Foreign exchange rate forecasting based on daily news

Balint David, Budapest University of Technology and Economics

Abstract— **In my research work, I examined the possibilities of forecasting stock market elements, especially the Euro to Hungarian forint (EUR/HUF) foreign exchange rate (FOREX), using deep learning based models that learned from daily news data. Given several number of daily news headline for a given day, what shall be the average bid and ask quote of the forex mentioned above the next day? I collected historical foreign exchange rate data as well as historical daily news headlines, and used them to train different type of neural network architectures. I examined the effectiveness of one-dimensional convolutional neural networks, as well as the potentiality of recurrent neural networks, especially the long-short term memory cells. As my input for the prediction is a set of human-written text, I discovered the world of natural language processing by using pre-trained word embeddings to represent my data. I went through the classical deep learning approach of model design, implementation, train – validation – test as well as hyper parameter optimization to get the most well-performing model.**

deep learning, forex, natural language processing, stock market forecasting

I. Introduction

With the evolution of the stock market there has also been theories created to describe the operational mechanisms of this volatile, ever-changing market. Besides theories, researchers aimed to create such models, that can also be used in practice: with the application of these methods, one can do predictions, which can serve as a base of several trading strategies and speculations.

Independent from the evolution of the financial market, the science of deep learning was also evolving in the last decades. Nowadays it is present everywhere in our life: from our smartphones through the automotive until the healthcare industry. It was inevitable that the potential that this new approach holds, shall also break into the financial market.

In my research work, my goal was apply the latest deep learning technologies on an exact task from the financial market. I chose the international foreign exchange market as the place of my research. This might be the most volatile and liquid field of the financial market, influenced by numerous factors: companies, countries, politics and so many undiscovered, not predictable variables.
It is open to any investor between Sunday and Friday (23:00 CET).

I focused especially on one forex currency pair: euro to Hungarian forint. I aimed to discover whether there is some connection between the daily news of some of the most popular websites and the tendency in the exchange rate. As there is a large amount of historical data of the forex values, and also a significant archive of the daily news headlines on several websites, the prerequisite of a large dataset to train a deep learning model is given.

II. Deep learning architectures

Deep learning architectures are neural networks, where the neurons have a well-defined place in a graph representing the network. These neurons are aligned in separated layers. The topology of the neural networks are given by the neurons and their type of connections to each other. I will present the most popular layer structures in the following paragraphs, that I also used during my research work.

A. Fully connected layers

The most general layer type is the so called “fully connected layer”. All the neurons in neighbor layers are connected to each other, and between the neurons of the same layer are no connections. These layers can be used for any regression or classification problems, but they also occur as a sub-network as a part of more complex deep learning architectures. I used fully connected layers mainly as the final layers in my deep learning models.

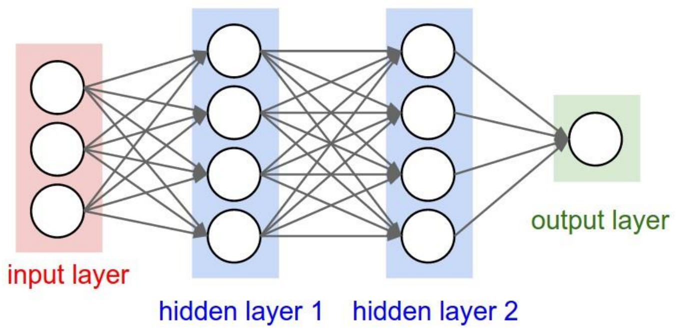

Fig. 1. FC network with 2 hidden layers

B. Convolutional layers

Convolutional layers have a very similar structure to the fully connected layers: the layers consist of neurons in well-defined places, and they are connected to neighbor lanes with trainable weights. The name "convolutional neural network" indicates that the network employs a mathematical operation called convolution. Convolution is a specialized kind of linear operation. Convolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers.

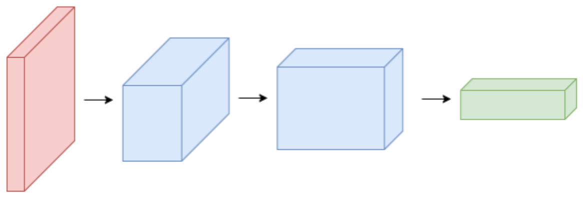

Fig. 2. Convolutional neural network

The first application of convolutional neural networks is connected to Yann LeCun, who trained his LeNet network in 1998, to recognize numbers on post boxes [1]. It has become very popular after 2012, when Alex Krizhevsky and their colleagues used such architecture to win the ImageNet challenge [2]. The real strength of convolutional networks is their ability to extract features ("feature learning") from multidimensional inputs. In case of 2-dimensional inputs (eg. pictures) they can learn shapes, while in case of 1-dimensional inputs, they can learn tendencies. This ability can be used for time series analysis [3] and also for natural language processing [4]. I will also use this ability in my implementation later on to extract information from the daily headlines as transformed word sequences.

