computer vision [8–14]. However, it is has yet not commonly used in financial field.

A study conducted early in 2005 used a graphic neural network for computing customized web page rank values and was able to show strong learning capacity [15]. Some recent studies have been conducted to use GCN in other areas. A study conducted in 2020 used a multimodel graphic neural network for microvideo recommendations [16]. The result significantly outperformed other popular recommendation methods. Another study was conducted in 2021 to use graphic neural network in traffic prediction [17]. It aimed to transform transportation information into optimized graphs to be input into the network so it can learn the relationships between road segments and predict transportation conditions.

2.3. The Automated Trading Systems. Up to now, several studies have attempted to evaluate the feasibility of automated trading systems using various methods, including classical time series prediction and machine learning [18]. Here, we gave some of the latest literature review about this topic.

A study constructed trading system based on ANN and triple Exponential Moving Average (EMA) [19]. It turned to use ANN by demonstrating the ARIMA predictions of stock price completely out of measure. The network with input of 5-day data predicted the next day opening highest, lowest, and closing (OHLC) stock price, respectively, by adopting adaptive moment (Adam) optimization algorithm and Mean Absolute Error (MAE) loss function. For the part of the trading strategy, it combined triple EMA and ANN to define the entry/exit rules: when predicted lowest or highest stock price is lower than triple EMA lowest or highest, and predicted closing or opening is lower than the triple EMA closing or opening, the system will buy in; when predicted lowest or highest is higher than triple EMA lowest or highest, and predicted closing or opening is higher than the triple EMA closing or opening, the system will exit current position.

Another study proposed a pattern-based stock trading system and also was predicted by ANN [20]. It applied three algorithms to form the clusters of data with high fluctuation patterns, with which input features including distance between MA and current price, rate of change (RC),

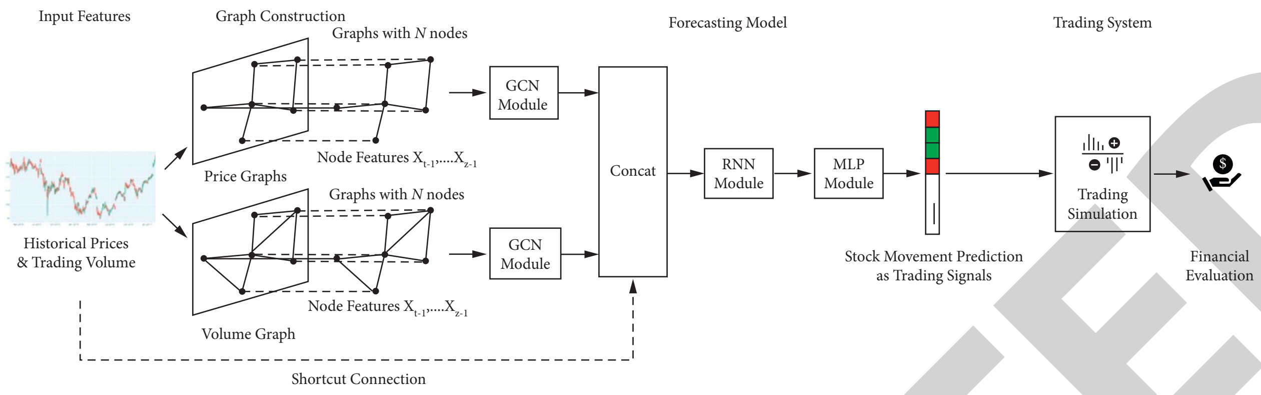


Figure 1: GCN-based forecasting framework with multiple graphs as inputs.

candlestick body, upper shadow (US), lower shadow (LS), opening/highest/lowest price, slope of the volume moving average line, difference between the volume moving averages, and the total volume, are calculated. All features are normalized to 0~1. Finally, the neural network will output a binary result marked by 1 if the price rises by more than 10% within 5 days and otherwise it is 0. The experimental result showed achieved accuracy of 96.23%. The trading policy of this system was 20% in profit realization rate and −12% in stop loss rate, with holding period of 19 days. A fund simulation showed a profit rate of 65% within 8 months.

Some derived machine learning method is also popular in trading system. A long-short term memory based on leading indicators (LSTMLI) was used to classify the change of stock prices [21]. +0.01 and −0.01 are cutoff points of price rise and fall, between −0.01 and −0.01 is classified as unchanged. Then, a genetic algorithm is used to find the threshold of trading signals. It initialized the chromosome by producing two values for buy signal and sell signal. One of the two signals will be output if the predicted value is higher than corresponding chromosome value. After signal appears, Kelly criterion is used to optimize proportion of money invested in stock. The experimental results showed that Kelly criterion helps obtain much higher profit.

The above literature discussed various evaluation metrics for models, profit, and risk. Mean Square Error (MSE), Root Mean Square Error (RMSE), MAE, Mean Absolute Percentage Error (MAPE), and Explained Variance Score (EVS) were applied to evaluate continuous prediction models' performance. Accuracy, precision, recall, and $F1$ score are used to evaluate binary prediction performance. Maximum drawdown (MDD), Sharpe ratio (SR), Sortino ratio (SoR), and Calmar ratio (CR) were used for estimating the potential loss in value of the stock. The goodness of each trade was calculated by simple returns.

As one of the new efficient machine learning methods, reinforcement learning plays an important role in the trading system as well [22–24]. It involves using a reward function that specifies how the system should behave. The algorithm then learns how to maximize the reward function by performing actions that lead to a higher reward and learns from its own past experiences how to improve future rewards. This paper did not include reinforcement learning but we consider it as a future research direction.

3. Framework

The whole framework proposed in this study is shown in Figure 1. The historical prices and trading volume are used as input features. Multiple graphs are built first, with N stocks as the nodes. The historical prices and trading volume in the past L days are used as node features, in which the node feature for a single day is $X_i \in R^{N\times 5}$, where the open, high, low, and close prices and trading volume add up to a total of 5 numerical values for graph i. Then, GCN modules are leveraged to capture the interdependencies among different stocks and the output is $Y_i \in R^{N\times 5}$. Then, the shortcut connection is added to combine the original features with GCN module outputs into $Z \in R^{L*M}$, where L is the look-back window and $M = N \times 5 \times (1 + K)$; N is the stock number and K is the graph number used. Then, RNN module is further used to extract the temporal dependency and MLP module is to create binary movement prediction. Finally, the proposed trading system is used to conduct the trading simulation and financial evaluation.

3.1. Graph Construction Method. In this study, the financial graphs are built as correlation graphs, to model the mutual influence among stocks. Choose two historical input time series t_i, t_j from two different stocks i, j, e.g., the open, high, low, and close prices or trading volume; the element $a_{i,j}$ of the adjacency matrix $A \in R^{N\times N}$ is calculated as the correlation between t_i and t_j; e.g., $a_{i,j} = corr\,(t_i, t_j)$. A truncated adjacency matrix can be further proposed and used, in which the absolute value of an individual element below a threshold is reset to zero, i.e., no relationship between the corresponding two stocks.

3.2. Forecasting Model

3.2.1. GCN Module. In this study, GCN [6] is proposed to extract the interdependencies among different stocks, which is a truncated expansion in terms of Chebyshev polynomials up to 1st order from Chebyshev's spectral CNN. Given the

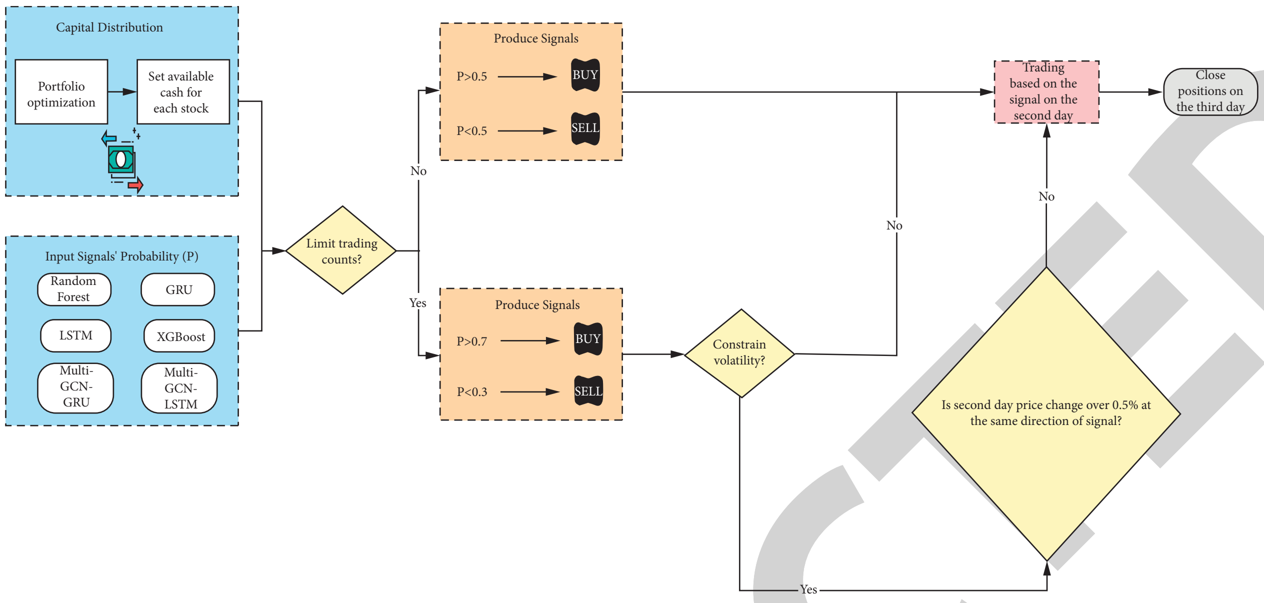


Figure 2: Trading system design.

adjacency matrix A for a single graph and the node features X in a day, the output from the GCN module is as follows:

$$Y = W\left(I_N + D^{-1/2}\mathrm{AD}^{-1/2}\right)X, \tag{1}$$

where W is the learnable parameter, I_N is the identity matrix, and D is the degree matrix in which each element is the number of neighbor nodes.

