BT2101 FINAL PROJECT REPORT

Lim Jin Ming, Jeremy Denzel (A0172720M)
Cai Shuhang (A0167788A)
Kang Choon Kiat (A0171128L)

NOVEMBER 24, 2018
TEAM 10

Table of Contents

1. Summary

The application of deep learning approaches to finance has received a great deal of attention from both investors and researchers (Faggella, 2018). This final report details the strategies and techniques used by the team for trading financial instruments on QuantConnect. Models such as Support Vector Machines and Recurrent Neural Networks were studied in this report. The performance of the models was assessed, and the optimal configurations of the models were described. A Jupyter notebook with our exploratory data analysis and a video demonstration of our code are attached to support our documentation.

2. Asset Selection

The team had an option to trade in the Stock, Forex and Cryptocurrency market and decided to trade in the Forex market after weighting each market's pros and cons as well as the time period of trade. The following table summarises the advantages and disadvantages of each market.

Market	**Stocks**	**Forex**	**Cryptocurrency**
Advantages	• Dividend Income	• High liquidity • Allows use of Leverage • 24-hour market	• The market does not close • Mobile trading
Market	**Stocks**	**Forex**	**Cryptocurrency**
Disadvantages	• Better for long-term investment rather than short-term trade as	• High volatility: Greater loss • Leverage has the potential to	• QuantConnect does not allow for shorting • Trends are hard to predict

	stocks may pay dividends • Leverage has the potential to enlarge profits or losses by the same magnitude.	enlarge profits or losses by the same magnitude.	• High transaction fee • Liquidity not as high as stocks and forex • Very high volatility

Given two weeks to trade, forex will be the most optimal as it has a higher volatility thus increasing the chances of gaining profit. It has a lower transaction fee as compared to trading cryptocurrency (deCsesznak, 2015). The team feels that stocks are more suitable for long-term investment (Forex Market Size: A Traders Advantage, 2014).

3. Choice of Currency

EURUSD was the chosen traded currency pair. The Euro and U.S. dollar are the world's two largest currencies, representing the world's two largest economies. Many corporations conduct business in both the United States and Europe. They have an almost constant need to hedge their exchange rate risk. It has a high liquidity due to its popularity. It is also not as volatile as the other currency pairs due to its low bid-ask spread (Murphy, n.d.).

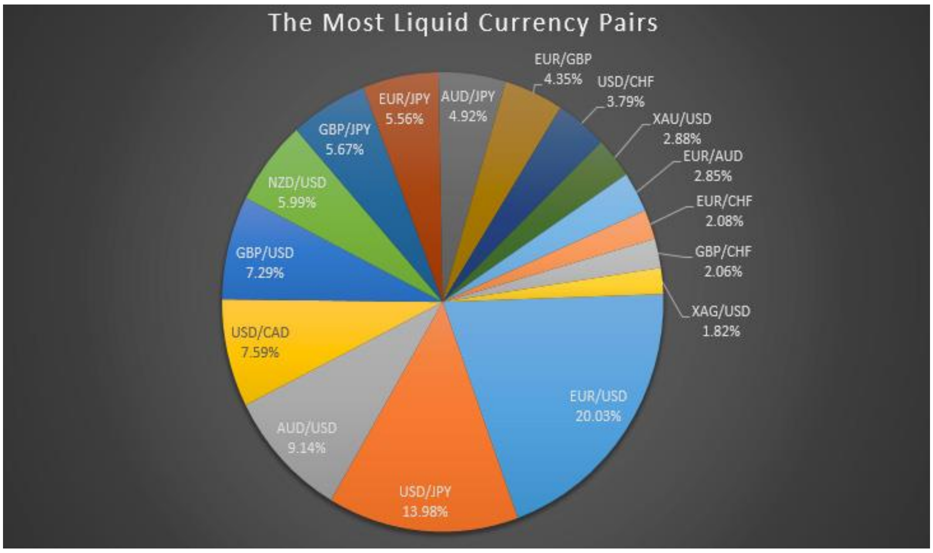


Figure 1: Currency Pairs

Source: Most Liquid Currency Pairs

4. Feature Engineering

In addition to open, high, low, close price, attributes were constructed from the available data to build the features that comprise the training set. A diverse range of features such as seasonality features, lagged values and technical indicators were used in the construction of the models to improve the classification capability of the model. The characteristics of the data and the selected attributes are shown in Table 1:

