Department of Computer Science

Submitted in part fulfilment for the degree of BEng.

Forecasting Macroeconomic Parameters With Deep Learning Neural Networks

Seif Hussein

1st May 2024

Supervisor: Dr. Poonam Yadav

Contents

List of Figures

List of Tables

Executive Summary

Macroeconomic forecasting has been an area of active research throughout history because it can aid policymakers and businesses in knowing the future state of an economy or a country. Macroeconomic indicators contain inflation, GDP, interest rates, producer price index, unemployment, and many other important factors that help in defining the economy. Indicators such as inflation and GDP are considered lagging indicators as they change following economic shifts, on the other hand, the producer price index is regarded as a leading indicator as it changes before the happening of economic shifts. Having an accurate prediction for macroeconomic indicators can act as a mirror for the future state of the economy, which in return will help people when planning for the future and making strategic decisions.

In recent years, there has been a growing interest in using deep neural networks for macroeconomic forecasting due to their ability to capture complex and non-linear relationships found in most time series data. This allows them to offer superior predictive abilities compared to traditional methods such as auto-regressive integrated moving average (ARIMA) and equation discovery.

Neural networks have hyperparameters that are not learned during training, therefore, they need to be tuned manually as they play a huge role in the performance of the models. Long-short term memory is a special type of neural network that is particularly suitable for modelling time series data that have long term dependencies.

This report aims to build a deep neural network model that can accurately predict the consumer price index when trained and tested on both the US economy dataset and the UK economy dataset. There are three objectives for this paper. The first one is to investigate if having a leading indicator as one of the features will enhance the prediction performance. Secondly, the ability of the introduced models to capture unexpected economic shocks such as Covid 19 is tested. Lastly, the capacity of the hyperparameter tuning algorithm to find a set of hyperparameters that lead to the optimal performance of a model when trained and tested on both economies is assessed.

Before building the model, time series data of needed indicators were collected and preprocessed, after that, the LSTM architecture was designed. A hyperparameter tuning algorithm was run to get the optimal set of hyperparameters for every model. Models for both economies were trained using their training data, then tested on their respective test data, and lastly, the results were analysed and evaluated. Models are evaluated and compared based on their MSE and RMSE values.

Results show that adding a leading indicator to be used as a feature didn't benefit the prediction performance of either the US or UK models and only added noise that hindered the accuracy of the models, particularly the UK model.

It was seen by the results that both our US and UK models were able to capture and predict accurately the values of CPI on their testing data. However, only the UK model was able to forecast the decline that happened because of Covid 19 in time. It was concluded that using only error metrics wasn't sufficient to judge a model's performance, as sometimes a human eye or an expert opinion is needed to make a fair judgement.

It was found by the results that the hyperparameter tuning algorithm was effective in finding a set of hyperparameters that resulted in models having the best prediction accuracy on both the US and UK datasets when using this set.

This work is intended to contribute to academic literature and assist future studies on the topic of macroeconomic forecasting. If this work is ever used outside of academia by entities or businesses, it should be used at their own risk. The data used for this work is obtained from publicly available data sources, and all libraries and modules used in this work are public open source properties. There are no ethical issues identified in this study.

1 Introduction

Throughout history, the ability to forecast macroeconomic parameters has played an essential role in strategic planning, policy formulation, and decision making. Macroeconomic parameter forecasting is the prediction of future trends and values of essential macroeconomic indicators that are used to predict the future state of a country's economy. The two main types of indicators are leading and lagging indicators. Leading indicators change before economic events. Lagging indicators change after economic events.

Traditional forecasting techniques such as linear regression and Auto regressive Integrated Moving Average (ARIMA) have a long history of forecasting macroeconomic parameters, yet they have limitations when it comes to capturing non-linear relationships. Alternatively, neural networks have the ability to express non-linear relationships in data and handle non-stationary data, which puts them at an advantage over traditional forecasting techniques.

This project aims to explore the intersection of neural networks with macroeconomic forecasting by implementing a set of Long Short Term Memory models using the Keras software library and TensorFlow to be used to forecast the consumer price index for the US and UK. In order to achieve a dependable prediction, the framework uses hyperparameter tuning algorithms that help identify the optimal hyperparameters. This study aims to build deep neural network models that can capture complicated patterns and dependencies within our macroeconomic data sets.

This work seeks to answer the following questions: Does using lagging indicators with a leading indicator benefit the prediction performance? To what extent do the forecasting models show adaptability and flexibility in handling unexpected economic events and shocks and giving an accurate prediction on test data? Can a specific set of hyperparameters be identified to optimise the performance of the neural network model across both the UK and US economies? The research pursues to present insights into the adaptability of models to economic shocks, if optimal hyperparameters can be identified that can be used across different economies, and if adding a leading indicator as a covariate will enhance the prediction accuracy on the test data. The outcomes of this research will contribute to both academic literature and practical use by policy makers and businesses.

2 Background and Literature Review

This chapter is divided into two parts: background and related work. The background section will identify key concepts, terminology, and foundations for understanding the landscape of this research. The related work section will provide information on previous research, reports, and articles that tackled the same or similar problems as our work and gaps found in literature.

2.1 Background

In the background section, we will first provide some context on macroeconomics, types of indicators, and inflation (CPI). After that, an introduction to macroeconomic forecasting is given, and then we will dive into time series models and machine learning. Deep learning will be discussed after this, with its different types of architectures. Finally, we will touch upon the process of hyperparameter tuning and its different algorithms.

2.1.1 Macroeconomics

Like many matters in life, economics can also be studied at different levels. Economics is typically divided into two sub-fields: microeconomics and macroeconomics. Microeconomics focuses on the behaviour and decisions of individual economic agents [16]. In macroeconomics, the economy is studied as a whole, with a focus on the structure, behaviour, and performance of the economy [18]. Macroeconomics analyses the overall performance of an economy using macroeconomic indicators and government policies. Overall economy performance is crucial for both governments, who need to make decisions and plan for the future, and individuals, as the economy directly impacts their well-being and the success of businesses [18]. Therefore, the study of macroeconomics is essential [17].

Leading factors	Coincident factors	Lagging factors
Producer price index	Personal income	Gross domestic product
Manufacturing activity	Industrial Production	Inflation(CPI)
Building permits		Unemployment
Level of business start-ups		Interest rates

Table 2.1: types of indicators [19], [23], [50]

Types Of Indicators

Giving attention to indicators is essential, as they help reveal the economy direction [19]. An indicator is a variable that can be used to forecast another variable [20]. Indicators can also be also informative in understanding and forecasting financial and economic trends [22]. There are three types of indicators. Leading indicators usually change before the beginning of larger economic changes in any direction [25], so they may be used to forecast shifts in an economy before it starts to change, they include producer price index, manufacturing activity, building permits, and level of new business startups. Coincident indicators are the indicators that change at the same time as the economy changes and can be seen as a mirror of the state of the current economy [22]. They are not very helpful in forecasting the future of the economy, but can give beneficial insights into the economy's current state [23], personal income and industrial production are examples of coincidental indicators. Lagging indicators are indicators that reflect economic shifts after happening, they can be used to confirm economic patterns and trends [21]. Lagging indicators only change after events take place, but that does not make them useless, as they can be used to confirm trends that are happening over time [22]. Lagging indicators include gross domestic product, inflation, unemployment, and interest rates. By using all three types, we can know more about the current state of the economy and where it is heading [22].

Inflation (CPI)

The consumer price index (CPI) is an indicator used to give consideration to the increase in the cost of living. Inflation measures the increase in the cost of the basket of total goods over some time. CPI is an index number indicator; index numbers are crucial performance indicators that can help in assessing economies as they provide a foundation for performance and comparison analysis [25]. Inflation is a lagging indicator, as described above, that only starts changing after an economic change takes place. Both inflation and deflation can be measured using CPI; a positive change

in the price of goods indicates inflation, while a negative change indicates deflation [25]. CPI as many indicators have benchmarks that are sometimes put up by governments, benchmarks are used to indicate if the value of CPI is acceptable or poor [24]. The following equation is used to calculate the annual CPI [26].

$$\text{Annual CPI} = \frac{ValueOfBasketInCurrentYear}{ValueOfBasketInPastYear} * 100 \quad \textbf{(2.1)}$$

The inflation rate can be calculated using this following equation [13].

$$\text{Inflation Rate} = \frac{CPIofCurrentYear}{CPIofPastYear} \quad \textbf{(2.2)}$$

2.1.2 Macroeconomic Forecasting

Macroeconomic forecasting is a way of estimating financial trends and indicators using historical data and projecting trends found to predict the financial status of an economy or business [45]. Macroeconomic forecasting can be divided into two types: qualitative and quantitative. Qualitative methods don't rely on numerical data but use a judgement based approach that relies on expert opinions to try to catch non-quantifiable variables and factors that the economy is influenced by but cannot be represented numerically. Quantitative methods make use of numerical data and statistical techniques with the use of trends and patterns identified in historical data and statistical models to predict the financial future of an economy.

Macroeconomic forecasting is one of the hardest forecasting types as it usually relies on the use of non-stationary financial time series data, which is famous for the level of noise it has. Also, not always history repeats itself; unexpected economic recessions and shocks took place, which a lot of macroeconomic forecasting models failed to accurately predict them, which led to several people believing in the efficient market hypothesis (EMH) [11], which assures the challenge of constantly achieving abnormal results just by analysing historical indicators or publicly available information, as most relevant information is reflected through the market itself. Despite theories like this, macroeconomic forecasting is still a very active research area that is accompanied by good results.

2.1.3 Time Series

A time series is a set of observations collected sequentially and maintained over time intervals [14]. It represents the evolution of a value or multiple values over time, giving an excellent chance to analyse the behaviour of out

of sample data [14]. Time series display temporal dependence as the values of variables at one time point are correlated with the values of previous time points [4]. Time series always contain variation in the observations, which is displayed by the changes in the values of observations through time points [15]. Variation can be decomposed into four components: seasonal variation, trend, cyclic variation, and residual [7]. Seasonal variation is exhibited by the regular changes and patterns that are repeated at fixed time intervals such as oscillations happening at a fixed quarter or month every year. The trend component of variation is usually shown by the steady change of the values of observations over at least multiple consecutive time periods [14]. Cyclic variation refers to patterns in observations that are exhibited over cycles of time other than the annual cycle. These cycles have more length than patterns displayed by seasonal variation, but without a known predetermined duration [14]. Irregular variations are the result of unexpected and unpredictable events and influences such as natural disasters or wars, as there is no specific pattern that is followed leading to these events [12]. There are various time series models, which are mainly statistical models, that are used in literature to forecast macroeconomic parameters.

Auto regressive Integrated Moving Average

ARMA is one of the main statistical models used in macroeconomic forecasting [12], as it is highly effective in forecasting univariate time series. It is the fusion of two models: AR and MA. The AR component uses past values of a time series variable to forecast future values, assuming that the current value is linearly dependent on its historical values [4]. MA which is the moving average component incorporates past residuals and noise into the model, because it assumes that the current value is impacted by past noise. To be used with non stationary datasets, ARIMA is used. The integrated component involves differencing the time series to remove trend to make the time series stationary. The definition of ARIMA model is *ARIMA (p, d, q)*, where *p* is the order of autoregressive component, *d* is the degree of differencing, which is the number of times differencing is applied to turn the data to stationary, and *q* is the moving average component order [12]. To directly model seasonal patterns in a time series, SARIMA models can be used. SARIMA is an extension of ARIMA and is defined as *SARIMA (p, d, q) (P, D, Q,s)* where the *p, d*, and *q* are the same as in ARIMA, and the seasonal part can be modelled as a separate model where *P* is the order of seasonal autoregressive component, *D* is the degree of seasonal differencing, *Q* is defined as the order of seasonal moving average component, and *s* is the seasonal period.

VARMA

VARMA, which is an extension of ARIMA, is used to model multivariate time series. VARMA is a merger between Vector Autoregressive (VAR) and Vector Moving Average (VMA) [12]. It can be defined as *VARMA (p, q)* where *p* is the order of the autoregressive component, and *q* is the order of the moving average component. VARMA assumes stationarity but it can be extended to VARIMA in the same sense that ARMA can be extended to ARIMA by using differencing to achieve stationarity. Using the fusion of VAR and VMA, it exhibits the shared changes of time series variables to be able to show the relationship between them and the impact that they have on each other [12].

2.1.4 Neural networks

A neural network is a computational model that is influenced by the human brain structure, which is formed of a huge number of interconnected neurons that are the basic building unit of the brain, communicating with each other through signals and exchanging data [2]. This inspired the creation of neural networks by replacing brain neurons with nodes that are grouped in layers. Nodes transmit information and data between each other, and data is learned by the process of adjusting the weights. The weights define the influence that the data transmitted in the connection has on the receiving neuron [5]. The simplest form of neural networks consists of two layers; input and output layers. However, a hidden layer can be added between these two layers to assist in identifying complex patterns and improve the forecasting process [10]. Unlike other traditional forecasting models, neural networks are able to capture and understand non-linear patterns and relationships [7].

