Enhancing Stock Trend Prediction Models by Mining Relational Graphs of Stock Prices

Hung-Yang Li
Department of Computer Science
National Chiao Tung University
Taiwan, R.O.C.
chc71340@gmail.com

Vincent S. Tseng
Department of Computer Science
National Chiao Tung University
Taiwan, R.O.C.
vtseng@cs.nctu.edu.tw
*Corresponding author

Philip S. Yu
Department of Computer Science
University of Illinois at Chicago
USA
psyu@uic.edu

***Abstract*—Stock trend prediction has been a subject that attracts lots of attentions from a diverse range of fields recently. Despite the advance by the cooperation with artificial intelligence and finance domains, a large number of works are still limited to the use of technical indicators to capture the principles of a stock price movement while few consider both historical patterns and the relations of its correlated stock. In this work, we propose a novel framework named RGStocknet (Relational Graph Stock Enhancing Network) that can boost performance on an arbitrary time series prediction backbone model. Our approach automatically extracts the relational graph into which the graph embedding model can be easily integrated. Treated as an additional input feature, company embedding from the graph embedding model aims to improve performance without the need for external resources of the knowledge graph. The experiment results show that the three benchmark baseline can benefit from our proposed RGStocknet module in relative performance gain on the S&P500 dataset with 2.97%, 2.48%, and 7.03% on profit-score and with 25.50%, 17.53%, and 12.75% on accuracy respectively. Applied to a real-world trading simulation environment, our approach also outperformed the backbone model and doubled the average return on ResNet over the buy and hold (BH) strategy from 4.42% to 7.38%. Visualization of the generated relational graph and company embedding also shows that the proposed method can capture the hidden dynamics of other correlated stocks and learn representation across the whole stock market. Moreover, the proposed method was shown to carry the potential to incorporate relations with external resources to achieve higher performance further.**

***Index Terms*—Stock trend prediction, Time series, Knowledge graph embedding model, Dynamic time warping**

I. Introduction

In the financial market, price movements of stocks, bonds, and other crucial indices depend not only on their historical records but also on other economic factors [1] [2]. As the world getting more and more interconnected through globalization, more worldwide connection, like cash flow or supply chain, has been thoroughly established. Therefore, this phenomenon results in a higher correlation in financial markets. For example, the financial crisis in 2008 is not confined to emerging markets but spread like contagion instead [3]. The United States or other mature financial markets transmit and receive contagion due to the enormous cross-market linkages.

Stocks, as one of the major investing tools, have lots of relations with each other as the world becomes more interactive. For instance, there is a well-known relationship called the lead-lag relationship in the stock market, where the price movement of a stock is highly correlated to other stocks. It reveals how fast one stock reflects new information relative to the other stock, and how well the two stocks are linked [4]. For instance, the downstream company will react faster than the upstream company in the supply-chain relationship. On stock trend prediction, the lead-lag relationship that models the asynchronous correlation can somehow provide strong evidence to the latter stock movement. Therefore, relations among financial markets may play an essential role in future trend prediction.

In the last decades, there has been a surge of interest in the use of artificial intelligence for accurate stock trend prediction. Most research regarding how to extract an effective historical pattern of the predicted stock itself [5] [6] focuses on technical analysis. Some works may consider related news or other fundamental information as to their strategy [7] [8]. Although relations such as the lead-lag relationship may contain valuable clues for stock prediction, few of the recent works consider the inter-relation except from the external resource in the stock market [9] [10] [11]. Despite these existing works being effective and producing better performance over baselines, there are still three aspects that may be the problems. First, the relational graph from external resources may be static, which means the relational graph will not be evolved without a manual update. It may cause an outdated relationship still being used during the future prediction phase. Second, it may be full of noise since external resources are mostly created by humans. The last one is the non-general property of these models, which makes it difficult to incorporate other available relational graph data.

To provide a good solution to the above problems, we propose a general framework RGStocknet that can boost performance on arbitrary time series prediction models with relational graph extracting from raw time series data. To model lead-lag relationships, we treat dynamic time warping (DTW) as an extraction method to calculate distance from original stock prices and other delayed stock prices and further utilize extracted knowledge graph to train a company embedding model. As an additional feature to the final prediction layer,

978-1-6654-0483-9/20/$31.00 ©2020 IEEE
DOI 10.1109/ICPAI51961.2020.00028

company embedding can be regarded as lead-lag representation in high dimensional vectors. Experiments are conducted to demonstrate the enhancing performance on future trend prediction and simulating trading profit over the vanilla baseline. To the best of our knowledge, this is the first work applying graph embedding model constructing from relations extracted from raw time series data for the stock trend prediction field.

II. Related Work

A. Stock Movement Prediction

Owing to the potentially large profit, stock movement prediction has been recognized as an interesting problem in the research area. According to the core concept of predicting future trends in the stock market, the related works can be further divided into two categories, which are technical analysis and news-oriented analysis.

