

Download My Powerful Weekly Pivot
Trading Strategy FREE!

Name

Email

Download 

Home

About Me

Privacy Policy

Contact

Trading Challenge ▼

Courses ▼

Member Login

FTC

Home

Home

Day Trader Blog

Can We Predict GBPUSD Flash Crash With GRU & LSTM Model?

Hassam

0 Comments

March 13, 2017

 Tweet  0 

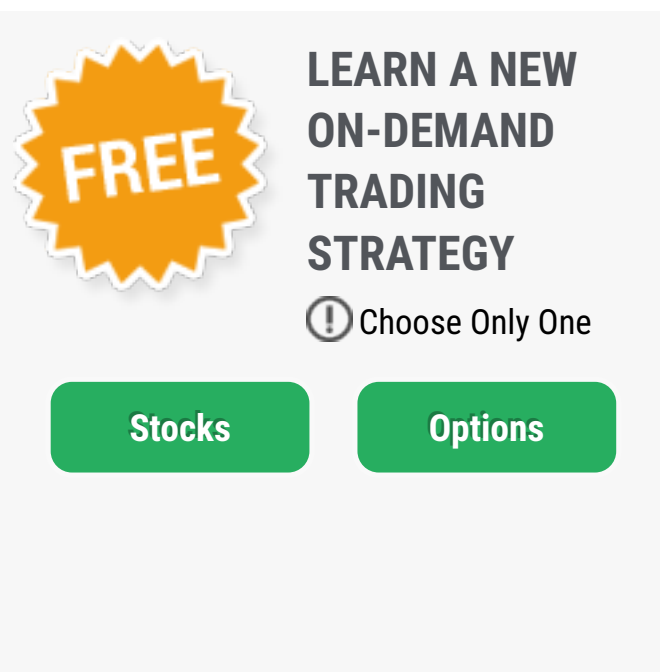
In this post we are going to construct first a Gated Recurrent Unit

(GRU) neural network using Python. Then we will construct a Long Short Term Memory (LSTM) neural network and try to make predictions. We will use GBPUSD daily data. The last observation is the GBPUSD Flash Crash in which GBPUSD fell more than 1000 pips then recovered. We will see if GRU or LSTM can predict GBPUSD Flash Crash just one day before it happened. So we will make predictions and then check if we have good predictions. If we get good predictions and the time to train the GRU or LSTM is fast then we can also build an intraday trading system using a GRU or a LSTM NN. In the past few years extensive research has been done in neural network and deep learning by the big companies like Google, Facebook, Amazon, Apple etc. [**Join our Million Dollar Trading Challenge today and trade forex with us daily.**](#)

Today when you visit Facebook, you should know that it uses a lot of Artificial Intelligence and deep learning in predicting what you will do on Facebook. Similarly Google depends a lot on deep learning now a days. Google has incorporated deep learning artificial intelligence into its algorithm. Deep learning is being hailed as the technological revolution of this century. Soon we will be having self driving cars thanks to deep learning. So let's try to build a deep learning day trading system. [**Join our Million Dollar Trading Challenge II and trade binary options daily**](#) with us as well.

We will be using Python as our scripting language to build the Gated Recurrent Unit (GRU) and the Long Short Term Memory Neural Network. We will be using keras with tensorflow at the backend. In a future post we will build a few trading systems using tensorflow. You should know python. You should have tensorflow and keras libraries installed. I am using Rodeo as my IDE. Rodeo is much better than Spyder. If you don't know python then I urge you to learn it. it is not

difficult. Future belongs to algorithmic trading. [You can also try our Million Dollar Trading Challenge EA](#). Watch the video below that introduces you to python and how to use it for predicting stock prices.



