Stock market prediction with neural network

Részvényárfolyam előrejelzés neurális hálóval

Bence Babics, Osváth Barnabás, Tamás Ferenc

In this paper we introduce our deep neural network model that we use to predict short term trend movement on the Forex trading market, the movement of USD-EUR exchange rate to be precise. We've used technical analytical data, prices, and other indicators that could be calculated from them to train this model. A platform has been implemented that automatically obtains short term (hourly, daily) training data, and trains the network to predict the exchange rate for the following short time interval.

Ebben a tanulmányban bemutatjuk a mély neurális háló modellünket, amit a Forex kereskedelmi piac részvényeinek rövidtávú mozgását igyekszünk előrejelezni, konkrétan az USD-EUR árfolyamát. Használtunk technikai analitikus adatot, historikos árfolyamokat, és egyéb indikátorokat, amiket ezekből ki lehetett számolni a modell tanításához. Létrehoztunk egy platformot, ami rövid időközönként (óránként, naponta) a háló tanításához szükséges adatot gyűjt be, majd tanítja azt, így a következő hasonlóan rövid intervallumra előrejelezhető a részvények árfolyama.

Introduction

In this project we aimed to predict trend movements in Forex trading markets, using with a Long Short Term Memory network.

Literature overview

For the purpose of predicting stock market rates with machine learning algorithms several studies have been written. One of the most promising was (Chen, Zhou, & Dai, 2015) who used LSTM network in order to predict the stock return. The aim of their study was to take advantages of memory cells which were able to capture a grasp of the structure of the timeseries dynamically. Analyzed stocks were based on the China market which were transformed into 10 learning features. As they mentioned, 4 types of stock data could be differentiated:

1. Historical stock data, as it can be observed from the market

2. Technical analysis data, which is calculated from the historical data such as the moving average

3. Historical prices of market indexes and other related stocks

4. Fundamental data about the market like oil price or GDP

Prediction accuracy of their network was able to reach 27.2% after the learning process.

We have selected the foreign exchange part of the market. A great introductory article was written by (Galeshchuk, 2015). Their study created a time-series prediction using forex data by a neural network. They challenged (Meese & Rogoff, 1983) study who stated that an estimation model that is better than random walk doesn't exist for the purpose of predicting exchange rates. They have used a fully connected neural network to capture non-linear relationship between features of the data. In this case they have used exchange rates of 3 different currency pairs as feature and the daily and monthly and quarterly rates were predicted. They found that application of the neural network for short term predictions provided good accuracy.

During (Lahmiri, 2015) study we were able to acquire and store high frequency forex data from the market using the combination of Google Cloud platform and Oanda's REST API. Lahmiri also wrote a study which analyzed intraday stock data in order to predict prices by variational mode decomposition. In the study it was highlighted that have not made any rule to estimate the number of the neurons as a result they arbitrary chosen it as double of nodes in the input layer. Before the learning they normalized the data between -1 and 1 to reach better convergence. They used 4 weeks American based stocks' prices. With their new multiresolution technique, a performance of the fully connected network could be increased.

Another study which aimed to predict from high frequency data was written by (Levendovszky & Kia, 2012). They also aimed that neural network is able to capture non linearities in data and they used it for predicting forex data. Their approach was to trading by estimating the future price of the stock by using nonlinear predictor. For that purposes a fully connected neural network was created and the underlying data was EUR/USD real time data. It was documented that using their strategy on intraday data they were able to

reach more than 1% profit in one month.

Forex exchange rate prediction with Neural Network was created by (Evans, Pappas, & Xhafa, 2013). Their study was focused on the 3 most traded currency pairs GBP/USD, EUR/GBP, EUR/USD. They made a statistical test on their dataset and they found that forex currency rates are not randomly distributed. Furthermore, they highlighted 2 additional results: their model was able to achieve 75.2% prediction accuracy and using optimal trading strategy resulted 23.3% annualized net return. It was mentioned that during the feature selection using both bid and ask prices are redundant and unnecessary. Choosing one from them would be enough. In order to determine the number of the layers they used (Kaastra & Boyd,, 1996) work which detail that using one or two hidden layers are enough to estimate a smooth bounded function.