Technical analysis takes historical prices of the stock as a feature to forecast its future movement. Tao Lin, Tian Guo, and Karl Aberer [12] introduce Trenet to extract features from raw time series and construct the dependency across multiple time steps. Zhang et al. [5] inspired by Discrete Fourier Transform, they propose a modified version of long-short term recurrent neural network (LSTM) which called State Frequency Memory (SFM) recurrent network to capture the multi-frequency trading patterns from past price data to make a future prediction in both short and long term. All of them are also evidence to prove that deep learning methods are beneficial to mine effective patterns and capture nonlinearity from historical prices of stock.

The news-oriented analysis takes historical prices as features, but they also extract feature from natural language processing (NLP) method as additional input, simulating the social impact on the stock market. For instance, Hu et al. [7] imitate the learning process of human beings facing such chaotic online news to predict future trends based on recent related news. Yumo Xu and Shay B. Cohen [8] also treat the stock market as a stochastic environment, jointly exploiting text and price signals for the prediction task.

The above studies show that related features of the stock itself are essential in stock movement prediction. However, few consider inter-market or inter-company relations in the current financial market environment. Liu et al. [10] comprehensively use the news sentiment as the correlation between stock companies, accompanied by stock price data, to predict the future trend. Utilizing the knowledge graph and its embedding model, they find that their method can improve significantly over baseline. Matsunaga et al. [9] utilize the knowledge graph as additional input, proposing a graph convolutional neural network combining prices and knowledge graph data. Their result also proves that knowledge graph and historical prices in financial markets holds a strong promise in creating a stable and practical stock market predictor. Deng et al. [11] extract knowledge graph from the online news of each company. Treat all entities and edges as TransE input to train a knowledge-driven event embedding model for each company. In combination with TCN and TransE model, their framework can significantly outperform deep models and be explainable over prediction results. Li et al. [13] assumes the stocks in the market are not dependent, modeling the connection among the whole market with correlation matrix. Further result also shows that their structure enable news-oriented model not only directly associated with news but also the whole market.

Although they use the knowledge graph as additional input, extracting from external resources makes it static and unreliable. It may cause model performance to fluctuate according to the quality of external resources. As compared to our method, we propose a general plugin module utilizing a graph extracted from raw data to construct a graph embedding model that can improve the accuracy of time series backbone. Our proposed framework can be easily adapted to arbitrary deep learning models of stock movement prediction.

B. Lead-lag Relationships

The lead-lag relationship has been recognized as a critical stylized fact no matter on high frequency as intra-day prices or low frequency as inter-day prices. Specifically, some stocks tend to follow the other stocks on its price movement at later times. This phenomenon can be aroused for some reasons such as supply-chain, information diffusion, policy change, event-driven trading, and asynchronous trading [14] [15] [16] [17]. It also has considerable evidence to point out that the lead-lag effect should be more prevalent from the same industry, while small firms will follow big firms due to the related post-announcement drift [14].

As our approach is to extract the relational graph automatically from raw time series, we use dynamic time warping as a distance function to generate graphs from individual stock and the other delayed stocks movement. Try to capture the lead-lag effect to help forecast from the whole stock market.

C. Graph Embedding

The graph has a natural connection with a wide diversity of real-world scenarios, user relationships in social media, transactions and users in shopping websites or entities in financial markets. Graph embedding provides an effective way to convert each element of the knowledge graph into a lower dimension, while the graph structure is preserved.

Trans series models have been widely researched and applied to learning knowledge graph representation recently [18] [19] [20]. This type of model represents relationships by interpreting them as a translation on the entities' low-dimensional embedding. Compared with the traditional method, the Trans series is easy to train but not liable to overfitting. Boders et al. [18] proposed TransE model. It makes the sum of the head vector and relation vector as close as possible with the tail vector, which treats triplet (head, relation, tail) as a transition from head to tail. That is $h + r \approx t$. Wang et al. [19] took TransE as a reference, they proposed TransH which models a relation as a hyperplane. TransE [18] and TransH [19] model both assume that entity and relation are vectors in the semantic space so that the similar entities will close to each other. To address this issue, Lin et al. [20] let

entities and relations in two distinct spaces and perform the translation in the corresponding relation space. The relation-specific projection can make the head/tail entities that actually hold the relation close with each other, and also get far away from those that do not hold the relation.

III. Proposed Method

In this section, we provide the problem definition of stock trend prediction. Then the following part will be the detail of our proposed RGStockNet module.

A. Problem Definition

We use bold-face capital letters (e.g., $\mathbf{X}$) for matrices and bold-face lower-case letters (e.g., $\mathbf{x}$) for vectors. Moreover, normal lower case letter (e.g., x) represent scalars and Greek letters (e.g., Θ) are used to represent hyper-parameters.

We formulate stock trend prediction as a time series classification problem, learning a mapping function $\hat{y}^s_{t+1} = f(\boldsymbol{X}^s_t, \Theta)$ from a series of sequential features in the lag of past T time step to the subsequent trend of next time step $t+1$. $\boldsymbol{X}^s_t = [x^s_{t-T+1}, x^s_{t-T+2}, ..., x^s_t] \in \mathbb{R}^{T\times C}$ is the input window of stock s in time t as matrix form, which contains the sequential features from time $t-T+1$ to t, where C is the dimension of input features. Θ is the parameters of mapping function f and $\hat{y}^s_{t+1}$ is the predicting trend at next time step $t+1$. Trend calculation is defined by the difference percentage of closing price at time t and closing price at time $t+1$. And y^s_{t+1} is the ground truth of future trend at time t+1.