Feature	Formula	Description
Average Price	(High + Low) / 2	Average price for the day
Range	High - Low	The trading range for the day
Open-High-Low-Close Price	(Open + High + Low + Close) / 4	Average of the OHLC price for the day

Open Close Price Difference	Open - Close	Difference between the open and close price for the day.
5-day Simple Moving Average of Close Price (SMA)	Sum(Close Price of past 5 days) / 5	Mean of the close price of the past 5 days
Relative Strength Index (RSI)	RSI(Close price,14 days)	RSI oscillates between zero and 100. RSI is considered overbought when above 70 and oversold when below 30. (Investopedia, n.d.)
Stop and Reverse (SAR	SAR(High,Low,acceleration =0.2, maximum=0.2)	SAR is a method to find potential reversals in the market price direction.
Average Directional Index (ADX)	ADX(High,Low,Close,14 days)	ADX is an indicator of trend strength in a series of prices of a financial instrument
Average True Range (ATR)	ATR(High,Low,Close,14)	ATR reflects the volatility of a price in absolute price levels.
Principal Component Analysis	PCA = PCA(n_components=1)	PCA is a method that brings together a measure of how each variable is associated with one another, the directions in which the data are dispersed as well as the relative importance of these different directions.

Intraday Percentage Change	% $change_i = (closePrice_i - openPrice_i) / openPrice_i$	The relative change in the price within the current day
Interday Percentage Change	% $change_i = (closePrice_i - closePrice_{i-1}) / (closePrice_{i-1})$	The relative change in the price between day_i and day_{i-1}

Regressions Models employed were trained with one year of data at daily intervals. The team backtested the model within a two-week period to simulate the actual trading period. These models would output the predicted Price at the next time stamp. They were evaluated based on their mean squared error (MSE).

5. Primary Machine Learning Model

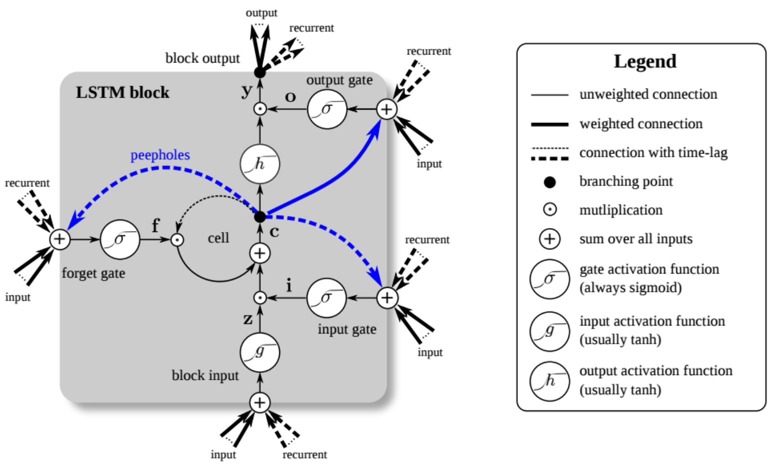
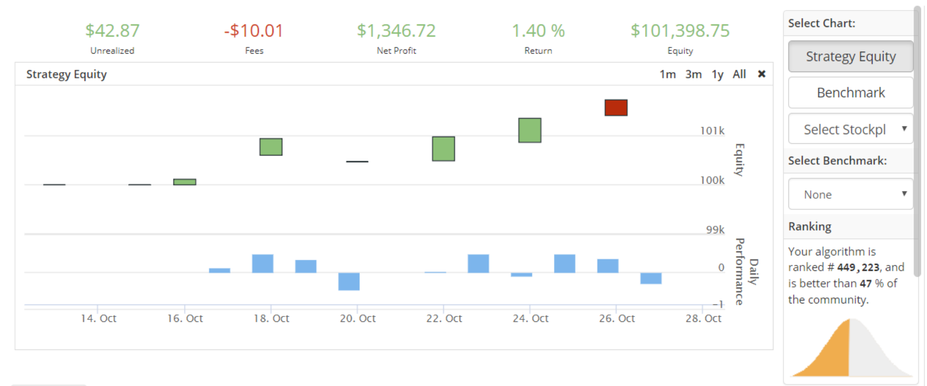


