Evaluating the performance of different Long Short-Term Memory networks (LSTM’s) on financial timeseries data using mean squared error in order to identify the optimum LSTM variant for regression performance on financial timeseries data

Patrick O’ Connor

A Thesis Submitted in Partial Fulfilment

of the requirements for the

Degree of

Master of Science in Data Analytics

Diagram

Description automatically generated with medium confidence

September 2024

Supervisor: Vikas Tomer

## Abstract

Financial forecasting models have been a prevalent fixture in modern society over a number of decades, and even extending to centuries. This field has evolved to utilising artificial intelligence, with deep learning the most recent development in this space, resembling processes similar to the human brain. Recurrent neural networks are a branch of deep learning which deals with sequential data, mimicking the short term memory of humans. This branch contains performance issues when it comes to longer sequences (long term memory) which can be mitigated through the use of an updated method called Long Short Term Memory networks (LSTM’s). Within the field of LSTM’s there are variations of this model which can be applied, incorporating different strengths and weaknesses based on the nuances of each LSTM variant. This study follows a CRISP-DM process, to perform prediction on oil price data through the use of 6 LSTM variants, comparing their performance by the metric of mean squared error (MSE). This study aims to discover the optimal parameter settings for each LSTM variant to achieve its performance potential, identify the strongest performing LSTM variant, as well as to synthesise the strengths and weaknesses of each variant against its relative performance. Primary research was also conducted in the form of in-depth interviews with industry professionals in the field of data analytics in order to validate these experiments as well as suggest methods to improve the achievement of these research objectives. GRU was found to be the highest performing LSTM variant achieving an MSE of 0.100, among other simpler variants who outperformed the more complex LSTM variants. Given the simple nature of the dataset, it can be hypothesised that the simpler LSTM variants are more suited to performing regression tasks on simpler financial timeseries datasets, a hypothesis which was supported in the primary research findings. Future improvements can be made to boost the reproducibility and durability of the results in this study through the use of a random seed or through multiple code implementations, documenting a distribution of results across each variant in order to find the highest performing variant based on mean scores among other statistical basis. These improvements would add to the robustness of the findings of this study.

Table of Contents

[Table of Figures ii](#_Toc178277145)

[Table of Tables iii](#_Toc178277146)

[1. Introduction 1](#_Toc178277147)

[2. Literature Review 4](#_Toc178277148)

[3. Research Methodology 13](#_Toc178277149)

[4. Evaluation & Implementation 24](#_Toc178277150)

[5. Results & Discussion 35](#_Toc178277151)

[6. Conclusion 44](#_Toc178277152)

[7. Bibliography 46](#_Toc178277153)

[8. Appendix 50](#_Toc178277154)

# Table of Figures

[Figure 1: CRISP-DM Process Model for Data Mining (Wirth & Hipp n.d.) 13](#_Toc178277065)

[Figure 2: Standard Scaler formula (Anon n.d.) 14](#_Toc178277066)

[Figure 3: Sigmoid Function (Datta, Agarwal, Kumar, et al. 2019) 15](#_Toc178277067)

[Figure 4: Hyperbolic Tangent Function (Datta, Agarwal, Kumar, et al. 2019) 15](#_Toc178277068)

[Figure 5: Example of a Vanilla LSTM layer 16](#_Toc178277069)

[Figure 6: Example of Bidirectional LSTM (Mardjo & Choksuchat 2024) 16](#_Toc178277070)

[Figure 7: Example of basic GRU architecture (Lakshmi & Maheswaran 2024) 17](#_Toc178277071)

[Figure 8: LSTM-Am Formula (Mardjo & Choksuchat 2024) 17](#_Toc178277072)

[Figure 9: Example LSTM-AM model (Mardjo & Choksuchat 2024) 17](#_Toc178277073)

[Figure 10: Extended LSTM algorithm to include a positive bias for the forget gate (Gers, Schmidhuber & Cummins 2000) 18](#_Toc178277074)

[Figure 11: Demonstration of a model averaging previous states producing state h3 and combining with w3 (word 3) in a language model to produce w4 after using softmax. This model also sets a forget gate bias at 1.0 combining uniform attention and forget gate bias 18](#_Toc178277075)

[Figure 12: Distribution of the data from 1988-2023 26](#_Toc178277076)

[Figure 13: Boxplot showing the distribution of price values 27](#_Toc178277077)

[Figure 14: Scatterplot of oil price over the timeframe of the dataset 28](#_Toc178277078)

[Figure 15: Boxplot of the Oil Prices distribution post-slicing to 1988-1998 29](#_Toc178277079)

[Figure 16: Histogram of distribution of Oil Price values post-slicing to 1988-1998 29](#_Toc178277080)

[Figure 17: Vanilla LSTM predictions vs real data 38](#_Toc178277081)

[Figure 18: GRU predictions vs real data 38](#_Toc178277082)

[Figure 19: LSTM-AM predictions vs real data 39](#_Toc178277083)

[Figure 20: LSTM-FGB predictions vs real data 39](#_Toc178277084)

[Figure 21: LSTM-FGB-AM predictions versus real data 40](#_Toc178277085)

[Figure 22: B-LSTM predictions versus real data 40](#_Toc178277086)

# Table of Tables

[Table 1: Vanilla LSTM: Study results versus the literature (Silva & Meneses 2023) (Dutta, Kumar & Basu 2020) (Ran, Shan, Fang, et al. 2019) 35](#_Toc178277043)

[Table 2: LSTM-AM result from this study versus the literature (Ran, Shan, Fang, et al. 2019) (Mardjo & Choksuchat 2024) 36](#_Toc178277044)

[Table 3: LSTM Variant MSE comparison with Optimal Hyperparameters (Implementation 1) 37](#_Toc178277045)

[Table 4: Findings from the in-depth interviews with data analytics practitioners. Number of interviewees connected to each insight is given in brackets beside the insight 41](#_Toc178277046)

[Table 5: Results of Code Implementation 2 50](#_Toc178277047)

[Table 6: Thematic Results of In-Depth Interviews 51](#_Toc178277048)

# Introduction

Financial forecasting has long been an area of concern among industry leaders. This capability allows organisations and individuals to predict future values of sales, stock and energy consumption among other use cases. The ability to predict future figures allows organisations to prepare themselves or take corrective action in order to avoid unfavourable outcomes with regard to their organisational goals. Financial markets have also drawn attention to the prediction of future values due to the opportunity to profit from buying stock and commodities when they are deemed to be underpriced and selling them on for a profit when a projected price rise occurs. This has led to multiple methods of forecasting being developed such as autoregressive models. Examples of autoregressive models include autoregressive moving average (ARMA), autoregressive integrated moving average (ARIMA) and seasonal autoregressive moving average with exogenous regressors (SARIMAX) which use a linear combination of past values in order to predict future values. Forecasting has since evolved to incorporate machine learning models followed by more advanced deep learning models which are able to identify non-linear patterns and relationships in data in order to make predictions. Recurrent neural networks (RNN’s) are a form of deep learning model which can handle sequential data. Through the use of a hidden state, recurrent neural networks can store information about past data as a short term memory informing its predictions when taking in the next data input. However, RNN’s can underperform with longer term sequences, necessitating the creation of Long Short Term Memory networks (LSTM’s). LSTM’s combine the hidden state capability of the RNN (short term memory) with its own cell state capability (long term memory) in order to mitigate against long term memory issues such as vanishing and exploding gradients.

## 1.1 Research Question

### 1.1.1 Title

Evaluating the performance of different Long Short-Term Memory networks (LSTM’s) on financial timeseries data using mean squared error in order to identify the optimum LSTM variant for regression performance on financial timeseries data.

### 1.1.2 Topic

Long Short-Term Memory networks (LSTM’s) are a variant of recurrent neural networks (RNN’s) created to address the vanishing gradient problem. RNN’s specialise in short-term memory, but can struggle when long-term memory is required for a regression problem. LSTM’s address this issue by keeping a cell state which acts as the long-term memory. In a typical LSTM network, various gates and activation functions are at work to decide which information to add or block from entering the long-term memory of the cell state to carry forward to the next layer. The operations of the gates, activation functions and cell states occur in a specific sequence for each observation passed into the LSTM layer.

There are variations of LSTM networks also, which perform similar operations but in a different sequence to one another. A couple of examples include the Bidirectional LSTM, which contains one layer which processes input data in a forward direction followed by another layer which processes input data in a backward direction along with the LSTM with Forget-Gate Bias which allows the forget gate to employ a bias in the form of a default value which it considers when deciding which information to forget and retain. There are a number of other variations of LSTM networks which can be explored in order to evaluate their impact on regression problems from the perspective of minimizing the mean squared error. Each variant can be tested on timeseries financial data in order to recommend an optimal variant for this kind of data problem as well as the reasons behind this finding. Having an optimal variant will save practitioners time and improve the performance of their LSTM networks on financial timeseries data.

### 1.2 Research Objectives (RO’s)

#### 1.2.1 RO1

Identify optimal parameter settings for each LSTM variant to increase predictive performance via mean squared error.

RO1 enables the performance of each LSTM variant to be maximised on the chosen financial data. This maximisation allows the LSTM variants to be accurately compared as parameter settings can skew the performance of each variant if not optimised. In addition, documenting the optimum parameter settings of each LSTM variant contributes to the literature informing practitioners on ways to maximise the results of their LSTM models in future projects.

#### 1.2.2 RO2

Evaluate the relative strengths and weaknesses of each LSTM variant in predictive performance via mean squared error to guide practitioners on suitable LSTM variants for specific regression problems based on the nuances of each variant.

RO2 plays a key role in identifying factors affecting superior performance (RO4) by documenting each LSTM variant’s strengths and weaknesses. This insight can inform practitioners on which LSTM variant might be most suitable for their projects when working with regression problems where LSTM networks are deemed suitable approaches.

#### 1.2.3 RO3

Identify the LSTM variant which achieved the lowest mean squared error when predicting the financial timeseries data in order to recommend a specific LSTM variant to lead with in similar regression problems for practitioners.

RO3 is a key objective in this project. This objective involves identifying the LSTM variant which achieves the lowest mean squared error on the financial timeseries data. The lower the mean squared error, the better the performance of the LSTM variant. This finding can also inform the field of LSTM network research by recommending a specific LSTM variant to lead with in a regression problem of similar conditions.

#### 1.2.4 RO4

Evaluate factors drawn from RO2 which may have contributed to the identified LSTM variant outperforming other variants in order for practitioners to understand the factors which can contribute to superior LSTM performance for similar regression problems.

RO4 builds on the documenting which takes place in RO2 of the relative strengths and weaknesses of the LSTM variants. This insight is then focused in on the identified LSTM variant with superior performance (RO3) and attempts to synthesise this superior performance with the relative strengths and weaknesses identified in RO2. This brings together knowledge of the outcome and factors contributing to the outcome which aids practitioners in understanding the reasons behind the superior performance of the chosen variant. Knowledge of these strengths and weaknesses can inform practitioners on the suitability of this variant’s use in their own specific regression problems.

# Literature Review

## 2.1 Introduction

There are a number of LSTM variants which will be explored as part of the literature review. The aim is to ascertain the relevant strengths and weaknesses of each variant and how this may relate to timeseries data. Academic resources will consist the majority of the sources consulted, with grey papers and less formal media being sought where relevant academic literature is limited. Authors and papers will be reviewed for elements of bias or validity concerns, which may distort a true picture of the state of the art around LSTM variants in regression problems. The research objectives of this study will inform the investigation into the existing literature with works on each individual LSTM variant explored.

The literature review will be organised thematically, with different LSTM variants comprising each theme. This approach allows sufficient background to be ascertained on each LSTM variant, along with areas of applicability and performance comparison across LSTM variants. The LSTM variants to be explored include:

1. *Vanilla LSTM*
2. *Bidirectional LSTM (B-LSTM)*
3. *Gated Recurrent Unit (GRU)*
4. *LSTM with Attention Mechanism (LSTM-AM)*
5. *LSTM with Forget Gate Bias (LSTM-FGB)*
6. *LSTM with Attention Mechanism & Forget Gate Bias (LSTM-FGB-AM)*

Each LSTM variant has been chosen according to widespread use in order to provide the most relevant findings for LSTM research in regression problems along with those which provided the most library support for robust implementation. A brief explanation of each variant is provided below before its exploration in the literature review:

### 2.1.1 Vanilla LSTM

Vanilla LSTM aligns closely with the original LSTM model discovered by Hochreidter & Schmidhuber (Hochreiter & Schmidhuber 1997). The LSTM network was originally created to address the vanishing gradient problem in RNNs by utilising a cell state to act as long term memory for the network along with an input gate and an output gate controlling which information enters the cell state.

The Vanilla LSTM contains a forget gate not originally present in LSTMs to allow the network to reset its state (Van Houdt, Mosquera & Nápoles 2020). This allows greater control for the model on information entering the cell state.

### 2.1.2 Bidirectional LSTM (B-LSTM)

B-LSTM consists of two LSTM layers, one layer which processes input data in a forward direction followed by another layer which processes input data in a backward direction, this is done in order to capture additional context from both the before and after states of the data.

### 2.1.3 Gated Recurrent Unit (GRU)

GRU combines the forget and input gates into an ‘update ‘ gate and merges the cell state and hidden state of the network resulting in a simpler model.

### 2.1.4 LSTM with Attention Mechanism (LSTM-AM)

LSTM-AM focuses on different sections of the input sequence dynamically, which aids in capturing long term dependencies. It does this through integrating an attention mechanism.

### 2.1.5 LSTM with Forget Gate Bias (LSTM-FGB)

LSTM-FGB allows the forget gate to employ a bias in the form of a default value which it considers when deciding which information to forget and retain. This approach can mitigate against the vanishing gradient problem.

### 2.1.6 LSTM with Forget Gate Bias & Attention Mechanism (LSTM-FGB-AM)

LSTM-FGB-AM combines the LSTM-AM which focuses on sections of the input sequence dynamically to capture long term dependencies as well as the LSTM-FGB which mitigates against the vanishing gradient problem by employing a bias default value when considering which information to forget and retain.

## Exploration

### 2.2.1 Vanilla LSTM

With financial timeseries data, Fischer and Krauss present Vanilla LSTMs as outperforming Random Forests (RFs), Artificial Neural Networks (ANNs) and Logistic Regression (LR) models (Fischer & Krauss 2018). This finding supports the relevance of Vanilla LSTMs as a widespread approach to regression problems, allowing important features to be discovered also in the data such as short term reversal profiles and high volatility in the financial timeseries data examined. Further displays of Vanilla LSTM’s strong performance in financial time series data can be found through the work of Yan and Ouyang (Yan & Ouyang 2018). The authors found Vanilla LSTM’s were able to outperform ANNs, Support Vector Machines (SVMs) and K-Nearest Neighbours (KNNs), with notable performance in both static and dynamic prediction with financial data. One notable element which may skew the insights from a compatibility perspective is the inclusion of wavelength analysis which contributed to the performance of the Vanilla LSTM. This inclusion may have improved the performance of the Vanilla LSTM to the extent which prevents relevant and compatible comparisons to be drawn between the performance of Vanilla LSTM’s and other LSTM variants on regression problems with financial timeseries data. Sagheer and Kotb found stacked Vanilla LSTM’s outperformed other models such as statistical models, neural networks and hybrid approaches when predicting petroleum production (Sagheer & Kotb 2019). Although displaying the superior performance of the Vanilla LSTM, the difference in behaviour between petroleum production and financial metrics data such as stock prices lowers the relevance of this result for the purposes of this study. Cen and Wang explore the performance of Vanilla LSTM models on crude oil prices (Cen & Wang 2019). They find Vanilla LSTM’s outperform standard approaches. This further supports the high performance of Vanilla LSTMs. However, a limitation exists in that transfer learning is utilised to enhance the performance of the LSTM model. The use of transfer learning skews the results of this study reducing its compatibility when measuring against other LSTM variants, leaving a gap in the literature to fill. Liu compared Vanilla LSTM’s to Support Vector Machines (SVM’s) on regression problems such as financial stock volatility to find that although comparable in performance, there are occasions when Vanilla LSTMs can be useful in predicting stock volatility when SVMs are struggling to provide adequate results (Liu 2019). This study compares the Vanilla LSTM with SVMs and also Generalized Autoregressive Conditional Heteroskedasticity (GARCH) models but does not compare Vanilla LSTMs to other LSTM variants, providing an inter-model comparison rather than intra-model comparison, which leaves room to explore for the purposes of this study. Rodriguez et al. employs Vanilla LSTM’s to improve the prediction of taxi demand (Rodrigues, Markou & Pereira 2019). However, this approach is incorporated with CNN layers, word embeddings and attention mechanisms which veer away from the pure evaluation of the Vanilla LSTM, not to mention the absence of comparison across different LSTM variants. Lei, Liu and Jiang utilise LSTM as a fault classification tool on energy systems outperforming state of the art approaches (Lei, Liu & Jiang 2019). However this approach does not compare the Vanilla LSTM to other variants but merely against other state of the art approaches to fault detection. This study also focuses on classification as opposed to regression which is outside the scope of this study. Hong, Wang and Yao also apply Vanilla LSTM in order to predict voltage of electric batteries to find LSTM provides a strong predictive performance (Hong, Wang & Yao 2019). This approach also includes classification as opposed to regression. Yang, Zhao and Guo compare the performance of an advanced LSTM network with a Vanilla LSTM network (Yang, Guo & Zhao 2019). Changes are made to the advanced LSTM in order to take advantage of the correlation between the sensors in the study as well as the use of a sliding window for fault isolation. This is then compared with Vanilla LSTMs as well as SVMs, with superior performance observed in the advanced LSTM network. This piece of work does compare two variants of LSTM, but is focused on the classification space as opposed to the regression space. Uddin uses wearable sensor data through a Vanilla LSTM network to classify human activity types based off multiple sources of sensor data (Uddin 2019). This resulted in superior performance in the Vanilla LSTM network compared to other traditional approaches. Similar to Lei et al, Hong et al and Yang et al, Uddin focuses on classification as opposed to regression which allows room in the literature to explore regression approaches to LSTMs. Similarly Uddin does not compare across different LSTM variants for performance benchmarking and best practise. Finally the compatibility of the results based off classification do not align with regression results, which creates difficulty when attempting to answer the questions of this study using Uddin’s work. Elsheikh, Yacout and Ouali utilise LSTM networks when predicting the remaining useful life (RUL) of physical systems (Elsheikh, Yacout & Ouali 2019). The difference in this approach is the use of a Bidirectional LSTM which contains two different input directions, the feedforward direction processing the inputs before using the LSTM final state to initialise backward processing cells. The benefit of this approach provides two different mapped values to the actual RUL which can be optimised to predict the RUL. Although this may provide a comparison of Vanilla LSTMs and Bidirectional LSTMs, the study of Elsheikh, Yacout and Ouali limits the relevant comparison of Vanilla LSTMs versus the other LSTM variants selected in this study. Graves and Schmidhuber compare the framewise phoneme classification of Bidirectional LSTMs vs Vanilla LSTMs and RNNs (Graves & Schmidhuber 2005). Graves and Schmidhuber found that the Bidirectional LSTM model outperformed the Vanilla LSTM and regular RNN in this regard. Although this finding compares two variants of LSTMs, it compares them using framewise phoneme classification as opposed to regression. Sang and Di Pierro compare the performance of Vanilla LSTMs to traditional financial naïve trading strategies which are commonly used consisting of Simple Moving Average (SMA), Relative Strength Index (RSI) and Moving Average Convergence Divergence (MACD) (Sang & Di Pierro 2019). They found the Vanilla LSTM consistently outperformed the traditional strategies when measured by trading performance in dollars. Sang and Di Pierro highlight the strength of Vanilla LSTMs when predicting time series financial data, but do not compare this LSTM variant with other LSTM variants; therefore the findings of Sang and Di Pierro suffer from relevance to the scope of this paper from a validity perspective as well as suffering from a comparability point of view due to the different units of measurement utilised of trading performance in dollars as opposed to mean squared error.

### 2.2.2 B-LSTM

As previously mentioned above, Elsheikh, Yacout and Ouali along with Graves and Schmidhuber provide comparisons between B-LSTM’s and Vanilla LSTM’s in specific scenarios such as RUL and framewise phoneme classification, two methods which differ from the proposed problem set of this study. Mardjo and Choksuchat compare the performance of B-LSTM with LSTM-AM on Bitcoin stock price prediction (Mardjo & Choksuchat 2024). This study found the B-LSTM model outperforms the LSTM-AM in this regard, however the compatibility and relevance of this study is limited due to the hybrid approach of the authors. Both LSTM models were combined with classical forecasting methods such as ARIMAX and GARCHX as part of the methodology which can skew results beyond the point of validity for this particular study. Shi, Jen, Che et al. used a B-LSTM in order to classify medical records for the purpose of enhancing the efficiency of medical triaging (Shi, Ye, Chen, *et al.* 2023). Despite the progress made in this paper in improving the efficiency of medical triaging, the authors deal with a classification problem which is irrelevant to the study of the author. In addition to this Shi et al. feed a multivariate dataset into the models which contrasts with the pure univariate dataset approach used for the purposes of this particular study, lowering the relevance of their findings further. Dash, Sahu and Mishra compare the performance of B-LSTM, Vanilla LSTM and other models such as Support Vector Regressors (SVR’s), Artificial Neural Networks (ANN’s) and ARIMA across a range of metrics including R Squared (R2), Mean Squared Error (MSE), Mean Poisson Deviance (MPD) and Mean Gamma Deviance (MGD) when predicting Forex trading closing prices (Dash, Sahu & Mishra 2023). Dash, Sahu and Mishra found the MSE performance of B-LSTM was superior to Vanilla LSTM across different currencies, providing a relevant comparison for the purposes of this study. Where the relevance of Dash, Sahu and Mishra’s study falls short is their two stage combination of classification and regression approach pursued as part of the problem set, which doesn’t align to the single stage regression approach being undertaken for this particular study. Dash, Sahu and Mishra compare several models but step outside the scope of LSTMs, and instead focus on two LSTM models as part of a broader search for optimal predictive models on FOREX trading providing less depth into the LSTM models surveyed as part of this particular study. Silvan and Meneses compare B-LSTM to Vanilla LSTM to predict electricity consumption across four unrelated datasets in order to understand the superior performing and most robust network (Silva & Meneses 2023). Silva & Meneses found B-LSTM’s consistently performs at a superior level to Vanilla LSTM’s on these regression tasks, which takes in univariate datasets similar to this study. However the metrics used by Silva & Meneses differ to those chosen for this particular study, giving rise to the issue of compatibility between the studies. Silva & Meneses use Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), R2 and Mean Absolute Percentage Error (MAPE). Although these metrics crossover with metrics used in this particular study, the absence of MSE as a metric prevents the most relevant comparison between the studies in order to achieve the research objectives of the author (the author acknowledges RMSE can be squared in order to compute MSE). Vijayalakshmi, Thanga Ramya & Ramar K compare the performance of GRU’s and Vanilla LSTMs in a section of their study classifying traffic congestion states (Vijayalakshmi, Thanga Ramya & Ramar 2023). As part of their methodology, Vijayalakshmi, Thanga Ramya & Ramar K perform regression to predict the traffic flow values before using a CNN for classifying congestion status. Ultimately the proposed Stacked LSTM Auto Encoder with B-LSTM (SLSTM\_AE\_B-LSTM) outperformed the other models in the regression element of the study. GRU is noted to outperform Vanilla LSTM in this study in both a singular and stacked state. Despite the regression comparison between GRU’s and Vanilla LSTM’s along with the partial comparison of B-LSTM’s (via SLSTM\_AE\_B-LSTM) Vijayalakshmi, Thanga Ramya & Ramar K do not include the breadth of LSTM variants to be included in this particular study which limits their study’s relevance. The metrics used also exclude mean squared error, which provides compatibility issues with the research objectives of this study (the author acknowledges RMSE can be squared in order to compute MSE). Finally, the problem area of traffic value prediction can be described as embodying different traits and behaviours to financial timeseries data procured for this study. Aung, Pang, Chua et al. evaluate a B-LSTM model forecasting number of new Covid 19 cases over a fourteen day period in January 2021 (Aung, Pang, Chua, *et al.* 2023). Two B-LSTM’s were compared, (one with multiple variables, one with fewer variables) along with a classic ARIMA model. When measured by the median values for MAE, RMSE and Percentage Error (PE), it was found that ARIMA outperformed the other models in this particular task. B-LSTM with fewer variables outperformed the multidimensional B-LSTM model. Aung, Pang, Chua et al.’s study highlights limitations of B-LSTM compared to a classic forecasting model which can be considered for this particular study also, but does not compare the B-LSTM to any other variant of LSTM models, limiting its relevance and scope in relation to this project. The metrics utilised also exclude MSE which affects compatibility (RMSE values can be squared to provide MSE values however). Huang, Yang, Li et al. compare the performance of GRU, Vanilla LSTM, B-LSTM and a Sparse Autoencoder-Based B-LSTM (SAE-B-LSTM) using RMSE, MAE and R2 on wastewater flow-rate prediction (Huang, Yang, Li, *et al.* 2023). The SAE-B-LSTM is found to provide superior performance using the metrics included in the study, followed by Vanilla LSTM, B-LSTM and GRU respectively from an RMSE perspective. Huang, Yang, Li et al. include the performance evaluation of Vanilla LSTM, GRU and B-LSTM models on a regression task, however the three remaining variants within the scope of this study are not included by Huang, Yang, Li et al., leaving a gap in the literature to be explored. RMSE is utilised as a method of measurement by Huang, Yang, Li et al., however without squaring their results the metric is incompatible with the MSE metric chosen for the research objectives of this study.

### 2.2.3 GRU

Lakshmi and Maheswaran utilise GRU to predict the academic grades of computer

science and engineering students using four semesters worth of data (Lakshmi &

Maheswaran 2024). These predictions were then compared against the predictions of a

convolutional neural network (CNN) and a Vanilla LSTM with L1-Norm optimization. The

authors found the GRU model outperformed the other two models. This paper

compares the GRU model to Vanilla LSTM with the GRU model displaying the superior

performance, which may partially inform the research objectives of this study. However

the lack of other LSTM variants analysed leaves open the possibility to explore LSTM

variant performance further. The authors also did not go into detail on the factors

contributing to the superior performance of the GRU model over the Vanilla LSTM which

is another area which can be explored. Shahid et al. compare the performance of GRU

and Vanilla LSTMs in predicting wind power generation (Farah, David A, Humaira, et al.

2022). These forecasts were made at both daily and multi-step hour intervals. The

authors found GRU outperformed the Vanilla LSTM model while learning faster than

Vanilla LSTMs over longer sequences. These findings concur with Lakshmi and

Maheswaran’s findings on GRU’s superior predictive performance to Vanilla LSTMs.

Similarly to Lakshmi and Maheswaran, Farah et al. provide partial evidence which can

inform the research objectives of this particular study based on the comparison of two

LSTM variants. However, in order to draw a comparison across the full extent of

identified LSTM variants included in this study, more variants are required to be selected

and compared. Bai et al. compare the forecasting performance of RNNs, Vanilla LSTMs

and GRUs with regard to enterprise electricity consumption (Bai, Xie, Liu, et al. 2021).

This study was conducted using data from three different Chinese enterprises. The GRU

and Vanilla LSTM models were presented as outperforming the RNN from the point of

view of root mean squared error, mean absolute percentage error and threshold

statistic. Due to its simple structure and prediction performance, the GRU model was

recommended by the authors as the first choice for practitioners to consider when

predicting enterprise energy consumption. The findings of Bai et al. concur with

Lakshmi and Maheswaran as well as Farah et al. This study possesses the same

limitation in that its analysis compares two LSTM variants, as opposed to the six which

are to be explored as part of this study. Dutta et al. compare the performance of GRUs

on cryptocurrency prices against other models such as a simple neural network (RNN),

a Vanilla LSTM, and other variants of GRUs (Dutta, Kumar & Basu 2020). The

performance of each model was measured using root mean squared error (RMSE). GRU

was shown to outperform the Vanilla LSTM and the standard RNN. GRU with a recurrent

dropout rate was shown to have the highest performance among the different

approaches on the specific cryptocurrency price prediction problem. One possible

explanation provided by the authors behind the GRU with recurrent dropout

outperforming the Vanilla LSTM was the superior computational speed of GRUs with

their lesser number of gates and tensor operations. A strength noted of GRU’s was

learning better with less training data vs Vanilla LSTMs where Vanilla LSTMs may be

more efficient in remembering longer sequences. The findings of this study provide

fruitful information as they compare two relevant LSTM variants on a regression problem

featuring financial data (cryptocurrency prices). A comparison in performance is drawn

between the two relevant LSTM variants as well as an outline of their relative strengths

and weaknesses which may have contributed to the recorded outcome. A different

measurement in RMSE was utilised to assess model performance by Dutta et al.

compared with this study (mean squared error) affecting its compatibility, however the

approach taken by Dutta et al. presents a pathway and methodology which can be

followed by this study when extended to the other LSTM variants.

### 2.2.4 LSTM-AM

Lin et al. propose an LSTM-AM to tackle the topic of electricity consumption prediction (Lin, Cheng & Huang 2020). The benefits of the LSTM-AM include its ability to assign weights to the input sequence data. Their study found that LSTM-AM outperformed the state of the art by 6.5%. While this study demonstrates the strength of LSTM-AM models, it does not compare this model against other LSTM variants in order to paint a picture of the LSTM variant field. This lowers the relevance of Lin et al’s study in the context of this paper. Lin et al in a separate paper employ an LSTM-AM to predict zonal electricity loads (Lin, Ma, Zhu, *et al.* 2022). The results of this model were measured using a pinball loss function and found the LSTM-AM displayed higher accuracy and generalizability compared with state of the art forecasting models. Similar to the study by Lin, Cheng and Huang, Lin et al’s, paper demonstrate the strength of the LSTM-AM model in prediction without delving into comparisons with other LSTM variants, providing some context on strengths for RO2 of this paper, but not containing the breadth of LSTM variants required. Abbasimehr and Paki implement a multi head LSTM-AM in predicting timeseries data (Abbasimehr & Paki 2022). This approach was benchmarked against state of the art approaches using multiple datasets and measured using symmetric mean absolute percentage error (SMAPE). This approach was deemed to outperform the state of the art methods based on average rank across the datasets. This approach differs to Lin et al in that the attention mechanism is multi-head as opposed to regular attention mechanism. Multi-head attention acts as multiple attention mechanisms which can average or concatenate multiple learned representations in order to capture additional complexity in variable relationships. Abbasimehr and Paki highlight strengths of the LSTM-AM but do add additional complexity with the use of multi-head attention mechanisms, which may skew a relevant comparison between the standard LSTM-AM and other LSTM variants from a validity perspective. Additionally Abbasimehr and Paki do not compare this version of LSTM-AM to the other LSTM variants which this study is concerned with. Ran et al. employ an LSTM-AM in order to predict travel time using data from Highways England (Ran, Shan, Fang, *et al.* 2019). The attention mechanism was focused on departure time, an element which the authors felt was missing from previous practise. This model’s performance was compared to benchmarks such as linear regression (LR), seasonal autoregressive integrated moving average (SARIMA), K-nearest neighbour regression (KNNR), Vanilla LSTM among other models. The results show the LSTM-AM outperforming the benchmark models as measured by mean absolute error (MAE), mean absolute percentage error (MAPE) and root mean squared error (RMSE). This paper provides a comprehensive set of benchmark models which to compare the LSTM-AM performance against and provides a comparison of LSTM-AM and Vanilla LSTM which can partially inform the objectives of this study. However the breadth of LSTM variants required to satisfy the objectives of this study are not provided, nor is the depth of reasoning behind the comparative performance of the LSTM-AM and Vanilla LSTM networks apart from the inclusion of the attention mechanism on departure times.

