LSTM-GCN：

A Deep Learning Framework for Industry Index Forecasting

Xie, Xinyu (xinyuxie@link.cuhk.edu.cn)

Abstract

Industry index forecasting is a very hot topic in the financial field. The relationship between the secondary industries and its primary industries is likely to provide valuable clues for industry index forecasts. However, this clue is difficult to establish connections and quantify. With the development of methods such as machine learning, more and more methods such as machine learning are applied to this field.

This group will use graph learning techniques based on deep learning models that perform well in the traffic transportation field to solve this topic. The relationship between the secondary industry and the primary industry is analogous to the main road and similar Branch roads. And the model is used to learn the relationship between each industry indicator, which is analogous to the relationship between traffic flow. Finally, this article will use the methods of Long Short-Term Memory Neural Network (LSTM) and Graph Convolutional network (GCN) to predict the industry index and explore the relationship between the secondary industries and its primary industries.

Keyword: Industry Index, LSTM, GCN

INTRODUCTION

Industry index forecasting is one of the most challenging topics in the finance area. With the growing volume and variety in the industry index data, data-driven industry index forecasting methods have shown considerable promise in their ability to outperform conventional and simulation-based methods. Recently, there has been a surge of interest in the use of machine learning to help aid in the accurate predictions of financial markets. Despite the exciting advances in this cross-section of finance and AI, many of the current approaches are limited to using technical analysis to capture the temporal dynamic relationships of each industry index and thus limited to certain experimental setups to obtain good prediction results. However, the rich relations between secondary industries and their belonging primary industries may contain valuable clues for industry index prediction. For example, the industry indexes under the same primary industry may have a similar trend.

Previous work on this topic roughly categorizes existing models into two categories: classical time series models and machine learning models. Most of the studies focusing on industry index forecasting using traditional methods were developed when the industry index was less complex, and the sizes of the datasets were relatively small.

Due to the conventional graph learning techniques cannot capture the temporal relationships of each industry index, the deep learning models are proposed in this project, such as Long Short-Term Memory Neural Network (LSTM) and Graph Convolutional network (GCN) can effectively learn high dimensional relational features and achieve good forecasting performance. Recurrent neural network (RNN) and its variants, including long short-term memory (LSTM) networks, have also shown great potential for solving forecasting problems.

In this project, we learn the industry relationships as a graph and conduct convolution on the industry network-based graph. To incorporate temporal dynamic characteristics, this project proposed an industry index graph convolution operator. Base on this operator, this project proposes an industry index graph convolutional LSTM (TGC-LSTM) to model the dynamics of the industry. The main contributions of our work include:

1. An industry index graph is proposed to capture the relationships between industries.
2. An industry index graph convolutional LSTM neural network is proposed to learn the dynamic temporal relationships presented in industry index data.
3. Introduce the principle of TGC-LSTM.
4. Using the real-world industry index data set, implement the model on the dataset and interpret the result.

As for the project question, it aims at exploring how the temporal dynamic in each industry affect the industry index and further make more accurate predictions on industry index.

LITERATURE REVIEW

The project was mainly inspired by the paper called 'Traffic Graph Convolutional Recurrent Neural Network: A Deep Learning Framework for Network-Scale Traffic Learning and Forecasting 'by Zhiyong Cui, this paper learns the traffic network as a graph and propose a novel deep learning framework, Traffic Graph Convolutional Long Short-Term Memory Neural Network (TGC-LSTM), to learn the interactions between roadways in the traffic network and forecast the network-wide traffic state. Referring to this idea, this project turns the traffic intersections in the paper into secondary industries and whether the traffic intersections are connected to each other is converted to whether secondary industries belong to the same primary industry for subsequent analysis, and uses TGC model with each secondary industry as the vertex and the edge of each graph is the weight of whether two secondary industries belong to the same primary industry, if they belong to the same primary industry the edge equals to 1, else equals to 0.