C. Recurrent layers

For the fully connected layers, we assumed, that each input (and therefore also each output) is independent, but this approach can not be used to solve some problems. The Recurrent Neural Networks (RNN) were implemented, when such problems were need to be solved, where the sequence of the input data needs to be considered as well. An example of this can be the prediction of the next word in a sentence: the previous words influence the next upcoming word [5]. Such networks are called recurrent, because at a given timestamp, the output is not only influenced by the actual input, but also by the previous ones. As a result, the network is capable of storing data from the past, just as it had memory. Previous researches showed, that the longer the network needs to remember, the lower its performance shall be [6]. The solution for this problem was provided by Hochreiter and Schmidhuber in 1997. They defined a special architecture, called Long Short-Term Memory cell (LSTM), which solves the long-term dependency problem of the regular recurrent networks. Remembering information for long periods of time is practically their default behaviour [7]. During my research work, I also tried to get this advantage by training a network that contained LSTM cells.

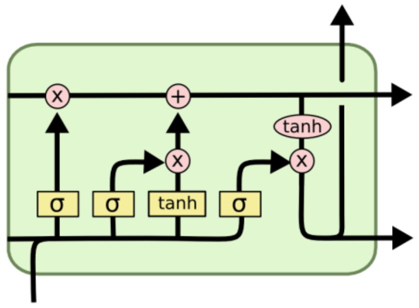

Fig. 3. LSTM cell

III. Previous Solutions

I started my work by looking for the previous solutions from the past: I tried to focus on research areas that were strongly connected to forex data forecasting and natural language processing.

Matthew Dixon and his colleagues tried to predict the change direction of 43 foreign exchange rate in 5 minute blocks. They gave the model 9896 input values. As a result, they reached 42 percent accuracy [8].

Barack Wamkaya Wanjawa observed during his research several shares on the Shanghai Stock Market [9]. He used 4 years of data as input, and created a very simple fully connected architecture of 5 input, 21-21 hidden neurons on 2 hidden layers and one output neuron.

Hengjian Jia made a research about stock market prediction as well, using a more complex, LSTM based neural network architecture. The goal was to predict the Google's share prices on the market, and as a result, he stated, that LSTM is really good at forecasting these prices as well as trends [10].

There has also been research projects, that focused on the connection between the economical daily news and the stock market. Fehrer and Feuerriegel observed several elements on the German stock market based on daily news in 2015. Their goal was a classification task: shall the given share's income increase, remain unchanged or decrease? They used an English database of 8359 daily news between 2004 and 2011. They produced an accuracy of 56 percent [11].

Researchers of the Mumbai University applied several machine learning algorithms (Random Forest, Support Vector Machine, Naïve Bayes) to analyze financial time series' as well as economical newsletters. According to their statements, they had an accuracy of 80% on predicting the effects (positive/negative) of the daily news on several Google stock market elements [12].

IV. Realisation

A. Data acquisition, preparation

For my implementation, I prepared two datasets: the daily news dataset and the forex dataset. For the daily news dataset, I used a partially prepared dataset from Kaggle [13]. It contains daily news headlines from Reddit WorldNews Channel. They are ranked by reddit users' votes, and only the top 25 headlines are considered for a single date (range: 2008-06-08 to 2016-07-01). The data needed further transformation, so that it can be used as an input for my deep learning models. First, I merged all the 25 headlines for a given day. Then, I used several natural language processing techniques: transforming everything to lower case, removing stopwords (very frequently appearing words that has no meaning or would not influence, just distract the deep learning model from learning based on the important words), lemmatizing (transforming words to their “regular” form). Then I tokenized the sentences and applied a padding, so that for each day, the merged headlines has an equal length. One more important step was, that I used the Glove word embeddings provided by the Stanford University [14]. They applied the Word2Vec algorithm on a very large corpus, which resulted in a very large dictionary of words, represented in vectors. These vectors created in such way, that they also represent the context of the words, and therefore representing the words in a more effective way.

The other dataset contained the foreign exchange rates: the target variables for my regression task. I collected this data from the histdata.com website, and it contained bid and ask quotes for the EUR/HUF exchange rate from 2010 until nowadays. For each day, it had several measurement point in milliseconds sampling time, therefore for a given day, I took the average daily bid and ask quotes.