### 2.2.5 LSTM-FGB

Chien et al. propose an LSTM with a power law forget gate bias in order to control the level of decay which occurs from forget gates within LSTM networks (Chien, Turek, Beckage, *et al.* 2021). This allows the network to adjust its information decay rate based on the task at hand through the control provided by a law decay factor p. This LSTM-FGB was compared with Vanilla LSTM and other gated recurrent models. The results show the LSTM-FGB was superior in capturing long term dependencies, even with smaller datasets. This means LSTM-FGB is more robust when it comes to unit dropout. The LSTM-FGB was also seen to display temporal precision, allowing it to capture long term dependencies. Chien et al. manage to compare the LSTM-FGB with a Vanilla LSTM, evaluate its relative performance and theorise a factor behind its superior performance. Chien et al’s study does not contain all the LSTM variants included in this study, however it compares the performance of two LSTM networks across regression, classification and copy memory tasks which spans a broader range of use cases compared to the scope of this study. Mahto et al. compare the performance of an LSTM-FGB and Vanilla LSTM on language models (Mahto, Vo, Turek, *et al.* 2021). The authors impose an inverse gamma distribution on the model during training in order to create the forget gate bias. The results show the LSTM-FBG outperforming the Vanilla LSTM on correct output by 10% when faced with more than 75 timesteps, demonstrating its superior ability to capture temporal dependencies. Mahto et al. utilise textual data in order to test the two LSTM variants, which focuses on a different application of LSTMs in terms of data type and output to this study. However, Mahto et al. provide a comparison of the temporal performance of the two LSTM variants which may inform the objectives of this study, if the temporal performance translates across datatypes. This finding could then be extended upon to compare the performance across all LSTM variants which fall within the scope of this paper. Hoedt et al. compare the performance of an LSTM-FGB with a novel Mass-Conserving LSTM (MC-LSTM) along with Neural Arithmetic Logic Units (NALUs) and Neural Accumulator Units (NAUs) (Hoedt, Kratzert, Klotz, *et al.* 2021). They found that although the LSTM-FGB outperformed the NALU and NAU models on addition tasks measured by mean squared error, the MC-LSTM achieved a superior performance. The MC-LSTM contains a property allowing it to conserve the mass of particular inputs, which renders it suitable for modelling physical systems which involve the conservation of mass such as hydrological systems which must conserve water mass. Hoedt et al. uncover an LSTM variant in MC-LSTM which may be well suited to a nuanced task, but they do not compare its performance on more general regression tasks, nor do they include many LSTM variants in this particular project. This bias can be seen in the superior performance of the LSTM-FGB when measured by Nash-Sutcliffe Efficiency (NSE) compared with the proposed MC-LSTM, which Hoedt et al. note is not the most significant metric in hydrology. Delving into a detailed comparison between LSTM-FGB and MC-LSTM along with the inclusion of other LSTM variants would aid in addressing the research objectives of this paper.

### 2.2.6 LSTM-FGB-AM

There is not a substantial amount of studies exploring the capability of LSTM-FGB-AM’s nor comparisons of LSTM-FGB-AM’s performance versus other LSTM variants. Salton & Kelleher touch on the LSTM-FGB-AM when introducing an attention mechanism which provides weighting to past information based what was persisted by the gating mechanism across timesteps, allowing information persisted over more timesteps to have a higher baring over the memory buffer while assigning a forget gate bias of 1.0 (Salton & Kelleher 2019). Although this approach combines an attention mechanism and forget gate bias, it is focused on natural language understanding which embodies different datatypes, behaviour and metrics (such as perplexity). For these reasons the findings of Salton & Kelleher may not be as relevant or compatible to this study.

## 2.3 Conclusion

What is clear from the literature review is there are numerous papers written comparing the performance of an individual LSTM variant against the Vanilla LSTM. This can be observed from Dash, Sahu and Mishra who compare a B-LSTM to a Vanilla LSTM, Lakshmi and Maheswaran who compare a GRU model and Vanilla LSTM to predict student grades and Ran et al. who compare an LSTM-AM against a Vanilla LSTM to predict travel times. There are relevant pieces of research which provide a pathway which can be followed for this study’s research objectives such as Dutta et al’s comparison of GRU and Vanilla LSTM for predicting cryptocurrency prices, outlining the strengths and weaknesses of the models as well as measuring prediction performance. Insight may also be taken from Mahto et al’s comparison of LSTM-FGB and Vanilla LSTM’s prediction performance on language models where a focus is given to the temporal performance of each model, provided these results translate across datatypes from textual to numerical data. The insights taken from Mahto et al and Dutta et al’s studies can thus be applied across the full scope of LSTM variants included in this study.

# Research Methodology

This section details the research methodology undertaken as part of this study. It incorporates quantitative research on secondary source data through the build of a coded artefact comparing model variant performance using numerical metrics along with qualitative primary research in the form of in-depth interviews with practitioners in the field of data analytics.

## 3.1 Secondary Research

The CRISP-DM process was followed in order to structure the secondary research of this study as outlined in Figure 1 below. The insights of this research were then combined with the output from the primary research to validate the findings of the coded artefact and paint a more comprehensive picture in achieving the research objectives of this study.

Diagram of a diagram of data

Description automatically generated

Figure 1: CRISP-DM Process Model for Data Mining (Wirth & Hipp n.d.)

### 3.1.1 Business Understanding

Prediction of stock prices and other financial data is a widespread practice observable in the finance industry. The goal of these predictions is to position organisations to profit from the difference between current stock and commodity prices versus the future prices of that same stock and commodities. This drives widespread use of forecasting models such as autoregressive integrated moving average (ARIMA) (Khashei & Hajirahimi 2017). In more recent years practices have evolved to incorporate deep learning neural networks (NN’s) to improve the predictions of these financial metrics in order to maximise profits for organisations. This drive for profit precipitates research in advancing the field of regression models in order to improve the accuracy of predictions regarding financial timeseries data.

### 3.1.2 Data Understanding

Brent Crude Oil Prices per Barrel in the EU was selected as the dataset for this study (Anon 2024). This dataset contained the essential ingredients in order to assess the performance of different LSTM variants, namely timeseries financial data which contained financial data at regular intervals along with datetime data. Licensing of this data was inspected to ensure it complied with the terms and conditions of the Federal Reserve Bank of St. Louis, the source of this data. Since the aim of this study is for educational and non-commercial purposes, the use of deep learning is deemed compliant with the usage policy of the bank. This was queried with academic faculty in CCT College Dublin to ensure compliance.

The Pandas library was used in order to perform exploratory data analysis as part of this project. Exploratory data analysis (EDA) was performed by importing the csv dataset, understanding the shape of the data, the levels of measurement along with the statistical properties of the data utilised. Seaborne and Matplotlib were selected for their capability to visualise data, aiding in EDA and evaluating whether data takes a gaussian shape, an attribute which determines future decisions such as the mode of normalising the data if appropriate. This will be discussed in more depth in the ‘Evaluation & Implementation’ section of this report.

### 3.1.3 Data Preparation

The Numpy library was used for its ability to create arrays and manipulate them before converting them into Tensorflow tensors as part of inputs for deep learning architectures. Numpy’s conversion of data from dataframes to arrays also allows for normalisation to take place on the data, which can be a powerful tool in ensuring the scale and outliers in a dataset do not introduce noise to deep learning models when training. Data was split into training and testing data in order to perform normalisation, fitting the normalisation method onto the training data before transforming the testing data. The normalisation method was fitted to the training data only in an attempt to avoid data leakage, which if leaked would result in better training results but poorer generalisation for any models trained on the leaked data. Standard Scaler (SS) was employed as the method to normalise the data due to its ability to preserve the properties of a normally distributed dataset (the data was found to be gaussian after visualising its distribution on the selected years of data). SS centers the data around a mean of 0 with standard deviation of 1 as outlined below where U is the mean of the training samples and S is the standard deviation:

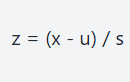


Figure 2: Standard Scaler formula (Anon n.d.)

A series of numpy operations were implemented in order to organise the data into a tensor named ‘final\_x\_test\_data’ which contains 20 arrays of timesteps 20 for the purposes of predicting the y test values (these y values were placed in a variable named as ‘testing\_data’). These operations were performed in order to satisfy the input requirements of Tensorflow.

### 3.1.4 Modelling

Tensorflow and Keras were imported for the building of the LSTM variants. Keras was selected because of its ability to simplify the build and experimentation of neural networks through its API and integration with Tensorflow. Tensorflow is optimised for high performance which aids in training deep learning models. Both Tensorflow and Keras contain library support for LSTM layers, simplifying the implementation of LSTM layers with room for customisation where necessary. Keras also contains advanced features for improving the performance of the Deep learning models such as Dropout to avoid overfitting of the data and Bidirectional LSTMs to facilitate forward and backward processing of the data to capture additional context, which can aid in the predictions and performance of the model.

#### 3.1.4.1 Vanilla LSTM

The Vanilla LSTM layer contains a forget gate, an input gate and an output gate as demonstrated in Figure 3 below. The forget gate decides which information should be discarded from the cell state through the output of a number ranging from 0 to 1 as a result of a sigmoid activation function, the formula for which can be seen below:



Figure 3: Sigmoid Function (Datta, Agarwal, Kumar, et al. 2019)

where 1 keeps all of the cell state and 0 removes all of the cell state info. The input gate decides which information from the input should be updated within the cell state. This gate contains an input layer with a sigmoid activation along with a Hyperbolic Tangent layer which performs hyperbolic tangent activation (TanH) using the below formula:

A mathematical equation with a plus and a line

Description automatically generated

Figure 4: Hyperbolic Tangent Function (Datta, Agarwal, Kumar, et al. 2019)

The final gate in the Vanilla LSTM layer is an output gate, which runs the input through a sigmoid function and TanH function before passing on to the output of the layer which is the hidden state.

A diagram of a tank

Description automatically generated

Figure 5: Example of a Vanilla LSTM layer

#### 3.1.4.2 Bidirectional LSTM (B-LSTM)

B-LSTM’s are a variant of the Vanilla LSTM which consists of two layers. The first layer processes input data from back to front similar to the Vanilla LSTM, but B-LSTM’s also contain a second layer which processes the input data backward from front to back. The outputs of the two layers are then combined into the hidden state.

A diagram of a cell

Description automatically generated

Figure 6: Example of Bidirectional LSTM (Mardjo & Choksuchat 2024)

#### 3.1.4.3 Gated Recurrent Unit (GRU)

The GRU provides a simpler architecture to Vanilla LSTM’s. GRU’s contain only two gates, a reset gate which controls which past information to exclude, along with an update gate which dictates which past information to retain and which new information to add to the long term memory. The GRU trains faster than a Vanilla LSTM as it has fewer parameters to train. An example architecture of a GRU layer can be seen below in Figure 7:

A diagram of a complex flow

Description automatically generated

Figure 7: Example of basic GRU architecture (Lakshmi & Maheswaran 2024)

#### 3.1.4.4 LSTM with Attention Mechanism (LSTM-AM)

The LSTM-AM allows models to focus on the important and relevant parts of input sequences in dynamic vectors as opposed to the Vanilla LSTM which captures the entire input sequence in one context vector. The LSTM-AM computes a weight to different hidden states assigning importance to each input sequence, and takes a weighted sum of the hidden states as its output:

A mathematical equation with numbers and symbols

Description automatically generated

Figure 8: LSTM-Am Formula (Mardjo & Choksuchat 2024)

This computation allows the LSTM-AM to reemphasise important parts of the input sequence, improving information retention and performance over longer sequences.

A diagram of a cell

Description automatically generated

Figure 9: Example LSTM-AM model (Mardjo & Choksuchat 2024)

#### 3.1.4.5 Forget Gate Bias (LSTM-FGB)

The LSTM-FGB involves initialising a bias on the forget gate in order to improve retention of information over longer sequences of inputs. The forget gate will not forget anything until it has learnt to do so as it is initially set to be almost 1.0 which results in the model behaving as a normal LSTM initially before learning how to forget (Gers, Schmidhuber & Cummins 2000).

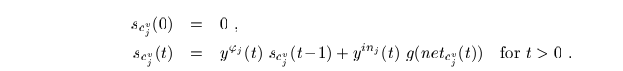


Figure 10: Extended LSTM algorithm to include a positive bias for the forget gate (Gers, Schmidhuber & Cummins 2000)

#### 3.1.4.6 Forget Gate Bias & Attention Mechanism (LSTM-FGB-AM)

LSTM-FGB-AM combines the variants of LSTM-FGB and LSTM-AM in order to gain the benefits of both approaches. This variant aims to focus on important inputs within the sequence over longer sequences as well as setting a bias on the forget gate in order to mitigate against the vanishing gradient problem. While the forget gate bias persists information over time the attention mechanism retrieves this information by averaging previous states. The below example was applied to a Language Model (LM) for a Natural Language Processing (NLP) task:

A diagram of a computer program

Description automatically generated

Figure 11: Demonstration of a model averaging previous states producing state h3 and combining with w3 (word 3) in a language model to produce w4 after using softmax. This model also sets a forget gate bias at 1.0 combining uniform attention and forget gate bias

### 3.1.5 Evaluation

#### 3.1.5.1 Validation

Due to the size of the models, input data and overall architecture, automated parameter tuning was employed in order to explore different combinations and architectures within the time constraints of the project. Automated parameter tuning can be implemented using cross validation techniques such as RandomisedSearchCV and GridSearchCV. Given the time constraints of the project, RandomisedSearchCV (RSCV) was selected for validation due to its ability to search a broader hyperparameter space in the same computation time as GridSearchCV (GSCV). GSCV may have provided a more thorough search for the optimal hyperparameters which is one limitation of RSCV, however RSCV was deemed the most appropriate method given the size of the hyperparameter space and time constraints of the project. Alternative approaches to hyperparameter tuning included Bayesian Optimisation, Hyperband and Genetic algorithms, however these approaches can be complex to implement, require careful parameter tuning of their own or may require many iterations to converge respectively, rendering them less favourable for the purposes of this short term project.

#### 3.1.5.2 Loss Function

Mean squared error (MSE) was selected as the loss function calculated in the output layer of each LSTM variant because of its centrality to the research title and topic under discussion and its standard use as a metric in regression tasks (Chicco, Warrens & Jurman 2021). LSTM variants were compared using MSE with the variant achieving the lowest MSE result deemed the superior performer in prediction.

#### 3.1.5.3 Metrics

Along with MSE, a second iteration was performed of the LSTM variants utilising additional metrics in order to provide a broader view of the variants’ all-round performance which included Root Mean Squared error (RMSE), R Squared (R2) and Mean Absolute Percentage Error (MAPE). RMSE is measured in the same units as the target variable, allowing for increased interpretability of model performance. RMSE also punishes larger errors in a harsher manner which can be desirable when the margin for error is small in problem areas such as commodity price prediction. MAPE is scale-independent as it calculates the percentage difference between predicted and actual values. This percentage reading is universal across scales and datasets allowing for improved comparability of performance across studies. Unlike MSE and RMSE, MAPE is less sensitive to outliers which can balance out the punishment served on larger errors by MSE and RMSE in assessing a model’s overall performance. R2 is easily interpretable as it provides a value between 0 and 1 with a clear threshold of 0.5 indicating a model’s ability to explain the variance in target variable outputs. This metric can also be compared across models for performance evaluation.

#### 3.1.5.4 Reproducibility

The consistent selection of libraries, LSTM variants and metrics allowed for reproducible experimentation across different projects. However, since no random seed was set the metrics changed with each running of the code, throwing into question the consistency, reliability of the results as well as reproducibility for the purposes of comparison and validation. Two separate iterations were implemented, with a broader set of metrics utilised for the second iteration in order to aid the comparability of these models versus the state of the art, as well as to provide limited validation over the output results.

### 3.1.6 Deployment

The deployment of this project consists the presentation of results and background in the form of this report, along with a Jupyter Notebook file containing the coded artefact. This report presents the relevant background, motivation, methodology employed along with implementation, results and discussion while the notebooks contain the coded artefact itself, displaying the build and performance of the LSTM models along with comments inline to provide additional context.

## 3.2 Primary Research

### 3.2.1 Population of Interest - Data Science professionals

The research objectives to be fulfilled during this project with regard to LSTM variant predictive performance are:

1. *Identify optimal variant hyperparameter configuration*
2. *Evaluate variant strengths and weaknesses*
3. *Identify the variant which produces the lowest mean squared error*
4. *Evaluate the factors behind the superior performance of the chosen variant*

Due to the technical nature of these research objectives, individuals with technical expertise in the domain area of data analytics/data science and specifically deep learning neural networks were required in order to validate model findings and add insight toward the fulfilment of the research objectives. These individuals worked in a role relating to data science such as data scientist, data science lecturer/professor, data analyst, machine learning engineer and data engineer among other titles as well as containing relevant education or experience in this field. These individuals were expected to be able to understand the domain area under consideration in this project along with the regression modelling using LSTM networks. These individuals were able to validate model findings, understand its implications and suggest areas of improvement in parameter optimisation, methodology employed, technical nuances to consider among other relevant areas of interest.

### 3.2.2 Sampling Method – Non-Probability

A non-probability sampling method was pursued based on the population of interest and form of information sought. The identified population of interest consisted of individuals with specific knowledge or expertise in a given area, namely data analytics. The information sought from this population consisted of validation and areas of improvement on LSTM models employed by the author which is qualitative in nature. Providing limited choices in the form of a survey/questionnaire to a large number of data scientists would have limited the scope of exploratory feedback which could be given on the models. To gain statistical significance, the sample would also have required such a size which rendered this approach impractical in sourcing the sample population. This specific population requirement created a bias in the sourcing also which moved the sampling method away from probability methods and toward a more suitable non-probability approach.

### 3.2.3 Sampling Type – Judgement Sampling

Judgement sampling was pursued as the sampling type based on the sample population and the sampling method employed in this research. It has been established that data scientists and those in similar roles are required to provide relevant information to fulfil the research objectives of this study. Non-probability sampling took into account the nature of information required and the bias in sampling which was employed. Judgement sampling was selected as the sampling type to ensure the sample units possessed the relevant job title, education and area of expertise in order to provide relevant information for the study. This was ascertained through job title searches within the organisation in which the author is employed. Due to limited responses from potential interviewees searches were also conducted on LinkedIn and other professional channels, where job title and work experience of individuals are publicly displayed and accessible. This led to more interviewee responses but required another iteration of sampling, as it was found LSTM’s as an area of data analytics required specialisation for an interviewee to be able to add value to the research objectives of this study, therefore LinkedIn as a channel was used again with a specific search for profiles containing ‘LSTM’ were conducted. This keyword search was combined with the previous method of reviewing candidate profiles for relevant experience and education in order to curate interviewees with the required expertise to add value in achieving the research objectives of this project. Purposive sampling was not selected because the population of interest was not deemed to be inaccessible to the extent it need be pursued through an official channel, as well as the fragmentation of the data science community which would not congregate in one particular channel of access which may be more typical for purposive samples. Snowball sampling was not employed due to the caution which must be taken in snowball sampling to avoid sampling error and validity issues. Similarly the author believed relevant individuals could be sought through judgement sampling allowing more control over sampling as opposed to relying on 3rd parties providing additional samples through snowball sampling. Judgement sampling carried pitfalls in limiting the sampling range to the author’s network and carried bias as the author subjectively considered and selected individuals deemed appropriately knowledgeable in the area of data analytics to be included in the study. These limitations may have resulted in sampling error and prompted the author to carefully assess biases when selecting individuals to minimize bias where possible.

### 3.2.4 Primary Research Methodology – In-Depth Interviews

Semi-structured in-depth interviews were selected in order to drill down into the qualitative information being sought to fulfil the research objectives of this study. Within judgement sampling, options of research methodology included both in-depth interviews and focus groups. Due to the infinite numbers of approaches and optimisations which can be taken to build deep learning models, the feedback of the sample individual was deemed to be based on expertise but also subjective and individualistic. For this reason, focus groups were not selected as group think could take root, particularly when evaluating the minute details involved in constructing the deep learning models in terms of code. At the same time the number of different opinions supplied in the same room concurrently would make it difficult to capture and record multiple views on multiple details in the coded artifact, as well as attributing each view to the correct person within the focus group. Due to the specific sampling criteria it may also have proven difficult to organise a focus group on a time and date which aligns with the availability of a sufficient number of data scientists given the small size of the sample population sought. In-depth interviews allowed for a focused discussion with each individual, allowing for the collection of individualistic feedback on detailed nuances of the coding artefact and findings, along with the feasibility of recording these findings and attributing them to the correct individual. The in-depth interview also allowed for a customised line of questioning to be given to each individual interviewee based off of their initial responses, allowing the researcher to delve deeper into the motivations and reasoning behind the views of each individual around relevant areas in order to gain richer insights and accommodate the fulfilment of this study’s research objectives. There was also the opportunity to prepare a small number of questions for the beginning of the in-depth interviews in order to steer the conversations in a direction which focuses on the research objectives of this study, which resulted in the semi-structured approach taken.

### 3.2.5 Ethics & Legal Considerations

The paragraphs which follow discuss numbered sections of the European Commission’s ethics and data protection legislation as it is deemed relevant to this study, along with measures which were taken to minimize any potential impact in these areas.

#### 3.2.5.1 VII: Use of previously collected data

For secondary data as part of this research a dataset containing financial timeseries data was utilised. This dataset contains variables with numerical values at consistent time intervals which were recorded as part of the dataset. This data didn’t contain any personally identifiable information, and appropriate licensing of the dataset was sought. Assurances were made by the author the data was publicly available and didn’t contain any personally identifiable information which can be traced back to individuals. The data subject was a commodity to which a certain price is attributed to, whose data was knowingly published publicly for the purposes of public access by the relevant authority. Therefore legal and ethical risk was deemed low for this particular section. Assurances were made that findings reported related to the particular data at the specific time period under analysis with the stipulation that findings can change based on the analysis of a different time period or different industry, commodity or company.

#### 3.2.5.2 IV Data Protection by Design & Default

With the primary research undertaken in this project, ethical considerations can arise in the sourcing of research participants. This activity involved researching individuals, their job titles and relevant professional and educational backgrounds in order to establish suitability for project participation. Contacting individuals also involved procuring contact information for this individual, who may not have shared this information publicly for the purposes of being contacted for research projects such as this. The analysis and recording of interview insights can pose challenges from an ethical standpoint also if personally identifiable information is documented. Data minimisation was undertaken in order to mitigate these risks. Firstly when reviewing profiles of potential candidates for participation, only relevant information which needed to be considered for the purposes of assessing suitability was reviewed, such as job title, work experience and tertiary education. Club memberships, hobbies and secondary level education of individuals were not necessary to review in order to achieve this aim. When recording insights in the results section of this research, individuals’ names were not included nor their gender, sexual identity, religious affiliation or union membership in order to minimise data recorded and publicised in this research. Job title was recorded which can be seen as a form of pseudonymisation, a recommended practise of the European Commission. No mention was made of the organisations these participants are associated with, in order to further mitigate any risk of personal identification.

#### 3.2.5.3 III Pseudonymisation & Anonymisation

As previously stated, pseudonymisation was employed wherever necessary in order to protect the identity of individuals participating in this project. Including job title was not deemed specific enough to place unreasonable risk on an individual’s data protection rights if no other specific form of information is provided regarding the individual, with the exception of their insights which contribute to the study’s findings and research objectives.

#### 3.2.5.4 V Informed Consent to Data Processing

Data subjects were informed of their rights and provided with full information on the data processing activities undertaken as part of this project. This came in the form of email communication where the relevant information was shared with the individual, affording them time to review the process and assess their willingness to participate before providing or refusing their consent without any undue pressure. Participants’ interview responses were recorded, transcribed and stored on a personal laptop accessible to the researcher along with the job title of the participants. Both verbal recording and insights provided will be deleted from this device at the end of the project process, with the only remaining information to be contained within the project output which may be publicly shared by CCT Dublin after course completion. This output includes only job title and interview insights from the individual. Data subjects were informed of this process and afforded ample information and opportunity to consider whether this process is acceptable to them from a data protection rights point of view before deciding whether to provide their consent to participate.

#### 3.2.5.5 XIII Deletion & Archiving of Data

As previously discussed, the relevant information on the data subjects will be deleted from where they are stored – on a personal laptop accessible to the researcher (which contains adequate security protection) as soon as the project is submitted and completed, as the data will no longer be required for the purposes of the study. Any data stored with the use of cloud providers will be deleted as soon as it is no longer required for the purposes of the project, with the author ensuring the cloud provider adheres to European Commission legislation regarding data protection and ensuring all backups of the data are also deleted.

### 3.3 Summary

The purpose of this research project was to explore LSTM variants and their performance on financial timeseries data, while fulfilling the four research objectives outlined by the author. Relevant literature has been explored with gaps identified which are pursued in this research project. Data scientists constituted the population of interest, with a non-probability judgement sampling approach executed using in-depth interviews. Relevant legal and ethical considerations have been discussed with actions taken to remedy and mitigate any potential risks posed.

# Evaluation & Implementation

This section evaluates the implementation of the research methodology. Challenges faced will be discussed as well as the solution implemented to each challenge and how this compares to the original plan set out in the research design.

## 4.1 Cleaning

### 4.1.1 Missing Values

A series of Pandas operations were conducted to investigate and clean the data in order to find a data representation suitable to be fed into the LSTM models which were built. After importing the dataset as a pandas dataframe, the .info() method was used to gain an overview of the data, with the aim of this approach to understand the number of columns, observations in the pandas dataframe along with the datatypes of each feature and whether there were null values. What was first observed was the datatypes of both features in the dataframe are in a string format, the target variable ‘DCOILBRENTEU’ needed to be altered to a numeric format such as floats before being fed into the LSTM models, which is the expected input of the models. Without this float datatype present in the target variable the models may simply produce type errors, incorrect loss calculations and training instability. The next task was to identify null or NaN values which introduce noise or create problems in running the LSTM models. Since the datapoints are taken over a range of dates, these dates were broken into year, month and day in order to isolate and validate for missing values. The year was extracted from each observation in order to count the values in each year as a method of identifying missing values. The logic behind this move was that all years should have similar number of values (trading days) and any substantial variance would indicate missing data in the years with substantially less recorded values. A variance of one or two days between years was considered explainable through leap years and public holidays in the EU, along with the day of the week each year happens to begin with over the course of the 37 years extracted. The first and last years in the series, 1987 and 2024 were deemed incomplete or missing data due to their substantially smaller size of observations. This could also be attributed to the data recording beginning sometime into the year of 1987 and the end of the series ending before the end of 2024 (since 2024 had not ended yet at the point of this study taking place). However, for cleanness of spread across years, the two smaller years were removed from the dataset. This operation was repeated across month, where the difference in observations for each month were not deemed substantial enough to remove any month from the series. February contained the lowest number of entries but this can be attributed to the month of February containing fewer days (typically only 28 days, with 29 days every four years), therefore no months were removed from the dataset. Removing a month would also upset the time interval of the dataset since trading days across every week of each year were constituting the intervals. Extracting the day values proved illuminating in understanding missing values in the dataset. Despite the number of day values being deemed equal across the dataset, a simple .head() call showed there were days with no numerical value, despite not being strictly classified as null or NaN. These entries were filled with special characters in the place of a numerical value, which would upset an LSTM model as they are not numeric as previously explained. Since these non-numerical values were not showing up as NaN, a separate column was created which displays all values from the target variable to either its genuine numerical value or a NaN value. This would allow for the isolation of the special character entries for evaluation and imputation if deemed appropriate. Once the NaN values were found, they were displayed per year, per month and per day for any discernible patterns or specific periods with missing data. The year search found no discernible patters, with the months search showing December missing more data than other months, which could be attributed to public holidays across the EU during this month each year. The days found Monday and Friday over-indexing on missing values, this can again be attributed to public holidays which tend to be held on these days across the EU. The ‘DCOILBRENTEU’ column was renamed to ‘price’ for simplicity with all unnecessary columns removed as the missing values were now found, leaving only the ‘DATE’ column and ‘price’ column. To address the missing values, imputation through forward filling was chosen. Forward fill was opted for as the method of imputation instead of the long mean of the dataset because of the temporal nature of timeseries data which would be lost holding a mean value in certain parts of a dataset which may be out of step with the values around it at that particular time in the timeseries period, particularly considering the scale of change in the target values over the three decade period of the dataset. Back fill was another method which was considered, but was not opted for as it can introduce unrealistic expectations in the present value based off future values. It is worth noting forward fill also has limitations in that it can create bias in the data where trend or seasonality is present. Due to an opening observation containing a missing value, this was back filled as there was no preceding value with which to forward fill it.

### 4.2 Exploratory Data Analysis (EDA)

The data was then visualised in order to understand the distribution of values and statistical nature of the dataset, which would inform future operations such as mode of normalisation. This visualisation can be seen below through a histogram with a kernel density estimate (KDE) which was selected for its ability to offer a quick overview of the distribution of a dataset in an interpretable manner:

A graph of a distribution of price

Description automatically generated

Figure 12: Distribution of the data from 1988-2023

As can be seen in the dataset above, the oil prices skew heavily to the left, which can result in issues around regression problems. Skewed data can create difficulty for models to capture the signal in the data, or may result in the model overcompensating in order to fit to noise resulting in overfitting in the training period, which brings sub-optimal test results. A further visual was made in order to confirm the distribution of this data with more delineation through the form of a boxplot which captures the quartile distribution of the data: A blue rectangular object with black lines

Description automatically generated

Figure 13: Boxplot showing the distribution of price values

This visual confirms the left skew of the dataset. Further statistical properties of the data were sought in order to understand the optimum way forward for this dataset through the calculation of the interquartile range, which helps highlight outliers in the dataset, however no outliers were detected, meaning no further consideration was required regarding outliers at this time. Since the dataset is in the form of a timeseries, a scatterplot was created to show the distribution of the data over time in order to further assess the characteristics of the data:

A graph showing the growth of the stock market

Description automatically generated

Figure 14: Scatterplot of oil price over the timeframe of the dataset

Given the volatility of the data after the year 2000 and the performance of initial models, it was decided to drop all years except those between 1988-1998. This time period was deemed to have sufficient data to train LSTM models and make adequate predictions with stable data, which will prevent problems such as under and overfitting which can occur with more volatile data. Reducing the size of the dataset was also deemed suitable in order to reduce model training times which were placing pressure on the timelines of the project. Furthermore, the shape of the data became more gaussian once the dataset was sliced as described which can be viewed in the following boxplot and histogram:

A graph with a bar chart

Description automatically generated with medium confidence

Figure 15: Boxplot of the Oil Prices distribution post-slicing to 1988-1998

A graph of a distribution of a number of bars

Description automatically generated with medium confidence

Figure 16: Histogram of distribution of Oil Price values post-slicing to 1988-1998

The new gaussian shape of the data aids in LSTM models as the models often assume a gaussian shape, meaning gaussian inputs can result in improved model performance in the form of accurate predictions and faster convergence. Furthermore, gaussian datasets can be robust and are less prone to outliers mitigating against poor generalisation.