The second paper the team consult is that exploring graph neural networks for stock market predictions with rolling window analysis. And this paper investigates the effectiveness of work at the intersection of market predictions and graph neural networks, which hold the potential to mimic the ways in which investors make decisions by incorporating company knowledge graphs directly into the predictive model. Using the dataset which shows mainly supplier relations among Japanese and foreign companies, the paper implements the Graph convolutional network (GCN) and LSTM methods to make predictions. These algorithms present an opportunity to make predictive models which better represent the increasingly fast-paced and interconnected nature of the global financial system by incorporating knowledge graphs.

Feng and He (2019) propose a novel neural network-based framework, named Relational Stock Ranking (RSR), to solve the stock prediction problem in a learning-to-rank fashion. And they also devise a new component in neural network modeling, named Temporal Graph Convolution, to explicitly capture the domain knowledge of stock relations in a time-sensitive manner. Sequential data and stock relation data are used to complete the experiment. The experiment is aimed to solve three questions which are study of stock ranking formulation, impact of stock relations and study on back-testing strategies. In conclusion, experimental results on NASDAQ and NYSE demonstrate the effectiveness of the model—with three different back-testing strategies, the RSR framework outperforms the S&P 500 Index with significantly higher return ratio. And the model can be used to explore the potential of emphasizing top-ranked entities with more advanced learning-to-rank techniques. In addition, risk management techniques in finance can be put into the RSR framework to force the predictions to be risk sensitive. Given that the proposed TGC is a general component to model relational data, the potential of TGC in enhancing the neural network solutions for tasks with such relational data is also can be explored, especially recommender system and product search.

METHODOLOGY

3.1 LSTM

LSTM is a kind of neural network that contains LSTM blocks or other types of neural networks. It can memorize values of variable length of time. There is a gate in the block to determine whether the input is important enough to be remembered and whether it can be output.

Meanwhile, LSTM is a special RNN, mainly to solve the problem of gradient disappearance and gradient explosion in the training process of long sequences. It means, compared to ordinary RNNs, LSTM can perform better in longer sequences.

The following specifically analyzes the internal structure of LSTM.

First, use the current input x^t of the LSTM and the h^{t-1} passed down from the previous state to stitch and train to obtain four states.

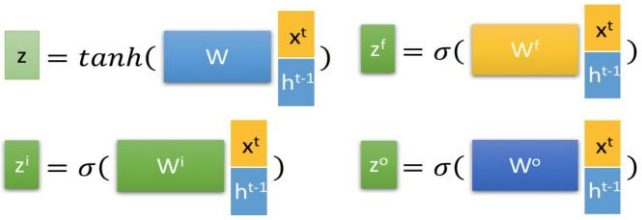

Fig. 1. internal structure of LSTM

Among them, z^f, z^i, z^o are multiplied by the splicing vector by the weight matrix, and then converted into a value between 0 and 1 through a function as a gated state. The Z will convert the result into a value between -1 and 1 through a tanh function.

Then, the use of these four states in LSTM will be introduced by the graph below.

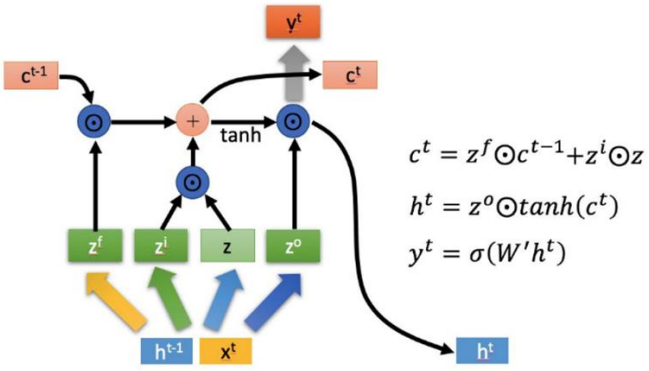

Fig. 2. the use of these four states in LSTM

⊙ is Hadamard Product, it means, the corresponding elements in the operation matrix are multiplied, so the two multiplication matrices are required to be of the same type. ⊕ represents matrix addition.

