N-BEATS DEEP LEARNING TRANSFORMER MODEL FOR NOWCASTING CONSUMER PRICE INDEX

BY

JULIUS M. MAINA

MASTER OF SCIENCE IN DATA ANALYTICS

KCA UNIVERSITY

2024

N-BEATS DEEP LEARNING TRANSFORMER MODEL FOR NOWCASTING CONSUMER PRICE INDEX

BY

JULIUS M. MAINA

A RESEARCH DISSERTATION SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF DEGREE OF MASTER OF SCIENCE IN DATA ANALYTICS TO THE SCHOOL OF TECHNOLOGY AT KCA UNIVERSITY

OCTOBER 2024

DECLARATION

I declare that this research dissertation is my original work and has not been previously published or submitted elsewhere for award of a master's degree. I also declare that this contains no material written or published by other people except where due reference is made, and author duly acknowledged.

Student Name: JULIUS MWANGI MAINA Reg No: 22/00243

Sign: _____ Date: 11/10/2024

I do hereby confirm that I have examined the master's dissertation of

JULIUS MWANGI MAINA

And have certified that all revisions that the dissertation panel and examiners recommended have been adequately addressed.

X _____ Date: 11/10/2024

Dr. Lucy Waruguru Mburu
mburul@kcau.ac.ke

Dr. Lucy Waruguru

N-BEATS DEEP LEARNING TRANSFORMER MODEL FOR NOWCASTING CONSUMER PRICE INDEX

ABSTRACT

Accurate modelling of time-series data is vital across various domains, particularly in economic forecasting, such as predicting inflation rates. With inflation data typically released monthly, the limited number of observations poses a challenge for traditional modelling techniques. This study explores the applicability of the Neural Basis Expansion Analysis for Interpretable Time Series Forecasting (N-BEATS) transformer architecture to predict the Consumer Price Index (CPI). Transformers, commonly pre-trained on extensive datasets, offer promising capabilities for fine-tuning to specific tasks, even with limited data. In this research, we aim to replicate the N-BEATS transformer model architecture, utilizing monthly CPI data from the Kenya National Bureau of Statistics (KNBS). The analysis includes exploratory data analysis (EDA) to uncover patterns and trends, followed by model evaluation using Root Mean Squared Error (RMSE) and Mean Squared Error (MSE). This research endeavours to provide an alternative approach for inflation predictions to conventional deep learning and the traditional statistical modelling methods.

ACKNOWLEDGEMENT

I express my gratitude to the Almighty for granting me the wisdom, courage, and dedication to carry out this project. I extend my heartfelt appreciation to my family for their unwavering support and patience while I dedicated a significant amount of time to complete this project.

LIST OF DEFINITIONS

Consumer Price Index (CPI): This is a proxy measure used to gauge the inflation in an economy.

Nowcasting: This is a forecasting approach where the aim is to predict the near future values of a certain target variable without much information.

Deep learning: A sub-field of artificial intelligence that uses artificial neural networks to learn previous patterns and make future predictions.

Temporal Convolutional Network (TCN): A type of Convolutional Neural Network (CNN) that use one dimensional convolutions instead and is designed to recognize patterns in sequential data, such as time series, natural language, and audio signal

Recurrent neural network (RNN): a category of deep learning artificial neural networks that is designed to process sequential data, such as time series or natural language processing data in order to generate patterns and insight.

Long Short-Term Memory (LSTM): LSTM is a type of recurrent neural network architecture that addresses the vanishing gradient problem and is capable of learning and remembering long-term dependencies in sequential data.

Gated Recurrent Unit (GRU): GRU is a variant of recurrent neural networks featuring fewer parameters than LSTM, yet equipped with gating mechanisms for regulating information flow, making it efficient for sequential data processing.

Transformers: Transformers are powerful models that learn relationships between words in sentences without needing to process them in order.

Encoder: The encoder takes in information and condenses it into a simpler form.

Decoder: The decoder takes condensed information and turns it back into a more understandable format.

TABLE OF CONTENTS

Contents **Page**

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS AND ABBREVIATIONS

TCN	Temporal Convolutional Networks
CPI	Consumer Price Index
EDA	Exploratory Data Analysis
GRU	Gated Recurrent Unit
KNBS	Kenya National Bureau of Statistics
LSTM	Long Short-Term Memory
MLP	Multi-Layer Perceptron
MAE	Mean Absolute Error
RMSE	Root Mean Squared Error
RNN	Recurrent Neural network
N-BEATS	Neural Basis Expansion Analysis for Interpretable Time Series Forecasting

CHAPTER ONE

INTRODUCTION

1.1 Background

Economic variables such as inflation are very important as they show the current state of the economy and guide decision-making by policymakers such as central banks and governments. Inflation has been defined as the continuous increase in the general level of prices in an economy (Kara & Keskin, 2021), which reduces the purchasing power of money. The two main proxies for gauging inflation levels are the Producer Price Index (PPI) and the Consumer Price Index (CPI). The PPI proxy shows the price changes in the markets of the goods used in production activities, while the CPI shows the changes in the prices of the goods and services used by consumers.

Forecasting inflation, particularly using the CPI, is crucial for anticipating economic conditions and implementing timely policy interventions. Raj et al. (2024) discussed the current economic crisis in Sri Lanka, which began in 2019 and escalated in 2022.They underscored the importance of accurate inflation forecasting. The crisis was marked by acute shortages of food, fuel, and other essentials, soaring inflation, prolonged power outages, and a collapsing economy with few job opportunities. These conditions led to significant unrest and highlighted the severe impact of unexpected inflation levels on a nation's stability. The primary driver of Sri Lanka's financial collapse was inflation, exacerbated by factors such as tax cuts, money printing, and external shocks like the COVID-19 pandemic. Accurate forecasting of inflation rates, as demonstrated by the use of time series forecasting deep learning algorithms in predicting Sri Lanka's inflation for the next six months (Raj et al., 2024), could have provided critical insights for policymakers. Such forecasts acted upon

would have enabled the government to take proactive measures to mitigate economic crises, ensuring better preparedness and more effective responses to inflationary pressures.

Inflation control is therefore a critical process as inflation affects the spending habits and economic growth in an economy. Governments, through central banks, control inflation through monetary policy that control interest rates in an economy. Monetary policy encompasses a toolkit employed by a country's central bank or governmental authority to shape economic growth through the adjustment of the money supply. This involves strategic control over the supply of money and credit, coupled with modifications to interest rates, aimed at expanding or constraining economic activity within that specific region. These policies guide actions taken by the central bank to influence the availability and cost of money and credit, which affect aggregate demand and inflation. Other strategies governments use to control inflation include government spending and taxation.

PPI and CPI numbers are normally released with a periodicity of one month in most economies. There is always a delay in releasing these monthly numbers. For example, the CPI data for a specific month, such as February, is not available until the end of the following month, in this case, March. This delay highlights the importance of accurate nowcasting techniques to estimate inflation in an economy. Nowcasting involves predicting current and immediate future economic variables, even when data is scarce or delayed (Bańbura, Giannone, & Reichlin, 2010).

The focus of this research is to explore the applicability of transformers for nowcasting CPI. The utilization of traditional statistical methods like Exponential Smoothing and Autoregressive integrated moving average (ARIMA) has already been explored for several decades now. These methods are celebrated for their high accuracy, simplicity, and robustness, offering reliable forecasts, especially in short-term and univariate scenarios

(Hewamalage et al., 2020). However, these statistical methods face limitations related to data quantity and quality for individual time series, potentially missing complex non-linear relationships and cross-series information. Additionally, these methods might be less user-friendly for non-experts due to the need for a-priori assumptions, parameter tuning, and model selection.

Neural networks exhibit exceptional flexibility, universal approximation capabilities, and the ability to leverage cross-series information, making them competitive alternatives (Hewamalage et al., 2020). Deep learning neural networks for timeseries such as the recurrent Neural Networks (RNN) and its variants have been used and have shown good results already in several use cases. This research sought to explore a deep learning transformer architecture. Transformers, unlike RNNs and their variants, can process sequential data in parallel without suffering from vanishing gradients, making them more effective and efficient. Transformers are particularly advantageous for long-term time-series forecasting and are more interpretable than the recurrent networks. Transformers also allow for the use of modern hardware like GPUs and TPUs for parallel processing, which is not as feasible with RNN-based models (Ahmed et al., 2023).

The specific transformer architecture that was explored in this research was the Neural Basis Expansion Analysis for Interpretable Time Series Forecasting (N-BEATS). The N-BEATS model presents an alternative for time series forecasting, owing to its deep neural structure that surpasses conventional statistical approaches while avoiding dependence on time-series-specific elements. Its interpretability (Kumari & Singh, 2022) mirrors that of traditional decomposition techniques, rendering it applicable across various domains compared to the recurrent networks and other machine learning approaches. Furthermore, its swift training duration and exceptional performance on established datasets position it as an

appealing choice for researchers and practitioners seeking precise and expedient forecasting solutions Challu et al. (2023).

1.2 Problem Statement

The demand for timely information on current economic activity, especially during periods of economic shock like pandemics or wars, underscores the critical need for accurate nowcasting methods. Nowcasting, which provides predictions of economic conditions in the present or near future, has become increasingly vital for making quick and informed decisions. However, existing information often suffers from considerable delays, prompting the exploration of nowcasting techniques to bridge this gap (Jamil 2021). The COVID-19 pandemic has accentuated the necessity for real-time decision-making under high uncertainty, emphasizing the urgency of nowcasting methods to forecast economic trajectories (Stundziene et al., 2023).

In this context, our research aims to utilize transformer-based models to accurately predict economic variables, focusing specifically on inflation, using Consumer Price Index (CPI) data. By leveraging advanced deep learning techniques, particularly transformers, we seek to address the limitations of traditional econometric and machine learning approaches and enhance the timeliness and accuracy of economic forecasts. Traditional time series methods are proving insufficient to meet the demand for real-time insights, necessitating exploration of alternatives (Zahara et al, 2019). This study presents a novel approach to nowcasting economic activity by integrating state-of-the-art machine learning methodologies with real-time economic indicators. By systematically reviewing existing literature and identifying gaps in nowcasting research, we aim to contribute to the advancement of nowcasting methods and provide valuable insights for policymakers and economists.

Through our analysis of transformer-based methods of nowcasting, we endeavor to pave the way for future research directions in the field of economic forecasting.

1.3 Research Objectives

1.3.1 General objective:

The primary objective of this study was to adapt the N-BEATS deep learning transformer architecture for nowcasting CPI numbers using past series of CPI data. This was accomplished through the specific objectives that have been listed below.

1.3.2 Specific objectives

i. To Examine the historical patterns in monthly CPI values.
ii. To design and develop a robust N-BEATS model capable of accurately capturing observed historical patterns in monthly CPI values.
iii. To test and validate the developed model.

In addressing the above specific objectives, the current research examined the following questions in detail:

1.3.3 Research questions

a) Are there noticeable trends or cyclical patterns in the monthly CPI numbers?

b) How can N-BEATS be designed and developed to best capture the observed historical patterns in CPI values?

c) Can the developed model reliably predict the monthly CPI values for the current month?

1.4 Motivation of the Study

The study endeavored to transform the landscape of inflation prediction by harnessing the capabilities of transformer-based architectures, a paradigm shift in deep learning methodologies. Amidst the backdrop of dynamic economic landscapes and the necessity for accurate inflation forecasts, the surge in availability of economic indicators, such as consumer price index (CPI) data, underscores the demand for sophisticated predictive frameworks adept at discerning intricate trends and interrelationships. While traditional methods like statistical methods and machine learning approaches have been prevalent, deep learning transformer architectures present a compelling alternative due to their ability to handle long-term dependencies, parallelize computation, and accommodate variable-length inputs. By adapting transformer architectures to CPI data analysis, the research aimed to revolutionize inflation forecasting, paving way for an alternative approach and a more accurate and insightful predictions of CPI data. This endeavor aligns with the quest for economic stability and informed decision-making in financial markets.

1.5 Significance of the Study

Precise forecasting of the Consumer Price Index (CPI) carries substantial weight in the realm of policymaking, particularly for entities like central banks. This accuracy is pivotal as it equips policymakers with the necessary information to implement strategic and effective adjustments, thereby fostering enhanced economic performance and societal well-being.

Moreover, the importance of timely CPI predictions becomes evident in their ability to counteract the challenge posed by delayed CPI announcements. By providing accurate forecasts in a timely manner, these predictions contribute to the overall efficiency of inflation management.

Furthermore, extending the benefits beyond policymaking, timely CPI predictions have a direct impact on consumers. Offering insights into inflationary expectations, these forecasts empower individuals to make informed decisions and take proactive measures against the potential rise in living expenses. This dual significance—catering to both policymakers and consumers—highlights the far-reaching implications and crucial role that accurate and timely CPI predictions play in shaping economic landscapes and individual financial planning.

In addition to its practical implications, this study offers a significant theoretical contribution to the field of inflation prediction using CPI data. By adopting transformer-based architectures, the research expands the range of deep learning methodologies applied to economic forecasting, particularly in the context of time-series analysis. This theoretical advancement not only showcases the versatility and adaptability of transformer models but also underscores their potential to revolutionize traditional approaches to economic modeling. Furthermore, by exploring the efficacy of transformer architectures in capturing complex temporal dependencies and nonlinear patterns inherent in CPI data, this study enriches our theoretical understanding of deep learning's applicability to macroeconomic forecasting. Consequently, the insights gained from this research contribute to the ongoing dialogue surrounding the evolution of predictive modeling techniques in economics, fostering innovation and progress in the field.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

This section presents an in-depth exploration of deep learning methodologies in inflation prediction. It highlights the evolution of deep learning architectures, including Temporal Convolutional Networks (TCNs), Recurrent Neural Networks (RNNs), Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), and Transformer models. Through empirical evidence, the effectiveness of deep learning in addressing conventional econometric challenges such as data scarcity, nonlinearity, and uncertainty is demonstrated. Moreover, comparative analyses with traditional econometric methods underscore the superior predictive capabilities of deep learning models, marking a significant advancement in the field. The section also discusses practical applications of these models in forecasting inflation, showcasing their adaptability and performance across various economic contexts.

2.2 Traditional Statistical Methods of forecasting

Mei (2022) in an effort to come up with a robust prediction model for inflation forecasting using CPI data in China found out that LSTM performed well compared to the ARIMA model. However, he also discovered that the performance of the two models differed across various years, indicating that each model had its own strengths and weaknesses depending on the particular time frame.

Riofríoa et al (2020) did a comparative study of predictive models using Ecuador's CPI data. The study evaluated different predictive models including SVR, LSTM Neural Network, SARIMA, Exponential Smoothing, and Prophet, and found that the SVR model with the RBF kernel and the LSTM Neural Network had the lowest Mean Absolute

Percentage Error (MAPE).The study also included the effectiveness of Seasonal Autoregressive Integrated Moving Average (SARIMA), Exponential Smoothing, and Prophet models in forecasting the monthly values of the CPI for Ecuador during the specified period, although with varying degrees of accuracy compared to SVR and LSTM models.

Nyoni, T. (2019) attempted to predict inflation in Kenya using ARIMA and GARCH (Generalized Autoregressive Conditional Heteroskedasticity) both of which are statistical methods. The choice of ARIMA and GARCH models as the primary analysis methods may have limitations, as there could be other modeling techniques that could provide better accuracy in forecasting inflation in Kenya. Comparative evaluation of SARIMA and Holt-Winters Triple Exponential Smoothing, for predicting inflation in Kenya using Consumer Price Index (CPI) data has also been previously explored.

2.3 Deep Learning Architectures

Deep learning is a specialized branch of machine learning used to construct models for various applications. Neural network architectures containing three or more layers typically fall under the category of deep learning networks. Deep learning architectures imitates how humans process incoming data, create knowledge and use that knowledge to make decisions. This is a field that has seen exponential growth over the last few years. While the concept and experiments have existed over a long time, recent advances in large scale data processing and high performing Graphical Processing Units (GPUs) have spurred deep learning's wide adoption.

Kim, (2020) applied deep learning methodologies for macroeconomic and financial market analyses, utilizing Korean data as a case study. He argued that deep learning offers solutions to conventional econometric challenges, specifically addressing issues like data scarcity, nonlinearity, and uncertainty. Through empirical evidence, his work demonstrated

that deep learning surpasses traditional methods in prediction accuracy and provides more insightful measures of uncertainty, marking a significant advancement in the field.

Deep learning has witnessed significant evolution, resulting in diverse architectures tailored to specific use cases. Temporal Convolutional Networks (TCNs) are widely employed for image recognition, whereas Recurrent Neural Networks (RNNs) excel in modeling sequential data. RNNs come in various flavors, including the Gated Recurrent Unit (GRU) and Long Short-Term Memory (LSTM). Additionally, other deep learning architectures suitable for time series applications include the Multi-Layer Perceptron (MLP) and Transformer models.