### 4.3 Data Pre-Processing

The decision was made to predict the last 20 values (days) in the timeseries, which constitutes a full month in trading days and therefore an intuitive test size for interpretability. Training data made from all the oil price values in the dataset minus the final 20 values were created along with the test data which constituted those final 20 values. These would be built upon later to include timestep x values and the next respective y value. After converting the oil prices into an array and making 2 dimensional (2D) which is required for neural networks, the data was normalised through the use of standard scaler (SS). The reason standard scaler was chosen as the method of normalisation over min-max scaling was the suitability of SS for dealing with gaussian data which the dataset now took the shape of. The timesteps for the x training data and y training data were then created in the form of python lists through the use of a for loop with 20 given as the number of timesteps. 20 was chosen as a standard figure for the number of timesteps which similar to the test data, corresponds to four trading weeks, constituting a month, an intuitive period of time. No formal process was explored for arriving at an appropriate timestep number as this intuitive reason was deemed sufficient. The x training data list was converted to 3 dimensional (3D) shape in order to comply with the recurrent neural network library requirements which will be discussed in a later section. The final\_x\_test\_data variable was created through concatenating the final 20 values of the training data with the 20 values within the test data before building on this by creating 20 arrays of 20 timesteps which precede each of the 20 test values to be predicted. This variable was then converted to 3D to be compatible for the neural network (NN). The y test values were in a variable named ‘testing\_data’. It was suggested through the interviews that data leakage may have occurred due to some training data being present in the test data. After investigation the author believes this is confirmed as the x\_test\_data variable contains the last 20 values of the training data along with the 20 y test values to be predicted. What would remedy this issue is to have the training data end at the 41st last value in the original oil price array instead of at the 21st last value which was immediately before the predicted y test values. This remedy would allow for the 20 values before the y test values to be included in the final\_x\_test\_data variable as timesteps for the predicted y test values without having been included in model training and leaking trained information into the test set. Unfortunately due to time constraints the author was unable to implement this remedy although it is noted as a proposed fix of this issue for future reference. The impact of this data leakage can be seen in the test results’ graphs which show a lag between a change in real values and change in predicted values which is present in multiple LSTM model variants’ output. The metrics may also show a more optimistic picture of model performance during testing which may not be applicable to real world unseen scenarios where this data leakage is not possible.

### 4.4 LSTM Models

#### 4.4.1 Libraries & Tools

The Sequential class was used to build layer upon layer within a neural network in an intuitive manner, this allows for the incremental build of a network. Necessary layers and classes such as Dense, RNN, LSTM, Dropout and Input were included for building a complex RNN network with LSTM layers, with the possibility of regularisation and an output layer. Scikit-learn (Sklearn) was selected for its robust capabilities around model evaluation and selection, through tools such as RandomizedSearchCV (RSCV) and GridSearchCV (GSCV). Sklearn also contains TimeSeriesSplit which allows cross validation to be performed on timeseries data while respecting the temporal nature of the data, which is critical in timeseries tasks. Sklearn also provides the essential metrics required to achieve the research objectives of this project through measuring the mean squared error (MSE) of each LSTM variant. SciPy includes parameter optimisation tools such randint and uniform which enable hyperparameter tuning through generating random numbers. SciKeras was also imported due to its ability to accommodate wrapping complex neural networks in the form of Keras models into the sklearn framework, aiding in cross validation and hyperparameter tuning. The combination of the above libraries were proven useful in efficiently creating multiple LSTM models with different variants, wrapping the models into a KerasRegressor to perform automated cross validation, specifying the hyperparameter search space and fitting a random search onto the training data to discover optimal hyperparameters.

#### 4.4.2 Baseline LSTM Build – Vanilla LSTM

The Vanilla LSTM was built using a build\_rnn function which set initial parameters (units = 5, dropout\_rate = 0.2, optimiser = Adam, num\_layers = 2). The initial parameters were deemed acceptable starting points which can be explored in the hyperparameter search space as part of cross validation. An initial LSTM layer was built using the LSTM class followed by a loop which added more LSTM layers depending on the number of layers to be added as defined in the hyperparameter search space for cross validation. A standard output layer of 1 unit was added which is the standard output for a regression model. The loss function was hard-coded as MSE given the centrality of this metric to the objectives of this research.

##### 4.4.3 Cross Validation

Within the hyperparameter search space multiple hyperparameters were included in order for the model to thoroughly explore available permutations in order to optimise predictions. The number of nodes in each layer was defined as model\_units where a search was performed of integer values between 5 to 50 inclusive. This range adequately covered a realistic number of nodes, with the only limitation being a smaller number such as 1 may have provided a better low point as models can and do contain layers with a singular node. However due to the expected complexity of an LSTM network, 5-50 was deemed a more ambitious range in order to capture complexities in the data. The results also indicate the higher numbers in the 20+ range where finding superior accuracy compared to if the range was lowered down to 1. These results will be shown in a later section. The dropout rate was searched for between float values of 0.1-0.5 inclusive. This was deemed an appropriate range for the dropout range which typically contains a value of 0.2, which falls between this range. This range was deemed to provide adequate leeway either side of the typical 0.2 value and the results supported this choice. The batch size searched between the three values 16, 32, and 64 with no number in between searched for. The choice to exclude any other number within this range may have provided a limitation but the author was content these three values were appropriate landmark values which gauge from small, medium to larger batch size performance. The results indicate the lower side of these figures provided the majority of the success, which would support the lowering of the search range as well as including all integers within the chosen range to be searched which could have been implemented using randint. The epochs searched between 10, 20, and 30 for the optimal number exclusively. The same observation may be made of epochs as was made with batch sizes, however the results show successful number of epochs tend to average in the middle of the three numbers specified. However, assigning a randint to search within the range between each of these individual values may have led to superior results. The model optimizers searched for include Adaptive Moment Estimation (Adam) and Root Mean Square Propagation (RMSProp). RMSProp works by adjusting the moving average of squared gradients, mitigating against the vanishing gradient problem. Adam is the default selection for LSTM tasks as it combines the benefits of RMSProp with Stochastic Gradient Descent (SGD) with momentum. SGD incorporates momentum to accelerate convergence and avoid local minima. The limitation of searching for only two optimisers may have limited the ability of the models to optimise their accuracy in this study. Other optimisers which could have been included are SGD, Adaptive Gradient Algorithm (Adagrad) which adapts the learning rate based on the frequency of updates and AdaDelta, which is an extension of Adagrad where it seeks to decrease Adagrad’s aggressive, monotonically decreasing learning rate to be more dynamic. Finally the number of layers were also searched for in the hyperparameter space. In this case the integers 1, 2, 3, and 4 where specified with no other values included. The results demonstrate the optimal number of layers generally falling comfortably within this range without placing upward pressure on the 4 value which would have indicated a higher range of values should be explored. For cross validation the data was split into 5 folds, which is a standard practise to begin with. RandomizedSearchCV (RSCV) was selected as the form of cross validation as explained in the methodology section. It was deemed to provide sufficient breadth of hyperparameter searching within the time constraints of the project. Using GridSearchCV (GSCV) may have improved results but would require further computational resources and time which was not deemed suitable for this particular project. Bayesian optimisation was suggested through the interviews as an approach to consider, however due to the complexity of implementation, this approach was considered less preferable. 10 iterations of RSCV were chosen as this number balances exploration with computational resources, all cores of the author were utilised with a specification of:

*11th Gen Intel® Core™ i5-11400H @ 2.70GHz 2.69GHz*

*8GB RAM*

*Windows 11 Version 23H2.*

All cores were deemed necessary in order to train the model within the time constraints of the project, despite the dataset not consisting of a particularly large size. This was in order to fulfil the hyperparameter search for cross validation. Once the model was trained, predictions were made on the test data before inverse transforming the output to de-normalise and reflect real values. A line plot from matplotlib is utilised to visualise the predicted values against the real values. Line plots are considered appropriate for displaying trends over time, which is compatible with the use case of mapping timeseries predictions over real data representing the same linear period of time. This visual will be displayed in a later section of the report. Finally, the metrics to evaluate the performance of the Vanilla LSTM were calculated and displayed. Initially only MSE was calculated as this falls within the scope of the research objectives as a key metric for evaluation. However, findings from the primary research suggested broadening the metrics utilised would provide a fuller picture of overall model performance. For this reason, R2, RMSE, and MAPE were also calculated and displayed. These results will also be shown in a later section.

#### 4.4.4 GRU

The data was re-normalised before going into each respective LSTM variant in order to mirror the same process as the Vanilla LSTM for comparability, as well as to optimise the performance of the models through the benefits obtained from normalisation. For GRU the GRU layer was imported from Tensorflow’s Keras API. The main difference between the implementation of the GRU variant versus the Vanilla LSTM was the use of GRU layers as opposed to LSTM layers as described previously. The same values were initialised for the hyperparameters as well as the same search space defined for cross validation on the GRU model. An attempt was made as much as possible to mirror the same conditions and configurations across all LSTM variants in order to provide a clear and fair basis of comparison, where a difference in configuration may bias the performance of one variant over another, preventing accurate real performance comparability. After training the model, predictions were made and de-normalised before being plotted on a line plot comparing the predicted oil price values to the real oil price values. Similarly the MSE, MAPE, RMSE and R2 were calculated and displayed. These standard steps were performed across all LSTM variants for fair comparison and consistency.

#### 4.4.5 LSTM-AM

The Input, Model and Attention classes were imported in order to implement the LSTM-AM. The Model class was used instead of Sequential because of the complexity an attention mechanism brings to a model. Model provides support for this kind of complexity in a way Sequential does not, as Sequential is more suited to simple linear stacked models. After the initial LSTM layer was built, a loop was run to build further layers based on the values of num\_layers in the hyperparameter search space, as this model would go through RSCV similar to the other LSTM variants. An attention mechanism was placed after the loop so the attention could take advantage of the context captured by the preceding layers in order to assign importance to inputs. After model training and prediction making, the results were visualised indicating inflexibility in the attention mechanism in this particular case versus the relative volatility of the real data.

#### 4.4.6 LSTM-FGB

For the LSTM-FGB the Constant class was imported from Tensorflow’s Keras API, which is required for adding the forget gate bias. The forget bias was also added as an additional parameter with a setting of 1.0 initially in order to provide a stable beginning for the model as the model learns a lot from the data initially with the forget bias kicking in as the model becomes more mature. The forget bias was added as a searchable term in the hyperparameter space in order to optimise its value. Uniform was used to search floats between and including 0.5-2.0 as this was deemed a reasonable range with which to set the forget bias. The forget bias was placed in each layer of the model in order to maximize its influence and effectiveness, with the exception of the output layer, where the hidden state is converted into an output. The LSTM-FGB was then trained with predictions made, data de-normalised and predictions visualised versus the real data over the same period for the accurate comparison across variants.

#### 4.4.7 LSTM-FGB-AM

The LSTM-FGB-AM model combined the two previous models in architecture and construction. When attempting to run this model, a TypeError was experienced stating ‘got an unexpected keyword argument’. This error was mitigated against and future proofed by including the ‘\*\*kwargs’ parameter in the initial function definition. This parameter allows the model to accept extra arguments which the function doesn’t explicitly handle. This action resolved the TypeError issue once implemented. Since this model contained forget bias, the forget bias was included in the initial setup of the model and throughout, with searchable values in the hyperparameter space for cross validation. These searchable values for the forget bias were kept consistent to the LSTM-FGB model previously built for comparability. This model appeared to struggle similarly to the LSTM-AM, where the model’s predicted values were too rigid compared to the volatility of the real oil price values. This will be explored further in a later section of the report.

#### 4.4.8 B-LSTM

The B-LSTM was more straightforward to build and implement in comparison to the LSTM-AM, LSTM-FGB or LSTM-FGB-AM models. The B-LSTM architecture more closely resembled the structure of the Vanilla LSTM and GRU models. The Bidirectional class was imported from the Keras API which provides direct support for B-LSTM layers. The only difference in the build of the B-LSTM and Vanilla LSTM is the inclusion of the Bidirectional class in each layer, coming before the LSTM class within the layer. Once trained and predictions made, the B-LSTM graph comparing predictions to real data came closer to embodying the performance of the Vanilla LSTM and GRU models implemented earlier on compared with the more complex models of LSTM-AM, LSTM-FGB and LSTM-FGB-AM. These findings will be discussed in the next section of the report.

# Results & Discussion

This section will discuss the results of the experimentation along with findings from the primary research and attempt to synthesize these findings. Model performance will be compared to the literature where comparable examples are available in order to validate their hyperparameter optimisation along with comparison of mean squared error across all LSTM variants to fulfil the objectives of this study.

## 5.1 Parameter Optimisation

### 5.1.1 Vanilla LSTM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Study** | **LSTM Variant** | **Root Mean Squared Error (RMSE)** | **R Squared (R2)** | **Mean Absolute Percentage Error (MAPE)** |
| Current study | Vanilla | 0.317 | 0.596 | 0.022 |
| Silva & Meneses | Vanilla | 0.476 | 0.752 | 0.254 |
| Dutta, Kumar, Basu | Vanilla | 0.024 |  |  |
| Ran et Al. | Vanilla |  |  | 0.073 |

Table 1: Vanilla LSTM: Study results versus the literature (Silva & Meneses 2023) (Dutta, Kumar & Basu 2020) (Ran, Shan, Fang, et al. 2019)

The Vanilla LSTM results of this study are compared to a sample of the literature above. This study is in the median range when comparing RMSE to the literature samples chosen. Although the study’s R2 score is lower at 0.596 compared to Silva & Meneses’ model at 0.752 the MAPE of this study is superior to both Silva & Meneses and Ran et Al’s models at 0.022 versus 0.254 and 0.073 respectively. Although different datasets with different characteristics and ranges were applied which limits the comparability of these figures, a brief comparison shows the Vanilla LSTM is performing in line with the literature. The use of RSCV may well be a factor in searching and obtaining optimal hyperparameters for the Vanilla LSTM in this study.

### 5.1.2 LSTM-AM

|  |  |  |
| --- | --- | --- |
| **Study** | **LSTM Variant** | **Mean Absolute Percentage Error (MAPE)** |
| Current study | LSTM-AM | 0.042 |
| Ran et al. | LSTM-AM | 0.070 |
| Mardjo & Choksuchat | LSTM-AM | 0.088 |

Table 2: LSTM-AM result from this study versus the literature (Ran, Shan, Fang, et al. 2019) (Mardjo & Choksuchat 2024)

Using MAPE, it can be seen that this study has outperformed the literature on the second run, with a MAPE of 0.042 versus Ran et Al’s 0.070 and Mardjo & Choksuchat’s 0.088. This indicates the hyperparameter search space effectively found hyperparameters for the LSTM-AM model as measured by MAPE which is a common metric across the studies. Other metrics such as RMSE were not used as the scale of the values appeared incomparable which can be interpreted as evaluated on datasets with different characteristics and ranges.

### 5.1.3 MSE Variant Comparison

The results of the first iteration of code running (before metrics other than MSE were included) are displayed below along with the optimal hyperparameters found for each variant according to the RSCV:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Rank** | **LSTM Variant** | **Mean Squared Error (MSE)** | **Batch Size** | **Epochs** | **Dropout Rate** | **Layers** | **Optimizer** | **Units** | **Forget Bias** |
| 1 | GRU | 0.100 | 16 | 20 | 0.11 | 2 | Adam | 45 | - |
| 2 | Vanilla | 0.104 | 16 | 20 | 0.32 | 1 | Adam | 30 | - |
| 3 | B-LSTM | 0.107 | 16 | 20 | 0.37 | 1 | Adam | 44 | - |
| 4 | LSTM-FGB | 0.153 | 32 | 30 | 0.11 | 1 | Rmsprop | 23 | 1.44 |
| 5 | LSTM-AM | 0.157 | 16 | 30 | 0.13 | 3 | Adam | 44 | - |
| 6 | LSTM-FGB-AM | 0.337 | 16 | 20 | 0.14 | 3 | Adam | 22 | 2.46 |

Table 3: LSTM Variant MSE comparison with Optimal Hyperparameters (Implementation 1)

GRU achieved the superior performance with an MSE of 0.100, followed by Vanilla LSTM with 0.104 and B-LSTM with 0.107. These three models performed above the rest and are placed closely together as measured by MSE. LSTM-FGB and LSTM-AM are paired closely together in the middle of the distribution with an MSE of 0.153 and 0.157 respectively. These were considerably behind GRU, Vanilla LSTM and B-LSTM in performance. A more substantial gap still was found between these middling variants and the worst performing variant, which was the LSTM-FGB-AM with an MSE of 0.337.

The visual comparison of predictions versus real data can be seen below for each variant along with the findings of the in-depth interviews against each research objective:

A graph showing the price of oil

Description automatically generated

Figure 17: Vanilla LSTM predictions vs real data

A graph showing the price of oil

Description automatically generated

Figure 18: GRU predictions vs real data

A graph showing the price of oil

Description automatically generated

Figure 19: LSTM-AM predictions vs real data

A graph showing the price of oil

Description automatically generated

Figure 20: LSTM-FGB predictions vs real data

A graph showing the price of oil

Description automatically generated

Figure 21: LSTM-FGB-AM predictions versus real data

A graph showing the price of oil

Description automatically generated

Figure 22: B-LSTM predictions versus real data



Table 4: Findings from the in-depth interviews with data analytics practitioners. Number of interviewees connected to each insight is given in brackets beside the insight

### 5.2 GRU

One immediate insight shown in the graphs is the visual of the GRU which achieved best performance via MSE at 0.100 is not showing as tight a fit with the real data. This is because this snapshot was taken on a different running of the code. What this may indicate is a range of implementations of the code and results need to be recorded to provide a comprehensive picture of variant performances, with distributions and mean performance calculated to ascertain the most robust and consistent variant. This was not possible due to the time constraints of this project and the training time required for each LSTM variant. However based on the first round of results recorded and shared above, the GRU has outperformed the Vanilla LSTM which aligns with the literature (Farah, David A, Humaira, *et al.* 2022; Lakshmi & Maheswaran 2024). This study has also found the GRU to outperform all other LSTM variants as measured by MSE which fills a gap in the literature regarding the comparison of GRU and these other LSTM variants. The strengths of GRU including efficiency and performance were highlighted with this dataset which didn’t challenge the weak flexibility of GRU models.

### 5.3 Vanilla LSTM

The Vanilla LSTM achieved a strong performance as measured by MSE at 0.104. The Vanilla LSTM achieving superior performance to B-LSTM’s goes against the literature which has B-LSTM as a superior variant (Dash, Sahu & Mishra 2023; Silva & Meneses 2023) in forex trading forecasting and electricity consumption prediction respectively. This finding also conflicts with the outcome of Ran et al. who found in their study the LSTM-AM performs to a higher level than Vanilla LSTM (Ran, Shan, Fang, *et al.* 2019). This contrast may partially be explained with the nature of data being predicted with Ran, Shan, Fang et al. constituting travel time versus oil price data of this study. In addition, the primary research found simpler models tend to perform in a superior way to more complex models when predicting on a simple dataset, which the oil price dataset is deemed to be, particularly once the dataset was isolated to 1988-1998 which gave a more stable and gaussian shape. LSTM-AM would be considered a more complex model versus the Vanilla LSTM which may explain the anomalous disparity between this study’s outcome and the broader literature. The weaknesses of the Vanilla LSTM regarding capturing more complex signals in the data versus more advanced LSTM variants was not tested here due to the simpler nature of the dataset, which allowed the Vanilla LSTM to leverage its capability to capture long term dependencies in the dataset.

### 5.4 B-LSTM

As previously mentioned, the B-LSTM was among the higher performing variants with an MSE of 0.107. However it did not perform as well as the Vanilla LSTM which goes against the literature as described in the Vanilla LSTM section. Its simpler structure allows the B-LSTM to fit the simple dataset in a more affective manner than the more complex datasets which correlates with the findings of the primary research. This can partly be explained by the B-LSTM’s ability to capture context through backward as well as forward processing which can lead to higher performance, despite being resource intensive and requiring higher training time than the standard LSTM model.

### 5.5 LSTM-FGB

The LSTM-FGB achieved an MSE of 0.153, a considerable distance from the top three performing variants. This lower performance versus a Vanilla LSTM conflicts with the study of Chien, Turek, Beckage et al. which found LSTM-FGB as outperforming the Vanilla LSTM (Chien, Turek, Beckage, *et al.* 2021). The more complex structure of the LSTM-FGB may not have been as suited to the simpler dataset, exposing weaknesses in the model in terms of finding optimal hyperparameter settings and overfitting despite its strength of initialisation of the forget bias for control. The primary research supports this finding stating more exploration may be required on the LSTM-FGB in order to arrive at more optimal hyperparameters, which may have been a limitation of the automated validation versus manual parameter tuning which was also found in the primary research. LSTM-FGB was also found not to be suitable for financial timeseries data, with a superior performance expected over non-financial data.

### 5.6 LSTM-AM

The LSTM-AM performed similarly to the LSTM-FGB with an MSE of 0.157. As previously mentioned the inferior performance of the LSTM-AM versus the B-LSTM and Vanilla LSTM conflicts with the findings of Mardjo & Choksuchat along with Ran, Shan, Fang et al. respectively (Mardjo & Choksuchat 2024; Ran, Shan, Fang, *et al.* 2019). One explanation for this is the simplicity of the dataset which is not what the LSTM-AM model is designed for. As was found in the primary research, the LSTM-AM has a strength of focusing on important inputs with multivariate data series as opposed to univariate. Once a univariate data series is fed into the LSTM-AM, the attention mechanism may begin focusing on different parts of the temporal sequence which is not conducive to optimal performance for the model.

### 5.7 LSTM-FGB-AM

The LSTM-FGB-AM was the worst performing model as measured by MSE with 0.337. The extra complexity associated with this model, combining the benefits of LSTM-AM’s and LSTM-FGB’s was not compatible with the type of dataset selected, resulting in performance significantly below the other LSTM variants. The high complexity of this model can also result in training difficulty along with its incompatibility to a simple univariate dataset. Despite the strong performance of LSTM-FGB-AM’s on natural language processing tasks found by Salton & Kelleher, this did not translate into a regression task with a relatively simple dataset (Salton & Kelleher 2019). The findings on the limitations of the LSTM-FGB and LSTM-AM models can be combined to provide further insights into the limitation of LSTM-FGB-AM within this study, with the biggest take away being the unsuitability of complex models to the simpler datasets.

### 5.8 Final Primary Findings

As well as the primary research findings detailed above, some final findings include the models for the most part are relatively well optimised with appropriate hyperparameter tuning (RO1). There is some room for further exploration of optimisers within the hyperparameter search space with Adam and RMSProp constituting a limited number to search. RMSProp was found the most suitable optimiser for LSTM-FGB with Adam found to be the highest performing optimiser for all other model variants within this study. The most supported finding regarding RO2, RO3 and RO4 is the suitability of the simple dataset for the simpler LSTM models such as GRU, Vanilla LSTM and B-LSTM over LSTM-FGB, LSTM-AM and LST-FGB-AM, with this difference in suitability reflected in the results as measured by MSE.

# Conclusion

This study was conducted to evaluate the performance of different LSTM models on financial timeseries data using mean squared error. The research objectives pursued as part of this research included:

1. Identifying optimal parameter settings for each LSTM variant to increase predictive performance via mean squared error
2. Evaluating the relative strengths and weaknesses of each LSTM variant in predictive performance via mean squared error
3. Identifying the LSTM variant which achieved the lowest mean squared error when predicting the financial timeseries
4. Evaluating factors drawn from RO2 which may have contributed to the identified LSTM variant outperforming other variants

Out of the 6 LSTM variants assessed as part of this study through mean squared error on the first run, the order of best performing models on financial timeseries data were as follows:

1. GRU
2. Vanilla LSTM
3. B-LSTM
4. LSTM-FGB
5. LSTM-AM
6. LSTM-FGB-AM

Cross validation was performed through RSCV in order to obtain the optimal parameters for each LSTM variant and maximise their performance. The primary research found this was broadly achieved through this methodology, with scope for further exploration around the use of optimisers and also the fine-tuning of the LSTM-FGB related models. The GRU was found as the superior performing LSTM variant due to its simple structure to match the relatively simple dataset the model was fed, along with the performance and efficiency benefits of GRU which was backed up by the primary research conducted. This trend of simpler models displaying superior performance on the simpler dataset were echoed across the other variants and supported by the primary research, with a clear distinction in performance between the simpler models and more complex variants. The more complex models such as LSTM-FGB and LSTM-AM were considered less suitable for the dataset due to the financial nature of the data which does not suit LSTM-FGB’s along with the univariate nature of the data which prevents the strengths of LSTM-AM’s being realised. In summary through the experimentation and primary research conducted, it can be said the four research objectives have broadly been satisfied with limitations and areas for improvement in the study.

One area of improvement involves the reproducibility of the results and experiments, which casts a limitation of the applicability of these findings across the field of LSTM’s. One suggested improvement found in the primary research phase is the use of multiple experiments constructed with a distribution of metric readings for each model listed, with a mean metric score utilised to indicate the highest performing variant. The setting of a random seed could also improve the reproducibility, the lack of which created challenges in presenting the results of the first implementation of the models, as these results can vary each time the code is run. Careful formulation of the training and test data can also improve the validity of these findings, through ensuring all training data is excluded from the test data in order to prevent overly-optimistic results. This too can call into question comparisons of these experimental results with those found among the broader literature into LSTM’s. Broadening the metrics utilised to include R2, RMSE and MAPE can provide a more complete view of the performance of each model, particularly the comparability afforded by MAPE versus the literature. This approach was employed in a second implementation of the models and used to compare this study’s models with the literature.

Nonetheless, this study contributes to the literature by displaying the relative strengths of simpler LSTM models such as GRU over more complex models when the dataset contains simpler characteristics and structure. What this demonstrates is computationally expensive models are not always necessary in order to perform accurate predictions, when the relative improvement (or lack thereof) in performance is negligent versus the increase in computational costs which will continue to be a prevalent topic from an industrial and environmental point of view as this technology proliferates and becomes commoditised.

Further research in this area could constitute the corrections detailed above in assessing the performance of these LSTM variants on different types of datasets, spanning both multivariate sets, non-financial datasets and more volatile datasets which will add robustness and credibility to claims made regarding the performance of each LSTM variant, with the strengths and weaknesses of each LSTM variant more thoroughly challenged.

# Bibliography

Abbasimehr, H. and Paki, R. (2022) Improving time series forecasting using LSTM and attention models. *Journal of Ambient Intelligence and Humanized Computing*. [Online] 13 (1), 673–691. Available at: doi:10.1007/s12652-020-02761-x (Accessed: 26 April 2024).

Anon (2024) *Crude Oil Prices: Brent - Europe*. [Online]. 7 August 2024. Available at: https://fred.stlouisfed.org/series/DCOILBRENTEU (Accessed: 7 August 2024).

Anon (n.d.) *StandardScaler*. [Online]. scikit-learn. Available at: https://scikit-learn/stable/modules/generated/sklearn.preprocessing.StandardScaler.html (Accessed: 14 September 2024).

Aung, N.N., Pang, J., Chua, M.C.H. and Tan, H.X. (2023) A novel bidirectional LSTM deep learning approach for COVID-19 forecasting. *Scientific Reports*. [Online] 13 (1), 1–11. Available at: doi:10.1038/s41598-023-44924-8 (Accessed: 8 September 2024).

Bai, Y., Xie, J., Liu, C., Tao, Y., et al. (2021) Regression modeling for enterprise electricity consumption: A comparison of recurrent neural network and its variants. *International Journal of Electrical Power & Energy Systems*. [Online] 126106612. Available at: doi:10.1016/j.ijepes.2020.106612 (Accessed: 26 April 2024).

Cen, Z. and Wang, J. (2019) Crude oil price prediction model with long short term memory deep learning based on prior knowledge data transfer. *Energy*. [Online] 169160–171. Available at: doi:10.1016/j.energy.2018.12.016 (Accessed: 21 April 2024).

Chicco, D., Warrens, M.J. and Jurman, G. (2021) The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. *PeerJ Computer Science*. [Online] 7e623. Available at: doi:10.7717/peerj-cs.623 (Accessed: 20 September 2024).

Chien, H.-Y.S., Turek, J.S., Beckage, N., Vo, V.A., et al. (2021) *Slower is Better: Revisiting the Forgetting Mechanism in LSTM for Slower Information Decay* [Online]. Available at: http://arxiv.org/abs/2105.05944 (Accessed: 27 April 2024).

Dash, S., Sahu, P.K. and Mishra, D. (2023) Forex market directional trends forecasting with Bidirectional-LSTM and enhanced DeepSense network using all member-based optimizer. *Intelligent Decision Technologies*. [Online] 17 (4), 1351–1382. Available at: doi:10.3233/IDT-230183 (Accessed: 8 September 2024).

Datta, D., Agarwal, S., Kumar, V., Raj, M., et al. (2019) Design of Current Mode Sigmoid Function and Hyperbolic Tangent Function. In: Anirban Sengupta, Sudeb Dasgupta, Virendra Singh, Rohit Sharma, et al. (eds.). *VLSI Design and Test*. Communications in Computer and Information Science. [Online]. Singapore, Springer Singapore. pp. 47–60. Available at: doi:10.1007/978-981-32-9767-8\_5 (Accessed: 20 September 2024).

Dutta, A., Kumar, S. and Basu, M. (2020) A Gated Recurrent Unit Approach to Bitcoin Price Prediction. *Journal of Risk and Financial Management*. [Online] 13 (2), 23. Available at: doi:10.3390/jrfm13020023 (Accessed: 27 April 2024).

Elsheikh, A., Yacout, S. and Ouali, M.-S. (2019) Bidirectional handshaking LSTM for remaining useful life prediction. *Neurocomputing*. [Online] 323148–156. Available at: doi:10.1016/j.neucom.2018.09.076 (Accessed: 23 April 2024).

Farah, S., David A, W., Humaira, N., Aneela, Z., et al. (2022) Short-term multi-hour ahead country-wide wind power prediction for Germany using gated recurrent unit deep learning. *Renewable and Sustainable Energy Reviews*. [Online] 167112700. Available at: doi:10.1016/j.rser.2022.112700 (Accessed: 26 April 2024).

Fischer, T. and Krauss, C. (2018) Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*. [Online] 270 (2), 654–669. Available at: doi:10.1016/j.ejor.2017.11.054 (Accessed: 14 April 2024).