Deep Neural Networks

Deep neural network is a neural network that has more than one hidden layer between the input layer and output layer. The number of hidden layers usually spans from two to eight [3]. Deep learning neural networks are typically used to forecast complex nonlinear relationships that are usually present in a time series because of their number of hidden layers [8]. Hidden layers apply the activation function to the calculated weighted sum of their inputs, and the result is sent to be the input of the next layer [2]. Weights can be learned by many algorithms, the most popular way is by using back-propagation algorithm.

Feed Forward Neural Networks (FNN)

In feed forward models, information only flows in one direction, which is forward from the input layer into hidden layers and finally into the output layer. FFN is made of several nodes that are put into layers. The first layer is the input layer, which receives the initial inputs [3]. Each node in the input layer portrays a feature of the input data. The second layer corresponds to the hidden states of the model, it can be any number. The term "hidden" refers to the fact that these layers are not directly observable in the input or output [27]. Hidden layers are used to capture complex, non linear patterns in the input data and learn from them. The last layer is the output layer, which is used to deliver the prediction. The main idea of FFN is that there can't be any cycles or loops in a directed graph, only forward flow [6]. Each neuron only communicates with neurons in the previous layers that supply it with the inputs, and the output is only outputted to the next layer, so there is no feedback between the layers [12]. FNN is considered by many to be the basic example of a neural network and is used a lot as a baseline in the process of comparing more complicated neural networks [3].

Long Short Term Memory (LSTM)

LSTM is a type of Recurrent Neural Networks (RNN). As the name suggests, feedback connections between layers are present in RNNs, unlike FNNs. RNN neurons depend on two things: the current input of time *t* that is passed to them as well as the state of the network at time *t-1* [6]. This mechanism is effective in storing memories of the previous time, allowing for a broader perspective on the variable to have more accurate predictions. It also prevents over-fitting. RNNs suffer from two big issues: vanishing gradients and exploding gradients. This happens in the process of updating the weights of the network, as the repeated multiplication in back-propagation can make the gradient prone to vanish towards 0 or to explode uncontrollably [13].

To deal with the problems stated above, Hochreiter and Schmidhuber introduced LSTM in 1997 [28]. LSTM is a type of RNN designed to overcome the problem of long-term dependency that RNNs suffer from [11]. Instead of using basic nodes, LSTM employs LSTM cells that have an internal memory [8]. LSTM memory cell is a composite unit made up of an input node that takes input from the current time step as well as the output of the hidden layer at the previous time step using the recurrent communication edges [6]. There are three gates in the LSTM cell: The input gate, which governs the passing of the input to the cell and controls how much of the input value from the input neuron is accepted [8], it is called a gate because

if the value outputted from the input gate is zero, then no information from the input is contributing to the cell, while output of one means that all input information is taking part in the prediction [6]. The forget gate determines which part of the LSTM cell is not essential to contribute to the output and drops it [9]. By doing this, it manages the weight of the state unit [11], and the output gate decides which information in the cell state should be contained in the output of the final prediction of the cell in this current time step [9]. The vanishing and exploding gradient problem is addressed by the regular error carousel performed by the gates, allowing the gradient to backpropagate through time steps without vanishing or exploding [6], [16]. Compared to other traditional neural networks, LSTM is better equipped to understand and describe more complex trends exhibited in the time series, especially when the time series exhibits strong trends [12].

2.1.5 Hyperparameter Tuning

Hyperparameters are settings of the neural network model that must be configured, they are not learned from the training data and updated like model parameters, but are set before the training process and not updated after that. Hyperparameters contain parameters such as the number of layers, number of nodes, number of epochs, and batch size [9]. Hyperparameters choice directly affects the model's accuracy and performance, making it essential to choose the right set of hyperparameters. Hyperparameters Tuning is the procedure of determining the ideal set of hyperparameters, which results in a neural network model having maximum performance [29]. In the past, choosing the hyperparameters depended a lot on the experience of the researcher. Now, some algorithms can carry out the procedure of choosing the optimal parameters.

Grid Search

Grid Search algorithm is used for hyperparameters tuning. It belongs to brute force algorithms, but its effectiveness is restricted by the combinations of hyperparameters [46], It achieves good results, but the drawback is that it is very resource intensive and computationally expensive.

Random Search

Random search is another algorithm used in hyperparameter tuning. Potential hyperparameters are randomly sampled from a predefined range of

values [46], then they are tested on validation set, and the best combination that achieves the best loss on the validation set is selected. It has been shown that random search performance exceeds grid search performance in various cases [4], and it's more efficient in terms of computation.

Genetic Algorithms

Genetic algorithm is an optimisation algorithm inspired by the process of natural selection and the principle of evolution in nature [30]. It involves six steps: population initialisation, fitness evaluation, termination condition check, random selection, crossover, and random mutation [11]. The drawback of genetic algorithms is that there are various hyperparameters other than the model's hyperparameters that need to be tuned to enhance performance, which increases the complexity.

2.2 Related Work

Macroeconomic forecasting has been an active area of research for years. There have been a number of studies that looked into using neural networks for macroeconomic forecasting that have been references to this research.

One of the studies that informed this work was conducted by Cook and Hall [1], where different architectures, including fully connected convolutional neural networks, LSTM, and encoder decoder networks, are analysed for forecasting unemployment rate. This study shows that the encoder decoder network achieved the best results out of all four. Gonzalez [2] uses a fully connected network to forecast the quarterly growth of Canada's real GDP with the help of early stopping procedure. The results show that the fully connected network outperformed the linear regression model in both in-sample and out of sample, but the statistical tests showed that the results are not statistically significant. Kokov's [4] study examines the difference between the results of ARIMA, Multilayer Perceptron, and LSTM in the task of forecasting macroeconomic parameters. Moreover, the effect of differencing the data is inspected to see if differencing enhances the performance of neural network models. Different hyperparameter tuning algorithms were tried to see which one yields the set of hyperparameters that result in the best performance. Results of Kokov's study suggest that deep neural network models outperform ARIMA, as both LSTM and MLP achieve better results in multi-step forecasting. Differencing the data resulted in better predictions in one-step forecasting, according to the results. It was shown that using random search resulted in smaller model errors, however, the

difference in performance wasn't statistically significant according to statistical tests. Another study which was lead by Skea [13], where GDP, interest rate, and inflation are forecast using the LSTM architecture. The study also investigates the benefit of having leading indicators as features. The results of Skea's study show that the lowest RMSE is scored by the uni-variate model in both one-step and multi-step prediction of all indicators except for inflation, in which the lowest RMSE was achieved using both leading and lagging factors as features in a multi-step model. In Griffiths's [47] work, a LSTM model is used to forecast US CPI. In the process of building the model, Griffith investigates macroeconomic parameters to identify the strongest determinants of US CPI to be used in the forecasting process. The results showed that using the determinants found, the model was able to score a very low MSE and RMSE on testing data and was able to predict US CPI accurately. One of the studies that benefited this paper was Paranhos's [3] paper, where LSTM architecture is used to forecast inflation. Three sets of features were created and compared. The results show that LSTM was able to score less errors compared to common benchmarks in medium-long horizon forecasting and was at least as good as feed-forward models in all horizons, which shows that LSTM is the most applicable model for predicting long term relationships and dependencies. Moreover, it was shown that using broad macroeconomic information results in better performance and prediction than using only information related to inflation in periods of high uncertainty.

Most of the existing research on macroeconomic forecasting uses shallow feed forward neural networks, which don't have enough hidden layers to capture complex non-linear relationships and patterns, which often lead to bad prediction performance [3], highlighting a gap in the literature regarding the use of deep neural networks in macroeconomic forecasting. Moreover, most studies on using leading factors as features are conducted on stock market data sets, resulting in a lack of studies on the effect of using leading indicators as features in predicting macroeconomic indicators. One of the few studies that examined the effect of having leading indicators as features for macroeconomic forecasting is Skea's [13] research, however, industrial production output was used as a leading indicator when in fact it is more of a coincident indicator. Furthermore, there have been a lack of research on the ability of the hyperparameter tuning algorithm to find a set of hyperparameters that can result in an optimal performance of a model on different datasets. These gaps in studies are addressed in our work by having deep neural network models that use data sets containing both leading and lagging features and comparing the prediction of the model when using both types of features. Hyperparameter tuning algorithm is used, and its ability to output an optimal set of hyperparameters that works best in both economies is also assessed.

3 Methodology

In the methodology section, the research questions of the paper are stated first. Following that, the research approach that was used in this work is illustrated. Then, the programming environment and programming languages utilised in this research are discussed. The process of data collection is then described, followed by the features selection section, which also contains how the data were manipulated and the results of the correlation analysis between CPI and the selected features. The data preprocessing techniques are then outlined. Finally, the structure of the LSTM used is described, including all of its components and the process of hyperparameter tuning through random search.

3.1 Research Questions

In this study, three questions have been investigated and answered. The first question is: does having both lagging and leading indicators as features enhance the model's performance? The second question is: can the introduced models adapt and forecast different economic events and shocks with acceptable accuracy? The third question is: is it possible to identify a specific set of hyperparameters that result in the most optimised performance of a neural network model when trained and tested on both US and UK economies?

3.2 Research Approach

The research methodology used for this project was the Agile methodology. Tasks were defined and broken down into small increments that were completed and delivered each sprint. At the end of each sprint, a self-review session was conducted to assess the work done in the sprint to be able to track the progress and identify areas for improvement or change. Every few sprints, a meeting was held with my supervisor to ensure that the work done in the previous sprints was going on the right path and to see if there was anything that needed to be adjusted or improved. This approach

introduced a lot of ways to adapt to changing and evolving requirements, increasing flexibility and adaptability.

3.3 Programming Language and Environment

Python was used as the primary language for this project due to its simplicity and readability. Additionally, it offers a huge collection of statistical and deep learning libraries, as well as powerful visualisation tools and libraries.

Several Python modules were used in this work. Pandas [40] was used for loading the dataset, transforming the data sets into DataFrames, and pre-processing of data. Handling data arrays and splitting the dataset was done using Numpy module [41]. Preprocessing steps such as normalisation of data and correlation tests were done using Scikit-learn [42] preprocessing module, SciPy [42], and StatsModels [44]. Models were implemented in Google Colab as it already had the necessary libraries pre-installed.

3.4 Data Collection

In the data collection process, there was a focus on only collecting data from reputable sources to have reliable datasets that could assist in the journey of forecasting CPI. Data for both US and UK economies were collected from online economic databases of Federal Reserve Economic Data (FRED) [31], which are maintained by the Federal Reserve Bank of St. Louis. This source was chosen because of its reliable data, customisable frequencies, and time spans that are updated regularly, making sure that the data used is up to date and accurate. The data was collected in the Excel (.xlsx) file format, making it easy to be coverted into a DataFrame. Data was collected at a quarterly frequency without being seasonally adjusted to preserve seasonal patterns.

3.5 Features Selection

Choosing the variables that were used by the model to forecast CPI for both the US and UK economies was a crucial step. Extensive research was carried out to ensure the appropriate features were chosen. Because of the first research question, we have to have both lagging and leading indicators. The procedure of choosing the features was based on three

things: literature review of past research on forecasting CPI, economic theory, and correlation tests between CPI [32], [33] and other features. GDP [34], [35], interest Rates [36], [37], and PPI [38], [39] are indicators that have been seen many times in literature. Additionally, according to economic theory, these indicators have a direct impact on CPI. GDP and interest rates are both chosen as lagging indicators because of their economic relationship with CPI, according to economic theory, as changes in GDP influence CPI and interest rates change the cost of borrowing and decisions about investment, which affect CPI. PPI is the most used indicator for forecasting CPI in literature, it is chosen as a leading indicator as it typically changes before an economic shift. Furthermore, PPI directly affects CPI, as PPI is the measure of changes in input prices for producers, which over time can be turned into a change in consumer prices.

3.5.1 Data Manipulation

To determine if the features chosen will benefit the process of forecasting CPI, correlation between CPI and the features was calculated using Pearson correlation test, but as Pearson correlation test uses pairwise comparisons, all the datasets have to be of the same length. For US datasets, the shortest dataset started in the second quarter of 1963 and ended in the third quarter of 2023, so all US datasets starting before 1963 had their columns deleted to the second quarter of 1963. For UK datasets, the shortest dataset started from the second quarter of 1960 to the last quarter of 2022, so all UK datasets starting before 1960 or ending after 2022 had their columns deleted to fit the range of the shortest dataset.