2.3.1 Multi - layer perceptron (MLP)

This is a neural network (NN) that consists of an input layer, at least one hidden layer and an output layer. They are fully connected meaning that every neuron is attached to each neuron in the next layer. MLP allow backpropagation to adjust learning weights between neurons and help improve learning of these models.

In their investigation, Ghosh and Bhargava (2019) aimed to assess the predictive capabilities of basic neural network (NN) models in forecasting inflation in India. They conducted a comparative analysis with univariate autoregressive (AR) models, utilizing monthly Consumer Price Index (CPI) inflation data spanning from 2012 to 2017. The findings revealed that NN models demonstrated superior performance over AR models for short-term horizons (one and two quarters), attributed to their adaptability and the early stopping mechanism that mitigated overfitting. However, for longer forecasting horizons, NN models did not maintain the same level of superiority.

2.3.3. Temporal convolutional networks (TCN)

Convolutional networks represent a class of neural networks widely employed for analyzing images and other spatially-structured data. These networks utilize distinct layers, including convolutional layers, pooling layers, and fully connected layers, to extract features and execute tasks like classification, segmentation, and detection. Convolutional layers apply filters to input data, yielding feature maps that encode local patterns. These filters are adaptable and learnable during training, serving as detectors for edges, shapes, textures, and similar features. Pooling layers, on the other hand, perform down-sampling operations, like max-pooling or average-pooling, to reduce feature map complexity and dimensionality. They contribute to the network's ability to maintain robustness against minor translations and distortions in input data. Finally, fully connected layers establish connections between all neurons in consecutive layers, thereby forming a conventional multilayer perceptron. Typically situated towards the network's end, these layers are responsible for executing the final task, such as classification or regression.

Although networks were initially developed for two-dimension image data, they also do have the capability of working on one dimension data for example time-series (Chen & Chen, 2020). Borovykh et al., (2019) evaluated the TCN model on an artificial example (the Lorenz system) and on real-world financial data (S&P500, volatility index, interest rate, and exchange rates). They showed that the TCN model can learn both linear and non-linear dependencies and outperform linear and recurrent models. They explained how the TCN can learn temporal dependencies in and between time series using dilated convolutions, residual connections, and conditioning.

2.3.4 Recurrent neural networks (RNNs)

RNNs are also called sequential models as they are primarily used to model patterns in sequences of data. They have the ability to model relationships across time unlike basic deep learning architectures. Basic deep learning architectures such as MLP do not consider time as a factor. They simply take a set of inputs, run through the model and predict output. The input happen at a point in time and outputs are generated from these inputs at that time. These models therefore do not consider what happened in the past and what will happen in the future.

RNNs predict the future values on a certain sequence based on the past patterns in the sequence. They have the ability to capture information about the past and store it in memory (Du et al., 2022). They will then use this memory to predict future occurrences. This way, they are capable of modelling seasonality directly if the series in the dataset possess homogeneous seasonal patterns, otherwise a deseasonalization step is recommended. RNNs are competitive alternatives to statistical methods in many situations, but they require careful tuning and preprocessing. RNN also face the problem of vanishing gradient where the most current inputs are given more weight compared to previous ones. For this reason, the RNN cells are not able to carry longerm dependencies in the data necessitating use of other RNN variants rather than the vanilla RNN model (Hewamalage et al., 2020).

2.3.5 Gated recurrent unit (GRU)

This is a variant of the conventional RNN that can handle reasonably sized sequences by prolonging memory of certain time steps therefore taking care of long running dependencies. It was first proposed in 2014 and was referred to as gated recursive Convolutional Neural Network (grConv) back then. Empirical studies on LSTM and GRU has shown that GRU converges faster than the LSTM.

GRU will calculate an additional update gate at each time step that produces a value of 0 or 1. If the value is 0, then the current input is ignored and the previous hidden state becomes the current hidden state. Conversely, if the value is 1, then the previous hidden state is fully ignored and a new hidden state is computed entirely based on the input at that time step. The update gate therefore decides if the previous hidden state needs to be fully passed on or fully ignored.

However, when the time step is significant, GRU models tend to face vanishing gradient too. The big timestep makes the gradient become too small or too big affecting the model's ability to learn long-term dependencies.

2.3.6 Long short-term memory (LSTM)

Another popular architecture in RNNs that addresses the vanishing gradient problem is LSTM. According to Schimunek et al., (2021), the LSTM architecture can learn to bridge long time lags of more than 1000 steps, without losing the ability to handle short time lags. LSTM manages to achieve this by using multiple gates to decide if the previous hidden state needs to be passed on or ignored. It therefore ensures long term memory for certain features that are from the previous time-steps.

Paranhos (2023) compared the LSTM model with other neural networks, machine learning models, and traditional benchmarks, using United States of America (USA) CPI. The results showed that the LSTM model performed well at long horizons and during periods of instability, and that the LSTM factors were highly correlated with business cycle indicators.

Kaushik and Giri (2019) conducted a study aiming to predict the USD/INR exchange rate through the utilization of three multivariate forecasting models: Vector Autoregressive

(VAR), Support Vector Machine (SVM), and Long Short-Term Memory (LSTM). Their investigation, based on monthly historical data spanning from April 1994 to December 2019 for the USA and India, incorporated various macroeconomic indicators such as the Consumer Price Index (CPI), Index of Industrial Production (IIP), interest rates, money supply, total reserves, stock market index, and trade-related data. The results revealed that LSTM demonstrated superior performance compared to both the VAR statistical method and the SVM machine learning model in terms of accuracy and error metrics. Specifically, LSTM achieved an accuracy rate of 97.83% and a Mean Absolute Percentage Error (MAPE) of 0.0217.

In recent literature, there has been a surge in the development of hybrid models integrating Long Short-Term Memory (LSTM) networks. These models, which combine LSTM with other techniques, have demonstrated significant advancements in performance. Notably, Bakhit, Nderu, and Ngunyi (2024) showcased the effectiveness of LSTM hybrid Bidirectional Long Short Term Memory (BiLSTM), illustrating their capability to enhance predictive accuracy and deliver promising outcomes when combined with the RNN.

Unlike GRU, LSTM is able to use the previous hidden state as well as the current input to calculate the next hidden state. LSTM calculates the candidate hidden state hc_t and an update gate Gu_t just like the GRU. This update gate is used to filter the candidate hidden state. In addition, a forward gate -Gu_f is computed. Its formula is the same as that of Gu_t except that it has its own weights and biases. This gate is used to filter the previous hidden state. Another gate- output gate (Gu_o) is then computed again with its own weights and biases. This gate is important in that it is used to weigh the resulting outcome to produce the final hidden state (h_t)

The ability to capture long time lags however comes with the challenge of having high computational overhead. LSTM may require a large number of weights and memory cells to store precise, real values for these very long time periods (Schimunek et al., 2021). This model may therefore demand costly infrastructure to set up and pose challenges with long time frame datasets as well as hamper scaling efforts.

2.3.7 Transformer architectures

Transformer-based-architectures have become very popular in the Artificial Intelligence (AI) world. They are already being used in production with good results and in most cases as pre-trained models (Wolf et al., 2020).

Transformers use attention mechanisms only to model sequential data, such as time series. Compared to recurrent or convolutional models, transformers can capture long-term dependencies, parallelize computation, and handle variable-length inputs and outputs (Benidis et al., 2022). Transformers have been successful in natural language processing, speech recognition, and generative modeling tasks. It has also been adapted for time series forecasting by modifying the attention layers and incorporating temporal information.

A transformer model consists of an encoder and a decoder, each composed of a stack of self-attention and feed-forward layers. The encoder encodes the input sequence into a sequence of continuous representations, and the decoder generates the output sequence from the encoder output and the previous outputs (Vaswani et al., 2017). A Transformer relies on positional encodings to inject information about the order of the symbols in the sequence, since it does not use recurrence or convolution. The positional encodings are added to the input and output embeddings and have the same dimension as the embeddings. A transformer uses multi-head attention to jointly attend to information from different representation subspaces at different positions. Multi-head attention consists of several attention heads that

perform attention in parallel and then concatenate and project the results. A transformer employs residual connections and layer normalization around each sub-layer in the encoder and decoder, as well as dropout and label smoothing for regularization.

A few researches have been undertaken seeking to apply transformers for timeseries use cases. Application of transformers for time-series use cases is still under research and no much breakthrough has been achieved to this day. There are even lesser researches aiming to apply this deep learning approach for inflation prediction. For instance, there is no research attempting to use transformers for inflation prediction in Kenya at the time of writing this work.

This notwithstanding, Solís & Calvo-Valverde, 2023, in their work found out that deep learning models with transformers tend to perform better than deep learning models without transfer learning and other machine learning models in the context of time-series data. They also found out that the statistical models still provide better results in most cases, although their performance is mainly similar to that of transformers. Their work offers a promise that transformer architecture could be the next breakthrough in time series prediction since it can simplify the generation of forecasts as, with a single pre-trained model on these architectures, several economic variables can be predicted for different countries.

According to Laptev et al. (2019), transfer learning can improve the accuracy and efficiency of time-series forecasting. Their proposed transfer learning method, using a deep LSTM network with a novel loss function that combines regression and reconstruction objectives, achieved better forecasting accuracy than the baseline methods they compared it with, especially when the target data is limited, noisy, or sparse. They also demonstrated that their proposed method could reduce the computational cost of training and inference by using a single model for multiple time-series. They illustrated how their model could extract

general features from time-series data that are transferable to different tasks and domains, such as forecasting, classification, disaggregation, and feature generation.

2.4 Conceptual Framework

The N-BEATS transformer model adopted for this work only works with univariate data. Only monthly CPI data is used as input for the model without considering other independent variables that could have influenced the CPI level. Our dataset therefore has only one measure, the CPI level sequence, collected over time. The previous 6-months' data points to predict the current month's CPI level. The current research has identified the conceptual framework shown in Figure 1.

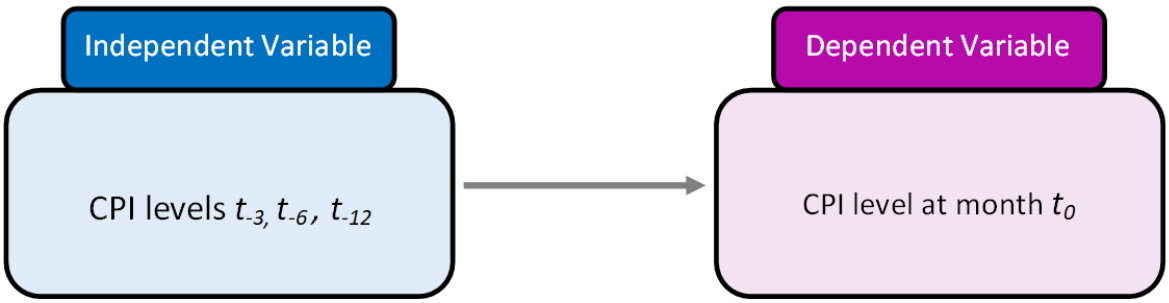


Figure 1 Conceptual Framework

To use the study variables, they need to be in a measurable manner. Table 1 illustrates the indicators and values that will represent each study variable.

Table 1 Operational definition of variables

Variable	**Abbreviation**	**Description**	**Values**
Dependent variable	C_{t-0}	The CPI level for current month, t	Continuous/ number
Independent	C_{t-3}	The CPI data points from the past	Continuous/

variable		three months, *t-3*	number
Independent variable	C_{t-6}	The CPI data points from the past six months, *t-6*	Continuous/ number
Independent variable	C_{t-12}	The CPI data points from the past twelve months, *t-12*	Continuous/ number

2.5 Knowledge Gaps

This study will contribute to the knowledge gap that exists with previous studies that have been concluded on inflation prediction. First is the application of deep learning models in inflation predictions which is quite rare as most studies have concentrated on statistical modelling and other machine learning approaches. Secondly transformer models have not been popularly applied in timeseries prediction. This especially the case of, N-BEATS transformer where only a few research have ventured to show that this is a promising architecture in the timeseries prediction space.

2.6 Conclusion

Based on the review of exiting literature, several gaps were established which have been addressed in this study. Results of the study are expected to help central banks and governments and other institution access timely and accurate forecasts using deep learning transformer models. Furthermore, the findings of this study are expected to contribute to the body of knowledge in the field of time series predictions especially for economic variables.

CHAPTER THREE

METHODOLOGY

3.1 Introduction

This chapter provides a description and justification of the research design, research methods and analysis methods. The case study is described and the techniques for approaching the problem are identified. The testing and validating processes are then described. The chapter concludes with ethical considerations.

3.2 Research Design

To better understand the data for informed modelling, exploratory data analysis (EDA) was carried out. EDA is a group of statistical techniques that aim to explore, describe and summarize the nature of data in order to guarantee its objectivity and interoperability (Garg et al., 2023).

Exploratory Data Analysis (EDA) was carried out for several reasons, each serving a specific purpose in understanding the underlying data. First, EDA was used to confirm the data types and their associated probabilities, ensuring that each variable's structure matched the expected format. This step is crucial to identify any inconsistencies, such as categorical variables incorrectly recorded as numerical, or vice versa, which could affect downstream analysis. Secondly, measures of central tendency, such as the mean, median, and mode, were summarized to gain insights into the data's overall distribution. This helps in understanding the typical values around which data points cluster, giving a general idea of the dataset's behavior.

In addition, graphical illustrations, such as line plots and box plots, were employed to detect trends, seasonality, and potential outliers in the data. Identifying trends and seasonal patterns is particularly important in time series analysis, as it can inform the selection of appropriate models and tuning parameters. Skewness analysis was conducted as part of the normality testing process to assess whether the data followed a normal distribution or was skewed. Skewness, whether positive or negative, can provide insights into potential biases in the data, prompting transformations or adjustments before model implementation.

Furthermore, EDA was used to assess measures of dispersion, including variance and standard deviation, which reveal the extent of variability in the data. Understanding how spread out the data is around the central values is essential for selecting models that are well-suited to the inherent variability in the dataset. The ultimate goal of EDA is to enhance the knowledge of the observed phenomena, providing a foundation for more informed analysis. It also guides the data pre-processing steps that may be necessary, such as handling missing values, normalizing data, or applying transformations to better fit the assumptions of different modeling techniques.

3.2.1 Data collection

The analysis in this study leveraged Consumer Price Index (CPI) data provided by the Kenya National Bureau of Statistics (KNBS) to underpin the research. This dataset spans an extensive period from 1969 to 2023, offering a rich historical context for economic analysis. The year 2019 serves as the base period from which all subsequent CPI values are indexed, allowing for consistent comparison across time. Data is provided on a monthly basis, ensuring detailed granularity, except for the years prior to 1984, where the records were published quarterly. The dataset's breadth, covering 39 years of monthly observations, offers a robust foundation for conducting time series modeling and forecasting. A header sample of

the dataset structure is presented in Table 2, illustrating the format of the information utilized in this analysis.

Table 2 Sample data showing a time series of CPI numbers availed by KNBS

YEAR	MONTH	INDEX
1982	Mar	2.26
1982	Jun	2.28
1982	Sep	2.41
1982	Dec	2.52
1983	Mar	2.56
1983	Jun	2.63
1983	Sep	2.75
1983	Dec	2.92
1984	Jan	2.89
1984	Feb	2.89
1984	Mar	2.94
1984	Apr	2.95
1984	May	2.95
1984	Jun	2.98
1984	Jul	3.05
1984	Aug	3.07
1984	Sep	3.11
1984	Oct	3.12
1984	Nov	3.14
1984	Dec	3.18
1985	Jan	3.20

1985	Feb	3.23
1985	Mar	3.31
1985	Apr	3.31
1985	May	3.33
1985	Jun	3.35
1985	Jul	3.35
1985	Aug	3.39
1985	Sep	3.40
1985	Oct	3.41
1985	Nov	3.41
1985	Dec	3.48

The sample dataset in Table 2 represents Consumer Price Index (CPI) data, spanning from 1982 to 1985, and provides insights into the inflationary trends within this period. The data is organized into three columns:

1. YEAR: This column indicates the year of the CPI measurement, ranging from 1982 to 1985.
2. MONTH: The second column lists the month in which the CPI value was recorded. In the early part of the dataset (1982 and 1983), CPI data is recorded quarterly (March, June, September, and December). Starting from 1984, CPI data is available for every month, offering more granular insights into monthly inflationary changes.
3. INDEX: This column represents the CPI value for each corresponding month. The CPI shows a gradual increase from 2.26 in March 1982 to 3.48 in December 1985, which suggests a steady rise in the cost of goods and services over time.