There are three main stages in LSTM. The first stage is Forgetting stage. This stage is mainly to selectively forget the input from the previous node to forget the unimportant and remember the important. Specifically, the calculated z^f (f means forget) is used as the forget gate control to control the C^{t-1} of the previous state which needs to be left and which needs to be forgotten.

The next stage is selecting memory stage. This stage selectively "memorizes" the input of this stage. The main reason is to select and memorize the input X^t. Record what is important, and write less what is not. The current input content is represented by the Z calculated above. The selected gating signal is controlled by z^i (i stands for information). Add the results obtained in the above two steps, C^twhich is transferred to the next state can be gotten.

The last stage is output stage. This stage will determine which will be regarded as the output of the current state. It is mainly controlled by z^o. It also scales the C^o obtained in the previous stage.

3.2 GCN

The financial data of different industries comes in the form of graphs. To better illustrate the relationships of data between different industries, the algorithm GCN is used. GCN, short for Graph Convolutional Networks, is a branch of Graph Neural Networks that extends existing neural networks for processing the data represented in graph domains (Zhou, J. et.al, 2018).

In a graph, each node is defined by its features and the related nodes. The target of GCN is to learn a state which contains all the related information of neighborhood for each node. More specifically, given a graph $G = (V, E)$, a GCN takes an input feature matrix $N \times F^0$, named X, where N is the number of nodes and F^0 is the number of input features for each node, and an $N \times N$ matrix representation of the graph structure, named A as the adjacency matrix of G as input. A hidden layer in the GCN can thus be written as $H^i = f(H^{i-1}, A))$ where $H^0 = X$ and f is a propagation. Each layer H^i is corresponded to an $N \times F^i$ feature matrix where each row is a feature representation of one node. At each layer, these features are aggregated so as to form the next layer's features by using the propagation rule f. In this way, features become increasingly more abstract after each consecutive layer. In the setting of this GCN framework, variants differ only in the choice of propagation rule f. For example, one of the simplest possible propagation rules is $f(H^i, A) = \sigma(AH^iW^i)$, where W^i is the weight matrix for layer i and σ is a non-linear activation function such as the ReLU function ("How to do Deep Learning on Graphs with Graph Convolutional Networks", 2020).

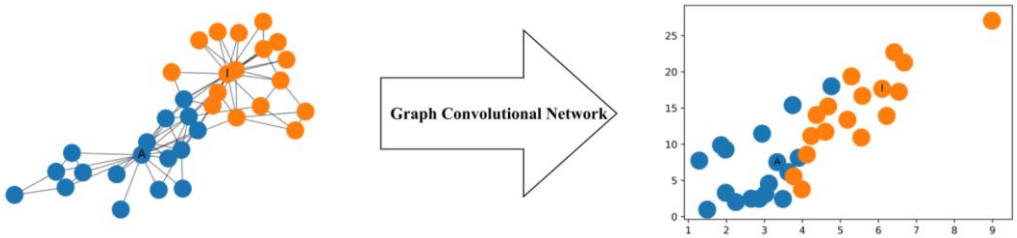

Fig. 3. A 2-dimensional representation of each node in a network produced by a GCN

3.3 The proposed model: GCN-LSTM

By combining the above two algorithms, GCN and LSTM, we propose a new model, named Graph Convolutional Network LSTM (GCN-LSTM) recurrent neural network, as shown on the

right side of the Fig. 2. This model takes both the advantages of GCN and LSTM, which engaged the ability of learning both the complex spatial dependencies and the dynamic temporal dependencies presented in financial data of different industries. In this model, the gates structure in the LSTM and the hidden state are unchanged, but the input is replaced by the graph convolution features, which are reshaped into a vector $GC_t^{\{K\}} \in \mathbb{R}^{KN}$. The forget gate f_t, the input gate i_t, the output gate o_t, and the input cell state $\tilde{C}_t$ in terms of time step t are defined as follows:

