# CNN-LSTM Neural Network Model for Quantitative Strategy Analysis in Stock Markets

Shuanglong Liu[1], Chao Zhang[2], and Jinwen Ma[1](✉)

[1] Department of Information Science, School of Mathematical Sciences and LMAM, Peking University, Beijing 100871, China
jwma@math.pku.edu.cn

[2] Academy for Advanced Interdisciplinary Studies, Peking University, Beijing 100871, China

**Abstract.** In this paper, the convolutional neural network and long short-term memory (CNN-LSTM) neural network model is proposed to analyse the quantitative strategy in stock markets. Methodically, the CNN-LSTM neural network is used to make the quantitative stock selection strategy for judging stock trends by using the CNN, and then make the quantitative timing strategy for improving the profits by using the LSTM. It is demonstrated by the experiments that the CNN-LSTM neural network model can be successfully applied to making quantitative strategy, and achieving better returns than the basic Momentum strategy and the Benchmark index.

**Keywords:** Neural network · CNN · LSTM · Quantitative strategy · Stock markets

## 1 Introduction

The complexity of the internal structure in stock price system and the diversity of the external factors (the national policy, the bank rate, price index, the performance of quoted companies and the psychological factors of the investors) determine the complexity of the stock market, uncertainty and difficulty of stock price forecasting task [1]. The stock market has the characteristics of high return and high risk, which has always been concerned on the analysis and forecast in the stock prices [2,3]. One of the main ideas of the quantitative strategy is to predict and judge the future price of the stock by using the trend of the stock market, and draw up the corresponding investment strategy [4].

A convolutional neural network (CNN) is a mapping from input to output in essence, which can study the mapping relationship without precise mathematical expression between any input and output. As long as convolutional network training using the known pattern with a pooling layer to extract the most representative global features, network has the mapping ability between the input and

S. Liu and C. Zhang—The two authors contributed equally to this paper.


D. Liu et al. (Eds.): ICONIP 2017, Part II, LNCS 10635, pp. 198–206, 2017.
https://doi.org/10.1007/978-3-319-70096-0_21

output [5]. Thus we can use CNN for stock ranker to achieve the quantitative stock selection strategy.

To achieve better returns, we adopt the recurrent neural networks (RNN) which have proved one of the most powerful models for processing sequential data. Long Short-Term Memory (LSTM) is one of the most successful RNNs architectures to fix the vanishing gradient problem in neural network [6]. LSTM introduces the memory cell, a unit of computation that replaces traditional artificial neurons in the hidden layer of the network. With these memory cells, networks are able to effectively associate memories and input remote in time, hence suit to grasp the structure of stock data dynamically over time with high prediction capacity [7]. Hence we can use LSTM to achieve the quantitative timing strategy.

The experimental results show that this CNN-LSTM neural network model can find potential rules from historical datasets, and the corresponding quantitative selection and timing strategy is valid and profitable. The rest of this paper is organized as follows. In Sect. 2, we give a brief review of the CNN and LSTM, then describe the CNN-LSTM framework. Section 3 presents the CNN-LSTM flow chart, and the experimental results of as well as the comparisons of the basic Momentum strategy and Benchmark index. Finally, we conclude the paper and present future work in Sect. 4.

## 2 CNN-LSTM Neural Network

### 2.1 CNN

For supervised classification, CNN is among the most successful models and gets the state-of-the-art result in many benchmarks [8]. Actually, it involves many more connected weights. A form of regularization is realized in the architecture, and some degree of translation invariance is provided automatically. This particular kind of neural network assumes that we wish to learn filters, in a data-driven fashion, as a means to extract features describing the inputs. The full CNN framework and formula derivation can be seen in the literatures [9].

CNNs are hierarchical models whose convolutional layers alternate with subsampling layers, reminiscent of simple and complex cells in the primary visual cortex [10]. At a convolution layer, the previous layer's feature maps are convolved with learnable kernels, which form the output feature map through the activation function. Multiple input maps can be combined as the output with convolutions. For convenience we just introduce the convolution layer:

$$x_j^l = f\left(\sum_{i \in M_j} x_i^{l-1} * k_{ij}^l + b_j^l\right), \tag{1}$$

where $M_j$ represents a selection of input maps.