The full dataset comprised of 564 rows of data as provided by KNBS. The data was then collated and cleaned in the preprocessing steps that have been discussed below.

3.2.2 Data preprocessing

To ensure reliable results and accurate interpretation of the collected data, comprehensive data cleaning and preprocessing procedures were undertaken. This process involved multiple stages, including data cleaning, normalization, and transformation, all aimed at preparing the data for model training and ensuring the integrity of subsequent analyses. A thorough understanding of the data's structure and potential flaws was critical in this step, allowing for the identification of irregularities and ensuring that the data was validated prior to analysis.

Specific preprocessing tasks involved filtering out data points prior to 1984 due to their lower granularity, as these were reported on a quarterly basis rather than monthly. This step was essential to ensure consistency and comparability across the dataset, as time series models rely on uniform time intervals for accurate predictions. By standardizing the data to a monthly frequency starting from 1984, we ensured that the model would be trained on data with consistent temporal resolution, leading to more reliable and precise forecasts. The final dataset had 480 rows available for training and validation.

Another preprocessing step involved partitioning the dataset into training, validation, and testing sets, following an allocation of 80% for training, 10% for validation, and 10% for testing. This approach was crucial for evaluating the model's performance accurately and ensuring its generalizability. The training set enabled the model to learn from historical data, while the validation set facilitated hyperparameter tuning and model selection. Finally, the testing set served as an independent dataset to assess the model's performance and robustness. This structured approach ensured that the model's predictive capabilities could be rigorously evaluated, ultimately contributing to the reliability of the analysis.

The input to the models was the historical monthly CPI data, and the objective was to forecast the CPI for the next one step. As the task involved predicting a continuous quantity rather than a categorical outcome, it was classified as a regression problem. Preparing the univariate time series data for regression-based supervised learning required careful transformation. In supervised learning, the goal is for the model to learn the relationship between input data (X) and the target variable (y). Accordingly, the historical CPI data was transformed into a format suitable for predicting future CPI values, aligning with the study's focus on regression predictive modeling.

$$y = f(X)$$

Different models have different requirements on how the data should be formatted for ingestion. We therefore needed to further prepare our data for ingestion by the various deep learning models we were going to employ. For TCNs and RNNs the data must be transformed from two dimension to three dimension. Our data is already in two dimensions (rows by columns). So, we added a third dimension to the data when working with these deep learning models.

3.2.3 Model development & training

Our forecasting approach focused on generating a single-step prediction into the future, specifically targeting the current month's Consumer Price Index (CPI) values by leveraging past historical sequences (Brownlee, 2019). We conducted experiments using six distinct models: Multilayer Perceptron (MLP), Temporal Convolutional Network (TCN), Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and the N-BEATS transformer model. Each model was evaluated over three different lookback periods: 3 months, 6 months, and 12 months, to assess their performance and adaptability to varying historical input lengths and ensure a comprehensive analysis and

evaluation of the models. This approach allowed us to gauge the models' performance under varying timeframes: the shorter 3-month window, which may be susceptible to overfitting, and the longer 12-month window, which may include irrelevant sequences unrelated to the current context.

To facilitate a fair comparison across all models, an attempt to adopt a set of common hyperparameters was made with only a few added parameters specific to different models. The horizon was consistently set to one, aligning with the single-step prediction requirement. Each model was subjected to a maximum training step limit of 1000 to balance training duration and overfitting risks. In addition, all models utilized the Huber loss function, which adeptly combines squared loss and absolute loss to enhance robustness against outliers that could be prevalent in the CPI data, Desifatma et al. (2024). This was crucial given the potential for erratic economic fluctuations that could skew the predictions.

The N-BEATS architecture, as introduced by the original developers of the model Oreshkin et al., (2020), is depicted in Figure 2 below.

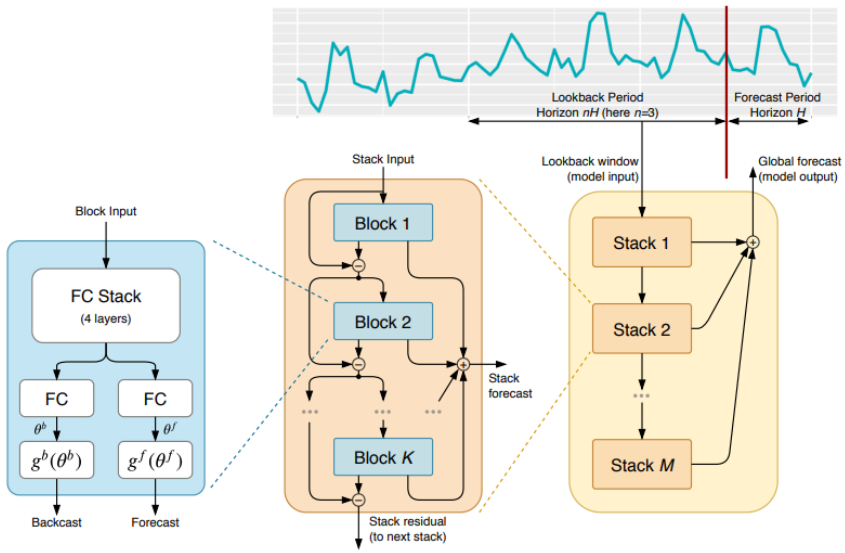


Figure 2: N-BEATS Architecture, Source (Oreshkin et al., 2020)

The N-BEATS (Neural Basis Expansion Analysis for Time Series) model is a flexible architecture and effective in handling a wide range of forecasting challenges. N-BEATS is built using a deep stack of fully connected layers (multi-layer perceptrons) with doubly residual connections, which allow the model to capture complex temporal patterns. The model's architecture is also highly modular and interpretable, as it can decompose a time series into its fundamental components such as seasonality, trend, and noise. This decomposition is particularly valuable when working with limited data, as it regularizes the predictions and helps the model generalize well to unseen data.

The key parameters of the N-BEATS model play a central role in determining its performance. One of the most critical parameters is the forecast horizon, which specifies the number of future time steps that the model is required to predict. In this study, the forecast horizon was set to one, reflecting a focus on short-term forecasting. Another significant parameter is the input size, which defines the number of past time steps used as input to generate forecasts. The input size in this case was varied across different configurations using lookback windows of 3 months, 6 months, and 12 months time steps. This variation enabled an assessment of how the model performs with different amounts of historical data, with the goal of understanding the relationship between the length of the input and the accuracy of the forecasts.

In this implementation, the model was initialized with three types of blocks; identity, seasonality, and trend blocks. The identity block captures the direct, raw values of the time series, while the seasonality block captures recurring patterns using harmonic terms. This component is essential for time series data that exhibits cyclical behavior. The trend block, on the other hand, models long-term changes in the time series using polynomial terms, which helps the model capture gradual upward or downward shifts in the data. These block types are

crucial in providing the model with the flexibility to handle different types of patterns present in the time series.

Several of the model's parameters were left at their default values due to their general effectiveness in most forecasting scenarios. For instance, the number of blocks within each stack of the model, which governs the depth of each stack, was kept at its default setting. Similarly, the dropout probability, which helps prevent overfitting by randomly disabling some units during training, was also left at its default value, ensuring that the model retains a balance between complexity and regularization. The model uses fully connected layers with pre-configured numbers of hidden units and layers within each block, which were also kept at their defaults to maintain the generality of the model.

The N-BEATS model was optimized using the default Adam optimizer, which is widely regarded for its efficiency and stability in neural network training (Llugsi et al., 2024). The learning rate, a key parameter in the optimization process, was set to its default value of 0.001. This value is commonly used as it allows the model to update its weights at a moderate pace, balancing convergence speed and stability. Additionally, the maximum number of training steps was set to 1000, with validation checks conducted at regular intervals to monitor the model's performance on unseen data. Early stopping was employed to prevent overfitting by halting the training process if the validation loss did not improve after a certain number of steps.

Furthermore, padding was enabled in the model, which ensured that all input sequences had a uniform length. This was particularly important in dealing with variable-length time series data, as padding with zeros allowed the model to process shorter sequences without compromising the integrity of the input. This feature was essential in ensuring that

the input size remained consistent across different series, allowing the model to generalize better.

Finally, the batch size for training was set to the default value of 32, which is standard for most neural network models. This batch size ensured a balance between computational efficiency and model performance. The number of windows sampled during each training batch was set to its default of 1024, ensuring that the model received an adequate amount of data for training without overwhelming the computational resources.

The N-BEATS model developed was therefore tailored to this study through the use of varying input sizes and a combination of block types, while many of its core parameters were left at their default values due to their general applicability. By leveraging the flexibility of N-BEATS, the model was able to effectively capture the underlying patterns in the time series data, providing accurate and interpretable forecasts. This setup allowed for an exploration of the model's performance with different input sizes and its capacity to handle both short-term and long-term forecasting challenges.

The other model explored in this research was the Multi-Layer Perceptron (MLP) which is a foundational architecture in the field of artificial neural networks (ANNs), characterized by the presence of hidden layers that facilitate the modelling of complex relationships within data. As a type of feed-forward neural network, MLPs operate by allowing information to flow in a unidirectional manner, from the input layer through the hidden layers to the output layer. Each of the nodes, or neurons, in this architecture functions as a basic computational unit, equipped with an activation function that enables it to produce an output value. The MLPs learn to approximate complex nonlinear functions through the adjustment of weights and biases between these interconnected layers, effectively modelling intricate relationships in the data.

The architecture comprises multiple layers of fully connected nodes, with each node contributing to the overall modelling process by applying a non-linear activation function. This capability is particularly important for time series tasks, where relationships can be intricate and varied. Among the various activation functions, the Rectified Linear Unit (ReLU) has gained prominence due to its effectiveness in mitigating the gradient vanishing issues that often hinder traditional functions like Sigmoid and Tanh. By enabling the model to learn more efficiently, ReLU significantly enhances the MLP's capacity to generalize across different datasets.

The final layer of the MLP is specifically designed to accommodate the autoregressive nature of forecasting, allowing it to generate predictions based on historical observations of the time series. This design choice ensures that the model remains well-suited for time-dependent tasks, effectively translating past patterns into future forecasts. Overall, the MLP's structure and functionality make it one of the most popular research areas in neural computing, illustrating its versatility and effectiveness in addressing complex modelling challenges across various domains (Cinar, 2020).

A central parameter in the MLP model is the forecast horizon, which defines the number of future time steps the model predicts. For this study, the forecast horizon was set to one, indicating that the model produces a one-step-ahead forecast. Another important parameter is the input size, which determines the number of historical time steps used to predict the next value. Different input sizes, corresponding to lookback windows of 3, 6, and 12, were explored to evaluate the impact of input length on model performance. By varying the input size, the study aims to examine how much past information is necessary to provide the most accurate forecast.

The MLP model allows for the inclusion of additional input features, such as exogenous variables. These can be categorized into three types: static exogenous data, historical exogenous data, and future exogenous data. Static exogenous data refers to features that do not change over time, while historical exogenous data consists of time-varying features that are observed in the past, and future exogenous data pertains to known or forecasted values of certain variables in the future. In this work, the MLP model was specifically configured to learn the relationships solely between the target variable, adopting a fully connected approach without incorporating any static, historical, or future exogenous variables. This focused approach allowed for a thorough examination of the inherent patterns within the target variable itself, providing insights into its temporal dynamics without external influences.

The number of layers and hidden units are crucial aspects of the MLP architecture. For this model, the number of hidden layers and units per layer were configured to ensure that the network has sufficient capacity to capture the underlying patterns in the data. The default values for the number of layers and units were employed, as they provide a reasonable trade-off between model complexity and computational efficiency. The choice of the ReLU activation function was retained for its effectiveness in learning deep representations, further enhancing the model's ability to fit non-linear trends and patterns within the time series.

Regarding the model's optimization, the Adam optimizer was used for training, as it is known for its ability to adapt learning rates during the optimization process, ensuring efficient convergence. The learning rate was set to the default value of 0.001, a standard choice that balances the speed of learning with stability during training. The model was trained over a maximum of 1000 steps, with validation checks occurring at regular intervals to assess the model's performance on unseen data. Early stopping was applied to prevent

overfitting, halting the training process if the validation loss did not improve over a predefined number of steps.

Furthermore, the model's batch size was set to the default value of 32, meaning that 32 time series are processed simultaneously during each training step. This choice allows the model to leverage the benefits of mini-batch gradient descent, which combines the advantages of both stochastic and full-batch gradient descent. The number of windows sampled during each training batch was also set to the default of 1024, ensuring that sufficient data is utilized during training without overwhelming computational resources.

Another parameter worth noting is the use of padding. The MLP model was configured with padding enabled, ensuring that all input sequences are of uniform length by adding zeros to the beginning of shorter sequences. This is particularly important when working with time series data of varying lengths, as it allows the model to maintain consistency in the input size and prevents the loss of valuable temporal information. Padding, combined with a step size of one, ensures that the model effectively captures the dynamics of the time series while preserving the temporal order of the data.

The MLP model employed in this study was therefore configured with a combination of default and customized parameters to address the specific needs of the forecasting task. By leveraging the flexible architecture of the MLP, along with its ability to incorporate autoregressive inputs and exogenous variables, the model was able to capture both short-term and long-term dependencies in the time series data. The inclusion of varying input sizes allowed for a comprehensive evaluation of the model's forecasting performance, providing insights into how the amount of historical information impacts the model's ability to generate accurate predictions.

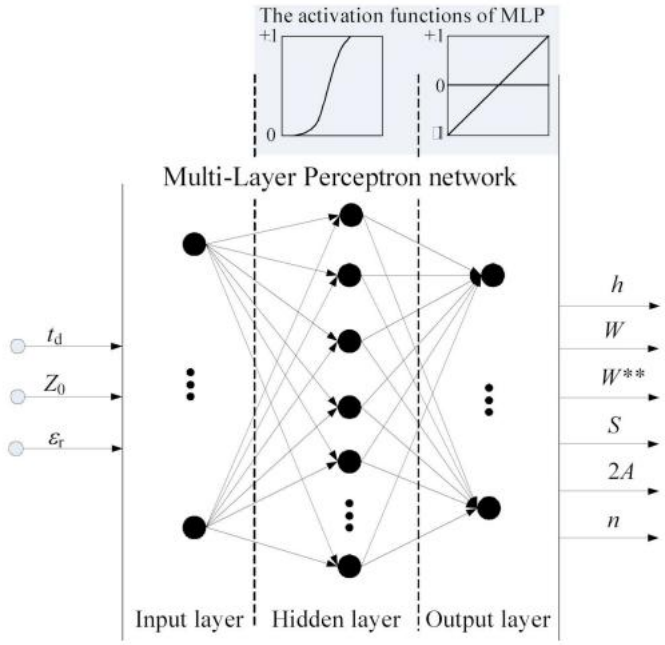


Figure 3: MLP Architecture, Source (Plonis et al., 2019)

Next to be explored was the One-dimensional Temporal Convolutional Network (1D-TCN) that processes multi-channel time series data to extract temporal features through its temporal dimension. This network architecture includes multiple layers of 1D convolution and pooling, followed by a temporal pyramid pooling layer. This innovative layer accommodates variable-length inputs by segmenting the feature maps into various levels and computing the maximum value within each, resulting in a fixed-length feature vector. This vector is then passed through a fully-connected layer designed for classification purposes, enabling the 1D-TCN to effectively learn and classify time-dependent data.

Unlike traditional recurrent neural networks, such as LSTMs, TCNs leverage convolutional layers with dilated skip connections to capture temporal dependencies efficiently. This architecture is particularly effective in capturing long-term dependencies while maintaining computational efficiency, making it well-suited for forecasting tasks that involve large temporal spans. The TCN's ability to bypass sequential processing and operate

on larger time scales through dilated convolutions allows it to maintain memory over extended periods, surpassing the limitations typically associated with recurrent networks.

Once again like in the other models, the model's forecast horizon was set to one, meaning that it predicts one time step into the future. The input size, which refers to the length of the historical sequence used for predictions, was varied during experimentation with lookback windows of 3, 6, and 12, aiming to assess how different amounts of past data affect the model's performance. By experimenting with these different input sizes, the study was able to gauge how much historical information is necessary for the model to generate accurate predictions. For this study, the input size was determined as the product of the lookback window and the forecast horizon.

Key architectural parameters of the TCN model include the kernel size and dilations, both of which control how the temporal data is processed. The kernel size refers to the size of the convolutional filter applied to the time series, and in this case, it was set to 2. A larger kernel size would increase the span of each convolution, but by using dilations, the model is able to extend its effective receptive field without increasing computational complexity. The dilations in this study were set to [1, 2, 4, 8, 16], enabling the model to access increasingly distant time steps in the input sequence by exponentially spacing out the positions where the convolutional kernel is applied. This allows the model to capture patterns over various time scales, which is crucial for forecasting time series data with complex dependencies.