$$f_t = \sigma_g(W_f \cdot GC_t^{\{K\}} + U_f \cdot h_{t-1} + b_f)$$

$$i_t = \sigma_g(W_i \cdot GC_t^{\{K\}} + U_i \cdot h_{t-1} + b_i)$$

$$o_t = \sigma_g(W_o \cdot GC_t^{\{K\}} + U_o \cdot h_{t-1} + b_o)$$

$$\tilde{C}_t = tanh(W_c \cdot GC_t^{\{K\}} + U_c \cdot h_{t-1} + b_c)$$

Where $\cdot$ is the matrix multiplication operator. W_f, W_i, W_o, and $W_c \in \mathbb{R}^{N \times KN}$ are the weight matrices, mapping the input to the three different gates and the input cell state, while U_f, U_i, U_o, and $U_c \in \mathbb{R}^{N \times N}$ are the weight matrices for the preceding hidden state. At the same time, b_f, b_i, b_o, and$b_c \in \mathbb{R}^N$ are four bias vectors. The σ_g is the gate activation function, which typically is the sigmoid function, and tanh is the hyperbolic tangent function for the cell state (Cui, Z. et.al, 2019).

Due to the fact that each node in the industries network graph is influenced by the preceding states of itself and its neighboring nodes, the LSTM cell state of each node in the graph should also be affected by cell states of neighboring nodes. Thus, a cell state gate is designed and added in the LSTM cell. The cell state gate, as shown in Fig. 2, is defined as:

$$C_{t-1}^* = W_N \odot (\tilde{A}^K \odot \mathcal{FFR}) \cdot C_{t-1}$$

Where W_N is a weight matrix to measure the contributions of neighboring cell states, $\odot$ is the Hadamard product operator, i.e. the element-wise matrix multiplication operator, $\mathcal{FFR}$ is a Free-flow reachable matrix $\in \mathbb{R}^{N \times N}$ defined as $\mathcal{FFR}_{i,j} = 1$ when average speed that a motorist would travel minus distance larger or equal than zero. To correctly reflect the financial industries network structure, the W_N is constrained by multiplying a $\mathcal{FFR}$ based K-hop adjacency matrix, $\tilde{A}^K \odot \mathcal{FFR}$. With this gate, the influence of neighboring cell states will be considered when the cell state is recurrently input to the subsequent time step (Cui, Z. et.al, 2019). Then, the final cell state and the hidden state are calculated as:

$$C_t = f_t \odot C_{t-1}^* + i_t \odot \tilde{C}_t$$

$$h_t = o_t \odot \tanh(C_t)$$

Through this proposed GCN-LSTM model, predictions of the yield rates of sector indices can be obtained for deeper analysis.

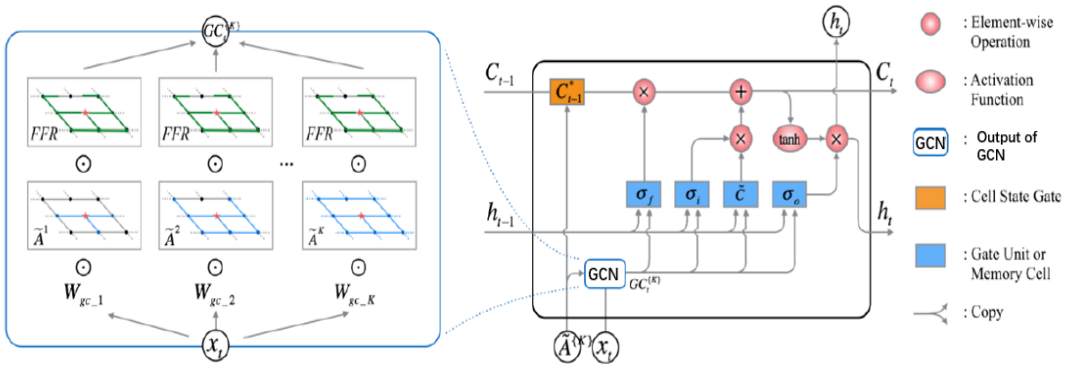

Fig. 4. The architecture of the proposed Graph Convolution Network LSTM is shown on the right side. The GCN as a component of the proposed model is shown on the left side in detail by unfolding the financial industries graph convolution at time t.