By accessing a long history of each stock, we can construct many training examples for mapping function by moving the lag along with the whole history. We then can formulate our problem as follows:

Input: A set of training sequences: $\{(\boldsymbol{X}^s_t), y^s_{t+1}\}$.

Output: A prediction mapping function $f(\boldsymbol{X}^s_t, \Theta)$ that predicts future trend movement in the following time step.

B. Proposed Framework

Figure 1 shows the proposed RGStockNet framework that consists of two stages. In stage 1, we extract relation triplets from raw time series data of each pair of stocks as knowledge graphs, and then train a graph embedding model as a pre-trained knowledge embedding (KE) model for stage 2. In stage 2, we extract a time period with lag time T from an arbitrary company. Company id will pass through the pre-trained model in stage 1, generating company embedding vector, while time period data will pass through the time series encoder to generate context vector. These two vectors will be concatenated to make trend prediction through the last fully-connected layer.

1) Relational Graph Generation: The relational graph generation block extract relation triplets from raw time series of each pair of stocks. By applying inter-times-series computing method to normalized time series of each pair of stocks, adjacency matrix of whole stock market is generated. Then triplets can be generated by the decision threshold of graph edge τ.

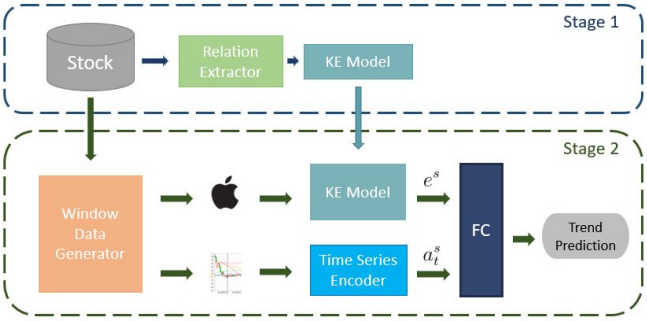

Fig. 1. Framework of proposed RGStockNet module.

Time Series Normalization. Since our goal is to capture relation from each pair of raw time series data, in order to make time-series comparable to each other, we first transform the time series of every stock into the difference of two consecutive time steps along with the whole history.

$$X^{i_{norm}} = \{x_0^{i_{norm}}, x_1^{i_{norm}}, ..., x_T^{i_{norm}}\}$$
$$x_t^{i_{norm}} = \frac{x_t^{i_{close}}}{x_{t-1}^{i_{close}}} - 1, t \neq 0$$
$$x_0^{i_{norm}} = 0$$

Adjacency Matrix Generation. There are several inter-time-series computing methods $\mathcal{G}$ transforming each pair of time series into a numeric value of relation, which is adaptable to raw time series data. The simplest one is dynamic time warping (DTW), which measures the similarity between two time series. The general idea of these methods is to extract the strength of certain relations from each pair of raw time series. As such, we use this idea to extract relation from raw time series as,

$$\mathcal{M}^r_{ij} = \mathcal{G}(X^{i_{norm}}, X^{j_{norm}}) \tag{1}$$

where $\mathcal{M} \in \mathbb{R}^{S\times S}$ is the adjacency matrix of all available raw time series; and $\mathcal{G}$ is the inter-time-series computing method that extracts relation between each two raw time series. $X^i \in \mathbb{R}^{T\times C}$ is the time series data of stock index i.

However, our intuition is to find lead-lag relationship from raw time series data. Our extraction process can be as,

$$\mathcal{M}^r_{ij} = \mathcal{G}(X^{i_{norm}}_{0:T_w}, X^{j_{norm}}_{d:T_w+d}) \tag{2}$$

where T_w is window length that we take into consideration in extraction process and d is hyper-parameter that decide how long lead-lag relationship are going to extract.

Graph Generation. Given several adjacency matrices concerning a different combination of relation extraction method and lag time step, threshold τ^r are lower bound to decide specific edge E^r_{ij} to relation r in the relational graph should be existed or not. Edge E^r_{ij} will be expressed as triplets (head, relation, tail).

$$E = \{(i, r, j) | \frac{1}{\mathcal{M}_{ij}^r} > \tau^r\} \tag{3}$$

2) Learning Embedding of Relational Graph: To utilize relational graphs generated from the previous section, we conduct a graph embedding model to convert each element of the knowledge graph into a lower dimensional representation, while graph structure is preserved.

We use TransR [20] as graph embedding model. For each relation triplet (h, r, t), we'll have the following definition as

$$h_r = hM_r,\ t_r = tM_r \tag{4}$$

where $M_r \in \mathbb{R}^{k \times d}$ is the projection matrix mapping vectors from entity space to relational space, $h \in \mathbb{R}^k$ and $t \in \mathbb{R}^k$ are the embedding of head and tail respectively in entity space.

