We apply Deep Recurrent Neural Networks with Long Short-Term Memory on time series forecasting; in particular to predict the very volatile Bitcoin (BTC) financial asset price over a number of input sequence lengths and forecast lengths. We reference prior work on ARIMA [1] to give insight on our baseline time series model.

Our Data

Our Data is the daily historical data of Bitcoin (BTC) sourced from a reputable exchange, along with other features including technical indicators. Our sample was drawn from daily prices since 01/01/2017. Our model performed best after regularization and normalization.

For the Bitcoin index, three categories of daily variables are used as model predictors. The first category contains historical Bitcoin trading data, such as the open, close, high, and low Bitcoin prices of the day or hour.

The second category consists of volume and percentage change in price recorded during trading. Those variables are widely used in stock and commodities trading.

The third category of features is derived from the third type of input. Derived features include the other attributes’ variation, exponentially weighted moving averages, seasonal decomposition, and trend decomposition, as well as Technical Indicators: MACD, Average True Range, BBI, Bollinger Bands. Prior work on Quantitative trading strategies provides some guidance on this front [2, 3].

In terms of model architecture, we explore a variety of LSTM models with up to six layers, with 3-5 LSTM layers and 1-2 Dense layers with a final Dense output layer were the main focus.

We implement dropout and batch normalization after every LSTM layer in the model. Each LSTM layer has the same number of hidden units, which we use as a hyper-parameter in our search.

We model the problem as a many-to-many sequence forecasting problem, and thus the final layer in our models have a number of output units equal to the number of time steps to be forecasted. Our models were trained with a mean squared error loss function applied to the back propagation phase of the learning algorithm. [4]

We implemented a simple learning construct to search the optimal set of hyper-parameters similar to the following pseudocode:

```
for h in hyper-parameters:
    for l in LSTM layers:
        for d in Dense layers:
            Train with h, l, d
            Evaluate and record scores
```

We then used the top three models with the lowest MAPEs to evaluate 15-day absolute percentage error (MAPE), which is defined as

\[
\text{MAPE} = \frac{\sum_{t=1}^{T} |\text{Actual}_t - \text{Forecast}_t|}{\sum_{t=1}^{T} \text{Actual}_t}
\]

We initially focus on quantity over quality, trying different architectures and approaches including binary classification to determine optimal window length and forecast length, finally settling on a 5-layer LSTM architecture which output a continuous future expected price value. We then move quality to the head of our decision making process to tune the final models. Rapid exploration turned out to be advantageous in uncovering our most promising models early on.

As you can see below, our predictions showed a solid amount of learning (low mean average absolute error).

![Fig. 5 Plot of our best performing models](image)

LSTM no. 14 outperformed the top 25 models with a MAPE of 0.022 on the training set and 0.061 on the test dataset. Fourteen other LSTMs outperformed the top performing VARN model and the top ARIMA model ranked at the bottom of the top 25 models list. The best VARN model ranked no. 16 with a MAPE of 0.26 on the training set and 0.837 on the test dataset. The best ARIMA model ranked no. 22 with a MAPE of 0.26 on the training set and 0.477 on the test dataset.

![Fig. 6 Best Performing Models](image)

References