

Evaluation of Deep Learning Models for Forecasting Fresh Produce Demand

By

Ajibola Olaosebikan

A Thesis Submitted to Solent Southampton University to fulfil the
thesis requirement for the Master of Science degree.

In

Applied Artificial Intelligence and Data Science

Under the supervision of Dr. Muntasir Al-Asfoor

Southampton, Hampshire, United Kingdom, September 9, 2022.

© Ajibola Olaosebikan 2022.

Author's Declaration

I am aware that this thesis might be made available for public use.

Abstract

Fresh produce (FP) is one of the most important items traded in most retail outlets due to its nutritional benefits. However, the downside of these products is that their quality depreciates over time which may mean losses for the retailers. In a bid to curb profit decline incurred by product waste, retailers have managed to leverage traditional timeseries forecasting to determine customers' demand. This approach, however, is limited because it cannot capture the non-linearity in customers' purchasing patterns, which makes deep learning techniques a viable alternative

The fresh produce supply chain network (FPSC) is complex and this thesis only focus on the post-harvest marketing operations of the network by leveraging deep learning (DL) models to determine the likely demand of FP and specifically, strawberry because of its difficulty in modelling.

This demand forecast technique intricacies were explored systematically premised on the established characteristics of two major deep learning models: the Long Short Term Memory (LSTM) neural network— which uses the temporal relationship of data to make predictions and the Convolutional Neural Network with LSTM (CNN-LSTM)— which uses the spatio-temporal relationship of data to predict.

Furthermore, a self-attention layer was added to the LSTM model and the LSTM portion of the CNN-LSTM in a bid to optimize them before making decision on the better model. To further strengthen the decision process, three key things were factored in determining the models' (both the optimized and non-optimized) ability: their; respective error levels on the validation dataset using the mean absolute error as the metric, predictive ability with the test data using the coefficient of determination (R^2 -score) as the metrics and forecasting (across 28weeks horizon) skill using R^2 -score.

The outcome revealed the optimized version of LSTM, that is the self-attention LSTM outperform other models in all ramifications. Although not part of the decision process, a two end of an argument was explored to observe the findings of a manual hyperparameter optimization or an automatic approach using the Bayesian optimization technique. This was done for the LSTM model alone.

Acknowledgements

I appreciate all the advice and assistance my supervisor, Dr. Muntasir Al-Asfoor, gave me while working on the projects. I value all of the employees in the Solent department of Applied AI and Data Science, including the members of the dissertation committee.

I am incredibly appreciative to my siblings and parents, Mr. Olaosebikan A. Saheed and Mrs. Rasheedah Olaosebikan, for their esteemed support, love, and extraordinary sacrifices that made this Master's degree possible. I appreciate it. May Allah be pleased with them.

Mr. Adrian Mensah Coker, thank you. I value the constant advice you provide. I also want to express my gratitude to Oguntola Ibrahim and Abdul Lateef Ogundipe for taking the time to clarify complex ideas.

I acknowledge and appreciate Badiru Oluwaseun, Oyeyemi Adedapo, Fashola Sunkade, Mustapha Fatai, and Omoleke Mopelola for all their display of affection and understanding. I hope you all always have access to kind people in your life.

Contents

List of Figures.....	7
List of Tables	9
Chapter 1	10
1.1 Introduction.....	10
1.2 Problem Statement:	11
1.3 Motivation:.....	12
1.4 Research Scope:.....	13
1.5 Research question, Aim and Objectives	13
1.6 Thesis Organization	14
Chapter 2	15
2.1 Background and Literature	15
2.2 Management of the fresh produce (FP) supply chain.....	15
2.2.1 The enablers and inhibitors of fresh produce demand planning	16
2.3 Time Series	18
2.3.1 Univariate Time Series	19
2.3.2 Multivariate Time Series	20
2.4 Forecasting and Predictive Techniques	20
2.5 Deep Learning.....	21
2.6 Artificial Neural Networks (ANNs).....	22
2.6.1 Recurrent Neural Networks (RNNs)	23
2.6.1.1 Long Short Term Memory Network (LSTM)	25
2.6.2 Attention Mechanism	26
2.6.3 Convolutional Neural Networks (CNN).....	27
2.7 Performance Metrics.....	28
2.8 Related Work.....	30
2.9 Summary.....	30
Chapter 3	32
3.1 Methodology and Implemented Solution	32
3.2 Domain Knowledge	32
3.3 The Dataset.....	32
3.3.1 Feature selection and engineering	34
3.4 Models	35
3.4.1 Model 1: Long Short Term Memory (LSTM)	36
3.4.2 Model 2: Convolutional Neural Network with LSTM (CNN-LSTM)	37
3.4.3 Model Optimization	38

3.4.3.1 Bayesian Optimization	38
3.4.3.2 Optimization with Self-Attention layer	40
3.4.4 Evaluation Metrics	41
3.4.5 Hyperparameter Tuning.....	41
3.5 Technical implementation	42
3.6 Summary.....	42
Chapter 4	43
4.1 Experiments, Results and Discussion.....	43
4.2 Experiment Set-Up	43
4.2.1 Experiment idea	44
4.2.2 The Central Arguments:	44
4.3 Experiment Results.....	47
Examining the manual versus automatic tuning arguments	47
Experiment ID: 1 (LSTM vs CNN-LSTM).....	57
Experiment ID: 2 (Self-Attention LSTM vs LSTM)	65
Experiment ID: 3 (Self-Attention CNN-LSTM vs CNN-LSTM)	69
4.3 Discussion	70
4.3 Summary.....	73
Chapter 5	74
5.1 Conclusion	74
5.2 Limitations:.....	76
References	77
Appendix A	83
Model Artifact and Requirement.....	83
Appendix B.....	84
Graphs and chart	84

List of Figures

Figure 2.1: A simple ANN architecture	23
Figure 2.2: An unrolled RNN recurrent neural network.....	24
Figure 2.3: LSTM cell showing the three gate.....	26
Figure 2.4: CNN architecture.....	27
Figure 3.1: Assumed demand trend over a 12 year period.....	33
Figure 3.2: Assumed demand trend over a year period.....	34
Figure 3.3: The VIF of the retained variables.....	33
Figure 3.4: Summary of the implemented LSTM model.....	36
Figure 3.5: Implemented LSTM model architecture.....	37
Figure 3.6: CNN-LSTM model architecture.....	37
Figure 3.7: CNN-LSTM model summary.....	38
Figure 3.8: Self-Attention LSTM model summary.....	40
Figure 3.9: Data showing the impact of the model checkpoint.....	42
Figure 4.1: The mlflow uri through the anaconda prompt.....	
Figure A.1: A screenshot of the MLFlow experiment set up and runs.....	
Figure A.2: A screenshot of the logged metrics with MLFlow.....	
Figure 4.2: Graphical illustration of the Bayesian detected least Validation MAE(val_mae) and the Training data MAE(mae) over a fixed epoch size of 300.....	48
Figure 4.3: Graphical representation of the Bayesian detected Training-loss (loss) and Validation loss (val_loss) based on the least Validation Mae over a fixed epoch size of 300.....	48
Figure 4.4: Proposed LSTM layers by the Bayesian Process of automatic tuning.....	50
Figure 4.5: Graphical representation of the Bayesian detected Training-loss (loss) and Validation loss (val_loss) based on the least Validation Mae over a fixed epoch size of 300.....	50
Figure 4.6: Graphical illustration of the Bayesian detected largest Validation MAE(val_mae) and the Training data MAE(mae) over a fixed epoch size of 300.....	51
Figure 4.7: BP-detected model summary with layers based on the largest validation MAE.....	51
Figure 4.8: The train and validation loss of manually tuned LSTM at 200 epochs.....	52
Figure 4.9: Graph of the training and validation MAE for manually tuned LSTM hyperparameters.....	53
Figure 4.9(a): Training and validation MAE of the manually tuned hyperparameters based on the largest validation MAE.....	54
Figure 4.9(b): Training and validation loss of the manually tuned hyperparameters based on the largest validation MAE.....	55

Figure 4.9(c): Model summary showing the layers of the manually tuned LSTM with the least MAE.....	56
Figure 4.9(d): Model summary showing the layers of the manually tuned LSTM with the largest MAE.....	56
Figure 4.9(e): Log graph showing the validation MAE, training MAE, and the final MAE on the validation data for the CNN-LSTM model.....	58
Figure 4.9(f): Log-scaled graphical representation of the relationship between the CNN-LSTM MAE and the R2-score on the test dataset.....	59
Figure 4.9(g): Log graph showing the validation MAE and training MAE on the validation data for the LSTM model.....	60
Figure 4.9(h): Log-scaled graphical representation of the relationship between LSTM MAE and R2-score on the test dataset.....	60
Figure 4.9(i): Graphical Illustration of the LSTM 7weeks likely demand forecast.....	61
Figure 4.9(j): Graphical illustration of the LSTM 14weeks likely demand forecast.....	62
Figure 4.9(k): Graphical illustration of the LSTM 28 weeks likely demand forecast.....	63
Figure 4.9(l): Graphical illustration of the CNN-LSTM 7weeks likely demand forecast.....	63
Figure 4.9(m): Graphical illustration of the CNN-LSTM 14weeks likely demand forecast.....	64
Figure 4.9(n): Graphical illustration of the CNN-LSTM 28 weeks likely demand forecast.....	64
Figure 4.9(o): Graphical training trend of the Self-Attention based LSTM model over an epoch size of 300.....	66
Figure 4.9(p): Log-scaled graphical representation of the relationship between Self-Attention LSTM MAE and R2-score on the test dataset.....	67
Figure 4.9 (q): 7 weeks forecast with Self-Attention LSTM.....	68
Figure 4.9 (r): 14 weeks forecast with Self-Attention LSTM.....	68
Figure 4.9(s): 28 weeks forecast with Self-Attention LSTM.....	69

List of Tables

Table 4.1: Experiment type and hypothesis testing.....	46
Table 4.2: Table showing various case scenarios and the analysed data.....	47
Table 4.3: combination of hyperparameters from a BP.....	49
Table 4.4: combination of hyperparameters from a BP(largest MAE).....	52
Table 4.4: Table showing the hyperparameter combinations of the two models.....	61
Table 4.5a : Key metrics of LSTM and CNN-LSTM in 7 weeks forecast.....	65
Table 4.5b : Key metrics of LSTM and CNN-LSTM in 14 weeks forecast.....	65
Table 4.5c : Key metrics of LSTM and CNN-LSTM in 28 weeks forecast.....	65
Table 4.6: Table showing the hyperparameter combinations and the key performance metrics of the two models.....	66
Table 4.7: Summary of the LSTM and LSTM with Attention forecasting metrics.....	68
Table 4.8: Table showing the hyperparameter combinations and the key performance metrics of the two models.....	69
Table 4.9: Summary of the CNN-LSTM and CNN-LSTM with Attention forecasting metrics.....	70
Table 4.9(a) : Table showing the hypothesis and decision made.....	73

Chapter 1

1.1 Introduction

Achieving sustainability in supply chain management (SCM) is extremely difficult, especially in agri-food supply chains due to the lengthy procedures broadly classified as pre-harvesting and post-harvesting stages. Pre-harvesting factors such as soil water, soil fertility, amount of sunlight, weather, and so on determine yield (quantity produced or output), which is the most desired component in this stage.

The influence of yield extends beyond the pre-harvesting stage and serves as the foundation for all post-harvesting planning (supply and demand). Demand is defined as the quantity of goods or products that customers are willing to buy. Demand is an important factor in the post-harvesting stage because it can increase customer retention, reduce operational costs (transport, storage, labor, etc.), reduce waste, and increase revenue and profit, which is the primary goal of organizations.

Demand is critical; its uncertainty is a concern because of factors such as price changes, seasonality, and so on. To that end, demand can be viewed as central to agri-food supply chain decision-making. Forecasting is important for all decision-making tasks, according to Fatious et al. (2014), from inventory management and scheduling to planning and strategic management.

Forecasting agricultural product sales is difficult due to the factors mentioned above. It is even more difficult for the fresh produce (FP) category of agricultural produce because of their susceptibility to spoilage, shorter life-cycle, and loss of quality and freshness as they remain on the shelf for longer periods of time. This is a major issue with FP sales that necessitates a cutting-edge solution.

In this study, the Long Short-Term Memory neural network (LSTM) and the hybrid Convolutional Neural Network with LSTM (CNN-LSTM) are used as base line models for predicting and forecasting future demand trends of FP using a twelve-year old strawberry dataset from the US Department of Agriculture. The research of Okwuchi et al. (2020) led to the decision to consider these models.

The two models were improved by incorporating a self-attention layer. For the LSTM model, in addition to the manual hyperparameter tuning approach, the Bayesian process of automatic hyperparameter determinant was implemented and the results of the two techniques were compared. The coefficient of determinant score (R2-Score) and the mean absolute error (MAE) were used as the yardstick in the three experiments conducted to determine the best performing model.

1.2 Problem Statement:

It is estimated that 1.3 billion tons of food intended for human use are lost or wasted globally each year (FAO, 2011). The financial costs of such waste and the enormous sums of money lost when food is not consumed by people were acknowledged by Ventour (2008). In a world where it is encouraged to turn waste into wealth, producing waste (via food loss) from wealth (as measured by the monetary value of food) is counterproductive for businesses and others.

Food loss is common along the entire food supply chain (FSC). Losses at the wholesale and retail level are one of three levels along the FSC that were mentioned by Buzby et al. (2009). They identified a number of factors, including inadequate supply and demand analysis and transportation infrastructure, as the architects of the losses at this level. In essence, an outstanding demand analysis and projections would have considered effective transportation systems as a constraint. Therefore, ineffective demand planning can be considered to be the main factor causing food loss at the retail level.

Planning the demand efficiently would necessitate making predictions about expected future sales over a number of forecast horizons. For FPs, this can be challenging because their pre-harvest stage conditions have an impact on them, and even at the delivery stage, FPs need specialized transport conditions in addition to many other contributing elements. Profit maximization is every company's objective. However, there are several limiting factors, making it difficult and requiring complex approaches to forecast the likely demand for FP. In this study, the expected demand for weekly strawberries across the United States is modelled using the number of retail outlets announcing the availability of at least one unit of strawberry for sale every weekend on the United States Department of Agriculture (USDA) website.