The score function is defined as L2 norm with the summation of projected embedding of head entity h_r and relation r embedding r_r, and projected embedding of tail entity t_r.

$$f_r(h, t) = ||h_r + r_r - t_r||_2^2 \tag{5}$$

To extend Equation 5 into objective function, we have the definition as follows,

$$\sum_{(h,r,t) \in E} \sum_{(h',r,t') \in E'} max(0, f_r(h, t) + \zeta - f_r(h', t')) \tag{6}$$

where E is set of correct triplets from previous section, E' is set of incorrect triplets by replacing entities of correct triplets and ζ is the margin value that tolerate the maximum difference between scoring function value of correct triplet and incorrect triplet.

Noted that the embedding of specified stock s in entity space will be treated as e^s in the following fusion phase.

3) Time Series Encoder: For the stock trend prediction problem, quantitative features such as daily price-related data are also important. The general idea of this block is to project temporal sequence data into hidden representation a^s.

$$a_t^s = \varphi(X_{t-T+1:t}^s) \tag{7}$$

where φ can be any model dealing with time series problems. T is the lag period, and X^s is time series data of stock s.

4) Fusion: Finally we consider time sequence embedding and entity embedding of its corresponding company together by concatenating them together into latent representation of stock s.

$$e_t^s = [e^s, a_t^s] \tag{8}$$

where e^s is the embedding of predicted company which is generated from pretrained TransR model and a_t^s is hidden representation vector from time series encoder. With e_t^s that consider both quantitative data and relation among financial markets, we use fully connected layer with softmax to calculate probability distribution of prediction label.

$$\hat{y_t^s} = W_p^T e_t^s + b_p \tag{9}$$

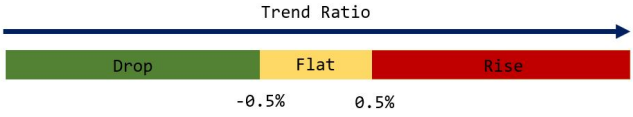

Fig. 2. Demonstration of split points in the dataset.

IV. Experiments and Evaluations

A. Dataset

We evaluate our model on U.S. stock market data from the S&P500 stock dataset at the Kaggle platform [21]. It contains historical data from Aug-13-2012 to Aug-11-2017 of the 503 constituent company. We first drop the company with an empty record on the trading day since it may be deleted from S&P500 or be added after Aug-13-2012. Finally, we will get 446 constituent companies and consist of 561,068 day-trading records. Each record contains the following six features: date, open, high, low, close, and trading volume.

Observation has revealed that properties may be different among different companies, bias value may locate between small values for stable stocks; meanwhile, the high variance may occur in cyclic stocks. Each stock also has different statistic properties, such as maximum and minimum values. To diminish their difference, we normalize each stock using z-normalization [22] by the mean and standard deviation of each corresponded company.

$$\hat{X_t^s} = \frac{X_{t-T+1:t}^s - \mu(X^s)}{\sigma(X^s)} \tag{10}$$

where $\mu(X^s)$ and $\sigma(X^s)$ are mean and standard deviation of stock s respectively.

We chronologically separate the dataset into three periods for training (2012-08-13 to 2016-02-16), validation (2016-02-17 to 2016-08-12) for tuning model, and testing (2016-08-15 to 2017-08-11) for performance measurement.

We use the difference percentage of closing price between two consecutive days as the labeling criteria. Unlike most related works, define parameters to split each class into balance. Our split points are described in the following Figure 2. To make split points reasonable, we first take the transaction fee and tax into consideration. While 0.5% is the most common transaction fee while trading in the stock market, which means $< 0.5\%$ trend is not profitable in two consecutive days. Therefore, we define $\pm 0.5\%$ as a flat class, where negative return may occur in consecutive day trading (buy at close price at time t and sell out at next day close price). Movement percentage over 0.5% will be labeled as rise class, and below -0.5% will be marked as drop class.

B. Baseline

Our goal in this work is to test whether the RGStockNet module can enhance the performance of the baseline model or not. We will compare the baseline models with the RGStockNet module using the following baselines.

- **LSTM** is a long-short term memory model that belongs to the case of recurrent neural network [23]. It contains one input layer, one LSTM layer with hidden size 128, and a fully-connected decision layer [24].
- **ALSTM** is the Attentive LSTM [25] which is an encoder-decoder structure. By assigning attention weighting of each time step in the encoder, the decoder decides part of the source data sequence to pay attention to.
- **ResNet** [26] uses residual block and convolutional layer to construct classification model. It is considered a strong baseline for time series classification.