Figure 2: LSTM Framework

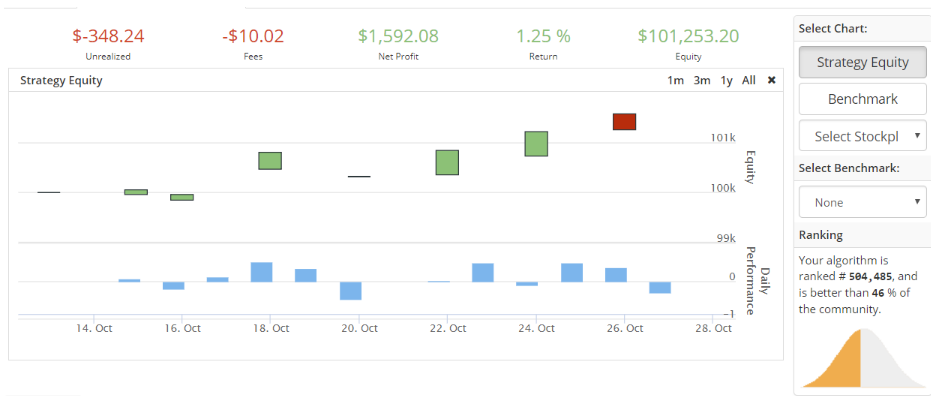
The team utilised a type of Recurrent Neural Network (RNN) for the final model. An RNN cell not only considers its present input but also the output of RNN cells preceding it. Long short-term memory (LSTM) is one of the variations of RNN architecture.

6. Experimental results and Evaluation of Models



Total Trades	5	Average Win	0.67%
Average Loss	0%	Compounding Annual Return	43.641%
Drawdown	0.500%	Expectancy	0
Net Profit	1.399%	Sharpe Ratio	5.53
Loss Rate	0%	Win Rate	100%
Profit-Loss Ratio	0	Alpha	0.349
Beta	-5.274	Annual Standard Deviation	0.049
Annual Variance	0.002	Information Ratio	5.224
Tracking Error	0.049	Treynor Ratio	-0.051
Total Fees	$10.01		

Figure 3: Backtest Results for SVM

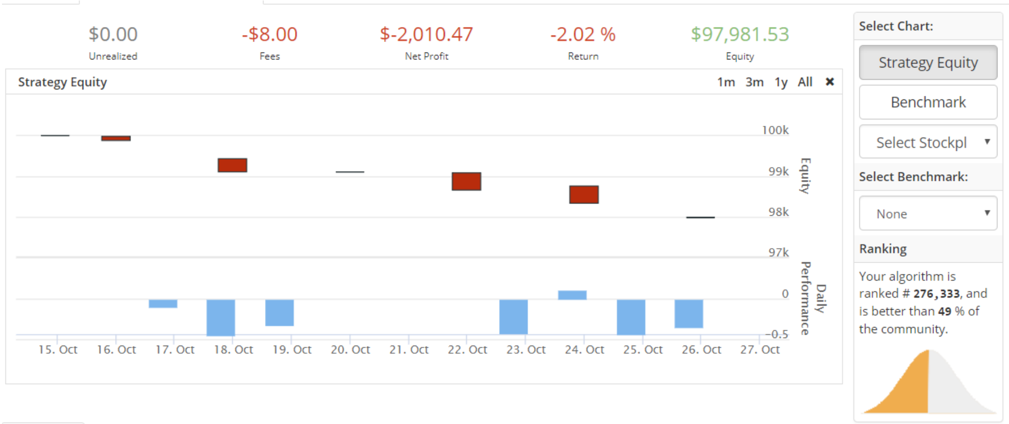


Overview | Report | Trades | Logs | Code | Share

Overall Statistics — Download Results

Total Trades	5	Average Win	0.79%
Average Loss	0%	Compounding Annual Return	38.361%
Drawdown	0.500%	Expectancy	0
Net Profit	1.253%	Sharpe Ratio	4.797
Loss Rate	0%	Win Rate	100%
Profit-Loss Ratio	0	Alpha	0.259
Beta	-1.108	Annual Standard Deviation	0.051
Annual Variance	0.003	Information Ratio	4.502
Tracking Error	0.051	Treynor Ratio	-0.219
Total Fees	$10.02		