### 2.2 LSTM

Recurrent neural networks have the capability to dynamically incorporate past experience due to internal recurrence [11]. RNNs can project the dynamic properties of the system automatically, so they are computationally more powerful than feed-forward networks, and the valuable approximation results are obtained for chaotic time series prediction [12,13]. One of RNN models is long-short-term memory which works when there is a long delay, and the signals with a mixture of low and high frequency components can be able to handled. The learning process of RNN models however requires a relatively long time because there is a recurrent network architecture [14].

A schematic of the vanilla LSTM block [15] can be seen in Fig. 1. It features three gates (input, forget and output), block input, a single cell (the Constant Error Carousel), an output activation function, and peephole connections. The output of the block is recurrently connected back to the block input and all of the gates. The vector formulas for LSTM layer forward pass are given in [15]. In order to facilitate your understanding, just listed below:

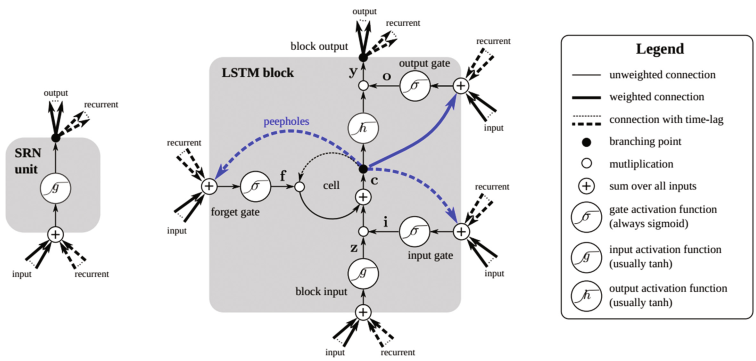


**Fig. 1.** Detailed Long Short-Term Memory block as used in the hidden layers of a recurrent neural network.

$$z^t = g(W_z x^t + R_z y^{t-1} + b_z) \qquad \textit{block input} \tag{2}$$

$$i^t = \sigma(W_i x^t + R_i y^{t-1} + p_i \odot c^{t-1} + b_i) \qquad \textit{input gate} \tag{3}$$

$$f^t = \sigma(W_f x^t + R_f y^{t-1} + p_f \odot c^{t-1} + b_f) \qquad \textit{forget gate} \tag{4}$$

$$c^t = i^t \odot z^t + f^t \odot c^{t-1} \qquad \textit{cell state} \tag{5}$$

$$o^t = \sigma(W_o x^t + R_o y^{t-1} + p_o \odot c^t + b_o) \qquad \textit{output gate} \tag{6}$$

$$y^t = o^t \odot h(c^t) \qquad \textit{block output} \tag{7}$$

where $x^t$ is the input vector at time $t$, the $W$ are input weight matrices, the $R$ are square recurrent weight matrices, the $p$ are peephole weight vectors and

$b$ are bias vectors. Functions $\sigma$, $g$ and $h$ are point-wise non-linear activation functions: *logistic sigmoid* $\left(\frac{1}{1+e^{-x}}\right)$ is used for as activation function of the gates and hyperbolic tangent is used as the block input and output activation function. The point-wise multiplication of two vectors is denoted as $\odot$. The corresponding Back-Propagation Through Time(BPTT) formulas can be found in [15]'s supplementary material.

### 2.3 CNN-LSTM Framework

The details of the CNN-LSTM framework are as follows:

**Algorithm 1.** The CNN-LSTM framework

```
Initialization of parameters and data.
repeat
   repeat
      CNN-quantitative selection step:
      input: 32*1 dimensional matrix, i.e. the monthly rates of return from the first
      13 month to the first 2 month and the daily rates of return from the first 20
      day to the first 1 day.
   until Either the component remains the same in the previous iteration, or the
   iterations reach certain threshold.
   Output the predicted current monthly rate of return.
until All shares are traversed in the A stock market.
Select the top one percent stock in the CNN step output.
repeat
   repeat
      LSTM-quantitative timing step:
      input: 30*6 dimensional matrix, i.e. before 30 days' features: ['open', 'close',
      'high', 'low', 'amount', 'volume'].
   until Either the component remains the same in the previous iteration, or the
   iterations reach certain threshold.
   Output the predicted next 5 days' rate of return: 1 if positive rate, otherwise -1.
until All selected shares are traversed.
Output the current monthly total return.
```

## 3 Experiment Results

We implement CNN-LSTM neural network model for quantitative selection and quantitative timing strategy on the training dataset, and verify its performance on the test dataset. We are exploring a parallel implementation of the learning algorithm that could be run on GPUs. Our experiments are implemented in the Linux system (Ubuntu 16.04.2 LTS) with GPU (device 0: GeForce), and 16.00GB RAM with running Python 2.7 source codes.

