A Study of Stock Price Function Based on Hybrid Deep Learning Model

Yangwenyuan Deng

Business School, University of New South Wales, Sydney 1466, Australia
A02494328@gadsdenstate.edu

Abstract. Recently, global economy has recovered from the COVID impact. Stock index is reasonable indicator of economy. The prediction of stock index contributes to both forecasting trend of global economy and portfolio construction. According to the characters of price factors, this paper proposes a hybrid model, combining Variational Autoencoder (VAE), Multi-directional Delayed Embedding (MDT), Gated Recurrent Unit (GRU), and Multi-Head Self-Attention (MHSA) modules, named VMGM, to predict the movement of Shanghai Stock Exchange (SSE 50). It combined deep learning, unsupervised learning, and transformer to one model whose prediction result is more accurate than traditional model. The original data contain 7 price factors of SSE 50 from 2004 to 2020. Writer uses VAE and moving average convergence/divergence (MACD) to generate new price factors for preprocessing and feature engineering. Final price factors are transformed by MDT and convolutional neural network (CNN) before entering GRU and Multi-Head self-attention. The prediction result of model is more accurate than controlled sets which prove the choice logistic of each module.

Keywords: Hybrid Model, Deep Learning, Transformer, Unsupervised Learning, Price Function.

1 Introduction

Stock price function is still uncovered for researchers. Many researchers have put their effort to discover the complete price function. The all-round hidden logic between the stock price and stock factor is hard to find out while the machine learning could offer approximate function relationship for price and factors. Stock prediction starts with machine learning prediction, such as logistic regression, decision tree and support vector machine (SVM). However, the accuracy of these tool is limited by the linear function, Computing bottleneck and abnormal data [1].

Deep learning could mitigate above issues by stacking and activation function, which make it mainstream method of current research [2]. Deep learning model will replace the original problem into easy and simple ones. This characteristic is suitable for price function discovering. It has been widely adopted in prediction related tasks.

Recently, more researchers paid attention to unsupervised learning in prediction area as it could excavate the hidden part of price function. However, the accuracy of

P. Kar et al. (eds.), *Proceedings of the 2023 International Conference on Image, Algorithms and Artificial Intelligence (ICIAAI 2023)*, Advances in Computer Science Research 108,
https://doi.org/10.2991/978-94-6463-300-9_5

unsupervised network in stock prediction is not satisfactory [3]. Thus, this paper uses Variational Autoencoder (VAE) to generate hidden price factors to further mine the price function.

Transformer is introduced by Vaswani et al. in 2017 which is based on self-attention mechanisms to process the nature language tasks [4]. The hidden relationship between inputs become an obstacle for the exploration of price function. Self-attention is designed to compute correlations between different parts of the entire input. This character is suitable for stock price prediction as price factors are time series with massive data with correlations.

Based on the convolutional neural network gated recurrent unit (CNN-GRU) model, this paper proposal a hybrid model for stock prediction. The experimental data are the price factors of Shanghai Stock Exchange (SSE 50) download from Baostock.com. the efficiency of VAE, Multi-directional Delayed Embedding (MDT), and Multi-Head Self-Attention (MHSA) will be tested their contribution to original model. The prediction result will be presented by three evaluation indexes.

2 Related Work

With the development of technology, stock predict with deep learning become more popular. From Artificial neural network (ANN) to recurrent neural network (RNN), researchers improve the time efficiency of model by accepting sequence data and state information reseveration [5]. In further study, the long-short term memory (LSTM) and its variants (GRU) were proposed to address the issue of gradient vanishing [6]. In 2017, Samarawickrama and Fernando prove GRU has higher accuracy than LSTM in stock prediction [7]. Following research focus on the improvement of CNN-GRU model. Liu et al. propose the CEGH model whose prediction result is satisfied [8]. Inconclusion, there are two methods to improve model efficiency. Model superposition is the first method. Karim et.al construct a hybrid model Bi-LSTM and GRU which outperform the tradition model in forecasting [9]. Li and Qian improve prediction accuracy by combining GRU with attention module [10]. The data processing and feature engineering is another way. Research by Huang et al. prove the additional price factors contribute to better model performance [11]. Yu et al. verified that more inputs data positive affects the deep learning results [12]. However, most of above research are supervised learning which focuses on the model accuracy and dominance relationship between factors and price [13]. More research should explore the power of unsupervised learning in price function. Polamuri et. introduce the hybrid prediction algorithm module to GAN model which improves the accuracy in the long time series [14].