3.5.2 Correlation Analysis

After the initial set of features was chosen, correlation tests were used to test the correlation between CPI and other features to demonstrate how the changes in CPI are associated with the other feature's changes. The test was performed on both US features and UK features. GDP is used for gross domestic product, PPI is used for producer price index, and IR is used for interest rates. Pearson correlation test outputs two test results: correlation coefficient and P-value. The correlation coefficient indicates the direction and strength of the linear relationship between the two features, while the P-value is the statistical significance of the correlation. If the P-value is less than the chosen significance level, which is 0.05, there is a significant relationship between CPI and the assessed feature. Table 3.1 displays the Pearson correlation test results between CPI and the chosen variables for both economies. The results showed that there is a significant correlation

Variables	Dataset	Correlation Coefficient	P-value
CPI and GDP	**US**	0.230	2.06×10^{-6}
	UK	0.233	1.97×10^{-4}
CPI and PPI	**US**	0.802	1.58×10^{-55}
	UK	0.710	9.32×10^{-40}
CPI and IR	**US**	0.265	3.02×10^{-5}
	UK	0.296	1.75×10^{-6}

Table 3.1: Pearson Correlation test results between CPI and features

between CPI and GDP, PPI, and IR for both economies, PPI had the most correlation with CPI in both economies. Based on the results, for models using only lagging indicators, GDP and interest rates were used as features to forecast CPI, for the other models that use both lagging and leading, GDP, interest rates, and producer price index were used as features.

3.6 Data Preprocessing

Before feeding data into models, further preprocessing was done to the data to turn it into a form that could be used by our deep neural network models.

3.6.1 Normalisation

Normalisation aims to bring all the features used in the model to a similar scale to restrain the dominance of one feature because of the differences in magnitudes. Normalisation was applied to all the datasets that belong to the same economy because most of the features are measured using different units and indexes, so this can allow power of features over other features. Deep neural networks perform better when their features are normalised and within the same scale because it allows for a smoother, more stable training process. Min-Max scaling method is the most mentioned method in literature review for normalisation, which scales the features in the range of [0,1], each feature is scaled proportionally, preserving the differences between the data points. Min-Max function was applied by first initialising the Scaler variable to be MinMax scaler, then using fit-transform() method from SciKit.learn.preprocessing library. After normalisation was applied, all the features were within the same scale in both the US and UK economies, which allowed for better forecasting performance.

3.7 LSTM Model Configuration

As mentioned in the literature review section above, LSTM is a type of recurrent neural network model that was created to solve the problem of the vanishing gradient and exploding gradient, thus making it efficient in capturing long term dependencies. What makes LSTM suitable for forecasting CPI is its ability to handle irregular patterns because of its memory cell, which allows the model to selectively store and discard information, allowing important past patterns to be remembered. In this section, steps that were taken to transform the data into a format that can be used by the LSTM and the structure of the LSTM used will be presented.

3.7.1 Window Size

Window size is the number of time steps in the time series that is used to make predictions by the model. Our current format is in 2D, formed of dates that are used as indexes and indicators, however, for the LSTM model to function, the data must be in 3D, with a matrix of I data points, a window size of W, and N indicators. There is no right answer for the window size, but it depends on the desired outcome of the time series and the patterns the model aims to capture. It is crucial for our model to be able to capture short term patterns and trends between the data and also capture long term dependencies. Therefore, a window size of 6, equivalent to a year and 2 quarters, was selected for the LSTM model. Once the window size was determined, the data was divided into X (features) and y (target value), with X being used to predict y.

3.7.2 Training and Testing Split

Both X and y were split into training data, validation data, and testing data. Training data were used to train the LSTM model, where the model learns the underlying patterns and features from the input data and the target that corresponds to it. During this process, the weights are tuned and optimised to minimise as much as possible the loss function. Validation data was used in the evaluation of the model performance during training, and the hyperparameter tuning is performed based on the performance of the model on the validation data. The validation data helps to assess the model performance on new unseen data to make sure that the model is learning and not memorising the training data. The testing data was used to test the model after training. The performance of the LSTM model was assessed based on its performance on the testing data, because this

data is where the model's ability to generalise is put to the test. The data were split into 70% going into training data, 10% for validation data, and the remaining 20% for testing. This is a common split that was seen many times in the literature because it gives an optimal balance for the training process, validation, and evaluation to allow the learning of the underlying trends and patterns that were seen in the data, while at the same time giving a good space to evaluate the model performance on unseen data. Table 3.2 shows the split percentage for every part and the corresponding number of data points. The testing set consists of the last 20% of the data to ensure the model is tested on the most recent data.

Split	**Percentage**	**Number of data points US**	**Number of data points UK**
Train	70%	165	171
Validation	10%	23	24
Test	20%	48	50
Total	100%	236	245

Table 3.2: Train, Validation, and Test data split

3.7.3 Model's Hidden Layers

As discussed in literature review sections, hidden layers are the layers that are implemented in between the input layer and output layer. According to theory, a neural network with one hidden layer and an appropriate number of hidden neurons should be able to accurately forecast any continuous function [48], but as we are interested in forecasting CPI using deep neural networks, more than one hidden layer was utilised, as a deep neural network is defined as having two or more hidden layers. The model consists of 3 LSTM layers and 2 dense layers, as our model doesn't suffer from overfitting as the number of data points is not a small amount. Additionally, since we did not use ReLU in training, dropout layers weren't incorporated to prevent zero activations that the network is not used to, which can negatively impact performance. Dense layer is a fully connected layer with the role of learning complex patterns and relationships from the output of the last LSTM and producing an output after adjusting the weights. The hidden layers were followed by a final dense layer, which serves as the output layer and transforms the data into the target value of CPI. This layer has a single output neuron that is the final prediction of our model.

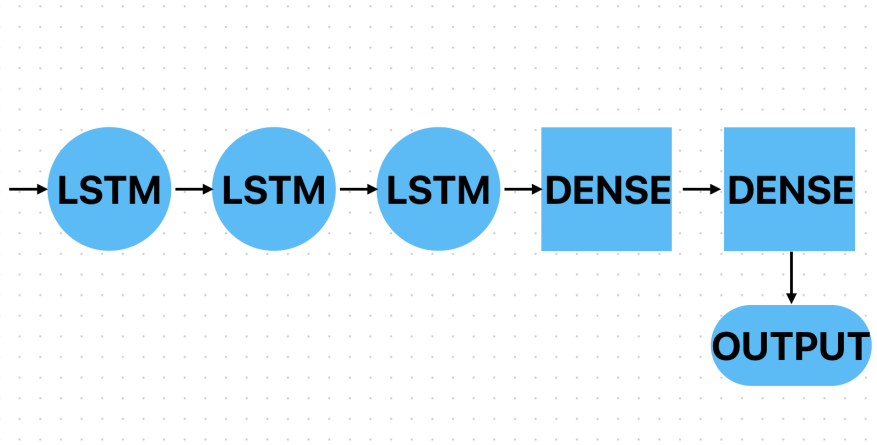

Figure 3.1: LSTM model structure

3.7.4 Activation Function

Non-linearity is introduced through the use of activation functions. Activation functions are mathematical operations that are applied to the output of the neural network's neurons in a way that transforms the output to be able to represent non-linear relationships and patterns. Two types of non-linear activation functions were considered: ReLU and tanh. They are both two of the most commonly used activation functions in neural networks. ReLU is an activation function that simply returns the output if the output is positive or returns 0 if otherwise. The hyperbolic tangent function (tanh) is a non-linear activation function that bounds the values of the input to range between -1 and 1 by mapping the input value into an S-shaped curve. The formula of tanh function is defined as:

$$tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{3.1}$$

where x is the input value. When the values of the inputs to the tanh function are close to 0, only small changes happen that result in an output that is also close to 0, allowing for the preservation of gradients and information during the learning process. When the inputs sheer away from 0, the output of the tanh function goes closer to the extremes 1 and -1, which helps capture larger trends and patterns in the dataset.

During backpropagation, tanh activation function may suffer from the vanishing gradient problem, as we were using LSTM, this wasn't a problem because of the LSTM gating systems that were specifically designed to address the vanishing gradient problem.

The hyperbolic tangent function (tanh) was chosen over ReLU due to its

wider range, encompassing both negative and positive numbers, making it able to capture both positive and negative trends, which is important to our problem as some variables may have a negative correlation with CPI, making it able to capture the relationship between CPI and all the features.

3.7.5 Loss Function

The loss function is used to measure the disparity between the predicted values of the LSTM model and the target values. It is used in the optimisation algorithm when trying to find the set of weights that minimises the loss function. The LSTM model learns to make accurate predictions that minimise the difference between the prediction and real values. There are various types of loss functions, such as Mean Squared Error (MSE) and Mean Absolute error (MAE). According to literature, MSE is the most commonly used loss function. MSE calculates the average squared difference between the predicted values and the target values.
MSE is defined by the following formula:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.2}$$

Where n represents the number of data-points in the dataset, y_i is the actual value for the $i-th$ data point, and $\hat{y}_i$ is the predicted value for the $i-th$ data point.

The decision to use MSE as the loss function was made due to its ability to heavily penalize large errors and prevent positive and negative errors from canceling each other.

3.7.6 Optimisation Function

One of the essential components of the LSTM model is the optimiser. The role of optimiser is to adjust the weights of the model through the learning process, which is done by minimising the loss function. The weights that minimise the loss function were used by the model to forecast.

One of the most referenced optimisers in literature is the Adaptive Movement Estimator (Adam) [47]. Adam optimiser is an extension of Stochastic Gradient Descent that works by combining two methodologies of Gradient Descent: Momentum and Root Mean Square Propagation (RMSP). Using the idea of momentum makes the Adam optimiser converge in the direction

of the minima at a faster rate, especially in the presence of flat or shallow regions. Being inspired by RMSP, the exponential moving average is taken rather than the cumulative sum of squared gradients [49]. The effectiveness of the two above methods is taken over and built on them to introduce the Adam optimizer, which outperforms other optimisers in most cases, therefore, Adam was used in this work.

3.7.7 Hyperparameter tuning

Hyperparameters are parameters that are not learned by the model but must be set manually before the beginning of the model's training. As mentioned in the literature review section above, various algorithms can be used in the hyperparameter tuning process, such as Grid Search, Random Search, and Genetic Algorithms. According to Kokov [4], literature has shown that Random Search generally outperforms Grid Search, except in cases where a small dataset is used. The choice of the hyperparameter tuning algorithm was between Random Search and Genetic Algorithm, Random Search was chosen over Genetic Algorithm because Genetic Algorithm is very computationally expensive. Moreover, Genetic Algorithms have many other parameters that need to be tuned that affect their performance in addition to the hyperparameters that need to be tuned. Due to this complexity, Random Search was used as the hyperparameter tuning algorithm. One of the benefits of the Random Search algorithm is that it balances between exploration and exploitation by traversing the hyperparameter space and exploring it without resulting in an exhaustive exploiting search.

The first step done in the Random Search algorithm was defining the hyperparameter space, the space is defined as shown by table 3.3:

Hyperparameter	**Hyperparameter Space**
1st LSTM layer neurons	Integers between 8 and 512
2nd LSTM layer neurons	Integers between 8 and 512
3rd LSTM layer neurons	Integers between 8 and 512
1st Dense layer neurons	Integers between 8 and 512
2nd Dense layer neurons	Integers between 8 and 512
Learning Rate	[0.00001, 0.0001, 0.001, 0.01, 0.1]
Number of epochs	[100,150,200]
Batch size	[8,16,32,64,128]

Table 3.3: Hyperparameter Space

As shown by table 3.3, these were the hyperparameters that were chosen

to be tuned by the random search with their space assigned. The next step involved randomly sampling N combinations of these hyperparameters from the assigned space, the N chosen was 100. The third step was to train the specified model explained above on every single generated set of parameters from the previous step. During the process of training the model on the generated sets, every trained model's performance was evaluated using MSE on the validation set. The set of parameters that led to the model with the best performance was kept track of, these steps were repeated for every model. Once all models were trained and evaluated, the configuration of parameters that yielded the best performance across all configurations was saved and used as the final hyperparameters for its model.

3.7.8 Final LSTM Structure

As shown by table 3.4, these are the final values of the hyperparameters that were chosen to be tuned by the random search algorithm. These sets of hyperparameters resulted to the lowest Mean Squared Error on the validation set for their respective models.

Hyperparameter	**US lagging**	**US All**	**UK lagging**	**UK All**
1st LSTM layer neurons	341	186	110	301
2nd LSTM layer neurons	384	271	235	257
3rd LSTM layer neurons	210	170	84	458
1st dense layer neurons	182	178	221	511
2nd dense layer neurons	102	306	149	195
Learning rate	0.001	0.001	0.01	0.001
Epochs number	200	200	200	150
Batch size	8	8	16	8

Table 3.4: Optimal set of hyperparameters for all models.