FREE

**LEARN A NEW
ON-DEMAND
TRADING
STRATEGY**

ⓘ Choose Only One

Stocks **Options**

After watching this video you should know this fact that college kids are doing algorithmic trading from their dorm rooms. Days of manual trading are coming to an end. Today algorithms are trading against each other on Wall Street. In the case of GBPUSD Pound Flash Crash that took place last year, nobody was expecting it. It happened during the Asian Market Hours when most traders are least active. A rogue algorithm is being blamed for crashing GBPUSD 1000 pips in just 1 minute. So you can see how fast markets move now a days. But if you look at the weekly chart you can see a strong bearish weekly candle which is a strong signal that GBPUSD will be a sell next week as well. This is how we technical traders trade by looking at the candles. We can be wrong. We alleviate this by entering with a small stop loss on a lower timeframe. As a manual trader I trade on H4 chart. I take the market direction from the weekly and daily chart. It is simply too difficult for me to trade on the 1 minute chart. It is humanely

impossible to react in such a short time. But algorithmic trading systems are programmed to react on each tick. So we will see if an algorithmic trading system can beat our manual trading system. [Did you read the post on how to download intraday stock data from Google Finance?](#) So let's start our journey.

What Is A GATED RECURRENT UNIT (GRU)?

As a forex trader, we know that past price patterns can be used to predict the future price. This is the basis of technical analysis. We call it charting. We look at the chart which is all history and try to predict future price. Does it work? Sure it does. There are many traders who made fortunes just by looking at charts. As said charts are just showing the past price. By looking at the charts we are looking at the past price and attempting to predict the future price. [Did you watch this documentary on the day in the life of a currency trader?](#)

In statistics we say this that present value of price is serially correlated with the past values. Different models have been developed like the Autoregressive (AR) and Autoregressive Integrated Moving Average (ARIMA) models that take into account serial correlation of a time series and then try to predict the future. But these linear models are not good at predicting price in the forex market as well as price in the stock market. We need a non linear model that has memory.

Memory means it remembers the past and tries to use that memory in predicting the future. Gated Recurrent Unit is exactly that sort of a thing. It is a non linear model that has memory. Since we know price depends on previous past values, we think that trying a GRU model can be good. We will see in the end how good was our hunch about GRUs. GRUs are recurrent neural network. Getting bored? Watch this documentary on [the history of trading in London and the rise of](#)

London forex millionaires.

Feeling confused? Watch this video lecture on deep learning and recurrent neural networks especially the LSTM NNs. The lecture starts with a short introduction to neural networks NNs. So if you are totally new to NNs, you can understand a lot by watching the video below. As explained in the video below standard NNs lack memory. NNs have been tried a lot in predicting the stock market and the forex market but precisely because of lack of memory, predictions have not been good. So recurrent neural networks (RNNs) were developed. RNNs introduce feedback between the output and the input. Watch the video and the listen to the presenter he tries to explain these things in easy terms. [Read this post on a EURUSD swing trade that made 200% in 20 days.](#)



GRUs are special type of recurrent neural networks known as Long Short Term Memory Networks (LSTM). The problem with LSTM NNs is that they can take sometime to train. If we are doing day trading then you know we need a system that makes predictions in as short time

as possible so that we can use it in our trading. GRUs are a modified form of LSTMs that take less time to train. I had searched youtube a lot to find a lecture on GRUs but found this video on LTSM. Since GRUs are special modified type LSTMs, understanding LSTMs will help you understand GRUs. Below I will explain in words also what GRUs are. Before we continue read this shocking post on [how a tiny broker won and the big brokers went bankrupt trading Swiss Franc](#).

Can We Predict The British Pound Flash Crash With GRU?