Data

Data was collected by Google Cloud platform and the following currency pairs were analyzed: EUR/USD, USD/CHF, GBP/USD and AUD/USD. Describe statistics from the data can be seen on the following table.

	name	AUD_USD	EUR_USD	GBP_USD	USD_CHF
ask	count	25826.000000	96923.000000	64002.000000	28269.000000
	mean	0.682721	1.106783	1.286681	0.991697
	std	0.002573	0.005263	0.005478	0.003006
	min	0.677030	1.098970	1.276970	0.982380
	25%	0.680460	1.101900	1.283650	0.989460
	50%	0.683410	1.105450	1.285420	0.990310
	75%	0.684910	1.112280	1.288160	0.994500
	max	0.689280	1.116350	1.318830	0.997970
bid	count	25826.000000	96923.000000	64002.000000	28269.000000
	mean	0.682582	1.106659	1.286496	0.991528
	std	0.002571	0.005262	0.005480	0.003002
	min	0.676900	1.098850	1.276820	0.982250
	25%	0.680320	1.101780	1.283490	0.989290
	50%	0.683260	1.105320	1.285200	0.990150
	75%	0.684770	1.112160	1.287980	0.994350
	max	0.689150	1.116210	1.318640	0.997790

Data collection method

The data is provided by Oanda through their v20 trading engine. This engine has a convenient REST API, and Oanda also provides a Python utility library for consuming it easily.

We used the said Python library to listen for events of stock changes. Oanda pushes these events every few seconds while the stock market is open.

Upon receiving an event, we store its details in an SQL database. The details include a stock's important values such as ask and bid prices, as well as their liquidity and the event's date itself.

The script that we wrote for data collection was hosted on a Google Cloud Compute instance, and its source code is available in the project's GitHub repository.

Feature engineering

In fact, our currency pairs have a correlation with each other that we want to use in order to reach higher pricing accuracy. Correlation matrix of our currencies can be seen in percentage form in the following table.

|	EURUSD_ask	USDCHF_ask	AUDUSD_ask	GBPUSD_ask
EURUSD_ask	100.00	-4.58	12.10	73.27
USDCHF_ask	-4.58	100.00	79.49	-59.90
AUDUSD_ask	12.10	79.49	100.00	-37.53
GBPUSD_ask	73.27	-59.90	-37.53	100.00

More intuitively correlation can be observed by line charts too.

Furthermore, in our training set we added these currencies as feature. Because of the act that data was collected as real-time data prices were depend on the trading. It means data received randomly for different currencies. In this case if we want to merge these data by date it would result that not so many matchings we have. In order to avoid it every observation was aggregated by date in every 15 second. It results that dates have a common date period and it can be merged. Moreover, this aggregation method made possible to determine a low and high values in the range. These values are required

Additionally, 3 more features were added into our dataset. First one was moving average which often used in technical analysis. It is a trend-following method which is able to identify the trend direction. It was calculated from historical timeseries data.

Another technical analysis indicator was MACD – Moving Average Convergence Divergence. It is calculating from 26- and 12-period exponential moving average. When two lines are crossing each other often trend change will be followed on the market. MACD is the difference of the two-exponential moving average values. A frequency chart highlights well visually when the trend changed.

Last, the Bollinger Band was calculated for the stock that we wanted to predict. This technique calculates two standard deviation of the moving average. It tries reflects an upper and lower bound of the stock changes. Most of the cases stock changes between the two limit lines.

Neural Network Architecture
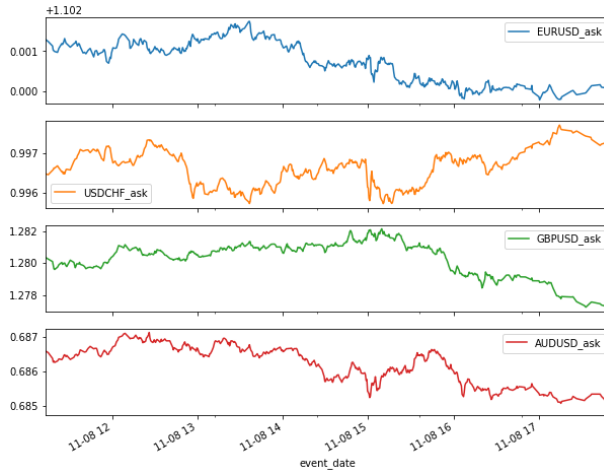
Methodology

In our final model we had two separate models. One of them try to predict the ask- and other one the mid-price changes. Network which predicted mid-price used only mid prices from the market. It also true for ask prices.

