Extreme Market Prediction for Trading Signal with Deep Recurrent Neural Network

Zhichen Lu[1,2,3], Wen Long[1,2,3(✉)], and Ying Guo[1,2,3]

[1] School of Economics and Management, University of Chinese Academy of Sciences, Beijing 100190, People's Republic of China
longwen@ucas.ac.cn
[2] Research Center on Fictitious Economy and Data Science, Chinese Academy of Sciences, Beijing 100190, People's Republic of China
[3] Key Laboratory of Big Data Mining and Knowledge Management, Chinese Academy of Sciences, Beijing 100190, People's Republic of China

Abstract. Recurrent neural network are a type of deep learning units that are well studied to extract features from sequential samples. They have been extensively applied in forecasting univariate financial time series, however their application to high frequency multivariate sequences has been merely considered. This paper solves a classification problem in which recurrent units are extended to deep architecture to extract features from multi-variance market data in 1-minutes frequency and extreme market are subsequently predicted for trading signals. Our results demonstrate the abilities of deep recurrent architecture to capture the relationship between the historical behavior and future movement of high frequency samples. The deep RNN is compared with other models, including SVM, random forest, logistic regression, using CSI300 1-minutes data over the test period. The result demonstrates that the capability of deep RNN generating trading signal based on extreme movement prediction support more efficient market decision making and enhance the profitability.

Keywords: Recurrent neural networks · Deep learning
High frequency trading · Financial time series

1 Introduction

Financial time series forecasting, especially stock price forecasting has been one of the most difficult problems for researchers and speculators. The difficulties are mainly caused by the uncertainty and noise of samples, the generation of samples are not just consequence of historical behavior information contained in samples, but also influenced by information beyond historical samples such as macro economy, investor sentiment etc. Traditional statistics methods were well prefered to fit financial time series consider the their robustness to noise

Y. Shi et al. (Eds.): ICCS 2018, LNCS 10861, pp. 410–418, 2018.
https://doi.org/10.1007/978-3-319-93701-4_31

and good explaination. But consider it's pool fitting capability, their implement on sending signal for trading were mostly undesirable. Machine learning method were exploited to this problem and get considerable progress but bottleneck are lead by their sensitivity to parameters and tendency to overfitting.

In recent years, deep learning method have shown remarkable progress in many tasks such as computer visions [9,15], nature language process [7], speech recognition [5] etc. The deep architecture have shown powerful capabilities of feature extraction and fitting,and the auxiliary tricks such as dropout [14],batch normalization [6] etc. and optimizer such as Rmsprop, Adam [8], Nadam etc. were designed to improve the efficiency of training and figure problems of overfitting, gradient vanish, gradient explosion that substantially led by the deep architecture and non-linear mapping during training. In application on financial time series prediction,numerous studies have shown that neural network is a very effective tool in financial time series forecasting [2,13,16]. Weigend et al. [12,17,18] compared the performance of neural network with that of traditional statistics methods in predicting financial time series and neural network showed superior forecasting ability than tradition ways. NN models were firstly applied to solve problem in financial domain in White research [19], five different exchange rates were predicted by feedforward and recurrent networks and it was shown in their finding that performance of predictions can be improved by applying NN. Some works show that neural networks are efficient and profitable in forecasting financial time series [4]. Some combinations of multiple neural networks or NN with other method are also proposed for financial time series forecasting. For example, a hybrid artificial method based on neural network and genetic algorithm was used to model daily exchange rates [11].

In this paper, we extended recurrent neural network into deep architecture as a classifier to predict the movement trend of stock price. The performance of models were evaluated on CSI 300 stock index and the results of classification were considered as trading signal to evaluate the profitability.