3.2.2. RNN Module. The recurrent neural network (RNN) module is used to capture the temporal dependency, in which the GCN output and the shortcut input are concatenated as the input variable. Two types of RNN are used in this study, namely, long-short term memory (LSTM) and gated recurrent unit (GRU), in which GRU is a simplified variant of LSTM.

3.2.3. MLP Module. To generate a binary movement direction prediction, a multilayer perceptron (MLP) module is further used as the feedforward part, which takes the RNN module output as a vector input and the binary stock movement prediction for N stocks as the output vector.

3.3. Trading System Design. The trading system is designed as shown in Figure 2. It sets available cash for each stock according to the portfolio optimization result and takes probability of signals to produce "Buy" or "Sell." The signal is calculated through two sets of thresholds. If we do not limit the trading counts, "Buy" will be output if probability is bigger than 0.5; otherwise "Sell" will be output. Then, we place orders matching with the signal on the second day. With the limitation on trading counts, we only allow "Buy" when probability is over 0.7, while allowing "Sell" when it is lower than 0.3. If there is no further constraint on volatility, we trade on the second day. However, if volatility constraint is considered, the trading will occur only when the second day price change does not exceed 0.5% at the same direction of signal. All positions will be closed on the third day.

Table 1: Ten stocks used in this study.

Symbol	Company	Market capitalization (in dollars)
AAPL	Apple	2.668T
MSFT	Microsoft	2.204T
GOOG	Alphabet (Google)	1.751T
AMZN	Amazon	1.482T
TSLA	Tesla	908.28B
BRK-A	Berkshire Hathaway	711.45B
NVDA	NVIDIA	571.56B
UNH	UnitedHealth	469.17B
V	Visa	436.35B
JPM	JPMorgan Chase	399.69B

4. Trading Evaluation

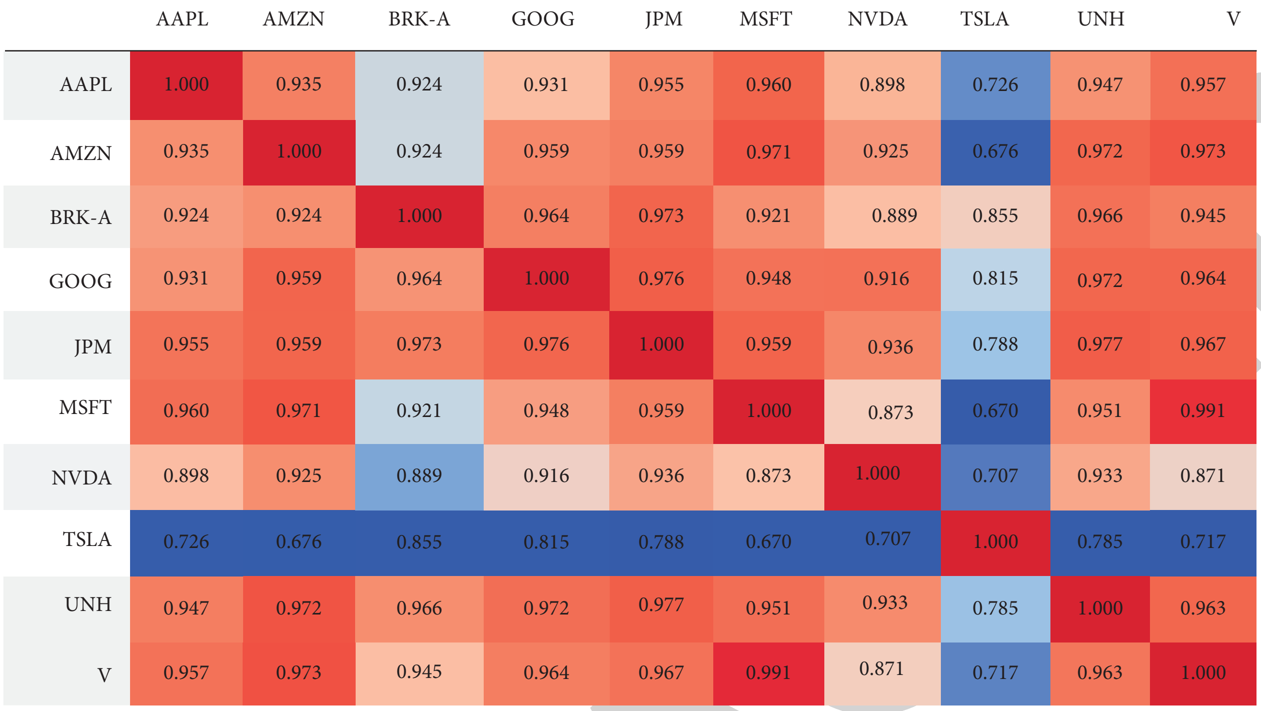
4.1. Settings

4.1.1. Input. In this study, we choose ten stocks traded in the US stock market, with the largest market capitalization and an IPO date before January 1, 2012. Our selection is based on the market capitalization on March 3, 2022, and the selected stocks are listed in Table 1.

The whole time period considered in this study ranges from January 1, 2012, to December 31, 2021, and is split into training, validation, and test subsets as follows:

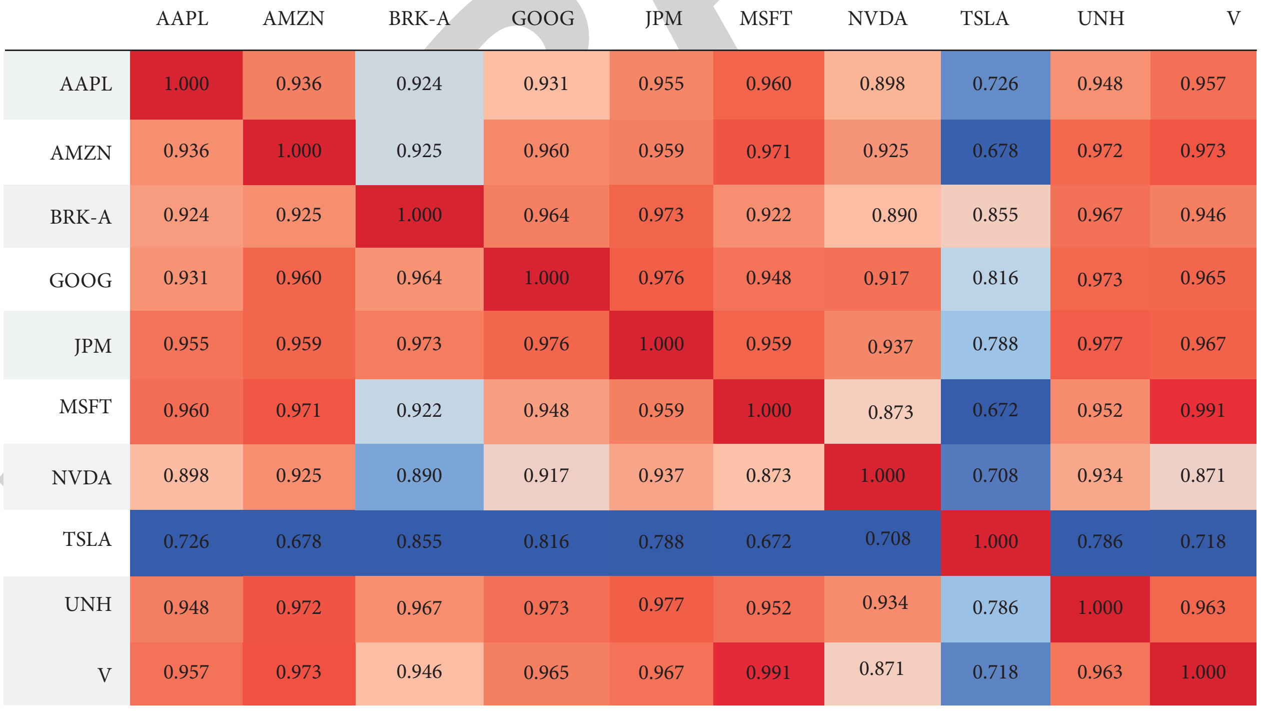
(i) Training: 2012-01-03 to 2017-12-31
(ii) Validation: 2018-01-01 to 2019-12-31
(iii) Testing: 2020-01-01 to 2021-12-30

We use the historical price and trading volume time series in the training time period to calculate the correlation graphs. The adjacency matrices for price graphs are shown in



	AAPL	AMZN	BRK-A	GOOG	JPM	MSFT	NVDA	TSLA	UNH	V
AAPL	1.000	0.935	0.924	0.931	0.955	0.960	0.898	0.726	0.947	0.957
AMZN	0.935	1.000	0.924	0.959	0.959	0.971	0.925	0.676	0.972	0.973
BRK-A	0.924	0.924	1.000	0.964	0.973	0.921	0.889	0.855	0.966	0.945
GOOG	0.931	0.959	0.964	1.000	0.976	0.948	0.916	0.815	0.972	0.964
JPM	0.955	0.959	0.973	0.976	1.000	0.959	0.936	0.788	0.977	0.967
MSFT	0.960	0.971	0.921	0.948	0.959	1.000	0.873	0.670	0.951	0.991
NVDA	0.898	0.925	0.889	0.916	0.936	0.873	1.000	0.707	0.933	0.871
TSLA	0.726	0.676	0.855	0.815	0.788	0.670	0.707	1.000	0.785	0.717
UNH	0.947	0.972	0.966	0.972	0.977	0.951	0.933	0.785	1.000	0.963
V	0.957	0.973	0.945	0.964	0.967	0.991	0.871	0.717	0.963	1.000

Figure 3: The adjacency matrix for open price graph.