Figure 4: Backtest Result for LSTM



Overview | Report | Trades | Logs | Code | Share

Overall Statistics — Download Results

Total Trades	4	Average Win	0%
Average Loss	-1.01%	Compounding Annual Return	-46.102%
Drawdown	2.000%	Expectancy	-1
Net Profit	-2.018%	Sharpe Ratio	-12.749
Loss Rate	100%	Win Rate	0%
Profit-Loss Ratio	0	Alpha	-0.413
Beta	-1.034	Annual Standard Deviation	0.034
Annual Variance	0.001	Information Ratio	-13.158
Tracking Error	0.034	Treynor Ratio	0.413
Total Fees	$8.00		

Figure 5: Backtest for Logistic Regression

7. Best Performing Model

The team had backtested three different models initially and the LSTM model had the best performance in terms of net profit, compounding return and Sharpe ratio. The LSTM model is programmed to predict the next day's closing price. This gives the team a better indication of whether to enter a short or long position.

8. Feature Selection for LSTM Model

```
df = df[['open', 'high', 'low', 'close']]              #Features to be included
df['avg_price'] = (df['low'] + df['high']) / 2       #Average Price
df['range'] = df['high'] - df['low']                 #Price Range
df['ohlc_price'] = (                                 #Open-High-Low-Close Average Price
    df['low'] + df['high'] + df['open'] + df['close']) / 4
df['oc_diff'] = df['open'] - df['close']             #Open Close Difference
df['percentage_change'] = (df['close'] - df['open'])/df['open']          #Percentage change in price fo
df['cp_change'] = (df['close'] - df['close'].shift(1))/(df['close'].shift(1))       #Change in closing price
df['SMA'] = df['close'].rolling(window=5).mean()           #Simple Moving Average of 5 days
df.dropna(inplace=True)

dataset = df.copy().values.astype('float32')
pca_features = df.columns.tolist()
```

Figure 6: Features used

The team chose to use Open, High, Low, Close, Average Price, Range, Open-High-Low-Close Price (OHLC), Open-Close Difference (OC-diff), Change in closing price (cp-change), percentage change in price for the day (percentage_change) and a Simple Moving Average (SMA) of 5 days as features for the model. The features are chosen through a series of trial and error with the intent to maximise the accuracy and net profit.

9. Model Buy/Sell Execution

```
self.Schedule.On(
    self.DateRules.Every(DayOfWeek.Monday, DayOfWeek.Tuesday,  # Schedule function to enter a position
                         DayOfWeek.Wednesday, DayOfWeek.Thursday,
                         DayOfWeek.Friday),
    self.TimeRules.Every(TimeSpan.FromMinutes(360)),
    Action(self.Rebalance))

self.Schedule.On(  # Schedule function to exit a position
    self.DateRules.EveryDay(self.symbol),
    self.TimeRules.Every(TimeSpan.FromMinutes(10)),
    Action(self.Rebalance2))
```

Figure 7: Schedule Function

The team used schedule functions to determine whether to enter or exit a position at fixed intervals. This ensures that opportunities for profit are maximised as trades are carried out throughout the day.

```
df1 = self.hist_data  # Calculate and update new atr everyday
df1 = df1[['open', 'high', 'low', 'close']]
self.atr = ta.ATR(df1['high'], df1['low'], df1['close'], timeperiod=14)
self.Debug("atr" + str(self.atr[-1]))
```

Figure 8: ATR Code

```
if self.currency in self.long_list:
    cost_basis = self.Portfolio[self.currency].AveragePrice  # To calculate the price of our FOREX currency pair

    if ((curr_price <= float(cost_basis) - float(self.atr[-1]))  # Stop loss-Take Profit conditions
            or (curr_price >=  # Use of Average True Range to determine price to sell
                float(self.atr[-1] * 1) + float(cost_basis))):
        self.Debug("price is: " + str(curr_price))
        self.SetHoldings(self.currency, 0)
        self.long_list.remove(self.currency)


if self.currency in self.short_list:
    cost_basis = self.Portfolio[self.currency].AveragePrice

    if ((curr_price <= float(cost_basis) - float(self.atr[-1] * 1))  # Stop loss-Take Profit conditions
            or  # Use of Average True Range to determine price to buy back
            (curr_price >= float(self.atr[-1]) + float(cost_basis))):
        self.Debug("price is: " + str(curr_price))
        self.SetHoldings(self.currency, 0)
        self.short_list.remove(self.currency)
```

Figure 9: Use of ATR in Stop Loss-Take Profit

ATR was incorporated to determine the stop loss and take profit levels (England, 2013).