2 Recurrent Neural Networks with Deep Architecture

2.1 RNN

RNNs [20] are sequence learners which have achieved much success in applications such as natural language understanding, language generation, video processing, and many other tasks [1,3,10]. A simple RNN is formed by a repeated application of a function F_h to the input sequence $\mathcal{X}_t = (X_1, \ldots, X_T)$.For each time step $t = 1, \ldots, T$,the function generates a hidden state h_t:

$$h_t = F_h(X_t, h_{t-1}) = \sigma(W_h X_t + U_h h_{t-1} + b_h) \tag{1}$$

for some non-linear activation function $\sigma(x)$, where X_t denotes the input at time t, W_h denotes the weight of connection between input and hidden state, U_h denotes the weight of connection between the hidden states h_t and h_{t-1}, and b_h denotes the bias of activation.

2.2 Batch Normalization

With the depth of a net work growing, problems such as gradient explosion and gradient vanish may be incurred, and some approach were proposed to alleviate these problems, one of them was batch normalization [6]. The main idea of batch normalization is to perform normalization on the output of each layers for each mini batch [BN], and to reduce internal covariate shift of each layer's activation, the mean and variance of the distribution are parameterized and learned while training. A batch normalization layer can be formulated as:

$$\hat{x}^k = \frac{x^k - E[x^k]}{\sqrt{Var[x^k]}} \tag{2}$$

$$y^k = \gamma^k \hat{x}^k + \beta^k \tag{3}$$

where x^k is the activation of kth layer, y^k is the output after batch normalization, γ and β are parameters of batch normalization to be learned.

2.3 Deep Recurrent Architecture

To address the problem of stock price prediction, we extend recurrent neural networks into deep architecture. The input of model are multi-variance time series of high frequency market data. At each frame, the hidden outputs h_t from recurrent layer are fully connected to the next recurrent layer so that the recurrent units are stacked into deeper architecture. Between each stacked recurrent layers, batch normalization are performed on each time axis so that the output of each recurrent units can be normalized to avoid the problems that may led by scale of activation while training on mini-batch. At the last recurrent layer, the last normalized frame was connected to a fully connected perception and output with a softmax layers. The details of our deep architecture are presented in Fig. 1.

3 Data and Preprocessing Methodology

3.1 Sampling

To exploit trading signal from historical market behavior (open, close, high, low, amount, volumns), market data of CSI 300 from the period Jan. 2016 to Dec. 2016 with frequency of 1-minute were sampled into short sequence by constant windows with length of 120, normalization are performed on each univariate time series of each segmented sequence.

3.2 Labeling Methodology

The profitability not only depend on the correctness of prediction on the movement direction of price, but also the margin of price movement that captured

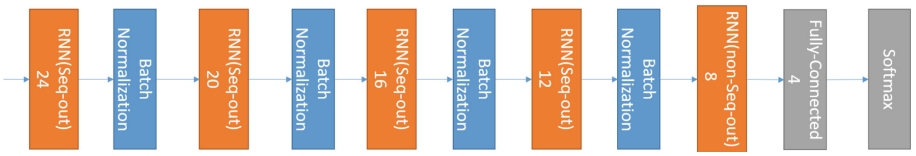


Fig. 1. RNN architecture for financial time series prediction.

by trading signal. So we label samples by assign those whose future prices rise or fall sharply into two single classes and the others as another class, which is defined as:

$$L_t = \begin{cases} 1 & r_t > r_\theta \\ 0 & \text{Others} \\ -1 & r_t < r_{1-\theta} \end{cases}$$

where L_t denotes the label of sample X_t,$r_t = ln\frac{close_{t+t_{forward}}}{close_t}$ denotes the logarithm return of the stock index $t_{forward}$ minutes after t, and θ denotes the threshold of labeling with $p(r_t > r_\theta) = \theta$ and $p(r_t < r_{1-\theta}) = \theta$. Another reason of the labeling methodology is that samples contain higher noise when the price fluctuates in a narrow range, dependency between history behavior and future trend are tend to be weaker than other two situations. Detail statistics of training and test sets are shown in Table 1.

Table 1. Statistic of data sets

(a) Number of samples in each class with different θ.