	AAPL	AMZN	BRK-A	GOOG	JPM	MSFT	NVDA	TSLA	UNH	V
AAPL	1.000	0.936	0.924	0.931	0.955	0.960	0.898	0.726	0.948	0.957
AMZN	0.936	1.000	0.925	0.960	0.959	0.971	0.925	0.678	0.972	0.973
BRK-A	0.924	0.925	1.000	0.964	0.973	0.922	0.890	0.855	0.967	0.946
GOOG	0.931	0.960	0.964	1.000	0.976	0.948	0.917	0.816	0.973	0.965
JPM	0.955	0.959	0.973	0.976	1.000	0.959	0.937	0.788	0.977	0.967
MSFT	0.960	0.971	0.922	0.948	0.959	1.000	0.873	0.672	0.952	0.991
NVDA	0.898	0.925	0.890	0.917	0.937	0.873	1.000	0.708	0.934	0.871
TSLA	0.726	0.678	0.855	0.816	0.788	0.672	0.708	1.000	0.786	0.718
UNH	0.948	0.972	0.967	0.973	0.977	0.952	0.934	0.786	1.000	0.963
V	0.957	0.973	0.946	0.965	0.967	0.991	0.871	0.718	0.963	1.000

Figure 4: The adjacency matrix for high price graph.

Figures 3–6. Since the price graphs are similar with each other, only the close price graph is used later in our experiments. The adjacency matrix for trading volume graph is shown in Figure 7.

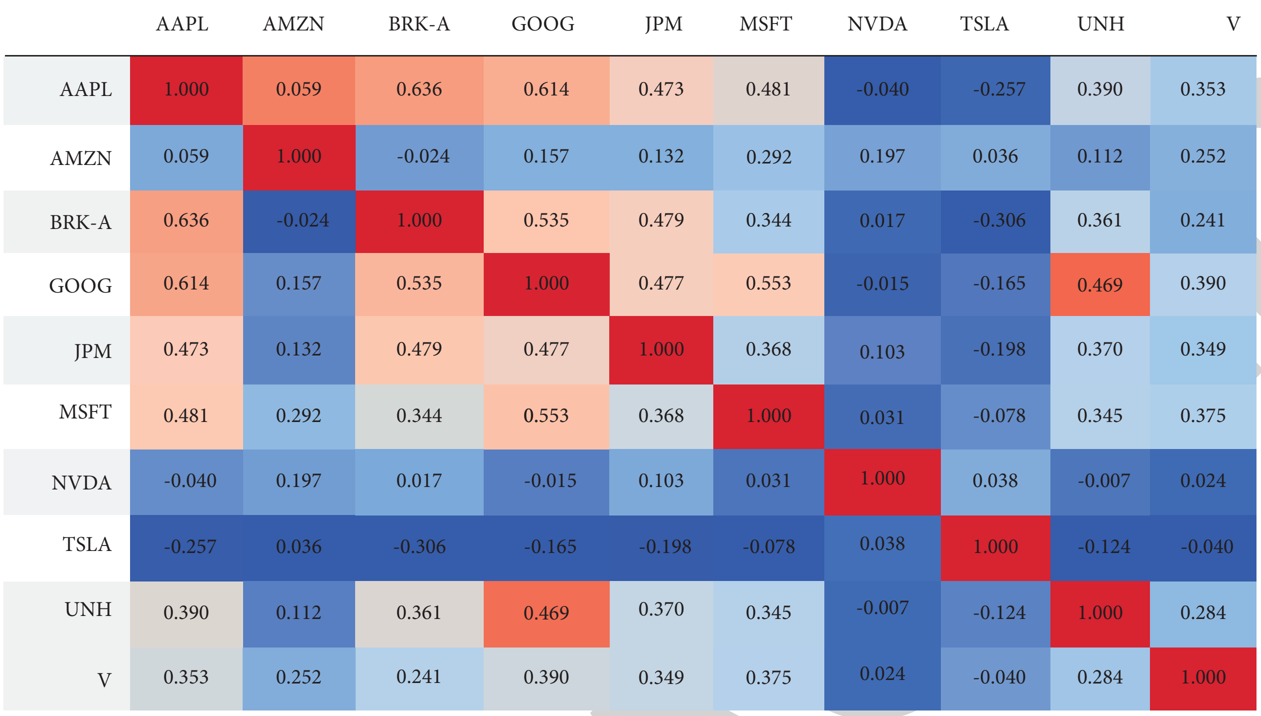
4.1.2. Model Settings. The historical prices and trading volume in the past ten days as the lookback window are used as input features in our forecasting framework, along with the close price and trading volume graphs.

	AAPL	AMZN	BRK-A	GOOG	JPM	MSFT	NVDA	TSLA	UNH	V
AAPL	1.000	0.936	0.925	0.932	0.956	0.960	0.899	0.727	0.947	0.957
AMZN	0.936	1.000	0.924	0.959	0.959	0.971	0.924	0.675	0.971	0.974
BRK-A	0.925	0.924	1.000	0.965	0.973	0.922	0.889	0.855	0.966	0.946
GOOG	0.932	0.959	0.965	1.000	0.976	0.948	0.916	0.814	0.972	0.965
JPM	0.956	0.959	0.973	0.976	1.000	0.959	0.935	0.788	0.977	0.967
MSFT	0.960	0.971	0.922	0.948	0.959	1.000	0.873	0.670	0.950	0.991
NVDA	0.899	0.924	0.889	0.916	0.935	0.873	1.000	0.706	0.933	0.871
TSLA	0.727	0.675	0.855	0.814	0.788	0.670	0.706	1.000	0.786	0.718
UNH	0.947	0.971	0.966	0.972	0.977	0.950	0.933	0.786	1.000	0.962
V	0.957	0.974	0.946	0.955	0.967	0.991	0.871	0.718	0.962	1.000

Figure 5: The adjacency matrix for low price graph.

	AAPL	AMZN	BRK-A	GOOG	JPM	MSFT	NVDA	TSLA	UNH	V
AAPL	1.000	0.935	0.924	0.931	0.955	0.960	0.898	0.727	0.947	0.957
AMZN	0.935	1.000	0.924	0.959	0.959	0.971	0.925	0.677	0.971	0.973
BRK-A	0.924	0.924	1.000	0.965	0.973	0.922	0.890	0.855	0.966	0.946
GOOG	0.931	0.959	0.965	1.000	0.976	0.948	0.916	0.815	0.972	0.965
JPM	0.955	0.959	0.973	0.976	1.000	0.959	0.936	0.788	0.977	0.967
MSFT	0.960	0.971	0.922	0.948	0.959	1.000	0.873	0.671	0.951	0.991
NVDA	0.898	0.925	0.890	0.916	0.936	0.873	1.000	0.708	0.933	0.871
TSLA	0.727	0.677	0.855	0.815	0.788	0.671	0.708	1.000	0.786	0.718
UNH	0.947	0.971	0.966	0.972	0.977	0.951	0.933	0.786	1.000	0.963
V	0.957	0.973	0.946	0.965	0.967	0.991	0.871	0.718	0.963	1.000

Figure 6: The adjacency matrix for close price graph.



	AAPL	AMZN	BRK-A	GOOG	JPM	MSFT	NVDA	TSLA	UNH	V
AAPL	1.000	0.059	0.636	0.614	0.473	0.481	-0.040	-0.257	0.390	0.353
AMZN	0.059	1.000	-0.024	0.157	0.132	0.292	0.197	0.036	0.112	0.252
BRK-A	0.636	-0.024	1.000	0.535	0.479	0.344	0.017	-0.306	0.361	0.241
GOOG	0.614	0.157	0.535	1.000	0.477	0.553	-0.015	-0.165	0.469	0.390
JPM	0.473	0.132	0.479	0.477	1.000	0.368	0.103	-0.198	0.370	0.349
MSFT	0.481	0.292	0.344	0.553	0.368	1.000	0.031	-0.078	0.345	0.375
NVDA	-0.040	0.197	0.017	-0.015	0.103	0.031	1.000	0.038	-0.007	0.024
TSLA	-0.257	0.036	-0.306	-0.165	-0.198	-0.078	0.038	1.000	-0.124	-0.040
UNH	0.390	0.112	0.361	0.469	0.370	0.345	-0.007	-0.124	1.000	0.284
V	0.353	0.252	0.241	0.390	0.349	0.375	0.024	-0.040	0.284	1.000

Figure 7: The adjacency matrix for trading volume graph.

Table 2: Movement prediction evaluation result.

Model	Accuracy	Recall	Precision	*F*1 score
Random Forest	0.509	0.633	0.542	0.584
XGBoost	0.511	0.636	0.543	0.589
GRU	0.513	0.722	0.539	0.617
LSTM	0.511	0.636	0.543	0.586
Multi-GCN-LSTM	0.621	0.711	0.635	0.671
Multi-GCN-GRU	0.622	0.764	0.625	0.687

Two GCN layers are used in the GCN module, using the same input and output feature size. Two RNN layers are used in the RNN module, using 100 neurons in each layer, for both LSTM and GRU. Two fully connected layers with 100 neurons are used in the MLP module as the hidden layers and the output layer has 10 neurons and the sigmoid activation function. The predicted movement direction is up (i.e., 1) if the output value is greater than 0.5 and down (i.e., 0) otherwise. For deep learning modules, ReLU is used as the activation function, binary cross entropy loss is used as the loss function, Adam is used as the optimizer, and the training epoch is set to 1000 with a batch size of 32.

LSTM and GRU models are both used as baselines. In other words, only the shortcut connection is used, without the graphs and GCN modules. Two machine learning models are further used as our baselines, namely, XGBoost and Random Forest models, with the hyperparameters searched with grid search in the validation set.

4.2. Evaluation Metrics

4.2.1. Movement Prediction Evaluation. The evaluation metrics used for binary movement prediction evaluation include accuracy, recall, precision, and *F*1 score. The evaluation results in the test set are shown in Table 2. Two variants of our proposed methodology, namely, Multi-GCN-LSTM and Multi-GCN-GRU achieve the best accuracy and *F*1 score with a close performance.