This approach should lead to a substantial decrease in training time as the algorithm can take advantage of parallelization at the data-level (since it uses mini-batches) as well as at the network layer level. Alternatively, a more straightforward approach would be to retrain the classifier each month, but update the LSTM more frequently in order to improve profits which are infinitely close to local optimization.

The details of CNN-LSTM flow chart and parameters are described in the Fig. 2 and in the Table 1 as follows:

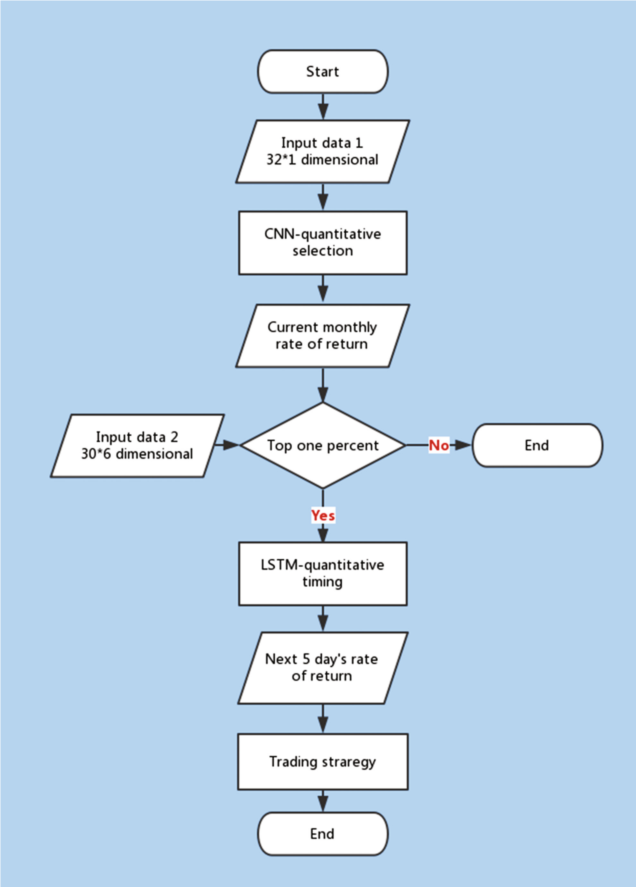


**Fig. 2.** CNN-LSTM flow chart.

We obtain data on individual Chinesse stocks from the SINA FINANCE web. The training set covers the period from 2007-1-1 to 2013-12-31, and the test set covers the period from 2014-1-1 to 2017-3-31. Data setting and preprocessing of CNN-LSTM neural network are described in the framework.

**Table 1.** The parameters for CNN-LSTM.

| Parameters | CNN | LSTM |
|---|---|---|
| Input layer | 1 | 1 |
| Conv/LSTM hidden layer | 2 | 1 |
| FCN hidden Layer | 2 | 1 |
| Output layer | 1 | 1 |
| Epoch | 500 | 100 |
| Activation | ReLU, Tanh | Tanh |
| Weight | Normal(0,1) | Normal(0,1) |
| Optimizer | Adam | Adam |
| Learning rate | 0.001 | 0.001 |
| Objective function | Cross-entropy | Cross-entropy |

We did z-score standardization of data when necessary [16]. For every month $t$, we use the 12 monthly rates of return for month $t-13$ through $t-2$ and the 20 daily rates of return as the input of CNN quantitative selection step, and before 30 days features: [‘open’, ‘close’, ‘high’, ‘low’, ‘amount’, ‘volume’] as input of LSTM quantitative timing step. Only the features which are to be fed to the neural network are chosen and trained for prediction assigning random biases and weights. In our CNN-LSTM model, the LSTM part is composed of a sequential layers followed by 1 LSTM layer and dense layer with Tanh activation.