Following currency pairs were feed into the network as feature: EUR/USD, USD/CHF, GBP/USD, AUD/USD moreover, exponential moving average for 26 and 16 days, MACD and Bollinger Bands from EUR/USD prices. Before the learning process data was normalized between 0 and 1.
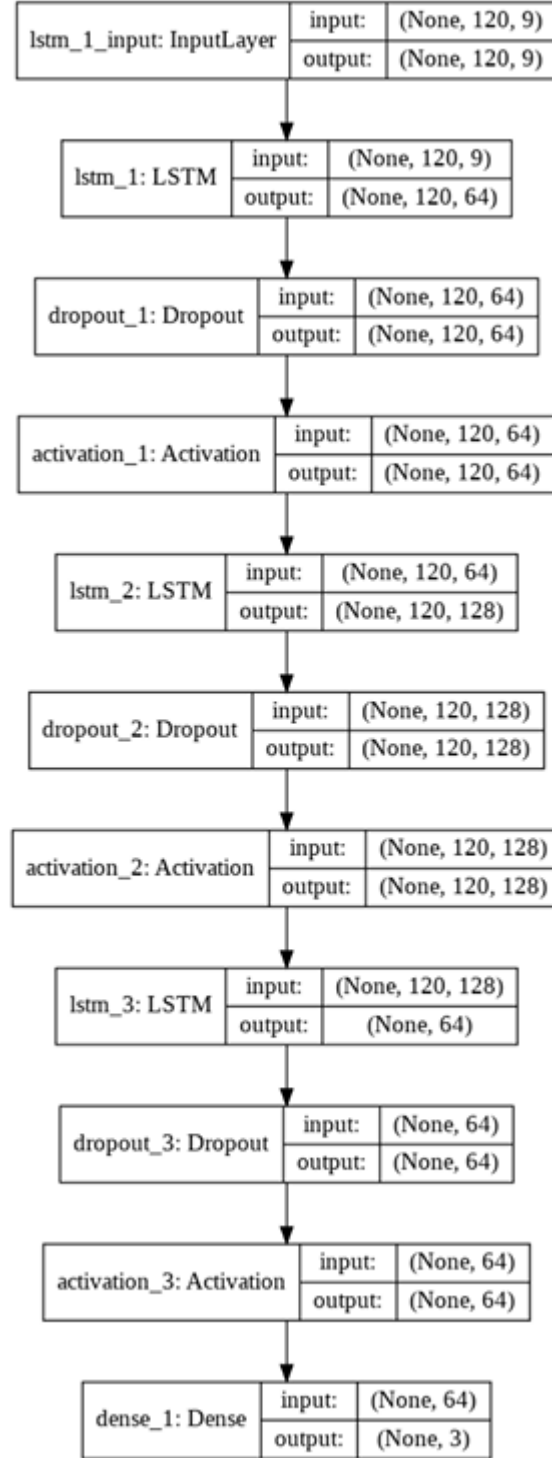
We transformed our timeseries prediction task into a classification problem. As a result, our target variable can be determined with the following function:

$$target(x_t) = \begin{cases} 1 \ if \ x_{t+s} - x_t > 0 + \varepsilon \\ -1 \ if \ x_{t+s} - x_t < 0 - \varepsilon \\ 0 \qquad else \end{cases}$$

Where ε is a predefined threshold which expands the limit of the stagnation. Another parameter is s which determine how many timesteps we look ahead in order to measure if stock moved upwards (1) or downwards (-1), or it stagnated (0). Moreover, x determines which currency change we want to predict.

In our analysis we used EUR/USD currency pairs for both bid and ask prices. Furthermore ε were determined as x_{t+s} *0.05 which means 5 percent of the future price.

The details of the net were enhanced by hyperparameter optimization. So, the activation function on the recurrent LSTM layers can be followed on the figure, and on the output layer it's 'softmax' to get probability, so the loss function is naturally categorical-cross entropy. We applied a dropout on the LSTM layers to avoid overfitting. The optimization

algorithm is Adam.

Hyperparameter optimization

For hyperoptimalization hyperas was used. The number of nodes in the layers had been chosen from the possibilities of 64, 128, 256. The activation could have been 'relu', 'leaky-relu', and 'swish'. The dropouts were chosen from the interval of [0, 0.5], and the possible optimizers were 'rmsprop', 'adam' and 'sgd'. In the optimal net the dropouts are rather high, the activation function is relu and the optimizer is Adam with a learning rate of 0.0001.

Architecture

The entire back-end of the project is written in Python and is hosted on a Google Compute Engine instance.

We use independent scripts for collecting data, training a neural network model, and predicting with it. These scripts can be run as services, e.g. training the model at certain intervals, can be used manually, or can be triggered through the Django web service.