1.3 Motivation:

The retailing unit of the food supply chain (FSC) has a big impact since it is the one that is most in close proximity to consumers and helps to understand their behavior. It provides an insightful setting for resolving demand-related problems because it is the go-to unit for identifying the heart of consumer needs.

Fruits and vegetables are most affected by food loss in supermarkets despite their significant contribution because they are more prone to spoiling, have a shorter shelf life, and are considerably more expensive to maintain. Loss of FP in this situation could result in understock: less product is available to satisfy consumer demand. Customer retention may suffer as a result.

On the other hand, overstock—more product than is needed by customers—has its own financial repercussions. It raises storage and preservation costs and, regrettably, given the nature of FP, may ultimately result in waste. This has the result of raising operating costs and lowering profits.

Whether to encourage customer retention or prevent profit minimization, achieving optimality in demand forecasting is the key to finding a balance between these two important but competing components. Specifically, estimating future demand that is neither too low to risk losing clients nor too high to create debts. This is the foundation from which the desire to conduct this investigation springs.

Traditional timeseries approaches have been used to pursue this route (forecast) in the past, but nonlinearity in the data is a limiting factor that renders traditional methods ineffective and makes the employment of high-end machine learning (ML) techniques desirable. The adoption of these ML techniques has been facilitated by the data deluge and reasonable access to computational resources, which offer not just an improved decision support system for merchants looking to estimate probable demand, but also the ability to mitigate the downsides of overstocking and understocking.

1.4 Research Scope:

The goal of this thesis is to create, improve, and pinpoint the best ML-based predictive models required to address the problem of retail outlets forecasting fresh produce demand. Key presumptions were made and strawberry data from the USDA was used. Convolutional Neural Network with LSTM (CNN-LSTM or C-LSTM) and Long Short Term Network (LSTM) are the two main models that have been developed. The models were tweaked in an effort to optimize them, and different combinations of their hyper-parameters have been logged to an online machine learning flow (MLFlow) server. To improve the models, a self-attention layer was also added.

In particular, Bayesian optimization was utilized to automatically tune the LSTM model just to find the optimal setting for the hyper-parameters. The results were seen and contrasted with the manual method. The highest performing models (including the optimized models) were identified after a thorough experiment and evaluation utilizing mean absolute error (MAE), coefficient of determination (R2-Score), and mean squared error (MSE).

1.5 Research question, Aim and Objectives

Research question: When using the coefficient of determination (R2-score) as a performance metric on the test dataset and the fundamental criteria for selection, which model—Long Short Term Memory (LSTM) or Convolutional Neural Network with LSTM (CNN-LSTM)—will perform better in predicting and forecasting the likely demand for strawberries after optimization with a self-attention mechanism?

Aim: The main aim of this dissertation is to develop and test the performance of ML-based forecasting techniques for improved demand decision and planning for the retailers (supermarkets, etc.) dealing with the sales of this fresh produce (strawberries). To achieve this aim, the following **objectives** are pursued:

- 1.) Gathering of data relevant to strawberry sales and identification of key features.
- 2.) Building and optimization of the selected models for the forecast of likely demand for strawberries.
- 3.) Experimentation, model evaluation, and comparison to come up with a high-performing model.

1.6 Thesis Organization

This thesis comprises of five chapters. This present chapter reviews the importance of retail stores in understanding the consumer's buying pattern and the challenges of fresh produce demand planning. The business effects of over and under planning demand were emphasized, and optimality was promoted as ideal. It went on to describe the challenges of traditional timeseries forecasting and went to state the research scope, research question, aim and the dissertation's objective.

Chapter 2 follows immediately, and it covers the background and literature review. It reviews the management of the fresh produce supply chain, the enablers and inhibitors of FP, time series modelling, deep learning models and architectures, pros and cons of some selected performance metric, review of some related works and ended with summary of the chapter.

The implementation of solutions and methodology, as well as the descriptions of the models, are covered in chapter 3. In Chapter 4, the experiment's setup and concept are covered, along with the main justification for the experiment's logic and a series of mini-scenarios designed to help the experiment go smoothly. Chapter 5 ends the study and analyses its limits as well as the findings.

Chapter 2

2.1 Background and Literature

This chapter examines the key themes of the thesis in depth. The themes under consideration are Fresh Produce (FP) supply chain management, the influencing factors that affect FP demand, the concept of optimal FP order quantity, related techniques germane to demand forecasting, specific deep learning architectures and models useful in timeseries forecasting, performance evaluation and justification of regression models used in this task.

2.2 Management of the fresh produce (FP) supply chain

The resulting outcome of inept supply planning breeds financial and material lost. Especially in the “big market sales” environment, a market condition in which the supplier and retailer are far apart (Juning Su, Jiebing Wu, and Chenguang Liu, 2013), decision making in this environment can be challenging. Furthermore, FP, the commodity of interest in this thesis, more to its dynamism in trading, consumers are also keen on the attributes that enhance its market value; Freshness and availability.

In agreement, Willem A. R et al. (2013) noted that consumers' demand for year-round availability of fresh products in retail outlets is consistent. The costs of inventory, transportation, and preservation, among other things, are limiting factors in achieving this. Inability to achieve the goal serves an unfavorable purpose: deviation from customers' pain points, which can have severe consequences. Leveraging cutting edge techniques in sales and market planning is key.

This thesis will add the following value to the growing research interest in perishable food supply chains (PFSC):

- Contribute to the global effort to reduce food waste, which is estimated to be 40% to 50%. (2011) (Gustavsson et al.).
- Contributes to the body of knowledge on the utilization of contemporary ML models in FP forecasting.

2.2.1 The enablers and inhibitors of fresh produce demand planning

The previous section discussed the 'big market sales environment' and how it makes PFSC decision making difficult. Customers' needs and preferred state of FP quality were also highlighted.

The cost element is a significant factor in PFSC; failing to consider it has far-reaching consequences. Malleswari Karanam et al., (2022) demonstrate the impact of opportunity and transportation costs in determining the value degradation in PFSC in their research. Concerning FP and their freshness as the quality of attraction, suppliers work tirelessly to ensure safe and timely delivery of this produce, with transportation cost minimization as the objective function. However, the supplier's lack of proximity to the targeted retailers prevents this objective from being mostly realized.

To address this issue, researchers proposed an intelligent routing system that would reduce the transport-cost element that influences FP demand. Hsiao (2018) used a genetic algorithm (GA) to solve a vehicle routing problem (VRP) with a time constraint, taking into account individual characteristics of many perishable items, their ongoing quality decline, and optimal temperature settings during transportation. Hsiao's (2018) work underpins the importance of optimizing transportation systems in FP demand, which aligns with the logic of this thesis: the proximity of FP shipping stations to retailers can reduce quality degradation and preserve freshness, the value sought by consumers.

Furthermore, the energy cost is a factor in PFSC and a critical variable in total logistic cost, which is also an important goal (minimization) of the VRP. To achieve this goal, Leng L. et al. (2020) used the multi-objective evolutionary algorithm. In most cases, the freshness of FP determines its quality. As a result, constant preservation at a safe temperature is deemed necessary, and Malleswari Karanam et al. (2022) argued that perishable products necessitate the use of additional fuel to maintain their quality. Awad, M et al., (2020) and Stellingwerf, HM et al., (2018) argue for a cost-effective goal: determining the path that uses the least amount of fuel for refrigeration and traction.

Moreover, using fuel with a minimalist focus not only saves cost, but it also improves the sustainability of PFSC. Along with the cost-elements of PFSC that have already been identified as high in this thesis, Samir G. et al. (2018) identified

climate dependence as another high characteristic element. Their study confirms the findings of Malleswari et al., (2022) and expands (Ali et al., 2010) study on prices, availability, seasonality, residual shelf life of FP, prices, and price of substitute produce as determinants of FP demand.

Climate conditions have been studied for their impact on FP quality. According to L. Jaxsens et al. (2009), ongoing climate change and increasingly complex international supply chains are expected to become factors in the near future that could affect the performance of food safety management systems (FSMS). This thesis does not investigate how climate change affects food safety. Its indirect relationship to the freshness and quality of FP, on the other hand, is acknowledged to be influential in FP demand planning.

Temperature rise, precipitation variation, and atmospheric carbon dioxide were identified as a few factors relevant to food safety by Miraglia et al. (2009). As a result, they discovered that socioeconomic changes caused by climate change could have an indirect impact on the food supply chain. Technically, their position is consistent with the thesis's logical proposition: because the quality and, by extension, the purchase of FP is determined by its "freshness," climate-induced socioeconomic changes (such as temperature, humidity, and so on) are critical not only to FP demand planning, but also to timely shipment of the produce to retailers to ensure availability—another factor that influences consumer choice.

Techniques have been deployed in literatures to ensure the perishability of FPs is preserved from distribution centers and/or retailers to end users while minimizing associated costs (shipment, energy, etc.) without jeopardizing the sustainable environment. Soysal et al. (2015) used a model to simulate a real-world supply chain in which a distribution center delivers fresh tomatoes to stores in their work that resulted in the optimization of metrics such as total cost and carbon emission.

Bortolini et al. (2016) investigated the impact of the distribution system on the perishability of FP. Their research, which focuses on Italian farmers, employed three distinct modes of transportation, leading to the development of a distribution planning system. This system accomplished two goals: regulating product perishability and lowering carbon dioxide emissions. Solina and Mirabelli

(2021) developed an optimization model in the same bid with the primary goal of reducing expenses and energy consumption.

The reviewed literature revealed that the freshness of FP is a yardstick for determining their quality: the most influential component that influences customer choice. Furthermore, the long-term viability of FSC and its indirect impact on demand were investigated. Indeed, some optimization models have been proposed, with transport, shipment, sustainability, and a variety of other factors modelled as constraints along the chain. While increased demand drives sales and possibly revenue at the distribution or retail unit, expenses associated with timely transport and storage at a regulated temperature, carbon emission reduction, and so on prior to shelving FP can reduce the possibility of maximizing profit via overstocking or understocking of demanded quantity.

To that end, a solution capable of forecasting probable demand would be ideal for reducing operational costs and waste caused by under and overstocking. However, demand uncertainty poses a problem. (Simangunsong et al., 2012) proposed two approaches to dealing with uncertainty. The first is to reduce demand uncertainty through a suitable pricing strategy, thereby mitigating uncertainty at its source. The second method is to deal with uncertainty by employing advanced forecasting techniques to mitigate the negative impact of demand uncertainty. The latter approach will be used in this thesis and will be discussed further below.

2.3 Time Series

The goal of time-series modelling is to forecast future output over a given time period. This is critical to the dissertation's goal, so before moving on to the predictive and or forecasting techniques used in this work, the two most well-known timeseries modelling methods and their inherent characteristics are briefly discussed.

2.3.1 Univariate Time Series

A univariate time series is a scalar (or one feature) of observations that are captured in the same time increments sequentially. (C.J. Cuaresma et al.,2004) and (J. Contreras et al.,2003) provided examples of using previous agricultural commodity prices to predict future prices. A key assumption in the univariate method, as proposed in the Naive Forecasting model, is dependency. That is, the next period is assumed to do the same as the previous period (J. Contreras et al., 2003).

One of the characteristics of univariate timeseries is **autocorrelation**. Simply put, it is the similarity of observations as a function of their time lag. Time lags could be plotted in minutes, hours, days, weeks, and so on. The Pearson correlation of a signal with a delayed copy of itself as a function of delay would be revealed by the lag plots. This allows for a thorough examination of whether a univariate timeseries should be modelled daily, hourly, or weekly. In practice, the logic is to find a duration with a moderate correlation: one that is neither too correlated nor too uncorrelated in order to improve modelling performance.

Trend is another characteristics. (A.T. Jebb and L. Tay., 2016; E.S Gardener and Ed. McKenzie., 1985) defined it as a systematic linear or nonlinear component of timeseries that changes over time and does not repeat.

Seasonality is also assigned to univariate timeseries data. It detects variations that occur at specific regular intervals of less than a year, such as weekly, monthly, or quarterly. Seasonality affects forecasting and is aided by factors such as weather. It is made up of periodic, repetitive, and generally regular and predictable patterns in a timeseries' levels (A. M. Davey and B. E. Flores, 1993).

Finally, the most important feature of univariate timeseries is **stationarity**. Stationarity is achieved when a timeseries has a constant mean and variance that is independent of time. A stationary time series is defined (D. Kwiatkowski et al., 1992) as one whose properties do not depend on the time at which the series is observed. A timeseries with a trend or seasonality, is considered non-stationary. Non-stationary models for short-term forecasting horizons include generalised autoregressive conditional heteroskedasticity, or GARCH (ES. Gardner, 1985), and auto-regressive integrated moving average, or ARIMA (ES. Gardner, 2006).

2.3.2 Multivariate Time Series

In contrast to the univariate timeseries, the multivariate timeseries has more than one time-dependent variable (i.e., two or more features). N. H. Chan (2001) noted in his work that each of these variables is distinct in that it is dependent on its previous values and is a function of other variables.

More input variables are assumed informally to improve the model's performance. However, J. H. Stock (2001) recommended an empirical approach supported by domain knowledge to determine whether multivariate series are appropriate for the intended task. He went on to argue that extensive experimentation is required to determine the best approach between univariate and multivariate series.

Based on this logic, while domain knowledge was used to determine relevant features in this thesis, a three-stage statistical approach was used to select the most important features required for modelling. This method will be covered in the following chapter under the proposed solution.

2.4 Forecasting and Predictive Techniques

Although the same techniques are used and the terms are used interchangeably, it is important to distinguish between forecasting and prediction. Forecasting is a subset of prediction, according to Teja K. (2019), and it applies to out-of-sample observations, whereas prediction applies to in-sample observations.

Teja continued by stating that; "for each observation in the sample used to estimate the regression, predicted values are calculated. Forecasts, on the other hand, are made for some dates that are beyond the data used to estimate the regression ". To summarise, forecasting makes use of past and current trends to predict future outcomes. It implies the passage of time and the future, whereas prediction does not. Prediction and forecasting tasks are accomplished in this dissertation.

Traditional forecasting methods such as ARIMA were used in determining future estimation prior to the era of big data and sufficient computational resources. However, the bottleneck is their assumption that all data is linear. Machine learning (ML) methods were gradually adopted, and their ability to handle non-linearities in data (H. Demuth, 2009) make them better.