Over fitting of neural networks is one of the most difficult things to avoid in training neural networks. Over fitting means that the model performs well in training data, but for the other data the predictor effect is poor. The reason is that “rote” data and noises usually lead to complicated model. To avoid over-fitting of the model, the dropout mechanism is added to the CNN-LSTM model and the regularization term is applied to the weights. Dropout refers to drop some features randomly to improve the robustness of the model. Regularization refers to add an L2 norm in the calculation of the loss function, so that some of the weight values close to 0 avoid forced adaptation for each feature. Then it improves the robustness, also gets the effect of feature choice.

When the LSTM predictive value is equal to 1, we buy and hold 5 days, and if previous positions, update the number of held days as 5 and continue held. When the LSTM predictive value is equal $-1$, it continues if short positions, and if already held shares, the number of held days will be decreased by one, and if the number of held days is equal to 0, we will sell the share. Figure 3 shows the position ratios of CNN-LSTM model in the test dataset. The gap between two consecutive months means that we make the quantitative stock selection strategy for each month by using the CNN, thus sells all of shares if possible. Meanwhile, the phenomenon that position ratio is less than 1 before the end of the month demonstrates that the LSTM mechanism makes the quantitative timing strategy effectively.

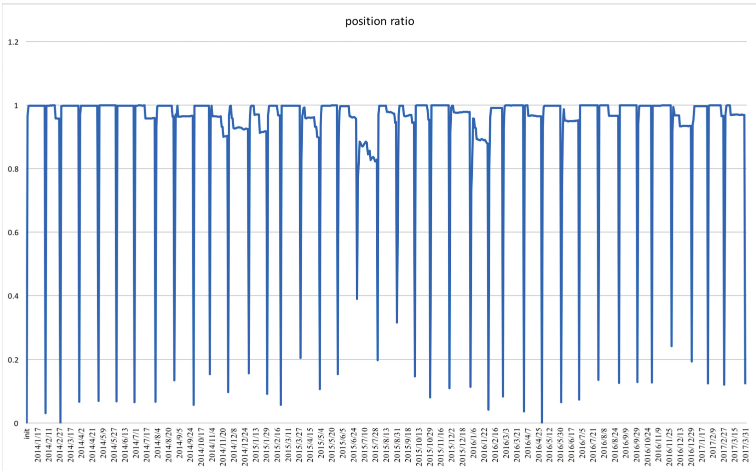


**Fig. 3.** The position ratios of CNN-LSTM model in the test dataset.

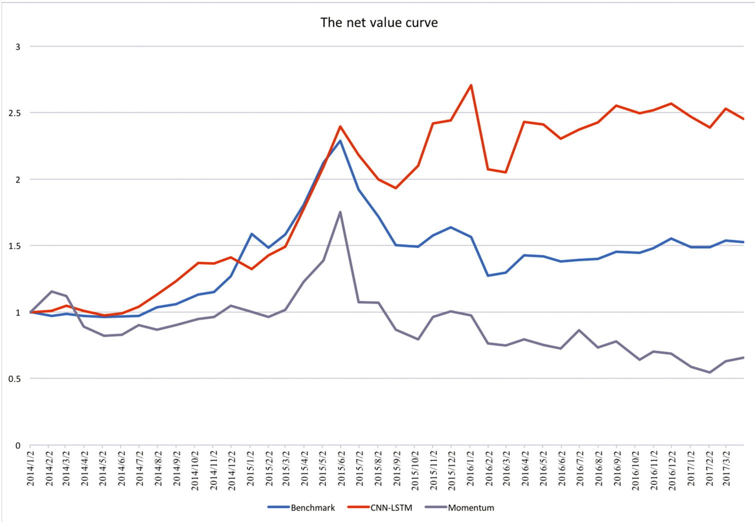


**Fig. 4.** The net value curves of Benchmark, CNN-LSTM and Momentum.

**Table 2.** The comparison of the results

| | Benchmark | CNN-LSTM | Momentum |
|---|---|---|---|
| Annualized rate of return | 0.136 | 0.309 | −0.118 |
| Maximum retracement | 0.443 | 0.241 | 0.689 |

We compare the annualized rate of return, the maximum retracement and the net value of our CNN-LSTM model with the basic Momentum strategy and Benchmark index respectively in the Table 2 and Fig. 4. Basic momentum strategy is the empirical finding that stocks with high past returns over 3-to-12 months (winners) continue to perform well over the next few months relative to stocks with low past returns (losers). The net value curves demonstrate a significant increase in the performances qualitatively. During the stock market crash, the maximum retracement of CNN-LSTM is tolerable. The annualized rate of return using our CNN-LSTM neural network model is more than 2 times as large as the annualized rate of return using Benchmark index. Meanwhile, the maximum retracement of CNN-LSTM neural network model is respectively 34%, 54% of the maximum retracement of the basic Momentum strategy and Benchmark index. The experiments fully illustrate our model is efficient and the investment return is impressive, and verify the robustness and practicability of the algorithm as well.