4 Results and Evaluation

4.1 Results

In this section, results of our research questions are shown. First, we show the results of our first research question, where the results obtained by using only lagging features are compared with the results achieved by using both leading and lagging features for both datasets. Then, results of the second research question about whether our models are able to achieve reliable predictions and capture unexpected shocks and trends are illustrated. Following this, we show the results of our final research question about the hyperparameter tuning algorithm ability to output a hyperparameter set that achieves the optimal prediction on both economies.

4.1.1 Benefit Of Having Lagging And Leading Indicators As Covariates

In this section, results of our first experiment and research question are illustrated. This is done by having two models for each economy (US and UK); one has only lagging indicators to forecast CPI, the second model has the same lagging indicators as the first model plus a leading indicator. The comparison of models for each economy is based on their Root Squared Error and Mean Squared Error scores on testing data. The lagging indicators used for this experiment are GDP and Interest Rates, the leading indicator used is PPI, which is known to be a reliable predictor of CPI [50]. Figures 4.1 and 4.2 are bar charts of both economies with the MSE and RMSE values of each model displayed. In every bar chart, two models are compared, one using only lagging features, the other using both leading and lagging. The MSE and RMSE values were obtained by running each model ten times and recording the results each time. The mean average of both MSE and RMSE was then calculated.

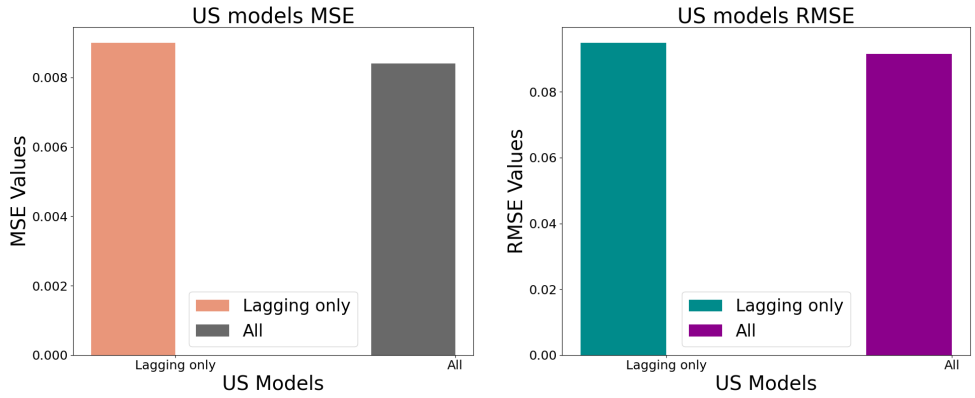

Figure 4.1: MSE and RMSE values of both US models - MSE values on the left, RMSE values on the right.

US Model

Figure 4.1 shows the results of the performance of US models for both the use of lagging indicators only and the use of lagging indicators plus a leading indicator, which is PPI. As seen in the figure, models with both leading and lagging features achieved slightly better MSE and RMSE values. The MSE of the model that has only lagging features is 0.0090, while the RMSE is 0.0949. For the model using both types of indicators, its MSE and RMSE are 0.0084 and 0.0916 respectively, so the model having both leading and lagging features has hardly less MSE and RMSE than the other model.

To investigate if the difference in performance between the two models is significant, a statistical test was conducted. The samples are naturally paired because both models are trained and tested on US CPI dataset, and the distribution is not normal, so Wilcoxon signed-rank test [51] was used as it fits the conditions. The test is performed on the 10 RMSE values obtained by running each model 10 times and the results are shown in table 4.1. As seen, the p-value is greater than 0.05, which means that the null hypothesis can't be rejected, so the difference in performance between the two models is not statistically significant. These results suggest that adding PPI as a leading indicator to the features didn't really improve the prediction performance, as expected according to economic theory.

Economy	**Test Statistic**	**p-value**
US models	11.0	0.1055
UK models	2.0	0.0059

Table 4.1: Wilcoxon signed-rank sum test comparing RMSE in models with lagged and leading features

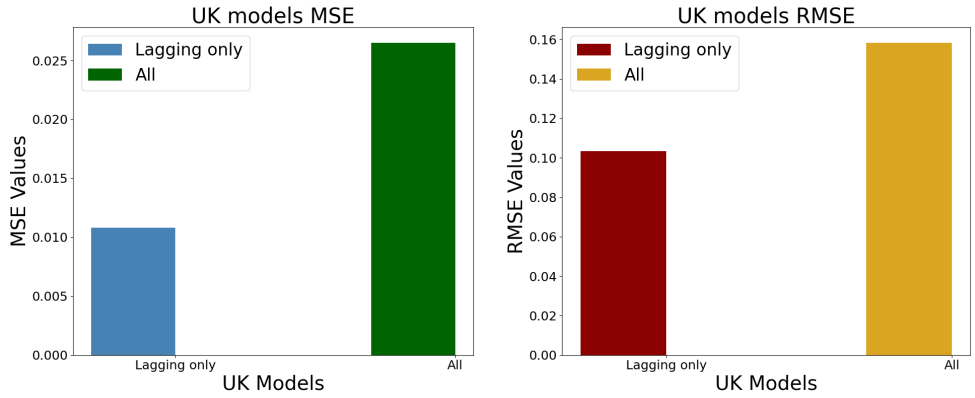

Figure 4.2: MSE and RMSE values of both UK models - MSE values on the left, RMSE values on the right.

UK Model

Contrary to the US model, based on figure 4.2, using only lagging indicators has achieved significantly better results than using lagging and leading indicators to forecast CPI. The UK model that has only lagging indicators as features scored an MSE of 0.0108 and an RMSE of 0.1033. On the other hand, the UK model that has both leading and lagging features has MSE of 0.0265 and RMSE of 0.1583.

To determine if the difference in performance between the two models is significant, Wilcoxon signed-rank test was used. The test was performed on the 10 RMSE values obtained by running each model 10 times. As shown in table 4.1, the p-value is smaller than 0.05, indicating that the null hypothesis could be rejected, meaning that the difference in performance is statistically significant. According to economic theory, this result is unexpected because the use of leading indicators, such as PPI, should improve the predictive ability of the model by allowing it to forecast patterns and economic trends before they happen. However, this result can be explained because of the high collinearity that was found between CPI and PPI, as shown in table 3.1, this high collinearity weakened the estimates of the model, leading it to make inflated errors that made the model less reliable. Additionally, adding a feature with a very high correlation doesn't add any new information the CPI didn't already add, it just adds unnecessary noise to the model and will increase its complexity without adding informative information.

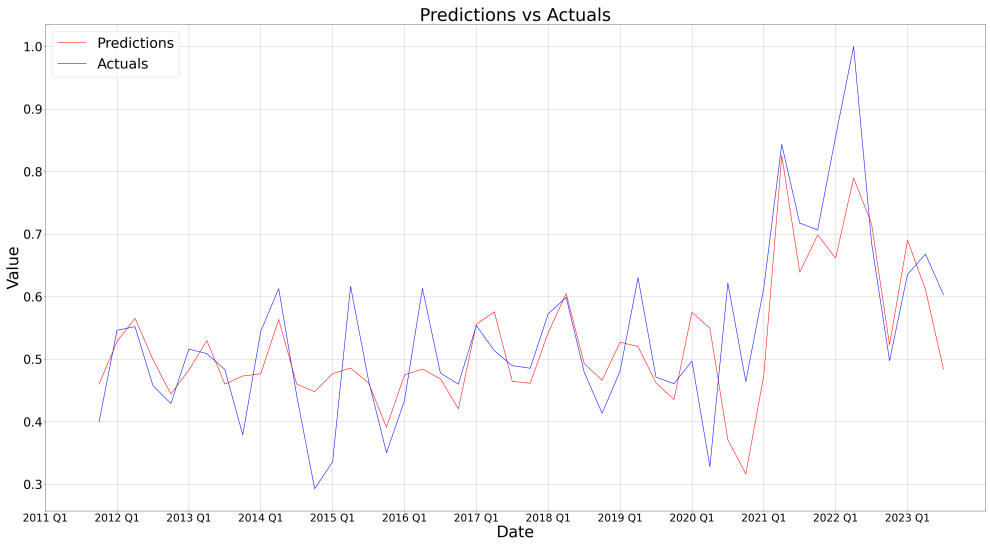

Figure 4.3: US model prediction on testing data

4.1.2 Ability To Handle Patterns And Unexpected Events And Achieve An Accurate Prediction

In this section, we will determine whether the US and UK models accurately forecasted and predicted the patterns and events of their respective economies using test data. For this research question, we will work with both US and UK models that have only the lagging features without the leading feature. This decision was based on the results of the previous section, which showed that adding PPI as a leading feature did not significantly improve the US model according to the statistical test. However, for the UK dataset, using only lagging indicators resulted in a significant improvement, as determined by the statistical test. To have a fair comparison between the models, same indicators have to be used. Therefore, in the upcoming sections, we will only use GDP and interest rates as covariates to forecast CPI.

US Model

Figure 4.3 displays the US model's prediction on the testing data. The red line is the prediction of the model, the blue line is the actual values, the X axis shows the years that are contained in the testing data, and the Y axis are the values of CPI. As seen by the graph, the model does a good job in predicting trends and patterns in the testing data at most times, but it struggles with accurately predicting the magnitude of these patterns. While it accurately predicts the general decline or incline shown in the graph, it fails to accurately predict the extent of these changes. This model achieved

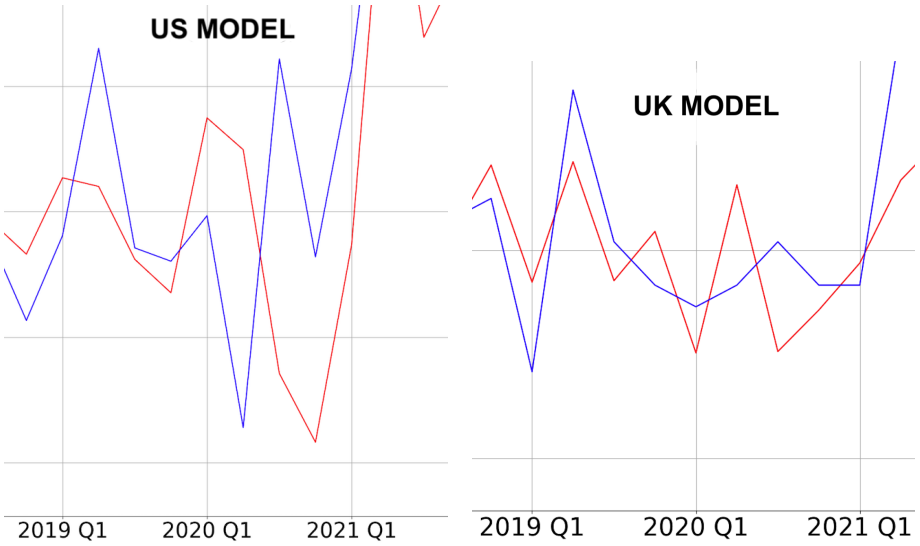

Figure 4.4: Models prediction on testing data - zoomed in on Covid 19 period, on left is the US model , on right is the UK model

an average MSE of 0.0090 and an average RMSE of 0.0949, this result was obtained by running the model ten times and taking the average of MSE and RMSE on the testing data.

As visible in figure 4.4, the model was unable to predict the Covid 19 recession that is shown starting in the first quarter of 2020 and ending in the second quarter of 2020. The model had a lagged response to it, starting the decline one time step later than the original time. This can be explained for various reasons. The first reason is that in the training data, few events are qualified to be called a recession, so the model couldn't learn enough from the training data to be able to predict in time the Covid 19 recession. Additionally, the unprecedented measures taken in response to the pandemic, such as lockdowns and restrictions on economic activity, had unexpected consequences on people's behaviour and economic activity. These outcomes were difficult for the model to predict in a timely manner.