Furthermore, the concept of demand and its influencing factors introduce volatility into its modelling, reducing the effectiveness of good models and even those thought to be better. This study, for example, considers factors such as price, number of stores, and so on that can affect market availability of at least one unit of strawberries and attempts to find a relationship between these factors and demand for at least one unit sale of strawberries based on the number of stores advertising. Thus, ML models, like time series, can be regarded as multivariate models (Okwuchi et al., 2020) and are implemented as such in this dissertation.

ML models are classified as either supervised or unsupervised. Unlike unsupervised learning, which is commonly used for clustering and is trained without labels, supervised learning is accomplished through the training of influential factors known as inputs or independent variables and their corresponding labels or target variables. Supervised learning is used to solve regression and classification problems. This work is a supervised learning and regression task that employs Deep Learning (DL), a family of ML techniques. The section that follows goes over the DL methods and some of the architectures used in this dissertation.

2.5 Deep Learning

Deep learning, also referred to as deep structured learning or differential programming, is a subset of machine learning (ML) that is based on artificial neural networks (ANN) and other architectures. The ability of DL to perform representational learning (RL) is an instinctive feature. In their published work "Deep Learning," I. Goodfellow et al. (2016) defined RL as a set of methods that, in addition to accepting raw data as input, automatically discover representations in the data required for detection or classification. According to Y. Bengio et al. (2013), DL representation methods have multiple levels of representation that are achieved by combining simple non-linear modules that each transform the representation at one level into a representation at a higher, slightly more abstract level.

The effectiveness of DL techniques based on ANN is significant. Prior to this study, they were widely used in image detection, computer vision, speech recognition, natural language processing, and other applications before being applied to time-series tasks, as is done in this study too.

Although they are mostly efficient in some use cases, one of their drawbacks is the high cost of computational resources. The ANN will be discussed next, followed by some of its selected subsets such as the recurrent neural network (RNN), under which the long-short-term memory (LSTM) will be explored, the convolutional neural network (CNN), and the attention mechanism, all of which are used in this study.

2.6 Artificial Neural Networks (ANNs)

The numerical capability of ANNs in approximating non-linear continuous relationships is well known (Nelson, 2015). This capability makes it possible to model ever-increasing amounts of data, necessitating the use of better enhanced forecasting models rather than the well-known ARIMA. Farouk (2010) reported that an early approach that uses a feed-forward neural network to model non-linear dependencies based on residues of sequential data, with the assumption that time-series data has both linear and non-linear dependencies was introduced.

ANNs are supervised ML that perform tasks without requiring specific task-related rules (Y. Chen et al., 2019). They are inspired by biological neural networks found in animal brains. Neurons in ANN are a network of interconnected nodes that receive a signal, process it, and transmit it to another neuron. The input, a real number, is transformed nonlinearly to produce an output.

As learning progresses, the weights of the network are used to connect neurons and are updated accordingly. An important component of the ANN is the layer: a stack of neurons at the same level. Various transformations can be performed at different layers, which means that the output at each layer may differ.

The ANN learns by minimising the cost function, which is the difference between predicted and actual value. Errors are discovered during learning, and the backpropagation technique is used to adjust the connection weights to compensate for each error. The error is then distributed to the various connections. Backpropagation precisely computes the derivative of the cost function associated with a given state with respect to the weights (which are updated using stochastic gradient descent). In this thesis, how well the implemented models minimize error was used factor in the decision process. The ANN architecture is depicted in Figure 2.1.

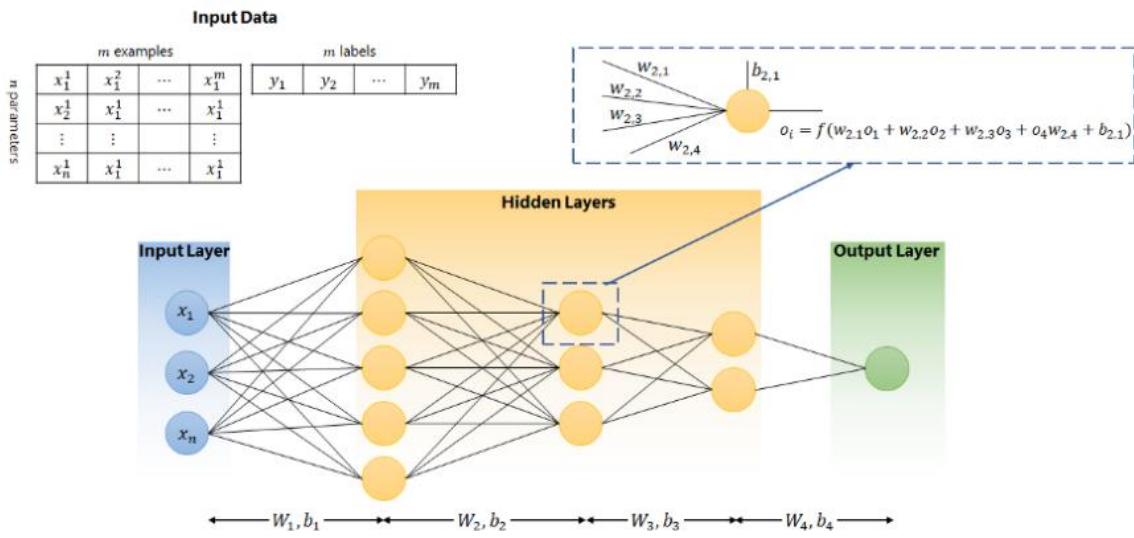


Figure 2.1: A simple ANN architecture 1

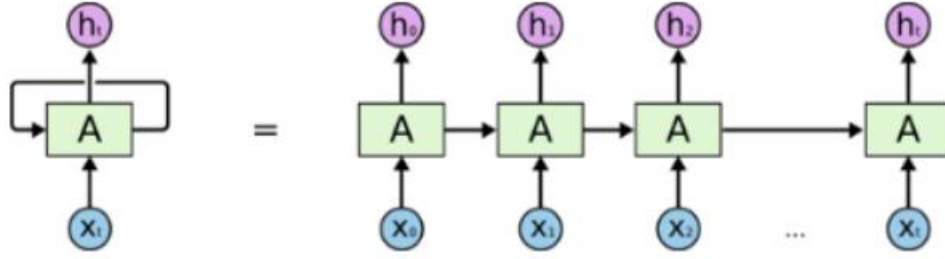
Figure 2.1: A simple ANN architecture (image by: Diego Manfre)

2.6.1 Recurrent Neural Networks (RNNs)

Recurrent neural networks are an ANN class in which node connections form a directed graph along a temporal sequence. Because of this graph, the RNN can exhibit temporal dynamic behaviour (P. Murugan, 2018). Murugan went on to explain that RNNs are derived from feedforward neural networks and that they can process variable length sequences of inputs using internal state memory.

Z. C. Lipton et al. (2015) described the RNNs' mathematical process, and figure 2.2 depicts the computations required for the forward steps. At time t , nodes with recurrent edges receive input from the current data point x_t as well as hidden node values $h_{(t-1)}$ from the previous state of the network. Given the hidden

node values $h_{(t)}$ at time t , the output $\hat{y}_{(t)}$ is calculated at each time t . The input $x_{(t-1)}$ at time $t-1$ can have an effect on the output $y_{(t)}$ at time t .



An unrolled recurrent neural network.

Figure 2.2: An unrolled RNN 1

Figure 2.2: An unrolled RNN (image credit: C. Olah's post)

The computations of the hidden node $h_{(t)}$ and the output $y_{(t)}$ are shown in Equations 2.1 and 2.2, followed by a description of each parameter.

$$h_{(t)} = \sigma(W_h x_{(t)} + W_{hh} h_{(t-1)} + b_h) \quad (2.1)$$

$$\hat{y}_{(t)} = \text{softmax}(W_y h_{(t)} + b_y) \quad (2.2)$$

The logistic function, σ , also known as the activation function, is a non-linear transformation that accepts any real number and returns a value between 0 and 1. W_h is the conventional weights matrix between the input and the hidden layer, while W_{hh} is the recurrent weights matrix between the hidden layer and itself at adjacent time steps. The bias parameters b_h and b_y enable each node to learn an offset (Z.C. Lipton et al., 2015).

The predicted next value of the sequence is represented by the output $\hat{y}_{(t)}$. An RNN can also be thought of as a deep network with a layer for each time step and shared weights between them. RNNs are considered similar to traditional time-series models because they can model both temporal and time-based relationships in data. However, the exploding and vanishing gradients pose a challenge to RNN's ability to model sequences. This limitation gave rise to the long-short-term memory network (LSTM). The LSTM is covered next.

2.6.1.1 Long Short Term Memory Network (LSTM)

Long Short Term Memory networks were developed to mitigate the effect of vanishing and exploding gradients in RNNs, as discussed above (S. Hochreiter et al.,). To overcome these constraints, the LSTM employs a memory cell comprised of nodes with self-connected recurrent edges of fixed weight, ensuring that the gradient can pass through it many time steps without vanishing or exploding. F. A. Gres et al. (2002) described the LSTM's input gate, memory, and output gates as outlined below.

- **Input Gate:** To modify the memory, it selects the most important values from the input. With the help of the sigmoid function, the selected values are transformed to be between 0 and 1. The tanh function, on the other hand, assigns weightage to the values that it receives and thus determines their level of importance, which ranges from -1 to 1. The input gate (i_t) and the candidate for cell state (C_t'). are represented by Equations 2.3 and 2.4, respectively.

$$i_t = \sigma(W_i [h_{t-1}, x_t] + b_i) \quad 2.3$$

$$C_t' = \tanh(W_c [h_{t-1}, x_t] + b_c) \quad 2.4$$

C_t' is the candidate cell; b depicts bias; W represents weights and h_{t-1} is the output of the previous LSTM block at time stamp $t - 1$.

- **Forget Gate:** It is significant in identifying irrelevant contents present in the internal state, as introduced by Gers et al., (2002). It takes values from the previous hidden state in addition to the current input. It is represented by equation 2.5 below.

$$f_t = \sigma(W_f [h_{t-1}, x_t] + b_f) \quad 2.5$$

- **Output Gate:** The memory cell generates the hidden node value (h_t) (see equations 2.6, 2.7 and 2.8), which is estimated by multiplying the value of the internal state by the value of the output gate O_t . Internal states are frequently run through a tanh activation function (Okwuchi et., 2020) because it gives each cell's output the same dynamic range as an ordinary tanh hidden unit. However, rectified linear units (RELU) were used in this

study due to their ease of training and wider dynamic range (C. Olah., 2015). The LSTM with the gates is shown in Figure 2.3.

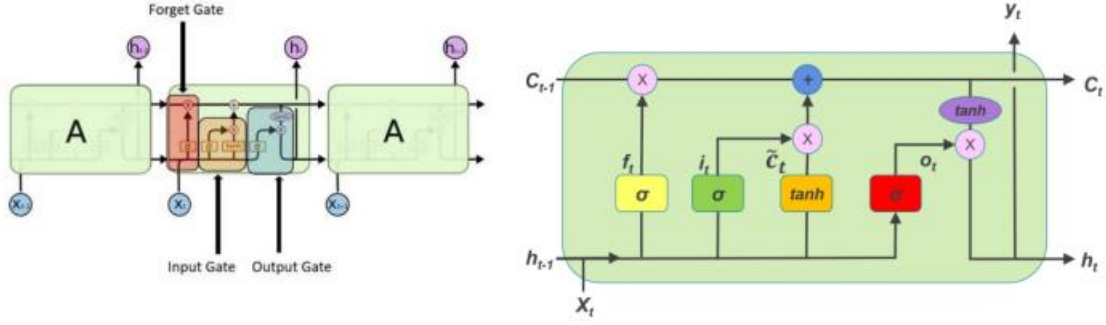


Figure 2.3: LSTM cell showing the three gates (credit: C. Olah.,2015)

$$O_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad 2.6$$

$$C_t = f_t * C_{t-1} + i_t * C_t' \quad 2.7$$

$$h_t = O_t * \tanh(C_t) \quad 2.8$$

So far, LSTM has been mentioned as a method for mitigating RNN's exploding and vanishing gradients. The vanishing gradient can still occur even with LSTM. As a result, researchers proposed the attention-based mechanism. This mechanism was used in this study in a bid to optimize the performance of the LSTM model. Following that is a discussion of the attention mechanism.

2.6.2 Attention Mechanism

The goal of attention mechanisms is to assist the model in focusing on the most important parts of the input sequence for each output. The attention function, according to A. Vaswani et al. (2017), is a mapping of a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors.

Attention aids in the resolution of the vanishing gradient problem by providing a direct path to the inputs (Anusha Lihala, 2019). This thesis made use of the self-attention mechanism out of many other variants of attention mechanism. With self-attention, each hidden state attends to the RNN's previous hidden states. Other attention mechanisms include additive attention, multi-head attention, and so on. Finally, the convolutional neural network (CNN) will be discussed as the final ANN architecture implemented in this thesis.

2.6.3 Convolutional Neural Networks (CNN)

CNNs are commonly used in image classification (J. Wang and Y. Hu., 2020), natural language processing (R. Collobert et al., 2008), recommender systems (A. Vandennoord et al., 2013), and, more recently, time series modelling (A. Tsantekidis et al., 2017). How does a well-known image detection network support time series modelling? (ref number: 52) reported that patterns in a sequence of time series data would be recognized by CNN regardless of where they appeared in the sequence.

This is because CNNs can use shared weight and translation invariance. If every pixel in an image is moved in the same direction the same number of times, the image retains its original property and is recognized as the same thing (ref number: 52). This explains the usage of CNN in time series data.

CNN's max pooling layers can reduce the risk of overfitting, even though the network is not completely immune to overfitting (Vincent Tatan, 2109). By reducing the number of neurons, this max pooling layer enables more automatic extraction of important features. The logic in this dissertation was guided by this feature extraction ability to place CNN layers over the LSTM in order to compare the effect of the spatial-temporal relationship to the temporal ability of the LSTM alone.

The Rectified Activation Unit (ReLU) in CNN mimics neuron activations to introduce nonlinearity for values $x > 0$ and returns 0 if the condition is not met. This method has proven to be effective for solving decreasing gradients (Vincent Tatan, 2019). Figure 2.4 shows the CNN basic architecture.