## 4 Conclusion and Future Work

We have applied the deep learning to stock trading and made two main contributions to the applied machine learning literature. First, we show that CNN can extract useful features even from low signal-to-noise time series data such as financial asset prices if the inputs are appropriately preprocessed. And we make the quantitative stock selection strategy for judging stock trends by using the CNN. Second, we use LSTM neural network to predict a high accuracy in future stock prices and the predicting outcomes are used as timing signals, which significantly improves the retracement of the CNN stock selection model in the backtesting stage. Our model easily accommodates returns of different frequencies as well as nonreturn data and produces investment results that exceed the basic Momentum strategy and Benchmark index in the vast finance literature. We have successfully applied the CNN-LSTM neural network to modeling and making the quantitative stock selection and timing strategy which is feasible, robust and highly profitable.

The issue for future work is reducing computational complexity and increasing computation speed, so that this method can be applied to hours or minutes data instead of the days data. Furthermore, if we apply the model to actual investment decisions, we need to improve on these aspects, such as feature selection, model construction and parameter optimization.

**Acknowledgement.** This work was supported by the Natural Science Foundation of China for Grant 61171138.

## References


1. Fu, C., Fu, M., Que, J.: Prediction of stock price base on radial basic function neural networks. Technol. Dev. Enterp. **4**, 005 (2004)
2. Sun, W., Guo, J., Xia, B.: Discussion about stock prediction theory based on RBF neural network. Heilongjiang Sci. Technol. Inf. **22**, 130 (2010)
3. Liu, S., Ma, J.: Stock price prediction through the mixture of gaussian processes via the precise Hard-cut EM algorithm. In: Huang, D.-S., Han, K., Hussain, A. (eds.) ICIC 2016. LNCS, vol. 9773, pp. 282–293. Springer, Cham (2016). doi:10.1007/978-3-319-42297-8_27
4. Chavarnakul, T., Enke, D.: Intelligent technical analysis based equivolume charting for stock trading using neural networks. Expert Syst. Appl. **34**(2), 1004–1017 (2008)
5. Ding, X., Zhang, Y., Liu, T., Duan, J.: Deep learning for event-driven stock prediction. In: International Conference on Artificial Intelligence, pp. 2327–2333. AAAI Press (2015)
6. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
7. Murtaza, R., Harshal, P., Shraddha, V.: Predicting stock prices using LSTM. Int. J. Sci. Res. (IJSR) **6**(4), 1754–1756 (2017)
8. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
9. Bouvrie, J.: Notes on Convolutional Neural Networks. Neural Nets (2006)
10. Hubel, D.H., Wiesel, T.N.: Receptive fields and functional architecture of monkey striate cortex. J. Physiol. **195**(1), 215–243 (1968)
11. Murtagh, F., Starck, J., Renaud, O.: On neuro-wavelet modeling. Decis. Support Syst. **37**(4), 475–484 (2004)
12. Terzija, N.: Robust digital image watermarking algorithms for copyright protection (2006)
13. Sak, H., Senior, A., Beaufays, F.: Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In: Fifteenth Annual Conference of the International Speech Communication Association (2014)
14. Fryzlewicz, P., Bellegem, S., Sachs, R.: Forecasting non-stationary time series by wavelet process modelling. Ann. Inst. Stat. Math. **55**(4), 737–764 (2003)
15. Greff, K., Srivastava, R.K., Koutnik, J., Steunebrink, B.R., Schmidhuber, J.: LSTM: a search space odyssey. IEEE Trans. Neural Netw. Learn. Syst. **pp**(99), 1–11 (2016)
16. Takeuchi, L., Lee, Y.: Applying deep learning to enhance momentum trading strategies in stocks. Working paper, Stanford University (2013)