The hidden size of the encoder, which defines the number of units in the hidden layers of the TCN, was set to 128. This parameter controls the capacity of the model to learn from the input data, with larger hidden sizes providing more flexibility but at the potential cost of increased computational demands. The context size, which defines the size of the context vector for each timestamp in the forecasting window, was set to 10. This context vector plays

a key role in transforming the hidden states into future predictions by summarizing relevant historical information for each forecasted point.

The decoder, responsible for producing the final forecasts, consists of a Multi-Layer Perceptron (MLP) with a hidden size of 128 and two layers. The MLP decoder takes the context vectors produced by the TCN and transforms them into the predicted values. The number of layers in the decoder influences the depth of the model's learning capacity, allowing it to learn complex relationships between the historical data and the future forecasts.

The model was optimized using the Adam optimizer, which was left at its default settings due to its adaptive learning rate properties that generally work well in a variety of settings. The learning rate was set to its default value of 0.001, which provides a balance between stability and speed during the training process. The number of training steps was capped at 1000, with the model trained in batches of 32 time series, ensuring a balance between computational efficiency and the amount of data processed at each step. Early stopping was enabled to prevent overfitting, terminating the training process if the validation loss did not improve for a certain number of iterations. The model's performance was evaluated at regular intervals during training through validation checks performed every 100 steps.

Padding was enabled in the TCN model, ensuring that shorter input sequences were zero-padded to maintain a consistent input size across all time series, which is particularly useful when handling sequences of varying lengths. The batch size was set to 32, meaning that 32 different time series were processed simultaneously during each training step. This helps the model generalize better by learning from multiple series at once while also speeding up training.

The TCN model is highly flexible in terms of its ability to incorporate exogenous variables such as static, historical and future variables. However, in this work, the model was specifically configured to learn the relationships solely between the target variable, adopting a fully connected approach without incorporating any static, historical, or future exogenous variables like in the other models. This focused approach allowed for a thorough examination of the inherent patterns within the target variable itself, providing insights into its temporal dynamics without external influences.

Overall, the TCN model, with its dilated convolutions and multi-layer decoder, had the potential to capture both short-term and long-term dependencies in the time series data. The combination of different input sizes and architectural parameters allowed the study to assess the model's ability to forecast under various conditions, providing valuable insights into the importance of historical context and model configuration in time series forecasting.

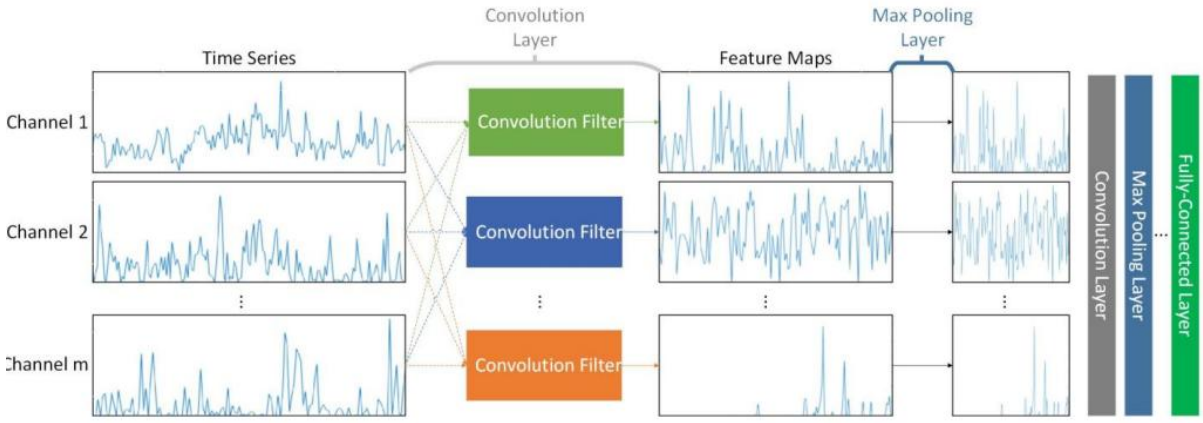


Figure 4: TCN Architecture, Source (Javidani & Mahmoudi-Aznaveh, 2019)

RNNs leverage their hidden state to process sequences efficiently, making them powerful tools for modeling time series, natural language, and other sequential data. Here are the key components as shown in figure 2 below: The RNN's internal memory, represented by ht, captures information from previous time steps. It influences the current output and

allows the network to learn temporal dependencies. At each time step, the input (xt) represents the data for that moment. The output (zt) reflects the RNN cell's prediction or representation at the current time step. It combines the hidden state with the input. Weight matrices (Wi, Wo, Vi) control information flow within the RNN. Biases (bi, bo) allow the network to adjust activations. The sigmoid function (σ) is commonly used for the hidden state. The hyperbolic tangent function (tanh) is used for the output.

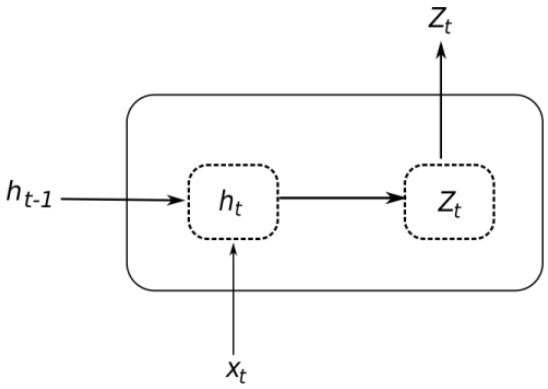


Figure 5: RNN Architecture, Source (Hewamalage et al., 2020)

In this research, the Recurrent Neural Network (RNN) model, specifically based on the Elman architecture, was employed to analyze temporal patterns and forecast future values. The RNN structure, as originally introduced by Jeffrey L. Elman in 1990, is designed to capture dependencies in sequential data by maintaining hidden states that evolve over time. Each hidden state at a given time step is updated based on the current input as well as the hidden state from the previous time step. This dynamic nature allows RNNs to handle temporal dependencies and learn patterns in time-series data, making them suitable for forecasting tasks. The hidden state in this model is updated using a weighted combination of the input data (including target variables and optional exogenous inputs) and the previous hidden state, followed by an activation function, which in this case could either be the hyperbolic tangent (tanh) or the rectified linear unit (ReLU), both available for selection.

A key parameter in the model is the forecast horizon (h), which defines the number of time steps into the future that the model predicts. In this work, the horizon was set to 1, meaning the model forecasts one time step ahead. The input size represents the sequence length used by the model during training, which was determined by the lookback periods, where three distinct lookback periods of 3, 6, and 12 months were tested to evaluate the model's performance under varying historical input lengths. This sequential length allows the model to effectively capture dependencies from the past data to predict future trends. Notably, no static, historical, or future exogenous data were utilized in this setup, meaning the model's predictions were solely based on the target variable itself.

The RNN employed a multi-layer architecture with two layers, where the hidden states are passed through multiple recurrent layers, enhancing the model's ability to learn more complex relationships. Each layer's hidden state size was configured to 128 units, a crucial parameter that determines the dimensionality of the hidden state. A larger hidden state size can allow the model to capture more detailed patterns, but it also increases computational cost and the risk of overfitting. Additionally, the context size, which defines the size of the vector representing the context of each time step within the forecasting window, was set to 10, ensuring that the model captures sufficient temporal context for the prediction task.

The decoder part of the RNN model, responsible for transforming the hidden states into final predictions, was also structured as a multi-layer perceptron (MLP) with two layers, each having 128 hidden units. The MLP decoder plays a critical role in converting the learned representations from the recurrent layers into the final output, i.e., the forecasted values. The optimization of the RNN model was handled by the ADAM optimizer, which is known for its adaptive learning rates and ability to converge quickly, particularly useful in time-series

forecasting problems. The training process involved 1,000 maximum steps, ensuring the model had sufficient iterations to learn the underlying patterns in the data.

Notably, most of the parameters in this work were left at their default settings, allowing the model to follow standard configurations in RNN training. For instance, the activation function used in the RNN layers was the default hyperbolic tangent (tanh), which is commonly used for its ability to handle vanishing gradients better in sequential models. Similarly, the dropout rate, which is typically employed to prevent overfitting by randomly dropping units during training, was left at its default value of zero, indicating that no dropout regularization was applied. Additionally, the biases within the recurrent layers, which help in stabilizing the learning process, were enabled by default.

The loss function chosen for this work was the Huber loss, which is a robust choice for regression tasks as it balances between mean absolute error (MAE) and mean squared error (MSE), making it particularly effective when dealing with outliers in the data. Furthermore, the learning rate was initialized with its default value of 1e-3, which ensures a moderate rate of convergence during training. While the default configurations were employed for many parameters, this ensured that the model remained flexible and capable of generalizing well without the need for extensive tuning.

Through this design, the RNN model had the potential to learn and forecast the target variable's future values, demonstrating its ability to capture temporal dependencies solely from the target variable, as no exogenous inputs were incorporated in this configuration. The experimentation with different lookback periods (3, 6, and 12 months) allowed for a thorough evaluation of how much historical information is necessary to achieve optimal predictive performance.

The Long Short-Term Memory (LSTM) model was also employed in this research and is a powerful tool for time series forecasting. This model builds upon the standard recurrent neural network (RNN) architecture, specifically addressing the issues of vanishing and exploding gradients that often hinder the training of deep networks. By utilizing LSTM cells, which include mechanisms to retain long-term dependencies, the model can effectively capture complex sequential patterns inherent in time series data. The architecture comprises a multilayer LSTM encoder and a Multi-Layer Perceptron (MLP) decoder, where predictions are made by transforming hidden states into contexts. These contexts are then decoded into forecasted values using the MLP.

A typical LSTM unit has 4 components: 1) cell state–C_t, 2) input gate–i_t, 3) output gate–o_t, and 4) forget gate–f_t as depicted in Figure 3. Cell states exchange information with each other. Each cell state will transmit historical information (C_{t-1}) and current updated information (C_t) to the subsequent cell via forget, input and output gates in order to update the information. The forget gate will receive new inputs (x_t) and past hidden states (h_{t-1}) in order to determine the information to be communicated to the cell state. At the same time, the input gate will determine which information should be updated among the various series of new information and will create a new information value ($\bar{C}_t$) using an activation function (tahn). The output information will thereafter identify the information that will form the output. The LSTM equation is as follows:

$$f_t = \sigma\left(W_f \cdot \left[h_{t-1,} x_t\right] + b_f\right),$$

$$i_t = \sigma\left(W_i \cdot \left[h_{t-1,} x_t\right] + b_i\right),$$

$$o_t = \sigma\left(W_o \cdot \left[h_{t-1,} x_t\right] + b_o\right),$$

$$\tilde{C}_t = tahn\left(W_C \cdot \left[h_{t-1,} x_t\right] + b_C\right),$$

$$h_t = o_t \cdot tahn(C_t),$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

where W_f, W_i, W_o and W_C represent the weights of the forget gate, input gate, output gate and cell state respectively (Kim et al., 2022). The structure of the LSTM is as shown in Figure 3.

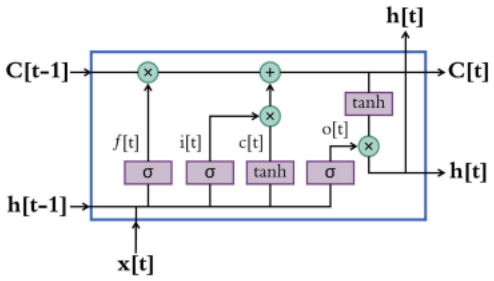


Figure 6: LSTM Architecture, Source (AlKandari & Ahmad, 2019)

The configuration of the LSTM model involves several key parameters that directly influence its performance. In this research, the forecast horizon was set to one, indicating that the model is designed to predict the next immediate value in the sequence. The input size, crucial for determining the maximum sequence length during training, was calculated as the product of the lookback period and the forecast horizon. The lookback periods evaluated were three months, six months, and twelve months, allowing for an investigation into the model's ability to leverage historical information effectively. By default, the input size is set to use all available history, providing the model with the necessary context for its predictions.

In terms of architectural depth, the LSTM encoder was configured with two layers, enabling the network to learn hierarchical features across multiple levels. Each layer consists of a hidden state size of 128 units, which offers a balance between model complexity and computational efficiency. This hidden state serves as a memory component that captures

relevant information from previous time steps, aiding in the model's ability to understand temporal dependencies. While the model was configured to include biases within the LSTM units, dropout regularization was set to zero, indicating that no dropout was applied to the outputs of the LSTM layers during training.

The context size was defined as ten, reflecting the dimension of the context vector produced for each timestamp in the forecasting window. This parameter plays a crucial role in determining how much information is carried forward to the decoder, influencing the quality of the generated predictions. The MLP decoder, which follows the LSTM encoder, also consists of two layers and utilizes a hidden layer size of 128 units. This design enables the model to effectively transform the encoded information into meaningful output values.

During the training process, the model leveraged the Huber loss function, known for its robustness in regression tasks, and is optimized using the Adam stochastic gradient descent algorithm. The maximum number of training steps was set to 1000, ensuring sufficient iterations for the model to learn the underlying patterns in the data. A batch size of 32 was employed, facilitating the processing of multiple time series during each training iteration. Validation checks were performed every 100 steps to monitor the model's performance, with early stopping criteria in place to prevent overfitting by halting training if validation loss did not improve over a defined number of iterations.

It is important to note that, in this study, the LSTM model was configured to learn the relationships solely between the target variable. Consequently, no static, historical, or future exogenous data were utilized in the modeling process. This focused approach enabled a thorough examination of the intrinsic patterns within the target variable itself, allowing the model to capture its temporal dynamics without external influences. By analyzing the effects of different lookback periods, this research aimed to identify the optimal historical context

needed for accurate predictions, further emphasizing the LSTM model's capacity to handle sequential data effectively.

The gated recurrent unit (GRU) serves as a simplified variant of the LSTM architecture, and addresses LSTM's lengthy training duration. GRU features a streamlined design with fewer control gates, notably lacking an output gate. Illustrated in Figure 4, GRU comprises solely two gates - the reset gate and update gate - regulating information flow within the units. The transition functions between GRU neurons are outlined as follows:

$$r(t) = \sigma(w_r x(t) + u_r h(t-1) + b_r)$$
$$z(t) = \sigma(w_z x(t) + u_z h(t-1) + b_z)$$
$$\hat{h}(t) = \sigma(w_h x(t) + u_h (r(t) * h(t-1)) + b_h)$$
$$h(t) = (1 - z(t)) * h(t-1) + z(t) * \hat{h}(t)$$

where r(t) is the reset gate, z(t) is the update gate, and w and u represent the parameter matrices in GRU. Furthermore, h(t), $\hat{h}(t)$ and b are the output, candidate output, and bias, respectively. The activation function is represented as σ.

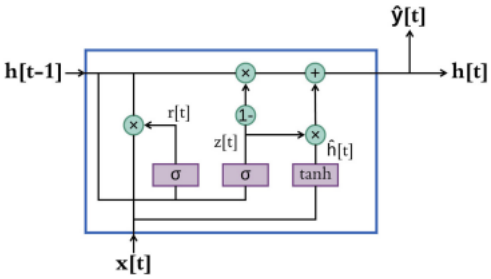


Figure 7: GRU Architecture, Source (AlKandari & Ahmad, 2019)

In this study, the Gated Recurrent Unit (GRU) model was selected to handle time series forecasting tasks, due to its ability to capture sequential dependencies effectively. The GRU, proposed by Cho et al., improves on the limitations of the traditional Recurrent Neural Networks (RNNs) by incorporating gated mechanisms, which control the flow of information

and prevent the vanishing gradient problem often encountered in long sequences. Unlike its counterpart, the Long Short-Term Memory (LSTM), the GRU simplifies the architecture by using fewer gates, specifically, an update and a reset gate, which streamline the learning process while maintaining the capacity to store relevant temporal patterns. This architecture is well-suited for time series analysis, as it enables the model to retain important historical information across varying lookback periods without overwhelming computational complexity.

The GRU model in this work was initialized with specific parameters to perform univariate forecasting, focusing solely on the relationships within the target variable, without incorporating exogenous data, whether static, historical, or future. The model was configured to explore the effects of different historical windows on forecast accuracy, with lookback periods set at 3, 6, and 12 months. This variation allowed for a detailed analysis of how much historical information is necessary for accurate predictions in the time series domain. The input size, which is tied to these lookback periods, determines how many past observations are used as inputs during training and prediction. By adjusting this parameter to match the different lookback windows, the model was effectively tailored to investigate how varying amounts of historical data influence its performance.