4.2.2. Financial Evaluation. These evaluations are based on simulated trading with Random Forest, XGBoost, GRU, LSTM, GCN-GRU, and GCN-LSTM signals. The portfolio's initial capital is $2 million. A baseline buying stocks with all cash on the first day and holding till the last day of backtesting is recorded for comparison. In the simulated trading with signals, we open position first with all cash when the price has been rising for last consecutive three days at close, and the signal generated yesterday is not "Sell." Then, on the second day after that, the system will sell the previously bought positions and then buy or sell with all cash based on the newest signal and the specific strategy. All positions will be liquidated on the following day.

The three trading strategies can buy long or sell short each day, depending on the model's prediction. They all have a general constraint with selling short that the margin is 1.5 times of market value. The first trading strategy has no

Table 3: 2021 short-term capital gains tax rates.

Tax rate	10%	12%	22%	24%	32%	35%	37%
Taxable income	Up to $9,950	$9,950 to $40,525	$40,525 to $86,375	$86,375 to $164,925	$164,925 to $209,425	$209,425 to $523,600	Over $523,600

Table 4: 2021 long-term capital gains tax rates.

Tax rate	0%	15%	20%
Taxable income	$0 to $40,400	$40,400 to $445,850	Over $445,850

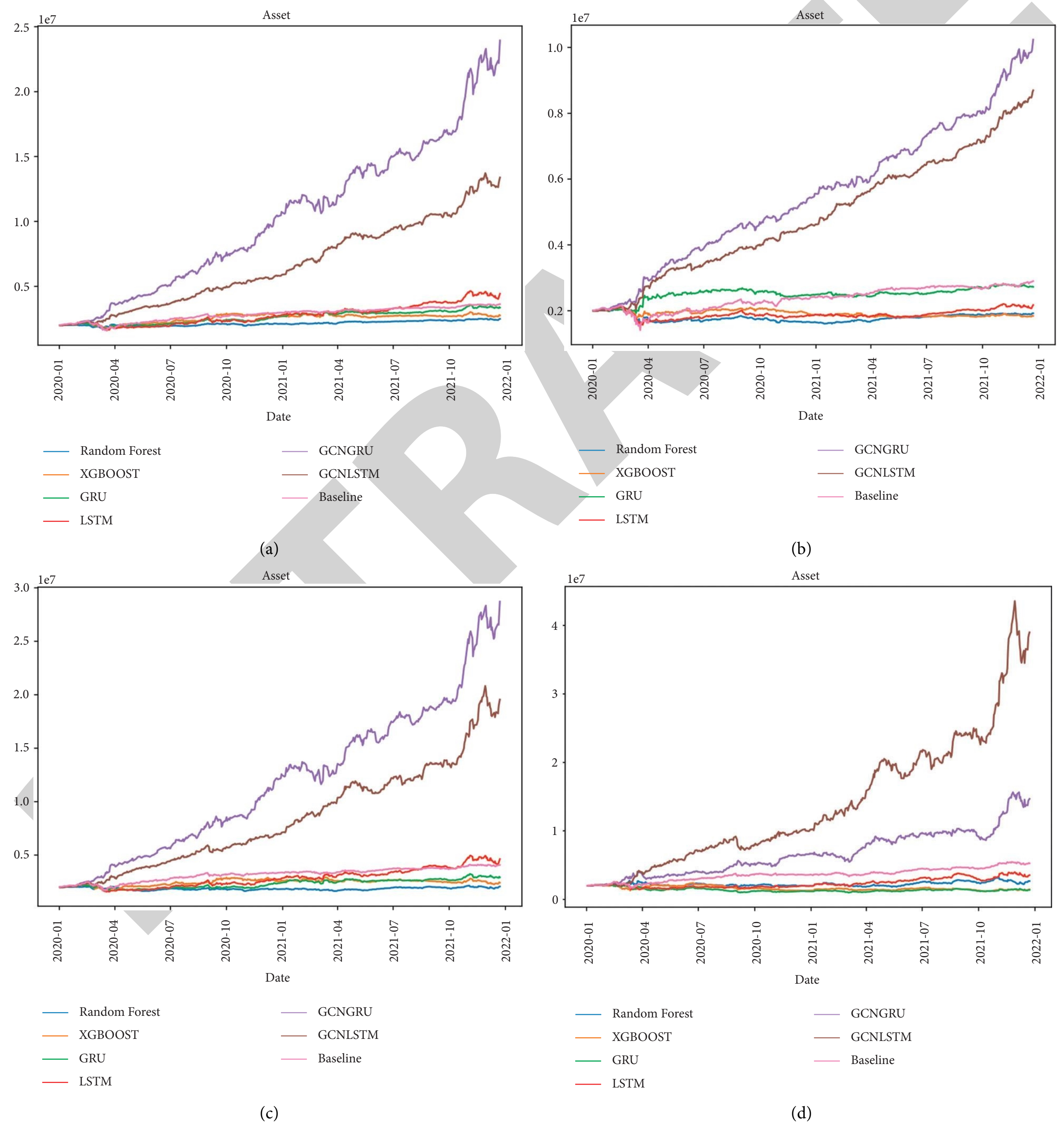


Figure 8: Asset change from 1/1/2020 to 12/31/2021 against different weights distribution for trading without limitation. (a) Portfolio with evenly distributed weights. (b)–(d) Portfolios with optimized weights against 20%, 30%, and 40% expected return in Markowitz portfolio optimization, respectively.

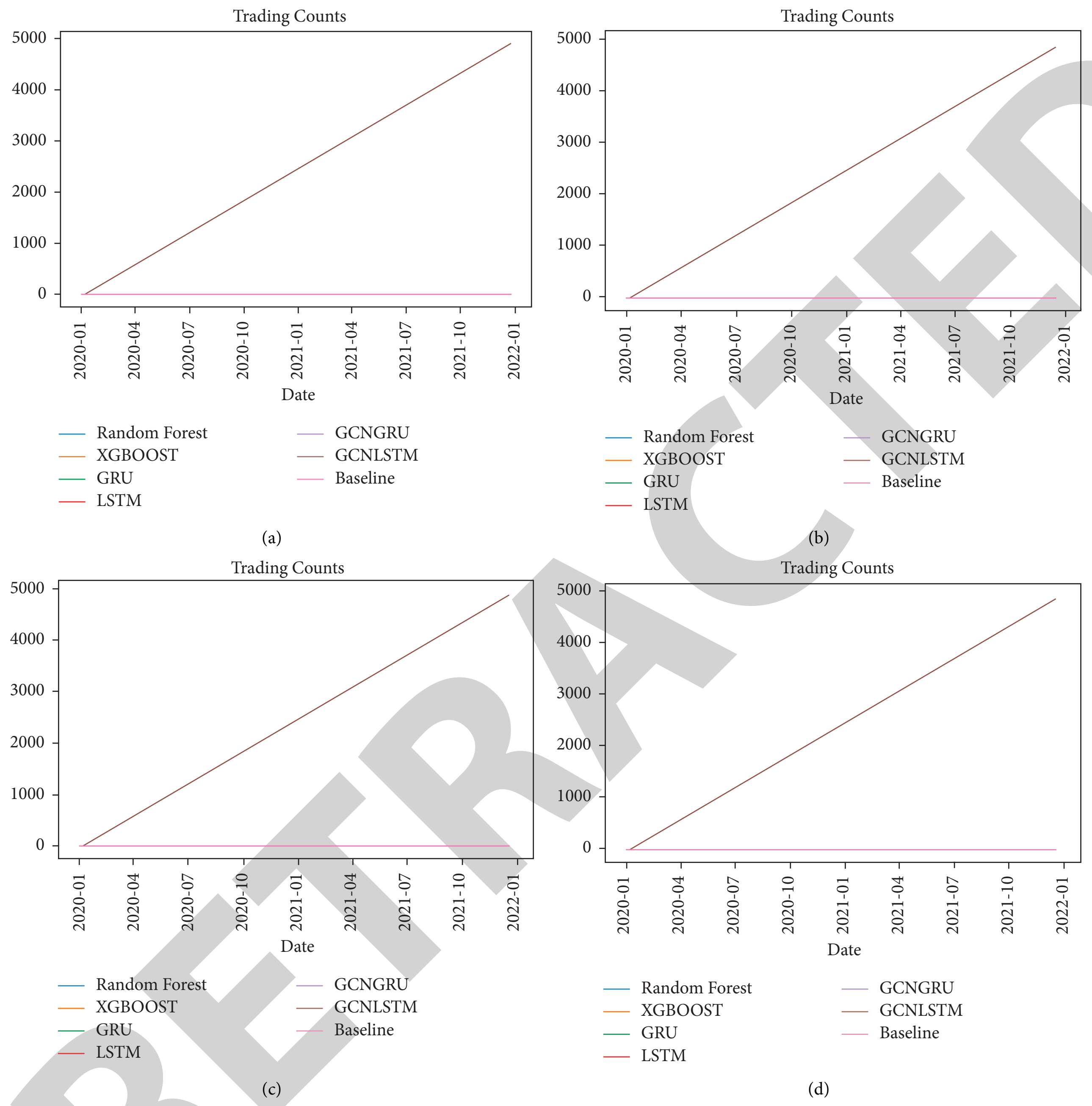


Figure 9: Trading counts from 1/1/2020 to 12/31/2021 against different weights distribution for trading without limitation. (a) Portfolio with evenly distributed weights. (b)–(d) Portfolios with optimized weights against 20%, 30%, and 40% expected return in Markowitz portfolio optimization, respectively.

special limits on trades. When our model anticipates the price to go up on the day after tomorrow (the probability of the price rising is higher than 0.5), the system will generate a "Buy" signal for the next day. Or when our model anticipates the price to go down on the day after tomorrow (the probability of the price rising is lower than 0.5), the system will generate a "Sell" signal for the next day.

The second trading strategy has a limit on the number of trades by adjusting the probability threshold. Previously, when the model predicts a probability higher than 0.5 for price going up, our system will generate a "Buy" signal. And if the probability is less than 0.5, the system will generate a "Sell" signal. Now, the threshold is set to 0.3 and 0.7, so, only when the probability is higher than 0.7, the system will generate a "Buy" signal, and only when the probability is less than 0.3, the system will generate a "Sell" signal. In this way, we will only trade when we are very certain that the price will rise or fall and not trade when we are not very sure. This limit is set to reduce the number of trades to avoid triggering the wash-sale rule.