Gers, F.A., Schmidhuber, J. and Cummins, F. (2000) Learning to Forget: Continual Prediction with LSTM. *Neural Computation*. [Online] 12 (10), 2451–2471. Available at: doi:10.1162/089976600300015015.

Graves, A. and Schmidhuber, J. (2005) Framewise phoneme classification with bidirectional LSTM networks. In: *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.* [Online]. July 2005 pp. 2047–2052 vol. 4. Available at: doi:10.1109/IJCNN.2005.1556215 (Accessed: 23 April 2024).

Hochreiter, S. and Schmidhuber, J. (1997) Long Short-Term Memory. *Neural Computation*. [Online] 9 (8), 1735–1780. Available at: doi:10.1162/neco.1997.9.8.1735 (Accessed: 14 April 2024).

Hoedt, P.-J., Kratzert, F., Klotz, D., Halmich, C., et al. (2021) MC-LSTM: Mass-Conserving LSTM. In: *Proceedings of the 38th International Conference on Machine Learning*. [Online]. 1 July 2021 PMLR. pp. 4275–4286. Available at: https://proceedings.mlr.press/v139/hoedt21a.html (Accessed: 27 April 2024).

Hong, J., Wang, Z. and Yao, Y. (2019) Fault prognosis of battery system based on accurate voltage abnormity prognosis using long short-term memory neural networks. *Applied Energy*. [Online] 251113381. Available at: doi:10.1016/j.apenergy.2019.113381 (Accessed: 21 April 2024).

Huang, J., Yang, S., Li, J., Oh, J., et al. (2023) Prediction model of sparse autoencoder-based bidirectional LSTM for wastewater flow rate. *Journal of Supercomputing*. [Online] 79 (4), 4412–4435. Available at: doi:10.1007/s11227-022-04827-3 (Accessed: 8 September 2024).

Khashei, M. and Hajirahimi, Z. (2017) Performance evaluation of series and parallel strategies for financial time series forecasting. *Financial Innovation*. [Online] 3 (1), 1–24. Available at: doi:10.1186/s40854-017-0074-9 (Accessed: 20 September 2024).

Lakshmi, S. and Maheswaran, C.P. (2024) Effective deep learning based grade prediction system using gated recurrent unit (GRU) with feature optimization using analysis of variance (ANOVA). *Automatika: Journal for Control, Measurement, Electronics, Computing & Communications*. [Online] 65 (2), 425–440. Available at: doi:10.1080/00051144.2023.2296790.

Lei, J., Liu, C. and Jiang, D. (2019) Fault diagnosis of wind turbine based on Long Short-term memory networks. *Renewable Energy*. [Online] 133422–432. Available at: doi:10.1016/j.renene.2018.10.031 (Accessed: 21 April 2024).

Lin, J., Ma, J., Zhu, J. and Cui, Y. (2022) Short-term load forecasting based on LSTM networks considering attention mechanism. *International Journal of Electrical Power & Energy Systems*. [Online] 137107818. Available at: doi:10.1016/j.ijepes.2021.107818 (Accessed: 26 April 2024).

Lin, Z., Cheng, L. and Huang, G. (2020) Electricity consumption prediction based on LSTM with attention mechanism. *IEEJ Transactions on Electrical and Electronic Engineering*. [Online] 15 (4), 556–562. Available at: doi:10.1002/tee.23088 (Accessed: 26 April 2024).

Liu, Y. (2019) Novel volatility forecasting using deep learning–Long Short Term Memory Recurrent Neural Networks. *Expert Systems with Applications*. [Online] 13299–109. Available at: doi:10.1016/j.eswa.2019.04.038 (Accessed: 21 April 2024).

Mahto, S., Vo, V.A., Turek, J.S. and Huth, A.G. (2021) *Multi-timescale Representation Learning in LSTM Language Models* [Online]. Available at: http://arxiv.org/abs/2009.12727 (Accessed: 27 April 2024).

Mardjo, A. and Choksuchat, C. (2024) HyB-LSTM: Multivariate Bitcoin Price Forecasting Using Hybrid Time-Series Models With Bidirectional LSTM. *IEEE Access*. [Online] 1250792–50808. Available at: doi:10.1109/ACCESS.2024.3386029 (Accessed: 8 September 2024).

Ran, X., Shan, Z., Fang, Y. and Lin, C. (2019) An LSTM-Based Method with Attention Mechanism for Travel Time Prediction. *Sensors*. [Online] 19 (4), 861. Available at: doi:10.3390/s19040861 (Accessed: 27 April 2024).

Rodrigues, F., Markou, I. and Pereira, F.C. (2019) Combining time-series and textual data for taxi demand prediction in event areas: A deep learning approach. *Information Fusion*. [Online] 49120–129. Available at: doi:10.1016/j.inffus.2018.07.007 (Accessed: 21 April 2024).

Sagheer, A. and Kotb, M. (2019) Time series forecasting of petroleum production using deep LSTM recurrent networks. *Neurocomputing*. [Online] 323203–213. Available at: doi:10.1016/j.neucom.2018.09.082 (Accessed: 14 April 2024).

Salton, G. and Kelleher, J. (2019) Persistence pays off: Paying Attention to What the LSTM Gating Mechanism Persists. In: Ruslan Mitkov and Galia Angelova (eds.). *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*. [Online]. September 2019 Varna, Bulgaria, INCOMA Ltd. pp. 1052–1059. Available at: doi:10.26615/978-954-452-056-4\_121 (Accessed: 8 September 2024).

Sang, C. and Di Pierro, M. (2019) Improving trading technical analysis with TensorFlow Long Short-Term Memory (LSTM) Neural Network. *The Journal of Finance and Data Science*. [Online] 5 (1), 1–11. Available at: doi:10.1016/j.jfds.2018.10.003 (Accessed: 23 April 2024).

Silva, D.G. da and Meneses, A.A. de M. (2023) Comparing Long Short-Term Memory (LSTM) and bidirectional LSTM deep neural networks for power consumption prediction. *Energy Reports*. [Online] 10 (3315–3334), 3315–3334. Available at: doi:10.1016/j.egyr.2023.09.175 (Accessed: 8 September 2024).

Uddin, Md.Z. (2019) A wearable sensor-based activity prediction system to facilitate edge computing in smart healthcare system. *Journal of Parallel and Distributed Computing*. [Online] 12346–53. Available at: doi:10.1016/j.jpdc.2018.08.010 (Accessed: 22 April 2024).

Van Houdt, G., Mosquera, C. and Nápoles, G. (2020) A review on the long short-term memory model. *Artificial Intelligence Review*. [Online] 53 (8), 5929–5955. Available at: doi:10.1007/s10462-020-09838-1.

Vijayalakshmi, B., Thanga Ramya, S. and Ramar, K. (2023) Multivariate Congestion Prediction using Stacked LSTM Autoencoder based Bidirectional LSTM Model. *KSII Transactions on Internet & Information Systems*. [Online] 17 (1), 216–238. Available at: doi:10.3837/tiis.2023.01.012 (Accessed: 8 September 2024).

Wirth, R. and Hipp, J. (n.d.) *CRISP-DM: Towards a Standard Process Model for Data Mining*.

Yan, H. and Ouyang, H. (2018) Financial Time Series Prediction Based on Deep Learning. *Wireless Personal Communications*. [Online] 102 (2), 683–700. Available at: doi:10.1007/s11277-017-5086-2 (Accessed: 14 April 2024).

Yang, J., Guo, Y. and Zhao, W. (2019) Long short-term memory neural network based fault detection and isolation for electro-mechanical actuators. *Neurocomputing*. [Online] 36085–96. Available at: doi:10.1016/j.neucom.2019.06.029 (Accessed: 21 April 2024).

# Appendix

## 8.1 Code Implementation 2 Results

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Rank** | **LSTM Variant** | **Mean Squared Error (MSE)** | **Root Mean Squared Error (RMSE)** | **R Squared (R2)** | **Mean Absolute Percentage Error (MAPE)** | **Batch Size** | **Epochs** | **Dropout Rate** | **Layers** | **Optimizer** | **Units** | **Forget Bias** |
| 4 | GRU | 0.305 | 0.552 | -0.225 | 0.042 | 16 | 10 | 0.26 | 1 | Rmsprop | 25 | - |
| 1 | Vanilla | 0.101 | 0.317 | 0.596 | 0.022 | 16 | 30 | 0.32 | 1 | Rmsprop | 39 | - |
| 3 | Bidirectional | 0.129 | 0.359 | 0.481 | 0.024 | 16 | 10 | 0.35 | 3 | Adam | 41 | - |
| 2 | Forget-Gate Bias | 0.117 | 0.342 | 0.530 | 0.021 | 16 | 20 | 0.13 | 1 | Rmsprop | 42 | 0.78 |
| 6 | Attention Mechanism | 0.320 | 0.566 | -0.286 | 0.042 | 16 | 20 | 0.32 | 2 | Adam | 41 | - |
| 5 | Forget-Gate Bias & Attention Mechanism | 0.307 | 0.554 | -0.233 | 0.040 | 16 | 30 | 0.30 | 1 | Adam | 38 | 2.12 |

Table 5: Results of Code Implementation 2

## 8.2 Thematic Results of Primary Research



Table 6: Thematic Results of In-Depth Interviews

## 8.3 Interviewee Job Titles

**Interviewee A –** Private Equity Fund Account Analyst

**Interviewee B –** Senior Digital Processes & Tools Specialist

**Interviewee C –** Senior Data Scientist

**Interviewee D –** Senior Software Development Engineer

**Interviewee E –** Senior Research Data Scientist

**Interviewee F –** Data Scientist

**Interviewee G –** Experience as Data Scientist

**Interviewee H –** Senior Data Scientist

**Interviewee I –** Senior Data Scientist

## 8.4 Interview A Transcript

**Interviewer:** OK great OK Anonymous so yeah thanks for agreeing to do this and so how I'm thinking of running it is just I'll take you through just some of the main elements of the code I shared and after that I'll basically just be looking for your thoughts from there and like I have a few kind of prompt questions it it's quite open-ended so like open to whatever thoughts you have um and um like I have notes myself to kind of help the conversation but um like you know your your thoughts can be as general as you like you know so don't feel under pressure there so anyway I'll just kick off then by or is there any questions you have before we start are you happy with that with that

**Subject:** if anything pops up as we go I'll be happy to talk to you then

**Interviewer:** great thanks again so I'll just share the screen you can kick off so just let me know when you can see the screen coming up you

**Subject:** can see that perfect

**Interviewer:** yeah so I'm just there on the first of the kind of LSTM variants that I've been working on so um so basically get the background sorry just before we go into it is that basically I've taken 6 variants of LSTM's and I'm just measuring the mean squared error what they achieve in each of their models and then like comparing each of them like which one does the best why does it do the best what are the strengths and weaknesses of each approach and that that kind of thing so anyway I'll just go to the kind of literally the model building part for each one because I'm just conscious of our time as well so this is the vanilla the STM so the kind of the kind of basic what the most standard 1 and so this is where I built it so I've defined the function and for the recurrent neural network to build it and I've kind of looped through that for adding layers I've created a loop there and then I've hard code at the mean squared error as the loss function because that's what the then I wanted to use cross validation to try and find the best hyperparameters so that's where I've built the to use a keras model as a scikit learn model for cross validation uh I've put in and these are the kind of different parameters I want the cross validation to explore to see what works best and then literally so here's the randomized search CV the cross validation kind of UM technique I'm using and then just down here I'm fitting the model and and then the actual test results and are down here so this is a visual of how the predictions did against the actual real data itself and then just down here is the actual mean squared error right achieved now because I didn't set a random seed the the the score that you're seeing here in front of you is different to what I have recorded before because each time I run it it gives a different result so at the end of this I'll just show you this table here of the kind of original results I got just just to kind of help with the kind of comparison between the models but anyway that's the that's the first one so that's the vanilla LSTM so the next one I did is it gru which is sorry yeah sure and so the Gru then is like a more simplified version combining some of the gates and um again I've built it here so it's it's similar I've defined a recurrent neural network uh function um and I've I've added in so here's where I've added in out of the crew and layer instead of what I would have thought before if auded in gru there and to kind of combine the the gates and I've added another one in in the layers coming after that so that's where crew comes in basically everything else is roughly the same the same loss function is used and then the same cross validation process I'll go down to the results so again this is the red is the predictions and the real data is in black so you can just see how closely that aligns there and then again I got the mean squared error kind of score for that one too and so I'll just I'll go through all the scores at the end and I'll just keep going through the models for now just to give us a chance to get feedback so the next one was a long short term memory network with an attention mechanism so where it kind of focuses on certain inputs that it thinks are important and this is the model so it's slightly different approach this time umm but anyway the defined the default parameters uh made the first LSTM layer and drop out but this time we've kind of done in a slightly different way creating variables called X um I've added additional layers and dropouts and then this is the thing that's very different I;ve added in an in attention mechanism to capture important parts of the input sequence um and then I've um yeah and then I've just kind of finished off the model they're creating the model with specified input and output layers and I've hard coded the mean squared error so again I'll go through the results for this one so the results are roughly uh so again red is predictions and black is real data and then the mean squared error is there and I'll go through them again at the end just make sure that's still working sorry it's just the dictation stops so and so the next one is with the forget gate bias so this is where the there's kind of the forget gate has kind of been initialized value and to kind of help it and not forget things that are important basically so this is how it's being built so there's a few differences in here and there's a forget gate biases start is set at the beginning but then also with the cross validation it's going to experiment with different values here to see which one works best and then I've kind of built the the model again kind of similar to the ones previously but I've added in the forget gate bias in each layer apart from the last one and then also here so here's where I've put in for the model to say OK explore the forget gate bias value try these different values and see which ones work best the results are down at the bottom so that's that's the result so prediction is in the red real data in the black and in the mean squared error is there below so that's the one I got in this particular running of the code so then this one is a combination of the last two together and so attention mechanism and forget bias so the hope was to kind of combine the benefits of the two previous ones to see what happens which is an interesting 1 so again I've put in so similar to before I put in the LSTM layer with the forget gate bias and and then yeah another one with another layer would forget gate bias in the loop and an attention mechanism and to help focus on important parts of the input um and then the rest of it isn't too different to what's been done previously and then sorry to results are here so that's not too great or like not quite as good the the predictions were very flat like consistent uh rigid and there were black is the real data and the result there was higher than the others can see there and then finally this is the final one be happy to hear and so this is the bidirectional LSTM where the processing kind of happens both forwards and backwards and to try and kind of get a better context on the the sequence of the time series data and so here again I've started the default parameters of the build and then I've added in just there's a bidirectional kind of class that you can use from a library for the layers and I've added that in and then also in this loop for the next uh numbers of layers it's in there as well and then yeah and again there and then the final layer doesn't have it because it's the the output layer um and then again just experimenting with the different parameters there they were kind of they were the same parameters for all of them all of the models just so there be some consistency and then let's get down to results so there's the results so the red is predictions the real data is in black and the mean squared error is there they're lower than 4 better than the last model so then just a summary of those so summary of the results the the kind of the first time I ran it so they might look a bit different to what I showed there but basically this is kind of the ranking of which one performed best to work the worst so the gru went actually and my first time doing it performed the best with the lowest mean squared error score and then I just have kind of a table there of like what were the best hyper parameters for each one uh I just thought I'd show you this just to kind of summarize uh the results and which ones did best which ones did worse so that that one where I combined forget gate bias and attention mechanism where the water was the worst so just quite workout and then so that's the the order so grew was best vanilla LSTM was second the bidirectional one that I did at the end there was third forget gate bias was then a bit of a gap to the forget gate bias that one was fourth attention mechanism was similar to forget gate bias in 5th and then the forget gate bias and the two of them combined was way worse so that would just didn't quite work out and that's that's it really I know that's very fast and I've gone through a lot there but um if uh just want to ask maybe your initial thoughts or if you have any questions or thoughts on any of those if you want to go back to the first one to just have a look at that or So what are your thoughts initially on on what's been done there and the approach is there anything that that's occurring to you as you're looking at those?

**Subject:** nothing is like immediately like screaming at me when it comes to like the methodology and I've actually covered it yeah I wouldn't be familiar enough when it comes to some of the intricacies of the actual Python code itself to have like online like critique yeah good seeing some of these outputs on your overall results but I wonder if you have any more specific questions like I'm struggling to think about like yeah like the stacking is quite good and especially when it's locked you can get through like a lot of complex down very fast but yeah I'll definitely go through some more particular questions cause yeah my overall though especially if you have the code or taking a long time actually appears through it yeah yeah familiar with yeah well like the programs itself

**Interviewer:** yeah yeah no worries at all like to be honest like I've I've written loads of notes myself to remind myself what's going on like so no worries there and so I suppose maybe I'll just ask around the one thing I'll ask around is basically how I tried to optimize the hyper parameters so the kind of approach I took was to basically use randomized search CV and then I chose basically these are the different elements I wanted to experiment with like so for example in you know an LSTM layer how many how many nodes are going to be in each layer and how many layers are there going to be you know and then it would experiment with one layer 2 layers 3 layers 4 layers all of that the number of different epochs that it would go through um you know how many times it would basically uh teach itself and the the dropout rate you know that's in there to you know experiment with to make sure that the model doesn't overfit and things like that and so it's being regularised or um and then batch size as well just what it's consuming how much it's consuming at a time and then optimizers itself the two that we're going with here were Adam and RMS prop so the so the kind of strategy I used for hyperparameter tuning is I tried to automate it so that so that the system instead of Me continually running the models and clicking a different number each time for how many layers in the units I had this run it for me because so it would cover more than I can cover basically and I don't know if you have any thoughts on those is there anything to do with that that you then you would like to explore further or do you think some of these numbers like would you like have liked me to go further with some of these should I have put more higher range of values to explore or is there something that I'm missing or anything like that

**Subject:** think your batch size is is looking fine I mean that's quite that's quite good yeah so with the model optimizers if I'm not 100% sure like which is the what are you using to control your computer on the dropout rate in this

**Interviewer:** to the dropout rate is here that's right guys so yeah it's just exploring from 0.1 to 0.5 which dropped out rate kind of works best I suppose and then the the model optimizer like Adam I have that in there because it's kind of like a standard one that's used and then rms prop is just there as an alternative and kind of less um widely used it's still one that's kind of good for can be good for regression problems so it's just there as an alternative so yeah is there anything else that's or do you want me to go onto one of the different uh kind of model that I went through later on and see if there's anything there that inside or

**Subject:** she's just like process that we don't have to like structure of coding it's almost like a slightly more advanced that I haven't worked on myself like I actually should have actually constructed out of the tools we use are parking in the house yeah yeah that's serious like code wise like I know that you basic concepts for applications but like actually optimizes without like having almost dealing more with theory than the actual like other implications and like like like how to approach improvising like that

**Interviewer:** yeah you know and that's perfectly fine like we can because I'll tell you I'll be honest with you most of the course that I'm doing it's actually more about they actually want to hear more about the theory than the the the the implementation itself so I know you've gone through the implementation here but if there's anything theory wise that you'd like to go through as well like that's that's perfect that's great like that's perfectly what we're looking for here as well so if there's anything I don't know on the surface that you'd like to just ask about or find out further about happy to to go through that

**Subject:** what made you go with that method of optimization because I do know a lot of people it's always like Bayesian optimization what made you choose your method and model like your hyperparameter training

**Interviewer:** yeah good question so um so like in terms of hyperparameter tuning is it they're kind of yeah yeah So what I did there is so I think that's fine and the recording is still there so I think what I did was so that the kind of choices I had was either um basically manually changed the figures myself and keep running it the the training and all that each time what I found that that would be very time intensive basically just for the size of the the model and and also yeah it was just taking a long time to train like I actually reduced the data significantly from what it was at the start just to try and help with that as well and then yeah so and it was it was mentioned be honest it was mentioned in my class as well that option like that automating hyperparameter tuning is something we could do and I thought ohh that sounds like a good idea like the computer itself the program will will go through at like lightning speed loads of different options and then just tell me which one is the best performing at the end so then from there I think that the two choices that I had were well there was more than two but the two that I was looking at was randomized search CV cross validation method and then all the other one is uh what's the main search CV and grid search CV so the reason I went with randomized search CV was because grid search cv if he might have taken too long as well because it kind of goes through every single option and that you can think of where that that's possible whereas randomized search CV just takes random values across the board and tests them so it's it mightn’t be as thorough but in terms of just time resources it's a bit more efficient so that's it you know it doesn't take as long basically and I thought it might still cover the bases so that was really the thinking behind the uh the the tuner of the

**Subject:** I know I know a lot of people use like cyclical learning rates so you can just keep testing different variants of the actual model so they can also look down on the individual like manual changing of the code time after time yeah yeah yeah yeah now that's something that it is very intense like time wise but once you actually get your overall result results output gosh you know when you have your book

**Interviewer:** yeah yeah that's very true yeah that is an approach that I could explore because um um I suppose what I what I've been focusing on is like the end result after all of the others have been tested maybe I haven't been looking at the inner workings of OK if I change the batch size from 16 to 32 or whatever what's the difference between those two settings like in the results so I probably won't get the approach you're saying would probably be better at getting under the surface of exactly what is changing and what instead of just looking at the the end result overall end result that's definitely something that I could I think I could think could help

**Subject:** that's very fair I would say always with code though it's always going to be temperamental and it doesn't always make the most sense a lot of times looking at outputs isn’t probably the best way to do it they show the result should be seeing

**Interviewer:** Yeah yeah yeah that's fair yeah I could be to maybe I was too outcome focused and wasn't looking and through the yeah the the kind of step by step methodology as much that's that's fair

**Subject:** and we still think the general though like it is structured so well and especially the way you structure it in the python like it was actually able to go like line by line and actually see each individual piece like I think it is and that is like very malleable and like you can actually change it depending on let's say if you're dealing with like a very different like commodities data set like in general though I don’t think it is it is really really well done and like I don't think you would see this out of place somewhere like where I was working even something like almost like the opposite that I've touched myself like I'm only in the job now on my seven months I wouldn't be on my training or building something like this myself like I'm still using in the house datasets so like today I don't think you should be in any way like concerned worried about the complexity or like actual results you're producing this

**Interviewer:** yeah yeah ohh no that's fair yeah like I'm I'm pretty like I am fairly compared to what I had when I was first started trying to do it and then what I ended up with like yeah I'm definitely happy with the progress there but absolutely like open to exploring what what could be done differently and and uh I suppose analyzing the Inns and outs of each model and what could have been done differently so yeah they take completely take on your feedback and and might be something I need to look into more actually just to achieve some of my research objectives which are to like look one of them is to look at the strengths and weaknesses of each model so maybe taking the approach you mentioned you know might better serve that that objective

**Subject:** is there anything else you'd like to move on to anything like to cover and see if I can give you

**Interviewer:** yeah sure so um yeah we'll move on to one more thing so variable best performing OK we'll move on to so the one that screen is freezing a bit there so the one that performed best for me was the the gru which was the second one I did which is kind of actually one of the more simpler setups so I don't know if you want to so I might need to stop sharing just my screen it's freezing there for some reason one second the state OK ohh there let's see if it moves it does so the yeah the one that actually achieved the best score was the second one which was this one that gru so this this was kind of like the first one which is the most kind of conventional standard going except except basically again it's supported the ones I chose were all supported by libraries so I wouldn't have to like make custom ones myself I just felt that was a little too advanced for where I'm at the moment and the time I had so I used ones that all had like library support already so I just added in in each layer the the gru kind of class and so it turned into a through layer and it combines a couple of the gates together so it resulted in a more simpler structure I suppose and it was actually this one that created the best result so I'm wondering then I'll just just without any just like from a distance even when you hear that some of the more simpler structures perform better and do you have any thoughts on on that or like what could the more complex ones might have performed as well like what would there be anything you would personally look into to see why uh the complex ones weren't as good as some of the more simpler approaches

**Subject:** it definitely can't be its ability to like like it's like drops out or removes data it finds extraneous so if you could be in even some of the packages you're using to analyze the data like who would be that you're getting let's say you have your don't know how many points you're using in the let's say you're commodities down there for oil but let's say it's actually is your point that it will take what it feels is the best like 20,000 points out like 35,000 points and trying to almost do such a good job to like have like getting into investigating data or you'll actually go overall damage output like now some simple models it will take almost everything doesn't really you know the regularization doesn't really like to prevent as much of the overfitting but like actual output will be more accurate to just using more of the data yeah now for gru like I've ever looked I have like some notes here but I'm not exactly in relation to that I do know that I had another one I have here Bayesian that there's like one or two libraries you can use I think altuna was when we use at work OK so like that can make a lot easier yeah that's what they have here search for so I wasn't I wasn't realized the scale of the work we do most of my notes were just slightly theory based or yeah no no worries though

**Interviewer:** yeah yeah no that's fine

**Subject:** that would be my initial thought sorry if they're almost a little bit scattershot but would you have any other questions

**Interviewer:** yeah yeah no no worries at all actually the the Bayesian approach you're talking about like that's interesting like that's kind of new that's something that I haven't done or thought about so like I'd be interested to hear about how that normally how you normally you know how that normally works and then you know from your experience as well with financial data as well it'd be interesting to hear or perhaps just because that's kind of something that I haven't done before it's I've heard I think I've heard of it but just haven't done anything on it so I'd be interested to hear more about that approach

**Subject:** ohh like I do know he used his like sort of models and it just has like a the way it explores data overall and just like reduce the amount of parameters yeah loved ones as well but like when it comes to actual products actually don't I don't quite understand the actual amount actual like the way it actually does its work I'm only used to just like if I have this issue this is a tool I will use

**Interviewer:** yeah yeah no that's perfectly fine

**Subject:** just like so it's like it's just like before she's so far but my my current my current ability

**Interviewer:** yeah no no no worries at all and so um I hear what you're saying it sounds like there's some parameter sharing going on with the Bayesian approach so it might be more basically more efficient take less time and less resource intensive

**Subject:** just use it a lot less evaluation when you start running your code with this you're asking to be faster and like especially if you're more complex models like especially works at the end like you'll find they'll actually like fly through faster it's less like expensive computing was like just the sheer time running between each model

**Interviewer:** yeah yeah yeah and that's a very valid point like some of the some of the ones I used they did take a while they were quite quite resource intensive like some of the notes I have here about some of the variants I used is that they have high complexity training difficulty resource intensive so I mean taking a Bayesian approach you know could with alleviate some of those issues so and that's absolutely something we could look into and you know just as a recommended uh kind of future consideration if if someone wants to do what I did again you know that they could look at the Bayesian optimization approach

**Subject:** yeah it's very good for yeah right now it's like I don't want to say too much but I just like seeing on like a almost like some shots in the dark I don't wanna be saying too much that actually being 100% out of myself yeah

**Interviewer:** no that that's OK don't worry and like uh like I've gone pretty deep in here on one specific area but like so I know that like I wouldn't necessarily expect that everyone else has also done the same thing and um like these are all you know qualitative interviews that we take in the defect the data and you know summarizing it and that's the don't worry about any claims or anything like that and that would be made about that what works sorry say that again

**Subject:** for myself as long as I can give you you know one or two pieces of information you can use or even explore further with that

**Interviewer:** yeah absolutely like and to be honest I think you've done that and I'm just looking at my research objectives here and I think you've you've given me some good stuff on the hyperparameter optimization and then again and then also just on the kind of why the the best performing variance where the best so I think that's that's like some of the key research objectives that I had and and actually that's 3 of the four research objectives and then you kind of touched on this just just when you're talking about that that piece about the more simpler structures also introduced that so I think that's I think that's everything I have and the outcome is to just be 1/2 an hour or so really appreciate those insights you gave and then yeah and just thanks again for that

**Subject:** you're very very welcome message

## 8.5 Interview B Transcript

**Researcher:** As well although that doesn't always I think it goes off but never mind and OK so I'll just share the screen so or just as a background then again so the project what that I did is I took six different LSTM long short term memory variants so variant would be good and I did some prediction on financial time series data and then just compares the mean squared error of each variant and then I suppose that the idea is to identify the best one and like look at why the best one performed the best what's the strengths or weaknesses of each one and and also just to get the best hyperparameters for each one I suppose as well. So through some of the main stuff around the models and then look for your comments then after and I have a few questions you know as well just to help the discussion. And so you can let me know when you can see the screen.

**Subject:** yeah I can see the screen

**Researcher:** MMM MMM OK great so I won't go through all the data preprocessing although if you want to go through that later we can but I mean it's just a lot and it's you know we we don't have too much time so we'll just focus on the models for now so the first one I did first variant I did is just the the kind of standard vanilla LSTM so this is where I I built it here so important all the libraries and I defined a function here to build it and then data loop to to kind of create different layers and you can see the LSTM class there that's coming in for the LSTM layer that's how I brought the LSTM kind of a structure onto the the network and then the reason I did it this way is because I wanted to not just do it once and then have to change the hyperparameters and do it again and again I wanted to be able to automate it so that I could use uh cross validation to find the best hyperparameters so I've kind of built it in a way that umm you know it gets automated umm how it searches through for the best type of parameters so umm anyway yeah so there's the loop for the different layers then this one here is the final kind of output and then just for the mean squared error like hard coded the mean square error as the last function just because that's what my project is all about is measuring that and then this here is what what I use to kind of I suppose wrap the model in a way that could be used for cross validation and then these are the hyperparameters that I wanted to kind of explore so which ones work the best so like the number of nodes in each layer the number of layers themselves the dropout rate to avoid overfitting the batch size the number of epochs the number of times the model you know goes through the training all that stuff and then I'll go down to sort of the training itself fitting the model happens here and then the actual results then I'll just skip down are here so the red line is the predictions on the black line is the real data so it's it's relatively close like the the data set is about 2600 let's say length of what I was trying to do was predict that the next 20 data points after taking in you know that many something like that so this how it looked and then the mean squared error was calculated there and now I have A have a table of results so that the table of results I have will look a bit different to this 'cause I didn't set a random seed so each time this code is round it will get a different mean squared error but I have like a table of the kind of results that I got initially up you can discuss at the end so anyway that's the the first one of the six is that all OK enough so far at least at the theory level?

**Subject:** so just on the layers he said so he said that you were doing a loop right

**Interviewer:** Yes

S**ubject**: or you said you’re only 1 layer yes so the number of layers is equal to 2?

**Interviewer:** and so I set up I set up the first layer here this is the the first layer and then after that I set up like a loop to kind of basically yeah go through different numbers of layers so so for example sometimes there will be just one layer sometimes there will be two layers sometimes there will be 3 sometimes there will be 4 so it's kind of flexible and that it it adapts and in the cross validation it adapts so it it trials it trials each number of layers and sees which one works best

**Subject:** and how do you decide that so do you just use the MSE to decide on that or?

**Interviewer**: and so I I made the decision at the beginning so I wanted to just I wanted to find a good way of tuning the hyper parameters so I figured that just due to time constraints how long it takes to train the data but that that that like it mightn’t be practical for me to just manually change those hyperparameters each time and then train the model all over again so I I wanted to automate it and that they could all be kind of trialled at the same time and then and then at the end it just kind of gives me which combination of hyperparameter values performed the best at the end so that's kind of the reason why why I went with that way so it does trial with just one layer trials with two layers and and so on along with along with these other micro parameters here also get get tested at different uh values to see which one combination of values perform the best

**Subject**: I know that's fair enough because I yeah you already know that like for LSTM like you're fine like even if you have just like 2 layers it's kind of like standard So what happens when you have more layers is like it gets more complex and you would have to tune the hyperparameters but also at the same time he concerned about not over fitting the model just so that you have more accuracy you shouldn’t be over fitting the model but yeah if you just have or find the right balance yeah you’re fine.