One of the key parameters in this GRU implementation was the forecast horizon, which defines the number of time steps into the future that the model is tasked with predicting. In this case, the horizon was set to 1, meaning that the model was designed to make one-step-ahead forecasts, focusing on short-term predictions. The GRU network itself consisted of two layers, controlled by the number of layers parameter, which adds depth to the model, allowing it to learn more complex temporal patterns by passing information through multiple processing stages. The hidden state size for each layer, specified by the

hidden state size parameter, was set to 128 units. This parameter governs the amount of information the model can retain and process at each time step, influencing the model's capacity to learn and represent the temporal dynamics of the data.

The GRU model was further supported by a Multi-Layer Perceptron (MLP) decoder, which transformed the hidden states into the final predictions. The decoder was configured with two layers, each with 128 hidden units, reflecting the need for an expressive architecture that could effectively map the complex hidden representations learned by the GRU into the final forecast values. The context size parameter, set to 10, defined the dimensionality of the context vector passed from the GRU layers to the decoder at each time step. This context vector captures the relevant information from the past that the decoder uses to make accurate predictions for the future time step.

The training process was governed by several key parameters. The maximum number of training steps was set to 1000, allowing the model enough iterations to converge on an optimal solution. The model was trained using the Huber Loss function, a robust loss function that combines the advantages of both the mean squared error and the mean absolute error, making it particularly suitable for time series data that may contain outliers. The choice of this loss function helped in penalizing large errors without overemphasizing small deviations, ensuring a balanced approach to minimizing forecast errors.

In terms of optimization, the default Adam optimizer was employed to update the model’s weights, utilizing adaptive learning rates to enhance the efficiency of the training process. While some hyperparameters, such as the learning rate and batch size, were left at their default settings, the overall architecture of the GRU remained consistent with standard best practices for time series forecasting tasks. For instance, a batch size of 32 was used, which strikes a balance between computational efficiency and model generalization. The

training process did not include dropout regularization, as the dropout rate was left at its default value of zero, reflecting the belief that the model was not overfitting the training data.

The exclusion of exogenous data was a deliberate choice in this study, focusing the model exclusively on the temporal patterns inherent in the target variable. By isolating the target variable and using no static, historical, or future exogenous data, the GRU model was able to concentrate solely on capturing and predicting patterns within the target series itself. This approach provides valuable insights into the capabilities of GRU in pure time series forecasting scenarios, where external factors are not considered, and the model must rely entirely on past observations to make accurate predictions.

Other aspects of the training process explored involved tuning the models' hyperparameters to settle on optimal parameters before training. The model tuning process followed a structured configuration, aimed at optimizing performance through a grid-search tuning strategy designed to minimize prediction errors. Grid-search is a classic hyperparameter optimization technique, systematically explores a predefined subset of the training algorithm's hyperparameter space. Although effective, it encounters challenges in high-dimensional spaces; nevertheless, it lends itself well to parallelization. This method involves training and evaluating every potential combination of network parameters specified by the user (Liashchynskyi & Liashchynskyi, 2019). Oreshkin et al., (2020) described the hyperparameters used for the different variants of N-BEATS.

Central to this process were several key hyperparameters, systematically adjusted across all models. The input size parameter was set to either 3, 6, or 12, indicating that the model would use data from the previous three, six, or twelve months, respectively, to forecast the current month's Consumer Price Index (CPI) values.

The validation check steps parameter was fixed at 50, allowing the model to assess its performance on the validation dataset after every 50 training iterations. This frequent evaluation facilitated timely adjustments during the training process. To ensure reproducibility of the results, a random seed parameter value between 1 and 10 was implemented. The learning rate, a critical parameter influencing the rate of weight updates during training, was varied between 1e-5 and 1e-1, offering flexibility in the model's convergence speed.

To further enhance model stability, the scaler type parameter supported multiple scaling methods, minmax, robust, standard, or no scaling, depending on the dataset's characteristics, ensuring that the input data was properly normalized. Lastly, the max steps parameter was set to 1000, providing the model with sufficient training iterations to learn effectively while preventing overfitting.

By carefully adjusting these hyperparameters using grid-search, the tuning process aimed to improve the predictive accuracy of the models, ensuring robust performance in forecasting CPI values and maintaining consistency for meaningful comparisons.

3.2.6 Model evaluation

During the model evaluation stage, a comparison was made on the prediction results for each of the models against the ground truth numbers. The metrics to be used are mean absolute error (MAE) and the root mean standard error (RMSE).

MAE is a metric used to measure the average absolute differences between predicted and actual values in a dataset. It is a simple and interpretable measure of prediction accuracy. The formula for calculating MAE is as follows:

$$MAE = \frac{\sum_{i=0}^{n} |y_i - \bar{\hat{y}}|}{n}$$

where y_i is the observed value and $\bar{\hat{y}}$ is the predicted value.

The RMSE was used to indicate inherent errors in the predicted and observed values. This metric is particularly important in determining how a model responds to outliers that may exist in the dataset. A lower RMSE value indicates an increase in performance. The RMSE equation is:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=0}^{n}(y_i - \hat{y}_i)^2}$$

where n is the number of observation points, y_i is the observed value and $\hat{y}_i$ is the predicted value.

3.3 Ethical considerations

Research ethics address the moral questions in a scientific or research study. The researcher in this study was honest and accurately reflect the work done using appropriate methods and techniques for data analysis and interpretation. Decision about the research design and the method of presenting results was justified to develop trustworthy research. The study will reveal all the research limitations and challenges in implementing any of the research methods.

The researcher obtained permission to access and utilize data for research endeavors from the designated officers at KNBS. The data was transmitted to the researcher via email and is additionally accessible to the public through the organization's website. Furthermore,

authorization was sought from the National Commission for Science, Technology, and Innovation (NACOSTI).

CHAPTER FOUR

RESULTS AND DISCUSSION

4.1 Introduction

This chapter presents the outcomes of applying the N-BEATS transfer learning architecture to CPI time series data related to inflation. It delves into the insights gained from analysing the data trends and seasonality and explores the potential of the model for accurate forecasting. Through a rigorous analysis of the results and findings, this chapter seeks to contribute to the growing body of economic variables forecasting and inflation modelling research and aid in the design of more targeted interventions, policy formulation, and resource allocation. The subsequent sections of this chapter will provide a detailed account of the observed results, model estimation, validation, and interpretations.

4.2 Descriptive Analysis of Consumer Price Index

The descriptive statistics in Table 4.1 provide valuable insights into the variation in the Consumer Price Index (CPI) across months. This summary allows us to observe central tendencies (such as the mean and median), spread (standard deviation, interquartile range), and the distribution shape (skewness and kurtosis), as well as outlier behavior over the observation period.

Table 3 a concise overview of various descriptive statistics factors related to CPI grouped by month

Month	Mean	Max	Median	Min	SD	Variation	Q1	Q3	IQ	Skew	Kurtosis	Count
1	43.71	129.29	30.03	2.89	37.46	1402.97	14.62	70.52	55.90	0.77	-0.66	40
2	43.97	130.13	29.96	2.89	37.72	1422.95	15.21	70.96	55.75	0.78	-0.66	40
3	44.41	131.18	30.45	2.94	38.02	1445.86	15.55	71.57	56.03	0.78	-0.67	40
4	44.92	131.83	30.86	2.95	38.44	1477.71	15.69	72.28	56.59	0.77	-0.69	40
5	45.31	133.01	32.00	2.95	38.63	1492.59	15.90	72.57	56.66	0.77	-0.68	40
6	45.51	134.01	32.59	2.98	38.63	1492.06	16.14	72.62	56.48	0.77	-0.65	40
7	45.56	134.01	32.20	3.05	38.66	1494.28	16.22	72.81	56.59	0.78	-0.64	40
8	45.75	134.15	32.66	3.07	38.76	1502.03	16.24	73.16	56.92	0.77	-0.65	40
9	45.96	134.02	32.91	3.11	38.86	1510.11	16.25	74.15	57.90	0.77	-0.66	40
10	46.11	135.32	33.14	3.12	39.07	1526.14	16.35	74.08	57.73	0.78	-0.63	40
11	46.34	136.71	33.14	3.14	39.30	1544.22	16.09	74.23	58.14	0.78	-0.62	40
12	43.28	128.99	29.60	0.00	37.37	1396.57	14.02	69.80	55.78	0.77	-0.65	40

Below are key observations from the data:

4.2.1 Central tendency: mean and median

The monthly mean CPI fluctuates between 43.28 (December) and 46.34 (November), showing an overall upward trend in the CPI as the year progresses. This reflects the general increase in price levels over time. Similarly, the median follows a similar trend to the mean, indicating a relatively symmetrical distribution of CPI data across months, with values ranging from 29.60 (December) to 33.14 (November). This is slightly lower than the mean, suggesting a right-skewed distribution where higher CPI values have a small effect on the overall mean.

4.2.2 Range and extremes: maximum and minimum

The highest CPI observed is 136.71 (November), while the lowest maximum is 128.99 (December). This highlights that November experienced the highest price surge in the observed period. On the other hand, the minimum CPI values remain consistently low showing the former years when inflation was low compared to the base period of 2019. December stood out with a minimum value of 0.00 in a certain year, indicating an outlier in that month and year.

4.2.3 Spread and variation: standard deviation and interquartile range (IQR)

Standard deviation ranges from 37.37 (December) to 39.30 (November), indicating that CPI values become slightly more dispersed as the year progresses. This growing deviation could suggest greater volatility or variability in prices toward the end of the year. The IQR (the difference between Q3 and Q1) is another important measure of variability. It increases steadily over the months, showing a broader distribution as the months progress. For instance, in January, the IQR is 55.90 (70.52 - 14.62), whereas by November, it is 58.14 (74.23 - 16.09). This increase indicates higher fluctuations in CPI toward the later months.

4.2.4 Distribution shape: skewness and kurtosis

The skewness for most months hovered around 0.77 to 0.78, indicating a moderate right skew, where larger CPI values pull the mean slightly above the median. This shows that there are a few extreme values in the right tail of the distribution, representing occasional significant price hikes.For the kurtosis, values are slightly negative (ranging from -0.66 to -0.62), suggesting a distribution that is slightly flatter (platykurtic) than a normal distribution. This implies that extreme CPI values (both high and low) occur less frequently than would be expected in a normal distribution.

4.2.5 Count and stability

The period contained 40 observations (40 years since 1984), indicating that the dataset is consistently populated throughout the year, ensuring that monthly comparisons are valid and that trends observed are reliable.

The descriptive statistics of the Consumer Price Index (CPI) and returns from 40 years without grouping by month are depicted in Table 4.4.

Table 4 a concise overview of various descriptive statistics factors related to CPI

	CPI	**Return**
count	480.00	480.00
mean	45.07	0.01
std	37.98	0.05
min	0.00	-1.00
0.25	15.32	0.00
0.5	31.31	0.01
0.75	73.20	0.01
max	136.71	0.10

These results also confirmed the same insights into the underlying inflationary trends as the results when we grouped the data by months. Over this extended period, the CPI data reflected a clear upward trajectory, consistent with the long-term nature of inflation, which generally drives prices upwards over time. The mean CPI of 45.07, while informative as a central tendency, must be considered alongside its standard deviation of 37.98, which highlights substantial variation across time. However, this variability does not suggest erratic

fluctuations but rather captures the natural progression of inflation, which increases steadily but not uniformly, as reflected in the growing CPI values.

The minimum CPI of 0.00 may seem like an anomaly, but it could represent early stages in the data or periods of very low price levels, likely occurring decades ago, when the base level for the CPI was much lower. As time progressed, the steady rise in inflation was evident in the upper quartile and maximum values: the 75th percentile of 73.20 and the maximum CPI of 136.71 showed the continual upward drift in prices. This progression illustrated the cumulative effect of inflation over 40 years, which results in the CPI more than tripling from the lower quartile to the maximum. This supports the narrative that the economy experienced steady inflationary growth, consistent with a trend rather than short-term volatility or shocks.

The distribution of returns also supported this inflationary trend. The mean return of 0.01, or approximately 1%, indicates a gradual yet persistent increase in the CPI over time. This is typical in periods of inflation where prices rise consistently at modest rates. The relatively low standard deviation of 0.05 shows that while returns varied slightly, they remained within a tight range, reflecting consistent growth. The quartiles for returns, which are tightly grouped around 0.01, further reinforce this observation. Most of the return values fall between 0.00 and 0.01, indicating that the majority of changes in CPI were incremental rather than driven by large, sudden jumps. This aligns with the characteristic behavior of inflation, where prices increase in small, compounded steps rather than large shifts.

The extreme values in the returns distribution, such as the minimum return of -1.00 and maximum return of 0.10, should be interpreted in the context of rare, exceptional circumstances. The negative return likely represents a period of deflation or an economic downturn where prices briefly fell, while the maximum positive return of 10% suggests

isolated episodes of accelerated inflation. However, these values are outliers compared to the overall pattern, which consistently shows small positive returns.

4.3 Graphical Depiction of the Trends in Data

Figure 4.1 shows a line chart depicting the monthly Consumer Price Index (CPI) for the Kenyan economy from 2010 to 2023. It reveals a consistent upward trend, indicating sustained inflationary pressures over this period. The CPI values, which start around 60 in 2010 and rise above 120 by 2023, suggest that the cost of living in Kenya has more than doubled in just over a decade. This persistent increase in CPI reflects the continuous rise in the prices of goods and services, pointing towards underlying economic factors such as currency depreciation, supply chain disruptions, and possibly expansionary fiscal policies. The absence of significant deflationary periods or sharp fluctuations indicates a relatively stable yet steadily increasing inflation rate, highlighting structural inflation challenges that may require targeted policy interventions to manage. This trend underscores the importance of addressing inflation to ensure economic stability and protect consumer purchasing power in Kenya.

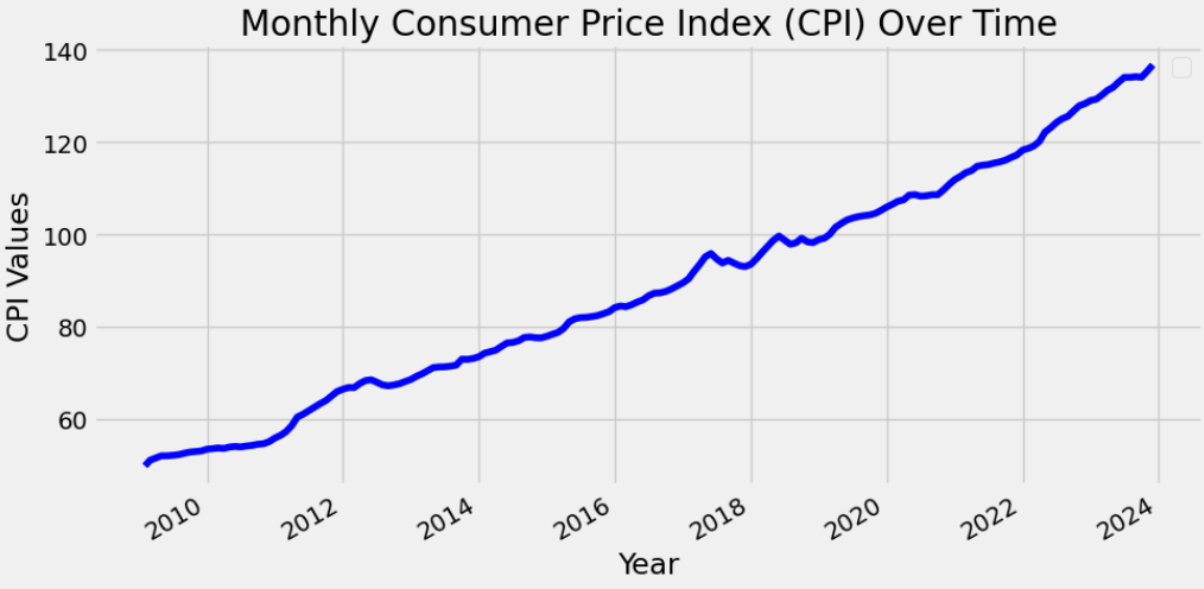


Figure 8:Trend of CPI values across time

Figure 4.1 shows the boxplot of the Consumer Price Index (CPI) data for Kenya from 1984, grouped by month. It reveals several key insights. The interquartile ranges (the boxes) show that the CPI values are relatively stable within each month, with no significant outliers or extreme variations, suggesting a steady inflation rate without erratic spikes. Interestingly, there is no distinct seasonal pattern where certain months consistently exhibit higher or lower CPI values compared to others. This uniformity across months implies that inflationary pressures in Kenya are not significantly influenced by seasonal factors but rather follow a more general economic trend. This stability and lack of seasonal fluctuation can be crucial for policymakers and investors when planning economic strategies and making investment decisions.