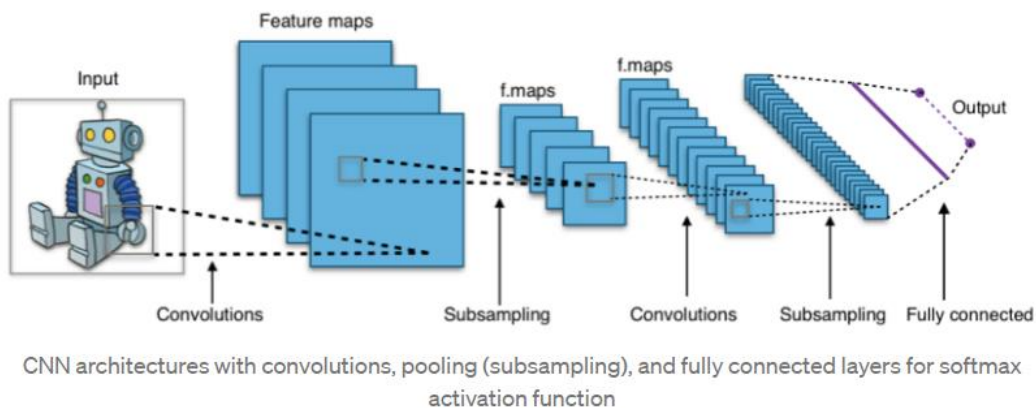


Figure 2.4: CNN architecture (Vincent Tantan, 2019)

Finally, the CNN has hidden layers composed of convolutional layers that convolve with a dot product. Kernel convolution is a critical component of CNN. A sliding window was used to create the kernel that passed over the 1D input sequences in this study. CNN receives a tensor as input. As a result, the input sequence in this work has been modified to generate a tensor.

2.7 Performance Metrics

The evaluation of machine learning models is a critical component of the process. Evaluating the performance of each model provides a solid foundation for comparison in a supervised learning regression problem. Most metrics used for regression tasks in the existing literature (N. K et al., 2010; S. Jharkharia & M. Shukla et al., 2013; R. J. Hyndman and A. B. Koehler, 2006) are the Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Mean Squared Error (MSE), Mean Absolute Percentage Error (MAPE), and the R2-score: a measure of the degree of correlation between the predicted and actual values (D. L. Alexander et al., 2015). The three metrics used in this thesis are outlined below:

- **Mean Squared Error (MSE):** The MSE is commonly used in regression tasks. It is the mean of the squared difference between the target and predicted values from the regression model. The concept of squared differences implies a high penalty for even minor errors, and an even higher penalty for major ones. It is significant because it has a distinguishing feature that allows for optimization. Unfortunately, MSE is susceptible to outliers (S. Jharkharia and M. Shukla, 2013; R. J. Hyndman & A. B. Koehler, 2006). In this dissertation, the MSE was used as the loss function to be minimised. Equation 2.8(a) is used to estimate the MSE.

$$MSE = 1/n \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad 2.8(a)$$

Where y_i is the i_{th} true value, \hat{y}_i is the i_{th} prediction and n is the sample size.

- **Mean Absolute Error (MAE):** The MAE, which is also commonly used in regression, estimates the absolute value of the difference between actual and predicted values, adds these absolute errors for all examples, and divides the total by the sample size. According to (C.J. Willmont & K. Matsuura), MAE is preferred over MSE and RMSE due to its robustness to outliers, as well as being unambiguous and natural. Its disadvantage is that

it is not differentiable. The goal of this dissertation is to track the model with the lowest MAE on the test dataset over a specified range of epochs. Equation 2.8(b) is used to estimate the MAE.

$$MAE = 1/n \sum_{i=1}^n |y_i - \hat{y}_i| \quad 2.8(b)$$

Where y_i is the i_{th} true value, \hat{y}_i is the i_{th} prediction and n is the sample size.

- **Coefficient of Determination (R2-score):** Based on the proportion of total variation explained by the model, the R2-score provides a measure of how well observed outcomes are replicated by the model (R. G. D. Steel & J. H. Torrie, 1960; N. R. Draper, and H. Smith, 1998). The R2-score is used to assess the goodness of fit of any linear least square regression model with an intercept. It has a value between 0 and 1.

In extreme cases, the score can be negative, indicating that the mean of the dataset fits the dependent variables better than the values provided by the model and that there is a complete lack of fit, according to T. O. Kvalseth (1985). This can also indicate a mismatch between a specific model and the regression problem under consideration. If the residual sum of squares is low, the R2-score is close to one; if the R2-score is close to zero, the residual sum is high.

One of the primary goals of this research is to experiment, compare and identify the best performing model. As a result, the R2-score is chosen as a metric to accomplish this because it provides a relative idea of model performance on a scale of 0 to 1, as opposed to the MSE and MAE, which provide absolute values and may not reveal model performance.

The R2-score has a few limitations, including not accounting for collinearity and not knowing if enough data points were used. To address the issue of multi-collinearity, the data preprocessing portion of this work used the variance inflation factor to select input features with low or no correlation among themselves. Equations 2.8c, 2.8.d and 2.8e defines the R2-score used in this dissertation.

$$ress = \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad 2.8(c)$$

$$tsos = \sum_{i=1}^n (y_i - \bar{y})^2 \quad 2.8(d)$$

$$R2\text{-score} = 1 - ress/tsos \quad 2.8(e)$$

“ress” is the residual sum of squares; “tsos” is the total sum of squares; y_i is the i_{th} true value; \hat{y}_i is the i_{th} prediction and \bar{y} is the mean of the actual values.

2.8 Related Work

A cursory examination of related works reveals significant progress in forecasting many agricultural products using traditional time-series models and advanced machine learning models. The ARIMA model was used to predict wheat yield by M. Ray et al. (2016) and L. Michel et al. (2013). Statistical models were used to forecast rice production by (A. Shabru, R. Samsudin, and I. Zuhaimy, 2009).

K. Yamamoto et al., (2018) published articles predicting fresh produce in relation to this thesis and the commodity under consideration. Okwuchi et al. (2020) thoroughly investigated the use of traditional and advanced deep learning models in forecasting strawberry yield in relation to weather and oil prices. While their research focused on yield forecasts, a pre-marketing stage in agriculture, this dissertation work concentrated on predicting and forecasting likely demand, a post-harvest operation that is a key determinant in marketing plans and a major driver of business revenue.

Furthermore, this thesis makes use of a dataset from multiple retail stores that advertise at least one unit of strawberry. This implies a representation of the customers' purchasing habits, which in turn dictates changes in annual yield. Thus, through structured experimentation, this work bridges the gap in post-harvest marketing by forecasting probable demand and identifying a better model, after optimization between the two in the outlined research question.

2.9 Summary

This chapter began by highlighting the research themes, then provided an extensive overview of the FP market, and even discussed the cost-elements of FP and how important they are in decision making. It went on to explain some of the factors that impede accurate demand planning, as well as what researchers have done to address this through proposals of various cost optimization models.

The distinction between univariate and multivariate timeseries data was explained, as well as the distinction between prediction and forecasting. A thorough report on ANN, RNN, and some specific deep learning models, as well as the optimization techniques used in these models, was justifiably highlighted. The limitations and strengths of the majority of techniques, as well as the performance metrics used for regression tasks, were discussed, followed by a brief review of related works.

Finally, the research hypothesis was stated, and the following is the research's contribution: this work bridges the gap in post-harvest marketing by forecasting probable demand and identifying a better model between two models: the LSTM, which models data's temporal relationship, and the CNN-LSTM, which models data's spatial-temporal relationship.

Chapter 3

3.1 Methodology and Implemented Solution

Choosing between the two proposed models for predicting and forecasting the commodity of interest in this study requires careful consideration of the most likely contributing factors. The factors considered in this chapter (domain knowledge, dataset, models, model optimization, and model evaluation) have a legitimate influence on the robustness of the intended solution. Each factor's detailed logic is explained next.

3.2 Domain Knowledge

The use of machine learning techniques is widespread. As a result, prior to selecting an appropriate feature for modelling, it is critical to have a good understanding of the specifics of the field in which an ML solution is intended to be implemented. The scope of this work does not include a detailed examination of demand and supply. However, a cursory review of marketing literature aided in narrowing the dataset used in this study.

Raplang Lapasam et al. (2021) found price to be the most important independent variable in determining the volume of market arrival when looking at both supply and demand sides of the equation in their study on factors affecting vegetable marketing. This demonstrates that price has a significant impact on demand. Other influencing factors include population growth, production level, per capita income, and so on. The section that follows goes over the data collection process, feature selection and reduction, key assumptions made, and some useful visual representation.

3.3 The Dataset

High-performance machine learning applications rely on data. The data (12years durations) used in this work was sourced from market terminal reports from the United States Department of Agriculture (USDA) ([link here](#)), and it has fifteen features, each of which is explained in this [document](#). Strawberry is the commodity of interest, and its data is updated every Friday across the country. As a result, the data was aggregated weekly prior to modelling. An attempt to aggregate it daily resulted in a negative R2-score, which is contrary to the goal of this dissertation.

The dataset did not include direct demand, or the total quantity of commodity sales. The number of stores advertising a unit of strawberry, on the other hand, was made available, and thus the target feature (likely demand) was engineered by multiplying the unit (1 pound) by the number of stores advertising. The number of stores advertising a unit of strawberry quantifies or measures the consumer's purchasing pattern in that locality.

As a result, the bottom line is that the greater the customer demand, the greater the number of stores or retail outlets advertising the sale or availability of at least one unit of strawberries. Figures 3.1 (12 year period) and 3.2 (a year period) depict this logic graphically, and the seasonality and uncertainty in demand trends were observed.

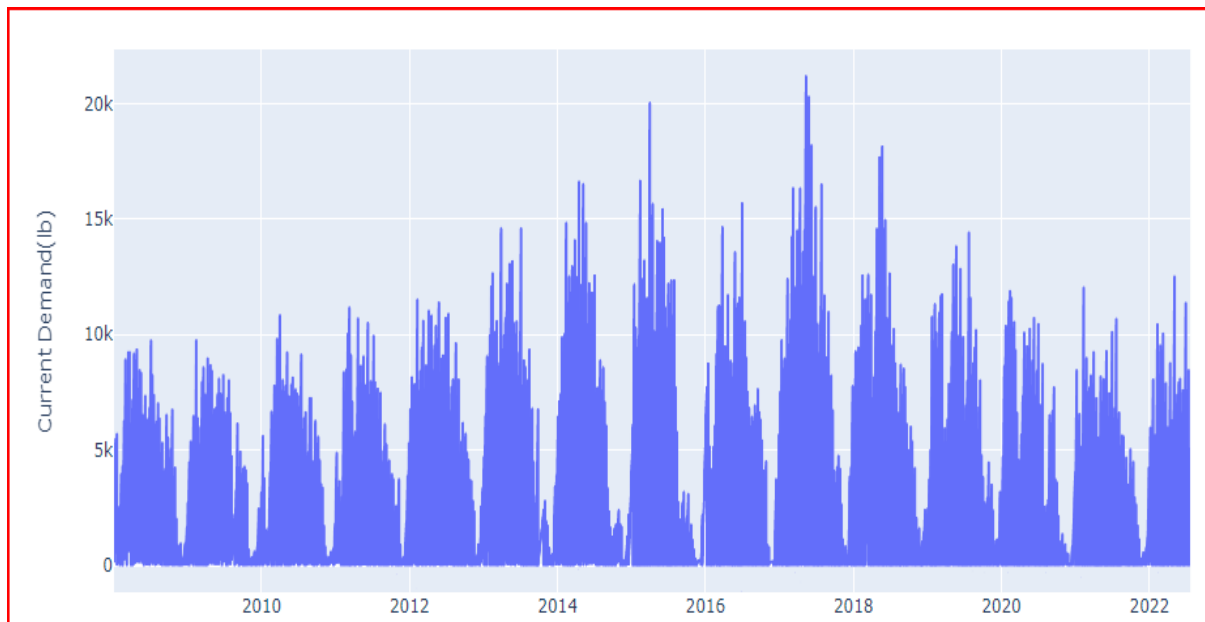


Figure 3.1: Assumed demand trend over a 12-year period.



Figure 3.2: Assumed demand trend over a year period

Seasonality and an irregular pattern are visible in the graphs. This supports the initial logic proposed. The following section describes the three-stage approach used in feature selection, which was hinted at in the previous chapter under section 2.6.

3.3.1 Feature selection and engineering

Relevant features can be a blessing in ML modelling, but irrelevant ones can be a curse. After the filling of missing values and preparing the data types in the desired format, the correlation heat map was created in the first stage using the Pearson correlation method to identify features that have a significant correlation to the target variable (current demand). Following that, the three features with the weakest correlation (<0.5) were removed. There is a problem with the retained features: multi-collinearity. This may have an impact on the model's performance.

As a result, the second stage of feature reduction got underway. The use of the Variance Inflation Factor (VIF), a statistical process aimed at determining the combination of features with the least collinearity, is the highlight of this stage. The VIF threshold is five. As a result, any retained feature with a VIF greater than 5 was removed. Price, as seen in figure 3.3, was discovered in this work to be a strong factor that determines demand in addition to previous demand. This is consistent with the findings of Raplang Lapasam et al. (2021).

	Variable	VIF
0	High Price	2.141686
1	Previous Demand(lb)	2.141686

Figure 3.3: The VIF of the retained variables

The final stage is the Xgboost feature-importance module implementation. The F-score of each feature is calculated and ranked using this technique. As a result of this, the "number of stores" was identified as a significant feature, and it was added as an input feature alongside the VIF-identified two. Even though it is the target variable, the estimated current demand also serves as an input because it can provide useful insight into the next week's forecast.

To create the engineered input data shape, four input features were combined with one target feature (760, 4). The sample size was reduced from 12000 to 760 after weekly aggregation. To fit the model requirement, the engineered data was further processed to fit into a three-dimensional shape of (753, 7, 4) which represents (sample, lookback, number of features). Features are neither normalized nor standardized because the implemented models include a "Batch Normalization Layer" that scales batch input. The following section goes over modelling techniques and architectures.

3.4 Models

Again, this work is a regression-based supervised learning, and the model selection is guided accordingly. Furthermore, the basic exploratory data analysis reveals non-linear patterns in the dataset, so deep learning (DL) models are used because of their ability to learn complex patterns in data as well as their ability to improve continuously as data size increases (I. Goodfellow et al., 2016). In this study, the DL model in section (2.6.1.1) and its hybrid with the model in section (2.6.3) are proposed for use. In an effort to optimize, a specific layer of the model in section (2.6.3) was added.

