USING TENSORFLOW TO SOLVE THE PROBLEMS OF FINANCIAL FORECASTING FOR HIGH-FREQUENCY TRADING

A.V. Bogdanov, A.S. Stankus [a]

St. Petersburg State University, University Embankment 7–9, St. Petersburg, 199034, Russia

E-mail: [a] alexey@stankus.ru

The use of neural networks significantly expands the possibilities of analyzing financial data and improves the quality indicators of the financial market. In article we examine various aspects of working with neural networks and Frame work TensorFlow, such as choosing the type of neural networks, preparing data and analyzing the results. The work was carried out on the real data of the financial instrument Si-6.16 (futures contract on the US dollar rate).

Keywords: Artificial Intelligence, recurrent neural network (RNN), financial market forecasting, TensorFlow.

1. Neural network selection

With an increase in the power of computing resources, it became possible to predict the price movement of stock markets using artificial neural networks (ANN). The most common form of ANN used to predict the stock market is a direct transfer network, using the back-propagation error algorithm to update network weights. These networks are commonly referred to as reverse error propagation networks. Another form of ANN, which is more suitable for price prediction, is a recurrent neural network (RNN) [1] or a time delay neural network (TDNN) [2]. Examples of RNN and TDNN are the networks of Elman, Jordan, and Elman-Jordan.

RNN was created with the ability to process long serial data and solve problems with the distribution of context in time. The model processes one element in a sequence in one-time step. After the calculation, the updated state is transmitted to the next step in time to facilitate the calculation of the next element.

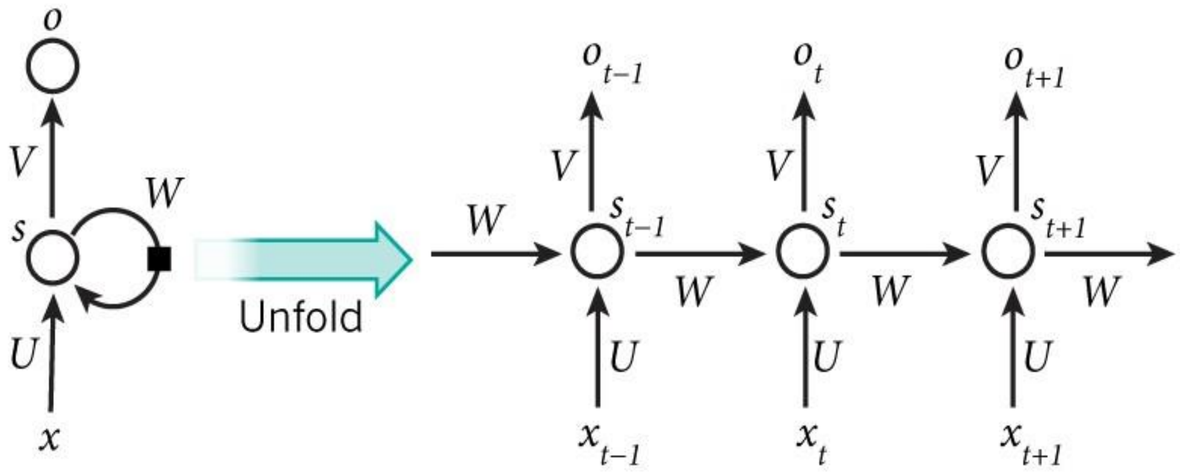

Figure 1. A recurrent neural network with one hidden element (left) and its unfolding version in time (right). The expanded version illustrates what happens in time: st - 1, st, and st + 1 are the same unit with different states at different time steps t - 1, t, and t + 1

However, simple networks that linearly combine the current input element and the last output element can easily lose long-term dependencies. To solve this problem, researchers created a special neuron with a much more complex internal structure to remember the long-term context, called the Long-Short Term Memory (LSTM) cell. He is smart enough to find out how long he has to memorize old information, when to use new data and how to combine old memory with new input [3].

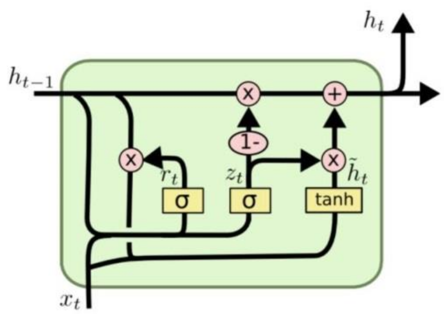

Figure 2. Structure of LSTM neuron

Stock prices are time series of length N, defined as p0, p1, ..., pN-1, in which pi is the closing price in the period i, 0≤i <N. We have a sliding window of fixed size w), and we move it to the right by w so that there is no overlap between the data in all sliding windows.

The RNN model we are building has LSTM cells as major hidden elements. We use the values from the very beginning of the training in the first sliding window W0 to the window Wt at time t:

$$W_0 = (p_0, p_1, \ldots, p_{w-1})$$
$$W_1 = (p_w, p_{w+1}, \ldots, p_{2w-1})$$
$$\ldots$$
$$W_t = (p_{tw}, p_{tw+1}, \ldots, p_{(t+1)w-1})$$

to forecast prices in the next Wt + 1 window:

$$W_{t+1} = (p_{(t+1)w}, p_{(t+1)w+1}, \ldots, p_{(t+2)w-1})$$

Essentially, we are trying to find an approximation function, f (W0, W1, ..., Wt) ≈Wt + 1.

