Long Short Term Memory

Kiar Fatah

***Abstract*—To understand the concepts and theory behind the research papers it is important to build a prototype to get experience of the structure that will be used through out the project. The limitation of this goal is that there are many frameworks that allow machine learning and therefore it can be difficult to choose between them. When building the first prototype, the accuracy is not the focus, it is to learn the architecture of the framework, to get the first hands-on experience with machine learning. Hence the built prototype will only have one input, which will be the opening price of an arbitrary stock, and therefore it is expected that the model will not predict very well. The prototype should plot the stock price and the predicted price in consideration to the time series. Implementation of error evaluation equations will also be included to facilitate check if the implementation is reasonably. The algorithm that will be used is Long short term memeory, LSTM, and the framework is Pytorch.**

***Index Terms*—IEEEtran, journal, LaTeX, paper, template.**

Supervisors: Rodrigo Rojas

TRITA number: 1

I. Introduction

For this project two libraries was of interest: Pytorch and Tensorflow, maintained by Facebook and Google respective. No matter which one is chosen the end result will approximately be equal. However, Tensorflow has been developed for a longer time in comparison to Pytorch. Hence there exist more materials and resources around Tensorflow. Furthermore the architecture behind Pytorch is more pythonic and was the reason why it was chosen [1].

The research paper Stock Prediction Using Deep Learning that is going to be reproduced in the future uses deep learning to predict the stock market [2]. However, A Comparative Study of LSTM and DNN for Stock Market Forecasting writes that LSTM outperforms deep learning in terms of weekly predictions [3]. Therefore due to freedom to gain experience, LSTM was chosen as the architecture to build for the first prototype.

II. LSTM

Figure 1 is a flow chart of how LSTM works in essence. An input x_{t-1} is feeded into the neural network and the result is three outputs, two outputs are forwarded for the next prediction to the right and one to h_{t-1}, the actual prediction for x_{t-1}. This concludes that each prediction of the LSTM model is depended on the previous one, in other words the model remembers.

How the data flows through the LSTM is according to the following points: forget, store, update and output. The first stage of the LSTM, seen in figure 2 is to decide what

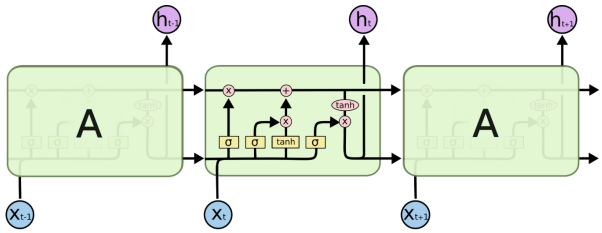

Fig. 1. LSTM flow chart.

information that should be forgotten. This is done by a sigmoid function and the result is

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f). \tag{1}$$

To store the values, the input and hidden state concatenate and

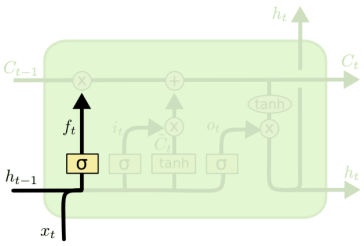

Fig. 2. The forget section of LSTM.

are passes through a sigmoid and tanh function and thereby become, i_t and $\tilde{C}_t$.

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i), \tag{2}$$

$$\tilde{C}_t = \tanh(W_C[h_{t-1}, x_t] + b_C). \tag{3}$$

The update section, as in figure 4, is a multiplication with the

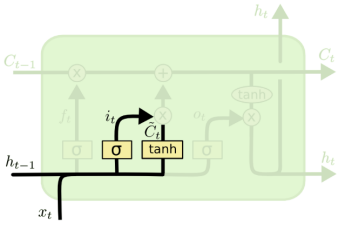

Fig. 3. The store section of LSTM.

forget state and an addition with the store state according to

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \tag{4}$$

Lastly the output happens with the equations

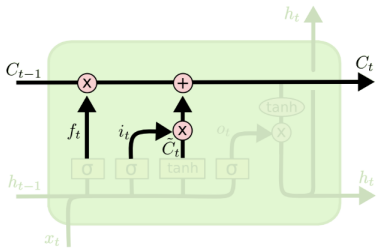

Fig. 4. The update section of LSTM.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o), \tag{5}$$

$$h_t = o_t \cdot \tanh C_t \tag{6}$$

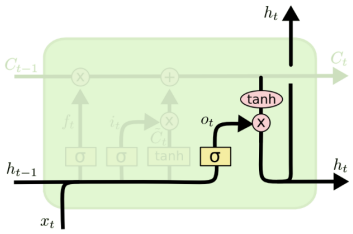

Fig. 5. The output section of LSTM.

III. Material

The following was needed to execute the method:

1) PyTorch,
2) Numpy,
3) Python,
4) Plotly,
5) Google Colab,
6) Personal Computer,
7) Internet.

IV. Method

The method can found in this repository.

V. Summary

This paper contains a description of how LSTM works, implementation and theory. The implementation is limited for one feature, next week it should take in more features.