The last step that needed to be done is to synchronize these two datasets: for each day, if daily news were given, I synchronized the forex values from the upcoming day, if those were given. As a result, I had a dataset of daily news and forex values on the next day between 2010-11-15 and 2016-06-30 (1129 measured days).

B. *Deep learning models*

During my research work, I tried two types of neural networks to reach the goal: based on the daily news on a given day, what shall be the average bid and ask quotes for the EUR to HUF exchange rate on the next day?

The first layer of each model was an embedding layer. The purpose of this layer is to transform the tokenized input words in a sentence into a word vector representation. As I mentioned before, I used the already prepared Glove word embeddings for this transformation. Normally, one would allow the embedding layer to learn the word embeddings from the corpus from itself, but in my case, I merged the headlines for a given day. This means, that the embedding layer would learn incorrectly some word contexts, and therefore, after initializing the embedding layer with the proper word embeddings, I set this layer to “frozen”: it is not trained just like the other layers, it behaves basically as a lookup table: for a given word, it provides the Glove word vector representation. It is important to mention, that due these constraints, the upcoming words in the sentences that are not present in my training corpus shall be parsed as “not a word”. This is a weakness of my approach, but the negative effects can be reduced if one has a large train dataset.

For the next part of the serial architecture, I used two types of layers: 1 dimensional convolutional layers and LSTM layers. The goal of the convolutional layer was to learn tendencies in the headlines. As the vectors are now represented with 1x100 vectors, the convolutional filters are learning from the dimensions of these vectors. The other approach with the LSTM layer learned considering the previous words in the headlines.

At the end of each architecture, I used fully connected layers to transform the output into 2 neurons: one for the bid and one for the ask quote.

The [4] and [5] pictures visualizes my two type of deep learning architectures to solve the given problem.

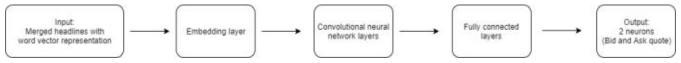
Input:
Merged headlines with word vector representation
Embedding layer
Convolutional neural network layers
Fully connected layers
Output:
2 neurons
(Bid and Ask quote)

Fig. 4. CNN based neural network architecture

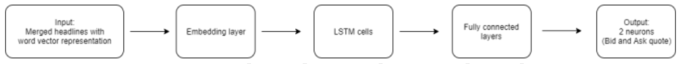

Fig. 5. LSTM based neural network architecture

C. *Train, validation and test*

To be able to validate and test my models, I splitted my datasets into train, validation and test datasets. As a result, I had a train dataset of 846, a validation dataset of 170, and a test dataset of 113 inputs.

As I mentioned before, I used the word vector representation for my daily news data – the inputs for my regression model. The word embeddings are already in an ideal format: vectors containing numbers around 0 and 1: ideal as an input to the deep learning model. I made it sure not to leak any information from the validation and test datasets to the model (eg. initializing the tokenizers and the embedding layer with the train dataset only).

To transform the foreign exchange rate data, I used Min-Max scaling to transform the data between 0 and 1.

As most of my models started to overfit quite early during the training, I used early stopping and model checkpoint callbacks: if the validation loss did not increase for a given number of epochs, the training stops, and the model with the lowest validation loss shall be saved and used for testing.

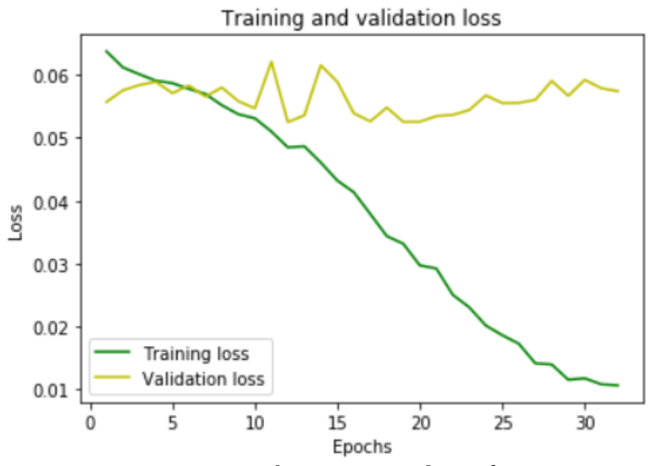

Fig. 6. Example training and overfitting

D. Hyper parameter optimization and final results

After the implementation of the models and some testing, I started the process of hyper parameter optimization: fine tuning all possible hyper parameters, so that we get the optimal solution in the terms of validation loss, model complexity, training time etc. The [1] table summarizes my hyper parameters for both models and the values that I tried with a grid search method.