**Interviewer**: no that's a good point and that's actually interesting we'll come back to that later because you'll be you'll be interested in the some of the results I think anyway that's the first one I'll just quickly run through the other ones not spent too long on them so the next one was a gru so the more kind of simplified version where couple of the gates are kind of mixed together into one and and anyway this is just the code where I built it it's it's not too different to before except I've I've added in the gru class to make a a groove instead of uh the vanilla LSTM uh I I you know so I I put in the initial uh layer they have the umm their loop then to like loop through experimenting different numbers of layers and I have the gru layer again there and then I have the output layer here and mean squared error again is the last function because that's what I'm using to measure it and I have the again the hyperparameters to to tune and I used kind of the same numbers for each model just to have some level of consistency and and then again all this stuff is the same as the model before the results are here so again predictions are in red and the real data is in black so they're you know relatively close not perfect but not many miles away uh not too far but anyway that means quite error there is 0.19 and so I think that that's higher than the first one but umm anyway I'll show you the full table of results at the end this time the third one I did with an attention mechanism so it's where it focuses on the kind of the important parts of the inputs and so here's the model so this one bit a bit different in how it's built so it created like a X variable for first LSTM layer and then the dropout rate and created a loop anyway and down here is where I added in the attention mechanism so this is again to capture the important parts of the input sequence and then the final layer

**Subject**: Can I ask a question, is it like giving weightage to epochs?

I**nterviewer**: yeah yeah and it's kind of yeah I suppose it is like it's it's trying to focus on it's putting a focus on inputs and and that kind of help it I suppose when when there's a longer sequence of time series data been taken in it helps it kind of focus on the what's and so it's it's a bit different from the next one I was going to show you where there is like a value you set and kind of like a weight that it puts on everything but this one I suppose it's a similar idea in that it does focus on umm it is looking out for specific uh inputs so uh there could be a waiting from that perspective just but from what I know there's not it's not like a hyperparameter that you tune a value to that was good question so anyway yeah the final LSTM layer this is just the code to build the the model I'll go down to the results sorry and I did use normalization of the data by the way just as a as an aside and so there's the results so again the red line is predictions the black line is the data the real data and then this result is there that's a little true again so this one is forget gate bias so this one is where so basically I've created a a bias in the default parameters I've set that bias to one initially but I also so the idea with this one is is that it like decides what things to not forget basically so it's again it's it's good for dealing with and I suppose initialization of errors in performance so it's kind of increases the performance by like not forgetting important information basically and it is kind of like a weight as you were saying before that you can set and then I set it for 1.0 initially but I have because I did cross validation though in the cross validation I've I've I forget bias value that can also go between anywhere from 0.5 to 2.0 so so the the cross validation kind of checked between you know searched all of those different or a few of those different points and found which value works best in terms of the weight in terms of the forget gate bias and so that's how that one worked and output is or the the results are here so again predictions in red real data is in black and and then the result is there and I'll go through again the results at the end this is the second last one be happy to hear so this is where I combined the last two together so there's an attention mechanism and there's a forget gate place so here's the where the model is built so again it's a similar to when the attention mechanism 1 was built earlier at this time i've included with forgate biases initialized as a constant value

Continuation

Again predictions in red real data is in black and and then the result is there and I'll go through again the results at the end uh this is the second last one be happy to hear so this is where I combined the last two together so there's an attention mechanism and there's a forget gate place so here's the where the model is built so again it's similar to when the attention mechanism 1 was built earlier except this time I've included for decade biases initialized as a constant value and I have the loop where X variables X are set as LSTM layers with a bias and also the dropout rate is there as well here's where I've included the attention mechanism and then here's the final layer with bias and then finally the output layer at the bottom there and so it's kind of a slightly different way of I suppose deploying the code was done for for the ones that involve the forgetting or for ones that involve the attention mechanism and then anyway and then there's the again the hyperparameters that that we're going to be searched through using randomized search cv and the results are here alright so this one didn't go as well so the predictions look very flat and consistent kind of too heavily biased and stuff like say and so maybe it was a bit too steady and then the real data is the real data there that's kind of the usual volatile so that one was the worst performing one and you'll see this is the final one so the bidirectional LSTM so this time The IT it looks both backwards and forwards in its processing of the data to try and get capture more context of the input data that's coming in so again and this time so the bidirectional is actually a class from tensorflow that I've umm imported and I just include that umm in the layer which kind of in the LSTM layer just included within this is wrapped into the LSTM layer bidirectional LSTM layer and I have the range and I'm sorry the loop again to create more layers and the final output layer there and then the just the final kind of compile like optimizer and the mean squared error as the last function and they're the hyperparameters that I'm exploring with the cross validation to see which values work best and then the results are here so predictions are in red and it's real data is in black and then there's the result of that one and so because I didn't set a random seed and each time I run this code I'll get different results so for the purposes just of this and for the project I just kind of recorded the results and as they came can you see this here?

**Subject:** I can only see up to me square

**Interviewer**: oh sorry yeah I hope that didn't happen in the last call actually your man couldn't see my table one second I'm just going to re share window yeah that's better and you can see it OK now OK I hope the last guy could see this it's alright so these were kind of the results I got kind of the first time I did like a full kind of run through of the model so this is in order of best performing in terms of mean squared error so the group was the best performing at 0.100 followed by vanilla LSTM or 0.104 then bidirectional the last one that I did forget gate bias was next there's a bit of a gap to the forget gate bias one and then the attention mechanism is quite similar to the forget gate bias one in terms of results and then the the worst performing one by distance was the one that was combination of the forget gate bias and attention mechanism so and then I just have these are like kind of the the hyperparameters that work best for each model so umm I suppose before I like give you any of my kind of take on that I was wondering maybe what your thoughts are umm overall if you want me to show code again like on or and like I have a few questions you know if this doesn't stand in output just initially what your thoughts just on that approach and and like it can be anything from a theory point of view or just general on the approach there

**Subject:** well you I think like you covered the basis of or at least the core concept of LSTM anyways like with all the different variations but it's like more importantly it's not just for instance but it's mostly for any RNN like that would be the core kind of let's say uh practices that you check now and like as I mentioned like I kind of have the feeling that yes the results are going to be too biased when you introduce both the forget gate bias and the additional the the the weightage bias and which was as expected so I was just in one of those two happened like it's either going to be completely biased or it's going to be overfitted and like it was very biased as he saw from the straight line and again with the number of layers like you increase it's possible that you overfit as well but again like with the forward and backward pass which is basically the bidirection that's and your model is not just learning one way it's it's also learning from the backward pass as well which means that it's capturing more content into what you're trying to predict so even though it may not be the best model as in like like like no model would give you 100% and if it does it's not the right model and you're probably just over fitting it and but like with this like you're capturing all the content that you want to and that would be your kind of ultimate portal which I very well see that you capture there well here so yeah I yeah I'd be interested to see like what's your take on it and like how would you go about selecting what kind of model and that that's in your outcome like you would go with but yeah yeah I think it would capture the essence of the overall the theory

**Interviewer:** yeah no that's great Anonymous thanks yeah I think the the insight you gave there earlier about the that always maybe making it too complicated I think that's the that's the one probably biggest take out that I got from it too just when you look at those results it's the simpler models that performs better and the ones that got too complex are the ones that performed worse so it was interesting you spotted spot that early on and that that's how the results turned out in the end as well so that's probably my main take away from from from my own experimentation of how it was and and so yeah completely agree there so in terms of I suppose how I how I went about looking for the best hyperparameters for each model so the the approach that I used was to use basically to try and automate it through randomized search CV so I was wondering if you have any thoughts on that approach and if if there was anything else you think I should try or do you kind of agree that that's a good approach or like are there other approaches that you would try yourself

**Subject:** I I remember like I I tried used uh gradient descent uh don't know if that's a scope in this project but again that's just to purely just uh look about for the uh hyperparameter tuning uh as the best hyperparameter tuning by looking at the gradient descent yes this could be a way of doing it like yeah yeah the optimizer yeah yeah yeah I I don't have the exposure to using this but yeah the theory could be the kind of the same as in the core principle would be very similar so here it's not like it's just like one answer for also like special machine learning app you would have different options to to go about stuff and do it just as you saw here with the different models so yeah there are also different ways to calculate the error mechanisms and like validation methodologies so yeah yeah no that's an issue

**Interviewer:** OK yeah cool that's great yeah that's interesting cause yeah and the optimizer table I have here yeah the two the two optimizers that I looked at were Adam and RMSprop but umm yeah one that I didn't include was stochastic gradient descent which could have yeah could have been another that umm that might could have performed just as well if not better so maybe that's a you know a limitation in my approach that I could consider for future approaches is to with more than just two different optimizers supporting the stochastic gradient descent there's another potential option as well there OK and then so I think in terms of strength and weaknesses you've kind of covered that I think when you talked about the overcomplication of some of the other models I don't know is there anything more you want to say on that or are you happy that you kind of covered that already

**Subject:** oh I'm happy I think you you captured it fine like with all the different uh metal methods that you chose to do this so you kind of captured it all and explode the right path so Oh yeah I'm happy

**Interviewer:** cool and then the last kind of area or one or two areas I wanted to book out was umm basically the one that performed the best which is the the gru so just before I like say anything from what I took on it and I suppose is there anything I think you've touched on already but is there anything more you'd like to comment on or ask or explore further on to like say why was the crew the best one what what kind of factors may have resulted in this one being the best one best performing one on this particular problem or anything and elaborate like if you were to theorise yourself why that one performed the best what would your initial thoughts be

**Subject:** well as you said like for me it's always a simple model with the perfect hyperparameters will always do the job rather than introducing over complexity to it so I think yeah that's the answer itself

**Interviewr:** yeah yeah no I couldn't agree more I think you're you're what you're saying is right and I think it's backed up by the fact that the three top performing variants there are kind of at least the top two there are probably two of the most simple setups out of this variants and the ones that are at the bottom are the more complex ones to set up so I think that kind of speaks for itself and definitely backs up what you're saying in terms of the simpler models doing the job you know when there's no need to overcomplicate it and there's just one further bit of context that might help or support that as well as that the initial data that I had I actually reduced the amount of data I had initially the data spread across let's let's say 30 years for example like 30 something years I actually I actually took a look at the data and saw how volatile it was after the year 2000 So what I did was I reduced the data to just be between 1988 and 1998 so that their data was a lot more stable and you know basically easier to predict and I suppose a theory that I have is that maybe that cause the data was kind of more stable and not too complex that is simpler model was all that's needed to predict on that more stable data that's that's kind of the theory I have anyway and so I think that might support again what you're saying like like if it was a more volatile data then maybe the more complex model might have been needed but in this case probably not

**Subject:** you're absolutely spot on here yeah because you couldn't account for being able such as like you know like especially like if it comes to prices you couldn't take into account the recession or bias so if you want to introduce those kind of variations that's where you introduced bias otherwise wouldn't actually introduce that so that's when introduced bias to the year because that's like knowing that you know that that year we have a waiting so that's when you use weightage but if it is a model which is performing like as in if the data is seen and everything that just let the model learn for itself that's machine learning and you don't want to overomplicate it but again you you did exactly the right thing I if you would have had like again like as you said like late for the years with like high volatile spikes and everything on those would be considered outliers so then you would have to do outlier handling which is basically what you can use just like OK let's just skip and keep the theme park but if you would have those kind of data yes then yeah you would need to increase bias well

**Interviewer:** yeah yeah well that's actually very good in so I thought I didn't necessarily like it's kind of crystallized it for me that like the data itself has it has a particular level of as a model with particular complexity that suits it to it and then so my two complex some are too simple and then there's some there's some that are in the middle that are just the right amount of complexity to to kind of fit that shape that data and not introduce the noise or the bias as they're saying or necessarily so I think that's will take away that I'll take from that as well and yeah so to be honest that's that's all I wanted to cover at umm so I don't know if you have any final thoughts on that or if you're happy to to just leave it at that but I think that's covered by kind of research objectives we're mostly covered there I think in what you're what you've gone through already so umm yeah just gonna leave it to you in case you have any final thoughts on it

**Subject:** I don't know I'm happy to leave it there yeah great job by the way so yeah good luck to you on this so yeah yeah

**Interviewer:** no thanks to me and yeah yes like I know it looks like I know what I'm doing more than I do but I'm like like you know yourself when you're working on a model you you kind of do research on how to do XY and Z and a lot of it was for like there are things that I haven't done before that like you know have worked so that's great but I'm still I'm still kind of learning myself what it is exactly that it means and stuff but umm it umm yeah I'm I'm happy enough so far with the approach and then thanks for your thanks for your insights I think you've given some good good insights add into my kind of research objectives there I can and that will definitely validate and support them and I think your your talk on the the right model for the right level of complex data is actually a really good one that's crystallized for me so thanks for that

## 8.6 Interview C Transcript

**Interviewer:** OK perfect so I'll share the screen anyway Anonymous I'll just take you through some of the code not not everything just the main parts and then I'll show you like the table of results I have at the end and then you can get your thoughts on it all so we'll share the screen see if I can share the window so you can let me know just when that appears for you

**Subject:** well it's I guess it's great thanks man

**Interviewer:** so out of the six variants the first one I tried was just the LST or the vanilla LSTM so like just the the original kind or the most straightforward 1 so they imported all the libraries here and then woke up

**Subject:** so sorry this one thing could you just start with the data set which you have used so that I can get a picture of how you have used the model so I just want to know how the data looks like and how what is the changes that you have done to the data to get to test your audience

**Interviewer:** yeah yeah that's a fair question so so the data set and originally started as kind of so basically I took two columns dates and then the the price of oil on those dates and so it was normally Monday to Friday kind of trading days and over the 10 year. Let's say um so initially anyway I'll just go so initially I was looking just for things like missing values and things like that and uh throughout the year throughout the throughout each year and throughout each month and throughout each week to see was there missing values or what they were and So what I found in the end so it looked like a some of the missing days that were there where it could be explained by things like public holidays and and and that kind of thing so that was OK but what I did find in the data as you can see here is that there is like uh some like non numeric values they're kind of like you know like characters and things like that entered so I had to do something for them as well and so I did it to do I decided to do forward fill and backfill basically for for those um scenarios just to find a way if I didn't want to use yeah didn't want to use the mean so oh there it is so forward fill and backfill is what I used to try and just fill in those values just so there would be some data in there and that it would hopefully not upset the the signal too much of the data and then I was just doing some visualization of the distribution so actually at this point in time I'm actually not dealing with 10 years here I'm dealing with like 35 years and so that's and it was very skewed to the left So what I and again skewed to the left there So what I did is after visualizing the prices here over the 35 years or whatever it is I I know is that the volatility of the data increased a lot after the year 2000 and So what I did is I just took the data for the 1st 10 years 1988 to 1998 and just used that just because there's a bit more stable for hopefully better better predictions and results and then when I did that the data became more gaussian when I just took those first ten years and that's what it looks like there yeah and so then I I did the train tests I I split the the data then into training and split there was kind of a lot of preprocessing there I don't know if you want to go through all of that

**Subject:** yeah that that's fine that's totally fine I just wanted to know what the features how it looked like

**Interviewer:** yeah yeah no that's that's a good question so thanks for for asking that and I did normalize the data as well

**Subject:** yeah yeah that's that's fine yeah OK so that's the OK

**Interviewer:** so the first model was the vanilla LSTM so for this I imported the libraries um I built or defined the function for building a recurrent neural network um by added in so I used the LSTM class there just to have some LSTM layers and I wanted to automate the hyperparameter tuning So what I did is I created a loop that would kind of create different numbers of layers depending on um so he's using randomized search CV to explore different numbers of layers so I have a loop here for the layers and then I have the LSTM layer in each in the loop that players will be LSTM layers and and then finally I have just the output layer here and then in terms of the loss function I used to mean squared error because just because in my thesis title and all that did the measurement I'm using is mean squared error so that's why I just wanted it always to be mean squared error just to get that view of each each model and then anyway I wrapped it up because I wanted to use randomized search CV to try and automate hyperparameter tuning just to save some time because it took a long time to train each model so I I got randomized search CV to search through all of these parameters here and to like basically explore which ones performed the best so the number of nodes in each layer the dropout rate the batch size the epochs the optimizer was either Adam or RMs. prop and then the number of layers as well in to include as well and then basically from there I'll just go through the results so I trained and and all of that and then I'll just go through the test results

**Subject:** so I'm just going to go to the input shape and output shape of the shape so the input shape and output shape of

**Interviewer:** sorry of which when will I listen

**Subject:** So what was the size of the input or the initial

**Interviewer:** ohh sorry the output layers wasn't mentioned as yes right so input shape sorry was uh 20 times steps

**Subject:** OK one what you had only one feature

**Interviewer:** yeah yeah yeah 20 times steps and one feature

**Subject:** yeah yeah OK thanks

**Interviewer:** yes that's right and so here's the test results so these are the 20 day predictions so the red line is the predictions and the black line is the real data and so that's how roughly similar they were and then there was a mean squared error calculating for that because I didn't set a random seed every time I run the code the the score is different so I have the table separately that just kind of takes kind of one run at all the models and what the results are just to have a fair comparison so the results might look a little bit different to this here but umm it's just just so they're all recorded in one running of the code

**Subject:** so just one second here you can just give me one minute yeah sure I'll be back

**Interviewer:** yeah ohh yeah no worries no worries take your time

Continuation

**Subject:** Yes go ahead

**Interviewer:** yeah no worries at all and so that was the first variant so the next one was the Gru and so it's simpler structure combining couple of the gates together is the is why this was picked or what it does so I started with the default parameters and the input shape there just the same as before I used the Gru class to make the layers Gru layers and then I had the loop to add in more layers like in the previous model and again all gru layers and powerful the output layer there which just outputs the final value and I hard coded the loss function that's the main squared error again again and I did all the same things as I did in the last model so that the randomized search CV searching all of these different hyperparameters was done and then and then again just training and testing and all of that so the results of the test for that are here so the predictions in red and the real data in black so that's uh Gru and if you want I can run through all of them but if anything it does occur to you know feel free to say but I will ask at the end as well you know your overall thoughts um next one is LSTM with attention mechanism so for this one structure is a bit different but still input shape is still the same 20 time steps one feature and LSTM layers and but here I've added in an attention mechanism to capture the important parts of the input sequence and then then the final LSTM layer is there and the output layer is OK and and yeah same as before the mean squared error is hard coded as the loss function and the hyper parameter is all the same and then the results are here so red is predictions on black is the real data so

**Subject:** that's the the peak has not been predicted it's it's considering a very smooth curve so any idea of what that peak is say is there any reason why there was a sudden increase in the prediction like oil price this is another tip is there any reason that you could find that's a good start looks like an anomaly or yeah a very non like non frequent occurring of a stock price yeah So what I in my case like well your case isn't is in terms of how these variants of LSTM works so you don't you might not worry about how the data is but what I would have I would have taken a more cleaner data yeah test the model model application on that because so here what happens is the variation might be occurring due to the model might still be trying to do the prediction but the data is not clean so it might be an anomaly whatever you are it's just an assumption so you need to give it a try from your end to understand whether the actual oil price was like this or was there a reason why it happened in this particular format so you can check out if that's an anomaly and if it's an anomaly you can replace it with if you can you have replaced the missing values with whatever the forward filling and backward OK so anomalies are also something that are not necessarily be forecasted by the models those are rare occurrences so when you're using the data you need to make sure that anomalies are also removed so initially when I actually worked at Honeywell where I was working on HVAC related data where I was supposed to focus in a particular part of the system is going to fail so I was trying to predict the shelf life of any of the system devices like sensors and cooling units so that time one of the major thing even before forecasting when a particular system device was failing what I did was were there any anomalies present in the training data because ideally the model when it's straight it should not have any anomalies present in it so it should learn on the actual pure data so if there are anomalies the model is going to get confused so model like you're gonna get a very bad prediction but the reason for the bad prediction is bad data yeah so my one solution would be to check for the data why there is a peak and if it's an anomaly get rid of it with any any mechanisms like you can do the averaging or for provision of tracker propagation or whatever whichever ways just replace the anomalies with probability data that's that's what

**Interviewer:** yeah well that's a very good point Anonymous thanks for that yet and something I didn't consider actually is that how that could be an anomaly that data point so yeah no thanks that's something I could look into and all right for that one and so I'll just move on to the forget gate bias 1 so this one the first having a bias in the forget gate and to decide what to forget or not forget and so there's a value I need to set for the forget bias which initially is one but I've also I've also included it in the randomized search CV as a parameter that can be tuned through randomized search CV so from that as well so in this model have the LSTM layers input there and then I have the output I have the uh sorry the forget bias also initialized there and and then just more after the loop again with more LSTM layers and with the bias and also included and then same all the way through until the last layer is just everything's taken out it's just the output layer and then then in the parameter distribution here in the search space here I've included forget bias as a as a parameter that can be searched for between 0.5 and 2.0 just to see which one works yes then in terms of results then result is here so red predictions and black is the real data so that's just how the forget gate bias will performed so and the next one so there's just two left the next one is basically a combination of the two that went before it so it's attention mechanism and and forget gate bias so this one was probably the the most complex put together one but anyway I have the LSTM layers with the forget bias in there as well it might have the loop for more layers and have the attention mechanism put in there and then you can see the forget bias and attention there is in there as well and output layer is here and then yeah say everything else is the same again in terms of the search space for randomized search CV and then the results of that are here so this this one didn't perform very well yeah so anyway yeah red predictions black real data so we can see what happened there that one wasn't one of the stronger ones and finally this is the last one so bidirectional LSTM and so looking both forwards and backwards to capture the context so this one wasn't too difficult to implement so just there was a bidirectional class we could include in each layer which I've done throughout there and everything else is very similar to some of the previous ones and then the results are finally here so this is the red is the predictions and the black is the is the real data so they're the they're the six variants if you want I can reshare just showing the table of the results or if you want I can kind of keep this

**Subject:** what mission is uh on what basis did you uh come up with the the situation that 20 should be the lag. For the prediction yeah so any specific that's a cool specific process we have specific process or specific experiments that you have considered to do to do this would you choose 20 basically

**Interviewer:** yeah that's that's a good question so so there there wasn't really um so in terms of the the time steps no 20 was was just considered as like a standard but there was no yeah there was no process followed to look into that like to to try other ones

**Subject:** because if you see the predictions and the time period of those predictions you can see that for this particular model bidirectional LSTM and also in the LSTM with the attention mechanism if you just scroll you can scroll the graphs sure yeah OK yeah if you see here it's it's having that at the time lag there is there is like a small gap between the prediction it is able to do the prediction it is able to identify the peak it is equal to identify the lower part as well but it's identifying it at a different timestep yeah for example if you see the tip is at 12.5 but it is identifying it as 30 at something like 13 or 13.5 so this shows us a bit of it's again an assumption so ideally when you choose a lag you need to have a specific process to choose the so I think you need to have domain knowledge to choose the lag or probably you can use some of the statistical approaches to identify if there are multiple statistical approaches like I might be naming the method wrong but please Google it like there's something called as Jenkins approach OK if I'm not wrong which can help you identify the lags and there are few built in time built in Python packages which will help you identify the lags if you just use chat gpt or anything in Google you are going to find a lot of packages and part of the method but there is one thing where you plot a graph between PO PAC and so there's something called simpax I if I'm not if I'm not mistaken it's been a while since I've worked on this but you they actually plot a graph and based on the plot they are able to identify when which particular lag is being should be chosen on what basis so my first suggestion would be to identify the correct lag and experiment with the client you have to notebook ready so just try to convert it into a Python script where lags can be taken as a params and just run it on multiple stuff because none of these approaches are concrete it's not going to give you the 100% perfect lag unless it's from the domain knowledge so I would suggest get our tryout all the lights which these packages are same and then see the predictions that will be the first thing also is this data fixed or is there any way that you can add features into this because LSTM is a model that learns more from the patterns and but so it's not like it's learns from the predictions like so if I'm putting it in a right way it's better I say it like this say I give you an example differentiation between LSTM and random forest random forest tries to associate itself with the output LSTM tries to associate it with with the input it tries to see the patterns within the data so you see the difference so random forest you don't need to check if there is a pattern within the data to understand the so it tries to associate the input with the but here it tries to find the pattern within the input to predict the output yeah so I would suggest if you are able to include the features into the current data so right now we have only the oil prediction so you have taste you have time so you have like these days about business working hours and stuff so positively there are if there are some features that are impacting the oil prices so probably I'm really unaware about the domain but for for an example if I give like if I'm trying to do a stock price prediction I would like to give the GDP as one of the beaches for the stock price prediction or probably give something like for example if I'm forecasting the amount of water that is going out of a particular dam then I'm gonna give the rainfall as a feature even if it is not present I'm going to extract it from the external source and include that feature to the data set and then to the prediction yeah yeah so yeah just just give it a thought if it's if you have had the resources and if you have the findings to add features into data that would be a really good because it's gonna work well with the LSTM because when I worked on LSTM I had around 1000 features 1000 different features and I used the lag of 18 minutes for the lag and it was creating features in the range of 10,000 to 20,000 so that's that's the amount of features that can like the bigger the features LSTM is still able to predict the buttons yeah it's not like with less features less time is not gonna predict it is gonna predictable patterns but higher the features as well other models might not be able to predict it but LSTM or Gru might be so yeah that's that's what I'm trying to say so have you uh check out auto encoders LSTM autoencoders or GRE auto encoders and I haven't like just give it a look OK those are those are the better versions of LSTM they are so you are aware of Transformers right yeah so the auto encoder is something like an encoder decoder architecture so if you have a set of features it's gonna encode it into a particular shape say if you have an input of 20 cross one it is going to encode it into four cross one and then try to decode the 20 cross one back from the four cross 1 so it's trying to learn the pattern from the very minimalistic data and try to recreate the and in that in that way it's going to understand the pattern very clearly so just give you the thought if you can make use of auto encoder as well because when you work with LSTM and Gru auto so it always starts like this LSTM Gru auto encoder yeah just give it a thought and just just check if you have the time and stuff you can just give it like give it like a look but apart from that it looks good the experiment looks good and I would suggest working on the data a bit more to make it more cleaner and more like resourceful I would say but the data should have the information that a model is trying to look for if there is no information that the data has then the model is not going to work so try to make the data in better and that's that's that's all I guess

**Interviewer:** yeah just one more question I know there's lot that you gave there so thanks a million for that I just want to show you the table of results and so in in order there I have them in separately once that performed the best in terms of mean squared errors so georgiu is at the top and vanilla and bidirectional forget gate bias and attention mechanism are in the middle and and then and then the combination of the two is is bottomed by far so they performed the worst so I just wanted to ask you do you have any thoughts on that performance difference between the different ones are you surprised or is there any observations you have on the ones that achieve the best mean squared error and ones that achieve the worst or surrounding what you have

**Subject:** just on that information I do think that why attention mechanism and forget gate bias as my dev work a bit bad because there is not enough features in the data because it's a it's a it's a univariate time series if I'm not wrong right you have a single feature yeah so it's a univariate time series but forget case get into mechanism and the attention mechanisms or if you're using deep LSTM like multiple layers of LSTM or TPCR use those models work well with huge information huge features so with multivariate time series specifically so that might be the reason why these models are not able to predict well also they might be a bit overfitting as well because like because they tried to another the minimalistic data as well so that might be the reason why they're working but Gru and vanilla they are very simple so now the data is also very simple you have anyway so that should be the the network is capable of learning very simple information from that sample data so that might be looks like the reason why it's performing better than the rest of the stuff

**Interviewer:** Ok yeah I think that's that sounds good to me as well and then anyway that's that's everything I wanted to cover so I just wanted to really thank you for your time Anonymous on that I know you're you're you've been busy but I really appreciate you taking that time if you if you want I should be finished in about a months time if you want I can share it a paper with you but it's entirely up to yourself

**Subject:** so let let me have a look at this paper once you have done you can just share it to me and I'll have a look

**Interviewer:** OK anyway thanks for me and I hope you get time for some lunch and just have a great day like thanks a million

## 8.7 Interview D Transcript

**Researcher**: OK great so I'll just share the screen now for now and get straight into the into what I what I have so you can just let me know when the screen appears for you there

**Subject**: Yeah I have it yeah

**Researcher**: Brilliant thanks so there's six variants to get through so I'll try and go through them as kind of quickly as I can but at the same time try and explain so the first variance that I was looking at is the vanilla LSTM so which is just you know the the the kind of basic original 1 and how I built it here as you can see here I've defined a function here and then I've added in so I've added in the LSTM layer here with the dropout rate and then I've created the loop to kind of add in more layers what I wanted to do was kind of for hyperparameter tuning I wanted to automate it so I used randomized search CV that could like experiment with different numbers of layers different numbers of units and that kind of thing and to see which ones give the best results so that's kind of what's happening with the the loop there and the and then the final output layer is there and I hard coded the mean squared error as the loss function just because for my thesis that's the measurement I'm using and then from there I use the keras model as a psychic learn model for the cross validation so I wrapped it so that I could do the the hyperparameter kind of cross validation this is where I defined kind of what the the hyperparameters space is what what I would be trying or the program would be trying differently each time in the randomized search CV to try and get the best results and anyway I'll skip down to the results because I won't go through everything for all the variants because it's a lot there

These are the kind of results so the red line is to predictions and then the black line is the real data so kind of relatively close there and then the mean squared error result I got there I actually have the results recorded in a table separately that I can show you at the end so we won't worry too much about what this says because the results are different every time at the moment because I haven't set a random seed for it so anyway the the next one is grew and so kind of the simpler

**Subject**: I'm sorry I just ask you for a second, I wanted to ask what type of data are you working on

**Researcher**: ohh sorry so sorry yes so it's so it's a regression problem and the data is uh financial time series data

**Subject**: your Financial Times series data for a particular company

**Researcher**: um ohh so sorry it's um so it's basically just what I could get so it's oil oil prices from 1988 to 1998 so that's the the data that I'm working with here

**Subject**: what is the objective of the thesis is it to predict the oil prices accurately

**Researcher**: so it's it's actually more to do with the the process it's so it's more to do with the four objectives are basically find the best hyperparameters for each LSTM variant to get the best mean squared error result and kind of identify the strengths and weaknesses of each of the variants that I use and then also identify the variance that has achieves the best score basically from mean squared error point of view and then the 4th one is just basically come up with the factors of why the best LSTM variant performed the best so like what were the factors behind that outcome really so it's more concerned with the variance themselves and how they perform and coming up with the reasons as to why they performed how they performed

**Subject**: The reason I was asking you about this was because I want to understand what the best means in this scenario. Because for every model or any deep learning model the best is dependent on the data and what we are trying to achieve the best on. Some data we want mean square, as you're doing so this one is mean square.