Table 5: Summary for evenly distributed weight portfolio trading without limitation.

	Evenly distributed weight							
Model	MDD	Return (%)	Counts	SR	Win rate (%)	Loss (10^6)	Profit (10^6)	Net volume (10^6)
Baseline	0.32	81.13	1	2.14			1.62	1.33
Random Forest	0.16	24.76	4893	1.40	0.52	2.33	0.50	−0.52
XGBoost	0.21	38.16	4909	1.42	0.50	3.56	0.76	−0.80
GRU	0.23	68.13	4905	2.40	0.51	4.95	1.36	−0.94
LSTM	0.28	121.56	4900	2.74	0.52	8.26	2.43	−1.49
Multi-GCN-GRU	0.12	1097.08	4903	9.21	0.56	15.52	21.94	8.12
Multi-GCN-LSTM	0.08	569.06	4894	9.79	0.56	10.19	11.38	3.43

Table 6: Summary for 20% expected return weight portfolio trading without limitation.

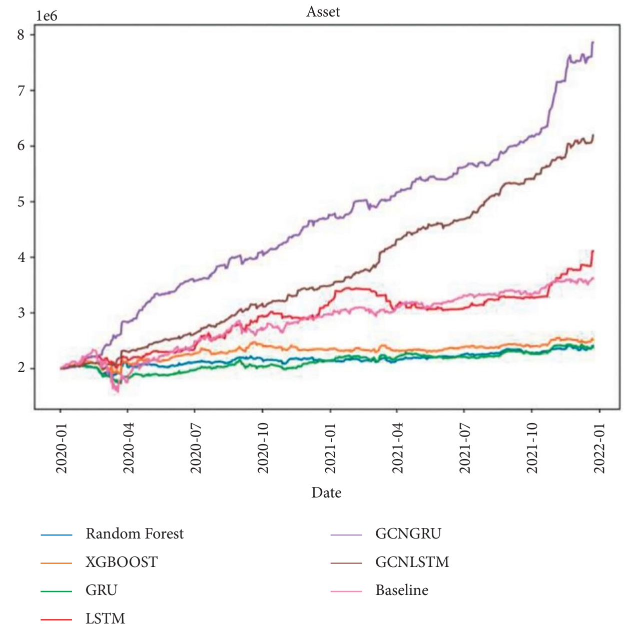
	20% Expected return weight							
Model	MDD	Return (%)	Counts	SR	Win rate (%)	Loss (10^6)	Profit (10^6)	Net volume (10^6)
Baseline	0.35	45.06	1	1.31			0.90	0.75
Random Forest	0.22	−3.81	4893	−0.26	0.52	2.09	−0.08	−0.79
XGBoost	0.14	−7.88	4909	−0.64	0.50	1.17	−0.16	−0.70
GRU	0.10	36.38	4905	2.06	0.51	1.94	0.73	−0.22
LSTM	0.26	8.65	4900	0.49	0.52	2.95	0.17	−0.95
Multi-GCN-GRU	0.05	411.82	4903	9.57	0.56	5.46	8.24	3.21
Multi-GCN-LSTM	0.08	334.82	4894	8.98	0.56	4.07	6.70	2.75

Table 7: Summary for 30% expected return weight portfolio trading without limitation.

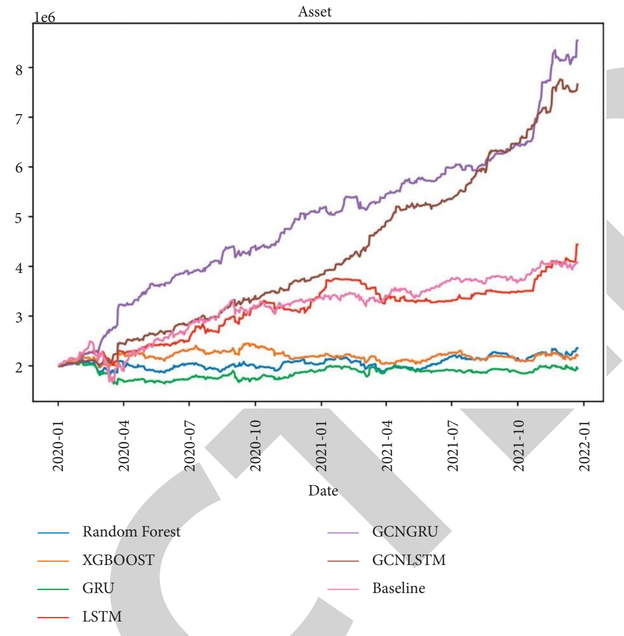
	30% Expected return weight							
Model	MDD	Return (%)	Counts	SR	Win rate (%)	Loss (10^6)	Profit (10^6)	Net volume (10^6)
Baseline	0.33	103.96	1	2.36			2.08	1.69
Random Forest	0.21	2.94	4893	0.11	0.52	2.79	0.06	−0.96
XGBoost	0.25	20.73	4909	0.64	0.50	4.12	0.42	−1.23
GRU	0.27	45.75	4905	1.29	0.51	5.83	0.91	−1.55
LSTM	0.36	131.43	4900	2.43	0.52	9.68	2.63	−1.89
Multi-GCN-GRU	0.15	1335.75	4903	8.57	0.56	19.80	26.72	9.54
Multi-GCN-LSTM	0.14	877.42	4894	8.69	0.56	13.74	17.55	6.01

Table 8: Summary for 40% expected return weight portfolio trading without limitation.

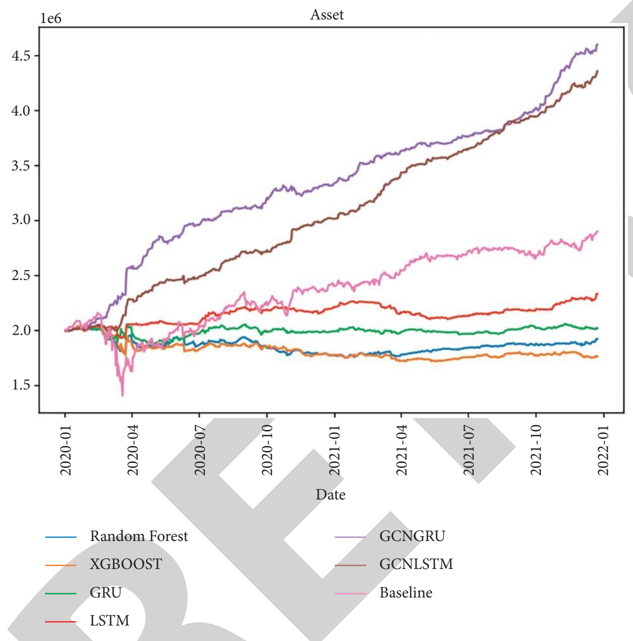
	40% Expected return weight							
Model	MDD	Return (%)	Counts	SR	Win rate (%)	Loss (10^6)	Profit (10^6)	Net volume (10^6)
Baseline	0.37	163.91	1	2.46			3.28	2.65
Random Forest	0.34	33.99	4893	0.64	0.52	3.16	0.68	−0.71
XGBoost	0.51	−27.03	4909	−0.70	0.50	2.01	−0.54	−1.05
GRU	0.54	−30.53	4905	−0.76	0.51	3.00	−0.61	−1.46
LSTM	0.43	76.92	4900	1.20	0.52	6.78	1.54	−1.5
Multi-GCN-GRU	0.27	634.06	4903	4.52	0.56	5.06	12.68	6.15
Multi-GCN-LSTM	0.21	1849.78	4894	6.82	0.56	14.68	37.00	17.91



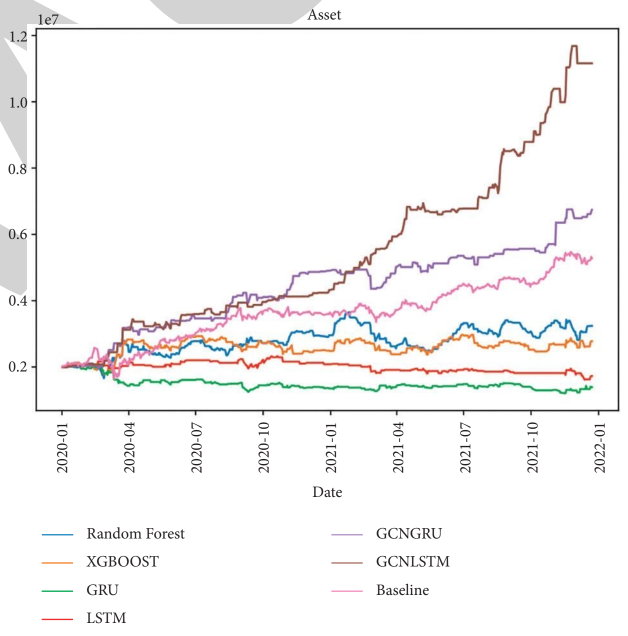
(a)



(b)



(c)



(d)

Figure 10: Asset change from 1/1/2020 to 12/31/2021 against different weights distribution for trading with adjusted probability threshold. (a) Portfolio with evenly distributed weights. (b)–(d) Portfolios with optimized weights against 20%, 30%, and 40% expected return in Markowitz portfolio, respectively.

The third trading strategy has another limit on trades based on the next day's volatility adding to the second strategy's limit. Even if our model anticipates the price to go up with a probability higher than 70% on the day after tomorrow, if the price goes up more than 0.5% on the next day, the system will not generate any "Buy" or "Sell" signal. Even though the price of the day after tomorrow is predicted to be higher than the price of today, if the price goes up too much tomorrow, the price we actually buy in tomorrow will probably be higher than the day after tomorrow, so we should not continue with the "Buy" signal. In opposite, when the price goes down too much the next day, the system will not generate a "Sell" signal. This limit will prevent some trades from happening when the volatility of the market is high. Also, the system is now possible not to make any trades on a day, instead of generating either a "Buy" or a "Sell" signal. This limit is also set to reduce the number of trades to avoid triggering the wash-sale rule.