C. Evaluation Metrics

We evaluate prediction performance with profit-score and accuracy for our 3-class classification task, accompanied by precision, recall, and f1-score for the supplement. Generally, our ultimate goal is to make a profit in real life. We will further simulate a trivial trading strategy to evaluate our proposed model with total return and Sharpe ratio (SR) for the entire dataset, a case study with total return, profitable percentage (PP), and hold in bullish percentage (HBP). The following part will be details of these metrics:

1) Classification:

- **Profit-Score** Profit-Score is a measurement of how profitable the model is. Since the flat class is defined as a trend of consecutive days difference lower than 0.5%, we will probably earn less than the transaction fee or tax even if we predict right on flat class. Therefore, we considered notable rise and drop are key to making a profit and taking the average of the macro f1-score of the rise and dropping classes as a new metric, profit-score.

$$Profit\text{–}Score = \frac{F1\text{–}Score(Rise) + F1\text{–}Score(Drop)}{2} \tag{11}$$

- **Accuracy**

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

- **Precision**

$$Precision = \frac{TP}{TP + FP} \tag{13}$$

- **Recall**

$$Recall = \frac{TP}{TP + FN} \tag{14}$$

- **F1-Score**

$$F1\text{–}Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{15}$$

2) Trading Simulation: Entire Dataset:

- **Return** The return is referred to the average return from our portfolio, which invests on all constituent companies equally.

$$Return = \frac{\sum_s \frac{F_t^s - F_0^s}{F_0^s}}{N} \tag{16}$$

where N is the number of constituent companies in the dataset, F_t^s is the balance after several transactions at the end and F_0^s is the initial investment fund of stock s at the beginning of the trading period respectively.

- **Sharpe Ratio** The Sharpe ratio evaluates the investment return relative to the amount of risk taken. The higher the ratio, the stabler our investment return is.

$$S = \frac{R_x - R_f}{\sigma(R_x)} \tag{17}$$

where R_x is the average return rate, R_f is the best available rate of return of a risk-free security, $\sigma(R_x)$ is the standard deviation of R_x.

3) Trading Simulation: Case Study:

- **Return** The return is the cumulative profit divided by the original investing money.

$$Return = \frac{F_t - F_0}{F_0} \tag{18}$$

where F_t is the balance after several transactions at the end and F_0 is the initial investment fund at the beginning of the trading period respectively.

- **Percent Profitable** The percent profitable (PP) metric can also be known as the probability of winning. For example, if we make four positive return transactions out of 10 transactions, 40% would be our percent profitable.

$$PP = \frac{WinningTrades}{TotalTrades} * 100 \tag{19}$$

- **Hold in Bullish Percentage** The hold in bullish percentage (HBP) metric measures how well our strategy can cover the bullish trend of trading days.

$$HBP = \frac{\left|\left\{t \mid (X_{c_{t+1}} - X_{c_t}) - 1 \geq 0 \cap t \text{ has stock}\right\}\right|}{\left|\left\{t \mid (X_{c_{t+1}} - X_{c_t}) - 1 \geq 0\right\}\right|} \tag{20}$$

where X_{c_t} is the closing price at time t.

D. Parameter Settings

For relational extractor, we extract adjacency matrices from stock companies by dynamic time warping with Manhattan distance. The window length of capturing the relational graph is set to 60, across three months of transaction days. The decision threshold to establish a knowledge graph τ is 3, where relation edges lower than the decision threshold will be dropped before entering the knowledge graph embedding training process.

For the knowledge graph embedding model, we implement the TransR model with the OpenKE toolkit [27], which is a well-known open-source framework for knowledge embedding. We set batch size and margin value to 128 and 4.0 respectively. The hidden size of entities and relations are 128. Stochastic gradient descend (SGD) [28] is used with learning rate 1e-3 and weight decay 5e-4 to optimize our model. We will run 200 epochs to minimize the validation loss as a criterion to select our final knowledge graph embedding model.

For the time series encoder and pre-trained knowledge graph embedding model at stage 2, there are two options on

TABLE I
COMPARISON WITH VANILLA BACKBONE MODEL AND MODEL WITH PROPOSED RGSTOCKNET MODULE.

	Profit-Score	Accuracy	Precision	Recall	F1-Score
LSTM	31.95%	30.82%	34.48%	33.02%	29.78%
RG-LSTM	**32.90%**	38.68%	38.32%	38.04%	37.79%
RG-LSTM-f	32.53%	**39.39%**	**38.44%**	**38.31%**	**38.16%**
ALSTM	32.25%	31.15%	34.70%	33.22%	30.31%
RG-ALSTM	**33.05%**	36.61%	37.67%	36.86%	36.23%
RG-ALSTM-f	31.62%	**38.81%**	**38.41%**	**38.01%**	**37.36%**
ResNet	33.84%	31.54%	34.24%	33.44%	31.08%
RG-ResNet	**36.22%**	**35.56%**	**38.14%**	**36.69%**	**35.43%**
RG-ResNet-f	35.18%	33.29%	37.44%	35.35%	32.71%

TABLE II
TRADING PERFORMANCE OF ENTIRE DATASET.

Model	Return [%]	SP	PS
B&H	4.42	0.156	-
LSTM	4.17	0.241	32.90%
RG-LSTM	**5.80**	**0.352**	32.53%
ALSTM	4.81	0.324	32.25%
RG-ALSTM	**4.91**	**0.335**	33.05%
ResNet	4.41	0.283	33.84%
RG-ResNet	**7.38**	**0.394**	36.22%

whether to freeze the gradient of the pre-trained knowledge graph embedding model or not. In the freeze knowledge graph embedding model, which denotes as -f in the following result, we will freeze the weight of the KE model. Otherwise, the non-freeze model will be further fine-tuned while training as the trend prediction phase goes. The model will be optimized by Adam optimizer [29] with learning rate 1e-3 and weight decay 5e-4 and trained 100 epochs with batch size 1024. For each iteration, we randomly sample mini-batches from training sets. Maximizing profit-score, which is our proposed metric to maximize profit return in the investment scenario, will be the objective in this stage.