θ	Training sets			Testing sets		
	Rise	Fluctuation	Fall	Rise	Fluctuation	Fall
0.1	12239	12277	12194	2454	2412	2370
0.15	18355	18397	18315	4511	4386	4261
0.2	24470	24504	24433	6880	6761	6642
0.25	30588	30622	30551	9667	9521	9375
0.3	36699	36738	36665	12982	12652	12322

(b) tuples $(r_\theta, r_{1-\theta})$ in different θ and $t_{forward}$

θ	$t_{forward} = 5$	$t_{forward} = 10$	$t_{forward} = 15$	$t_{forward} = 20$	$t_{forward} = 25$	$t_{forward} = 30$
0.1	(0.0026,-0.0025)	(0.0036,-0.0035)	(0.0044,-0.0042)	(0.0051,-0.0049)	(0.0057,-0.0054)	(0.0063,-0.0059)
0.15	(0.0019,-0.0018)	(0.0027,-0.0026)	(0.0033,-0.0031)	(0.0039,-0.0036)	(0.0044,-0.0039)	(0.0048,-0.0043)
0.2	(0.0014,-0.0013)	(0.0022,-0.002)	(0.0026,-0.0024)	(0.003,-0.0027)	(0.0034,-0.003)	(0.0038,-0.0033)
0.25	(0.0011,-0.001)	(0.0017,-0.0015)	(0.0021,-0.0019)	(0.0024,-0.0021)	(0.0027,-0.0023)	(0.003,-0.0025)
0.3	(0.0008,-0.0007)	(0.0013,-0.0011)	(0.0016,-0.0014)	(0.0019,-0.0016)	(0.0021,-0.0017)	(0.0023,-0.0019)

4 Experiment

4.1 Experiment Setting

We generate data sets with 5 different thresholds θ and 6 kinds of time window $t_{forward}$ of prediction to train 30 RNNs. While training models and learning the parameters, back propagation and stochastic gradient descent(SGD) are used for updating the weights of neurons, dropout rates are 0.25 among recurrent layers and 0.5 in fully connected layers, and the batch size is 320. The learning rate of optimizer are 0.5 at the start of training, and decayed by 0.5 if the accuracy on validation sets haven't improve for 20 epochs. A early stop condition is set, which is that accuracy on validation sets haven't improve for 150 epochs.

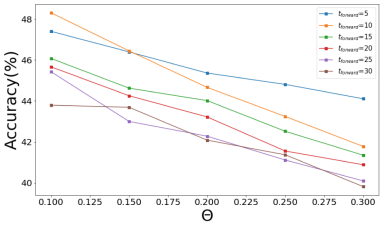
4.2 Results Discussion

The performance of each model on test set are shown in Fig. 2. We find that the prediction accuracy increases as the threshold decreases, which is likely because the samples corresponded to larger margin of rise or fall show stronger dependency between features and labels. However, the change of time windows of prediction do not show obvious effect on model performance. Specifically, the model with $\theta = 0.1, t_{forward} = 10$ reaches the best performance with the accuracy of 48.31%, which is remarkable for 3-classes financial time series prediction, and can give powerful support for market practice.

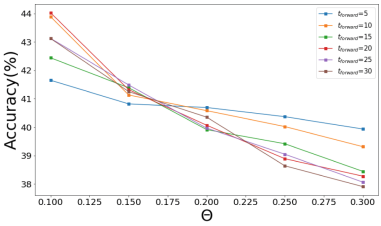
We further test our 30 data sets on SVM, Random Forest, Logistic Regression and traditional statistic model linear regression to compare results with RNN, the best five results of each model on 30 data sets are shown in Table 2. We can find that the performance of RNN is far better than any of the three traditional machine learning models or linear regression, and the accuracy of SVM, the best of the other four models, is outperformed by that of RNN about 4%.