3 Method

3.1 Modules

VAE. This paper use VAE and MDT as feature engineering. Variational Autoencoder combines elements of both autoencoders and Bayesian inference to learn a probabilistic mapping between the input data and the latent space [15]. The encoder network maps the input data to a low-dimensional latent representation, while the decoder network maps the latent representation back to the original input.
This paper creatively utilizes close price as inputs of VAE network to create hidden price factors. There are two reasons. Firstly, through this, the created factors may contribute to more accurate results. Secondly, this is a discovery of price function. The new factors could be the unknown factors related to function outputs rather than data have similar distribution with original factors.

MDT. This paper introduces MDT method to replace sliding windows. MDT will reshape the inputs with similar structure to image which further strengthen the relationship between price factors [16]. It transfers a vector $x = (x_1, x_2, ..., x_n), T \in \Re_n$ into a Hankel matrix $M_\tau(x)$. The MDT operation can be represented by following function:

$$M_\tau(x) = fold_{(n,\tau)}(Cx) \tag{1}$$

Function $fold_{(n,\tau)} : \Re^{\tau \times (n-\tau+1)} \rightarrow \Re^{\tau \times (n-\tau+1)}$ is a folding operator that converts vectors into a matrix.

CNN. CNN is widely used in image related commission for its efficiency and accuracy. This is because adjacent pixel points have strong relationship to ensure import information remain after the transform of CNN or LSTM. The stock price factors, such as ma5, ma30, open, volume and close, also have similar characteristic. CNN includes pooling layers which transform the data to reduce the feature dimension:

$$l_t = \tanh(x_t * k_t + b_t) \tag{2}$$

where l_t represent the output of after convolution neural network, x_t represents the input vector, k_t represents the weight of the convolution kernel, b_t is the convolution kernel bias, and tanh is the activation function.

GRU. GRU merge input gate and forget gat into an update gate to short training time while maintain the model accuracy. GRU has two gate structure which are update gate and reset gate. The process of GRU could be summarized as follow:

$$Z_t = \sigma(W_z \cdot h_{t-1} + W_z \cdot x_t), \tag{3}$$

$$rt = \sigma(W_r \cdot h_{t-1} + W_r \cdot x_t), \quad (4)$$

$$h_t = tanh(W_h \cdot (r_t \odot h_{t-1}) + W_h \cdot x_t), \quad (5)$$

$$h_t = (1 - Z_t) \odot h_{t-1} + Z_t \odot h_t, \quad (6)$$

where $\cdot$ represents matrix multiplication, and $\odot$ represents matrix corresponding element multiplication. r_t represent the reset gate which partly delete information from former hidden layer h_{t-1}. The update gate Z_t control which information of previous status is kept for current status h_t. W is the weight matrix, b is the bias vector, $[h_{t-1}, x_t]$ represents the connection of the two vectors. σ and $tanh$ are the sigmoid or hyperbolic tangent activation functions.

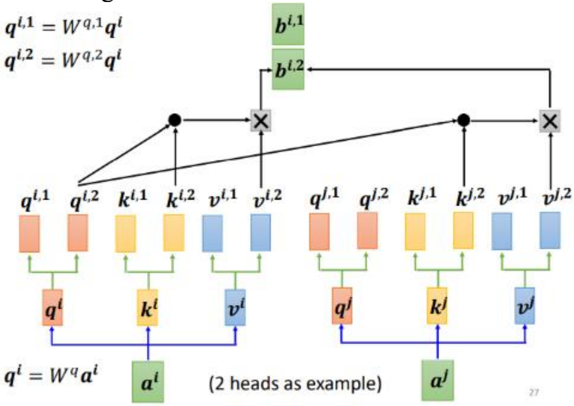

Fig. 1. Structure of MHSA module [4].