The performances will be analyzed by their max drawdown, return, number of trades, Sharpe ratio, win rates (the

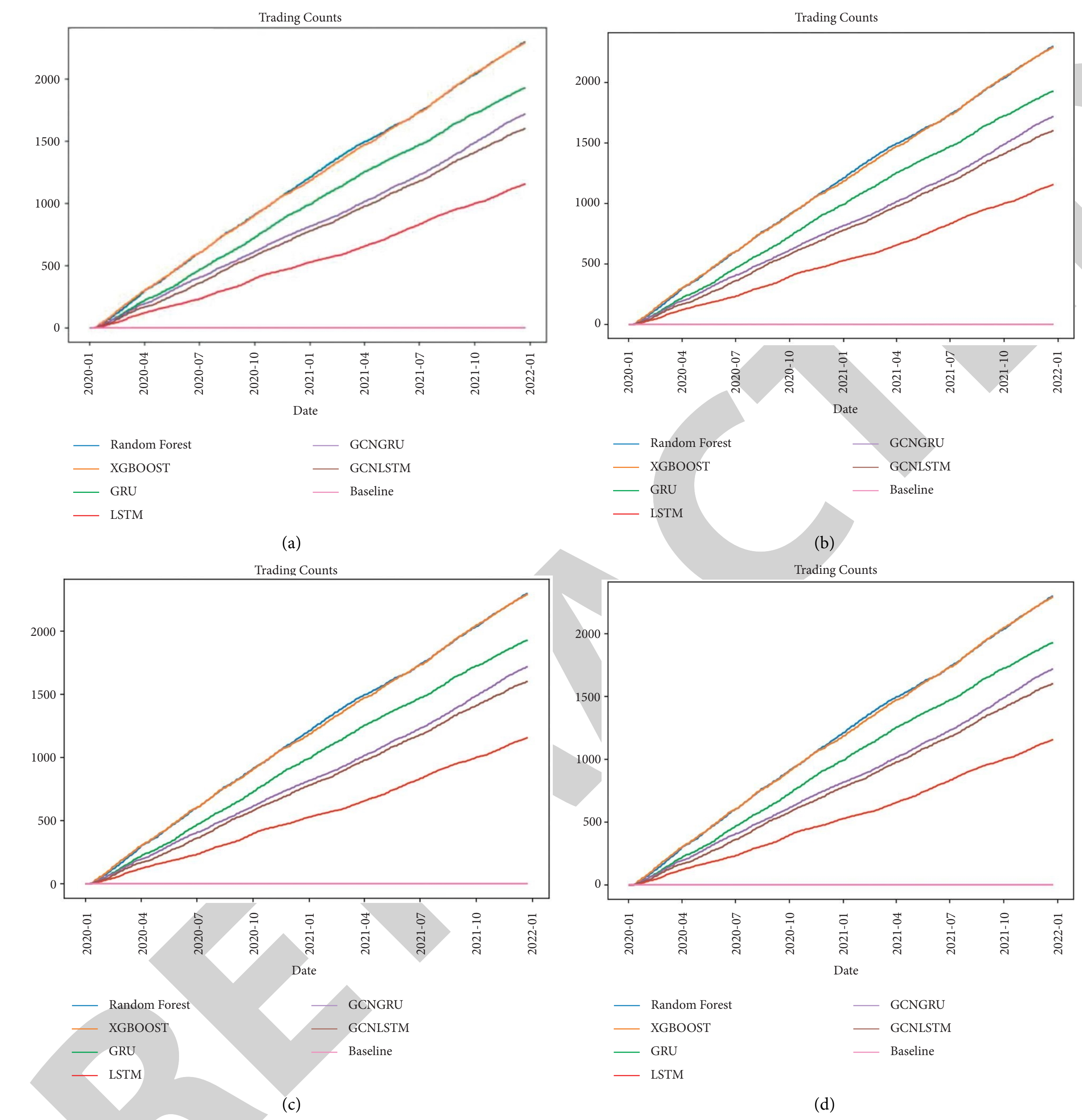


FIGURE 11: Trading counts from 1/1/2020 to 12/31/2021 against different weights distribution for trading with adjusted probability threshold. (a) Portfolio with evenly distributed weights. (b)–(d) Portfolios with optimized weights against 20%, 30%, and 40% expected return in Markowitz portfolio optimization, respectively.

number of trades that made profits/the number of trades), realized loss, final profits, and net income under four different weight distributions. The return is calculated based on profits not deducting taxes. The net income of the system is calculated by our final assets subtracted from the required tax to pay. The tax is calculated following the 2021 short-term and long-term capital gains tax rates for "single" status as shown in Tables 3 and 4. For the trades triggering wash sale, the regarding taxable asset would be the sum of profit and loss. The baseline asset that will be held for two years will follow the long-term rates, and any profits that we make from trades with signals will follow the short-term rates.

(1) Unlimited Trading. Figure 8 shows asset change from 1/1/2020 to 12/31/2021 against different weights distribution for trading without limitation. Figure 9 shows trading counts from 1/1/2020 to 12/31/2021 against different weights distribution for trading without limitation.

Tables 5 to 8 show the max drawdown, return, number of trades, Sharpe ratio, win rates (the number of trades that

Table 9: Summary for trading with adjusted probability threshold.

				Evenly distributed weight				
Model	MDD	Return (%)	Counts	SR	Win rate (%)	Loss (10^6)	Profit (10^6)	Net volume (10^6)
Baseline	0.32	81.13	1	2.14			1.62	1.33
Random Forest	0.11	20.53	2297	1.53	0.53	1.85	0.41	−4.09
XGBoost	0.13	25.94	2290	1.66	0.51	2.11	0.52	−0.42
GRU	0.16	19.31	1927	1.37	0.51	2.54	0.39	−0.66
LSTM	0.12	105.52	1155	4.75	0.51	2.72	2.11	0.36
Multi-GCN-GRU	0.04	293.21	1718	11.39	0.59	1.75	5.86	3.08
Multi-GCN-LSTM	0.06	209.66	1601	10.99	0.62	2.23	4.19	1.85

Table 10: Summary for 20% expected return weight portfolio trading without limitation.

				20% Expected return weight				
Model	MDD	Return (%)	Counts	SR	Win rate (%)	Loss (10^6)	Profit (10^6)	Net volume (10^6)
Baseline	0.35	45.06	1	1.31			0.90	0.75
Random Forest	0.14	−3.79	2297	−0.42	0.53	1.37	−0.08	−0.52
XGBoost	0.16	−11.72	2290	−1.34	0.51	1.21	−0.23	−0.56
GRU	0.09	1.05	1927	0.08	0.51	1.16	0.02	−0.38
LSTM	0.07	16.67	1155	2.23	0.51	0.65	0.33	0.00
Multi-GCN-GRU	0.03	129.97	1718	9.67	0.59	0.42	2.60	1.52
Multi-GCN-LSTM	0.03	117.89	1601	11.67	0.62	1.01	2.36	1.15

Table 11: Summary for 30% expected return weight portfolio trading without limitation.

				30% Expected return weight				
Model	MDD	Return (%)	Counts	SR	Win rate (%)	Loss (10^6)	Profit (10^6)	Net volume (10^6)
Baseline	0.33	103.96	1	2.36			2.08	1.69
Random Forest	0.14	17.83	2297	0.87	0.53	2.24	0.36	−0.57
XGBoost	0.17	10.48	2290	0.53	0.51	2.46	0.21	−0.74
GRU	0.21	−2.68	1927	−0.17	0.51	3.21	−0.05	−1.19
LSTM	0.13	121.96	1155	4.48	0.51	3.16	2.44	0.40
Multi-GCN-GRU	0.05	327.14	1718	8.89	0.59	2.33	6.54	3.30
Multi-GCN-LSTM	0.06	282.88	1601	9.33	0.62	2.94	5.66	2.51

Table 12: Summary for 40% expected return weight portfolio trading without limitation.

				40% Expected return weight				
Model	MDD	Return (%)	Counts	SR	Win rate (%)	Loss (10^6)	Profit (10^6)	Net volume (10^6)
Baseline	0.37	163.91	1	2.46			3.28	2.65
Random Forest	0.34	61.90	2297	1.33	0.53	2.56	1.24	−0.13
XGBoost	0.19	38.84	2290	0.97	0.51	2.18	0.78	−0.28
GRU	0.41	−30.52	1927	−1.24	0.51	2.26	−0.28	−1.19
LSTM	0.30	−13.78	1155	−0.70	0.51	2.34	−0.28	−1
Multi-GCN-GRU	0.12	237.22	1718	4.80	0.59	1.12	4.47	2.61
Multi-GCN-LSTM	0.09	458.13	1601	6.85	0.62	2.79	9.16	4.78

made profits/the number of trades), realized loss, final profits, and net income under four different weight distributions using the first strategy. This strategy has no special constraints on trades other than the 1.5 margin limit. The net income is the final result after deducting required taxes.

(2) Adjust Probability Threshold. Figure 10 shows asset change from 1/1/2020 to 12/31/2021 against different weights distribution for trading with adjusted probability threshold. Figure 11 shows trading counts from 1/1/2020 to 12/31/2021 against different weights distribution for trading with adjusted probability threshold.