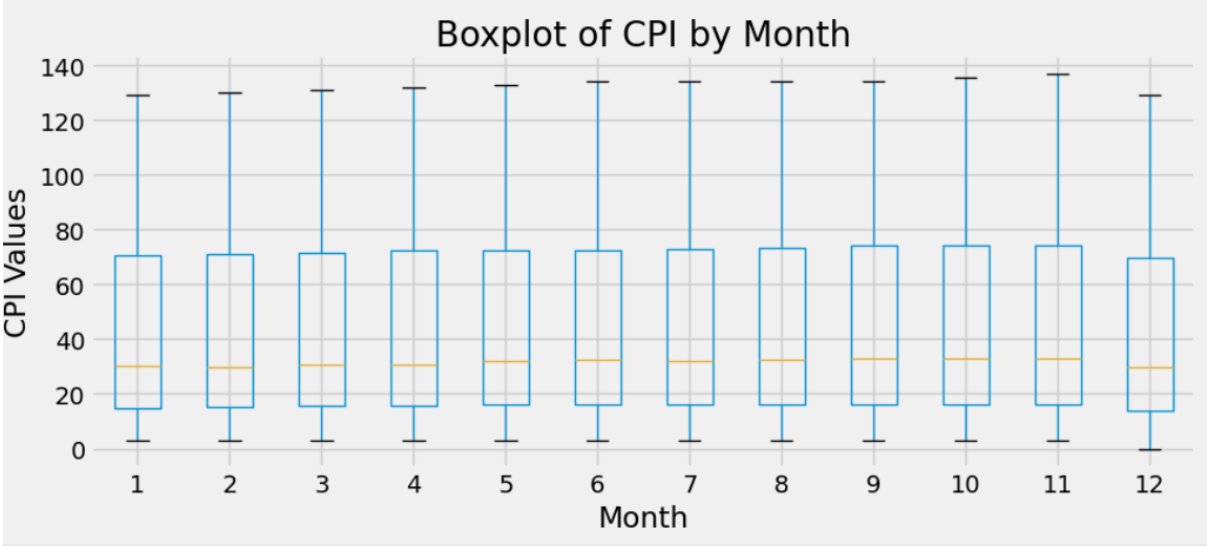


Figure 9: Average distribution of CPI values for different months

4.4 Prediction of CPI Numbers in Testing Data for N-BEATS Model

Figure 4.3 illustrates the performance of the NBEATS deep learning model in predicting values over the test period with a lookback horizon of three months. The actual values, represented by the black line, and the predicted values, shown in blue, closely follow each other, indicating the model's high accuracy.

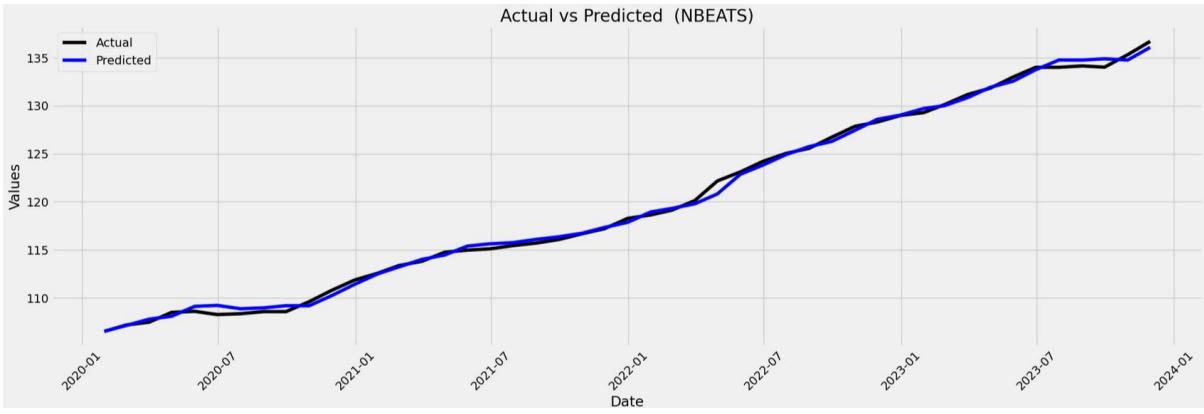


Figure 10: Time series results of the predicted and observed consumption patterns by the N-BEATS model.

Table 4.3 shows the model achieved a Mean Absolute Error (MAE) of 0.3478 and the Root Mean Square Error (RMSE) of 0.4364. This further quantify this accuracy, demonstrating the model's effectiveness in minimizing prediction errors. These metrics highlight the NBEATS model's capability to provide reliable forecasts, making it a valuable tool for time series prediction tasks. The visual overlap of the lines in the graph underscores the model's proficiency in capturing the underlying patterns of the data. The parameters used for the

Table 5 Performance metrics of N-BEATS deep learning model using the testing dataset

MAE	**RMSE**
0.3478	0.223

4.5 Comparison of Untuned NBEATS and other Untuned Deep Learning Models

The NBEATS model was thereafter compared with the other deep learning model using the test dataset using three lookback periods namely; 3 months, 6 months and 12 months to predict next month's CPI. Significant variation in performance depending on the lookback period were observed in some models. Also, significant variation in performance across the models was also observed.

Table 4.4 below shows the performance metric for the untuned versions of the model using the test dataset for three different lookback periods: 3 months, 6 months, and 12 months.

Table 6 Performance metrics of the untuned models using the test dataset for three lookback periods

	3 Months		6 Months		12 Months	
	MAE	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**
MLP	1.558	1.626	*1.254*	*1.338*	1.2530	1.3256
RNN	*8.889*	*10.25*	15.26	15.99	11.86	13.09
LSTM	*2.664*	*2.940*	6.747	7.375	9.338	9.838
GRU	*4.531*	*5.681*	9.930	10.45	14.66	15.44
TCN	4.696	4.741	*2.403*	*2.554*	10.41	10.59
N-BEATS	***0.3478***	***0.4364***	***0.34957***	***0.4411***	**0.3739**	**0.4552**

4.5.1 Short-term performance (3 months)

In the short-term (3 months), the N-BEATS model outperformed all other models with the lowest MAE (0.3478) and RMSE (0.4364). This indicates that N-BEATS is highly effective at capturing short-term trends in the CPI data. On the other hand, the RNN model struggled significantly in the short-term, with the highest MAE (8.889) and RMSE (10.25). This suggests that RNNs, may not be well-suited for short-term CPI predictions.

4.5.2 Medium-term performance (6 months)

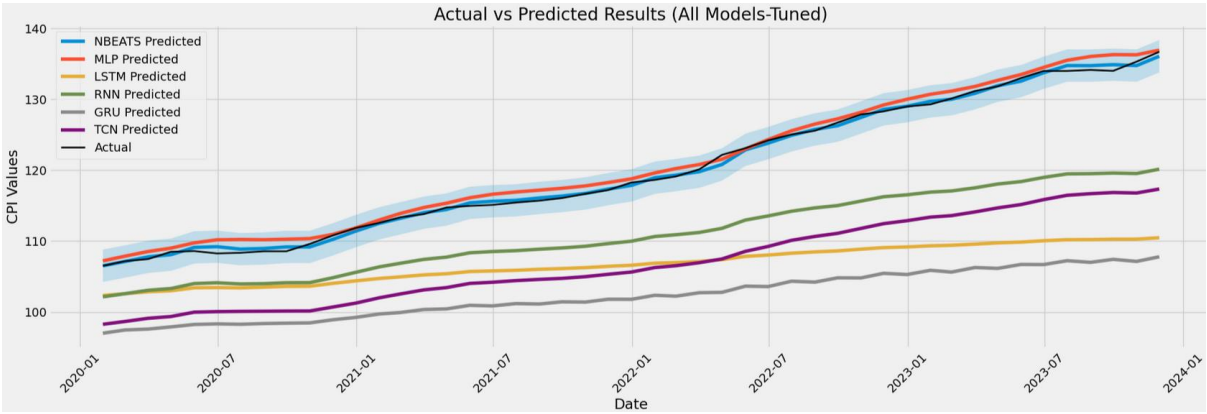
For medium-term predictions (6 months), the N-BEATS model continued to show strong performance with low MAE (0.34957) and RMSE (0.4411). Interestingly, the TCN model also performed well in this period, with MAE (2.403) and RMSE (2.554) significantly lower than its short-term metrics. This improvement suggested that TCNs may be better suited for capturing medium-term trends. The RNN model showed some improvement in the medium term, with reduced MAE (15.26) and RMSE (15.99), although it still lagged behind other models.

4.5.3 Long-term performance (12 months)

In the long-term (12 months), the N-BEATS model remained the best performer with the lowest MAE (0.3739) and RMSE (0.4552). The TCN and GRU model showed notable deterioration in the long-term, with MAE (11.86) and RMSE (13.09) and MAE (14.66) and RMSE (15.44) respectively. These errors were higher than their short-term and medium-term metrics. This indicated that these models may have a better capacity for short-term prediction horizons, for this type of data than a long-term horizon.

The figure 4.4 visually confirms the data presented in Table 4.4, illustrating how closely each model's predictions aligned with the actual CPI values for Kenya. Notably the N-BEATS model's predicted line closely followed the actual CPI, showcasing its high accuracy and low error rates. Similarly, the MLP model performed effectively, with its predicted line nearly mirroring the actual CPI values. In contrast, the recurrent models (RNN, LSTM and GRU) and TCN showed significant deviation from the actual CPI, reflecting their higher error rates

Figure 11: Time series results of the predicted and observed consumption patterns by various model in a 3-month lookback period.



4.6 Comparison of Optimized NBEATS model and Other Optimized Deep Learning Models

Table 4.5, which shows the performance metrics of the tuned models for a 3-month lookback. The table highlights how parameter adjustments can significantly affect the accuracy of the predictions, especially when compared to the untuned results in Table 4.4.

Table 7 Performance metrics of the tuned models using the validation dataset for a 3 months lookback

	3 Months	
	MAE	**RMSE**
MLP	0.9363	1.077
RNN	9.071	9.808
LSTM	13.32	14.99
GRU	17.709	18.74
TCN	12.97	13.40
N-BEATS	***0.3777***	***0.4558***

Starting with N-BEATS, this model consistently delivered superior results, but tuning did not lead to improvement. In its untuned form, N-BEATS already outperformed most models, with an MAE of 0.3478 and an RMSE of 0.4364 for a 3-month lookback period.

After tuning, these values show a slight increase, with the MAE rising to 0.3777 and the RMSE to 0.4558. This minor degradation reflects that while N-BEATS is inherently robust, the tuning process did not enhance its performance. Despite this, N-BEATS remained the top performer among all models, excelling even in its untuned configuration.

The MLP model, on the other hand, showed significant improvement after tuning. In the untuned state, MLP exhibited an MAE of 1.558 and an RMSE of 1.626. After tuning, these metrics improve considerably, with the MAE decreasing to 0.9363 and the RMSE to 1.077. This reduction in error highlights the sensitivity of MLP to parameter adjustments, demonstrating that tuning can substantially boost the accuracy of this model. It is evident that MLP became more effective after fine-tuning, particularly in handling the shorter 3-month lookback period.

In contrast, the RNN model displayed poor performance in both tuned and untuned states. The untuned RNN shows high error rates, with an MAE of 8.889 and an RMSE of 10.25, and tuning did little to improve these metrics, as the MAE remained high at 9.071 and the RMSE only dropped slightly to 9.808. This minimal improvement suggests that RNN struggles with CPI forecasting tasks, and tuning does not significantly enhance its ability to generate accurate predictions.

The LSTM model, while benefiting somewhat from tuning, continued to exhibit suboptimal performance compared to the other models. In its untuned form, LSTM achieved an MAE of 2.664 and an RMSE of 2.940 for the 3-month lookback period, but tuning increased these errors, with the MAE rising to 13.32 and the RMSE to 14.99. Although tuning often enhances model performance, in the case of LSTM, it seemed to lead to instability or overfitting, as reflected by the higher error rates.

GRU similarly struggled, with tuning having negative impact on its performance. In the untuned configuration, GRU posted an MAE of 4.531 and an RMSE of 5.681, but after tuning, these metrics deteriorated, with the MAE rising sharply to 17.709 and the RMSE to 18.74. These results suggested that GRU, like RNN, may be ill-suited for this particular forecasting task, and tuning exacerbates rather than alleviates its issues.

Lastly, the TCN model showed a decline in performance after tuning. In its untuned state, TCN achieved an MAE of 4.696 and an RMSE of 4.741. However, after tuning, these metrics worsened, with the MAE increasing to 12.97 and the RMSE to 13.40. This sharp rise in errors indicated that tuning negatively impacted the TCN model, making it less effective compared to its untuned version. In comparison, its overall performance remained inferior to that of models like N-BEATS and MLP.

4.7 Discussion of Results

The results obtained from the analysis of the Consumer Price Index (CPI) data for the Kenyan economy reveal significant insights into both the trends in CPI values and the performance of the N-BEATS model compared to other deep learning models.

4.7.1 Trends in CPI data

The graphical representation of CPI trends from 2010 to 2023 highlights a gradual and consistent upward trajectory in inflation, with values increasing from approximately 60 to over 120. This doubling of the CPI over a little more than a decade underscores the persistent inflationary pressures in Kenya. This phenomenon has also been observed in many economies, particularly in Europe, where historical trends reveal significant increase in consumer price indices across different economies especially in the 21st century (Ljungberg, 2024). The lack of notable deflationary periods post great depression indicates a stable yet

increasing inflation environment, potentially influenced by various underlying factors, including currency depreciation and global supply chain issues. Policymakers need to be acutely aware of these trends as they formulate strategies to mitigate inflation's impact on consumer purchasing power and overall economic stability.

The boxplot analysis further reinforces the notion of stability in CPI values, devoid of significant seasonal variations, which suggests that inflationary dynamics are predominantly driven by long-term macroeconomic conditions rather than seasonal trends. This understanding can help in crafting economic policies that address structural issues rather than reacting to short-term fluctuations.

4.7.2 Performance of the N-BEATS model

The N-BEATS model achieved state-of-the-art results on three challenging datasets: M4, M3, and TOURISM, improving forecast accuracy by 11% over a statistical benchmark and by 3% over the M4 competition winner (Wang et al., 2022). This exceptional performance is further reflected in this research, where the predictive capabilities of the N-BEATS model were benchmarked against various other deep learning architectures. The model demonstrated superior performance, achieving a low Mean Absolute Error (MAE) of 0.3478 and a Root Mean Square Error (RMSE) of 0.4364 in the three-month lookback period. These metrics underscore the model’s effectiveness in capturing short-term trends in CPI data. The close alignment of actual versus predicted values, as depicted in the corresponding graphs, illustrates the model’s capacity to navigate the complexities of time series forecasting with remarkable accuracy.

The comparative analysis of model performance across varying lookback periods (3, 6, and 12 months) highlights the critical role that lookback plays in determining model effectiveness, particularly in time series forecasting. While the N-BEATS model consistently

outperformed its counterparts across all lookback periods, other models, such as MLP, showed improvements with a medium-long lookback of 6 months. In contrast, recurrent models like RNN, LSTM, and GRU struggled with longer lookback periods, especially in long-term predictions, with RNN exhibiting the highest error rates. This challenge reflects the difficulty these models face in capturing long-range dependencies, a well-known issue for recurrent architectures, especially when forecasting in environments with limited historical data.

According to Yusoff et al. (2024), the lookback parameter is a crucial hyperparameter in configuring LSTM models, as it governs how much past information the network incorporates when generating predictions. In models like LSTM, which have internal memory mechanisms, adjusting the lookback period directly affects the information stored and retrieved from previous time steps. Longer lookback periods theoretically allow the model to capture more extended temporal dependencies, but they can also introduce noise or irrelevant information, leading to diminished performance, as seen in the recurrent models in this study. This is particularly problematic in datasets with structural changes or inflationary patterns, where the relationship between past and future values is more complex and difficult to model.

It was also observed that the N-BEATS model exhibited a significantly faster training speed compared to other deep learning models, a finding consistent with prior research. For example, Puszkarski et al. (2022) noted that while N-BEATS excels in terms of computational efficiency. In their study on ECG signal classification, N-BEATS outperformed LSTM and GRU in training speed, with results showing it was faster by one to two degrees of magnitude. This faster training time is particularly advantageous in time series forecasting tasks such as CPI prediction, where iterative model tuning might be required. The

ability to rapidly train and iterate over models enables faster optimization, which is crucial when handling real-time applications. This efficiency makes N-BEATS a compelling option for tasks requiring quick turnarounds without sacrificing accuracy.