**Researcher**: yeah

**Subject**: But for some other kind of data, it's sometimes if one score sometimes, it's recall. So you want to be sure of what we want to maximize. Because if you maximize one, they're going to be downstairs on the other side.

**Researcher**: yeah yeah

**Subject**: We can continue. I got the idea what we are looking for

**Researcher**: perfect yeah thanks thanks up enough yeah no good to any questions you have feel free to ask them. So yeah so mean squared error is the one I decided in the title so it's just that's just why like there might in practice be a better metric but that's just the one I've gone for this particular study. So the second variant is the GRU variant so it's a bit more simple in in its structure so again just the difference being different each layer i put in the GRU used the GRU class and to combine two of the gates and oops and then I sorry just have to open this up again so we're in OK so that's so this is where I built the the layers here again with the loop and again it's just the GRU class is in each layer and there's the output layer there with one unit and then I'll just skip down to the results for this one and so there's the kind of visual of it so the predictions are in red, the real data is in black, and in the mean squared error is what I used to measure it again. And the the next variant is an LSTM with an attention mechanism so where the focus is on you know the important kind of inputs or identifies important inputs and focuses on them. So this one is built a little bit differently but again I defined a build or an end function and brought in the LSTM there there and it's down below where I have the attention mechanism within the loop and then basically I build the LSTM layers again but included the attention mechanism and then yeah just all the usual then I inputs and outputs are there and I compiled the model there so there are no differences just that I included the attention mechanism this time in the LSTM model and the results are here so the red is the predictions and the black is the real data so yeah again and then the mean squared error is there and we'll go through those results at the end. The next one is an LSTM with forget gate bias so this time and it decides what not to forget and what to forget I've built it here and so I've set the forget gate bias as one initially value of 1 initially but then I I use that in the randomized search CVV I'll use to forget bias as another hyperparameter that can be kind of explored so even though it starts at one it goes through other values so so I've initialized the buyers value in the first LSTM layer and then I've gone through the built out the rest of the model with the forget bias in it throughout there and there as well the output layer is here and then anyway so in the parameter distribute in the search space I put in the forget bias as another value that can be searched and then yeah and then the results are here so predictions in red and real data in black so again they're the the results and the mean squared error was just calculated there so there's just two two more to go through so that the second last one is an LSTM layer with a combination of the two before that with the attention mechanism and with the forget gate bias so I'll go to the model so again. It's just a combination of the two I did before so I've said the forget bias to one and so yeah there's the bias and the attention mechanism is here and then again just a combination of the attention mechanism and the bias in the layers throughout until the end and and the output layer is there and then the results are below. So this is the results for that one so so the there's there's maybe too much of a bias there. The results were way too linear I suppose there's no kind of there are two simple there's no complexity there. And then the final LSTM was bidirectional so where it's kind of evaluating both forward and backwards and to get more context. And so so then the bidirectional class was used in the in the layers and again just bidirectional classes used there again in each of the layers up until the and then the final layer just is more simple. And the results are here, so the red is predictions and the black is real data so that's how that one performed. So anyway that's quickly what the models are. I'm just going to show you what the results are going to summarize the results. Let me make sure you can see that. Wait a second I think I need to re share my screen to show you the separate and separate view. Quote window if possible. Is that showing up okay?

**Subject**: Yes, yes

**Researcher**: OK perfect. So sorry I know that was very quick. But I just want to summarise the results that I got. So so starting from the top is the kind of best performing one as measured by mean square error. So the GRU was the best with 0.100, vanilla LSTM was next, bidirectional was after that and then basically attention mechanism and the forget gate bias and attention mechanism combined that was by far the worst one so performed a lot worse than the others they forget great bias and attention mechanism we're kind of in the middle they both here which is 0.15 S and then the kind of simpler models where the ones that kind of performed the best really here and so anyway I've talked a lot there so I just want to maybe get is there anything that stands out to you or any thoughts you have initially on what I've done there how I've done it or anything to do with the findings that you you want to ask or you think I should look into further

**Subject**: One thing I want to understand is about the data.

**Researcher**: yeah

**Subject**: So uh is the data point which you see on the graph, is it per day or is it per week?

**Researcher**: Yeah good question. So the data is basically it's one value a day from Monday to Friday. So it's the the work days Monday to Friday over a period of what I included was over a period of 10 years. There are some days where there were no value because it was a public holiday in Europe these are European prices. So there there are some days where there was no value so I used the forward fill and the back fill method to input those to to fill in a value for those missing days. And and then yeah that's it really so Monday to Friday throughout the year, for 10 years. I actually had a longer time range initially it was over like 35 years but then the data was very volatile of the second half of that kind of period. So I I actually took it out just to make it simpler. The data was a lot steadier in the first 10 years. So I just focused on that for now and then and yeah that's that's pretty much it. And then like I said I did normalize the data as well before doing each running through each model. I used I used it's a Min-Max scaler or which one did I use no I think it was standard scaler I used yet because the data was more normalized was more gaussian for the first 10 years so that's why I used standard scaler as the scaler because it was more gaussian basically.

**Subject**: Can you show me the jupiter notebook again. I want to see the graph…

**Researcher**: yes yeah yeah so that would be more helpful

**Subject**: The thing that is standing out that I would like to ask is attention-based LSTM is performing worse than GRU, I think. And that’s weird.

**Researcher**: The attention mechanism one was performing worse is it?

**Subject**: Ah yeah

**Researcher**: Yeah let see.

**Subject**: This this one here

**Researcher**: This one is attention mechanism I believe.

**Subject**: Yeah. I just want to look at the graph. So this black line I want to understand what one point of the black line means. So it's 2.5 and 12.50 what does that mean?

**Researcher**: Oh sorry yeah so yeah. So that the Y axis is the prices and then the X axis is that's meant to be representing the the days of from Monday to Friday so yeah the way I have it there. It's not showing the days. I think it's just taking them as either an index…

I was trying to predict the next 20 days so I think it's just like 0 up to 20 so. It's just I don't know it's it's picked them randomly 2 1/2 five 7 1/2 days but I think like the total value there should be going up to 20. And it's just going through the different…

**Subject**: Is it like starting from today, are we predicting what's it gonna be in the next 20 minutes?

**Researcher**: So it's starting from so the data is gone from 1988 to 1998 and the 20 days at the end of 1998. They are the days that I tried to predict. So that's what these days are, they are the last 20 days in 1998.

**Subject**: Understood, understood. And then you find out this mean squared error for everyone of the day like first day, second day, third day. Yeah makes sense. Ok.

So then one more thing which I want to clarify - what is the like whatever input you give, you just keep oil prices and the day or did you put any other information about it on the morning?

**Researcher**: Yeah it was just it was just the prices on the day yeah

**Subject**: Ah that explains why attention is getting bad.

**Researcher**: OK that's an interesting one yeah

**Subject**: Because because the thing LSTM is, I don't recommend LSTM. Like it's a good good one for time series data. I don't recommend it if you're doing a lot of years and just data point and time. You need more information in LSTM to make it stronger. For example in this scenario, oil prediction - what volume depend on let's see from the holistic perspective it depends on the oil prices yes previous oil prices that's it good metric yes it depends on geopolitical situation depends on tensions between governments, right? It depends on trade issues. So you need to fit in to make attention or forget bias more relevant, you need to fit in more data. You need to push in this as one of the cells data point. Then you need to push in maybe new sentiments. Sentiments of people uh over the period of that particular time frame. In the same time frame you can also put in uh trade how much trade happened between different countries but focus on gulf nations and Russia because they were they are more major oil importers exporters. See trade relation between Europe and then how much trade bulk is happening in that time frame. If you use these three metrics maybe LSTM attention will get more better. Because technically from what other people have done research, attention should perform better than normal LSTM, right? In your particular scenario, if you just have oil data and time item, I think that's why GAN is performing. Will you try GAN until now? You have tried GRU right?

**Researcher**: Yeah, Gru yeah

**Subject**: I think that's why it's performing better because it's it's basic. It does not require a lot of complicity that's why it's performing better. Because your input is basic. And if you try to put more complexity into basic stuff it kind of gets biased to what's happened. So I think that's what's happening, that's what happening with attention and forget bias that's why it's flattening out and getting overtrain basically. So one more thing you can do to make it better if you don't have any other data, is to find out the ***D curve*** of every model. I don't see that till now. Right. So for every model find out the ***D curve*** and how many epochs does the model get trained. Because if your epoch is like if you're doing 20 epochs for every model, some models will get overtrained, and some will get under-train. So you need to find a perfect epoch for each model.

**Researcher**: Yeah

**Subject**: So once you find the knee, it's called knee graph right? I did this a couple of years ago.

**Researcher**: No worries at all. So I so yeah for epochs, I explored between 10 to 30 epochs for a lot of them just to see which number performed the best. And then in the training results it would tell me which which number of epochs is optimal so for example in this one it said 30 was the best umm in some of the other models like I said 20 and and so on and I have that just in these results.

**Subject**: Here, you need to do it one by one. Like you need to start from epoch 1, then 2, and 3. I will share you one link of detecting a knee point the elbow point or some some people call it knee point some people call it elbow point.

**Researcher**: OK

**Subject**: You need to find that in your uh model. Because it's like a it's like a log graph. So it it trains faster then it flattens out. As soon as it starts to flatten out that's when your model is getting overtrained yeah and if that is happening your model is going to go in a bias situation and start to get bad results.

**Researcher**: yeah yeah

**Subject**: That’s the problem (19:26)

**Researcher**: So you think so do you think a more manual approach would have been better to identify where that elbow near elbow point occurs?

**Subject**: No, I think there are some libraries in Python which helps you find a knee elbow point. I have done some in my research when I was doing bot, let me see my knee elbow point (19:56). You're using Keras right?

**Researcher**: Right yeah yes that's it.

**Subject**: I got an example in stack overflow for K-means but it's similar.

**Researcher:** yeah

**Subject**: Something similar. If you're starting to find it Bing your link then I'll send you some more materials

**Researcher:** yeah yeah that would be great

**Subject**: umm like in in K means classification you have number of K right how many clusters you want to make. In this scenario you need how many epochs I want to make. And how many epochs what would be umm yup… First, figure out the epoch, I would say. Because I I think, the problem which is happening is because of more epochs, even bad sizes. So it's actually 3 cross three matrix three cross three matrix nine yeah. like 10/20/30 is a very huge number. {OK yeah} For attention span I think maybe the epochs would be two or three, {yeah} it would not be more than that. Because otherwise it's trying to learn more information which is not there. Like if you get more information there are two approaches, you get more information then put it in then it tries to understand the complexity of the system and return according to that. {right yeah} Otherwise make it less, like don't let it work a lot. Let it work for once or twice that's it.

**Researcher:** yeah so maybe I should lower the range of epochs that are explored here

**Subject**: yes, you were to lower down the epochs and you can make the batch size also less yeah or more. I'm not sure but you need you will get a 3 + 3 matrix technically. {yeah} So what this is what you can do, not the 3 + 3 matrix, first figure out the epoch. {yeah} from the knee curve. There are some libraries and keras can help you do that faster. You do that even model optimizer play a very huge role. As far as I remember there were 5 or 6 which are common. You are trying out 2 here.

**Researcher:** Over guard and optimizer yeah yeah that could have been yeah

**Subject**: When is your submission of this thesis is it still in the future or is it nearby? I can recommend a model

**Researcher: y**eah it's a it's it's towards yeah it's towards the end of this month. 27th yeah

**Subject**: Yup you have enough time. First things first, find out the knee point of each one. For GRU will be different, for everyone it will be different. So you need to find out a knee point and then you start to do this analysis on which optimizer works well then you can go afterwards that. After that if you are OK with optimizer, I don't think batch is going to affect a lot because you don't have a lot of data. So it's fine. Figure out the knee point then look at model optimizer. I think the results will get better. Right now the results are choppy {yeah}. One more thing which might help you is, like I did I think I did similar in oil one day before but I did not do it well. And then I did a stock price analysis of certain companies but I did not rely on past historical time series metric because that cannot predict the next move, right. So {yeah } one thing I realized in ML was after a lot of time I spent into it was that you need to get your data processing first very strong. You need to understand what all things affect this price. It's not just the past data. If it were just past data, everybody else would be millionaires right now. So we need to figure out what all points affect that price. And if you're making a complex model, push it with a lot of data. But you clean it first. So in your thesis and the scope of the thesis, I'm not sure if you have more data or you have time left to clean more data. So I think this is a better approach, getting a knee point and proceeding from there. {Yeah} If you're struggling with where to find the knee point and all, I can look at more research which is also after this call and I can send you on LinkedIn.

**Researcher:** yeah no no problem I get the I get the idea I think I should be able to yeah I'll just search you know libraries for yeah yeah using for for getting the knee point. so So in terms of…

**Subject**: That is one which will help you a lot like. It would be the most ROI for you.

**Researcher:** Yeah and just in terms of the results when you say choppy do you mean across all the variants or just the attention mechanism one?

**Subject**: the only thing which stands out is like I was sure that GRU is gonna be OK. I was not having any doubts I was also sure that vanilla LSTM would be OK-ish not so bad. {Yeah} But when you see uh comparing those results when I go and see LSTM with attention LSTM with attention and both of these combined together getting very bad results than vanilla, it just feels like the data has not been processed properly or we did not find the proper point for those particular models. Like it is possible that even when you find the knee point it would still be bad than…,but then realistically, it can be possible. But I was just shocked by the straight line around when we combined both of them together. Because that straight line clearly says it's over fitted model. {yeah yeah} So that that's the thing professors might be a bit like my professors used to be very familiar about how it fitted models {yeah yeah yea}. And that was that about that is what I meant by chopping and not the other things. The other things that I I like your thesis {yeah}, I understand what you're trying to do. But uh if you're I'm not sure what the professor would say. But I remember my professor scolding me when I showed something like this. {yeah} It was like what have you done and at that point of time I did not know what overfitting means. Because… (26:40). And then I tried very deep into understanding how we can remove overfitting stuff {yeah yeah}

**Interviewer**: no that's that's good you know that's great Anonymous thanks man you you've covered actually the areas that I wanted to discuss there yourself so and thanks main for for doing that and I know we're kind of on time now so I don't know if you have another meeting or anything but umm I really the the feedback you've given there and umm yeah I'll take that off like I'll I'll have this recorded so I'll take all that on board and be sure to include somewhere in the project as well that feedback because that's the kind of feedback I'm looking for so

## 8.8 Interview E Transcript

**Researcher**: Right anyway yeah so just a quick overview of this so I was doing a master's thesis and basically comparing 6 different variants of LSTM networks and models and then assessing which one performed the best from the point of view with mean squared error on financial time series data which was taken over it's like it's basically 10 years of data and that's one data Friday price of oil of in the EU from Monday to Friday and over 10 years so that's that's kind of the the basics of of what it was. So I was just planning on showing you some of the codes not not all of it but just some of the main parts where I built the different LSTM variants and I know I was just gonna ask for your thoughts afterwards on basically whatever occurs to you whether you think there's any improvements that could be made anything that's surprising to you and like I'll take you through the results that I got and you just let me know like it's transfer you to let me know yeah just what what your observations they're basically and like I'll just take them and and I'm sure there there could be some valuable in there so it does no pressure or anything but it's just just an exercise really and you can let me know along the way

**Subject**: yeah I took a quick look before Sunday. So I've already seen a bit but not everything. I didn’t have the time to follow the whole code in detail but I checked the full file. So you can go ahead and go through it pretty much.

**Researcher**: thanks very much and I'll just share the screen there or try share my entire screen if I can. right just the window then not sure what's going on there OK let me know if you can see the screen.

**Subject**: Yes

**Researcher**: OK and so like I said there were six different variants of LSTM models that I went through so the first one is just the vanilla LSTM the most one of the more basic ones and so how I built the model is after importing these libraries and I built uh uh to find the function here for building recurrent neural network and then we use the LSTM class in each layer and I made a loop to kind of create more layers and the output layer is here and I hard coded the mean squared error as the loss function because just as part of my project that's the metric I'm using to compare each model to one another and and the reason I have that loop in there is because I wanted to do some cross validation and examining kind of with different level validating different uh numbers of layers to see which you know numbers of layers performs the best so I wrapped that and it keras progressor and so I could perform so I could automate that hyperparameter tuning the hyperparameters and space that I defined is here so I was I wanted to see how many and how many nodes in each layer would work best well the best dropout rate would be the batch size the number of epochs the optimizer just between Adam and RMS prop and then the number of layers like I said as well so I did this for each variant and then yeah just used randomized search CV as the kind of cross validation to try and find the best hyperparameters great and I'll just go down to the results for this particular one or the least the graph so the predictions are there in the red line and then the black is the real data So what I was looking to predict basically was the last 20 data points in this series or the last 20 days prices for the last 20 days of 1998 I believe in the data and then so that's roughly how we performed in the in the vanilla LSTM so that's the first one

**Subject**: One question about that

**Researcher**: Sure yeah

**Subject**: So the lag. The lag that’s in there

**Researcher**: The lag here?

**Subject**: Yeah, so you see that it seems that it correlates quite well right? {yeah yeah} There's a lag of three days maybe, no not three, only one day at the start, right?

**Researcher**: yeah yeah I see I see that lag yeah yeah

**Subject**: So if you check with if it's an artifact of the way that you adjust or clean your dataset, right?

**Researcher**: and that's a good point so uh truth be told I haven't really um it is something that I've observed as well and I thought it might be just the model's way of learning and whether there's in terms of time steps um I used 20 time steps and so I just to be honest I haven't looked yet into why that lag occurs I thought it might be just the model's way of learning and trying to keep it's just a bit slower than the real data so um so that's an interesting so just out of interest and if you from seeing lags in the past or before what what would normally be a contributor to a lag like that. Have you ever experienced something like that before?

**Subject**: Honestly this is not something that I've seen whilst using LSTM. That's why I wonder if it has something to do with the way you do the algebra on tensor, right?

**Researcher**: OK yeah

**Subject**: Lagged predictions, I mean, there's something I would have changed in the way you did the training on these datasets for sure. So for instance I would have done the cross validation. I know it consumes more time, right. But this is instead of just doing it on the last 20 days which is very short due to being the same as your time window right. So you're only checking one time window. {yeah yeah that's right} But I will do is just divide so you know time series you can do a cross validation with different sizes of validation set, right. {OK} So you could start let's say you do 3 instead of 5 because I know that LSTM are great they take a lot of time to train. {yeah} So if you this like the way I said, you would start with 1/3 for training, 2/3 for test, then 2/3 and 1/3, right, so in that way you will have a bigger view. And always try to have it different than the time window. Because that in there is a bit you've seen the same number could be weird and {OK} the problem with the so the problem in general is time series regression, right. And this is already an ongoing problem in feedback. {yeah} You're not supposed to be able to predict the value itself, right. And if you can, I mean you can predict in a short range that you wouldn't be ever be able to make money on it because at some point your threshold for let's say we call this stop loss take profit uh should be so close uh in a mid term or out of gas right of of shorting or longing you will be losing money right because your predictions are really so close to it. So whenever a prediction or a regression on a time series lights for price of stocks and invested for it should be the same super close uh it's not it's actually not reliable on production. {yeah}. OK so that's why I tell you you should just give it a cross validation right as a time series of different sizes and in that sense don't use replacement OK because here I suspect that the way you gather the data and you define the the dimension that has a size of 20 right some things inside your extensor should contain too much information about the future by itself that makes this relation work better it works as some sort of replacement so for instance let's say you are in the in the minus 40 right {yeah} and then you go to the -39 in that place you improve the first step of your test set {yeah} then my etcetera you're improving some data from the test set I know I'm not saying it happened here but you have to check OK.

**Researcher**: so you're you're talking about kind of data leakage is it

**Subject**: yeah yeah {OK} you have to be very careful about some leakage from the test setting to the training set imagine that you even in your extremist (10:10) right this could happen somehow because your dimension that goes in depth with all the data it could improve as they said data it's actually happening acting as some sort of replacement right it's as I said you have the real data in there part of the X vector sorry as part of the extensor and then you're predicting the Y and since the data you're using in the X is not data you already predicted but just data that was originally in the wild right because it's a auto regression at the end of the right yes you use that then you're prediction will be very close to the real point because this one is correct because it's part of the extensor right yeah I mean so you're actually leaking from the future in there the way you do the the prediction right the the the way you you are doing the regression so you have to be very very very careful in separating this data and very careful on how you define extensor for each of your predictions so if if you want to be 100% honest with your model you have to to see the very beginning you have to use uh prediction data that's why it would also make these uh test datasets larger right so you have more than you can do it better because if you only have 28 bit 13 minutes (11:56) and then so you have years you can use let's say the whole past right let's say you have the 20 previous samples right in there and you ready the first point in the in the test set that should be very close because it's really connected but then what you have that's X as zero {yeah} when you have X as one it should be using your prediction instead of why why zero right as part of it yeah and then and that's uh successively right. In that sense, you're only using your predictions you are not of course there are models that include replacement you love there's (12:46) there is no argument in any place right but somebody could come in your test in your sorry, in your thesis um are you using the replacement here that's why it's so close Despite that right so I would use both cases right we can without replacement and just see you could see none let's call it in an online setting right if your model was uh in the server and you're doing logs and retraining etcetera etcetera you can always use replacement of the last point that will provide you of a correction right so I would make both and say we from without replacement here in the for the prediction and I would expect from things I've seen and I did this dataset of size when I started data science as well with stocks right I would expect that the longer you are from the future, the more it deviates from the real thing. {yeah} that's what happened in ther, right. But, as I have said, I don't mean this is wrong I would check the lag that's an interesting thing to check right. {yeah} by the same time I recommend you to have uh some sort of profile relation with larger test sets right if you want to do try two different sizes of the test set but always different in the time window just in case so otherwise it's straight. Different and also larger time window otherwise it won't work right.

**Researcher**: yeah I think I hear you so yeah

**Subject**: submit these make these trials (14:34) it will complete your experiments better. It will give them some robustness to any conclusion you you have that's my recommendation. yes everything else and…

**Researcher**: so just the question for in terms of the time steps should there be twice as many time steps as or sorry should the test set be twice as large as the number of time steps that you use in order to avoid that leakage

**Subject**: you can always avoid the leakage by hand right you have to adjust how are you defining the X test (15:18) or the sampling, right at any given point. So yeah so you can always solve that I would and this is something I made right. I may have to solve a predicting mathematics problem (15:34) besides the LSTM science rival (15:40) when I wanted a time window that was correct for my problem. I was not using LSTM that are more, I was using one-dimensional CML right {yeah} I need something for regression something focus dictation was two different things but it applies here I also define the time window so hyper parameter I know it's not the common thing. But sometimes if you want to optimize the problem, it's worth having some exploration of it. So {yeah} because what makes you decide to argue or not for full year or a month or a week now but there's no real reason behind yeah the the window of memory you want to preserve right so if you have time or if you have time right and I know many people do this arbitrarily people have time just toss sometime windows in your random set and see which one fits you better.

**Researcher**: yeah yeah. That make sense thanks for that question then that's a very good point that I'll need to look into and it'll I'm sure it will come up with all the other variants as well since I used the same data for each one and if it's OK I'll just quickly go through the other ones and then just the results and just to get your thoughts on how they compare and I'm sure they'll all have the same if there's a if there's an issue with the test set the way it was defined in the first one it will come up in all of the other ones as well I'm sure it's the same one was used well just I'll just go into to the other variants that I have. So the next one is the GRU. I used the Gru class and defined and defined the build RNM function again and put in the Gru class in each layer apart from the last one and then the results for this one so I did I did basically the same thing for every LSTM variant so this was the results then for the GRU. I think you'll notice again there there is a lag looks like and so it could be very exactly what you said before maybe could applies as well here maybe.

**Subject**: Just give it a look because it's worth checking why the lag is happening just try some some other stuff later to see what's going on I would expect the GRU to be worse than LSTM. Sometimes it’s funny but LSTM bites(18:22) are difficult depending on the data you know your data is for instance is not um adjusting to uh normal distribution right. you will be the beginning of the (18:36)

**Researcher**: well actually yeah what I did what I did sorry was so

**Subject**: that's right did you selected the year

**Researcher**: yeah yeah I selected the easier years

**Subject**: that’s good how you are looking at it, because otherwise LSTM can have these problems right any of these difficult to to just adjust whole hyperparameter when the data is not good enough you have I have a book chapter if you want to check with my name uh {yes} editor is in in tech open right yeah um and you see I had a particular problem in there and I had to adapt the hyperparameter normalization normalization not sorry the second parameter search to to some sort of QQ plots that because I wanted the errors to the errors in the prediction to go through normal distribution. so you say LSTM sometimes they have a lot of work to improve it and maybe they generate this stuff but if you have time explore as I said you need Very weird explorations you need to work around a bit the time windows check why the lag is happening for instance here the Gru I would expect worse results because there's no output gate right so if the LSTM pretty well they should underperform {yeah} right. maybe that's because the data is easy enough right for for any neural network to learn but it's still underperforming. I don't know if you you have the same inclusion or not but yeah and (20:30)

**Researcher**: I'll just quickly go through the other ones Anonymous just while I thought there and so attention mechanism was the next one LSTM attention mechanism and this time I built it slightly differently defined the build RNN function and and then the first LSTM layer would drop out then I put in a loop for additional layers and dropout functions on our dropouts and I put in the attention mechanism here and here and then I and the rest of it is fairly um similar nothing nothing different there and so just that I think that the attention mechanism and the results then are down here it's saying format again so red line this time the predictions again and it's so it's kind of not quite as close as the ones previously yeah the

**Subject**: the interest that you would expect something better right here but {yeah} but it's worse yes I'm trying this for a univariate 19 it's not attention is meant to do some sort of detailed instruction right and to get the most important features you are in a univariate setting so your features are the time steps themselves so if you are putting attention on something it's in these 20 dimension part right and at the same time that is messing with the nature of the reality and itself {OK yeah} because you are now giving data that it's not really so since you did the attention that the structure the data you're passing to it is not sequential it's a some extracted some file from the sequence right so the LSTM is getting insane that's why but it would be I would recommend eventually you have multivariate time series {yeah yeah} then it makes more sense so that's I would expect something like this yes

**Researcher**: OK yeah but that's that's it I think that's fair uh Anonymous and I'll just go through those last ones here so forget well there's three more actually forget gate bias is the next one and so I just included I forget gate bias and so I set the bias to one initially but then I included it in the randomized search CV to kind of look for a few different across a few different biases bias values what would work best so that's just where I set that here and then the results of this one are here so we're kind of similar accuracy let's say similar to the attention mechanism you know a bit a bit off as well and there is a bit of a lag there as well

**Subject**: At the end of the day this is just a matter of weighting and serialization (23:38) so {yeah} this is the same maybe if you give it more epochs or something like that or more data it will work well about initialization of with some biases it's just at the end of the day you make things converge quicker or or in a more it's all slower right. So I wouldn't put much thought into this it's just maybe you could have initialization initialize sorry initialized sorry Spanish all the way to ½ (24:15) right so much talk (24:16 onwards)

**Researcher**: yeah perfect and then second last one so you'll laugh when you see this one I think so this is combining attention mechanism and forget gate bias and so again just build when a building model just included both of those forget gate biases there and the attention mechanism is is there as well I'll just Scroll down through result results and there's the result

**Subject**: it combines the the full stuff right {yes yes} puttig a forget gate initialization will be slower to come back yeah and then the attention as I said since it's an univariate of the regression we'll mess a lot with the sequences there is getting (25:14), so I would expect that. Because sometimes vanilla is the best.

**Researcher**: yeah yeah very true. this is the final one so bidirectional one this time um so again just like I used the class from tensorflow directional in the in each of the layers and then the result is here. So this one is kind of back back towards the performance of maybe the 1st 2 and and then yeah again there's a might be a bit of a lag there as well see you there.

**Subject**: LSTM is many people (25:55), theoreticians of this etcetera recommend using them but nobody gives you a good reason behind it right. {yeah} so you know how it works right and it says passes sequence one way and the other and it tries to find some time in variance and if there's time in variance it will work well. Sometimes it works really well with LMB (26:21) because when LMB when there was no attention mechanisms and Transformers etcetera the sequences right the relationship between words needed to go from your from left to right and then from back to the beginning. Again for this kind of data again, I say I would say if you put more epochs in there and more data it may work. If not, the time variance I don’t see it that relevant that should be right because they're the same way it goes up it should go down one way or the other but I guess it's making more difficult to converge sometimes. {yeah} Many people are in theory and I mean developing theory on these things won't give you one reason why to use isolate some matter of try it and what I believe is what I said to you it has a good reason to exist. Here with univariate my guess is that it's more difficult to make it converge once it converges maybe it is not talking about our fitting of course but maybe it has a better generalization. Sometimes things finally good right(27:38) but generalize better in the long {yeah} this happens a lot so

**Researcher**: yeah I think that's fair and Anonymous and then the last thing I wanted to show you so they're they're all the variants so I'll just make I might need to re-share my screen let's see patient ohh I've been having some some issues with the browser because it's not being responsive. But let's see if I can even copy these into the chat it's completely froze So what I what I wanted to show you anyway was that I kind of put the in the table the results and basically what the the highest performing and lowest performing variants where and so I'll just call them out here just so you can hear at least here the the results so basically the first 3 the top performing ones were firstly the Gru n ^2 number of 0.100 and vanilla LSTM was second with 0.104 by directional was third with one point or 0.107 and then there's a bit of a gap and then the forget gate bias and then the the attention mechanism the two separate they're both on 0.153 or 157 and then the one at the very end is the forget gate bias and the tension mechanism combination that one was way worse at 0.337 so I hope that makes sense without the visual and and I was just wondering if if anything occurs to you just from here and those results and do they surprise you or would you be would you have expected those results based off you know just what we went through there and your observations so far

**Subject**: Yes, that what I expected from the notebook. But I recommend you uh is to do uh baseline data normal autoregression right so you get uh give me your number that's made from right uh so you have 10 layers only and you cannot dropout etc of course that's good (30:15) but recording yeah yeah reset your label you start with the base line (30:16 onwards).