UK model

The prediction of the testing data and the actual values of the UK model are shown in figure 4.5. Red line is the predictions, blue line is the actual values, X axis is the years that make up the testing data, and the Y axis is the values of CPI. As seen in the graph, the model is able to make some accurate predictions for the testing data. However, similar to the US model, the UK model also struggles with predicting the exact patterns of the testing data, such as inclines or declines in values. It is able to capture the general trend, but not the extent of the incline or decline. Moreover, it can be seen

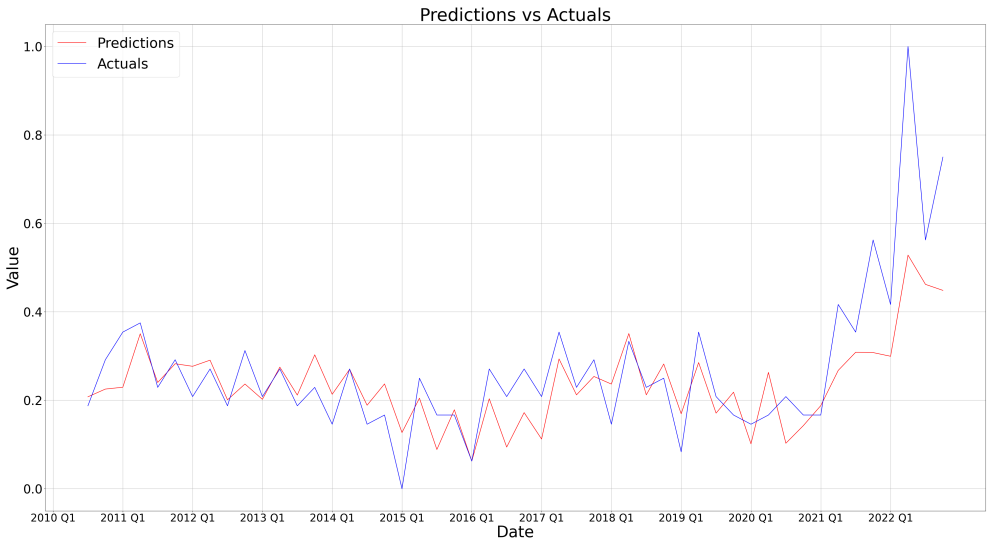

Figure 4.5: UK prediction on testing data

in the graph that when the model is approaching the last timestamps, it is struggling with the prediction compared to the forecasts at the beginning. This phenomenon is called "forecast horizon effect," which occurs because the uncertainty about patterns and trends in the future increases when moving further into future and away from training data. Moreover, as we move away from the training data, the underlying relationships and patterns differ more, making it easier to predict data points that are near the training set and harder to predict data points further away from the training set. This model achieved an average MSE of 0.0108 and an average RMSE of 0.1033, this result was obtained by running the model ten times and taking the average of MSE and RMSE on the testing data.

Despite the US model's inability to predict the Covid 19 recession and the resulting decline, the UK model was able to fully capture it. The decline can be seen in the graph reaching its peak in 2020. The reason that the UK model was able to predict Covid 19 recession, but the US model couldn't is because US economy was more affected by the Covid 19 virus than the UK economy because of the different size of population between the two countries, which led to more spread in US, resulting in more restrictions and measures such as lockdowns for longer periods that affected the economy to an extent that the US model couldn't capture. As shown in both graphs, the drop in the US economy due to Covid 19 was much deeper than that of the UK, highlighting the extent of the virus's effect on both economies.

Despite the issues mentioned for both models, the MSE and RMSE of both models are very low. This shows that to be able to make an accurate judgement of a model's performance, error metrics such as MSE and RMSE are not sufficient, they must be complemented by human evaluation.

4.1.3 Finding Optimal Hyperparameters For Both Economies

In this section,we will address our final research question by conducting an experiment to determine if the hyperparameter tuning algorithm successfully identified an optimal set of hyperparameters for both the US and UK datasets. In order to do that, for both economies, the results achieved using the optimal set found by the hyperparameter tuning algorithm for their own economy will be compared and tested against the results achieved using the optimal set found for the other economy model by the hyperparameter tuning algorithm. As in previous sections, the models for both US and UK economies will have GDP and interest rates as their features to forecast CPI for their respective dataset.

The models with different sets of hyperparameters are going to be compared based on their MSE and the RMSE on the testing set.

Model	Set	MSE	RMSE
US model	US optimal set	0.0090	0.0949
	UK optimal set	0.0127	0.1126
UK model	UK optimal set	0.0108	0.1033
	US optimal set	0.0137	0.1157

Table 4.2: Results for 2nd experiment

Table 4.2 shows the MSE and RMSE of the two different models (US and UK models) when using different sets that were found by random search. As the table shows, each model works best with its own set of hyperparameters, which were found by the hyperparameter tuning algorithm for its respective economy. When the models are trained and tested using the other set, the MSE and the RMSE increased. For the US model, the difference in MSE scores between the models with different sets is only 0.0037, and the difference in RMSE is 0.0177. For the UK model, the difference between the values of MSE and RMSE of the models with different hyperparameters sets got smaller compared to the US model, where the difference between the values of MSE is 0.0029 and for RMSE is 0.0124.

Models In Comparison	Test Statistic	p-value
US model using US set, UK set	1.0	0.004
UK models using UK set, US set	13.0	0.160

Table 4.3: Wilcoxon signed-rank sum test comparing RMSE in models with different sets of hyperparameters

To test whether the differences in RMSE values are statistically significant, a Wilcoxon signed-rank test was utilised. This test was chosen because

the sample distribution was not normal and the samples were paired, as they were obtained from two models with different hyperparameters but the same features and dataset. The test is performed on the 10 RMSE values obtained by running each model 10 times. Table 4.3 shows the results for both economies models. As seen in the table, the results indicate that the difference in performance between the two US models is significant, as evidenced by having a p-value less than 0.05, which allowed the rejection of the null hypothesis. This means that the US model using the US set of hyperparameters had a significantly better prediction than the US model using the UK set. However, for the UK models, the p-value is greater than 0.05, indicating that the null hypothesis could not be rejected and that there was no significant difference in performance. This means that both the two models had nearly the same prediction performance. These results demonstrate that the hyperparameter tuning algorithm was successful in identifying an optimal set of hyperparameters, the US set, which can be used with either of the two datasets and achieve better or as good performance compared to any other set of hyperparameters.

4.2 Comparison To Other Work

In this section, the performance of both our UK models: one that uses a lagging only dataset and one that uses both lagging and leading data are compared to the one-step LSTM models used in Skea's [13] study, which forecasts UK inflation (RPI). Skea's model is tried on two datasets; the first one only contains lagging features, including inflation, GDP, and interest rates, the other dataset has the same features plus the change in production output, which serves as a leading indicator. This specific study was chosen to be compared because in both studies, a component of inflation is forecasted, nearly the same indicators are used, both studies use LSTM, and both studies have datasets with nearly the same years worth of data. Models will be compared based on their RMSE values on testing data of their respective datasets.

Models	**Dataset**	**RMSE**
This model	**lagging only**	0.1033
This model	**lagging and leading**	0.1583
Skea's model	**lagging only**	0.8519
Skea's model	**lagging and leading**	1.0464

Table 4.4: Comparison of this work UK LSTM model performance and Skea's UK LSTM model performance

As shown in figure 4.6, our model beats Skea's model when using both

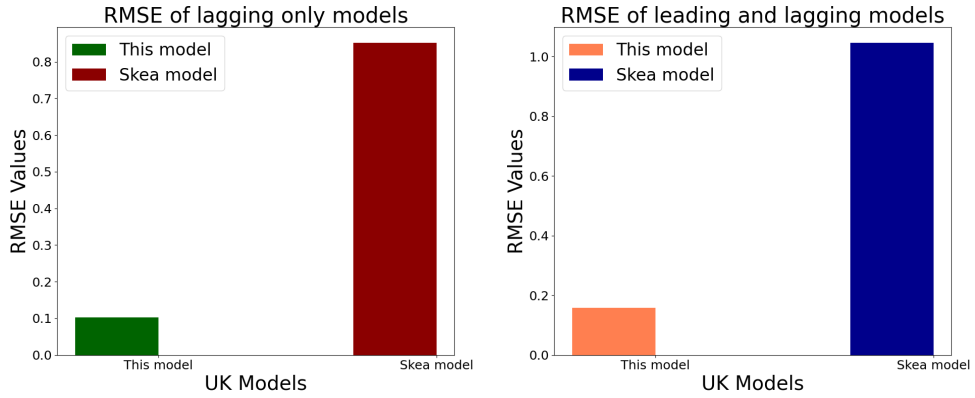

Figure 4.6: Comparison of RMSE between this work and Skea's work in both datasets.

datasets, our models scores much lower RMSE scores than Skea's models. As seen in table 4.4, the difference in RMSE for the lagging only dataset is 0.7486, for the other dataset, the difference between the 2 models is 0.8881, Both differences can be seen as significant, indicating that both of our models had much better performance than Skea's models.

The decrease in RMSE in our models can be attributed to several factors. Firstly, Skea's LSTM has only one LSTM layer with a small number of neurons, making it a shallow model that is unable to capture complex patterns because of the small number of layers and neurons. Moreover, our models use ADAM as the optimiser, while Skea's models use SGD. In literature, it is seen many times that ADAM outperforms SGD as an optimiser, as SGD often gets stuck in local minima, hindering the process of finding the optimal weights and ultimately affecting the model's performance. Also, the benefit of using a hyperparameter tuning algorithm such as Random Search can be seen in the difference between our models average RMSE and Skea's models RMSE. The results confirm that using hyperparameter tuning algorithm enhances the model's performance as it leads to the optimal set of hyperparameters that result in the best performance of the model. While using a hyperparameter tuning algorithm may be more computationally expensive and time consuming, the benefits outweigh the drawbacks, as shown by the results. In addition, Skea uses linear activation functions for his models, which don't allow the model to represent non-linear patterns. In contrast, our models use tanh, which allows us to introduce non-linearity to our models and have a better representation of complex patterns.

5 Conclusions

In this study, the utilisation of deep neural networks for forecasting macroeconomic parameters was analysed. Our framework included features selection, preprocessing of data, building models, hyperparameters fine tuning, model training, testing on a test set, and evaluation. In this work, the LSTM architecture was experimented with by having three versions built for each economy: a version using only lagging features, a version using both leading and lagging features, and a version that uses the other economy hyperparameter set. This was done to investigate the research questions. The outcomes of using leading indicators were investigated. Performance of the models in both economies was analysed. Lastly, results were interpreted to find out if random search was able to find an optimal set that worked on both economies. It has been found that adding a leading indicator as a feature didn't improve the performance of the models. The results showed that models for both economies were able to accurately predict the testing data at most time points, the US model wasn't able to capture the Covid 19 recession in time, however, the UK model predicted the Covid 19 recession in time. Finally, our random search algorithm was capable of finding a set of hyperparameters that resulted in an optimal performance of a model when trained and tested on any of the two economies.

Future Work

There are several potential avenues for further development of this work. One possibility is to incorporate additional macroeconomic indicators, including both leading and lagging indicators, and analyse their impact on the model's performance. The use of further architectures can be explored, such as CNNs and encoder decoders. Deeper levels of layers can be experimented with, which can lead to the estimation of the best number of layers for this field. Furthermore,expanding the model to include more economies, such as the EU and China, as well as smaller economies, could provide a more comprehensive understanding of its effectiveness. Moreover, different hyperparameter tuning algorithms can be tried, such as Bayesian optimisation and genetic algorithms. Another direction for research in the future is to test the model's ability to make accurate predictions on economies it was not trained on, in order to assess its generalisability.

A Appendix : Hyperparameter Tuning Algorithm Code

```
import random
def get_hyper_param(iterations_no):
  hyper_param = []
  for i in range(iterations_no):
    current_params = []
    current_params.append(np.random.randint(8,513))  #lstm1_neurons
    current_params.append(np.random.randint(8,513))  # lstm2_neurons
    current_params.append(np.random.randint(8,513))  # lstm3_neurons
    current_params.append(np.random.randint(8,513))  # dense_neurons
    current_params.append(np.random.randint(8,513)) # dense2_neurons
    current_params.append(random.choice([0.00001,0.0001,0.001,0.01,0.1]))
     #learning rate
    current_params.append(random.choice([100,150,200])) #epochs
    current_params.append(random.choice([8,16,32,64,128])) #batch size
    hyper_param.append(current_params)

  return hyper_param

hyper_parameter = get_hyper_param(100)
hyper_parameter
```

Figure A.1: Randomly sampling and generating combinations of hyperparameters.

```
train_loss, train_root_mean_sqr_error, val_loss, val_root_mean_sqr_error = list(), list(),
 list(), list()
best_params =[]
def fit_lstm_random(batch_size):
  min_val_loss = 999
  min_test_loss = 999
  for lstm1_neurons, lstm2_neurons,lstm3_neurons,dense_neurons,dense2_neurons, learning_rate,n_epochs
  , batch_size in hyper_parameter:
    model = Sequential()
    model.add(LSTM(lstm1_neurons,input_shape=(6,3),return_sequences=True,activation ='tanh'))
    model.add(Dropout(0.0))

    model.add(LSTM(lstm2_neurons,return_sequences = True,activation ='tanh'))
    model.add(Dropout(0.0))

    model.add(LSTM(lstm3_neurons,return_sequences = False,activation ='tanh'))
    model.add(Dropout(0.0))

    model.add(Dense(dense_neurons,activation = 'linear'))
    model.add(Dropout(0.0))

    model.add(Dense(dense2_neurons,activation = 'linear'))
    model.add(Dropout(0.0))

    model.add(Dense(1,activation = "linear"))
    model.compile(loss=MeanSquaredError(),optimizer = Adam(learning_rate=learning_rate),metrics =
     [RootMeanSquaredError()])
    lstm_model = model.fit(X_train,y_train, validation_data = (X_val,y_val),epochs =
                           n_epochs,batch_size = batch_size,verbose = 2)
    current_val_loss = lstm_model.history['val_loss'][-1]
    test_results = model.evaluate(X_test, y_test, verbose=0)
    current_test_loss = test_results[0]

    if current_val_loss < min_val_loss:
      min_val_loss = current_val_loss
      best_params= [lstm1_neurons, lstm2_neurons,lstm3_neurons,dense_neurons, dense2_neurons,
                    learning_rate,  n_epochs, batch_size]
      print('best_params',best_params)
      print('min_val',min_val_loss)

  print('final best params',best_params)
  print('final best val error',min_val_loss)
  return best_params
```

Figure A.2: Training each combination and testing it on validation set to get the optimal set.