Now you should keep this fact in mind, the prediction that we make will not be 100% accurate. The best that we can do is predict the market direction correctly. If we can do that we have come a long way.

```
#import numpy, pandas and matplotlib
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

#read the data from the csv file
df = pd.read_csv('E:/MarketData/GBPUSD1440.csv',
header=None)
df.columns=['Date', 'Time', 'Open', 'High',
'Low', 'Close', 'Volume']
df.shape
#show data
df.head()

#explore the data
zt=df.iloc[0:df.shape[0],5]
```

```
zt.head()
zt.tail()

#add time lags to the data
zt_1=zt.shift(1)
zt_2=zt.shift(2)
zt_3=zt.shift(3)
zt_4=zt.shift(4)
zt_5=zt.shift(5)
df1=pd.concat([zt,zt_1,zt_2,zt_3,zt_4,zt_5],
axis=1)
df1.columns=
['zt','zt_1','zt_2','zt_3','zt_4','zt_5']
df1.tail(6)
df1.head(10)

#drop NaN
df1=df1.dropna()
y=df1['zt']
cols=['zt_1','zt_2','zt_3','zt_4','zt_5']
x=df1[cols]
y.tail()
y.head()
x.tail()
x.head()
len(x)

#transform the data into train and test
from sklearn import preprocessing
scaler_x=preprocessing.MinMaxScaler(feature_range=(-1,1))
x=np.array(x).reshape((len(x),5))
```

```
x=scaler_x.fit_transform(x)
```

```
scaler_y=preprocessing.MinMaxScaler(feature_range=(-1,1))
```

```
y=np.array(y).reshape((len(y),1))
```

```
y=scaler_y.fit_transform(y)
```

```
#the train set
```

```
train=300
```

```
x_train=x[len(x)-train-2:len(x)-2,]
```

```
x_test=x[len(x)-train-1:len(x)-1,]
```

```
y_train=y[len(x)-train-2:len(x)-2]
```

```
y_test=y[len(x)-train-1:len(x)-1]
```

```
x_train=x_train.reshape(x_train.shape+(1,))
```

```
x_test=x_test.reshape(x_test.shape+(1,))
```

```
x_train.shape
```

```
x_test.shape
```

```
#build a 4 units GRU
```

```
from keras.models import Sequential
```

```
from keras.layers.core import Dense, Activation
```

```
from keras.layers.recurrent import GRU
```

```
fit1=Sequential()
```

```
fit1.add(GRU(output_dim=4,
```

```
            return_sequences=False,
```

```
            activation='tanh',
```

```
            inner_activation='hard_sigmoid',
```

```
            input_shape=(5,1)))
```

```
fit1.add(Dense(output_dim=1,
```



```

activation='linear'))
fit1.compile(loss="mean_squared_error",
              optimizer="rmsprop")

#Now fit the model
fit1.fit(x_train,y_train,batch_size=1,
         nb_epoch=10)
#print summary of the GRU model
fit1.summary()

#train and test for MSE
score_train=fit1.evaluate(x_train, y_train,
batch_size=1)
score_test=fit1.evaluate(x_test, y_test,
batch_size=1)
print( "in train MSE= ", round(score_train, 5))
print ("in test MSE= ", round(score_test,5))

#get the predicted value
pred1=fit1.predict(x_test)
pred1=scaler_y.inverse_transform(np.array(pred1)
                                   .reshape((len(pred1),1))))

#print the predictions
pred1[-1]
df.tail()

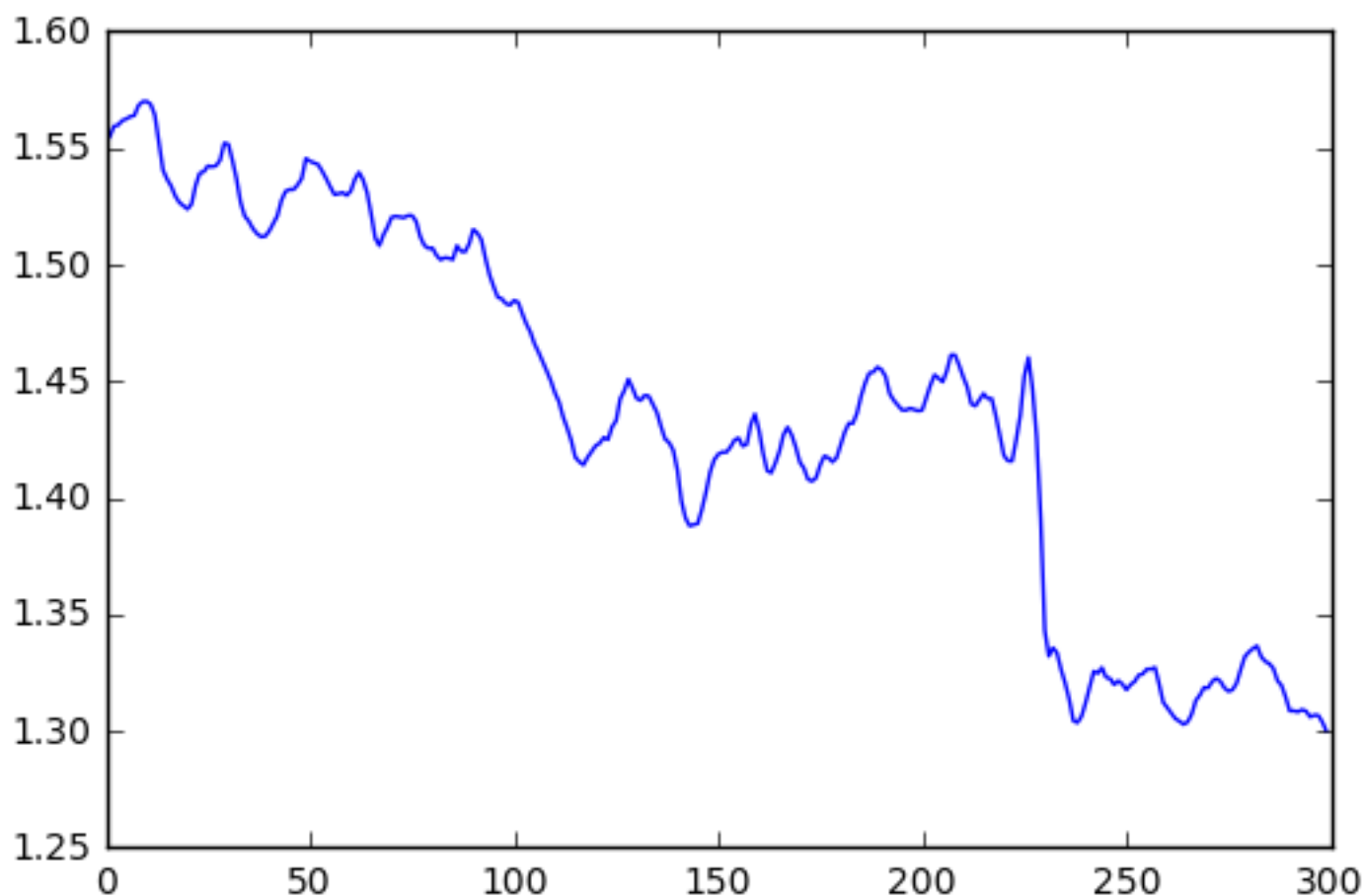
```