There is no communication between the independent services, the only source of triggers is the configuration during deployment or the common database which is polled (or just used) by each service (This doesn't apply for the tasks that are run by the Django web service).

The data is stored on a Google SQL (MySQL) instance with the following entities:

- **Stock:** The raw stock data from Oanda.

- **Model:** The trained neural network models.

- **Prediction:** The stock predictions.

We also made a VueJS (originally started as Angular) front-end SPA for visualizing the data, and also triggering predictions, and controlling other behaviors in the future. The continuous integration and deployment is done with Drone, services are automatically deployed from the GitHub repository into Docker containers.

Results

Our best results so far:

Loss	Train acc	Val loss	Val acc
0.5066	0.7825	0.5353	0.7500

On the test dataset the final accuracy is 0.66.

Plans for the future

A trading strategy could be implemented on top of this model, that automatically trades different currencies according to its predictions of trend movement.

A way to enhance our net would be the additional sentiment analysis of relevant news (this) or tweets (this) to provide us fundamental analysis and then to combine it with our net that already utilizes technical analysis. Another path we could take would be to expand our data with stocks as well (this). However, it is rather hard to find a data source with the same high resolution as our source of forex data.
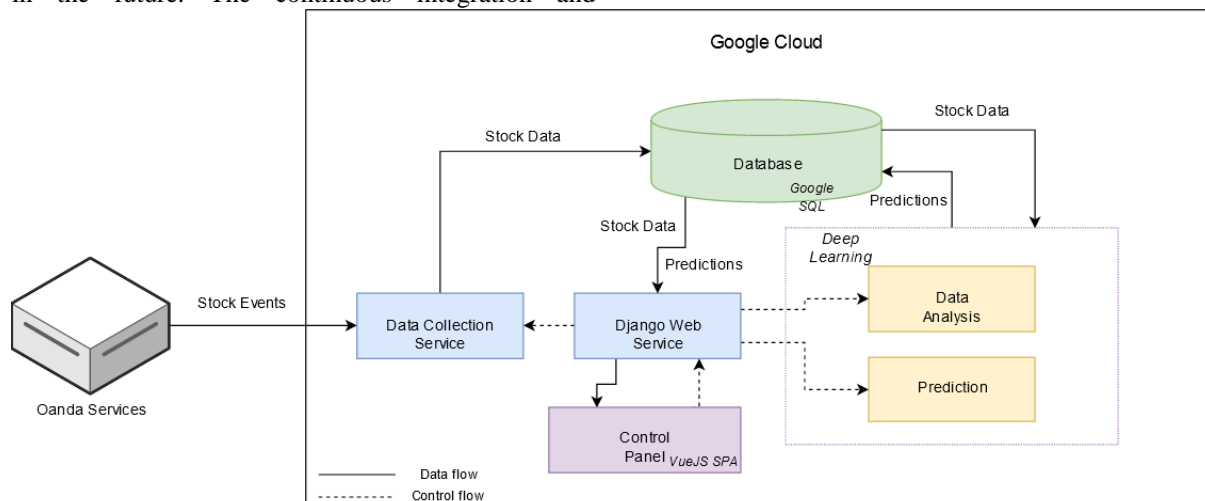
References

Chen, K., Zhou, Y., & Dai, F. (2015). A LSTM-based method for stock returns prediction: A case study of China stock market. *IEEE International Conference on Big Data (Big Data)* (old.: 2823-2824). IEEE.

Evans, C., Pappas, K., & Xhafa, F. (2013). Utilizing artificial neural networks and genetic algorithms to build an algo-trading model for intra-day foreign exchange speculation. *Mathematical and Computer Modelling*, 1-18.

Galeshchuk, S. (2015). Neuralnetworksperformanceinexchangerateprediction. *Neurocomputing*, 1-7.

Kaastra, I., & Boyd,, M. (1996). Designing a neural network for forecasting financial and economic time series. *Neurocomputing*, 215-236.

Lahmiri, S. (2015). Intraday stock price forecasting based on variational modedecomposition. *Journal of Computational Science*, 23-28.

Levendovszky, J., & Kia, F. (2012). Prediction based – high frequency trading on financial time series. *Electrical Engineering and Computer Science*, 29-34.

Meese, R., & Rogoff, K. (1983). Empiricalexchangeratemodelsoftheseventies:dothey fit outofsample? *Journal of International Economics*, 3-24.