B Appendix : First Research Question Results

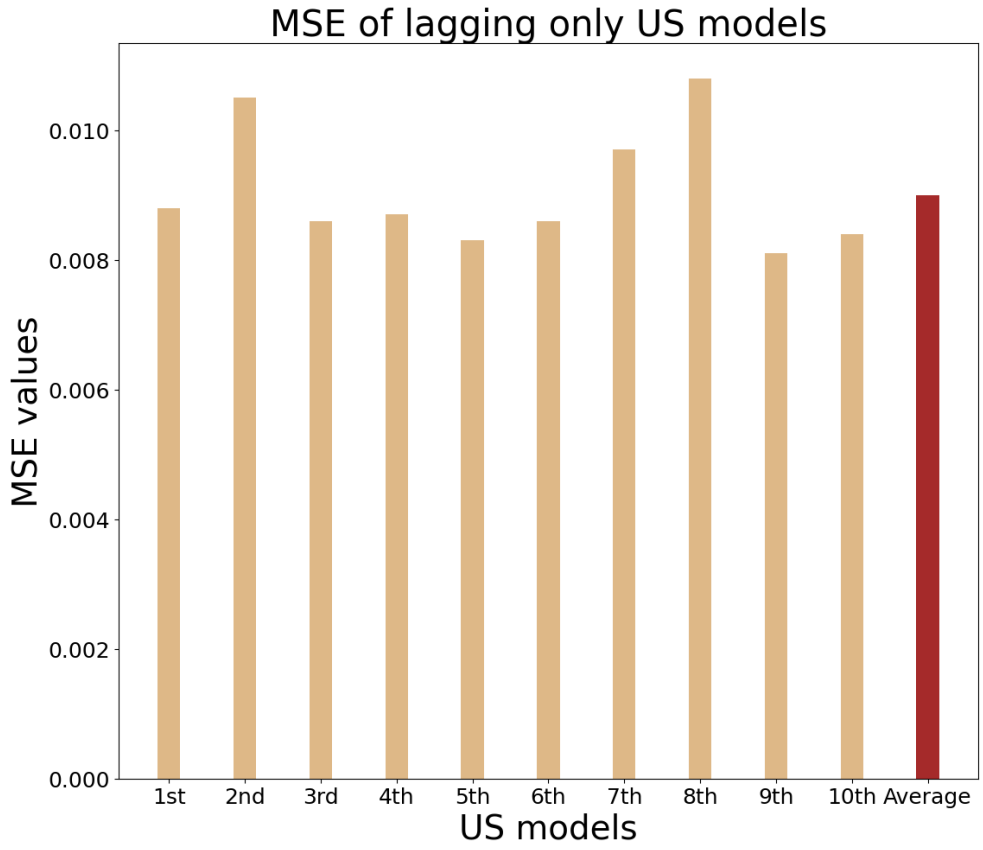

Figure B.1: MSE values of the ten times the US model with lagging features was run, along with the average value of MSE.

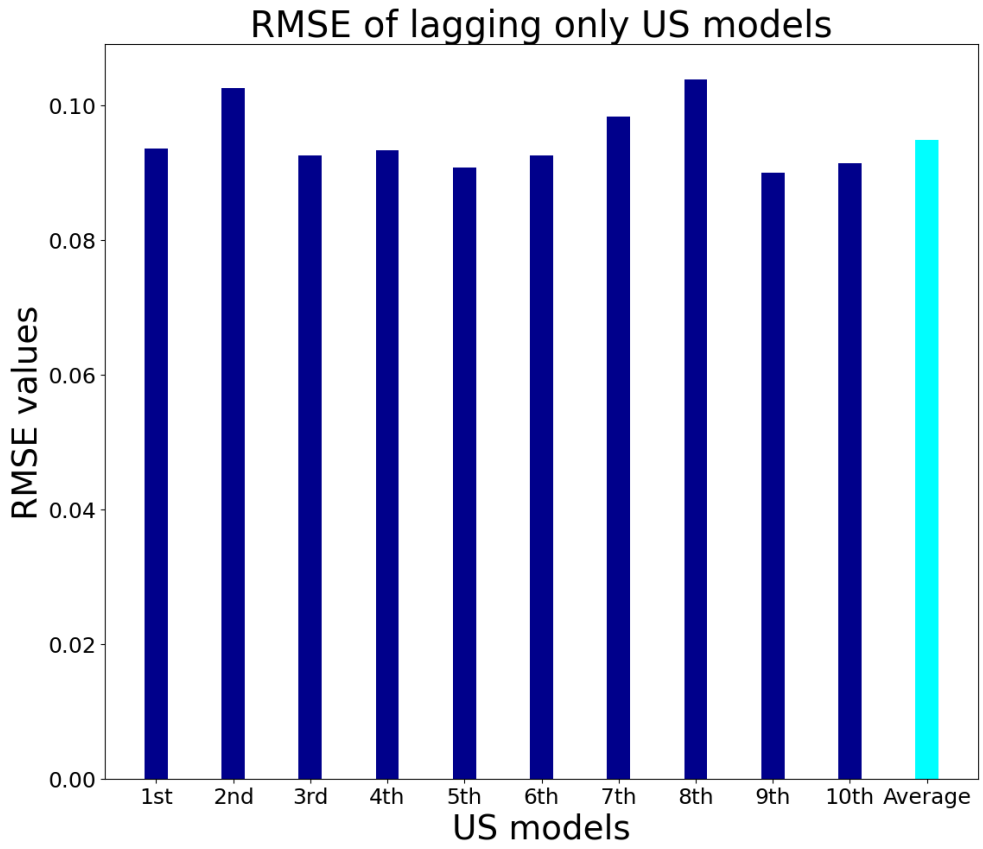

Figure B.2: RMSE values of the ten times the US model with lagging features was run, along with the average value of RMSE.

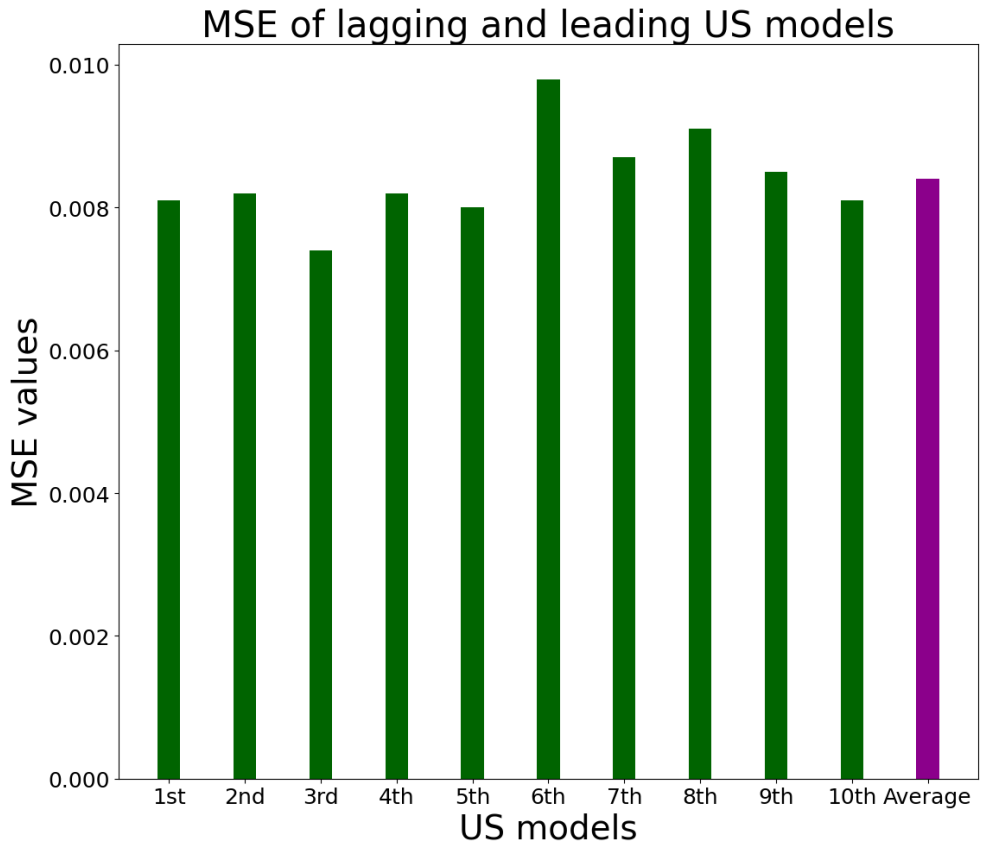

Figure B.3: MSE values of the ten times the US model with lagging and leading features was run, along with the average value of MSE.

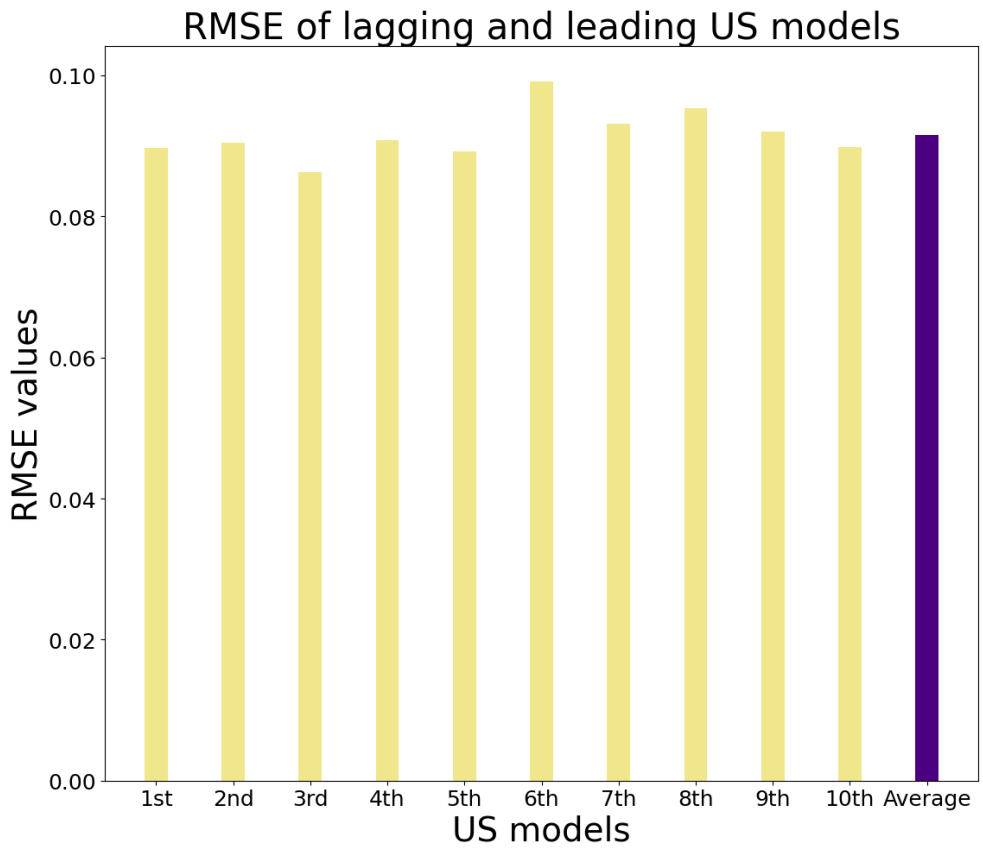

Figure B.4: RMSE values of the ten times the US model with lagging and leading features was run, along with the average value of RMSE.