Heteroskedasticity is a key feature of most economic time series (University of Pennsylvania). This means that the volatility is clustered: Usually, a long period of low volatility is followed by a short period of high volatility and this pattern is repeated. Following this principle, the team implemented a ten-day sliding window in our model. Since the weights of the model are based on their simulated profitability, the training periods are kept small so that the weights can adapt quickly and learn faster when the market enters a period of high volatility.

10. Model Optimization

1. Model checkpoint for reinforcement learning:

```
# Save the best weight during training.
from keras.callbacks import ModelCheckpoint
checkpoint = ModelCheckpoint("weights.best.hdf5", monitor='val_mean_squared_error', verbose=1, save_best_only=True, mode='min')

# Fit
callbacks_list = [checkpoint]
history = model.fit(trainX, trainY, epochs=200, batch_size=500, verbose=0, callbacks=callbacks_list, validation_split=0.1)
```

```
Epoch 00001: val_mean_squared_error improved from inf to 0.00033, saving model to weights.best.hdf5

Epoch 00002: val_mean_squared_error improved from 0.00033 to 0.00007, saving model to weights.best.hdf5

Epoch 00003: val_mean_squared_error improved from 0.00007 to 0.00004, saving model to weights.best.hdf5

Epoch 00004: val_mean_squared_error did not improve from 0.00004

Epoch 00005: val_mean_squared_error did not improve from 0.00004

Epoch 00006: val_mean_squared_error did not improve from 0.00004

Epoch 00007: val_mean_squared_error did not improve from 0.00004

Epoch 00008: val_mean_squared_error improved from 0.00004 to 0.00003, saving model to weights.best.hdf5

Epoch 00009: val mean squared error did not improve from 0.00003
```

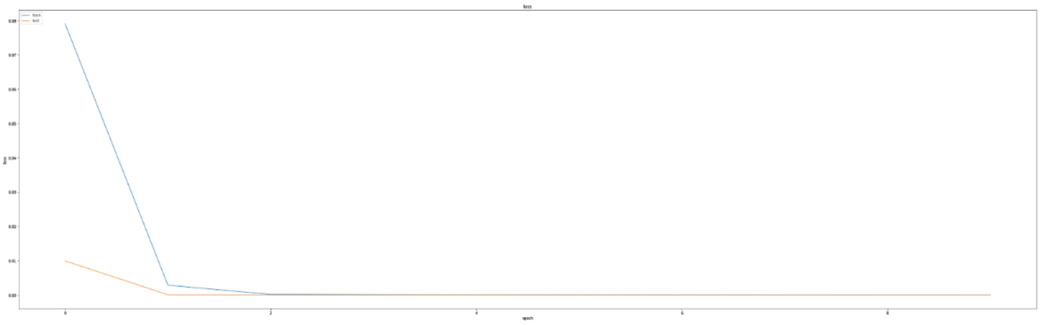


Figure 10: Optimization Graph

Every epoch is being checked for its mean-squared-error(MSE) and every epoch with the lowest MSE will be saved as a checkpoint. This information also serves as a metric to tune the hyperparameters of the LSTM.

2) Setting Dropouts in the LSTM model:

```
self.model = Sequential()  # Setting up of LSTM Model
self.model.add(LSTM(20, input_shape=(X.shape[1], X.shape[2]), return_sequences=True))
self.model.add(LSTM(10, return_sequences=True))
self.model.add(Dropout(0.2))
self.model.add(LSTM(10, return_sequences=False))
self.model.add(
    Dense(1, kernel_initializer='he_normal', activation='relu'))
```

Figure 11: LSTM Layers

Dropout is a regularization method where input and recurrent connections to LSTM units are probabilistically excluded from activation and weight updates while training a network. This has the effect of reducing overfitting and improving model performance (Brownlee, 2017).