V. EXPERIMENTS RESULTS

A. Performance Comparison

Our research aims to propose a plugin RGStockNet module that is flexible enough to adapt to an arbitrary deep model, enhancing performance on the vanilla baseline.

As we can see in Table I, the vanilla baseline with the RGStocknet module mostly outperforms the vanilla baseline on profit-score and accuracy, no matter the pre-trained model weight is freeze or non-freeze. Compared with the pre-trained knowledge embedding model under freeze and non-freeze settings, the RGStocknet module with non-freeze training may have higher accuracy over freeze settings and otherwise lower profit-score. There exists a trade-off between accuracy and profit-score due to the subtle imbalance distribution of our dataset. Further fine-tuned knowledge embedding model in stage 2 will make the prediction model tend to flat class, which is the majority class in our dataset. Since profit-score is a more significant metric than the accuracy of the stock movement prediction problem, RGStocknet with an unfreeze setting will be the better choice as future investment tools.

B. Trading Simulation

Although the classification result of RGStockNet can outperform the vanilla baseline, whether our framework can be applied in a real-life investment environment is another concern. Therefore, we make a trading simulation as backtesting for the entire testing dataset.

The trading strategy is followed below. If the model predicts **Rise** when there is no stock in hand, the buying signal will be triggered at the closing price on that day. **Flat** class is considered as nonsense. Once the turning point occurred, which is **Drop** class that means the closing price of the next day may fall below the close price today, we will leave current trade as profit-taking. It is noted that most well-known online brokers are commission-free for stock trading, such as TD Ameritrade, FirstTrade, etc. The commission will be ignored in the following trading simulation phase. Moreover, freeze setting will be used to our prediction framework.

1) Entire Dataset: However, to compare the RG-based model with the vanilla baseline, we also introduce buy and hold (B&H) strategy as the baseline, which buys at the first time point and sells out at the last trading day of the testing period. The simulation result is shown in Table II. All vanilla baseline with RGStocknet module beat B&H and outperform their vanilla baseline for cumulative return by 39.1%, 2.1%, 67.3%, and Sharpe ratio (SP) by 46.1%, 3.4%, 39.2% respectively. Noted that profit-score for 3-class stock classification has a positive correlation to the final return percentage and Sharpe ratio. This indicates that Rise and Drop class are more critical than Flat class prediction in the real-world investment scenario.

2) Case Study: In this section, we will inspect on simulation performance of individual stocks, including Apartment Investment and Management Co.(AIV), Berkshire Hathaway Inc. Class B (BRK.B), Costco Wholesale Corporation (COST), HP Inc. (HPQ), and Verizon Communications Inc. (VZ). Table III shows the trading performance of the companies mentioned before. Figure 3 shows the commutative return percentage plot thorough the testing period of Costco, while closing difference on two consecutive days and the prediction result is presented. Prediction results of each day are denoted as red, yellow, and green color, which are Rise, Flat, and Drop, respectively.

Our model can mostly achieve better performance than the baseline on all companies. We also observe the following phenomena: i) Recurrent neural networks are eager to make a sequence of the same prediction in a row, while the RGStocknet module can ease this situation and make a better profit than the vanilla baseline. ii) The RGStocknet module is also possible to be useless.

In addition to typical cases, we also consider the cyclic and non-cyclic company in our case study. The terms cyclical and non-cyclical refer to how closely correlated a company's price is to the fluctuations of the economy. We extract company with maximum and minimum standard deviation on consecutive closing difference during the testing period, HPQ and BRK.B are extracted concerning non-stable and stable stock

Fig. 3. Trading Simulation of COST.

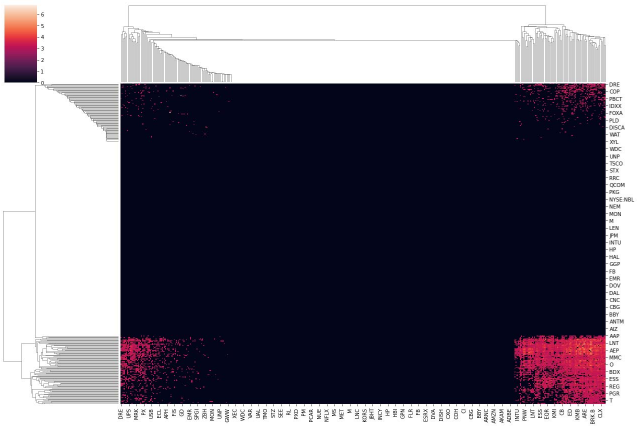

Fig. 4. Clustered heat map of 3-day delay DTW relation.

companies. HPQ as non-stable stock, RG-LSTM, and RG-ALSTM are struggling to make correct predictions but still try to make transactions as LSTM and ALSTM act like buy and hold strategy, which holds stock in hand until the end. However, RG-ResNet can provide better prediction results and outperform vanilla ResNet on the final return. Our model can make a good performance in a stable company like BRK.B, which is a well-known holding company investing in a diverse field of industries. The case can also prove whether the stock is stable or not; our module can also effectively enhance model performance on stock movement prediction.