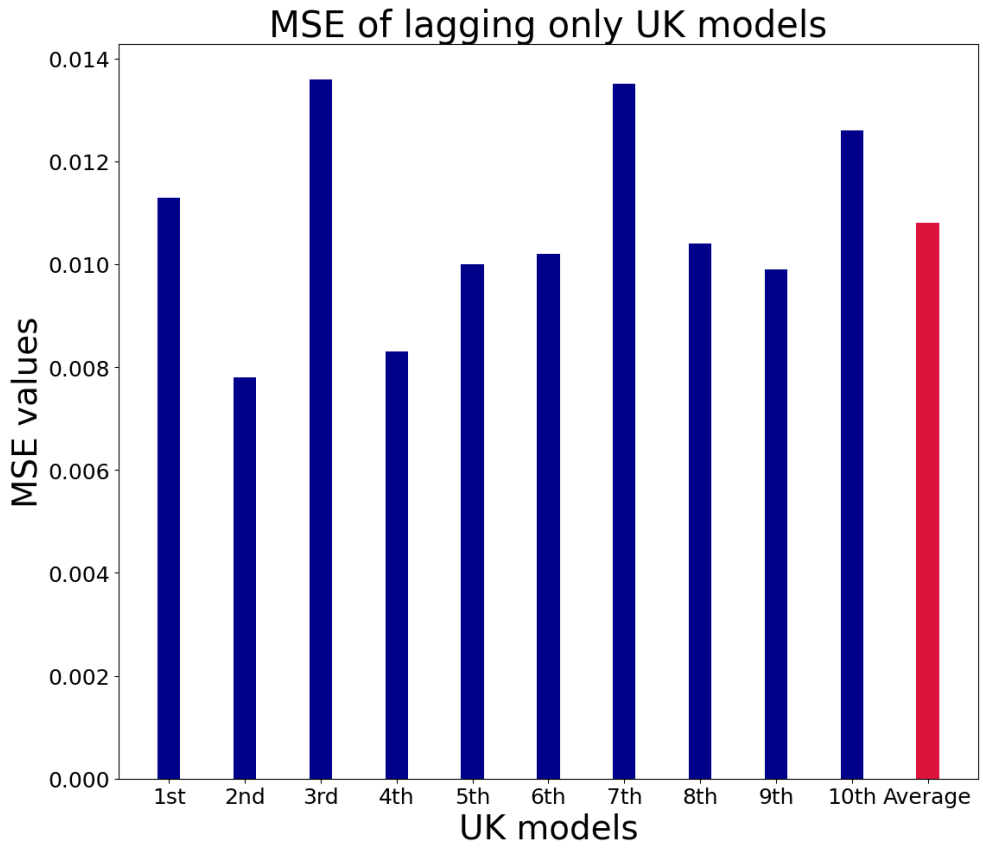

Figure B.5: MSE values of the ten times the UK model with lagging features was run, along with the average value of MSE.

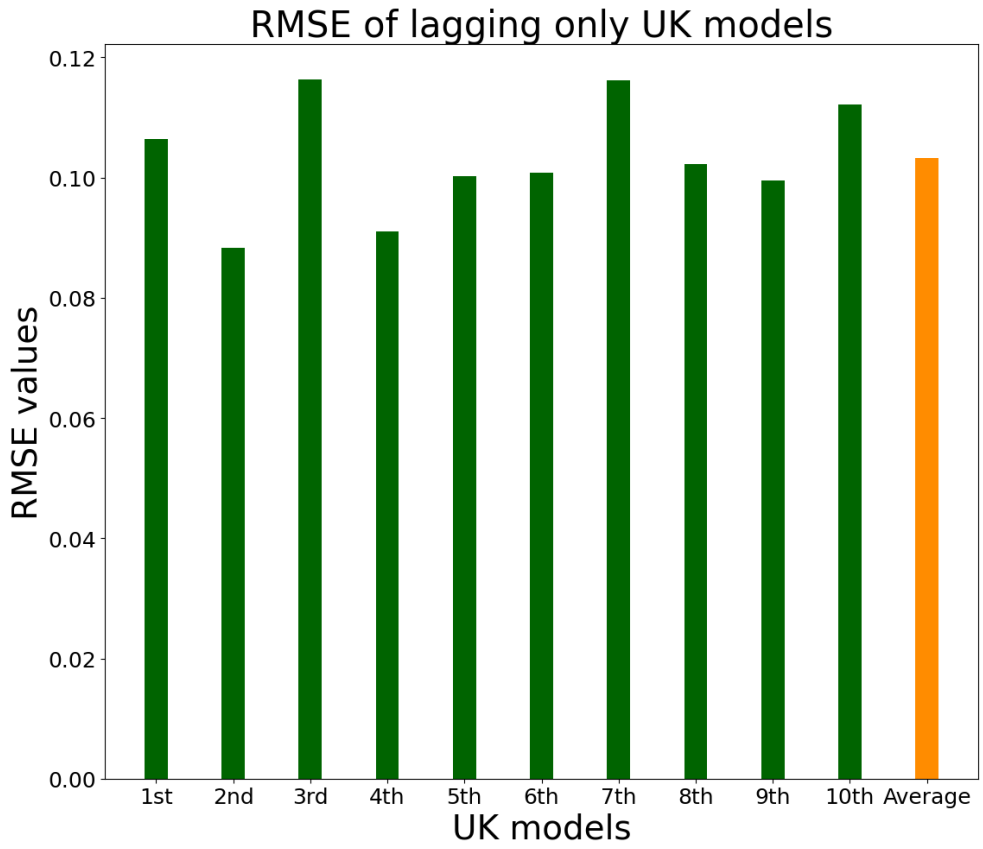

Figure B.6: RMSE values of the ten times the UK model with lagging features was run, along with the average value of RMSE.

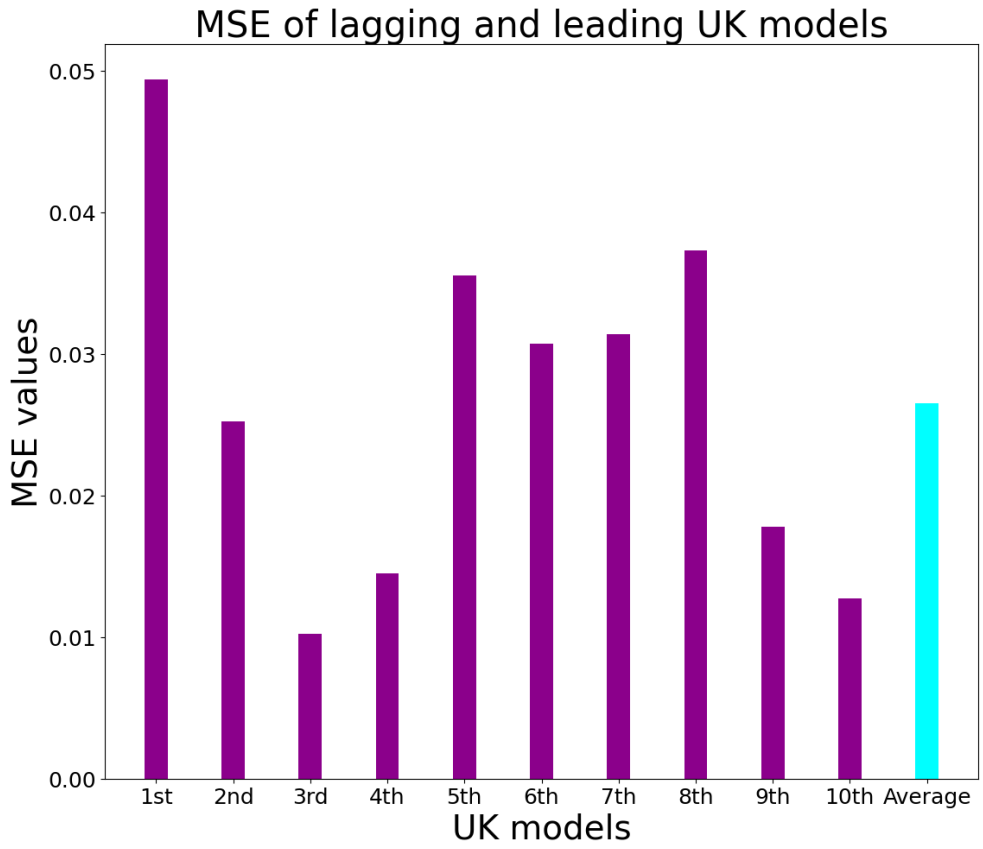

Figure B.7: MSE values of the ten times the UK model with lagging and leading features was run, along with the average value of MSE.

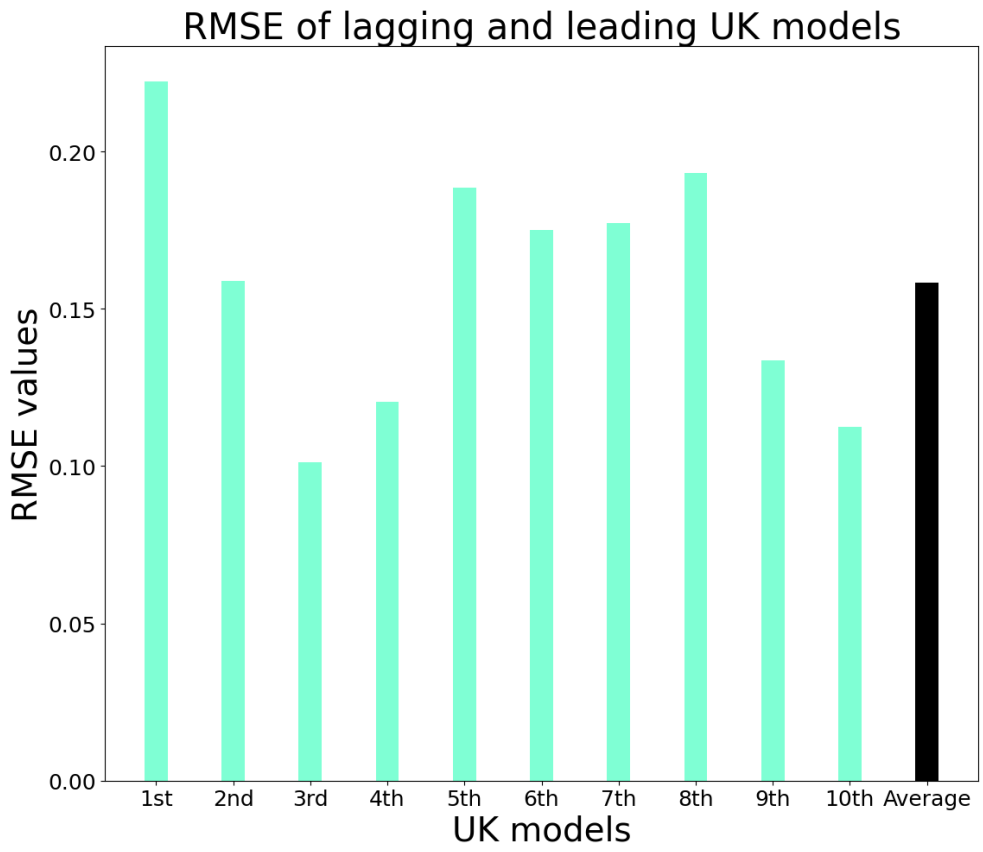

Figure B.8: RMSE values of the ten times the UK model with lagging and leading features was run, along with the average value of RMSE.

C Appendix : Third Research Question Results

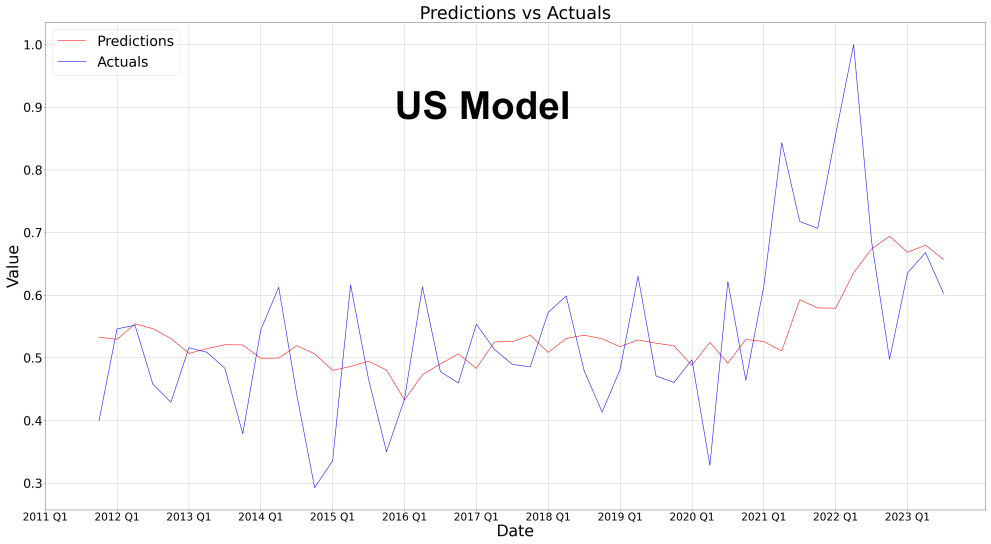

Figure C.1: Performance of US model when trained and tested using UK hyperparameter set.