4.7.3 Effect of lookback period

Furthermore, the varied performance of these models across different lookback periods underscores the need for careful tuning of this parameter. For instance, shorter lookback periods (3 months) allowed some models, such as the RNN, to perform more effectively, while the longer periods (12 months) seemed to overburden them, amplifying their error rates. This suggests that recurrent models may require more sophisticated techniques, such as attention mechanisms or hybrid architectures, to better manage the balance between capturing relevant historical information and mitigating noise when the data is limited. Yusoff et al. (2024) emphasizes that selecting an optimal lookback value is not only model-dependent but also data-dependent, as shown in their evaluation of LSTM models for crude oil price forecasting. Their findings highlight that tuning the lookback value, along with other parameters like the train-test split, can significantly improve model performance.

On average, the models yielded the best performance when using a 3-month lookback period, with significant improvements seen across most models compared to longer lookback windows. This shorter horizon seemed to allow models such as MLP, LSTM, and GRU to capture relevant patterns in the time series more effectively, without being overwhelmed by excessive historical data. For instance, RNN, which generally struggled with longer lookbacks, exhibited its most stable performance with a 3-month window, reducing its Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) compared to 6- and 12-month periods, where its errors escalated dramatically. Similarly, models like GRU and LSTM saw their errors surge as the lookback period extended, likely because they failed to differentiate

signal from noise as more historical data were introduced. N-BEATS, already a top performer, maintained its dominance across all lookback periods, though it also recorded its lowest error metrics with the shortest lookback of 3 months.

This trend highlights the critical importance of choosing an appropriate lookback period for each model, particularly in cases where recurrent architectures like RNNs and GRUs are involved. Longer lookback periods may cause these models to lose focus on the most relevant and recent data, which is particularly problematic in forecasting tasks involving dynamic variables like inflation. The results align with findings from Yusoff et al. (2024), who noted that the optimal lookback value is highly dependent on both the model and the specific dataset, underscoring the need for tuning and experimentation with lookback as a parameter.

4.7.4 Effect of hyperparameter tuning

The analysis further explored the implications of hyperparameter tuning on model performance, specifically through the use of grid search in this research. While N-BEATS maintained its top position even after tuning, showing only a minor degradation in accuracy, other models, such as MLP, benefited from the tuning process. This contrast highlights the varying sensitivities of different architectures to hyperparameter adjustments. The MLP's significant improvement post-tuning suggests that it may be more adaptable to parameter changes than N-BEATS, which remained robust despite tuning efforts. This finding reflects the tendency of simpler models to show greater responsiveness to hyperparameter changes, while more sophisticated architectures like N-BEATS, which are designed with a high degree of internal flexibility, may exhibit a more stable performance without drastic improvements after tuning.

In this work, grid search was chosen for its simplicity and systematic exploration of parameter combinations. However, as Du et al. (2022) point out, grid search can be inefficient and time-consuming, especially in cases involving large numbers of hyperparameters. The process can become biased and repetitive, leading to suboptimal results. Du et al. (2022) suggest that evolutionary algorithms, such as the Bayesian Optimization Algorithm (BOA), could offer a more effective alternative. BOA strikes a balance between exploitation (using known information to search for the best results) and exploration (seeking out new possibilities), making it particularly useful in cases where grid search might falter due to time constraints or complexity. Their findings show that BOA significantly improved forecasting accuracy for models like attention-based LSTM by optimizing their hyperparameters more efficiently than grid search could.

Furthermore, Luo et al. (2024) demonstrated the effectiveness of BOA in tuning Elman and LSTM neural network models for short-term load forecasting. The study revealed substantial improvements in prediction accuracy when BOA was applied, reducing the Mean Absolute Error (MAE) by 56.35%, Root Mean Square Error (RMSE) by 57.75%, and Mean Absolute Percentage Error (MAPE) by 55.62%. The Bayesian-optimized models, such as BOA-Elman and BOA-LSTM, outperformed their traditional counterparts, showing greater stability and closer alignment with real-world data. This suggests that while grid search served as a solid foundation in this study, more advanced optimization techniques like BOA could offer additional benefits, particularly for recurrent models like LSTM and GRU, which exhibited instability and increased errors after tuning in this analysis.

CHAPTER FVE

CONCLUSIONS

5.1 Introduction

The importance of forecasting CPI cannot be overstated. This chapter presents the conclusions drawn from the research conducted on forecasting Consumer Price Index values using the N-BEATS deep learning architecture. The study has systematically explored the objectives outlined in the introduction chapter of this work, employing a robust methodology that leveraged historical CPI data spanning several decades. By adapting advanced deep learning techniques and rigorously evaluating their performance, this research contributes to the existing body of knowledge in economic forecasting. Deep learning approaches have a rare application in the field of economic forecasting.

In this chapter, we will summarize the key findings that emerged from the specific objectives of the study, highlighting the contributions made to both theoretical understanding and practical application. The limitations encountered during the research will also be discussed, providing context for the generalizability of the findings. Additionally, recommendations for future research will be offered to address these limitations and expand upon the insights gained through this study.

The conclusions drawn here aim to underscore the significance of the findings and their implications for researchers, policymakers, and economists, in the field of economic forecasting. By synthesizing the results and identifying potential areas for further exploration, this chapter seeks to illuminate the path forward in understanding and predicting inflation dynamics in the context of a rapidly evolving economic landscape.

5.2 Key Results from the Study Objectives

This study set out with the main objective of adapting the N-BEATS deep learning transformer architecture for nowcasting CPI numbers using past series of CPI data. Based on this main objective, the current study has achieved the following results from its specific objectives:

5.2.1 Specific objective 1: to examine the historical patterns in monthly cpi values

The first objective of this study was to examine the historical patterns in monthly Consumer Price Index (CPI) values, with the aim of identifying underlying inflation trends and potential influencing factors. This objective was achieved through an in-depth analysis of CPI data descriptive statistics as well as visualisations using line charts and boxplots, which provided critical insights into the behavior of inflation over an extended period.

The descriptive statistics of the CPI provided strong evidence of a long-term inflationary trend. The upward movement in CPI values over time, along with the small but steady positive returns, suggested that inflation increased consistently over the 40-year period. Rather than short-term volatility or economic shocks, the dataset reflected a gradual and compounding nature of inflation, which steadily elevated price levels over time. For modeling purposes, this clear trend in inflation suggested that time-series models, especially those designed to capture long-term dependencies like LSTM or N-BEATS, would be well-suited to forecast future CPI values. These models can effectively account for the upward drift and incremental nature of returns while recognizing occasional variations around this trend.

The line chart representing the monthly CPI for the Kenyan economy from 2010 to 2023 revealed a clear, consistent upward trend in CPI values too, starting from approximately

60 in 2010 and rising to over 120 by 2023. This continuous rise reflected sustained inflationary pressures over the past decade, highlighting the persistent increase in the prices of goods and services. The doubling of CPI values within this period suggested that the cost of living in Kenya had increased substantially, with inflation being a major contributor to this economic reality. The steady rise, without any significant periods of deflation or sharp volatility, indicated structural inflation challenges that had remained largely unchecked, potentially due to underlying factors such as currency depreciation, global supply chain disruptions, and possibly expansionary fiscal policies.

Moreover, the boxplot analysis of monthly CPI data, which covered a longer historical period starting from 1984, reinforced the findings from the line chart. The boxplot demonstrated that the CPI values within each month remained relatively stable, with no significant outliers or large variations. This consistency suggested that inflation in Kenya followed a steady, long-term upward trajectory rather than being subject to short-term or erratic price shocks. Additionally, the absence of distinct seasonal patterns in CPI fluctuations across months indicated that inflationary pressures in Kenya were driven more by long-term macroeconomic factors rather than seasonal variations.

5.2.2 Specific objective 2: to design and develop a robust n-beats model capable of accurately capturing observed historical patterns in monthly cpi values.

The N-BEATS model was configured with a horizon parameter of one, indicating a one-step forecast for the current month. The input size was set to three times the horizon, and the Huber loss function was employed to enhance the model's robustness against outliers.

The architecture of the N-BEATS model consisted of three distinct stacks: an identity stack, a seasonality stack, and a trend stack. This design was chosen for its ability to effectively capture the underlying patterns in the data by leveraging both seasonal and trend

components, alongside identity mapping, enhancing the model's overall predictive performance. Additionally, model initialization included start padding to ensure proper alignment of input sequences.

To validate the model's performance, a monthly frequency was specified for the data and the cross-validation conducted by dividing the dataset into validation and test sets, allowing for a rigorous assessment of the model's predictive accuracy. The results from the cross-validation highlighted the model's ability to accurately forecast CPI values, confirming its robustness and reliability for economic time series analysis.

These results are demonstrated by the metrics in Tables 4.4 and 4.5, providing clear evidence that this objective was achieved. In its untuned form, the N-BEATS model consistently outperforms all other models, demonstrating its inherent robustness and superior ability to forecast CPI values with minimal errors. With an MAE of 0.3478 and an RMSE of 0.4364 for the 3-month lookback period, the untuned N-BEATS model significantly surpasses the accuracy of competing models, which exhibit much higher error rates. Even after tuning, N-BEATS maintains its leading position, with only a minor increase in error rates (MAE of 0.3777 and RMSE of 0.4558). This slight degradation indicates that while N-BEATS is already highly optimized in its default configuration, tuning did not provide additional improvements but did not detract significantly from its performance either.

The resilience of N-BEATS in both tuned and untuned configurations demonstrates its capability to capture the underlying historical patterns in CPI data with high accuracy, meeting the intended design goal. Moreover, the comparison with other models further emphasizes the success of N-BEATS in this context. Models such as RNN, LSTM, and GRU either fail to improve significantly with tuning or exhibit poorer overall performance, as indicated by their higher MAE and RMSE values. Even models that perform relatively well

after tuning, such as MLP, cannot match the baseline performance of the untuned N-BEATS model, let alone surpass its tuned performance.

Thus, the empirical results confirm that the N-BEATS model designed for this study is highly effective in meeting the objective of capturing historical CPI trends with precision. Despite tuning leading to minor variations in performance, the N-BEATS model consistently delivers the lowest prediction errors across the board, establishing it as the most reliable model for CPI forecasting.

5.2.3 Specific objective 3: to test and validate the developed model.

This objective aimed to test and validate the developed N-BEATS model in order to assess its effectiveness and generalizability in forecasting monthly CPI values. The validation process involved a comprehensive performance evaluation using multiple error metrics, including MAE (Mean Absolute Error) and RMSE (Root Mean Squared Error), to compare the predictive capability of N-BEATS against a range of benchmark deep learning models, both in tuned and untuned configurations.

The dataset was first split into training, validation, and testing sets, following an allocation of 80% for training, 10% for validation, and 10% for testing. This approach was crucial for evaluating the model's performance accurately and ensuring its generalizability. The training set enabled the model to learn from historical data, while the validation set facilitated hyperparameter tuning and model selection. Finally, the testing set served as an independent dataset to assess the models' performance and robustness. This structured approach ensured that the models' predictive capabilities could be rigorously evaluated, ultimately contributing to the reliability of the analysis.

The results presented in Table 4.5, which summarize the performance metrics of the tuned models, and Table 4.4 for the untuned models, serve as the foundation for this validation. In the validation process, the N-BEATS model consistently outperforms the majority of models across different evaluation criteria. With an MAE of 0.3478 and RMSE of 0.4364 for the untuned N-BEATS model (Table 4.4), and only a slight increase in errors after tuning (MAE of 0.3777 and RMSE of 0.4558, as shown in Table 4.5), the validation results affirm the model's strong predictive capabilities and robustness.

The validation of N-BEATS also highlights its ability to generalize well across varying forecast horizons, including the 3-month lookback period used in this study. Importantly, the model’s validation error metrics are consistently lower than those of alternative models, such as MLP, RNN, LSTM, GRU, and TCN, which exhibit higher MAE and RMSE values. For example, the tuned LSTM and GRU models, despite parameter optimization, display significantly higher error rates (MAE of 13.32 and 17.709, respectively). This disparity underscores the superiority of N-BEATS in capturing the intricate patterns in CPI data and delivering accurate forecasts.

In addition to the error metrics, the N-BEATS model's stability was evident. Unlike some other models where tuning led to instability and increased error rates (e.g., LSTM and TCN), N-BEATS demonstrated consistent performance both before and after tuning, reinforcing the robustness of the model in handling CPI forecasting tasks

5.3 Key Contributions of the Current Research

This This research has made significant strides in the field of economic forecasting, particularly in the context of predicting Consumer Price Index (CPI) values using advanced deep learning methodologies. One of the primary contributions of this study is the successful adaptation of the N-BEATS model, a state-of-the-art deep learning architecture, specifically

tailored for forecasting monthly CPI values. By employing this innovative model, the research has demonstrated its ability to effectively capture historical patterns and trends in inflation data, providing a reliable tool for economic forecasting.

A pivotal contribution of this study lies in its methodological rigor, particularly in the utilization of a robust dataset spanning several decades. The analysis of CPI data from 1984 to 2023 allowed for a comprehensive understanding of inflation trends and their underlying patterns. The use of visualizations, such as line charts and boxplots, facilitated a clear presentation of these trends, reinforcing the findings and enabling stakeholders to easily comprehend the behavior of inflation over time.

Moreover, the study's empirical results underscore the efficacy of the N-BEATS model compared to other prevalent deep learning architectures, including MLP, RNN, LSTM, GRU, and TCN. Through rigorous cross-validation and performance evaluation, the N-BEATS model consistently outperformed its counterparts, demonstrating lower prediction errors as evidenced by metrics such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). This not only establishes the N-BEATS model as a superior choice for CPI forecasting but also contributes to the growing body of evidence supporting the adoption of deep learning techniques in economic modeling.

Another notable contribution is the emphasis on hyperparameter tuning and its critical role in enhancing model performance. While previous studies have often overlooked this aspect, the current research highlights the importance of optimizing hyperparameters tailored to different models. This insight serves as a valuable guideline for future research, encouraging a more thorough exploration of hyperparameter configurations to achieve optimal predictive accuracy.

Furthermore, the study provides an important perspective on the structural characteristics of inflation in Kenya, revealing a consistent upward trend in CPI values over the past decade. By identifying the factors contributing to this sustained inflationary pressure, the research adds depth to the understanding of economic dynamics in the region, offering actionable insights for policymakers and economists.

5.4 Limitations of the Current Research

While this research has made significant contributions to the understanding and forecasting of Consumer Price Index (CPI) values using the N-BEATS deep learning architecture, it is important to acknowledge several limitations that may affect the generalizability and applicability of the findings.

Firstly, this study did not include a comparative analysis of statistical models. By focusing solely on deep learning architectures, the research may overlook the strengths and insights provided by traditional statistical models, such as ARIMA or exponential smoothing methods. Including such comparisons could have enriched the analysis and offered a more comprehensive perspective on the effectiveness of various forecasting approaches. As a result, researchers are encouraged to consider the findings within the context of modern machine learning techniques, recognizing that the performance of the N-BEATS model may differ when compared with established statistical methods.

Additionally, the research primarily utilized univariate modeling techniques without incorporating other variables that could have provided deeper insights into the factors influencing CPI values. Specifically, the study did not account for potential variables such as the month, lagged variables of the index, or rolling averages of the index. These elements could have played a crucial role in capturing more complex dynamics in CPI trends. By

excluding these features, the analysis may have missed opportunities to enhance model performance and better understand the underlying drivers of inflation.

Furthermore, the absence of exogenous variables in the current research limits the scope of the analysis and its applicability to real-world economic scenarios. Exogenous variables, which include external factors that can influence the Consumer Price Index but are not part of the CPI data itself, play a significant role in understanding inflation dynamics. For instance, economic indicators such as interest rates, unemployment rates, commodity prices, and fiscal policies can profoundly affect consumer prices. By incorporating these external factors into the forecasting model, researchers can capture a broader range of influences on CPI fluctuations. For example, interest rate changes can affect consumer spending and investment, thereby influencing demand-side pressures on prices. Similarly, commodity price shocks can lead to immediate changes in production costs, which may subsequently impact consumer prices. Including exogenous variables in future research could facilitate a more holistic understanding of inflationary trends and enhance the predictive accuracy of models (Wang et al., 2024). Integrating these diverse datasets would allow researchers to explore the intricate relationships among multiple economic indicators and their combined effects on CPI movements. This expanded analysis would not only improve model robustness but also provide policymakers with more actionable insights for managing inflationary pressures.

Moreover, the fine-tuning process for the models was limited to common hyperparameters, despite the fact that different models often have unique key parameters that may require specific tuning for optimal performance. This may have resulted in suboptimal configurations for certain architectures, limiting their predictive accuracy. Future research could benefit from a more granular approach to hyperparameter optimization, ensuring that each model's unique characteristics are appropriately considered during the tuning process.