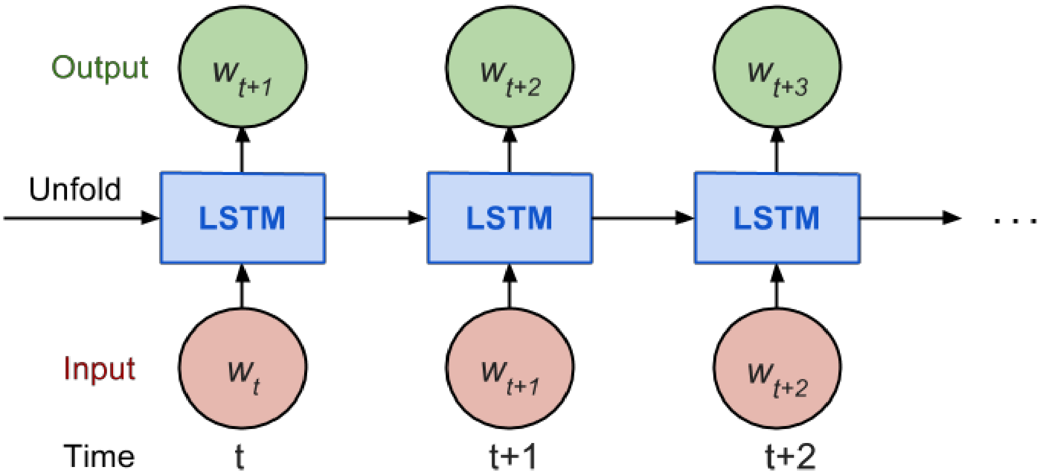

Figure 3. The detailed version of the RNN

Considering how Backpropagation through time, BPTT [4] works, we usually train RNN in a “detailed” version, so we don’t need to back propagate too far in time and increase the complexity of learning.

Before conducting the experiments, a comparison of the popular framework was carried out in order to select the optimal tool for predicting price movements (data are tabulated).

Table 1. Comparison of various frameworks

Framework	**Distributed execution**	**Architecture optimization**	**Visualization**	**Community support**
TensorFlow	High	High	High	High
PyTorch	High	High	High	High
CNTK	High	High	Regular	-
MXNet	Regular	High	Regular	-
Torch	-	High	Regular	Regular
Caffe	High	High	-	-
Theano	-	High	Regular	Regular

To solve the problem, was chosen TensorFlow as a framework for RNN neural networks.

2. Data preparation

The cost of the financial instrument Si-6.16 increases with time, which leads to the fact that most of the values in the test set are out of allowable limits, and thus the model must predict values that it has never seen before. In this situation, the network ceases to behave adequately (Figure 4).

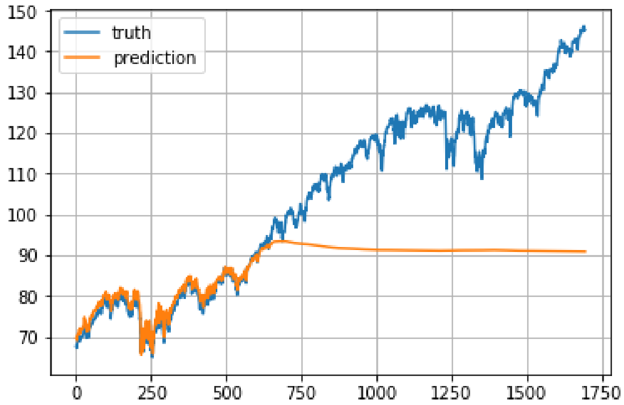

Figure 4. An example where the RNN tries to predict values that lie outside the training data

To solve the problem with the scale, it is necessary to normalize prices in each sliding window. Now the task is to predict the relative levels of change instead of absolute values. In the normalized sliding window **W't** at the moment of time **t** all values are divided by the last unknown price - the last price in **Wt-1**:

$$W_t' = \left(\frac{p_{tw}}{p_{tw-1}}, \frac{p_{tw+1}}{p_{tw-1}}, \ldots, \frac{p_{(t+1)w-1}}{p_{tw-1}}\right)$$

As a result of the data normalization, we get the following price movement:

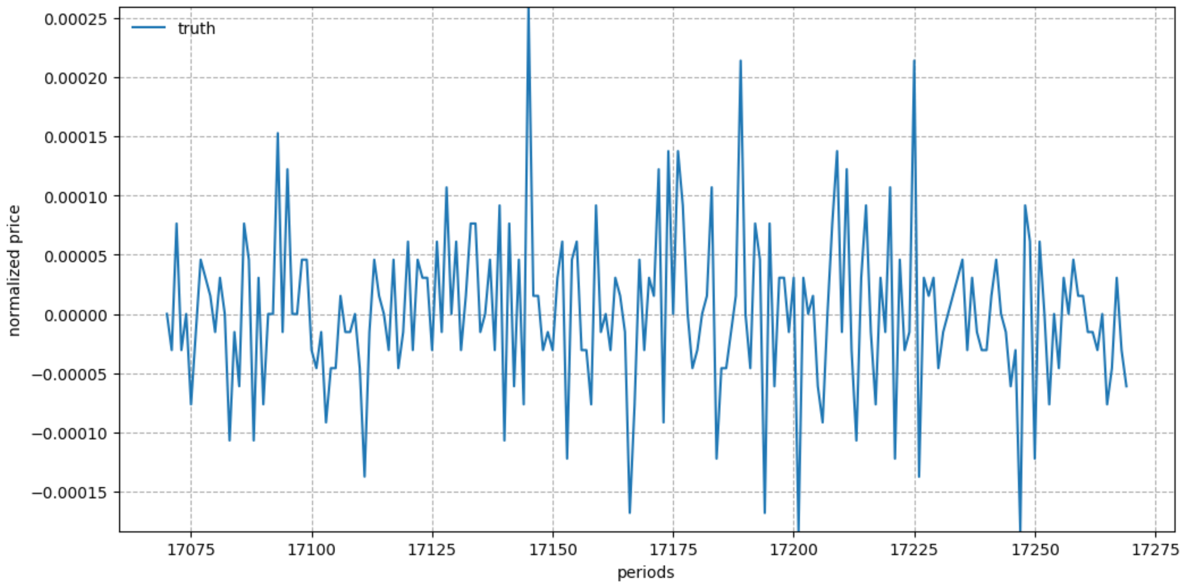

Figure 5. Normalized data Si-6.16

3. Experimental results

When conducting experiments, the following values were used in the TensorFlow settings:

Table 2. Variable environment

Variable	Value
num_layers	1
keep_prob	0.8
batch_size	64
init_learning_rate	0.001
earning_rate_decay	0.99
init_epoch	5
max_epoch	100
num_steps	30

As a result of experiments on forecasting financial instrument Si-6.16, we obtained the following results:

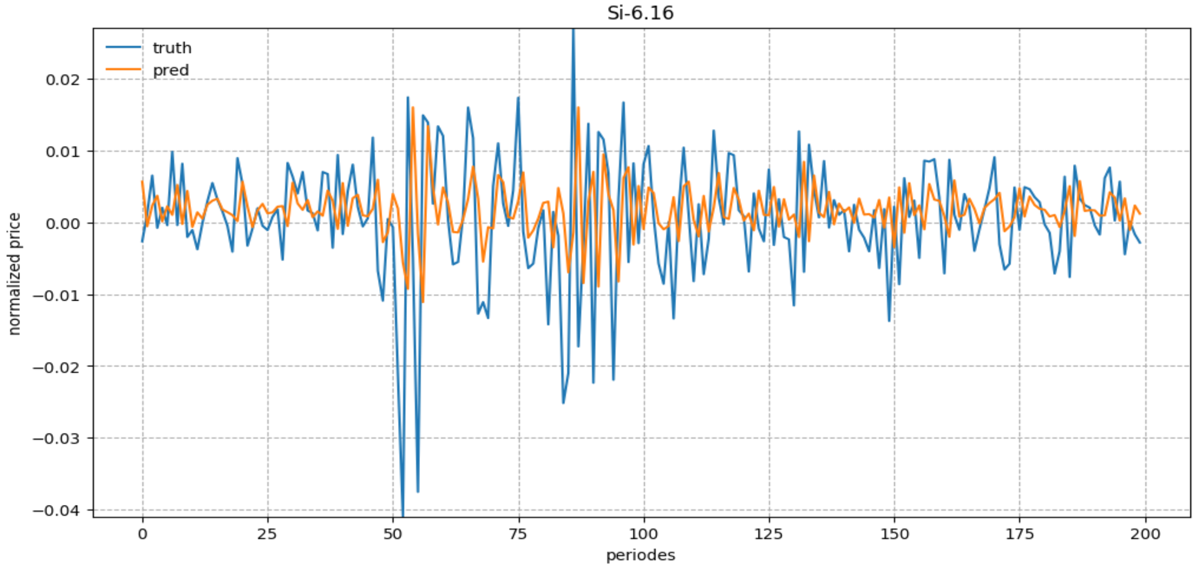

Figure 6. Experimental results

As you can see, the result of forecasting coincides quite well with the change in the price of a financial instrument.

4. Conclusion

Experiments have shown the possibility of predicting the price movements of stock markets using artificial neural networks.

However, at the moment, we can only predict the direction of price movement, rather than a specific value, which increases the probability forecasting.

The accuracy of prediction of the price movement ≈ 62%.

References

[1] Recurrent neural network - https://en.wikipedia.org/wiki/Recurrent_neural_network;

[2] Network with a time delay - https://en.wikipedia.org/wiki/Time_delay_neural_network;

[3] Understanding LSTM Networks. August 27, 2015 - http: //colah.github.io/posts/2015-08-Understanding-LSTMs/

[4] Backpropagation through time - https://en.wikipedia.org/wiki/Backpropagation_through_time