Hyper parameter	Value range
Embedding dimension	[100, 200, 300]
Number of Conv1D layers	[1, 2, 3, 4]
Number of filters	[64, 128, 256]
Filter size	[5, 10, 20]
LSTM size	[128, 256]
Number of Dense layers	[1, 2]
Dense layer neurons	[64, 128, 256]
Learning rate	[0.0001, 0.001, 0.01]
Batch size	[64, 128, 256]
Optimizer	Adam, SGD

1. table, hyper parameters

There were some parameters, that greatly influenced the validation and test loss of my model. For example, increasing the learning rate influenced the losses significantly. From these parameters, I obviously chose the one value which resulted the best loss. There were also parameters, that did not really influence the losses, e.g. the embedding dimension. From these parameters, I chose the value which produced the less complex architecture (e.g. only 100 embedding dimension). Surprisingly, both the LSTM and the convolutional network (with the best hyper parameters) resulted almost the same validation and test loss, which is summarized in the [2] table.

Network type	Validation loss	Test loss
Convolutional nn.	0.05251	0.05787
LSTM	0.05721	0.06124

2. table, validation and test losses

As the results are a bit better for the convolutional neural network, and the training time is significantly lower than for the LSTM based model, I would definitely choose the CNN based model. The hyper parameters of the most well-performing (and ideal) model is summarized in the [3] table.

Hyper parameter	Value range
Embedding dimension	100
Number of Conv1D layers	3
Number of filters	128
Filter size	5
Number of Dense layers	2
Dense layer neurons	128 (first) and 64 (second)
Learning rate	0.0001
Batch size	128
Optimizer	Adam

3. table, final hyper parameters

V. Conclusion

My CNN based deep learning model produced a good result on the test dataset, but the performance could be increased by extending the train dataset, which also extends my non-trainable embedding layer, and therefore reduces the occurrence of skipping words that are not present in the training dataset. In the future, I would like to try different types of daily news datasets: global news, only Hungarian news, daily news with different theme (financial, political, sport etc) and check the performance of the differently trained models.

During my research I achieved my goal, that is not certainly connected to the actual problem that was needed to solve: I learned the fundamentals of deep learning, got a hands-on experience on the most popular deep learning architectures and their applications in real-world problems, and therefore received a solid basic knowledge that I can further improve in the future.

References

[1] Y. LeCun, L. Bottou, Y. Bengio és P. Haffner, „Gradient-Based Learning Applied to Document Recognition," Proceedings of the IEEE, 86. kötet, 11. szám, pp. 2278 - 2324, 1998.

[2] A. Krizhevsky, I. Sutskever és G. E. Hinton, „ImageNet Classification with Deep Convolutional Neural Networks," in NIPS 2012: Neural Information Processing Systems, Lake Tahoe, 2012.

[3] A. Siripurapu, „Convolutional Networks for Stock Trading," Stanford University, Stanford, 2015.

[4] X. Zhang és Y. LeCun, „Text Understanding from Scratch," New York University, New York, 2015.

[5] C. Olah, „Colah's Blog," 2015.08.27.. http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[6] J. Schmidhuber, „Untersuchungen zu dynamischen neuronalen Netzen," Technische Universitat München, München, 1991

[7] Y. Bengio, P. Simard és P. Frasconi, „Learning Long-Term Dependencies with Gradient Descent is Difficult," IEEE Transactions on neural networks, 5. kötet, 2. szám, pp. 157-166, 1994.

[8] M. F. Dixon, D. Klabjan és J. H. Bang, „Classification-Based Financial Markets Prediction Using Deep Neural Networks," Stuart School of Business, Northwestern University, Chicago, Evanston, 2016.

[9] B. W. Wanjava, „Predicting Future Shanghai Stock Market Price using ANN in the Period 21-Sep-2016 to 11-Oct-2016," University of Nairobi, Kenya, 2016.

[10] H. Jia, „Investigation Into The Effectiveness Of Long Short Term Memory Networks For Stock Price Prediction," Colyton Grammar School, Colyton, 2016.

[11] R. Fehrer és S. Feuerriegel, „Improving Decision Analytics with Deep Learning: The Case of Financial Disclosures," University of Freiburg, Freiburg, 2015.

[12] J. Kalanyi, H. N. Bharathi és R. Jyothi, „Stock trend prediction using news sentiment analysis," University of Mumbai, Mumbai, 2016

[13] https://www.kaggle.com/aaron7sun/stocknews

[14] https://nlp.stanford.edu/projects/glove/