Reshaping the data into a three-dimensional form is an important aspect of this implementation. The data points of 7 weeks were used as the window size (the number of weeks or data points the model has to draw inference from the past to make future predictions) out of the 760 samples (samples reduced to 753 after deducting the window size) with four input features aggregated weekly.

Furthermore, the resulting data sample was divided into three parts: training, testing, and a random validation split that comprised 20% of the training set. For all models implemented, the procedure was the same. The remainder of this chapter will go over each implemented model's configuration, model optimization, tuning, and evaluation metrics.

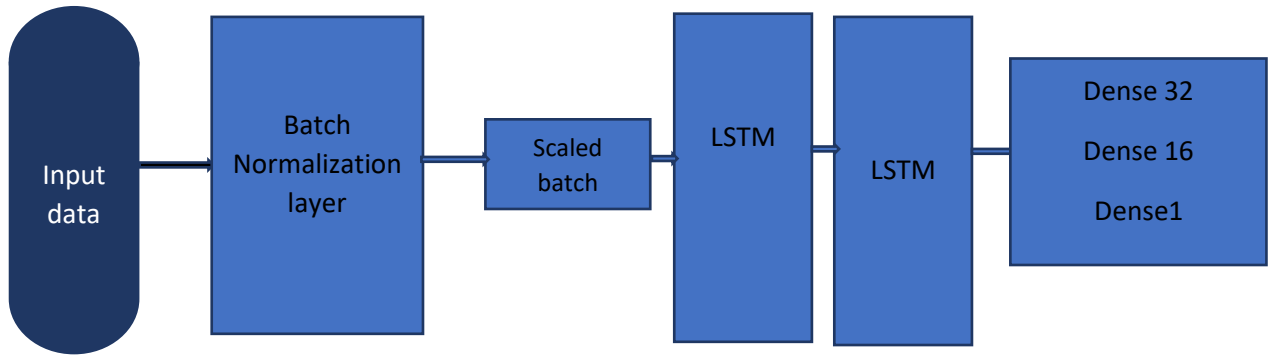
3.4.1 Model 1: Long Short Term Memory (LSTM)

The LSTM model's architecture begins with an input layer that feeds data directly into the batch normalization layer (as explained in section 3.3.1), followed by two LSTM layers with 300 and 128 units, respectively, followed by another batch normalization layer, followed by a 32-unit dense layer, a 16-unit dense layer, and a 1 unit dense layer. The ReLU non-linear activation function is used throughout, the loss function is mean squared error (MSE), and Adam is the optimizer. The MAE was used as the metric, and its minimum value was tracked throughout. Figures 3.4 and 3.5 show the summary and sketched architecture of the implemented model respectfully.

Model: "sequential_1"

Layer (type)	Output Shape	Param #
batch_normalization_2 (Batch Normalization)	(None, 7, 4)	16
lstm_2 (LSTM)	(None, 7, 300)	366000
lstm_3 (LSTM)	(None, 128)	219648
batch_normalization_3 (Batch Normalization)	(None, 128)	512
dense_3 (Dense)	(None, 32)	4128
dense_4 (Dense)	(None, 16)	528
dense_5 (Dense)	(None, 1)	17
Total params: 590,849		
Trainable params: 590,585		
Non-trainable params: 264		

Figure 3.4: Summary of the implemented LSTM model



2 LSTM Layers with Batch Normalization

Figure 3.5: Implemented LSTM model architecture

3.4.2 Model 2: Convolutional Neural Network with LSTM (CNN-LSTM)

The batch normalization layer is the first layer in this hybrid model's architecture, followed by a 1D convolution layer with a 64 filter, a kernel size of 7, and the same padding with ReLU activation function. This was followed by a size 2 1D max pooling layer. The following layer is a batch normalization layer, followed by a repeated 1D convolution layer with a 32 filter size, a kernel size of 3, a ReLU activation function, and the same padding.

Following that is a Maxpooling layer of size 2, followed by another layer of batch normalization (BN), and finally a 1D Convolutional layer with 16 filters, a kernel size of 7, ReLU activation, and padding all set to the same. The final 1D convolution layer is immediately followed by a 0.2 drop out layer before an LSTM layer of 300 units, which is then followed by another 0.2 drop out layer and 128 units of LSTM. The final three layers are 32 units, 16 units, and 1 unit dense, respectively. Figure 3.6 shows the model architecture.

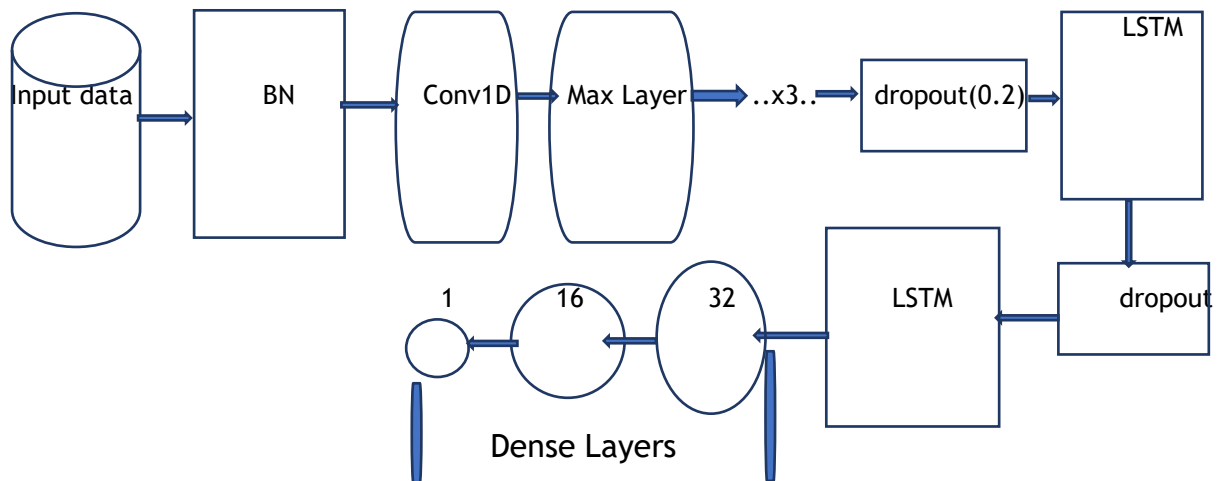


Figure 3.6: implemented CNN-LSTM model architecture

Figure 3.7 depicts the summary of the CNN-LSTM model design.

Layer (type)	Output Shape	Param #
batch_normalization_75 (Batch Normalization)	(None, 7, 4)	16
conv1d_57 (Conv1D)	(None, 7, 64)	1856
max_pooling1d_38 (MaxPooling1D)	(None, 3, 64)	0
batch_normalization_76 (Batch Normalization)	(None, 3, 64)	256
conv1d_58 (Conv1D)	(None, 3, 32)	6176
max_pooling1d_39 (MaxPooling1D)	(None, 1, 32)	0
batch_normalization_77 (Batch Normalization)	(None, 1, 32)	128
conv1d_59 (Conv1D)	(None, 1, 16)	3600
dropout (Dropout)	(None, 1, 16)	0
lstm_37 (LSTM)	(None, 1, 300)	380400
dropout_1 (Dropout)	(None, 1, 300)	0
lstm_38 (LSTM)	(None, 128)	219648
batch_normalization_78 (Batch Normalization)	(None, 128)	512
dense_54 (Dense)	(None, 32)	4128
dense_55 (Dense)	(None, 16)	528
dense_56 (Dense)	(None, 1)	17
Total params: 617,265		
Trainable params: 616,809		
Non-trainable params: 456		

Figure 3.7: CNN-LSTM model summary

3.4.3 Model Optimization

Optimization is critical to ensuring that the two base models do not underperform. It is even more important to achieve a satisfactory level of performance. The two optimization techniques used in this thesis are described below.

3.4.3.1 Bayesian Optimization

Machine learning algorithms are rarely parameter-free. Even more concerning are neural network models, which require a unique set of parameters and hyper-parameters to optimise the underlying model's capacity. The question then becomes how to get a model to perform at an expert level despite the fact that it has an infinite number of possible hyper-parameter combinations.

Alternatively, the Bayesian technique was used in this thesis to find the best combination of hyperparameters for the single-architecture LSTM model. The process is explained hereinafter.

The LSTM model used in this thesis is composed of stacked layers with various model parameters. Parameters are the model's coefficients that are chosen by the model, whereas hyperparameters are set manually. The layers are viewed as an objective function with a set of parameters in this technique. The objective function is compared to a "black box function" because its inner structure is unknown, but only the parameters it will use are. This function generates metrics like loss, MSE, and MAE to provide the foundation for performance evaluation.

Thus, the Bayesian process aids in determining the best set of parameters to use when evaluating the objective function with the Gaussian process (GP) or surrogate model. The importance of GP is that it aids in comprehending the structure of the objective function. The Bayesian algorithm optimises the objective function because the GP has already learned the structure of the objective function by selecting the appropriate set of parameters from the parameter space. The previously learned parameter space will be utilized to suggest the next parameter. The process will continue to search the set of parameters until a stopping condition for convergence is met (Yugesh Verma, 2021).

The largest iteration that ran in this thesis was 100. At the outset, the range of values for hyperparameters such as batch size, dropout rate, learning rate, and so on were defined, as were the optimizers, decay rate, and momentum. The epoch is the only manually adjustable hyper-parameter with a value change range of 10 to 300. A data pipeline was built (which will be discussed in the following chapter). The suggested combinations of other hyper-parameters (batch size, drop out, learning rate), optimizer, layers, and metrics from the outcome of the Bayesian process were automatically logged after each iteration for the specified epoch size to aid performance comparison with the base LSTM model.

3.4.3.2 Optimization with Self-Attention layer

To enhance the performance of each of the underlying models, a self-attention layer was added. The following elements were used to optimize the self-attention LSTM model:

- 1) Input Layer: input the time-series data of the model.
- 2) Batch normalization layer: scales each batch size of the input data.
- 3) LSTM layers: two layers stacked with 300 and 128 units as shown in figure 3.8.
- 4) Attention layer: this layer was added after the second LSTM layer (see figure 3.8) to generate a weight vector (Asmaa Fahim et al., 2021). It weights the hidden state of all timestep(s), and focuses attention on the more important ones in the entire hidden state information sequence. The activation function used in this layer is the sigmoid function.
- 5) Flatten Layer: to reshape the multidimensional input into an output of N·1 matrix. It still retains the underlying elements contained in the original tensor.
- 6) Dense layers: five layers of units 64, 32, 32, 16 and 1 that performs matrix-vector multiplication.
- 7) Dropout layer: set at 0.2 after the first dense layer to reduce overfitting.

Figure 3.8 shows the summary of the LSTM optimized with self-attention layer.

Layer (type)	Output Shape	Param #
batch_normalization_19 (Batch Normalization)	(None, 7, 4)	16
lstm_36 (LSTM)	(None, 7, 300)	366000
lstm_37 (LSTM)	(None, 7, 128)	219648
seq_self_attention_17 (Self-Attention)	(None, 7, 128)	8257
flatten_13 (Flatten)	(None, 896)	0
dense_59 (Dense)	(None, 64)	57408
dropout_19 (Dropout)	(None, 64)	0
dense_60 (Dense)	(None, 32)	2080
dense_61 (Dense)	(None, 32)	1056
dense_62 (Dense)	(None, 16)	528
dense_63 (Dense)	(None, 1)	17
=====		
Total params: 655,010		
Trainable params: 655,002		
Non-trainable params: 8		

Figure 3.8: Self-Attention LSTM model summary

Note: The hybrid CNN-LSTM model underwent the same optimization process. In other words, the fully - connected convolution neural network was just above the optimized LSTM with self-attention.

3.4.4 Evaluation Metrics

There have been references to the boundaries of the performance measurements discussed in section 2.7. The R2-score was for comparison between the baseline models and their optimized variants because on an average scale, it gives the general performance of the model. The MAE was used to compare the error levels of the models.

3.4.5 Hyperparameter Tuning

To obtain a model with good performance, model tuning becomes increasingly crucial. Particularly with deep learning, tuning hyperparameters to obtain the optimum set can be very challenging. Although a manual technique was used for the majority of the testing, an automated optimization technique detailed in section 3.4.3.1 was also used. Extensive runs were performed to achieve optimal hyperparameter combinations with the help of the data pipeline established utilizing the MLFlow cloud resource. In the chapter that follows, which deals with the setup of the experiment, this process will be explained in greater detail.

The learning rate is regarded as one of the most crucial hyperparameters (Y. Wu et al., 2019) because it might cause the algorithm to become stuck in a local minimum if it is too high or too low, delaying convergence (Y. Wu et al., 2019). Hyperparameter tuning aids in both model improvement and finding the hyperparameter set that best generalizes the model and prevents either overfitting or underfitting. To achieve a balance between overfitting and underfitting is the goal of each supervised learning project. A model check point that monitors and saves the model with the lowest validation MAE and best weight is employed in this implementation. Then, a prediction is made using the saved model. Figure 3.9 depicts the function of the model checkpoint and how it attempts to achieve a good fit.

Metric	Latest	Min	Max
r_score	0.764 (step=0)	0.764 (step=0)	0.764 (step=0)
mae	5069.1 (step=99)	4707.2 (step=86)	19667.2 (step=0)

Figure 3.9: Data showing how the model checkpoint works

Figure 3.9 depicts a data set enabled by the MLFlow cloud resource, which was used to log all parameters, hyper-parameters, and model artefacts at each run. It demonstrates that the model monitored and used the minimum MAE(4707.2) at 86 epochs to make predictions and compute the R2-score for a hundred epochs.

3.5 Technical implementation

This solution was implemented using Python 3.9, TensorFlow, Keras, MLFlow etc. and the IDE is Jupyter. The other requirement is attached to Appendix A as an image.

3.6 Summary

This chapter begins by describing the domain knowledge, the dataset, its source, and the documentation that explains the variables that were used to make an assumption about what the likely demand, or target variable, is. A quick exploratory data analysis revealed the seasonality in the understudy data. The feature selection technique consisted of three stages: Pearson correlation coefficient, statistical analysis using the variance inflation factor (VIF), and the Xgboost feature importance package. This VIF shows how previous customer purchasing patterns and price influence future purchases.

The structure of the implemented models was then described in detail. It was reported that two base line models (LSTM and CNN-LSTM) were implemented and optimized with the self-attention layer. Furthermore, the Bayesian optimization for hyperparameter combination was explored for the LSTM model only.