4.3 Market Simulation

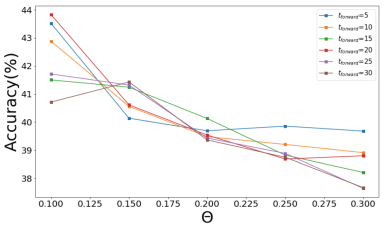
We simulate real stock trading based on the prediction of RNN to evaluate the market performance. We follow a strategy proposed by Lavrenko et al. are followed: if the model predicts the new sample as positive class, our system will purchase 100,000 CYN worth of stock at next minutes with open price. We assume 1,000,000 CYN are available at the start moment and trading signal will not be executed when cash balance is less than 100,000 CYN. After a purchase, the system will hold the stock for $t_{forward}$ minutes corresponding to the prediction window of model. If during that period we can sell the stock to make profit of r_θ (threshold profit rate of labeling) or more, we sell immediately, otherwise, at the end of $t_{forward}$ minute period, our system sells the stock with the close price. If the model predicts the new sample as negative class, our system will have a short position of 100,000 CNY worth of stock. Similarly, system will hold the stock for $t_{forward}$ minutes. If during the period the system can buy the stock at $r_{1-\theta}$ lower than shorted, the system close the position of short by buying the



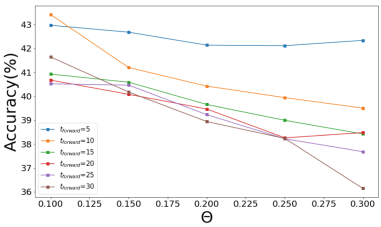
(a) RNN



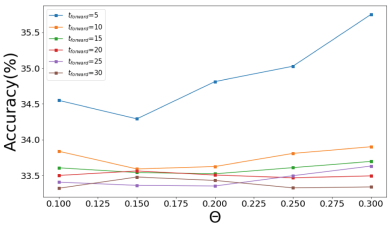
(b) SVM



(c) Random Forest



(d) Lofistic Regression



(e) Linear Regression

Fig. 2. Performance of each model on 30 datasets.

Table 2. Best 5 results of each model on 30 data sets

	RNN	SVM	Logistic regression	Random forest	Linear regression
1	$t_{forward} = 10\theta = 0.1$ 48.31%	$t_{forward} = 20\theta = 0.1$ 44.03%	$t_{forward} = 10\theta = 0.1$ 43.41%	$t_{forward} = 20\theta = 0.1$ 43.83%	$t_{forward} = 5\theta = 0.3$ 35.75%
2	$t_{forward} = 5 \ \ \theta = 0.1$ 47.40%	$t_{forward} = 10\theta = 0.1$ 43.89%	$t_{forward} = 5 \ \ \theta = 0.1$ 42.97%	$t_{forward} = 5 \ \ \theta = 0.1$ 43.52%	$t_{forward} = 5\theta = 0.25$ 35.03%
3	$t_{forward} = 10\theta = 0.15$ 46.45%	$t_{forward} = 25\theta = 0.1$ 43.13%	$t_{forward} = 5 \ \ \theta = 0.15$ 42.67%	$t_{forward} = 10\theta = 0.1$ 42.88%	$t_{forward} = 5\theta = 0.2$ 34.81%
4	$t_{forward} = 5 \ \ \theta = 0.15$ 46.40%	$t_{forward} = 30\theta = 0.1$ 43.12%	$t_{forward} = 5 \ \ \theta = 0.3$ 42.33%	$t_{forward} = 25\theta = 0.1$ 41.71%	$t_{forward} = 5\theta = 0.1$ 34.55%
5	$t_{forward} = 15\theta = 0.1$ 45.67%	$t_{forward} = 15\theta = 0.1$ 42.44%	$t_{forward} = 5 \ \ \theta = 0.2$ 42.13%	$t_{forward} = 15\theta = 0.1$ 41.50%	$t_{forward} = 5\theta = 0.15$ 34.29%

stock to cover. Or else, at the end of the period, system will close the position in the same way at the close price of the end of period.

To simulate this strategy we use models trained on training sets to predict the future trend of stock in each minute from April 18th 2016 to January 30th

2017, and send trading signal according to the prediction made by models. The profits of each model on market simulation are presented in Table 3. We can see from results that all simulations based on trading signals sent by prediction models are all significantly more profitable than randomly buy and sell strategy, which implies that prediction models can catch suitable trading points by predict future trends to make profit. Among these prediction models, all simulations based on machine learning prediction models result in higher profit than linear regression, which indicates that the non-linear fitting of machine learning models show better efficiency in extreme market signal learning than traditional statistic models. Specially, RNN achieves 18.13% more profit than the statistic model, even the second best model is 11.13% less profit than RNN.