DATA

The data is downloaded from wind. And research objects are Citic Securities'primary and secondary industry index which is also used to generate adjacency matrix. The data are collected from January 3, 2017 to June 8, 2020. Overall, there are 103 industries, and the data consists more than 80,000 records for 833 days. It is shown that column in red box represents stock code, line in yellow box means date, and data in blue box represents index. Stock name can be found corresponding to the stock code. According to the industry tables, stock code starts with "CI0050" means primary industry while stock code starts with "CI0051" means secondary industry.

	20170103	20170104	20170105
CI005101.WI	189.84	190.965	192.8242
CI005102.WI	3452.5162	3503.2294	3580.7721
CI005187.WI	1387.3166	1400.7029	1411.1283
CI005104.WI	1664.5642	1674.7485	1687.5818
CI005105.WI	2638.8041	2646.4393	2676.1842
CI005106.WI	11024.3764	11076.9605	11220.3551
CI005107.WI	4127.1039	4179.6776	4201.7268
CI005188.WI	1284.7374	1316.3625	1316.9219
CI005109.WI	2404.1839	2414.0296	2414.9416
CI005110.WI	6403.9441	6459.2082	6468.0259
CI005111.WI	1397.5973	1418.626	1418.1338
CI005189.WI	1278.9269	1291.8747	1293.1162
CI005190.WI	1573.1735	1594.1848	1595.5112
CI005113.WI	4635.4422	4706.2645	4714.4528
CI005191.WI	1483.8143	1514.6109	1515.9468
CI005192.WI	1757.6077	1772.927	1769.5655

Fig. 5. Data

Fig. 6. Primary Industries

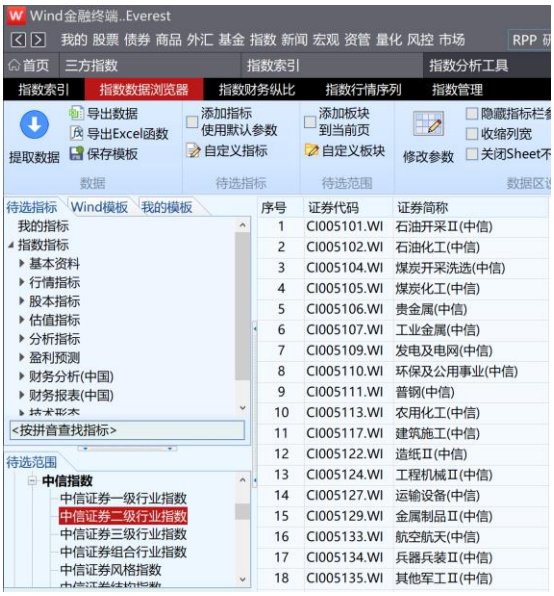

Fig. 7. Secondary Industries

After loading the data, json file is used to identify the industry. Then industries with incomplete data is deleted. Next, the adjacency matrix is made according to the primary industry which the secondary industry belongs to. And the primary industry of the same secondary industry is connected.

But there is a question needs to be solved, that is how to forecast with financial data. Traditionally, time series forecasting problem can be cast as a supervised learning problem. Previous timesteps are used as input features and next timestep is used as the output to predict. Then, the forecasting question can be modeled as predicting the feature value in the future, given the historical values of the feature for that entity as well as the feature values of the entities connected to the entity. In this project, how to forecast with financial data is treated as a supervised learning problem, and price of previous period is used to predict the rate of return over the next period.

Scaling is also important before putting data into analysis. Each timestep is standardized and

both independent variables and dependent variable are also normalized. Axis is set to be 1 to standardize each industry over time.

Just like for modeling any standard supervised learning problem, the data is divided into mutually exclusive train and test sets. In this project, first 80% observations are used for training and the rest for testing.