Tables 9 to 12 show the max drawdown, return, number of trades, Sharpe ratio, win rates (the number of trades that made profits/the number of trades), realized loss, final profits, and net income under four different weight distributions using the second strategy. This strategy has a special

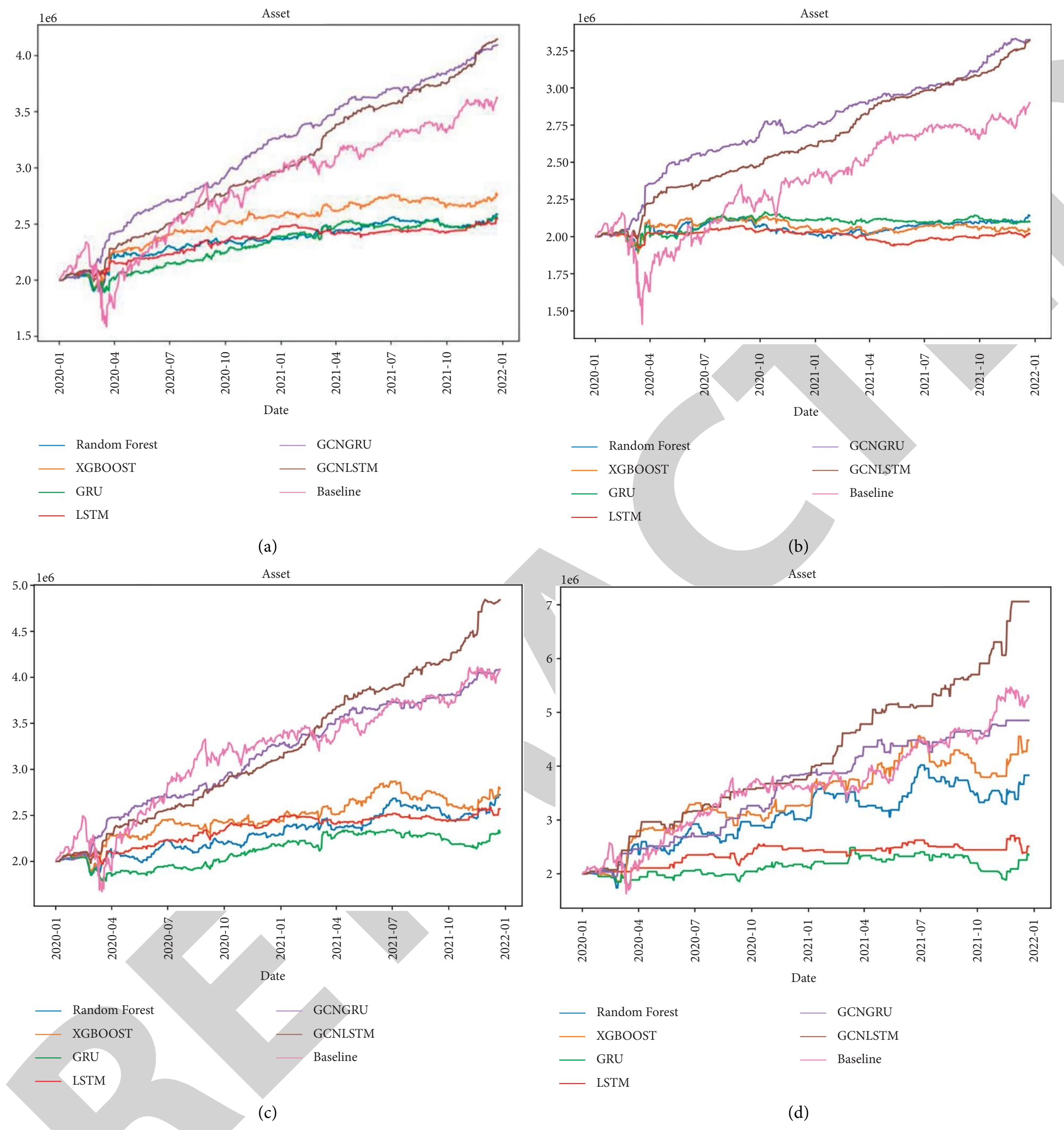


Figure 12: Asset change from 1/1/2020 to 12/31/2021 against different weights distribution for trading with adjusted probability threshold and constrained volatility. (a) Portfolio with evenly distributed weights. (b)–(d) Portfolios with optimized weights against 20%, 30%, and 40% expected return in Markowitz portfolio, respectively.

constraint on trades that trades will only take place when the system is very certain about its prediction. The net income is the final result after deducting required taxes.

(3) Constrain Volatility and Adjust Probability Threshold. Figure 12 shows asset change from 1/1/2020 to 12/31/2021 against different weights distribution for trading with adjusted probability threshold and constrained volatility. Figure 13 shows trading counts from 1/1/2020 to 12/31/2021 against different weights distribution for trading with adjusted probability threshold and constrained volatility.

Tables 13 to 16 show the max drawdown, return, number of trades, Sharpe ratio, win rates (the number of trades that made profits/the number of trades), realized loss, final profits, and net income under four different weight distributions using the third strategy. This strategy has a special

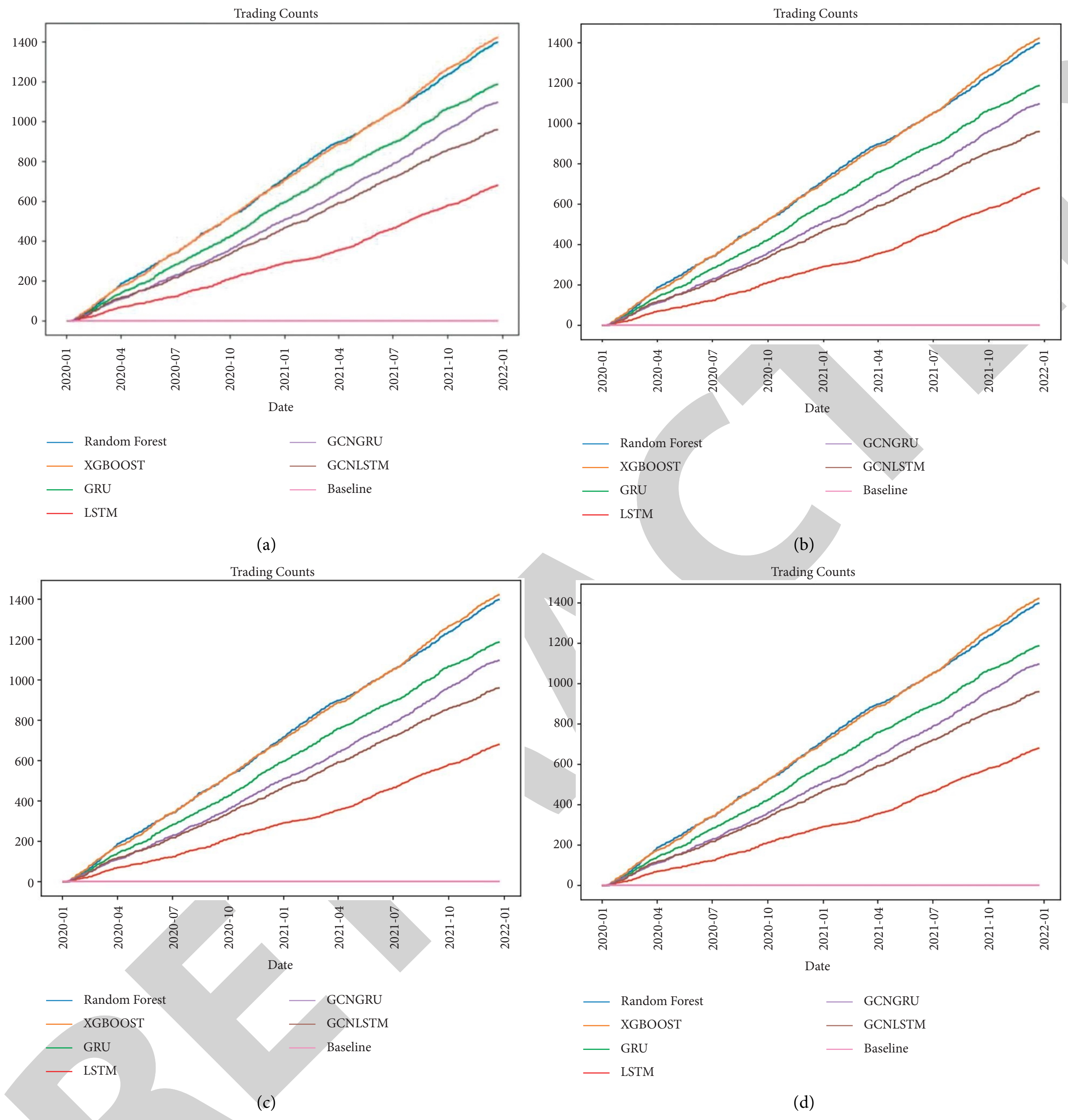


FIGURE 13: Trading counts from 1/1/2020 to 12/31/2021 against different weights distribution for trading with adjusted probability threshold and constrained volatility. (a) Portfolio with evenly distributed weights. (b)–(d) Portfolios with optimized weights against 20%, 30%, and 40% expected return in Markowitz portfolio, respectively.

constraint on trades based on the next day's movement. The net income is the final result after deducting required taxes.

5. Discussion

Tables 5 to 16 show the results for different models using different weights across three different strategies. Comparing the resulting net income of different models, the Multi-GCN-GRU and Multi-GCN-LSTM have all positive net income and are much higher than all the other models in nearly all the cases, with the best performance under 40% expected return weights in the first strategy, where Multi-GCN-LSTM resulted in net income of $17.91 million. The net income of most of the other models in most of the cases is lower than the baseline, and many of them are negative. This is not because the model itself is losing money; the profit column is mostly positive. But the net income became negative because the taxes that they need to pay exceeded the profit amount and brought the net income to negative. This shows that the effect of taxes is

Table 13: Summary for trading with adjusted probability threshold and constrained volatility.

	Evenly distributed weight							
Model	MDD	Return (%)	Counts	SR	Win rate (%)	Loss (10^6)	Profit (10^6)	Net volume (10^6)
Baseline	0.32	81.13	1	2.14			1.62	1.33
Random Forest	0.08	29.37	1398	2.49	0.33	1.09	0.59	0.00
XGBoost	0.07	37.93	1422	3.10	0.33	1.12	0.76	0.10
GRU	0.08	27.48	1187	2.55	0.33	1.38	0.55	−0.13
LSTM	0.04	27.34	680	3.65	0.29	1.01	0.55	0.01
Multi-GCN-GRU	0.02	104.69	1097	12.13	0.43	0.61	2.09	1.13
Multi-GCN-LSTM	0.05	107.29	960	10.61	0.45	0.83	2.15	1.08

Table 14: Summary for 20% expected return weight portfolio trading without limitation.