The importance of evaluating model performance was emphasized, and the limitations of well-known regression metrics were cited as the reason for using. Model performance can be harmed by both overfitting and underfitting. To address this issue, the model checkpoint was used. Finally, the technical implementation and the resources uses were outlined.

Chapter 4

4.1 Experiments, Results and Discussion

The methodology and technical solution to the defined problem were detailed in the previous chapter. The dataset, its processing, feature engineering, and evaluation metrics were all covered. The hyper-parameters and accuracy scores are the most important aspects of this experimental analysis. They serve as the foundation for the analysis and comparison. The following section describes these experiments and the reasoning behind them.

4.2 Experiment Set-Up

TensorFlow 2.9.1 and Keras 2.9.0 were used in Python 3.9.7 with the Jupyter notebook environment on a Windows intel core-i7 machine powered by an Nvidia RTX studio GPU for these experiments. The most important library for managing and logging all of the resulting experimental parameters is the Machine Learning Flow (MLFlow).

The steps for integrating MLFlow (version 1.2.8.0.) with Python are outlined below.

- I. The mlflow library was imported from Keras and the experiment was initiated with the “mlflow.set_experiment (name of the experiment)” command.
- II. The tracking uri was then set by running “mlflow” and then “mlflow ui” from the anaconda command prompt. Figure 4.1 depicts this more clearly. These experiments’ tracking uri is <http://127.0.0.1:5000>.
- III. The experiment was started with the "mlflow.start run" command, which takes the experiment number as a parameter.
- IV. Following that, "mlflow.log metric" was set. This command automatically logs all hyper-parameter combinations specified in the model, as well as the training and validation loss and model artefacts. However, the desired metrics for this thesis, MAE and R2-score on the test dataset, were created and added to the logged parameters for each experiment.

Appendix figure A.1 and A.2 contains the Python code for the experiment setup.

```

(tf-gpu) C:\Users\Dell>mlflow
Usage: mlflow [OPTIONS] COMMAND [ARGS]...

Options:
  --version  Show the version and exit.
  --help     Show this message and exit.

Commands:
  artifacts  Upload, list, and download artifacts from an MLflow...
  azureml    Serve models on Azure ML.
  db          Commands for managing an MLflow tracking database.
  deployments  Deploy MLflow models to custom targets.
  experiments  Manage experiments.
  gc          Permanently delete runs in the `deleted` lifecycle stage.
  models      Deploy MLflow models locally.
  pipelines   Run MLflow Pipelines and inspect pipeline results.
  run         Run an MLflow project from the given URI.
  runs        Manage runs.
  sagemaker   Serve models on SageMaker.
  server      Run the MLflow tracking server.
  ui          Launch the MLflow tracking UI for local viewing of run...

(tf-gpu) C:\Users\Dell>mlflow ui
INFO:waitress:Serving on http://127.0.0.1:5000

```

Figure 4.1: The generated mlflow uri through the anaconda prompt

4.2.1 Experiment idea

The experimental setup and its significance in data generation have been discussed. However, the analysis in this thesis is based on experimental ideas and techniques, the results of which will provide insight into which of the two proposed baseline models performs better. The central argument is outlined below, followed by the scenarios that supplement the analysis.

4.2.2 The Central Arguments:

For example, Rui Chen et al. (2019) proposed a CNN-LSTM hybrid model to learn the spatio-temporal correlation of atmospheric and oceanic variables for forecasting typhoon formation and intensity. They claimed that their proposed model outperformed the other existing models in terms of performance.

To further deconstruct this logic, the LSTM predicts using temporal relationships, whereas the CNN predicts using spatial data relationships. Furthermore, the CNN-LSTM employs spatio-temporal relationships, which can make it more effective than the previous two, and this is the basis of the argument in this dissertation.

Thus, on a strawberry dataset, this thesis evaluates the performance of two major models with distinct abilities (LSTM, which leverages temporal correlation, and CNN-LSTM, which leverages spatio-temporal relationships). According to domain knowledge, the default belief is that CNN-LSTM will outperform LSTM, and the task of the experiments is to prove otherwise.

A self-layer was added to the LSTM model and the LSTM portion of the CNN-LSTM to further investigate their efficiency after optimization. This fusion is based on the research of Asmaa Fahim et al. (2021), who explain that with self-attention, each data input can be assigned more detailed and different weights to avoid missing essential and critical information. They cited Bahdanu et al. (2014) and claimed that attention is primarily proposed to overcome RNN shortcomings and optimize the LSTM network. This thesis employs self-attention to optimize the LSTM model, and it is assumed that the LSTM with self-attention mechanism will outperform the ordinary LSTM, as will the attention-based CNN-LSTM outperform the ordinary CNN-LSTM. Additional goal of these experiments will be to prove otherwise.

The final argument is based on the sensitivity and significance of hyperparameters in Deep Neural Networks (DNN) models such as the ones used in this thesis. In a study that compared manual tuning by human experts to automated hyperparameter optimization for seven hyperparameters of a CNN model, J. Snoek et al. (2012) found that the automated technique outperformed human experts. In contrast, Hyunghun et al. (2020) stated that deep learning researchers are hesitant to use this automated technique because most existing automated procedures are incapable of using previously learned knowledge. Another reason given by the researchers is the possibility of a disastrous failure.

This study investigates both ends of the above arguments: the automatic technique using Bayesian optimization and the manual handling. As described in section 4.2, the hyper-parameters for the processes were logged automatically. This provides a wealth of data for analysis and discovery of the fair method based on the least amount of error of the test dataset. It is worth noting that this process was only performed on the LSTM model, and the goal is not to justify the better approach, but rather to report the observations based on the dataset used and the model implemented. As a result, the outcome may differ in other study.

This thesis pursues three fundamental experimental goals. Hypotheses will be tested in the three experiments, and decisions will be made based on majorly, the R2-score, MAE on the test data, and other relevant logged metrics. The details of the experiment are shown in Table 4.1.

ID	Experiment Type	Null hypothesis (H_0) Based on R2-Score	Alternate hypothesis (H_1) Based on R2-Score	
1	LSTM vs CNN-LSTM	CNN-LSTM > LSTM	LSTM > CNN-LSTM	
2	Self-attention(LSTM) vs LSTM	Attention(LSTM)>LSTM	LSTM > Attention(LSTM)	
3	Attention(CNN-LSTM) vs CNN-LSTM	Attention(CNN-LSTM) > CNN-LSTM	CNN-LSTM > Attention(CNN-LSTM)	

Table 4.1: Experiment type and hypothesis formulation

Data was gathered as described in section 4.2, and for the experimental task, five data samples with varying hyperparameter combinations and metrics were accessed from the cloud storage and used. The created data for the scenarios is embed in table 4.2 below.






Scenarios	Data File	Sample size
LSTM strawberry modelling	 LSTM_Hyperparams.csv	56 runs
CNN-LSTM strawberry modelling	 CNN_LSTM_Hyperparams.csv	72 runs
Self-Attention LSTM strawberry modelling	 Self-Attention LSTM Hyperparams.csv	17 runs
Self-Attention CNN-LSTM	 CNN-LSTM with Self-Attention.csv	58 runs
Bayesian Hyperparameter Optimization	 Bayesian-Hyperparams.csv	100 runs

Table 4.2: Table showing various case scenarios and the analysed data

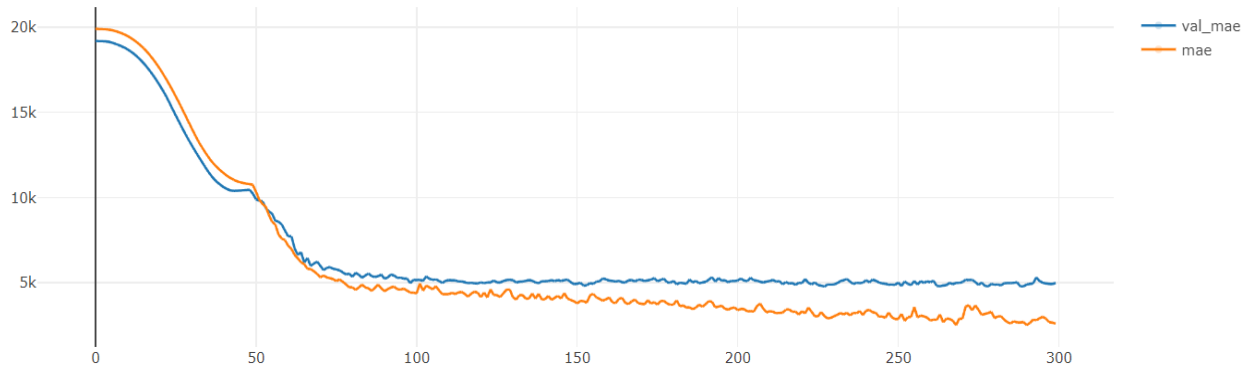
Section 4.2 contains the host link to the data contained in the embed CSVs. The following section presents the results of the experiments conducted in response to the scenarios created.

4.3 Experiment Results

The previous section discussed the scenarios that supplemented the experimental arguments and design. This section presents the results of the experimental type's analysis.

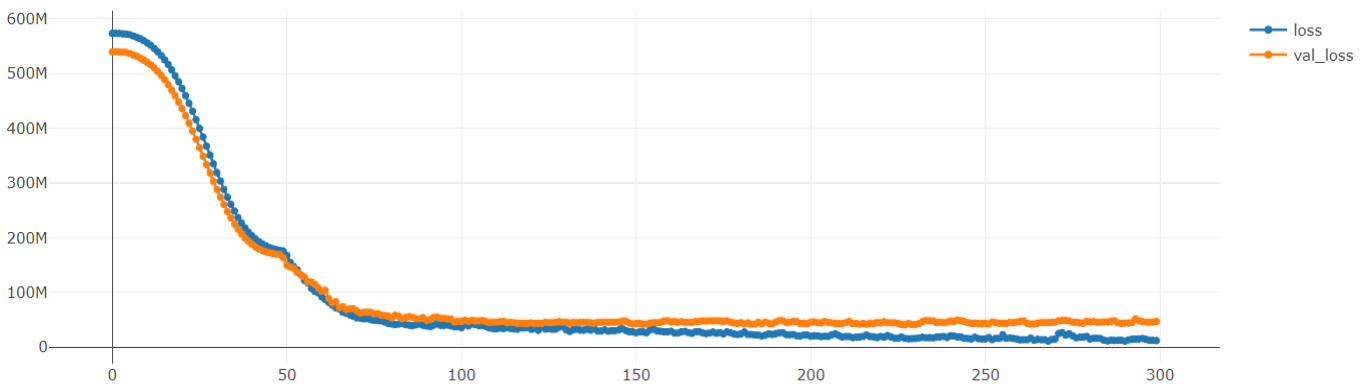
Examining the manual versus automatic tuning arguments

It begins by observing the automatic hyperparameter tuning (via a Bayesian technique) and the manual tuning process for the LSTM model only. The observations' outcome is graphically represented with appropriate details. This thesis does not choose which of the two ways is superior; instead, it concentrates on reporting the observations.



Metric	Latest	Min	Max
val_mae	4981.4 (step=299)	4744.5 (step=288)	19190.4 (step=0)
mae	2582.6 (step=299)	2461.8 (step=268)	19904 (step=0)

Figure 4.2: Graphical illustration of the Bayesian detected least Validation MAE(val_mae) and the Training data MAE(mae) over a fixed epoch size of 300.



Metric	Latest	Min	Max
loss	11700713 (step=299)	10754439 (step=268)	573517824 (step=0)
val_loss	47220832 (step=299)	40349244 (step=229)	539737216 (step=0)

Figure 4.3: Graphical representation of the Bayesian detected Training-loss (loss) and Validation loss (val_loss) based on the least Validation Mae over a fixed epoch size of 300.

The graphs of the training MAE and loss versus their respective validations are displayed in Figures 4.2 and 4.3. The graphed losses are based on the least MAE on the validation data because it is generally agreed that a model generalizes well with low errors on validation set.

In this optimization, the goal is to reduce the objective function loss on the train data and also minimize the error metric, MAE, on the test data. The graphs above represent the result of the hyperparameters that were optimized based on recommendations from the Bayesian process (BP) or automatic tuning. The other hyperparameters determined by the BP are presented in table 4.3 below. The training epoch was manually set at 300.

Determined Hyperparameters by the BP	Value
Batch size	30
Learning rate	0.001
Optimizer	Adam
Num of Layers	8

Table 4.3: combination of hyperparameters from a BP(least MAE)

Figures 4.2 and 4.3 indicate that the training loss and validation loss both gradually reduce and exhibit stability between epoch sizes of 50 and just above 100. The graph indicates that the model appears to begin overfitting at an epoch not far from 300. The training MAE and validation MAE both exhibit the same behavior.

Finally, the BP automatically identifies and suggests model layers with a high probability of achieving both the minimum validation loss and minimum validation MAE, as shown in the data in figures 4.2 and 4.3, in addition to the suggested hyperparameters, which could result in improved model performance. The summary and stacking of the layers of the BP-engineered model are shown in Figure 4.4.

Layer (type)	Output Shape	Param #
batch_normalization_11 (Batch Normalization)	(None, 7, 4)	16
lstm_14 (LSTM)	(None, 7, 300)	366000
dropout_5 (Dropout)	(None, 7, 300)	0
lstm_15 (LSTM)	(None, 7, 128)	219648
attention_9 (attention)	(None, 128)	135
dense_20 (Dense)	(None, 32)	4128
dense_21 (Dense)	(None, 16)	528
dense_22 (Dense)	(None, 1)	17
Total params: 590,472		
Trainable params: 590,464		
Non-trainable params: 8		

Figure 4.4: Proposed LSTM layers by the Bayesian Process of automatic tuning.

The train and validation loss of the largest validation MAE, together with the model summary as engineered by the BP is shown in figures 4.5, 4.6 and 4.7 below.