3) Monitoring the difference between the predicted and actual price

```
pred = self.model.predict(testX)  # Predict on test data
pred = y_scaler.inverse_transform(pred)
close = y_scaler.inverse_transform(
    np.reshape(testY, (testY.shape[0], 1)))
predictions = pd.DataFrame()  # Create dataframe for evaluation of prediction on test data
predictions['predicted'] = pd.Series(
    np.reshape(pred, (pred.shape[0])))
predictions['actual'] = pd.Series(
    np.reshape(close, (close.shape[0])))
predictions = predictions.astype(float)
predictions['diff'] = predictions['predicted'] - predictions[
    'actual']  # Calculate difference between predicted and actual value

p = df[-pred.shape[0]:].copy()
predictions.index = p.index
predictions = predictions.astype(float)
predictions = predictions.merge(
    p[['low', 'high']], right_index=True,
    left_index=True)  # Add in high low price to see whether predicted value falls between this range
self.Debug(predictions)

trainscore = mean_squared_error(close, pred)
self.Debug('MSE' + str(trainscore))
```

```
                   predicted    actual      diff       low      high
symbol time
EURUSD 2018-10-18   1.158362  1.155850  0.002512  1.149425  1.157795
       2018-10-19   1.158154  1.150035  0.008119  1.144915  1.152740
       2018-10-20   1.154455  1.145910  0.008545  1.143300  1.153475
       2018-10-22   1.151238  1.151385 -0.000147  1.149825  1.151980
       2018-10-23   1.153172  1.151120  0.002053  1.145220  1.155035
       2018-10-24   1.153576  1.145560  0.008016  1.143945  1.149385
       2018-10-25   1.150391  1.146835  0.003556  1.137905  1.147690
       2018-10-26   1.150297  1.141245  0.009052  1.135615  1.143275
       2018-10-27   1.147065  1.136850  0.010215  1.133570  1.142130
       2018-10-29   1.144308  1.140425  0.003883  1.138630  1.140405
       2018-10-30   1.145051  1.139195  0.005856  1.136060  1.141650
       2018-10-31   1.145212  1.138095  0.007117  1.133615  1.138795
       2018-11-01   1.144657  1.134215  0.010442  1.130215  1.136040
       2018-11-02   1.142835  1.134325  0.008510  1.133775  1.142420
       2018-11-03   1.142261  1.140895  0.001366  1.137220  1.145595
       2018-11-05   1.145162  1.138810  0.006352  1.138275  1.140440
       2018-11-06   1.145081  1.138480  0.006601  1.135375  1.142410
       2018-11-07   1.144804  1.139985  0.004819  1.139145  1.147315
2018-11-08 00:00:00 0.048690780997276306
2018-11-08 00:00:00 MSE4.4619977e-05
2018-11-08 00:00:00 Predicted price is 1.148163914680481
2018-11-08 00:00:00 Prev_price is 1.1399849653244019
2018-11-08 00:00:00 long
```

Figure 12: Computation of test result

The team observed the difference between the predicted and actual value and whether the predicted value falls between the low and high price and also printed out the MSE of the model. This serves as a metric for the team to evaluate the model. Gridsearch is then conducted across a different number of hidden layers and varying dropout rates to further optimize the model so as to minimize the difference, resulting in more profitable trades.

11. Model Evaluation

11.1. Strengths

Predicting the exact price instead of the direction allows the team to have more control and flexibility over the execution of a trade. If the profit margin is not large enough, the team can choose not to enter a trade. It also allows the team to have a measure of how accurate the model is in predicting the price and to improve it by minimizing the difference in the actual and predicted value to make more profitable trades.

The team used ATR to determine the stop loss and take profit levels. Using ATR is better than using a fixed percentage because they change based on the characteristics of the stock being traded, recognizing that volatility varies across issues and market conditions (England, 2013). It is more reflective of the volatility of the asset and maximises the potential profit to be gained.

LSTMs are a special kind of RNN, capable of learning long-term dependencies. They were explicitly designed to avoid the long-term dependency problem that is a limitation of traditional RNNs (Colah, 2015). They are also an effective solution for combating the problem of vanishing gradient by using memory cells (Wei Bao, 2017). With a large set of time series data, LSTM is able to learn from oscillatory patterns (seasonality and cycles) as it is able to retain useful information for future predictions (Brownlee, 2017).

11.2. Weaknesses

The model trains on daily data and predicts each day’s closing price but make trades throughout the day. Although it may accurately predict each day’s closing price, it is unable to capture the hourly trend of a day. It also does not capture the volatility in a day due to the lack of hourly data. As the price of the currency can fluctuate greatly in a day, there are many opportunities to enter a trade. Without the use of hourly data, the model fails to capitalise on such opportunities to maximise our profits.

As the model trains solely on historical data, it will not be able to predict external events that might affect the price of the market. They include political moves by countries, announcements by the major economic institutions like the central bank, as well as economic crisis. Machine Learning models, in general, are still far from being optimal solutions for financial forecasting due to the complex nature of the financial time series, erratic behaviours in the market and the possible presence of extreme events that can undermine any generalization or pattern found on them. These models may only be useful in times of “normal” behaviour of the market.