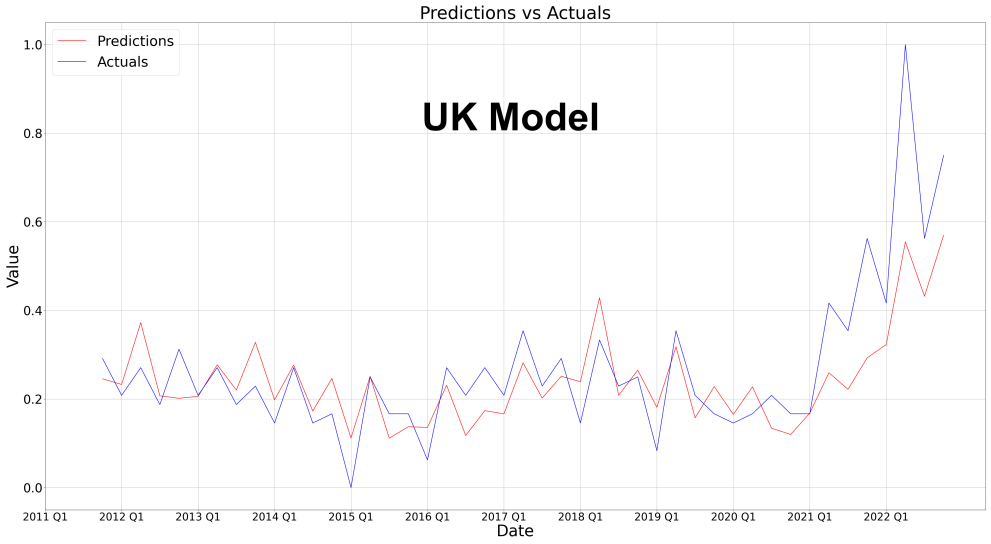

Figure C.2: Performance of UK model when trained and tested using US hyperparameter set.

D Appendix : Comparing Our Model

Hyperparameter	Skea's model
1st LSTM layer units	32
Number of lags	12
Optimiser function	SGD
Loss function	MSE
Dropout rate	0
Activation function	Linear
Epochs number	15

Table D.1: Skea's [13] model configuration for forecasting UK inflation.

Bibliography

[1] T. Cook and A. Smalter Hall, “Macroeconomic Indicator Forecasting with Deep Neural Networks,” The Federal Reserve Bank of Kansas City Research Working Papers, 2017, doi: https://doi.org/10.18651/rwp2017-11.

[2] S. Gonzalez, “Neural Networks for Macroeconomic Forecasting: A Complementary Approach to Linear Regression Models,” *Working Papers-Department of Finance Canada*, 2007, Available: https://ideas.repec.org/p/fca/wpfnca/2000-07.html

[3] L. Paranhos, “Predicting Inflation with Neural Networks,” The Warwick Economics Research Paper Series (TWERPS), 2021, Available: https://ideas.repec.org/p/wrk/warwec/1344.html

[4] S. Kokov, “Forecasting Macroeconomic Parameters with Deep Learning Neural Networks,” GitHub, Apr. 23, 2023. https://github.com/kokoff/deep-forecast

[5] Z. Wang, K. Li, S. Q. Xia, and H. Liu, “Economic Recession Prediction Using Deep Neural Network,” The Journal of Financial Data Science, vol. 4, no. 3, pp. 108–127, Jun. 2022, doi: https://doi.org/10.3905/jfds.2022.1.097.

[6] Z. C. Lipton, J. Berkowitz, and C. Elkan, “A Critical Review of Recurrent Neural Networks for Sequence Learning,” arXiv.org, 2015. https://arxiv.org/abs/1506.00019

[7] E. Onder, F. Bayır, and A. Hepsen, “Forecasting Macroeconomic Variables Using Artificial Neural Network and Traditional Smoothing Techniques,” *Journal of Applied Finance and Banking*, vol. 3, May 2013, doi: https://doi.org/10.2139/ssrn.2264379.

[8] M. H. Kim, J. H. Kim, K. Lee, and G.-Y. Gim, “The Prediction of COVID-19 Using LSTM Algorithms,” International Journal of Networked and Distributed Computing, vol. 9, no. 1, p. 19, 2021, doi: https://doi.org/10.2991/ijndc.k.201218.003.

[9] L. Longo, M. Riccaboni, and A. Rungi, “A neural network en-

semble approach for GDP forecasting," Journal of Economic Dynamics and Control, vol. 134, p. 104278, Jan. 2022, doi: https://doi.org/10.1016/j.jedc.2021.104278.

[10] N. Zyatkov and O. Krivorotko, "Forecasting Recessions in the US Economy Using Machine Learning Methods," 2021 17th International Asian School-Seminar "Optimization Problems of Complex Systems (OPCS), Novosibirsk, Russian Federation, 2021, pp. 139-146, doi: 10.1109/OPCS53376.2021.9588678.

[11] H. Chung and K. Shin, "Genetic Algorithm-Optimized Long Short-Term Memory Network for Stock Market Prediction," Sustainability, vol. 10, no. 10, p. 3765, Oct. 2018, doi: https://doi.org/10.3390/su10103765.

[12] Y. Wang, "Macroeconomic Forecasting," Dissertation, University of York, 2023.

[13] A. Skea, "Macroeconomic Forecasting Using Neural Networks," Dissertation, University of York, 2019.

[14] C. Chatfield, Time-series forecasting. Boca Raton: Chapman Hall/CRC, 2001.

[15] R. H. Shumway and D. S. Stoffer, Time series analysis and its applications : with R examples. Cham, Switzerland: Springer, 2017.

[16] N Gregory Mankiw, Principles of macroeconomics. Australia ; Boston, Ma: Cengage Learning, 2018.

[17] R. J. Barro, Macroeconomics : a modern approach. Mason, Ohio: Thomson South-Western, Cop, 2008.

[18] B. Snowdon and H. R. Vane, Modern macroeconomics : its origins, development and current state. Cheltenham ; Northampton: Edward Elgar, 2014.

[19] List of major leading Lagging Economic Indicators Leading Indicators, https://www.wtwealthmanagement.com/documents/pdf/WTWealth-2014-07-MajorLeadingList.pdf (accessed Nov. 2023).

[20] R. A. Emerson and D. F. Hendry, "An evaluation of forecasting using leading indicators," *Journal of Forecasting*, vol. 15, no. 4, pp. 271–291, Jul. 1996, doi: https://doi.org/10.1002/(sici)1099-131x(199607)15:4%3C271::aid-for623%3E3.0.co;2-7.

[21] N. Reiff, "Leading Indicators: Definition and How They're Used by Investors," Investopedia, Feb. 15, 2023. ht-

tps://www.investopedia.com/terms/l/leadingindicator.asp: :text=What%20Are%20Leading%20and%20Lagging

[22] Investopedia Team, "What Are Leading, Lagging, and Coincident Indicators?," Investopedia, Jan. 01, 2021. https://www.investopedia.com/ask/answers/what_are_leading_lagging_and_coincident_indicators/

[23] V. Collin, "Leading, Lagging Coincident Economic Indicators," Financial Edge, Oct. 09, 2021. https://www.fe.training/free-resources/financial-markets/leading-lagging-coincident-economic-indicators/

[24] Investopedia Team, "Economic Indicator: Definition and How to Interpret," Investopedia, Dec. 22, 2023. https://www.investopedia.com/terms/e/economic_indicator.asp

[25] "Macroeconomic indicators: List Performance" StudySmarter UK. https://www.studysmarter.co.uk/explanations/macroeconomics/economic-performance/macroeconomic-indicators/

[26] J. Fernando, "Consumer Price Index - CPI," Investopedia, Mar. 13, 2024. https://www.investopedia.com/terms/c/consumerpriceindex.asp

[27] J. Brownlee, "How to Configure the Number of Layers and Nodes in a Neural Network," Machine Learning Mastery, Aug. 06, 2019. https://machinelearningmastery.com/how-to-configure-the-number-of-layers-and-nodes-in-a-neural-network/

[28] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[29] D. Sivakumar, "Hyperparameters Tuning in Neural Networks," Scaler Topics, May 04, 2023. https://www.scaler.com/topics/deep-learning/neural-network-hyperparameters-tuning/

[30] S. Mehta, "How to use genetic algorithm for hyperparameter tuning of ML models?," Analytics India Magazine, Jun. 28, 2022. https://analyticsindiamag.com/how-to-use-genetic-algorithm-for-hyperparameter-tuning-of-ml-models/

[31] Federal Reserve Bank of St. Louis, "FRED Economic Data," Stlouisfed.org. https://fred.stlouisfed.org/

[32] "Consumer Price Index for All Urban Consumers: All Items," Stlouisfed.org, 1947. https://fred.stlouisfed.org/series/CPIAUCSL

[33] "Consumer Price Index: All Items: Total for United Kingdom," FRED, Federal Reserve Bank of St. Louis, Jan. 01, 1955. https://fred.stlouisfed.org/series/GBRCPALTT01IXNBM

[34] "Gross Domestic Product," FRED, Federal Reserve Bank of St. Louis, Jan. 01, 1947. https://fred.stlouisfed.org/series/NA000334Q

[35] "Gross Domestic Product for United Kingdom," Stlouisfed.org, 1955. https://fred.stlouisfed.org/series/UKNGDP

[36] "Long-Term Government Bond Yields: 10-year: Main (Including Benchmark) for the United States," FRED, Federal Reserve Bank of St. Louis, Jan. 01, 1955. https://fred.stlouisfed.org/series/IRLTLT01USM156N0

[37] "Long-Term Government Bond Yields: 10-year: Main (Including Benchmark) for the United Kingdom," FRED, Federal Reserve Bank of St. Louis, Jan. 01, 1960. https://fred.stlouisfed.org/series/IRLTLT01GBM156N0

[38] "Producer Price Index for All Commodities," FRED, Federal Reserve Bank of St. Louis, Jan. 01, 1913. https://fred.stlouisfed.org/series/PPIACO0

[39] "Producer Prices Index: Economic Activities: Domestic for United Kingdom," FRED, Federal Reserve Bank of St. Louis, Jan. 01, 1960. https://fred.stlouisfed.org/series/GBRPPDMQINMEI0

[40] Pandas, "Python Data Analysis Library," Pydata.org, 2018. https://pandas.pydata.org/

[41] Numpy, "NumPy," Numpy.org, 2005. https://numpy.org/

[42] Scikit-learn, "scikit-learn: machine learning in Python," Scikit-learn.org, 2010. https://scikit-learn.org/stable/

[43] SciPy, Scipy.org. https://scipy.org/

[44] "StatsModels,"://www.statsmodels.org/stable/index.html

[45] R. Bassi, "Forecasting Macroeconomic Data with Neural Networks," Dissertation, University of York, 2017.

[46] N. Bakhashwain and A. Sagheer, "Online Tuning of Hyperparameters in Deep LSTM for Time Series Applications," International Journal of Intelligent Engineering and Systems, vol. 14, no. 1, pp. 212–220, Feb. 2021, doi: https://doi.org/10.22266/ijies2021.0228.21.

[47] W. Griffiths, "Macroeconomic Parameter Prediction: Leveraging Recur-

rent Neural Networks for US CPI Forecasting," Dissertation, University of York, 2023.

[48] K. Hornik, "Approximation capabilities of multilayer feedforward networks," Neural Networks, vol. 4, no. 2, pp. 251–257, 1991, doi: https://doi.org/10.1016/0893-6080(91)90009-t.

[49] "Intuition of Adam Optimizer," GeeksforGeeks, Oct. 22, 2020. https://www.geeksforgeeks.org/intuition-of-adam-optimizer/

[50] S. Shah, "Consumer Price Index (CPI) vs. Producer Price Index (PPI): What's the Difference?," Investopedia, Aug. 23, 2023. https://www.investopedia.com/ask/answers/011915/what-difference-between-consumer-price-index-cpi-and-producer-price-index-ppi.asp: :text=CPI%20and%20PPI%3F-

[51] S. Ronaghan, "Statistical Tests for Comparing Machine Learning and Baseline Performance," Medium, Mar. 14, 2019. https://towardsdatascience.com/statistical-tests-for-comparing-machine-learning-and-baseline-performance-4dfc9402e46f