MHSA. The overview of self-attention mechanism is shown in Fig 1. Where q^{ij}, k^{ij}, v^{ij} are results of a^i multiplying the weight matrix W_i^j, the number of head i represents the number of correlations between inputs. Attention Matrix $A' = Q * K^T$, output $B = A' * V$. Where Q, V, K^T is the matrix form of q^{ij}, k^{ij}, v^{ij}. The total process could be summarized as follows:

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_n)W^O \quad (7)$$

$$head_i = Attention(QW_i^Q, KW_i^Q, VW_i^Q) \quad (8)$$

3.2 Hybrid Model Summarization

The model is presented in Table 1 with parameters of the model listed in Table 2.

Table 1. Model structure.

#	layer	shape	#	layer	shape
1	InputLayer	(None,11,11)	7	MHSA	(filters=3, head=5)
2	Reshape	(None, 11, 11, 1)	8	GRU	(None, 11, 128)

3	Conv2D	(None, 11, 11, 64)	9	GRU	(None, 64)
4	MaxPooling2D	(None, 11, 11, 64)	10	Dense	(None, 10)
5	Dropout	(0.2)	11	Dropout	(0.25)
6	Reshape	(None, 11, 704)	12	Dense	(None, 1)

Table 2. Model parameters.

Parameters	Value	Parameters	Value
Convolution layer filters	64	Dropout layers 2	0.25
Convolution layer kernel_size	3	Learning rate	0.001
MaxPooling2D pool_size	2	Time_step	1
Activation function	Relu	Loss function	Mean square error
Dropout layers	0.2	Batch_size	64
Self-attention head & filter	5 & 3	Optimizer	Adam
GRU layers	2	Epochs	200
Number of hidden units in GRU layer 1	128	Dropout layers 2	0.25
Number of hidden units in GRU layer 2	64	Learning rate	0.001

4 Result

4.1 Experiment Setting and Evaluation Indexes

The original data download from baostock only contain 7 price factors. Through the close price, this paper constructs moving average, exponential moving average, MACD, 20-day standard deviation, Bollinger to original data. The total factors are scaler from 0 to 1 by the Minmax Scaler as it does not impact the generation of VAE module. The final input of model is presented in Table 3.

Table 3. Input features of the model.

Open	High	Low	Close	Preclose	Volume	Amount
0.0133	0.0133	0.0143	0.0128	0.0131	0.0019	0.0010
PctChg	Index	ma26	ma7	26ema	12ema	MACD
0.4887	0.4624	0.0153	0.0092	0.0146	0.0115	0.5163
20sd	Upper band	Lower band	Momentum	VAE1	VAE2	VAE3
0.0411	0.0201	0.0160	0.0128	-0.2630	0.2946	-0.7568

Each model will run 20 times and the best prediction consequence will be saved and compared with others. The loss function is MSE of the validation set. It facilitates generalization ability of models. Prediction results are evaluated by three indexes include mean absolute error (MAE), mean square error (RMSE) and R-square (R2).

The functions of indexes are presented in follows:

$$R^2 = 1 - \frac{\left(\sum_{i=1}^{n}(y_i - y_i)^2\right)/n}{\frac{\sum_{i=1}^{n}(y_i - y_i)^2}{n}} \quad (9)$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - y_i| \quad (10)$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - y_i)^2} \quad (11)$$

where y_i and y_i respectively represent the prediction close price and real close price, n is the number of samples.

In this paper, model combinations such as VAE-WGAN, CNN-GRU, MHSA-GRU, VAE-CNN-GRU and VAE-MDT-CNN-GRU, are built to test the contribution of each module.

4.2 Quantitative result

The loss value of each model is shown in Table 4. On the one hand, prediction results are more accurate when more modules are integrated which prove the efficiency of above discussion of chapter 3. On the other hand, the accuracy improvement become less in later tests.

Table 4. Loss value of each model.