12. Project Reflection

The team could have conducted a more in-depth analysis of the correlation between different currency pairs, how it influences the movements of other currencies and using it as an indicator to predict the movement of the currency price.

Instead of focusing on one currency pair, the team could have expanded and traded more currency pairs or even other assets such as stocks and cryptocurrency. Portfolio diversification reduces the potential loss of investment from concentrating all the capital on one asset and increases return through multiple sources of income (Dixon Advisory, n.d.).

The incorporation of sentiment analysis into the model would improve the model as well. Sentiment analysis is the process of computationally identifying and categorizing opinions expressed in a piece of text. Sentiment analysis can be drawn from news articles about the current market situation or on twitter (Gupta, 2017). For example, tweets by Donald Trump can often have a huge effect on the financial market.

13. Conclusion

In this report, the team described a sample implementation of a model capable of predicting the close price of the EUR/USD currency pair for each trading day. A LSTM neural network is used to generate the one-step-ahead output in a supervised manner. Our input variables include the daily OHLC variables and technical indicators. Backtesting results provide evidence that the model is capable of being profitable while trading without supervision. It is also the team’s first attempt at going live, therefore there are many circumstances (e.g. slippage, data overloading) that are unaccounted for.

Although the LSTM model has a satisfactory predictive performance, it still has some insufficiencies. For example, a more advanced hyper-parameters selection process can be conducted to further optimize the proposed deep learning framework. In addition, a wider range of input variables like interest rates and Gross Domestic Product could be considered to allow the model to be more robust.

Lastly, the team would like to express their gratitude to Professor Keith Carter for his guidance and support through this project as well as Mr Shashank for helping us to debug our code.

References

1) Brownlee, J. (2017, 4 28). *How to Use Dropout with LSTM Networks for Time Series Forecasting.* Retrieved from Machine Learning Mastery: https://machinelearningmastery.com/use-dropout-lstm-networks-time-series-forecasting/

2) deCsesznak, S. (2015, 8 7). *Bitcoin vs Forex*. Retrieved from Bitcoin.com: https://news.bitcoin.com/bitcoin-vs-forex-2/

3) Dixon Advisory. (n.d.). *Benefits of diversification*. Retrieved from Dixon Advisory: https://www.dixon.com.au/investment-advice/benefits-of-diversification

4) England, W. (2013, 10 29). *How to Use ATR in a Forex Strategy*. Retrieved from DailyFX: https://www.dailyfx.com/forex/education/trading_tips/post_of_the_day/2013/10/28/How_to_use_ATR_in_a_Forex_Strategy.html

5) Faggella, D. (2018, 11 5). *Machine Learning in Finance – Present and Future Applications*. Retrieved from Tech Emergence: https://www.techemergence.com/machine-learning-in-finance/

6) Forex Market Size: A Traders Advantage. (2014, 1 24). Retrieved from DailyFx: https://www.dailyfx.com/forex/education/trading_tips/daily_trading_lesson/2014/01/24/FX_Market_Size.html

7) Gupta, S. (2017, 1 28). *Sentiment Analysis: Concept, Analysis and Applications*. Retrieved from Towards Data Science : https://towardsdatascience.com/sentiment-analysis-concept-analysis-and-applications-6c94d6f58c17

8) Investopedia. (n.d.). *Relative Strength Index - RSI*. Retrieved from Investopedia: https://www.investopedia.com/terms/r/rsi.asp

9) Murphy, C. (n.d.). *EUR-USD Pair*. Retrieved from Investopedia: https://www.investopedia.com/walkthrough/forex/intermediate/level6/eur-usd.aspx

10) University of Pennslyvania . (n.d.). *University of Pennslyvania .* Retrieved from Econometrics Slides: https://www.sas.upenn.edu/~fdiebold/Teaching221/EconometricsSlides.pdf

11) Wei Bao, J. Y. (2017). *A deep learning framework for financial time series using stacked autoencoders and long-short term memory.* Chang Sha: Institute of Remote Sensing and Geographic Information System, Peking University, Beijing, China.

12) Colah. (2015, August). Understanding LSTM Networks. Retrieved from colah.github.io: http://colah.github.io/posts/2015-08-Understanding-LSTMs/