	20% Expected return weight							
Model	MDD	Return (%)	Counts	SR	Win rate (%)	Loss (10^6)	Profit (10^6)	Net volume (10^6)
Baseline	0.35	45.06	1	1.31			0.90	0.75
Random Forest	0.07	7.05	1398	0.75	0.33	0.84	0.14	−0.18
XGBoost	0.07	2.32	1422	0.2479	0.33	0.77	0.05	−0.22
GRU	0.06	5.02	1187	0.58	0.33	0.71	0.10	−0.16
LSTM	0.06	1.00	680	0.14	0.29	0.29	0.02	−0.06
Multi-GCN-GRU	0.03	66.23	1097	7.50	0.43	0.07	1.32	0.84
Multi-GCN-LSTM	0.03	65.95	960	9.09	0.45	0.49	1.32	0.69

Table 15: Summary for 30% expected return weight portfolio trading without limitation.

	30% Expected return weight							
Model	MDD	Return (%)	Counts	SR	Win rate (%)	Loss (10^6)	Profit (10^6)	Net volume (10^6)
Baseline	0.33	103.96	1	2.36			2.08	1.69
Random Forest	0.11	36.11	1398	2.23	0.33	1.50	0.72	−0.06
XGBoost	0.11	39.48	1422	2.42	0.33	1.47	0.79	−0.01
GRU	0.15	15.54	1187	1.08	0.33	1.78	0.31	−0.43
LSTM	0.06	28.50	680	2.87	0.29	1.18	0.57	−0.04
Multi-GCN-GRU	0.03	104.24	1097	9.02	0.43	0.81	2.08	1.05
Multi-GCN-LSTM	0.06	142.03	960	8.91	0.45	1.05	2.84	1.44

Table 16: Summary for 40% expected return weight portfolio trading without limitation.

	40% Expected return weight							
Model	MDD	Return (%)	Counts	SR	Win rate (%)	Loss (10^6)	Profit (10^6)	Net volume (10^6)
Baseline	0.37	163.91	1	2.46			3.28	2.65
Random Forest	0.18	91.56	1398	2.40	0.33	2.39	1.83	0.31
XGBoost	0.17	124.15	1422	3.08	0.33	1.83	2.48	0.92
GRU	0.24	17.69	1187	0.69	0.33	1.57	0.35	−0.32
LSTM	0.12	25.55	680	1.38	0.29	1.34	0.51	−0.14
Multi-GCN-GRU	0.06	142.52	1097	5.49	0.43	0.28	2.85	1.73
Multi-GCN-LSTM	0.06	252.94	960	6.70	0.45	0.66	5.06	2.98

unneglectable and can easily offset the profits gained by trades under many popular models. The Multi-GCN-GRU and Multi-GCN-LSTM models also have relatively very small max drawdowns compared to other models across different weights and different strategies, which show their advantage of consistency. Even, in the case where net income was the highest ($17.91million), the max drawdown was only 0.21, whereas other models had max drawdowns higher than 0.3 under the same weighting method and strategy.

Another interesting trend shown in the tables is that the Multi-GCN-GRU and Multi-GCN-LSTM used in the first strategy generated higher net income than the second and the third strategies, while also having higher max drawdowns. And the highest net income also occurred in the first strategy when the expected return was 40%. Although the first strategy has no constraints on trades and therefore would need to pay more taxes and suffer higher potential drawdowns, the more trades that it made generated even more profits that not only covered the loss but also added more to the net income. This can be proven by comparing the trade counts column. The first strategy has much higher trade counts for all models, taking usually more than 4000 trades, while the second and the third strategy have much fewer trade counts because they are under different constraints, taking only about half of the trades compared to the first strategy. This result shows that although adding more constraints can reduce loss from taxes, it is still better to make more trades under a good performing model because the profit will be able to cover the taxes.

A factor that was not taken into account in this trading system results calculation is the transaction costs. This factor was not considered because the asset amount being traded is large enough that the transaction cost would not have any notable impact. Therefore, having more trade counts has no penalty in this aspect. However, if the asset amount is small, the calculations would need to include this factor to be more realistic. In that case, the first strategy might be performing as strongly as it is now because it has the most trade counts, while the second and the third strategies might not be impacted as much since they have much lower trade counts.

6. Conclusion

In this paper, we proposed a GCN-based framework on stock forecasting. We compared its performance with other popular models, which include Random Forest, XGBoost, GRU, and LSTM. This framework also considered the impact of taxes to be more realistic. The test was done under three different strategies with different constraints on trades, and each strategy has different weighting methods decided by the expected return levels. The results validated the usefulness of the framework that we proposed, as it generated the highest net income in all scenarios and are much higher than the other models' and benchmark's net income. Other popular models could also generate positive profits from trades, but their profits are very close to 0, so, after deducting from the required taxes, the overall net income is either very small or negative. GCN was able to generate large enough profits that, even after deducting from taxes, still have very high net income results.

Some future directions can be done following this research. One is adding more stocks to test the model's effectiveness. The second is to try this model in different markets to test its consistency, and the third is to add more features as inputs such as technical indicators and macroeconomics features to see if the model will be improved.

Data Availability

The datasets used and/or analyzed during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest

The authors declare no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

References

[1] J. Patel, S. Shah, P. Thakkar, and K. Kotecha, "Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques," *Expert Systems with Applications*, vol. 42, no. 1, pp. 259–268, 2015.

[2] S. Shen, H. Jiang, and T. Zhang, *Stock Market Forecasting Using Machine Learning Algorithms*, pp. 1–5, Department of Electrical Engineering, Stanford University, Stanford, CA, 2012.

[3] Y. Dai and Y. Zhang, "Machine learning in stock price trend forecasting," *Erişim Tarihi*, vol. 21, 2021.

[4] J. Chen, M. Chen, and N. Ye, "Forecasting the Direction and Strength of Stock Market Movement," Technical Report, Stanford University, Stanford, 2013.

[5] A. Kumar, L. Singh, N. Bindal, and R. Goyal, "Trading via recurrent reinforcement learning," in *Proceedings of the 2003 IEEE International Conference on Computational Intelligence for Financial Engineering, 2003. Proceedings*, Hong Kong, China, March 2013.

[6] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proceedings of the International Conference on Learning Representations (ICLR'2017)*, Palais des Congrès Neptune, Toulon, Fr, April 2017.

[7] I. Chami, R. YingYing, C. Ré, and J. Leskovec, "Hyperbolic graph convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 32, pp. 4869–4880, 2019.

[8] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *Proceedings of the International Conference on Machine Learning*, pp. 6861–6871, PMLR, Long Beach, CA. USA, 2019, May.

[9] J. Chen, T. Ma, and C. Xiao, "Fastgcn: fast learning with graph convolutional networks via importance sampling," 2018, https://arxiv.org/abs/1801.10247.

[10] R. Liao, Z. Zhao, R. Urtasun, and R. S. Zemel, "Lanczosnet: multi-scale deep graph convolutional networks," 2019, https://arxiv.org/abs/1901.01484.

[11] L. Yao, C. Mao, and Y. Luo, "Graph convolutional networks for text classification," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 3301, pp. 7370–7377, 2019, July.

[12] B. Han, P. Cook, and T. Baldwin, "Geolocation prediction in social media data by finding location indicative words," in *Proceedings of the COLING*, pp. 1045–1062, Mumbai, India, 2012 December.

[13] Y. Zhang, P. Qi, and C. D. Manning, "Graph convolution over pruned dependency trees improves relation extraction," 2018, https://arxiv.org/abs/1809.10185.

[14] X. Wang, Y. Ye, and A. Gupta, "Zero-shot recognition via semantic embeddings and knowledge graphs," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6857–6866, Salt Lake City, UT, USA, June 2018.

[15] F. Scarselli, S. L. Yong, M. Gori, M. Hagenbuchner, A. C. Tsoi, and M. Maggini, "Graph neural networks for ranking web pages," in *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI'05)*, pp. 666–672, IEEE, Compiegne, France, 2005, September.

[16] Y. Wei, X. Wang, L. Nie, X. He, R. Hong, and T. S. Chua, "MMGCN: multi-modal graph convolution network for

personalized recommendation of micro-video," in *Proceedings of the 27th ACM International Conference on Multimedia*, pp. 1437–1445, Nice, France, 2019, October.
[17] K. Guo, Y. Hu, Z. Qian et al., "Optimized graph convolution recurrent neural network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 2, pp. 1138–1149, 2021.
[18] W. Jiang, "Applications of deep learning in stock market prediction: recent progress," *Expert Systems with Applications*, vol. 184, p. 115537, Article ID 115537, 2021.
[19] I. Letteri, G. Della Penna, G. De Gasperis, and A. Dyoub, "A stock trading system for a medium volatile asset using multi layer perceptron," 2022, https://arxiv.org/abs/2201.12286.
[20] J. Oh, "Development of a stock trading system based on a neural network using highly volatile stock price patterns," *PeerJ Computer Science*, vol. 8, p. e915, 2022.
[21] L. Chen, L. Sun, C. M. Chen, M. E. Wu, and J. M. T. Wu, "Stock trading system based on machine learning and kelly criterion in internet of things," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 7632052, 9 pages, 2021.
[22] M. A. Dempster and V. Leemans, "An automated FX trading system using adaptive reinforcement learning," *Expert Systems with Applications*, vol. 30, no. 3, pp. 543–552, 2006.
[23] S. Almahdi and S. Y. Yang, "An adaptive portfolio trading system: a risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown," *Expert Systems with Applications*, vol. 87, pp. 267–279, 2017.
[24] J. Moody, L. Wu, Y. Liao, and M. Saffell, "Performance functions and reinforcement learning for trading systems and portfolios," *Journal of Forecasting*, vol. 17, no. 5-6, pp. 441–470, 1998.