Now my computer took around 20-30 minutes to run the above python code and model a GRU. In the beginning the computer freezed then recovered after 10-15 minutes. This was the prediction:

```
>>> pred1[-1]
array([ 1.29994762], dtype=float32)
>>> df.tail()
```

	Date	Time	Open	High
Low				
Close				
Volume				
2282	2016.10.03	00:00	1.29251	1.29464
1.28177	1.28410	29153		
2283	2016.10.04	00:00	1.28363	1.28609
1.27196	1.27285	33511		
2284	2016.10.05	00:00	1.27259	1.27714
1.26859	1.27502	30210		
2285	2016.10.06	00:00	1.27473	1.27599
1.26023	1.26159	29415		
2286	2016.10.07	00:00	1.26154	1.26227
1.14116	1.23531	25816		

The last observation is the day British Pound Flash took place. You can see GBPUSD opened at 1.26154 then made a low of 1.14116 and after that made a recovery and closed at 1.23531. GRU is predicting the closing price at 1.29999 which is the exact opposite of what had happened. Price went down and our model is predicting the price will go up. Even if our GRU model had predicted a close price of 1.22000 we could have taken it as a good model. But in the present state it miserably failed to train well from the data. This is the challenge in predicting financial time series. Price can abruptly change direction. We cannot predict it. Below is the plot of the daily close that we are trying to predict.



Now you can see in the above plot a few months before the British Pound Flash Crash we had Brexit. On the day of Brexit, GBPUSD had fallen around 2000 pips. WE can try GRU model using weekly data and see if it can make a good prediction. But why not try LSTM now!

Can LSTM Predict the British Pound Flash Crash?

