Foreign Exchange Rates forcasting with LSTM

Huy Phung, Tashi Choden, Sahil Pasricha
University of Konstanz

February 2019

Contents

1 Abstract

Foreign Exchange (abbreviation *Forex* or simply *FX*) Market is the decentralized market for currency investment. Forex market is the second most important market, after stock market. Supply and demand in the market determine Forex rate, in which a pair of currency can be exchanged. Forex Rates has been studied in econometrics as a financial timeseries. The purpose of studying Forex rates is to explain the market behaviour or forecast future prices.
In our project, we use statistical models and deep learning model to predict the future rates of one step ahead. Our goal is to compare the effectiveness of LSTM and statistical models (ARIMA and VAR) as timeseries models, in both accuracy and performance.

2 Problem Description

2.1 Forex rates

Foreign Exchange rates (short Forex rates) are decided solely by support and demand of Forex market. Each rate represents the price to buy or sell a pair of currency (e.g. EURUSD) at the moment. The price to buy is called Bid price; the price to sell is called Ask price. The difference between Bid price and Ask price is called Spread. In this project we consider only the Bid price. However, if both Bid and Ask (and therefore Spread) were available, our analysis would be more precise.
Forex brokers update rates according to the market within milliseconds by standardized FIX (Financial Information eXchange) protocol. The time interval between each FIX message to update rates is not uniform; it may varies from a millisecond to few seconds. Therefore, the timeseries is of continuous time step. In order to simplify our analysis, we convert it to a data form that has discrete, uniform time step, while still keep important information.
One possible way to do so is to format the rates into OHLC format. This approach is widely used in financial technical analysis. We partition the timeseries into intervals of uniform time length t. For each interval, we keep only 4 rate: the first (*open*), the last (*close*), the maximum (*high*), the minimum (*low*). Since the time intervals are uniform among dataset, we have the desired discrete, uniform time step for analysis.

2.2 Prediction

In this project, we concern about the prediction of future Open, High, Low, Close prices. Other features, either originally exists (volume) or later added (mean, median, momentum), are only considered as supporting features. These features are only used for prediction of OHLC features.
The problem we are trying to solve in this project is declared as follow: given history data in OHLC form of Forex rates $x_0, x_1, \ldots, x_k$, predict future rate of *one* step ahead, x_{k+1}.

2.3 Dataset

Since acquiring real-time data is expensive, due to the fact that most FIX data providers requires subscription contract, we have searched for a Cite published a dataset consists of OHLC of BID price (lack of ASK price) of EURUSD rates from 2010 to 2016. Time interval for OHLC value is uniformly set to 15 minutes. The dataset

2.4 Model selecion

2.4.1 Akaike Information Criterion (AIC)

For statistical model selection, we use Akaike Information Criterion (AIC). As we can see, AIC penalize a model by its number of parameters. The number of parameters of a neural network based model is at least total number of elements contained in all weights in its topology. Therefore, we should not use AIC as a measure of performance between statistical models and deep learning models.

$$AIC = 2k - 2\ln(\hat{L})$$

2.4.2 Root Mean Squared Error (RMSE)

Root Mean Squared Error is widely used to measure the difference between values predicted by a model and the actually observed values. Given y represents the actually observed values and $\hat{y}$ represents the values predicted, $RMSE$ is given by:

$$RMSE = \left(\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \right)^{\frac{1}{2}}$$

Root Mean Squared Error shows absolute difference between y and $\hat{y}$. However, since it does not take the range of possible values into account, it would

be difficult to intepret the $RMSE$ result without knowing the possible range of predicted and actual values.

2.4.3 Mean Absolute Percentage Error (MAPE)

In order to measure the difference between predicted values and actual values with regarding to the scale, we use Mean Absolute Percentage Error (MAPE)

$$MAPE = \frac{100\%}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right|$$

3 Statistical Models

3.0.1 Autocorrelation and discrete white noises

Let $\{x_t\}$ is a timeseries with $\mu = \mu(t)$ is its mean and $\sigma = \sigma(t)$ is its variance. **Autocovariance** of lag k is defined as $C_k = E[(x_t - \mu)(x_{t+k} - \mu)]$. **Autocorrelation** A time series $\{w_t\}$ is said to be*discrete white noises* if $\{w_t\}$ has zero mean and no autocorrelation between any of its values. Formally

3.0.2 AR(p)

3.1 ARIMA

The first model we consider is similar to linear regression to some extent. $ARIMA(p, d, q)$ model consists of three components $AR(p)$, $MA(q)$ and intergrated series I of order d. $ARIMA(p, d, q)$ rationale is that we find a linear relation between the value at time t and certain *lags* before it. More detailed description of ARIMA and VAR can be found at

3.1.1 MA(q)

3.1.2 ARIMA(p,d,q)

Detailed explanation of the model can be found at [**?**]

3.1.3 Parameters selection

It is important for ARIMA model that we select the proper parameter (p, d, q) so that it covers all the past values which has effect on the current value. In order to do so, we first look at survey the ACF and PACF

plot. For $AR(p)$ model , we find the furthest lag with significant autocorrelation in ACF plot. For $MA(q)$, we find at which lag the autocorrelation start to decay.

3.1.4 Result

3.2 VAR(p)

3.2.1 Model description

VAR is applied to multivariate data. It is similar to AR(p) model. However here we consider

3.2.2 Parameters selection

Parameters selection for VAR consists only testing a range of lag with AIC.

3.2.3 Result

4 Deep Learning Model

4.1 Recurrent Neural Network

Recurrent Neural Network (RNN) is introduced by [2] to process sequential input. In RNN, each state connects to the followed state to form a directed graph. The structure of RNN makes it capable of handling sequential data with temporal dynamic behaviour.

One problem that may occurs within RNN is *vanishing gradient*u

4.2 Long-Short Term Memory

Hochreiter and Schmidhuber (1997) [1] presented a network architecture to solve both vanishing gradient and exploding gradient problem from RNN Cummins presented

4.3 Proposed Network Topology

Our survey on the data implies that significant autocorrelation may appears even further than 20 timesteps into the past. In our proposed network topology, we

add more hidden units into each LSTM layers, so that the network can learn from data points of higher lags.

4.4 Univariate and Multivariate

We consider using LSTM network in two models. In the first usecases, we train the network to predict future values only by giving to history lags of one timeseries (univariate). For example, we train the network to predict 3 Open prices in the future given only the Open prices in the past. This usecases is designed to compare the performance with ARIMA(p, d, q) model.
In the second usecase,

4.5 Training and Validation

As we can see from training progress, after the first epoch, the loss within the network does not decrease any more, meanwhile the

4.6 Results

4.6.1 Single Variate

4.6.2 Multivariate

5 Implementation

6 Conclusion

As the results have shown, ...

However, computational effort spent for training LSTM deep learning model is much higher than ARIMA and VAR. Furthermore the, adding components to neural network topology results in higher computational cost.

References

[1] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with LSTM. 1999.

[2] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.