Another limitation of this study is its focus on data from a single country. While this localized approach provides valuable insights into the specific dynamics of the economy, it limits the generalizability of the findings to other contexts. Economic conditions, consumer behaviors, and inflationary pressures may vary across different countries and regions. Consequently, the applicability of the N-BEATS model and its performance metrics may differ in diverse economic environments and may call for change of its configurations to achieve high accuracy. This suggests that future studies should explore CPI forecasting in various economic settings to enhance the robustness of the findings.

Finally, the research does not delve into the human behavior aspects underlying CPI values. Understanding consumer behavior, such as spending patterns, savings tendencies, and responses to economic policies, could significantly influence CPI dynamics. The interplay between human psychology and economic indicators is complex and warrants further investigation. Incorporating behavioral economics into the analysis could provide richer insights into how CPI values are shaped by consumer sentiment and market dynamics, ultimately leading to more comprehensive forecasting models.

5.5 Recommendations for Future Research

Building on the limitations identified in this study, several recommendations for future research emerge that could enhance the understanding of Consumer Price Index (CPI) forecasting and address the gaps highlighted by the current work.

Firstly, future studies should include a comparative analysis of both traditional statistical models and modern deep learning architectures. Traditional models, such as ARIMA and exponential smoothing, have long been considered the standard due to their robustness, efficiency, and ease of use for non-expert users (Bharadiya, 2023). Their utility and capabilities cannot be overlooked, especially since they have consistently demonstrated

high accuracy in various forecasting scenarios. By integrating these traditional models with modern machine learning techniques, researchers can develop a more nuanced understanding of how different approaches perform under varying conditions. Such a comparative framework would enrich the analysis and provide valuable insights into the relative strengths and weaknesses of various forecasting methodologies. Bharadiya et al. (2023) noted that while Recurrent Neural Networks (RNNs) have emerged as competitive forecasting methods, most notably evidenced by their performance in the recent M4 competition, established models like ETS and ARIMA still possess advantages that RNNs have yet to match. This understanding could pave the way for hybrid models that leverage the strengths of both statistical and machine learning techniques, ultimately leading to improved predictive accuracy.

Secondly, future research should aim to adopt a more comprehensive modeling approach by engineering additional features (Galli, 2024). Variables such as related to dates, lagged CPI values, and rolling averages of the index could offer deeper insights into the temporal dynamics of inflation. By utilizing a multivariate framework, researchers can capture the intricate interactions among different economic indicators and better understand the drivers of CPI fluctuations. This expanded approach would likely enhance model performance and provide a more holistic view of inflationary trends.

The inclusion of exogenous variables in future research on Consumer Price Index (CPI) forecasting could substantially enhance model accuracy and robustness. Exogenous variables, which are external factors that influence CPI but are not intrinsic to the time series itself, encompass a range of economic indicators such as interest rates, unemployment rates, commodity prices, and fiscal policies. These variables often drive inflationary pressures, and integrating them into forecasting models allows researchers to capture the broader economic

dynamics that shape CPI movements. For example, fluctuations in oil prices have a direct impact on transportation costs, which in turn affect the overall price levels of consumer goods. Similarly, macroeconomic policies and global trade patterns influence domestic inflation, and accounting for such exogenous factors can provide a more holistic view of the forces at play. According to the findings of Zheng et al. (2023), incorporating alternative data sources like media reports and Internet search trends into CPI forecasting models significantly improved accuracy. Their study demonstrated that the inclusion of these diverse data streams provided a more timely and nuanced understanding of inflation trends, particularly during periods of economic uncertainty. By incorporating exogenous variables, models not only gain predictive power but also align with modern forecasting techniques that leverage comprehensive datasets to uncover complex relationships between inflation and broader economic conditions. This approach could further solidify the robustness of CPI forecasts and offer deeper insights into future price trends.

Furthermore, the fine-tuning process for deep learning models should be refined to accommodate the specific hyperparameters unique to each architecture. For instance, given that the 3-month lookback consistently provided the best results across most models compared to other lookback, it stands to reason that further tuning lookback as hyperparameter could potentially yield a more optimal lookback that could improve the results. Future research could also explore advanced techniques such as Bayesian optimization or genetic algorithms for hyperparameter tuning, ensuring that each model is configured for optimal performance (Foroutan & Lahmiri, 2024). This tailored approach to hyperparameter optimization could lead to significant improvements in predictive accuracy and robustness, allowing for a more precise forecasting of CPI values.

In their recent study on optimal architectures for weather forecasting, Habib et al. (2023) revealed that N-BEATS was the most adaptable and reliable among the tested architectures, which included LSTM and the Temporal Fusion Transformer. N-BEATS consistently delivered excellent results across various datasets due to its flexibility and balanced model size. This adaptability highlights the potential of N-BEATS for diverse forecasting applications, including the Consumer Price Index (CPI), suggesting it could serve as a robust tool for real-world economic predictions. To enhance the generalizability of the findings of this research, it is crucial to expand the geographical scope of the research and examine N-BEATS' robustness in forecasting CPI across multiple countries or regions with differing economic conditions. By assessing N-BEATS' adaptability and performance in these varied contexts, researchers can gain valuable insights into how local economic factors influence CPI dynamics. Such comparative analyses would provide essential information for policymakers and economists seeking to understand inflation on a broader scale.

Lastly, future research should explore the behavioral dimensions that influence the Consumer Price Index (CPI), as human behavior plays a significant, though often overlooked, role in inflation dynamics. Behavioral economics provides critical insights into how psychological, social, and emotional factors drive consumer decisions, ultimately affecting spending patterns and inflation. For instance, consumer sentiment can shift dramatically in response to perceived economic stability or uncertainty, leading to changes in demand that influence price fluctuations across various industries. Similarly, spending habits, shaped by both individual experiences and collective societal behaviors, add complexity to CPI forecasting. By integrating these behavioral aspects into forecasting models, researchers can develop more sophisticated tools that account for not only economic fundamentals but also human reactions to market conditions, policies, and global events. Incorporating such behavioral insights into CPI models could significantly enhance their adaptability and

accuracy. Recent studies, such as those by Tariq et al. (2024), suggest that deep learning (DL) methods have shown considerable promise in interpreting complex economic data, particularly in scenarios where traditional models fall short. DL models excel at identifying patterns shaped by human behavior, such as risk perception and market sentiment, factors that are often difficult to capture with conventional statistical approaches. While challenges like model interpretability and data limitations remain, future research could benefit from integrating behavioral economics into DL frameworks. This approach would not only improve CPI forecasts but also create tools that better reflect the realities of consumer behavior, offering policymakers a more effective resource for anticipating and mitigating inflationary trends.

REFERENCES

Wang, X., Li, C., Yi, C., Xu, X., Wang, J., & Zhang, Y. (2022). *EcoForecast: An interpretable data-driven approach for short-term macroeconomic forecasting using N-BEATS neural network*

Solís, M., & Calvo-Valverde, L.- A. (2023). A proposal of transfer learning for monthly macroeconomic time series forecast. Engineering Proceedings, 39, 58. https://doi.org/10.3390/engproc2023039058

Puszkarski, B., Hryniów, K., & Sarwas, G. (2022). Comparison of neural basis expansion analysis for interpretable time series (N-BEATS) and recurrent neural networks for heart dysfunction classification.

Reference: Kara, B., & Keskin, A. (2021). Producer price index, consumer price index and fiscal policy: 1996-2020 period and Turkey. KAUJEASF, 12(24), 881-899. 1

Javidani, A., & Mahmoudi-Aznaveh, A. (2019). Learning representative temporal features for action recognition. 2019 25th IEEE International Conference on Image Processing (ICIP). IEEE. 1

Plonis, D., Katkevičius, A., Urbanavičius, V., Miniotas, D., Serackis, A., & Gurskas, A. (2019). Delay systems synthesis using multi-layer perceptron network. Acta Physica Polonica A, 133(5), 1281-1286. 1

Cinar, A. C. (2020). Training feed-forward multi-layer perceptron artificial neural networks with a tree-seed algorithm. Arabian Journal for Science and Engineering, 45(12), 10915-10938

Foroutan, P., Lahmiri, S. Deep learning systems for forecasting the prices of crude oil and precious metals. *Financ Innov* 10, 111 (2024). https://doi.org/10.1186/s40854-024-00637-z

Kaushik, M., & Giri, A. K. (2019). Forecasting foreign exchange rate: A multivariate comparative analysis between traditional econometric, contemporary machine learning & deep learning techniques. In 2019 IEEE 9th International Conference on Advanced Computing (IACC) (pp. 1-6). IEEE.

Paranhos, L. (2023). Predicting inflation with recurrent neural networks. Bank of England Working Paper, (No. 1234)

Luo, G., Shen, Z., Zhou, S., & Mei, K. (2024). Short-term load forecasting based on Elman neural network optimized by Bayesian algorithm.

Yusoff, M., Sharif, M. Y., & Sallehuddin, M. T. M. (2024). Effect of hyperparameter tuning in long short-term memory for crude oil price prediction.

Llugsi, R., Lupera, P., El Yacoubi, S., & Fontaine, A. (2024). Comparison between Adam, AdaMax, and AdamW optimizers to implement a weather forecast based on neural networks for the Andean city of Quito.

Schimunek, J., Friedrich, L., Kuhn, D., Rippmann, F., Hochreiter, S., & Klambauer, G. (2021). A generalized framework for embedding-based few-shot learning methods in drug discovery. Journal Name, Volume (Issue), page range. Retrieved from URL

Du, L., Gao, R., Suganthan, P. N., & Wang, D. Z. W. (2022). Bayesian optimization based dynamic ensemble for time series forecasting. *Information Sciences*, 591, 155-175. https://doi.org/[DOI]

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., & Rush, A. M. (2020). Transformers: State-of-the-art natural language processing1. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (pp. 38-45)2. Association for Computational Linguistics3.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2019). Attention is all you need. In Advances in neural information processing systems (pp. 5998-6008).

Tariq, U., Ahmed, I., Khan, M. A., & Bashir, A. K. (2024). Deep learning for economic transformation: A parametric review. *Indonesian Journal of Electrical Engineering and Computer Science*, 35(1), 520-541. https://doi.org/10.11591/ijeecs.v35.i1.pp520-541

Hewamalage, H., Bergmeir, C., & Bandara, K. (2020). Recurrent neural networks for time series forecasting: Current status and future directions.

Laptev, N., Yu, J., & Rajagopal, R. (2019). Reconstruction and regression loss for time-series transfer learning.

Zheng, T., Fan, X., Jin, W., & Fang, K. (2023). Forecasting CPI with multisource data: The value of media and internet information. *Forecasting*, https://doi.org/10.1002/for.3048

Raj, A. S. A., Kumarasankaralingam, L., Balamurugan, M., Maheswari, B., Gowri, J., & Dutta, A. (2024). Forecasting the economic crisis of Sri Lanka: Application of machine learning algorithms for time series data. In *Proceedings of the International Conference on Machine Learning and Data Engineering (ICMLDE 2023)*

Ghosh, A., & Bhargava, P. (2019). Neural network forecasting for inflation in India

Brownlee, J. (2019). Deep learning for time series forecasting: Predict the future with MLPs, TCNs and LSTMs in Python. Machine Learning Mastery.

APA reference: Oreshkin, B. N., Carpov, D., Chapados, N., & Bengio, Y. (2020). N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. In Proceedings of the 8th International Conference on Learning Representations (ICLR 2020). Retrieved from https://openreview.net/pdf?id=r1ecqn4YwB

Kim, S. (2020). Macroeconomic and Financial Market Analyses and Predictions through Deep Learning12. BOK Working Paper, No. 2020-1834.

Bharadiya, J. P. (2023). *Exploring the use of recurrent neural networks for time series forecasting* (Doctoral dissertation). University of the Cumberlands.

Chen, C.-C., & Chen, C.-W. (2020). Time Series Analysis: Theory and Applications. InTechOpen. https://doi.org/10.5772/intechopen.89444

Du, X., Isufi, E., Sabbaqi, M., & Yang, M. (2022). Short-term earthquake prediction via recurrent neural network models: Comparison among vanilla RNN, LSTM and Bi-LSTMarXiv preprint arXiv:2201.134441.

Hewamalage, H., Bergmeir, C., & Bandara, K. (2020). Recurrent neural networks for time series forecasting: Current status and future directionsarXiv preprint arXiv:1909.005901.

Benidis, K., Sundar Rangapuram, S., Flunkert, V., Wang, Y., Maddix, D., Turkmen, C., Gasthaus, J., Bohlke-Schneider, M., Salinas, D., Stella, L., Aubet, F.-X., Callot, L., & Januschowski, T. (2022). Deep learning for time series forecasting: Tutorial and literature survey. arXiv preprint arXiv:2004.10240v2.

Garg, S., Mitra, S., Yu, T., Gadhia, Y., &Kashettiwar, A. (2023, June). Reinforced Approximate Exploratory Data Analysis. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 37, No. 6, pp. 7660-7669).

Stundziene, A., Pilinkiene, V., Bruneckiene, J., Grybauskas, A., Lukauskas, M., & Pekarskiene, I. (2023). Future directions in nowcasting economic activity: A systematic literature review

Zahara, S., Sugianto, & Ilmiddaviq, M. B. (2019). Consumer price index prediction using Long Short Term Memory (LSTM) based cloud computing

Mei, J., Guo M. (2022). Comparative Analysis of CPI Index Intelligent Prediction Based on ARIMA & LSTM Model.

Liashchynskyi, P., & Liashchynskyi, P. (2019). Grid search, random search, genetic algorithm: A big comparison for NAS.

AlKandari, M., & Ahmad, I. (2019). Solar power generation forecasting using ensemble approach based on deep learning and statistical methods. Applied Computing and Informatics.

Ljungberg, J. European consumer price indices since 1870. *Cliometrica* (2024). https://doi.org/10.1007/s11698-024-00283-6

Ahmed, S., Nielsen, I. E., Tripathi, A., Siddiqui, S., Ramachandran, R. P., & Rasool, G. (2023). Transformers in time-series analysis: A tutorial.

Kumari, S., & Singh, S. K. (2022). Deep Learning-based Time Series Models for GDP and ICT Growth Prediction in India2022 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), 250-256

Bakhit, D. M. A., Nderu, L., & Ngunyi, A. (2024). A hybrid neural network model based on transfer learning for Arabic sentiment analysis of customer satisfaction. Engineering Reports, e12874. https://doi.org/10.1002/eng2.12874

Desifatma, E., Djaja, I. G. P. F. S., Pratomo, P. M., Supriyadi, Mustopa, E. J., Evita, M., Djamal, M., & Srigutomo, W. (2024). Robust inversion of 1D magnetotelluric data using the Huber loss function. *Computational Geosciences, 28*(4), 629–643. https://doi.org/10.1007/s10596-024-10286-x

Wang, Y., Wu, H., Dong, J., Liu, Y., Qiu, Y., Zhang, H., Wang, J., & Long, M. (2024). *TimeXer: Empowering transformers for time series forecasting with exogenous variables* (arXiv:2402.19072). https://doi.org/10.48550/arXiv.2402.19072

Challu, C., Olivares, K. G., Oreshkin, B. N., Garza Ramirez, F., Mergenthaler Canseco, M., & Dubrawski, A. (2023). NHITS: Neural Hierarchical Interpolation for Time Series Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, *37*(6), 6989-6997. https://doi.org/10.1609/aaai.v37i6.25854

Habib, A. B., Ashraf, F. B., Hossain, M. I., & Alam, G. R. (2023). N-BEATS & Temporal Fusion Transformer-based surface temperature prediction and forecasting for realizing global warming trends.

Galli, S. (2024). *Python feature engineering cookbook*.

APPENDICES

Table 8 Project Schedule

Activity	**June**	**Nov**	**Jan**	**Feb**	**March**	**April**	**May**
Develop project idea							
Develop research questions							
Write research proposal							
Conduct literature review							
Present proposal							
Collect research data							
Develop Study model							
Conduct data analysis							
Conduct model validation							
Compile research results							
Prepare project dissertation							
Final project defense							

Table 9 Project Budget

No.	Item	Unit	Price	Total
1	Computer/ hard disk/ operating system	1	120,000	120,000
2	Internet connectivity/ data	30	1,000	30,000
3	Field data expenses	1	5,000	5,000
4	Final dissertation preparation (i.e., printing, binding, etc.)	1000	20	20,000
5	Flash drive	1	3,500	1,500
6	NACOSTI approval of research	1	1000	1000
7	Miscellaneous costs	1	20000	10,000
	Grand Total			**222,500**