You should be now familiar with Long Short Term Memory (LSTM) Neural Network after watching the above video. I have build this LSTM model in which I use past 10 values to predict the future value. Below is the python code that took around once again 15-30 minutes.

```
#Long Short Term Memory Recurrent Neural Network
For Day Trading
import numpy as np
import pandas as pd

#read the data from the csv file
```

```
data1 =
pd.read_csv('E:/MarketData/GBPUSD1440.csv',
header=None)
data1.columns=['Date', 'Time', 'Open', 'High',
'Low', 'Close', 'Volume']
data1.shape
#show data
data1.tail()

#explore the data

yt=data1.iloc[0:len(data1),5]
yt.head()
yt.tail()

#add time lags to the data
yt_1=yt.shift(1)
yt_2=yt.shift(2)
yt_3=yt.shift(3)
yt_4=yt.shift(4)
yt_5=yt.shift(5)
yt_6=yt.shift(6)
yt_7=yt.shift(7)
yt_8=yt.shift(8)
yt_9=yt.shift(9)
yt_10=yt.shift(10)
data2=pd.concat([yt,yt_1,yt_2,yt_3,yt_4,yt_5,
yt_6,yt_7,yt_8,yt_9,yt_10],
axis=1)
data2.columns=
['yt', 'yt_1', 'yt_2', 'yt_3', 'yt_4', 'yt_5',
```

```
'yt_6', 'yt_7', 'yt_8', 'yt_9', 'yt_10']
```

```
data2.tail(12)
```

```
data2.head(12)
```

```
#drop NaN
```

```
data2=data2.dropna()
```

```
y=data2['yt']
```

```
cols=['yt_1', 'yt_2', 'yt_3', 'yt_4', 'yt_5',  
      'yt_6', 'yt_7', 'yt_8', 'yt_9', 'yt_10']
```

```
x=data2[cols]
```

```
x.tail()
```

```
#transform the data into train and test
```

```
from sklearn import preprocessing
```

```
scaler_x=preprocessing.MinMaxScaler(feature_range=(-1,1))
```

```
x=np.array(x).reshape((len(x),10))
```

```
x=scaler_x.fit_transform(x)
```

```
x.shape
```

```
scaler_y=preprocessing.MinMaxScaler(feature_range=(-1,1))
```

```
y=np.array(y).reshape((len(y),1))
```

```
y=scaler_y.fit_transform(y)
```

```
y.shape
```

```
#the train set
```

```
x_train=x[len(x)-310:len(x)-2,]
```

```
x_test=x[len(x)-309:len(x)-1, ]
```

```
y_train=y[len(x)-310:len(x)-2]
y_test=y[len(x)-309:len(x)-1]
x_train=x_train.reshape(x_train.shape+(1,))
x_test=x_test.reshape(x_test.shape+(1,))
x_train.shape
x_test.shape
```

#Long Short Term Memory Network Specifications

```
from keras.models import Sequential
from keras.layers.core import Dense
from keras.layers.recurrent import LSTM
model1=Sequential()
model1.add(LSTM(output_dim=9, activation='tanh',
               inner_activation='hard_sigmoid',
               input_shape=(10,1)))
model1.add(Dense(output_dim=1,
                 activation='linear'))
model1.compile(loss="mean_squared_error",
               optimizer="rmsprop")
```

```
#Time to fit the LSTM model with shuffle set to
false we can set it to true
model1.fit(x_train, y_train, batch_size=1,
          nb_epoch=4, shuffle=False)
```

#train and test MSE

```
score_train=model1.evaluate(x_train, y_train,
                             batch_size=1)
print( "in train MSE= ", round(score_train, 4))
```

```
#get the predicted value
pred1=model1.predict(x_test)
pred1=scaler_y.inverse_transform(np.array(pred1)
    .reshape((len(pred1),1)))

pred1[-1]
```

When we run the above code we have the following prediction;

```
>>> pred1[-1]
array([ 1.27535546], dtype=float32)
```

Now we have a prediction of 1.2753 while the actual close was 1.23531. Once again we have a poor prediction. So both GRU and LSTM models miserably fail in predicting GBPUSD Flash Crash. We had only used the closing price. We have no other input. Maybe we have to rebuild the model using more inputs. Data science is all about using the right inputs. So we can safely say that our manual trading system can predict the market better than an algorithmic trading system based on a GRU and an LSTM. We will continue with our quest for an algorithm that can predict the market correctly 80-90% of the time in the next post. Suffice to say we will have to improve our feature selection if we want to make better predictions. [Did you take a look at my new course C For Traders?](#)

0 Comments

Comments

Trackbacks