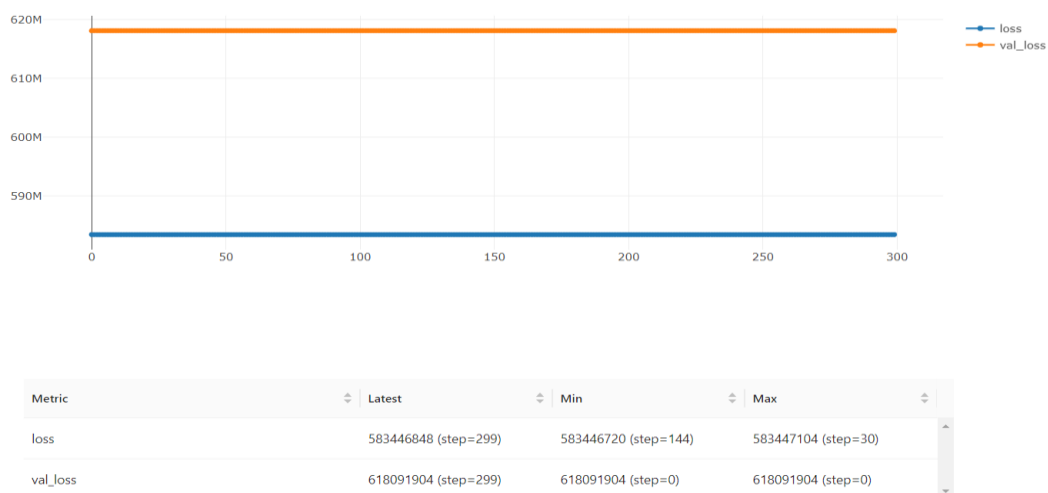


Figure 4.5: Graphical representation of the Bayesian detected Training-loss (loss) and Validation loss (val_loss) based on the least Validation Mae over a fixed epoch size of 300.

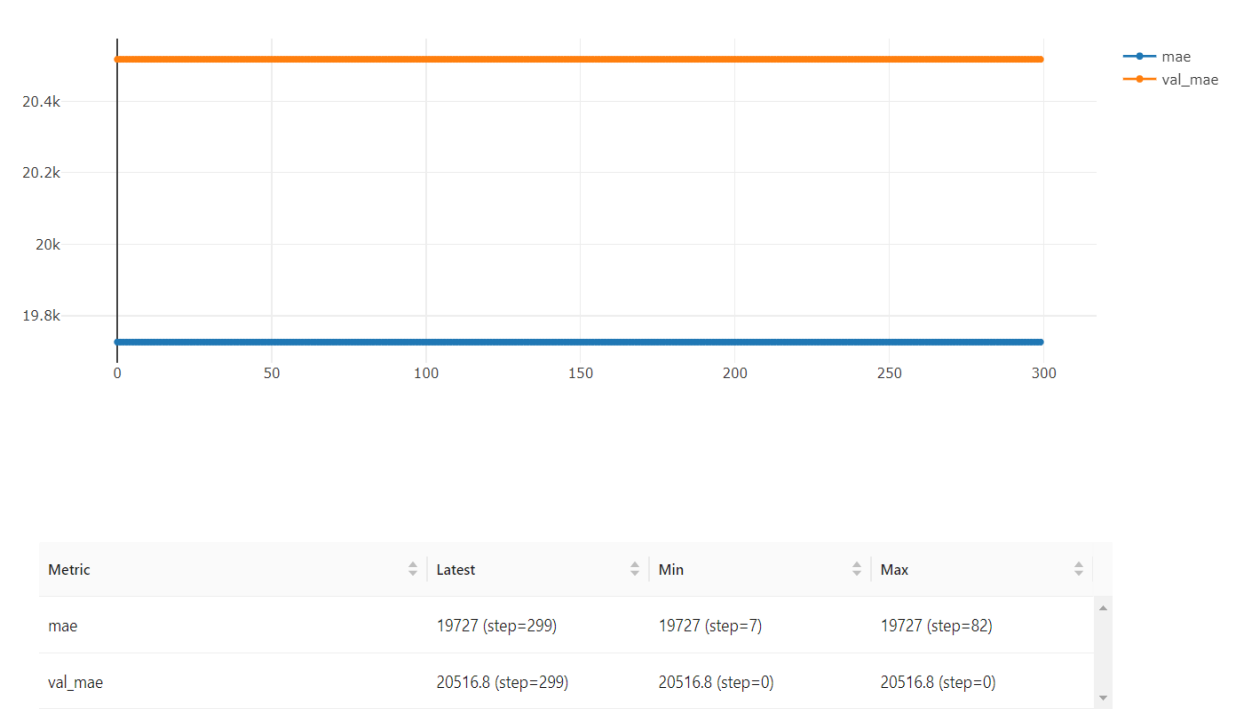


Figure 4.6: Graphical illustration of the Bayesian detected largest Validation MAE(val_mae) and the Training data MAE(mae) over a fixed epoch size of 300.

Model: "sequential_1"

Layer (type)	Output Shape	Param #
dense_3 (Dense)	(None, 7, 512)	2560
layer_dense_1 (Dense)	(None, 7, 13)	6669
dense_4 (Dense)	(None, 7, 1)	14

Total params: 9,243
Trainable params: 9,243
Non-trainable params: 0

Figure 4.7: BP-detected model summary with layers based on the largest validation MAE

Determined Hyperparameters by the BP	Value
Batch size	64
Learning rate	0.001
Optimizer	Adam
Num of Layers	3

Table 4.4: combination of hyperparameters from a BP (largest MAE)

Figures 4.5 and 4.6 show two parallel lines that don't meet at any point. This shows the inability of the model to learn and demonstrates how the error level affects how well the model performs.

Repeating the aforementioned steps for the LSTM model created through manual hyper-parameter tuning is the next step in the observation process. A manually tweaked LSTM model's training and validation loss graph at 200 epochs is shown in Figure 4.8.

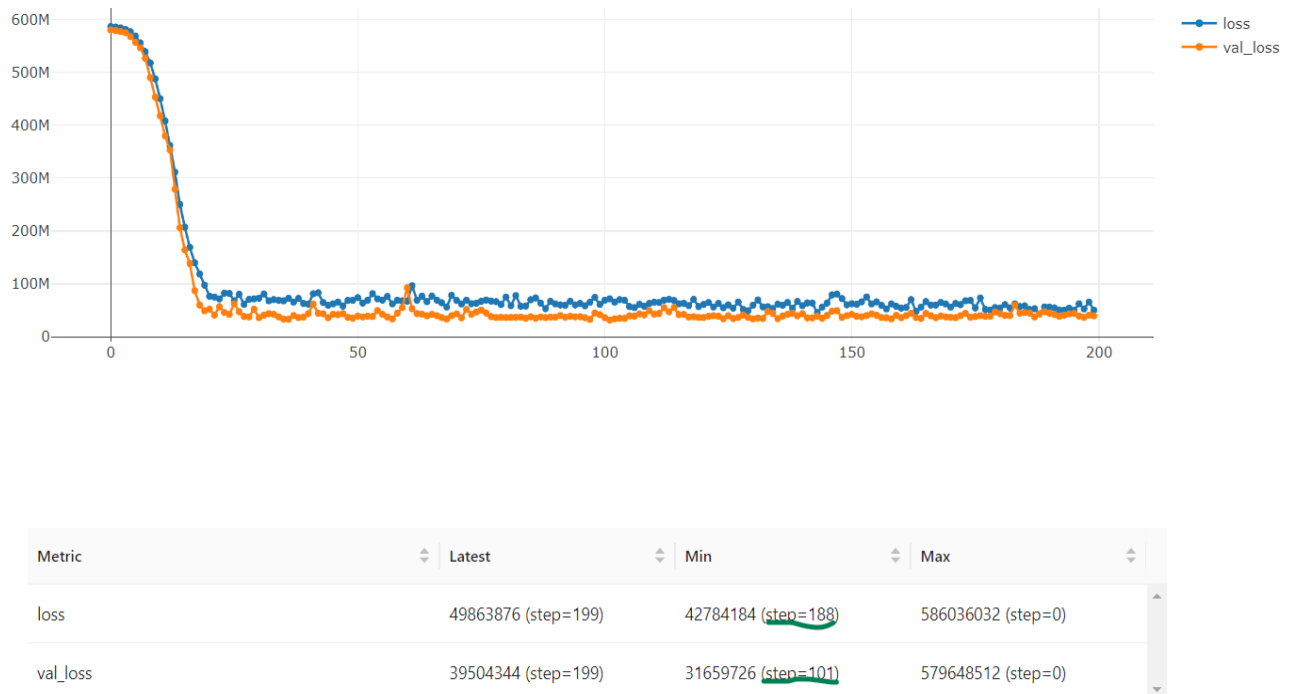
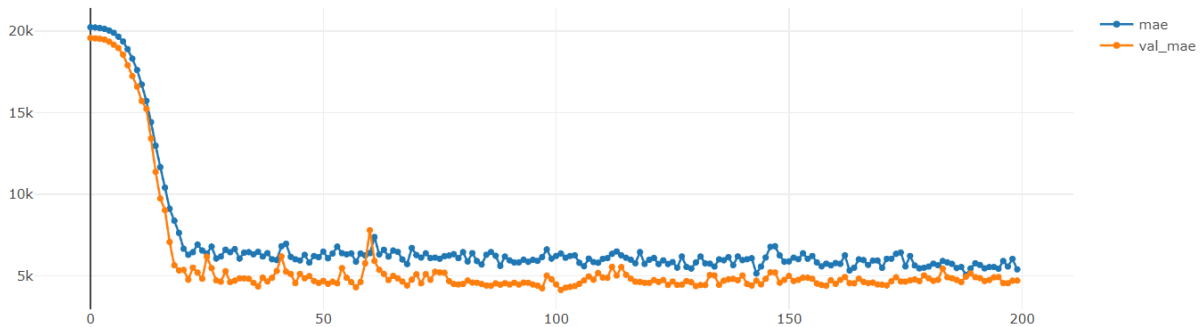


Figure 4.8: The train and validation loss of manually tuned LSTM at 200 epochs

Adam is the optimizer, and the other manually adjusted hyperparameters are batch size (16), learning rate (0.001), and number of layers (9). According to the data with green highlights, the validation loss converged after around 100 rounds. The training and validation MAE are graphically represented in Figure 4.9 under the identical set of circumstances as Figure 4.8.

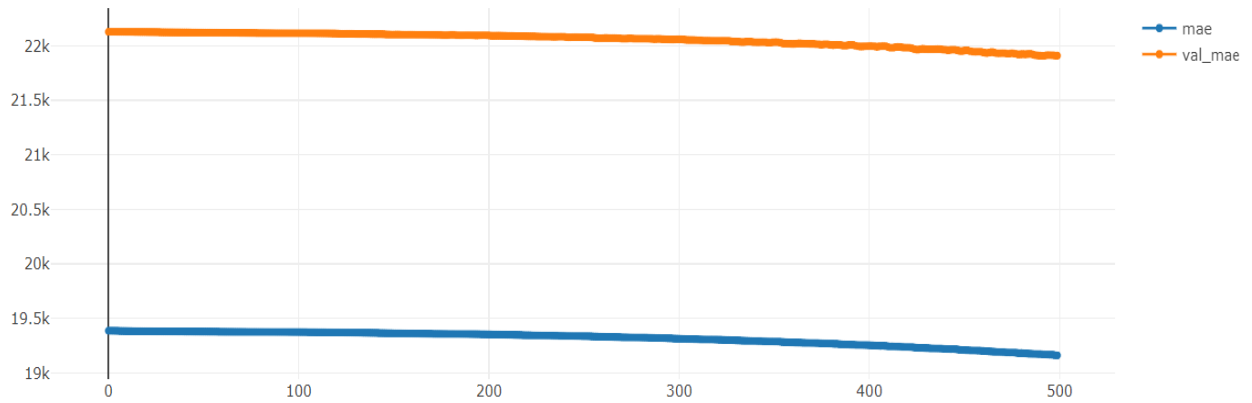


Metric	Latest	Min	Max
val_mae	4699.7 (step=199)	4080.8 (step=101)	19579.6 (step=0)
mae	5306.8 (step=199)	4972 (step=188)	20236.5 (step=0)

Figure 4.9: Graph of the training and validation MAE for manually tuned LSTM hyperparameters

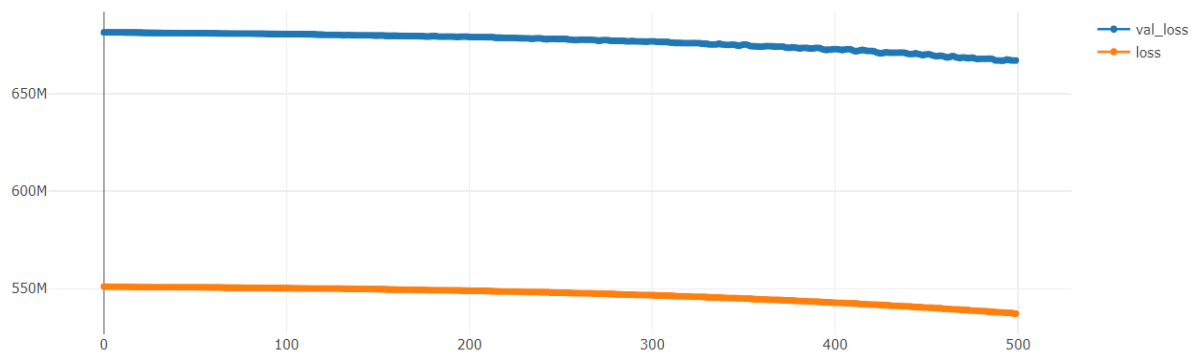
Figure 4.9 shows that in a 200-training epoch, the minimal validation MAE is attained after the 101st iteration. In this chapter's discussion part, a thorough comparison will be made with respect to previously shown graphs. It should be noted that figures 4.8 and 4.9 are plotted based on the least validation MAE.

The training and validation graphs of the MAE and loss, respectively, for the manually adjusted LSTM hyper-parameters based on the largest validation MAE are shown in Figures 4.9(a) and 4.9(b).



Metric	Latest	Min	Max
mae	19159 (step=499)	19159 (step=499)	19385.7 (step=0)
val_mae	21910.5 (step=499)	21906.6 (step=488)	22128.3 (step=0)

Figure 4.9(a): Training and validation MAE of the manually tuned hyperparameters based on the largest validation MAE



Metric	Latest	Min	Max
val_loss	667101632 (step=499)	666826432 (step=491)	681437888 (step=0)
loss	537143360 (step=499)	537143360 (step=499)	551072896 (step=0)

Figure 4.9(b): Training and validation loss of the manually tuned hyperparameters based on the largest validation MAE

Figures 4.5, 4.6, 4.9(a), and 4.9(b) all follow a similar graphical pattern and consistently show the nature of models that are rife with errors and fail to converge. The hyper-parameters for the graphical output in figures 4.91 and 4.92 are batch size (300), epochs (500), learning rate (0.0001), number of layers (7), and Adam is the optimizer.