EXPERIMENT

To the best of our knowledge, our work is the first one to incorporate industry relations into the models for industry index prediction, especially neural network-based ones. As such, in this section, we conduct experiments with the aim of answering the following research questions:

(1) How is performance of our model?

(2) How to do the back-testing for historical data? and how is the back-testing result?

(3) What is the difference of result between our work and another literatures' work?

4.1 Experiment Setting

In this paper, the experimental machine consists of Intel Core i7 CPU @ 1.8 GHz workstation running Windows 10 with 16 GB of RAM. The model and strategy are implemented in Python 3.6. The core library used are tensorflow 2.2.0, stellargraph 1.1.0. Tensorflow is most common structure to build neural network. Stellargraph is a library to build advanced graph neural network. According to the methodology, the model was built as Figure 7.

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 104, 50)]	0
tf_op_layer_ExpandDims (Tens	[(None, 104, 50, 1)]	0
reshape (Reshape)	(None, 104, 50)	0
fixed_adjacency_graph_convol	(None, 104, 50)	13420
reshape_1 (Reshape)	(None, 104, None, 1)	0
permute (Permute)	(None, None, 104, 1)	0
reshape_2 (Reshape)	(None, None, 104)	0
lstm (LSTM)	(None, 100)	82000
dropout (Dropout)	(None, 100)	0
dense (Dense)	(None, 104)	10504

Total params: 105,924
Trainable params: 95,108
Non-trainable params: 10,816

Fig. 7. The parameters of model

Here, many parameters are set. For LSTM, the timestep is 50, and layer_size is 100. For GCN layer, gcn_layer_size is 2, gcn activation function is Relu, and dropout is 0.2. For training setting, the optimizer is Adam, learning rate is 0.0001, loss is mean square error, and evaluation metric is spearman's rank correlation. Mean square error is

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Spearman’s rank correlation is

$$r_s = \rho_{\mathrm{rg}_X,\mathrm{rg}_Y} = \frac{\mathrm{cov}(\mathrm{rg}_X, \mathrm{rg}_Y)}{\sigma_{\mathrm{rg}_X}\sigma_{\mathrm{rg}_Y}}$$

The Spearman’s rank correlation between two variables is equal to the Pearson correlation between the rank values of those two variables; while Pearson's correlation assesses linear relationships, Spearman's correlation assesses monotonic relationships (whether linear or not). If there are no repeated data values, a perfect Spearman correlation of +1 or −1 occurs when each of the variables is a perfect monotone function of the other.

4.2 Training Result

To answer the question (1), we train the model and do the validation as well. In rolling training, we train 50 models in total from 3 Jan. 2017 to 18 Mar. 2020. Each model can inference a result including loss and evaluation. In the Figure 8, we choose two models result to present. One is the model which uses data from 3 Jan. 2017 to 28 May. 2018.

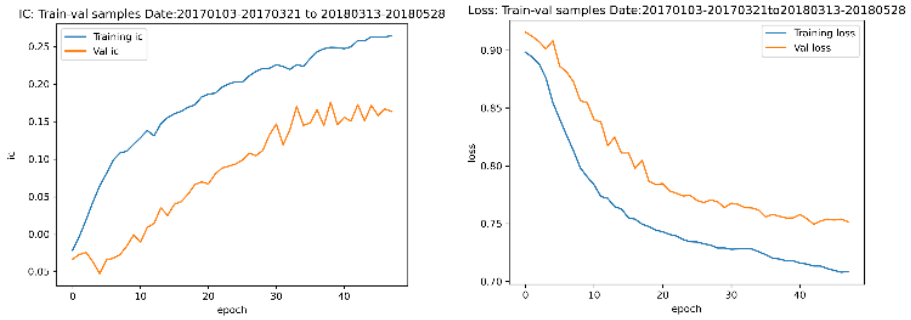

Fig. 8.1 the result of model from 3 Jan. 2017 to 28 May. 2018.

Another one is the model which uses data from 3 Jan. 2017 to 19 Jul. 2019.