C. DTW Relationship

In studies on extracted knowledge graph from raw time series data, we further visualize the adjacency matrix by clustered heat map. The order of the rows and columns is determined by performing hierarchical cluster analyses of the rows and columns, respectively.

Each row represents the dynamic time warping value of a particular company concerning n-delay companies. In contrast, each column represents the dynamic time warping value of a particular n-delay company concerning companies with no delay. Cells with high scores represent a strong lead-lag relationship between each other. For instance, as shown in Figure 4, Alliant Energy Corporation (LNT) and American Electric Power Company Inc. (AEP) are all major investor-owned electric utility in the United States of America. Therefore, they may have the same lead-lag relationship with other companies.

D. Learned Stock Company Representations

We operate PCA [30] visualization experiment to assess company embedding from high dimensional to low dimensional space. After mapping each representation to a two-dimensional space, we further color each node with its belonging sectors as Figure 5. Figure 6 plots the mean average embedding of each sector. It shows that Consumer Staples and Consumer Discretionary, which are both non-cyclic sectors of stock, are away from the others, which are cyclic sectors. Moreover, the utilities sector, referring to a category of companies that provide basic amenities such as water, electricity, and natural gas, is very closed to the energy sector producing and supplying energy. That is, our proposed method can learn a meaningful representation of constituent companies across the whole stock market.

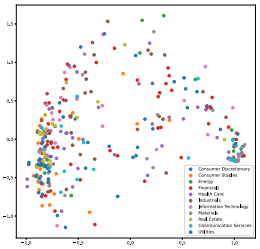

Fig. 5. PCA visualization of company embedding.

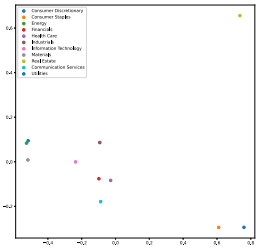

Fig. 6. Average PCA visualization of company embedding by sectors.

VI. Conclusion

In this work, we present an effective and flexible RGStocknet framework, which can be adapted to arbitrary time series forecasting model, learning trading patterns, and utilizing relations extracted from raw time series data to predict the future movement of stock companies. All economic factors can affect each other in this complex financial world. Predicting trends with the RGStocknet module can be seen as jointly considered historical time sequence and interaction between other factors in the financial market.

We evaluate the proposed RGStocknet framework on the real world S&P500 dataset, which includes 500 large companies listed on stock exchanges in the United States. From the experiment results, we demonstrate that our proposed framework can boost the vanilla baseline on classification performance. Our proposed model outperform baseline for Profit-Score by 2.97%, 2.48%, 7.03% and accuracy by 25.50%,

TABLE III
TRADING PERFORMANCE OF INDIVIDUAL COMPANIES.

	AIV			BRK.B			COST			HPQ			VZ		
Model	**RET¯%]**	**PP [%]**	**HBP [%]**	**RET¯%]**	**PP [%]**	**HBP [%]**	**RET [%]**	**PP [%]**	**HBP [%]**	**RET [%]**	**PP [%]**	**HBP [%]**	**RET [%]**	**PP [%]**	**HBP [%]**
BH	2.72	-	-	18.38	-	-	3.26	-	-	30.43	-	-	-8.66	-	-
LSTM	-2.55	33.33	1.70	0.29	50.00	0.82	5.27	40.00	7.50	**30.43**	**100.00**	**100.00**	**-0.63**	60.00	57.8
RG-LSTM	**5.77**	**73.33**	**40.68**	**11.51**	**100.00**	**27.87**	**12.96**	**66.67**	**42.5**	30.21	66.67	59.20	-2.29	**66.67**	**96.33**
ALSTM	-2.41	33.33	2.54	0.45	**66.67**	1.64	**5.25**	**100.00**	7.50	30.43	**100.00**	**100.00**	**-2.23**	**66.67**	**96.33**
RG-ALSTM	**4.54**	**64.71**	**36.44**	**5.90**	50.00	**18.03**	4.47	50.00	**50.83**	**23.19**	64.29	42.40	-4.45	0.00	92.66
ResNet	-7.45	57.69	38.14	3.31	44.23	34.43	15.9	56.82	35.00	4.49	56.25	23.20	5.09	**53.66**	40.37
RG-ResNet	**5.08**	**62.75**	**48.31**	**8.59**	**51.85**	**51.64**	**32.96**	**59.38**	**50.83**	**27.37**	**66.57**	**54.40**	**6.61**	47.47	**43.12**

17.53% , 12.75%. respectively. Besides, for real-world applications, we make investment simulations based on our prediction results. Our framework with any time series forecasting model can surpass the benchmark buy and hold (B&H) strategy, and improve for return by 39.1%, 2.1%, 67.3%, and Sharpe ratio by 6.1%, 3.4%, 39.2% respectively. We also visualize that the representations of company embedding have a meaningful distribution under principal component analysis (PCA) projection.