Model	Validation Set Loss	R square	MAE	RMSE
MHSA-GRU	4500.22	0.9474	59.15	76.16
CNN-GRU	4422.43	0.9463	58.90	76.01
MDT-CNN-GRU	2401.16	0.9754	39.17	52.49
VAE-MDT-CNN-GRU	2359.76	0.9766	38.98	51.76
VAE-MDT-CNN-GRU-MHSA	2190.43	0.9764	38.19	51.04

5 Conclusion

The proposed model has better prediction accuracy than the controlled models which prove the module selection is reasonable. MHSA module has similar prediction accuracy with CNN while it costs more time at the frontend. It also improves model when set at the backend of the model. MDT is the most efficient module to improve accuracy with little time costs. Price factors generated by VAE network positively contribute to model prediction prove the possibility of hidden factors. This idea will help further study to discover the price function. However, there are some improvements need to be done in future research: (1) The most suitable number of hidden price factors and generator model need more tests. This paper only generated 3 factors with VAE model. The alternatives of VAE, such as GAN, may has better contribution. (2) The parameter of each module only through three times tests, such as the MHSA. Study of parameter could further improve the model accuracy. (3) Writer has trained model with CBAM and MHSA attention modules while this module contribute to a little. These attention modules are not designed for time-series task. A more suitable attention module is needed.

References

1. Chen, S., Goo, Y. J. J., & Shen, Z. D.: A hybrid approach of stepwise regression, logistic regression, support vector machine, and decision tree for forecasting fraudulent financial statements. The Scientific World Journal, 1-9 (2014).
2. LeCun, Y., Bengio, Y., & Hinton, G.: Deep learning. nature, 521(7553), 436-444 (2015).
3. Zhang, K., Zhong, G., Dong, J., Wang, S., & Wang, Y.: Stock market prediction based on generative adversarial network. Procedia computer science, 147, 400-406 (2019).
4. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., et al.: Attention is all you need. Advances in neural information processing systems, 30 (2017).
5. Langkvist, M., Karlsson, L. & Loutfi, A.: A review of unsupervised feature learning and deep learning for time-series modeling. Pattern Recognition Letters, 42: 11–24 (2014).
6. Cho, K., Van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. & Bengio, Y.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv, 1406: 1078 (2014).
7. Samarawickrama, A. J. P., & Fernando, T. G. I.: A recurrent neural network approach in predicting daily stock prices an application to the Sri Lankan stock market. In 2017 IEEE International Conference on Industrial and Information Systems (ICIIS) 1-6 (2017).
8. Liu, Y., Liu, X., Zhang, Y., & Li, S.: CEGH: A Hybrid Model Using CEEMD, Entropy, GRU, and History Attention for Intraday Stock Market Forecasting. Entropy, 25(1), 71 (2023).
9. Karim, M. E., Foysal, M., & Das, S.: Stock Price Prediction Using Bi-LSTM and GRU-Based Hybrid Deep Learning Approach. In Proceedings of Third Doctoral Symposium on Computational Intelligence: DoSCI 2022, 701-711 (2022).
10. Li, C., & Qian, G.: Stock Price Prediction Using a Frequency Decomposition Based GRU Transformer Neural Network. Applied Sciences, 13(1), 222 (2022).
11. Huang, Y., Capretz, L. F., & Ho, D.: Machine learning for stock prediction based on fundamental analysis. In 2021 IEEE Symposium Series on Computational Intelligence (SSCI), 1-10 (2021).
12. Yu, SS., Chu, SW., Chan, YK. & Wang, CM.: Share Price Trend Prediction Using CRNN with LSTM Structure. Smart Science, 7: 189-197 (2019).
13. Li, J., Pan, S., Huang, L. & Zhu, X.: A machine learning based method for customer behavior prediction. Tehnicki Vjesnik Technical Gazette, 26: 1670–1676 (2019).
14. Polamuri, S. R., Srinivas, K., & Mohan, A. K.: Multi-model generative adversarial network hybrid prediction algorithm (MMGAN-HPA) for stock market prices prediction. Journal of King Saud University-Computer and Information Sciences, 34(9), 7433-7444 (2022).
15. Kingma, D. P., & Welling, M.: Stochastic gradient VB and the variational auto-encoder. In Second international conference on learning representations, ICLR 19, 121 (2014).
16. Cao, CF., Luo, ZN., Xie, JX. & Li, L.: Stock Price Prediction Based on MDT-CNN-LSTM Model. Computer Engineering and Applications, 58: 280-286 (2022).