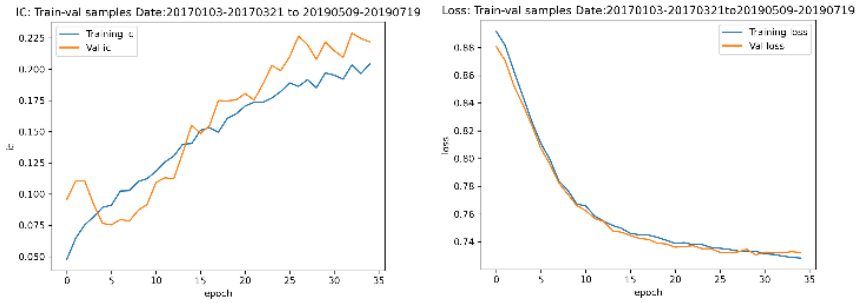

Fig. 8.2 the result of model from 3 Jan. 2017 to 19 Jul. 2019.

In the figures above, the left figure is about spearman’s rank correlation, and the right figure is about loss. As we can see, the performance of model is perfect, because the spearman’s rank correlation increases gradually with number of epochs increasing, and both losses decreases quickly at first then become flattening. Such phenomenon demonstrates that a standard training process in model training is machine learning.

4.3 Back-testing strategy

To answer the question (2), a trading strategy is designed according to the prediction model based on LSTM-GCN and secondary industries indexes. The main part of the strategy consists of two steps.

1) Predict the secondary industries index price trend by the trained model in 50 trading days.

2) Long 30 industries indexes which have highest rank of return ratio with equal volume.

3) Hold the indexes for 50 trading days, and then go back to step 1) to adjust the holdings.

Following this back-testing pattern, we compare the net value with strategy in which long all the secondary industries indexes equally. Unfortunately, we found that two return cluster with each other, but net value of top-industries strategy is higher than net value of all-industries strategy. It is not significant that our strategy works.

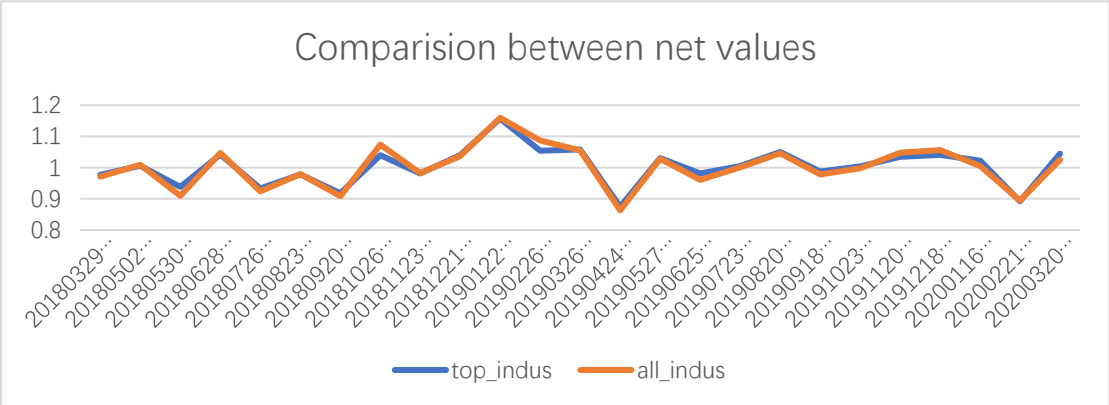

Fig. 9. The comparison between net values

4.4 Discussion

In our reference literature *Temporal relational ranking for stock prediction*, the result, in the below, seems more competitive.