The experiment results imply that the financial market does have interaction indeed. Our proposed method can not only extract the implicit knowledge graph from the raw time series but also available to integrate the external knowledge graph since the general property of the graph embedding model. This shows that our model has the advantage of consuming all kinds of resources to enhance further the prediction performance.

ACKNOWLEDGMENT

This research was partially supported by Ministry of Science and Technology, Taiwan, under grant no. 109-2218-E-009-014.

REFERENCES

[1] A. W. Lo and A. C. MacKinlay, "When are contrarian profits due to stock market overreaction?" *The Review of Financial Studies*, vol. 3, no. 2, pp. 175–205, 1990.

[2] L. Ramchand and R. Susmel, "Volatility and cross correlation across major stock markets," *Journal of Empirical Finance*, vol. 5, no. 4, pp. 397–416, 1998.

[3] K. F. Luchtenberg and Q. V. Vu, "The 2008 financial crisis: Stock market contagion and its determinants," *Research in International Business and Finance*, vol. 33, pp. 178–203, 2015.

[4] K. Chan, "A further analysis of the lead–lag relationship between the cash market and stock index futures market," *The Review of Financial Studies*, vol. 5, no. 1, pp. 123–152, 1992.

[5] L. Zhang, C. Aggarwal, and G.-J. Qi, "Stock price prediction via discovering multi-frequency trading patterns," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 2141–2149.

[6] F. Feng, H. Chen, X. He, J. Ding, M. Sun, and T.-S. Chua, "Enhancing stock movement prediction with adversarial training," *arXiv preprint arXiv:1810.09936*, 2018.

[7] Z. Hu, W. Liu, J. Bian, X. Liu, and T.-Y. Liu, "Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, 2018, pp. 261–269.

[8] Y. Xu and S. B. Cohen, "Stock movement prediction from tweets and historical prices," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 1970–1979.

[9] D. Matsunaga, T. Suzumura, and T. Takahashi, "Exploring graph neural networks for stock market predictions with rolling window analysis," *arXiv preprint arXiv:1909.10660*, 2019.

[10] J. Liu, Z. Lu, and W. Du, "Combining enterprise knowledge graph and news sentiment analysis for stock price prediction," in *Proceedings of the 52nd Hawaii International Conference on System Sciences*, 2019.

[11] S. Deng, N. Zhang, W. Zhang, J. Chen, J. Z. Pan, and H. Chen, "Knowledge-driven stock trend prediction and explanation via temporal convolutional network," in *Companion Proceedings of The 2019 World Wide Web Conference*, 2019, pp. 678–685.

[12] T. Lin, T. Guo, and K. Aberer, "Hybrid neural networks for learning the trend in time series," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, no. CONF, 2017, pp. 2273–2279.

[13] W. Li, R. Bao, K. Harimoto, D. Chen, J. Xu, and Q. Su, "Modeling the stock relation with graph network for overnight stock movement prediction," no. CONF, pp. 4541–4547, 2020.

[14] K. Hou, "Industry information diffusion and the lead-lag effect in stock returns," *The Review of Financial Studies*, vol. 20, no. 4, pp. 1113–1138, 2007.

[15] H. Shahrur, Y. L. Becker, and D. Rosenfeld, "Return predictability along the supply chain: the international evidence," *Financial Analysts Journal*, vol. 66, no. 3, pp. 60–77, 2010.

[16] J. Conrad and G. Kaul, "Time-variation in expected returns," *The Journal of Business*, pp. 409–425, 1988.

[17] T. Chordia and B. Swaminathan, "Trading volume and cross-autocorrelations in stock returns," *The Journal of Finance*, vol. 55, no. 2, pp. 913–935, 2000.

[18] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Advances in Neural Information Processing Systems*, 2013, pp. 2787–2795.

[19] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes." in *Aaai*, vol. 14, no. 2014, 2014, pp. 1112–1119.

[20] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[21] C. Nugent, "S&p500 stock data: Historical stock data for all current s&p500 companies," *Dataset available from https://www.kaggle.com/camnugent/sandp500*, 2020.

[22] S. Nayak, B. B. Misra, and H. S. Behera, "Impact of data normalization on stock index forecasting," *International Journal of Computer Information Systems and Industrial Management Applications*, vol. 6, no. 2014, pp. 257–269, 2014.

[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[24] D. M. Nelson, A. C. Pereira, and R. A. de Oliveira, "Stock market's price movement prediction with lstm neural networks," in *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2017, pp. 1419–1426.

[25] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[26] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," in *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2017, pp. 1578–1585.

[27] X. Han, S. Cao, L. Xin, Y. Lin, Z. Liu, M. Sun, and J. Li, "Openke: An open toolkit for knowledge embedding," in *Proceedings of Conference on Empirical Methods in Natural Language Processing*, 2018.

[28] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[30] K. P. F.R.S., "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.