This is what auto regression gives us, now let's explore what neural networks (30:35) can do these things {yeah} so help will be will give you competitive in your research when you have to present it to your team {yeah} and as I said I I don't as I said it's not criticism or anything I would study just uh top is the same placement if there's no replacement for these cases right {yeah} so people people will come those questions right um and this way you will be anle to answer everything because you explore every angle everything

**Researcher**: You can see the results there now yeah ohh yeah sorry I just wanted to give you that visual so these were these were what I was talking through there just the variance and they're being squared error scores so anyway that's sure that's fine yeah just wanted to show you that because I finally got unlocked here unfrozen and Anonymous anyway thanks to me and if you've covered in which they're on TV for gone into detail on each and each variant what you said makes sense to me and I'll definitely look into those recommendations you gave and you've you've probably captured the exact questions that my lecture will ask as well so like thanks for for that insight as well that's great I don't know anything last you want to add to that or you've covered everything there do you think uh

**Subject**: I think so it's a small what can I say best of luck to you you must have crossword and if you need anything you want to do it last four months right ask me something right just feel free correct it's no problem thanks for that question

## 8.9 Interview F Transcript

**Researcher**: Perfect so can you see the screen there

**Subject:** yes I can yeah I can

**Researcher**: great thanks so I'll just go straight to where I built the models it is because the data preprocessing and all that you know it take a while to go through that and so the first variant of the LSTM I did was just the was the vanilla one just the the straightforward 1 and and how I build that is basically umm here so I I defined a function to build the LSTM for the RNN sorry and then I used the LSTM class in the layers to make it an LSTM and I also created a loop here to add in more layers and again with the LSTM class and the reason that I've I've done that is because I wanted to experiment with different numbers of layers to see which you know number performed the best and we go through more of that in a little bit and then create another LSTM layer here and then the final LSTM layer there at the bottom and I hard coded the loss function to mean squared error because just as part of my project mean squared error is the the metric I'm using to compare each models so I just made sure that that's always mean squared error and for each one. and then what I've done here is I wanted to wrap this model into this model so that I could do some cross validation to try and try out different hyperparameters and to see which hyperparameter tunings work best so that's where you know trying the number of layers the number the different optimizers so Adam and RMS prop are are explored. you know the number of epochs the batch size the dropout rate and then also the number of nodes in each layer so I just always used randomized search CV to kind of automate that process just to save time because I felt I didn't have enough time to just manually change hyperparameters as I went on just cause of the training and all of that takes a while. and then instead of and and

**Subject:** it's good that you basically put it all in one place all the hyperparameters in one place so that's good

**Researcher**: yeah yeah that's great [yeah just go ahead sure] and so anyway yeah this is all I kind of used random search randomized search CV here anyway and then I'll just skip through that and just go around to the results so I predicted So what I was predicting was kind of like the last 20 days in the in the series of 10 years and so that's basically what was being predicted. and the results are here for this model so the red line is the predictions and then the black line is the real data. so and yeah this this is how the first one performed anyway and I did calculate a mean squared error for this one as well. and each time the code is run it it calculates a different mean squared error just because I didn't set a random seed. but I've recorded separately some results of like kind of like the first time I ran it and just to have have a comparison between all the different models so I can show you that table that I have of the results in a little bit. Anyway this is the first one and so the next one I went through is a GRU and which is a more simplified structure and this is where I built it so I just used the GRU class and in the layers to make to turn it into a Gru layer and then I I built in very much the same way as the previous one with the loop there for the different numbers of layers in my hard coded the loss function as mean squared error. and then and then I did all the same moves as the last variant as well in the hyperparameters wanted them all to be consistent so that it'd be a fair comparison and then the results down here are so these are the results here so again the red line is the predictions and the black line is the real data so you can see how that performs there and then that one OK in this particular running of the code this one didn't do as well and put in the results I have separately you'll you'll see kind of how they compared in that instance.

**Subject:**  so on a map is did it perform less than the 111 (4:37)

**Researcher**: and in this in this particular

**Subject:**  so when you calculated the mean squared errors, so did it perform less than the vanilla LSTM method

**Researcher**: and so in in from what you can see here it did perform not as good and it performed less good but umm in the table of results I have it actually performed better and so it's just because you know when I run the code at different times a different result comes up each time so that's just why there's such a discrepancy between what what it says here and what my table of results have so I'll show you those kind of results at the end so you can see that the comparison that I got there. But again yeah and so that's the group the Gru. And the next one is an LSTM written attention mechanism so that's where attention mechanism is where it can focus on important inputs of the important parts of the input data and to try and kind of find more important areas to focus on when making predictions so structure is a bit different for this one. but All in all I still just created the LSTM layers and what I added in was the attention mechanism here and this is to capture the important parts of the input sequence and I have the attention mechanism then in the in the LSTM layer there. And just calculated the usual from there with the mean squared error as the loss function and then the results are down here sorry so the results are here so again red line is the predictions and the black line is the real data. And so this one yeah it was it was kind of in between in terms of performance it was somewhere in between the two the two you saw before that or you know it was somewhere in the middle basically overall that one at the end again just to where it sits in the ranking of performance. The next one is LSTM with forget gate bias so this is where chooses what to forget and not to forget. So the forget gate bias I've set the value to 1 here but I've also included it in the randomized search CV so that randomized search CV could explore different values for the forget gate bias to see which one works the best. And then here again I've created the LSTM layers but this time I've included the the bias I've initialized the bias of the forget biases in the layer as well and I've included it in the other layers as well below that and then until the final one there which is the output layer so it's not included in the in the output layer final one. And then then in the hyperparameter search space for randomized search CV I've included the forget bias values as something that can be searched by randomized search CV so from 0.5 to 2.0 they're the values that it will search point to see which one works best. And then the results are here so again the red line is the predictions on the black line is the real data so that's forget gate bias LSTM. Then there's just two more so this one is a combination of the two before that so the attention mechanism and forget gate bias are both included in this one. So how it was built was here so this was the most complex one built. So the forget gate bias again was set the value was set to 1 initially but then it would be experimented with so and I forget gate bias is included in first LSTM layer and the other ones as well after that open up until the last one. And then the attention mechanism is also included there and you can see that in this layer and then yeah so then the the model is compiled and then the results are just below so these are the results for this one so the predictions are in red and then the real data is in black so you can see that the predictions are kind of very very very straight line this time not really following the the real data

**Subject:**  yeah it does not work {yeah yeah} (09:18 onwards)—*I didn’t understand anything from the poor audio* -\_-

**Researcher**: yeah that's it and then there's just one last one so this is the bidirectional LSTM so yeah this is the one here where it propagates both forward and backwards and to try and capture more context for the LSTM and so this is where it's built so it's it's back to it's it's more like the first couple of models we built and this time I've used the bidirectional class from tensor flow so it was it was more straightforward to include and I put the bidirectional class in each of the layers except the last one. and and then the result is here so this is the result so there's the red line there for the predictions and the black line there for the real data and the mean squared error calculated there. so I know I might just go to the table now and show you the the results I got the first time just because they're the ones that have compared to 1/7 going to kind of go with because it's different each time and

**Subject:**  yeah just one question there {sure} so you know the components here that you're doing between the predicted values and the actual values so like is this the entire data like so like is it the subset is from the data from which the model were trained or is it like a new set of data that the model is is looking at and then predicting.

**Researcher**: yeah that's right the second one yeah so it's it's test data that the model is predicting that it shouldn't have not seen before up until now on the training. [OK] so it's yeah it's the kind of test results and then but yeah yeah [OK so I just I think I might need to re share the screen because just I want to share a different window I'm not sure I how I shared it the first time so I'll try and share the whole screen if I can oh it's not letting me do that OK I'll just go with this window here you can just let me know if you can see that window appearing now

**Subject:**  yeah yeah I can see now

**Researcher**: perfect yes that's it perfect. So in this table here you'll see these two columns these are basically the results that I got the first time I ran all of the code and it's in order of what what performed the best at that time and what performed the worst at the bottom so the GRU performed best in this instance with 0.100 as the mean squared error and the vanilla LSTM was second best with 0.104, bidirectional was then next with 0.107 they're they're the three best ones that's fine and then there's kind of two in the middle here which are forget gate bias and the attention mechanism they both were 0.15 in their mean squared error. And then and last was the by a good bit was the forget gate bias on the attention mechanism combined they were a good bit worse than their performance with 0.33. And then I've just included in the rest of the table I just included the hyper parameters that were considered the best by the randomized search CV for each variant as well just just as a bit of info. But the main data, the main important data there is just the results really and the ranking of the the different variants. And so I suppose just off of that I'm wondering is is there any observations you have so far based on that based on either how the models were done or how the the results have turned out is there anything you observe there or anything that surprises you or that is consistent with what you would expect from that so far.

**Subject:**  yeah look I think the results that you have got is in line with what is expected from time series data. I don't know maybe the forget gate bias could happen better but it did not. It’s reflected in the results and you have already turned randomized with CV for you know the hyperparameters so the results are right there in the front. The other thing in LSTM it's it's important how you input the data so have you created lags before you before you like in the preprocessing of the data did you create lags

**Researcher**: Oh lags, sorry so yeah I included so basically I had time step of 20 and then the the test data itself was also 20 in length so does does that answer your question for lags so the time steps there was 20 times steps there.

**Subject:** So yeah it's like you can experiment with that number as well. Sometimes that that can also create an impact {yeah yeah that's an interesting point yeah}. You can experiment with the number of time steps as well. Because otherwise everything looks perfect to me it's just that 50 number of time steps sometimes if you take different time step it it it gives better results.

**Researcher**: yeah yeah that's a very fairpoint that's something I can look into alright for sure and

**Subject:** I think it's very good work. Honestly like it's it's like I was looking at the code and it seems very well written and it seems like a good work. See as I said like I I like in my experience I I have tried the one in love (15:15) and bidirectional one and forget gate but for me it was a classification model and it was not very complicated terms like so basically the vanilla one worked best and then wanted to complicate it more if it was working well enough. So yeah I think I think you're all good like just try it out some more so you're doing this is your thesis… You’re doing it from UCC?

**Researcher**: yeah yeah that's right yeah so it's the college is actually I'm not sure you would have heard of it it's called CCT college Dublin. It's a it's a it's like a private computing College in Dublin city center that have a a program sponsored by the government where if you want to upskill in certain courses the government will sponsor the program for you and the CCT Dublin was a college that was part of this program so that's where I heard of them and then yeah entered the course then and so but yeah I wouldn't have heard of them before myself but they're they are actually in the center of Dublin there and and yeah it's a good course anyway I'd recommend it and just sorry just another one last question on the material there in terms of how the variance compared to one another is there anything that you can think of excuse me as to why the the more simpler models the ones at the top are performing better than the others so the GRU, the vanilla, and the bidirectional LSTM and is there anything in your mind that does that surprise you or or is there some or would you have expected that the simpler models would perform better the way they have their for example the Gru at the top that would have been probably the most simple model and it's performing better than the others is there any thoughts you have on that

**Subject:** Yeah so basically it depends more on the on the kind of data that you have so sometimes you know it's your data already has very first (17:23) trends you don't really need too much complicated models. Because like in my experience when I was like I worked for cast members and I did something simpler. So we were predicting which will be the highest peak value of the customer start could reach in the next 20 years so the next five years on the basis of the historical data from past 20 years so you're looking at the data from past 20 years. And we were trying to predict peak gas demand which will which will happen in the next 5 years. And for that we used ordinary squares to traditional analysis (18:00).{yeah} It's not even machine learning so as if all like it all depends on the kind of data that you have. It's it's all about the trends that you have in data which then impacts the algorithms you know the different different algorithms that we use so for some data. Honestly it's like into statistics that we'll be able to then find out exactly yeah yeah it was it was great to talk to you it was pretty insightful yeah

## 8.10 Interview G Transcript

**Researcher**: OK yeah perfect sorry you continue now

**Subject:** yeah so you have a definitely I'll I'll I'll give my input regarding to this but before I'll start I would like to know more about you. So what's your background and how did you ended up you know selecting the LSTM's and the the the course that you are in right now. So what what uh what intrigues you uh into the field of data science not the data science it's machine learning and artificial intelligence.

**Researcher**: yeah and yes so my background is that I'm a proposition manager in the telecoms industry so my role is more kind of commercial in nature and marketing but and my kind of the direction I'm looking to go in is towards product management in the kind of tech industry so um I just felt that um one gap I had in my um that he was maybe more technical in nature and I was very interested in the developments in data analytics, artificial intelligence mainly just so I would could become better at finding insights in data so that's kind of what drove me to do the course. The course that I'm doing is excellent and like it covers it starts off it it's mainly focused on data science. So it it it starts off with the kind of basic more basic machine learning models and and then as it goes on it becomes more advanced than it comes into the deep learning and the LSTMs and all of that. The reason I've centered in on LTMS and and and and those that area is because I was interested in regression problems that might be applicable in I suppose in business situations when you're predicting number sales or stock levels or you know it doesn't have to be um limited to the business domain it could be anything but that was just an area that I wanted to at least explore first and I might broaden out as time goes on but for the moment that's where I'd like to kind of get an understanding in and focus first before I branch out. I may look in time out there areas but for now I want to I want to start with regression

**Subject:** yeah that was that was really I mean great explanation so I would also like to know is it your individual project or is it a group sort of thing

**Researcher**: yeah so it's an individual project yeah so it's uh it's like a 12 week project that started in July and it finishes at the end of September so we've to do the coding and then we've to like write a thesis and you know we'll do some primary research and then kind of produce the results and comment on the results and all of that stuff

**Part 2 recording**

**Researcher**: I will think about that and like if you're interested in where I got the data I can like share a link with you of where I got it was publicly available and all of that so yeah no problem at all can share that and then so anyway I I don't know

**Subject:** Also what’s with the file name? It's it says variance 17 so did you try like 17 different files and 70 different versions with their schemes. I know in reality it's it looks simple it looks easy to to understand not to understand but to see uh I mean data science and deep learning from far away. But if you are really good in data and you need to do the permutation and combination definitely requires more effort and I would say it's time complexity or so yeah I mean did you did you try it with the 17 different variants of the devices

**Researcher**: yeah yeah to be honest…that's a good question, to be honest I just I was just tracking my work on GitHub so I just was saving it under a different name each time so I could show the different version so I just got to number 17 really. So I mean I did try you know experimented and the different models but umm and what worked and what wasn't working and so we're just yeah got up to 17 but like if you looked at it in my GitHub that like if I had them all on GitHub you would look like the first few were just it's just like me starting the the document and things like that so there's not a lot in the first few versions it's just as it goes along that it becomes more complete and with the work

**Subject:** Definitely Patrick do share me your Github so I can look into that.

**Researcher**: yeah yeah absolutely I'll do that no problem at all

**Subject:** man crazy, your codes are literally superb, very well executed in a very well sequenced sequential order it it was very easy for me to understand like where the LSTMS and in the end you wrap up with the the attention mechanism and some I think you also included some power forward gates also.

**Researcher**: yeah the bidirectional one is the last one yeah

**Subject:** Did you try working with the vanilla, I mean artificial neural networks and you can I mean, it's one suggestion small suggestion for me like {yeah}. Explore the differences between the artificial neural network and the LSTM because at the end you can you can also you know explain to your mentors that it's also about the sometimes it's not about the accuracy it's also about how quickly the model gives you the output it's about the speed of the processing so it will I mean if God knows that the kidney or you're getting a good not a very good amount of accuracy or the loss percentage then it can be a good algorithm it could be a good choice of algorithm because anyhow the differences between the processing of LSTM and artificial neural network there would be the differences yeah so you can easily show them about I think also. And and I think you have also implemented hyperparameter tunings. {yeah} To get the best chip parameters and the combinations

**Researcher**: yeah that's it so I used the randomized search CV to kind of automate the hyperparameter tuning because one of my research objectives was trying to find the best hyperparameters for each LSTM variant. And I found that due to time constraints me manually tuning and retraining models will take a long time. So I just thought that using randomized search CV, I could automate it and they could do the calculate the experimentation for me and come out with the best values for the hyperparameters. The reason I chose randomized search CV instead of grid search, is just because again time constraints. I just wanted to get an overview of what the best parameters might be instead of like grid search CV might have been more thorough in exploring more options. But again just I was taking time into consideration so that's why I just went with randomized search CV. I thought it might be sufficient enough to give them some good hyper parameters for each one.

**Subject:** But there's one disadvantage with randomized search CV. I hope you know about that. So maybe maybe they can ask about how relevant or how confident you are on the outputs or the results that you are getting from randomized CV. Because it randomly picks the parameters and whatever parameters are. You will never know what accuracy that those parameters can offer you. {good point} I would also suggest that I did some research regarding uh it's uh why I didn't know much about it but I'm currently studying what what exactly is by you know optimization and it can be used as an alternative to like grid search CV and randomized search CV to find out the best parameters. So again it it it's a light additions from me that if you can also dive into the bayesian optimization and I know you have got very less time it's because you need to prepare with your presentations and your thesis. But just give it a give it time. {yeah} I would suggest just one variant of artificial neural network would be enough you know (6:04). And the bayesian optimization it's just I mean optional. Because whatever I mean you did a great job with randomized search CV. But in in the in the end they will they will they may asked about the confidence level of the the the uh the answers that you're getting or the the figures the numbers you are getting uh from the randomized search CV. How confident you are. {OK} Because just in case just in case.

**Researcher**: yeah yeah no that's a very good point I'll take that into consideration and that's exactly the type of feedback that I need as well just to find where potential gaps are considerations for future research so we'll absolutely take that on board and this is just one other thing Anonymous uh well not just one other thing but then. I have a table here of results that might look a bit different to what's in the Jupiter notebook because uh because I didn't set a random seed so the results are different every time you run the models but I have so because I need to report the results I just took one particular um running of the code and have the results there if you want I can share them on the screen here just so you can see what they are just to just so I can show what I'm reporting as performing the best and the worst

**Subject:** yeah yeah sure definitely it's it's a really good practice that you made a table so it would be easy for everyone to compare the variance.

**Researcher**: can you see the screen there? so basically these are the results in kind of order of best performing towards performing in terms of mean squared error so and I've just recorded in the table as well what the best hyperparameters found by the randomized search CV was for each LSTM variant so basically actually I don't I won't comment on it yet maybe just ask for your thoughts on what you see there at the moment and that might be a better better use of the time to just hear your thoughts on it

**Subject:** Yeah man that's perfect. I mean it's uh visually very very intriguing to to see the comparisons between different LSTM variants. So it's so it's uh in your case it's does it emphasize only on LSTM algorithm. Or is it about what should we say, like comparative analysis of different algorithms..

**Researcher**: yeah and so so this particular one is focusing on LSTM variants so that's why I've kind of yeah narrowed it down to to those just so I could go into a bit of depth in in each one and so yeah that's just why it's yeah just down to LSTM variants in particular and so I thought so just looking at those and I don't know how familiar you'll be with each variant but I was wondering is there any thoughts you have on like why GRU is the best performing and Vanilla. I'm just wondering is there any thoughts you have on why they might why particularly the GRU might be the best performing while while some of the others are further down the list in performances. Is there any thoughts or like would you have any theories yourself that you would explore as to why that might be the case

**Subject:** In the deep learning algorithms are really I mean in the very first scenario, I mean, initially they are very difficult to interpret {yeah} how exactly they're working in back end (9:45). So might be GRUs and also forget gates (9:46) and the kind of architecture they have as compared to vanilla LSTMs. I would say if you can also work with more optimization techniques sometimes it's not about being adam optimization (10:00). Right now it's considered to be the best but yeah RMS are all crap(10:08) also does quite a similar thing so maybe you can also work with different dropout values you know, and different uh layers layers of neurons and the different numbers of and and different activation also.

**Researcher**: yeah that's a good point and uh because in the code I show I like yeah I set a certain uh search space let's see if I can get the code up

**Subject:** And also the permutations or the the if I have to drive deep into the comparisons between Gru and vanilla so higher personally and to you know execute the goals with my own understandings and with my own inputs of the data. So I mean if I'm looking in on the table and if I have to uh you know uh give out the um and the outputs are the comparison so I would say if you can work with more uh number of neurons and the layers and betrothals, percentages it's also about the optimization. So I mean everything needs to be changed and everything needs to be you know opting you can you can you can use the variation optimizations and search CV. I think there is also keras tuner. It's also kind of humanization (11:30) so you can uh it's it's it works more like a grid search CV yeah compared to um {yeah} yeah it's a Keras regressor. I I work with keras tuner. {OK} I don't know how exactly the Keras regressor in comparison to keras tuner. But it's think about I didn't working at the same scenario it's it's working the same way. {yeah yeah} So just looking for from the table that's I mean that's all the inputs I think that's all from my from my side that I can you know suggest you.

**Researcher**: yeah no no they're great suggestions Anonymous. thanks me for them and just one one observation I noticed from I don't know if this presented in that you would agree with is that and we can give your thoughts on it is that some of the models that performed better seemed to be the more simpler architectures GRU and vanilla LSTM will be will be more straightforward than say the attention mechanism or the the the one at the bottom there which is the combination of forget gate bias and attention mechanism that that performed by far the worst. And it was probably the most complex one put together. um so today I think I don't know what your thoughts are and that the the fact that simpler architectures seem to have performed better and I wonder if you would you if you were looking at the reasons as to why simpler ones perform better but where would you look yourself to try and analyze why that's the case.

**Subject:** It's I think it's also the data is also similarly responsible of the parameters that we are using. It's I think maybe umm I think you have created a plot it's regarding the prices with respect to different you know months and years and {yeah}. There’s hardly any changes there's there's there's hardly any changes like there's hardly any major fluctuations. It's it's working on a similar patterns like the way our real world scenario, real world data should look {yeah} that's it (yeah}. So yeah the data is also not that complex, it's very simple {yeah} there are some dates and units unit also great thing to extract the months and years and the weeks. I think weeks also your included like Monday Tuesdays and Wednesday. {Yeah} I’m not sure whether the GRU included that part in in the final processing but you know the preprocessing of data was also good.

**Researcher**: that's great thanks a million yeah and you're right about the the data that I used so so initially I saw if I just go up to here I saw that from the year say 2000 onwards the data becomes a lot more volatile. So what I did is I I just cut out that data and I just used for 1988 to 1998 so the data is a lot more stable and easier to predict so so yeah I think when you were talking about how the data might be responsible there for the results and like without stating the obvious yeah I think I'd agree with with that as well that there might be some compatibility between more stable data and maybe the simpler models performing better than the the more complex ones.

**Subject:** Yeah and also in a nutshell, the less calculations a model performs like it it tends to give better accuracy. {yeah yeah} The simpler ones are less architecture architecturally complex so I I also think so that's the major reason that that they are giving us the better accuracy. {yeah} Just for the reference you can also implement one I think machine learning algorithms just to understand how if if you are if you are if you are getting a simple you know accuracy you just do not include that into your thesis and the the the the the process and the task. just for the reference just for the just for you know just to see if the data is really that simple so how could it is performing with vanilla you know random forest or decision tree.

**Researcher**: yeah yeah that's a very good suggestion yeah absolutely

**Subject:** if I was to good accuracy or a very less loss with respect to machine learning algorithm that I mean we can totally say that the the data is very quite simple it's I mean it's also almost completely cleaned and processed preprocessed before you received it so I mean we can also get to know how again on how good the data is in the shape

**Researcher**: yeah yeah I think that's a that's very fair point I'd absolutely do that if outside the project like in the real world let's say I would start with the as it's recommended I'd start with a more simple model and it's only if I found that there was issues um cut like with the UM capturing complexities in the data that I would spread out to a more I would like advance to a more deep learning methodology to try and capture those more complex relationships in the data yeah

**Subject:** OK you can you can you can definitely go with it

**Researcher**: yeah yeah and I think that's most of my own prompted material but I don't know is there any other questions you have Anonymous or any more suggestions from from what you observed when you looked through the codes that you think would be interesting to try or just consider as a limitation or anything along those lines.

**Subject:** I don't have anything as of now I mean if you can give me the data whenever you are done with the thesis and maybe at the time of the execution that I'll do with the code of the changes that I do on the corner of my I can also come with more insights and observations .but as of now I'm finding your port is very clean it's very very well built and it's almost ready to be deployed. if if you if any of any one of us select are you planning to deploy or what's your I mean definitely they are going to the mentors are going to ask you about the future prospect of your date and the model so how were you going to answer that also they did ask you know at least two or three times I mean how are you going to I'm every time I have to give out the different different kind of advice and answers questions

**Researcher**: that's a good question so to be honest it's not something I thought about yet so that's something I'll need to think about ahead of the Viva that we we will have a fever on it so that's something I'll need to consider and like I'm open to any suggestions as well that you have on that but that's something I'll have to take back and research a bit myself about how best to deploy it like my only plan for at the moment is to submit it for the assignment and then like when the when the the course is over soon to like I'll have it on my public GitHub instead of my we have like a special college GitHub but I'll I'll put it onto my public GitHub then you know just so it's a available for for people to see so yeah in terms of deployment. I haven't about it yeah but it's that's a very good question and something I need to consider.

**Subject:** have you heard about Streamlight about which sorry bring it it's SDRAM LIT ohh Streamlight

**Researcher**: no it's on I don't think I've heard that yeah

**Subject**: I would say just go to some YouTube tutorials or stream date it's all it's all about the deployment and it's the easiest one {Yeah}. the damn easiest one uh deployment you know it's for me it's the very first step if anyone wants to learn deployment with respect to machine learning and deep learning so streamlight rate would be the easiest one because by that you can you can generate a UI like a web page for the users to provide the data and just click on the button and any predict button or any button of your time that it will in the back end it it will run the course for you and it will provide you the answer so stream rate is the easiest 1 and it turns out a URL like it runs it runs on your local system it provides you a web web URL you can just share it with your and and you can even share it with me at the time you you run the stream rate at your system and with that link I can easily access your website and I can easily get the predictions of whatever data you provide to the website so at the time of you know presentations you can also do the same thing just by providing a link to the to the mentors and ask them to to get some predictions of their with with their own data. add stream rate is the easiest and the is the is the easiest one right now.

**Researcher**: OK yeah that's that's great I'll uh I'll take note of that Anonymous extremely and I'll look into that and maybe explore that as the potential deployment approach if if I get that that question as well so thanks thanks for giving me that steer in that direction

## 8.11 Interview H Transcript

**Interviewer:** OK great and I'll certainly come myself there and so yeah and so yeah just as a background to the session or the project I'm doing so it's a thesis it's look and taking 6 different variants of large long short term memory networks and it's doing some predictions on financial time series data and it basically what I'm trying to do is I'm trying to look at which one performs the best as measured by mean squared error and also like just looking at optimizing the hyperparameters so that they perform as well as they can and then looking at it like maybe some of the reasons as why some perform better than others particularly whatever one performs the best I want to like try and just analyze some factors as why it might have performed better than some of the other variants so that's kind of the background just because and then So what I've done is I've taken about 10 years of data and which are the data is like oil prices in Europe that um that are like Monday to Friday so just working you know business days and over those 10 years and then what I'm trying to do is I'm trying to so the years are 1988 to 1998 and then what I'm trying to do is I'm trying to predict the final 20 days of 1998 so I've kind of isolated those last 20 days from 1998 and we're trying to predict what those 20 values are going to be that's kind of my test kind of so to speak and I think that's yeah most most of it and yes so anyway I'll just share the screen if that's OK and just I'll just take you through the main parts of the code I won't go through all the preprocessing unless you're curious about something to do with that we can look at that but otherwise I'll just go straight to the models and the results and stuff and too

**Subject:** yeah I'm interested also like with the preprocessing if steps if if that's OK I think yeah so just just to understand so to make sure that I get this correctly so you have you have 6 different variants of the same more or less the same kind of general linear algorithm yeah you are just comparing the the telephone and the the prediction performance not not anything else between those using word squared rmac or MSE

**Interviewer:** MSE yeah yeah yeah

**Subject:** yeah OK that's it yeah that's it exactly yeah yeah and so for the data so you have all prices in Europe Monday to Friday of 10 years yeah you're trying to the final 20 days yeah any any before we can look at the results and just in terms of processing did you was there anything like significant that you did on the like anything that took some time or some code to actually do on the preprocessing and

**Interviewer:** sure I'll I'll just share the screen and just go maybe from the beginning just I'll just try and highlight the main bits without going through everything so initially I started off with umm with about 35 years worth of data and initially what I was just looking to do was check checking missing for missing values and all of that kind of thing So what I did is I isolated just trying to see exactly where affecting the thing So what I was trying to do is basically I isolated the number the months to see if there's any missing data over the months I isolated the weeks to see if there's any missing data over the weeks and um and

**Subject:** making sure it's consistent

**Interviewer:** yeah yeah just making sure that there's nothing missing from any of the the days weeks or months something yeah for years and anyway sorry I can confirm which it was tree variables I looked at it was year columns always play here uh by month and then by the day of the week ohh yeah so here month month and day of the week I just wanted to see that there was equal amount of values and for anywhere where where it was slightly inconsistent they could be attributed to things like you know public holidays in Europe and things like that it was as it was Europe prices and what else so So what I did for

**Subject:** So how how big is the data I guess the first question it was good enough like because I'm guessing all prices you should be clean enough in general

**Interviewer:** yeah so you would think like they were fairly clean but what what I did notice here is actually that there was number of values that OK they're not null or nan but they're they're just carrot special characters there's no numeric value there so that that was the problem so I did identify a number of those and so I had to just decide how to what to do with those whether to delete them or do imputation and So what I did was in the end I decided to do imputation and I used forward fill and backfill to fill in those values so there was about so there was about 259 values that were instead of a numeric value it was just this special character or something that was filling in the space so I had to do something about them so I in the end I did forward fill and backfill as well I'll just find that section here it is there yeah forward filling back feel like it there and the reason I did that was just because I didn't want to do the mean like just replacing them with the mean because it's time series data and from what I know just that's not the most effective approach because it's sequential etcetera it just the mean doesn't really take account of the sequential in nature

**Subject:** yeah for sure if you have it at the start of the analysis I'm guessing it's probably skewing it because with inflation and everything it's probably going to be way higher than the rest so no yeah I think that's that's how I would kind of do it's fairly simple yeah OK and I'm guessing 250 values is very low in terms of the overall total isn't it

**Interviewer:** yeah so it's it's probably in in the end so with the data that I ended up using in the end I would say that's probably under 10% anyway so I started off with data set of like 9000 but it actually reduced it to about 2000 and I could take you through that in a second so suppose 259 I would say it's still below 10% of the data and so it is somewhat now February that that we're used imputation was used to fill in but yeah we're still below 10% I would say So what I noticed with the data we had initially over the 35 years was that it was highly skewed to the left and I noticed as well you can see there it's skewed to the left and then here I took the the whole data set and just visualized it and I found that so these are the oil places over the years and they found that after the year 2000 lots of volatility came happened So what I did is I just took the data from 1988 to 1998 because the data is nice and stable fairly stable there most of the time compared to what comes after that just so it'll be easier to predict for the just for the purposes of the project because the data as it's stood was very left skewed and just made it more difficult and and when they did that when I when I isolated just those 10 years what happened to the data was it detained very gaussian and then her more gaussian and you'll see here so it was just better to work with with those 10 years of data that's why they did that and and then from there like that's really from there I just you know split the data up into training and testing and and all of that kind of thing so there is no more imputation or I suppose manipulations beyond just you know having data in this training and testing data and all of that stuff there

**Subject:** OK yeah I'm interested in how you kind of did this of training and testing so do you have like a train validation and test set or just to train and test and

**Interviewer:** so I have yes so I I did split and I I took the training data up until the last 20 values of um the the overall data and then I used the last 20 values as the kind of test set and that's kind of how how we did it and

**Subject:** so yeah because so to be honest I haven't done time series in a long while so I'm trying to remember like I'm trying to kind of compared to like compared to what I would do for regular models today that are usually not almost all of them are like classification problem so in classification regular classification problems most of them have detailed three data sets if you were going to do some hyperparameter tuning one being the training data set the validation data set is going to be the one that you're going to use and it's kind of like this the the the the one that you will or that you're going to use for tuning and then the test set kind of out of sample test set or try to not touch it at all and that's the that's kind of the the end result or what's the tuning is done you use the test set so I would say you could be correct to use the the latest data as test set to make sure umm I'm just wondering umm if there would be any use in time series to have a validation set that is at the end of the training set I I don't exactly know to be honest I think it's worth getting looking into yeah