	NASDAQ			NYSE		
	MSE	MRR	IRR	MSE	MRR	IRR
Rank_LSTM	**3.79e-4±1.11e-6**	4.17e-2±7.50e-3	**0.68±0.60**	2.28e-4±1.16e-6	3.79e-2±8.82e-3	**0.56±0.68**
GBR	3.80e-4±2.40e-7	3.32e-2±4.50e-3	0.33±0.34	**2.26e-4±4.20e-7**	3.64e-2±5.35e-3	0.65±0.27
GCN	**3.79e-4±9.70e-7**	3.24e-2±3.21e-3	0.11±0.06	**2.26e-4±6.60e-7**	3.99e-2±1.03e-2	0.74±0.30
RSR_E	3.80e-4±7.20e-7	3.94e-2±8.15e-3	0.81±0.85	2.29e-4±2.77e-6	4.28e-2±6.18e-3	**0.96±0.47**
RSR_I	**3.79e-4±6.60e-7**	**4.09e-2±5.18e-3**	**1.19±0.55**	**2.26e-4±1.37e-6**	4.58e-2±5.55e-3	0.79±0.34

Fig. 10. the result of Temporal relational ranking for stock prediction

However, this literature focuses the stock market of the U.S., and uses different relational information. The reasons why our strategy cannot get significant excess income may be in the followings:

1) The relation between secondary industries may not provide more information to do the price prediction. In other literature, the prediction target is stock price, and relation information is whether both companies becomes same industries. In this situation, the relation may provide

more useful information.

2) The graph convolutional neural network may be not suitable for prediction of price of industries of index. Initially, the graph convolutional neural network solved the semi-supervised learning problems. In addition, to use graph convolutional neural network, the raw data can only use time series data but bot factor data. Therefore, the pure time series data is weak for the prediction in stock market.

CONCLUSION

This paper introduces a model based on the method of Long Short-Term Memory Neural Network (LSTM) and Graph Convolutional network (GCN), which is very efficient in solving transportation problems. Because the relationship between the primary industries and the secondary industries is similar to relationship of traffic flow between main roads and branch roads.

Among them, the LSTM model perfectly solves the problems of long-term sequence gradient disappearance and gradient explosion, and it filters and records useful information in the time span. The GCN model uses the idea of graph structure to extract effective features, and solves the problem of mutual influence of information in the space span, thereby well capturing the characteristics of spatial data. The combination of the two methods solves the problem of mutual influence of financial information in time and space span, and extracts more accurate and effective features, so as to more efficiently explore the relationship between the primary industries and the secondary industries.

In this paper, the first and second industry index data of Wind database are used for model training and back-testing verification. The back-testing results show that this method indeed verifies that the primary industry indexes and the secondary industry indexes are correlated, and the strategies formulated will produce certain effects, but cannot obtain significant excess returns.

In the future, we will consider improving the model. On the one hand, the independent variables are considered to change. Adopt the use of stock prices instead of industry indexes. It may improve efficiency of forecasting and increase strategic returns. On the other hand, consider improving the graph neural network to achieve better results.

REFERENCE

1. Cui, Z., Henrickson, K., Ke, R., & Wang, Y. (2019). Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting. *IEEE Transactions on Intelligent Transportation Systems*.

2. Matsunaga, D., Suzumura, T., & Takahashi, T. (2019). Exploring Graph Neural Networks for Stock Market Predictions with Rolling Window Analysis. *arXiv preprint arXiv:1909.10660*.

3. Feng, F., He, X., Wang, X., Luo, C., Liu, Y., & Chua, T. S. (2019). Temporal relational ranking for stock prediction. *ACM Transactions on Information Systems (TOIS)*, *37*(2), 1-30.

4. *Hongyi, Li* (2019) *Deep Learning*. (2019). from https://www.youtube.com/ channel/ UC2ggjtuuWvxrHHHiaDH1dlQ/videos

5. Zhou, J., Cui, G., Zhang, Z., Yang, C., Liu, Z., Wang, L., ... & Sun, M. (2018). Graph neural networks: A review of methods and applications. *arXiv preprint arXiv:1812.08434*.

6. *How to do Deep Learning on Graphs with Graph Convolutional Networks*. Medium. (2020). Retrieved 26 July 2020, from https://towardsdatascience.com/how-to-do-deep-learning-on-graphs-with-graph-convolutional-networks-7d2250723780.