Table 3. Market simulation results

	Hyper-parameter	Profit
RNN	$\theta = 0.1$	**24.50%**
	$t_{forward} = 10$	
Linear regression	$\theta = 0.3$	6.37%
	$t_{forward} = 5$	
Logistic regression	$\theta = 0.1$	13.37%
	$t_{forward} = 10$	
Random forest	$\theta = 0.1$	9.65%
	$t_{forward} = 10$	
SVM	$\theta = 0.1$	12.93%
	$t_{forward} = 10$	
Random buy and sell	—	1.03%
	$t_{forward} = 10$	

5 Conclusion

In this paper we extend RNN into deep structure to learning the extreme market from the sequential samples of historical behavior. High frequency market data of CSI 300 are used to train the deep RNN and the deep structure do improve the accuracy of prediction compared with the traditional machine learning method and statistical method. In the sight of practice, this paper presents the applicability of deep non-linear mapping on financial time series, and 48.31% accuracy for 3-classes classification is meaningful for practice in market. And we further prove the better profitability of deep RNN in market simulation than that of any traditional machine learning models or statistic models.

Acknowledgement. This research was partly supported by the grants from National Natural Science Foundation of China (No. 71771204, 71331005, 91546201).

References

1. Bhattacharya, A., Parlos, A.G., Atiya, A.F.: Prediction of MPEG-coded video source traffic using recurrent neural networks. IEEE Trans. Signal Process. **51**(8), 2177–2190 (2002)
2. Cheng, W., Wagner, L., Lin, C.H.: Forecasting the 30-year us treasury bond with a system of neural networks. Neuroizest J. **4**, 10–16 (1996)
3. Dauphin, Y., Yao, K., Bengio, Y., Deng, L., Hakkani-Tur, D., He, X., Heck, L., Tur, G., Yu, D., Zweig, G.: Using recurrent neural networks for slot filling in spoken language understanding. IEEE/ACM Trans. Audio Speech Lang. Process. **23**(3), 530–539 (2015)
4. Emam, A.: Optimal artificial neural network topology for foreign exchange forecasting. In: Proceedings of the 46th Annual Southeast Regional Conference on XX, pp. 63–68. ACM (2008)
5. Graves, A., Mohamed, A., Hinton, G.: Speech recognition with deep recurrent neural networks. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 6645–6649. IEEE (2013)
6. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning, pp. 448–456 (2015)
7. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
8. Kingma, D., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
9. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
10. Mikolov, T., Karafit, M., Burget, L., Cernock, J., Khudanpur, S.: Recurrent neural network based language model. In: INTERSPEECH 2010, Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September, pp. 1045–1048 (2010)
11. Nag, A.K., Mitra, A.: Forecasting daily foreign exchange rates using genetically optimized neural networks. J. Forecast. **21**(7), 501–511 (2002)
12. Panda, C., Narasimhan, V.: Forecasting exchange rate better with artificial neural network. J. Policy Model. **29**(2), 227–236 (2007)
13. Sharda, R., Patil, R.B.: Connectionist approach to time series prediction: an empirical test. J. Intell. Manuf. **3**(5), 317–323 (1992)
14. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
15. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)
16. Van Eyden, R.J.: The Application of Neural Networks in the Forecasting of Share Prices (1996)
17. Weigend, A.S.: Predicting sunspots and exchange rates with connectionist networks. In: Nonlinear Modeling and Forecasting, pp. 395–432 (1992)

18. Weigend, A.S., Rumelhart, D.E., Huberman, B.A.: Generalization by weight-elimination with application to forecasting. In: Advances in Neural Information Processing Systems, pp. 875–882 (1991)
19. White, H.: Economic prediction using neural networks: the case of IBM daily stock returns. In: IEEE International Conference on Neural Networks, vol. 2, pp. 451–458 (1988)
20. Williams, R.J., Zipser, D.: A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. MIT Press, Cambridge (1989)