**Interviewer:** so So what I did do with the data is so I have the training set and the testing set but what I did as well as I used randomized search CV for validation and I I built the models in a way and basically and there's a there's a bunch of hyperparameters that randomized search CV can search through and just find which ones are performing best and so so I do have that but apart from that yeah I just have the training set and then the test set

**Subject:** but I think that works then because I think randomized search cv does this cross validation thing so it essentially can put everything your train set is gonna be splitted in all kind of like and then there's this yeah because meditation so yeah OK that works

**Interviewer:** great thanks and then and then just finally I did do some normalization as well as that just because there was no gaussian I was able to use standard scaler just to normalize the data just to try and then you have to see if that would improve the performance as well so that was also done and certainly to build started like to go through just how I came about

**Subject:** no I got it so I haven't looked at the code to and so far there's no worries but I'm guessing it's I'm guessing it's working so my main kind of 2 the things that I've kind of conscious about voting in general or in that sense is the first thing you reproducibility so you want to but I I don't see why this would be the case here because otherwise you wouldn't have done complete the subject like you needed in order to compare different models and stuff so from top to bottom you should be able to kind of rerun everything and have everything kind of the same unless there's some very specific reason you could you could not like for instance automation and there's no seeding there's no possibility to kind of set the seed so that you can have exactly the same but yeah if I can like that's that would be the main thing about the with the code and then the other that is probably more fun stuff around how to kind of manage data science project with Python usually what I'd go for is exploratory analysis with the notebook but if I have to rerun some experiments and I know exactly like kind of the shape of the data like preprocessing the exploration is done I just want to do stuff usually I go to kind of more like Python scripts so that you can actually make it a bit more handy for reproducibility in that field but yeah that's it I mean if you check your code and then it doesn't look like it doesn't look crazy and then nothing jumps at me so they would take me a long time I think too cold step by step tell myself yeah if you're a video on that so

**Interviewer:** yeah that that's a good point you're raised about the random seed because I didn't do that and anyway I'll show you how things go but it's an interesting point you made there something I'll consider and so and if you want I can just go forward to the first model first variant yeah and then I'll just show you the kind of key maybe for the first one I might go into a bit more detail on what I did and then for this it's five others so for those I won't go into as much detail maybe unless they really like did something specific you want to analyse so sure vanilla LSTM was the first one so that's kind of the most standard one that people would use so I just import it all the libraries here that would be using and then to build I defined a build recurrent neural network function and here and the initialized sequential so anyway I used to make it LSTM I use the LSTM class I put that into the layers to make in LSTM network instead of just a regular RNN network and then because I'm using randomized search CV and I wanted to explore different parameter set hyperparameter settings I created the loop here and so this loop basically can if I want to experiment with how many layers I'm putting in this loop could kind of help with that and and it just adds in LSTM layers and I'll show you how I kind of do that in in a second which was for the mean time I just added in some more another LSTM layer and then the the output layer is just here and and then the loss function I've hard coded as mean squared error just because it might project mean squared error is the metric that I call out in the title of all that I'm using to measure each variant against each other so that's just why I have that hard coded and not going to experiment with other loss functions and to to use a randomized search CV I I wrapped this model and so I could use it for randomized search CV and then there's hyperparameters that I can you know that it can search to find the best combination and this is kind of the search space so for numbers of nodes in in a layer I have between 5:00 and 50 you can choose a random between those for dropout rate and it'll search between 0.1 and 0.5 for batch size it'll search between 1632 and 64 then epochs as well and then optimizer as well it'll search under Adam or RMS prop see which one works the best and then for numbers and letters that's the one where it can experiment with one layer or two layers and up to four anyway and so that's what that was so that's kind of what the what the method or what the idea was behind that loop and all of that and and then yeah from there really this is just to take account of the time series split to maintain the temporal nature of the data so that uh random parts aren't taken for the cross validation and that like the parts taken and still follow the sequence properties at a time series element isn't lost with the cross validation and this is kind of where I'm applying the random search CV or where I'm setting it up and and then I hit it here to the training and and then I was just going to skip down to the test results from there and so the results are here so the red line is the predictions and the black line is the real data of the of those final 20 days in the run so in the 10 year period. So and that's how uh vanilla lstm performed anyway so and then the mean squared error is calculated down here but as you said earlier I didn't set a random seed so each time this is ran it gives a different result and I actually have a table of results that I can show you at the end that kind of they're the kind of results that I'm running with at the moment and I'm not really paying attention to what these say like it's a good point though like maybe I should have random seed in there so that they are consistent and reproducible as you say if someone else were to

**Subject:** say like if you're storing the results in like a table and then you have a distribution of results that all have been randomized I mean random like set of parameters if you have the distribution of a few for each model then it's also good I mean it's it's also worth having right because then you can you can say ohh this research would be good and then it was like pure chance 9 of the other experiments showed that it wasn't the case I'm guessing that's that's that's all right yeah it does look not too bad to me one thing that I want so if I remember so I haven't done regression in a while but I remember the RMSE is more commonly used because you can essentially interpret it as the unit that you're using in your predictions for us it's like it's about you you can compare things but you can't really so it would be worse to kind of have the the the RMSE if you are planning to kind of give the error as to to interpret it as that you're trying to predict but now that I'm thinking about it and you said that you weren't standardizing and normalizing the data so I'm guessing by default to start it's not really can’t really interpret it as like dollars or something else

**Interviewer:** ohh yeah so sorry what I did do and I didn't mention this is I did uh inverse transform it before displaying the predictions so I did find you know denormalize it OK and before showing the results should you train so yeah the result that you do

**Subject:** OK so so yeah yeah then for the error like I think that RMS is does make sense yeah but then you could say ohh it's offline that much OK first I think they're having a look here but I might be wrong

**Interviewer:** yes perfect yeah no I'll do that and I'll do that thanks to me here and so I'll just move on quickly to the the other variants so I won't go through them in too much detail apart from this might be the highlights so this one is the Gru which is just like a more simplified structure where two of the gates are kind of combined together so I've built this in very much the same way as the last model except I have the Gru Gru class in there for each layer and then everything else is pretty much the same and then the results are here and here so again the red line is the prediction and then the black line is the real data that it's that's the difference so that's the result for that one and I'll just go through the out some of the others so this one is with an attention mechanism and so let's put that in attention mechanism sorry this is just an LSTM LSTM with an attention mechanism other than so attention mechanisms just where it kind of focuses on and it looks at the inputs and then tries to focus on what's important for those you know from those inputs and points that focus on the important stuff from the inputs basically right and that's that's the idea right

**Subject:** So like big variations stuff like that maybe and well so I am not well versed at all with LSTM I mean I remember what it does in the general sense but I don't know that much I just know that it's kind of a it's kind of a neural network that does this long term and short term kind of process where there is like you can update some weights based long term weights based on the short term results and it resets if there's some like change in distribution or something like this this this mechanism of because it's very it's yeah it's it's it's not something I'm using everyday

**Interviewer:** yeah no that's that's fine yeah but like from what it is just as an overview is like it what it does is for recurrent neural networks they're very good dealing with sequential data in the short term but then what the problem with them is that over the long term they they tend to kind of forget what came earlier and there's something called the vanishing gradient problem where the kind of the the effects of the the kind of earlier um nodes or input is kind of forgotten about later on so at the for the LSTM is just a mechanism to kind of help it with the long term memory side of things as well as the short term memory side of things that's kind of what the the general gist of the LTMS are there

**Subject:** rings the bell yeah also the exploding I think there's also an issue with it could be vanishing or exploding

**Interviewer:** yeah yeah that's it exactly yeah so it's it's open to deal with that and so this is just another variant of LSTM here at this time I've put in and I built it a little differently but I've still some of the principles are the same there is a first LSTM layer is here what I've added in for this particular one is I've added in the attention mechanism so this is where it's to capture important parts of the input sequence so that's what's kind of special about this one but again just the rest they're just you know unless the layers but with the attention mechanism included and then then the results for this one are here so the red is the prediction black is the real data so that's how and then the results that's shown on this particular yeah no not not as great yeah and that's true and then next one is the forget gate bias so this is where there's a bias there but it's choosing what to forget and what not to forget so it's the value that it can assign so I've assigned it as 1 at the beginning but what they can do and what I did do is I used randomized search CV to also search for the forget bias value that would work best perform best so that's actually another searchable parameter that I've included in the parameter search space below that you'll you'll see in a second so anyway I've initialized the forget bias here and then I've included it just in in each of the layers along the way the forget bias is in there it's in this final LSTM layer as well and then the output layer it's not included just because that's the output layer and and then in the search hyper parameter search space included the forget bias down here that it can search values between 0.5 to 2 just to see which forget bias value works best and then what is called the results so there's the results um red again as the prediction and black is the real data and then you can see the score is kind of similar to the attention mechanism roughly in a in a similar vein and there's just two more left and just to check Anonymous you tight for time?

**Subject:** uh no no I'm good actually so you can I we have 15 minutes more

**Interviewer:** ohh yeah ohh yeah that's more than enough time yeah sorry I just I had some others in that 30 minutes so I just wanted to check and so there's two more to go through so the last this second last one is that this is a combination of the last two so it's attention mechanism and forget gate bias together together in the same model so this was probably the most complex 1 to put together but yeah but anyway it was just a combination of the forget bias and the attention mechanism they're they're both included in it basically that's just all it is very good and then in this layer here yeah forget bias and attention are built in that layer as well and then anyway the result of this one was forget bias is still a parameter that can be searched for by randomized search CV as well and then the results are so so these are the results so you can see how that one went by far the kind of well yeah yeah it just predicted the same thing almost for the for all of them so it wasn't yeah that's that one and then the final one is a bidirectional LSTM so just where there's both forward and back kind of uh propagation to learn more context around the sequence of data so this one was easier to build and because there's a bidirectional class that I could use from a library from tensor flow and I just added it added that bidirectional class into each layer and they're in there and here also and then the final layer there's no none included in the output layer then have again went through the same process randomized search CV all of that and then then the results are down here so this one was a bit better again and so red is the predictions and black is the real data so that's at this moment anyway so that's the the the 6th 1:00 and if you want I can just show you the table I have over results or if you want we can just focus on the code it's up to yourself if you'd like to see that I could show you the table that just with the results that I have recorded and you can see kind of which ones performed well which ones didn't and that kind of thing? OK I'm just going to share again because I think I have to sharing different window and then OK umm can you see that OK so so this is just the table of the results that I I've recorded when like one at a times I ran the code all the way through from start to finish and so it's a bit different to what you saw there but again the random seed not being set might be there you know that could be something I could look into to get more consistency reproducibility and so this is going from best to worst from the top so the best here was the Gru with 0.100 as the mean squared error and the vanilla LSTM was the next best with 0.104 then the bidirectional 1 was the next best with 0.107 and then the two in the middle here where the forget gate bias and the attention mechanism at about 0.15 and then finally the the worst one was the forget bias forget gate bias and the attention mechanism combined that was 0.33 so that was by far the worst one and then also on this table life just included you know which hyper which parameters what were the optimal hyperparameters found by randomized search CV based on the search space for each each of them as well so that's what I've included as well there so I just want to ask is there anything that stands out to you from these results or would they be as you'd expect or is there any observations you have about this or anything you'd like to point out

**Subject:** uh yeah sure so I think so there was a spoke to me it's it's I can't reach it so not being like on the subject but about the kind of experiment I think it'd be it'd be worth it to kind of so I'm just so first question how long does it take to kind of run for one model for instance how long did it take you or how how how much money does it cost um to have one training from end to end for a model yeah good good question it it it did take a while but it wasn't it wasn't like it was done within it within a day like and it wasn't like 8 hours or anything like that like it might have been uh might have been one hour or something that I remember exactly each time I trained the model each individual model might have been about 15 minutes or something and something like that so that's not that's not too long that's why that's what I'm guessing because there's only it's it's there's not a huge amount of data so even though this is deep learning you don't need a huge machine to train your model so so you could like I think it'd be worth it to actually try to we run this experiment and store more of these MSE results so that you have a distribution for each algorithm so say if you could if you can if you have the time like maybe you you you just plan it so that it kind of and that's where kind of Python scripting comes in handy is that you can actually get out get off the notebook and have this kind of scripting that will run and that will store the results in in say CSV or something incrementally and so then at the end you have this distribution for each model then you can look at some kind of a significance of results on to compare each other because for instance here GRU and vanilla very close as well as the bidirectional it could be due to some of the process in the randomized search that it stumbles onto a set of parameters that works and then you know it it kind of shows that it's actually better so I would be curious to kind of if if possible if it's it's not a huge load on the on any kind of processor or something I don't know if that runs on your own laptop or if you have a machine somewhere that you can have it kind of run through but it be worth it to kind of make it and have some more results then you can you can guess so that that would be the one of the first thing that I would think about that way you can have some some kind of an idea of what what is the the confidence intervals or the the statistical significance between those results and then my second like thing that I would kind of be interested in is OK so the goal of this was to kind of compare models for time series and I so you you took ten years of this data to kind of train these models um but in like from a global point of view I would be interested to understand all these models so for instance the best model is it better because the data is more simple it's simpler it's you know like the vanilla are simpler architecture does it work because time series problem is is simpler or when you kind of rerun this experiment with more like changing data with trends like strong trends or like some some unexpected changes do the forget bias or attention mechanism one kind of tend to work best on those you know and then have this this ability to kind of adopt better in certain situations that would be kind of an interesting piece and so so it's kind of like as a it could be something that you could say kind of is it would be a next step in your study kind of looking into goes and systematic kind of review on different space of the data or like different timelines I don't know yeah so the things

**Interviewer:** yeah yeah absolutely and I think that makes sense Anonymous and and in terms of optimizing the hyperparameters of bundles do you think there's anything I could look at you may have mentioned touched on it already think there's anything I can look at there to to improve that side of it um how how I've done the models from is there like I don't know if you have any suggestions there

**Subject:** um so to be honest I feel like it's the kind of thing where you can spend an infinite amount of time on yeah so like it's it's it's a yeah you can't really do perfect so I think it's as long as you make sure that there's no like bias in your the way that you're running these experiments so I would have a think of maybe kind of like some of the models you have one more parameter than the search grid would that affect but the overall performance of the model I mean the overall performance of the grid search yes no I'm not sure that it's like because in the end you want you want to have results yeah I mean so so yeah I would be kind of considering carefully that I'm that there's no like bias between one model to the other but from what it looks like you've done it quite well in terms of how you've you've set up those and again I am not using LSTMs or tensor flow that much so I can't really that nothing jumps out at me because just because I don't use those libraries that much but in an overall kind of looking at the study and the things that yeah those two things which are which are like okay rerun the experiments again and again and again and try to see OK does it changes a lot which would raise the level if if you have a completely different ranking at the next iteration or something that's probably like ohh there's there's something happening something is wrong or like there's some bias term somewhere for sure but yeah would help in kind of getting the confidence and then I would be just be curious in general as to if we apply those models architectures on other parts of the data where the distribution is less is more skewed as it handles that does it suddenly changes and one model becomes better than the others etcetera so so yeah that's where like I'd say that's where the the good there's a bit of like some work to kind of pull the the logic that from your notebook and kind of put it in some kind of scripts but I don't think it's it doesn't look to me that your work is it would be hard to kind of put into scripts so that you can rerun and have those experiments on different spaces of the data and stuff yeah I think it would be something interesting another way yeah it's great have a good job I think I think you did a good job

**Interviewer:** ohh yeah thanks thanks thanks man yeah I'll just stop sharing there yeah yeah no thanks man and I think they're all very good points especially from an experimentation point of view and validate and the experimentation and I think they're they're all very good points actually and they have made and what else the random seed is something I should take into account and yeah there's there's definitely a lot for me to think about so really appreciate your thoughts Anonymous I think that's probably everything I wanted to cover and you know we've kind of gone around the parameter optimization about you've commented on the best variance why they might have why might have performed better or they might have performed better and yeah I think that's pretty much everything on that side so just uh thanks again for giving your thoughts I don't know if there's any last thing you wanted to add or you've covered it all

**Subject:** no I don't know did you have fun And I was it like a huge pain to go through like the tensor flow library I remember doing tensorflow very low level and I was OK it was hard so

**Interviewer:** yeah yeah I did I did find it difficult as well yeah it wasn't it wasn't an easy task for me but I I figured that out just because at this stage I'm at that everything is difficult so so I just kind of went with what I could and tried I was just happy that that it worked in the end that it actually ran that was sort of the main thing I was happy with but uh right yeah but like I'd love to like do it again under less pressure you know outside of a college situation just in my own time be able to try things and all of that that would be something interesting maybe in a month's time when I'm finished I'll try that

**Subject:** yeah sure yeah yeah I mean if the data I mean if the studio and the data does seem to be like public data so you can just always like upload it on the GitHub yeah or cargo and just getting tackle it do you know do you know Cagle yeah yeah familiar with it yeah I forget community and that yeah they have like some some handy tools to kind of upload study then you can get even I mean maybe you can get when people get to read your stuff and can you say point to someone's part of the code and say ohh yeah you could have done that or something yeah can be interesting yeah anyway yeah so yeah thanks for showing me this good job

## 8.12 Interview I Transcript

**Interviewer:** OK perfect so I'll share the screen you can just let me know when that's appearing sorry yeah but just 'cause there's you know we take forever to get you I just want to focus on the highlights so that that's appearing OK is it

**Subject:** yes just before we get into the LSTM as we sell TSM itself he mentioned kind of some of the data turning up he did yeah kind of a forwarder backfill yeah any reason you didn't just average on either side or

**Interviewer:** yeah that's a good question and if you want we can go through some of those steps so the reason I didn't do the average was just because because it's time series data and just from from what I've I've heard and read that sometimes the average can kind of skew you know if if it's sequential data that's you know trending up or down you know the average might in kind of respect that kind of time series sequential nature of it so I just thought with a forward fill or a back fill at least the kind of missing values that's been filled in will be kind of closely related to the one that either came or after it and so that's kind of just that was kind of the thinking anyway on my side

**Subject:** yeah yeah and I definitely agree with I do on you wouldn't want to take like a long average but how did you decide when to forward or backfill and if this suggestion made around average would just need to take the two on either side of it and fill with the average 'cause that would help your have a look at it yourself right how many minutes that is you have but if you're say OK say your average say your score one day or your price one day is five and then it goes up to 10 yeah and then it comes back down and afterwards or it continues to go up if you backfill into 5510 there's a huge jump if you're back to 5/10/10 you've a huge jump if you were to average the five and the 10 and you're coming in seven half which is like being smooth or smoother for for doing your analysis on does that make sense

**Interviewer:** yeah yeah that's a fairpoint to something I didn't think of yet the reason I chose So what I wanted to choose was forward fill and yeah simply because I suppose it's an assumption that yeah if the data is going in One Direction that I will continue to which I know has its has its own limitations and then I just backfilled the the very first value in the sequence you know doesn't have a value to forward fill it with so that's just why I use backfill for the remaining for the first value really so it's forward filled what I used

**Subject:** for the majority in one situation where it was sorry you need to use the back hey Cortana he is one of them in one situation where there is no first value to fill it from yeah and the rest of the most consistently in the other program yeah yeah

**Interviewer:** yeah yeah that's it and yeah so that was one of the main highlights of the pro preprocessing and would you like to if you want I can take you through some of that too or if we can go straight to the models

**Subject:** I think that's that's probably the that's probably the biggest thing is you see how you cope with missing values from that perspective and I think I think both are absolutely valid ways to do it yeah I was just curious more than anything else

**Interviewer:** yeah yeah but that's a fairpoint so I didn't have thought of that before so point and anyway I'll just start with the first of the six so the first one was the vanilla LSTM which is just kind of like close to the original kind of most standard 1 so I'll just go through the code that I used to kind of build it without going through every line just cause of the time and So what I did here is I defined a function called build RNN and to kind of set up the initial parameters of the model and then I included this LSTM class here to make it an LSTM as opposed to a recurrent network and the input shape I used is 20 time steps and one feature so to speak and then I wanted to kind of automate the hyperparameter tuning if I could So what I did was I'm using randomized search cv to that and one of the parameters I wanted to experiment with different values was the number of layers in the model so that's why I have this loop set up here which will basically create different numbers of layers depending on the the iteration I suppose so that's really what that is there and then also there's just a third yeah

**Subject:** 20 days part of your shame so logic yeah yeah why 20 days

**Interviewer:** oh sorry yeah yeah again it's a good question so to be honest I took I took 20 as like a standard number I didn't I didn't go through a process to decide why 20 would be the right amount so yeah there there was no there is no process there really for that one it was kind of just a standard number that I just chose

**Subject:** I think it seems logical based on the fact that we're doing it per day and they're kind of like the time this thing takes change in in a oil price space like I think it suits her the for the space

**Interviewer:** yeah yeah fair question and and then finally that's the output layer there so and in the loss for the loss function I just hard coded it as mean squared error because in my thesis title and all that the whole evaluation is done through mean squared error so I wanted that to be consistent throughout and then for to do the randomized search cv I wrapped the model and using the keras regressor and then I wanted it to search different hyperparameter values and the values I wanted it to search are here so to take the number of nodes in each layer and the rate and the batch size the number of epochs the optimizer will either be adam or rmsprop and then the number of layers as you said earlier somewhere between 1:00 and 1:00 and 4:00 it would search between see which ones perform best and then this this part here is just about doing the cross validation but using time series split so that so that the kind of sequential nature of the data would be maintained even if the data is split up into batches as part of the cross validation that's what that is then that's just implementing the randomized search CV or sorry this is fitting the search and then I'll just skip down to the results of the testing so this is the results of the first one so the red line is the predictions and the black line is the real data so it's you can see just for there anyway how it's performing and the mean squared error I have down here it's actually because I didn't set a random seed the mean squared error is kind of different here to what I've recorded from the first time I run it but what I'll do is I'll just at the end I'll show you the table I have the full set of results from the first time I run it just to just so you have a consistent look and comparison between the different different models so that's the first one anyway so I just wanted to go through a bit of detail on the first one and then for the other ones hopefully I'll skip over a bit more but is that all OK so far got on y

**Subject:** our first first run or was it I

**Interviewer:** think on the first one I think I got a point .10 so it's it's you know it's not a million miles away.11 point 10 great thanks so next one is Gru so just a bit of a simpler structure this one where the gates are combined so again if built it in a similar way to the last one except I have the Gru class in there for each layer to to make that kind of combined structure that gru has for the the gates and yet build the layers in the same way with the group class and everything else is pretty much the same means word error is used again as the loss function and the model is wrapped there the hyperparameters until that randomized search CV will look through right here and all of the same as before that's the training as you can see and then the results are down here so again red is the prediction line and then the real data is in black so that one on that particular run was seems to be a lot worse actually than the first one but again I'll show you the results at the end because it is quite different to what I have recorded from from the previous previous run of the code OK next one is with an attention mechanism and so the attention mechanism is to try and focus on identifying important parts of the it is to capture some of the important parts of the input sequence so this one was built a bit differently but still with the LSTM layers and with a with a loop in there the attention mechanism was added in here and then you can see it in the subsequent layers just added in at the end and yeah that's that's the main difference there and

**Subject:** what was the logic of using the attention mechanism what did you think it would draw attention to or what was the what was the hypothesis for using that

**Interviewer:** so I suppose the attention mechanism 11 reason is it was supported with a library which was helpful for implementing but in other words that it's it's something that has come up in the literature and just that it's a choice that some practitioners might be interested in and I wasn't sure whether it would perform better or worse but just wanted to include in the comparison anyway just to be able to have a record of of its comparative performance so it's not that I expected it to be better or worse but just just thought it was a common enough one to to be explored so yeah anyway it's ran it's com it's compiled same mean squared error and no difference in the hyper parameters being searched I wanted those to be as consistent as possible and then the results are here so the red line is predictions black line is the real data and according to this particular role it's saying 0.15 is the mean squared error for that one then the next one is the with the forget gate bias so this is to decide what to forget and not to forget in the long term memory so for this particular one the the bias value is set first I said at 1:00 but I also wanted the bias to be a value that's searched in the hyperparameter tuning too so I've included that in the hyperparameter tuning as well so it's just initially 1 but then it can search other values at the different times it's around so the bias initializer the forget forget bias is being set in there and then it's the subsequent layers as well have the the bias as well in there and the output layer is just there and then mean squared error as usual and then in the hyperparameter space I've added in the forget bias here as a hyperparameter to search between 0.5 and 2 so we'll just search between those the the one that performs the best then I'll go to the results sorry sorry this is the results are here so red predictions again and the black is the the real data and that one is 0.15 S similar enough to the one just before that the attention mechanism just before that

**Subject:** very good seems to be the lag of mine for their yes to run a little bit with us that day but I think it's forgotten trying to forget the dip that's not maybe working out very well for that's yeah interesting yeah and then you put the two of them together then did you pay attention on the forget that's it can you hear me OK Patrick

**Interviewer:** Sure I can hear you fairly well yeah I think once you speak for a little bit it kind of becomes clearer but then it's just like here and there and it's a bit but I can let you know if there's any issues on the on the you know recording and don't worry I'll be giving you more of a chance in a in a minute to give your give your take on it

**Subject:** speak speak for longer introduce every sentence but yeah let me know if you if not catch me yeah

**Interviewer:** yeah that sounds great thanks and so this is the second last one I believe so this was kind of the most complex 1 so yeah as you said it's a combination of the attention mechanism and the forget gate bias so I bias in here again it's well one initially and I've included the bias here I have a loop again for the different layers the attention mechanism appears in here and then the forget bias and the tension mechanism kind of appear in the next layer after the loop is done yeah that's that's really how that one was built and again it's wrapped so it randomized search CV could be explored and forget bias again was included in the hyperparameters to search for and then the result is here so the red is the predictions and the black is the real data so and the result for that one was 0.33 so that one was significantly worse than than the the ones that came before it

**Subject:** and you saw on the previous two that we've combined them together that they've just yeah trying too hard to to consistent I think there to to cope with this type of data other data

**Interviewer:** yeah yeah that's it yeah that's too consistent definitely good right point alright and yeah that's the second last one so we'll just go through the last one and then I can go to the results and we can talk through each one so umm bidirectional LSTM was the last last one where kind of process it kind of propagates both forward and backwards to capture the more context for the sequential data so this one was easier to implement than some of the others there was a bidirectional class that I could take from tensor flow that I just put in each layer which which I've done there and then there's the output layer at the bottom and everything else is pretty much the same as it was for the other models and then results are there so red is the prediction black is the real data so I mean that's a bit better than and the last one anyway sorry the scores there 0.10 for that one so that one performed was one of the better performing ones in this particular run as well

**Subject:** Explain the by directional piece to me there sorry

**Interviewer:** so normally so normally so it's looking basically it's looking backwards and forwards it's back it's propagating both backwards and forwards and I suppose the idea is that instead of just capturing context from one side it's capturing context from both sides to try and basically give it a better understanding of the context of the data so it's it's going that's why it's bidirectional it's kind of propagating both ways if that makes sense

**Subject:** yeah it's just yeah it's it's interesting because what you're seeing is it's catching on to some of the other ones wouldn't shift direction when there was a change in trend or it took longer to shift direction with it’s a change trend whereas that one if you look at your results there at your your graph you can see it's quicker to catch on to a change in direction in some of those uplifts and some of the other ones for ourselves yeah that's interesting well that's vision there and particularly when it's saying consistently in the same direction and closes the thing to us yeah

**Interviewer:** yeah yeah you can trust yeah and anyway I'm just going to declare I'm going to re share just because I want to share the results and I think I need to re share for the other windows just to show you what I kind of summary results started appearing

**Subject:** OK so we want to walk through so cheer you oh sorry yeah

**Interviewer:** yeah you can see sorry yeah so so just going from sorry yes so these two columns are the main ones really to look at and so I've just I put them in order of which ones performed the best in the kind of first full time that I ran through all the code at once so the Gru one there ran the best 0.100 and the vanilla LSTM was second and 0.104 bidirectional 1 was third in one 0.107 and then kind of then there's a bit of a gap in the middle then the forget gate bias and attention mechanism were fairly close both were zero point 5 so 1 was 153 the other was 157 and then there's a big gap then to the one at the bottom which was the combination of forget gate bias and attention mechanism that was 0.337 so anyway just taking all that in just wondering if you have any thoughts in general Anonymous either on either on how things could be made better I know you've given a couple of ideas already and then also maybe on the results here is there anything that surprised you or anything you'd like to comment on about it and did it did it meet your expectations or or was it surprising or anything at all in that in that sense

**Subject:** so I think The performance of the forget gate and the attention individually as a result do not surprise me that the that the combination has performed so badly I think it's due to the nature of the data that you have but they're not performing as well and yeah something in non financial data might cope with this a little bit better and but it's they're just not bringing anything to the equation in this particular case and which is fine I would say your top three there really are close enough that there's not a there's not a best out of those right there's the best yeah but I'd say if you run that 10 times those top three might might move around a little bit maybe let's say bidirectional but it makes sense to me that they're that they're fitting well because they're good fit for it I think you've brought in enough enough variable components to be able to that that you're optimizing at a good level so I think this all makes sense as well and I think your challenge would be conclusion concluding your end kind of the best pieces or how do you maybe look at one of the three the only thing that I think would be missing for me is so LTSM is obviously what you've chosen to look at here but how does that compare to something more basic like an arima time series what is the advantages obviously you have a much bigger computational loads I can't think of the proper word of doing an LTSM and watch something much more basic are you getting a much better result or are you getting a similar result than you would with a very Standard Time series

**Interview:** yeah yeah absolutely

**Subject:** just might add to your results to say DLTSM generally performs better than sorry just not and Benjamin is generally performing much better I would assume that an arima model right but we don't like assumptions and then you have obviously your results kind of within that so yeah nothing here is is majorly surprising and it all seems quite well put together well thought out and it's interesting that the forget bias and the attention mechanism just didn't didn't bode well for this particular data set and I would make sure for kind of your end your thesis interviews or whatever do you have to do thesis interview within college?

**Interviewer:** yeah we have Aviva just in about a month time

**Subject:** yeah yeah so as far as if I was preparing for that what I would say to you is do a little bit of research around why you think that for that big bias and attention methods didn't work well for this data set I think it's the nature of being a financial data set and and then maybe looking at just do you have one more comparison that's non LSTM so I think it's helpful once you're in the LTSMI think you've done it really through a job but why an LTSM would be my question to you OK right after right which is making sure that yeah so if it was me I'd maybe look at something quite basic nothing too heavy lifting from an implementation wise but something like a ARIMA took approximately first to see what kind of your difference is that does that make sense yeah yeah no absolutely and but I set it up and I think this looks great and I think you've done a very good job push out more seconds yeah and all

**Interviewer:** thanks to me and Anonymous you've you've covered everything there anyway that i was going to ask so and yeah thanks million for your all your thoughts there and might just pause the recording there