Figure 4.9(c) displays the layers and summary of the manually tuned LSTM model with the least validation MAE. Figure 4.9(d) displays the same but with the largest validation MAE.

Model: "sequential"		
Layer (type)	Output Shape	Param #
batch_normalization (Batch Normalization)	(None, 7, 4)	16
lstm (LSTM)	(None, 7, 300)	366000
dropout (Dropout)	(None, 7, 300)	0
lstm_1 (LSTM)	(None, 128)	219648
batch_normalization_1 (Batch Normalization)	(None, 128)	512
dense (Dense)	(None, 32)	4128
dropout_1 (Dropout)	(None, 32)	0
dense_1 (Dense)	(None, 16)	528
dense_2 (Dense)	(None, 1)	17
Total params: 590,849		
Trainable params: 590,585		
Non-trainable params: 264		

Figure 4.9(c): Model summary showing the layers of the manually tuned LSTM with the least MAE

Model: "sequential"		
Layer (type)	Output Shape	Param #
batch_normalization (Batch Normalization)	(None, 7, 4)	16
lstm (LSTM)	(None, 7, 300)	366000
lstm_1 (LSTM)	(None, 128)	219648
batch_normalization_1 (Batch Normalization)	(None, 128)	512
dense (Dense)	(None, 32)	4128
dense_1 (Dense)	(None, 16)	528
dense_2 (Dense)	(None, 1)	17
Total params: 590,849		
Trainable params: 590,585		
Non-trainable params: 264		

Figure 4.9(d): Model summary showing the layers of the manually tuned LSTM with the largest MAE

Figures 4.9(c) and 4.9(d) clearly share the same model architecture, layers, etc., but because of the varied hyper-parameter combinations chosen, they perform very differently. This underpins the sensitivity and significance of hyperparameters in DL.

As anticipated in this dissertation, the last argument outlined in section 4.2.2 has been looked into, and both sides of the debate between manual and automatic tuning have been examined. The stated results will be compared, and also the outcomes of the other arguments will be looked at next in the discussion portion of this chapter.

Experiment ID: 1 (LSTM vs CNN-LSTM)

This experiment aims to compare the capabilities of CNN-LSTM with LSTM. The three main logic used for this performance comparison are;

- The train MAE and the validation MAE are used during the training process. On the test data, the MAE was used for evaluation. This was done to understand the generalization properties of each model and to estimate the loss function of the model.
- R2-score-based interpretation of the input variables by the model. This is done in order to determine the model that fit the regression line better. This is an interpolation task which uses in-sample data and gauges the predictive ability of the model.
- Forecasting strength using the R2-score and the MAE as indicators. This forecasting exercise aids in evaluating each model's capacity for extrapolating predictions from out sample. In other words, this assignment reveals how effectively the model can forecast long into the future, beyond the test dataset's last date. In essence, it will assess the model's aptitude in a real world noisy data.

Note: the models considered are those which give the best accuracy after several hyper-parameter combinations. Thus, the hyper-parameters for each of the presented models are noted for comparison for this experiment and others. Figure 4.9(e) shows the train and validation MAE graph for the best hyper-parameter combination of the CNN-LSTM model.

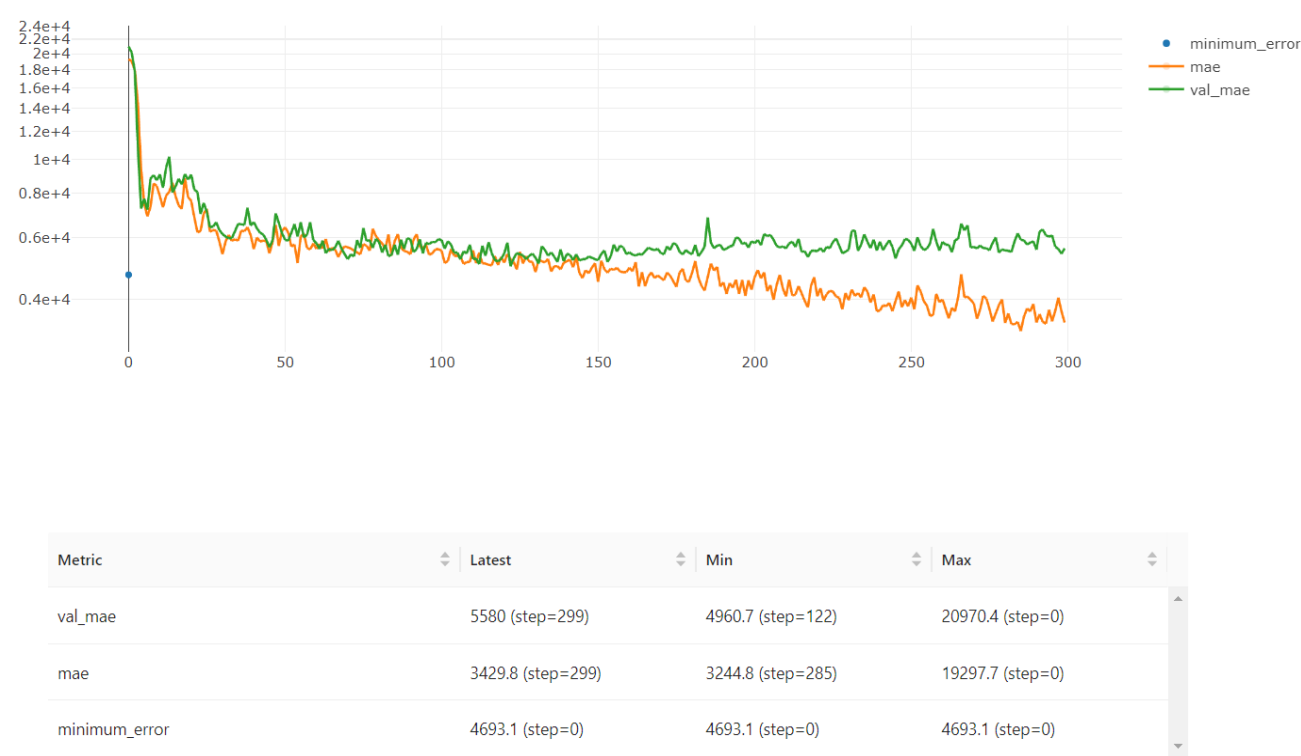


Figure 4.9(e): Log graph showing the validation MAE, training MAE, and the final MAE on the validation data for the CNN-LSTM model.

The variable minimum error in figure 4.9(e) represents the MAE for the test set of data. Except for a slight increase in errors near the end of the training phase, the graph appears to match well overall. The slight increase in error might indicate overtraining brought on by relatively long epochs. A batch size of 32, 300 epochs, a learning rate of 0.01, and Adam as the optimizer were the model's hyperparameters. In comparison to the MAE, the model's predicted accuracy on the test data was 79.7%. (see figure 4.9f).

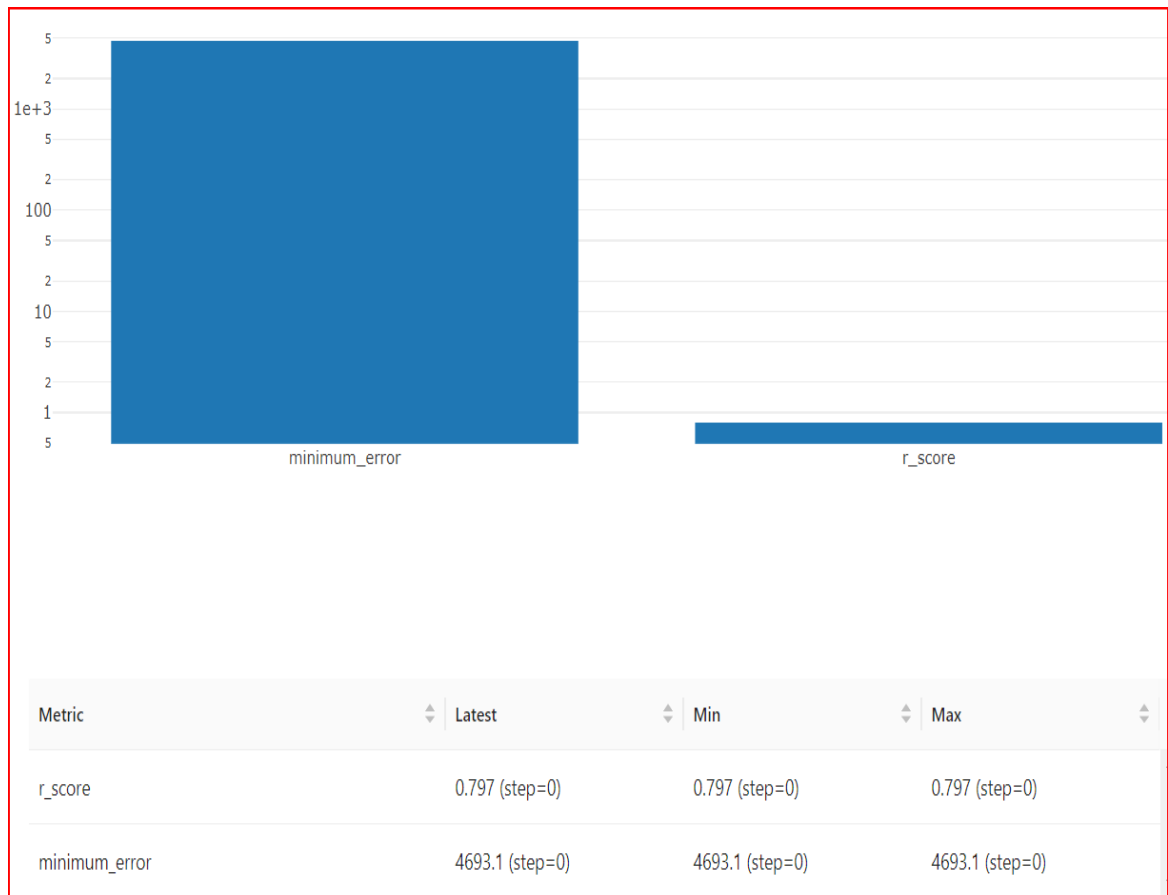


Figure 4.9(f): Log-scaled graphical representation of the relationship between the CNN-LSTM MAE and the R2-score on the test dataset

A predictive accuracy of 78.2% in regard to the MAE is shown by the LSTM with the optimal hyperparameter combination (batch size of 50, learning rate of 0.001, 100 epochs, Adam as optimizer) on the test data. The training and validation MAE over the training procedure is shown in Figure 4.9(g).



Figure 4.9(g): Log graph showing the validation MAE and training MAE on the validation data for the LSTM model.

Unlike the CNN-LSTM, the training process here converge faster although the training time is lower. The charts in fig 4.17 shows the relationship between the LSTM model’s MAE and it accuracy on the test data.



Figure 4.9(h): Log-scaled graphical representation of the relationship between LSTM MAE and R2-score on the test dataset

Figures 4.9(f) and 4.9(h) show that the CNN-LSTM has a slightly higher error level than the LSTM, with the latter having a higher accuracy of 79.7%. Perhaps a better fit (using R2) of the regression model doesn't quite imply a lower prediction error. On the other hand, the observation could be due to the insignificant difference in R2-scores between the two models. However, further investigation into other experiments may provide more clarity.

Table 4.1 shows the combination of hyperparameters for the two models and their resulting metrics

Model	Batch size	Epochs	Learning rate	Optimizer	R2-score (%)	Test MAE
CNN-LSTM	32	300	0.01	Adam	79.7	4693.1
LSTM	50	100	0.001	Adam	78.2	4628.5

Table 4.4: Table showing the hyperparameter combinations of the two models

On the test data, the training process, hyper parameter combinations, and predictive capacity of the CNN-LSTM and the LSTM have all been investigated. The forecasting capability of these models will be examined next.

This study takes into account forecasting horizons of 7 weeks, 14 weeks, and 28 weeks. The forecasting horizon refers to the number of weeks after the last date of test data. Thus, the skill of each model over a range of three horizons in order to know their reliability in forecasting future demand is used to evaluate their performance. Figures 4.9(i), 4.9(j), and 4.9(k) depict the LSTM model's forecast over these three horizons, along with their R2-scores and MAE.

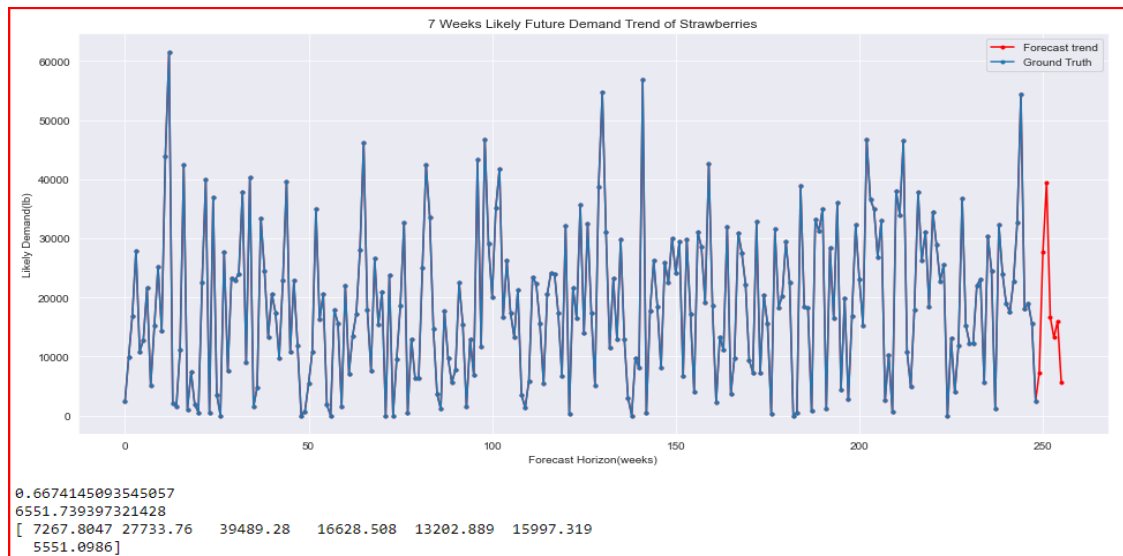


Figure 4.9(i): Graphical Illustration of the LSTM 7weeks likely demand forecast

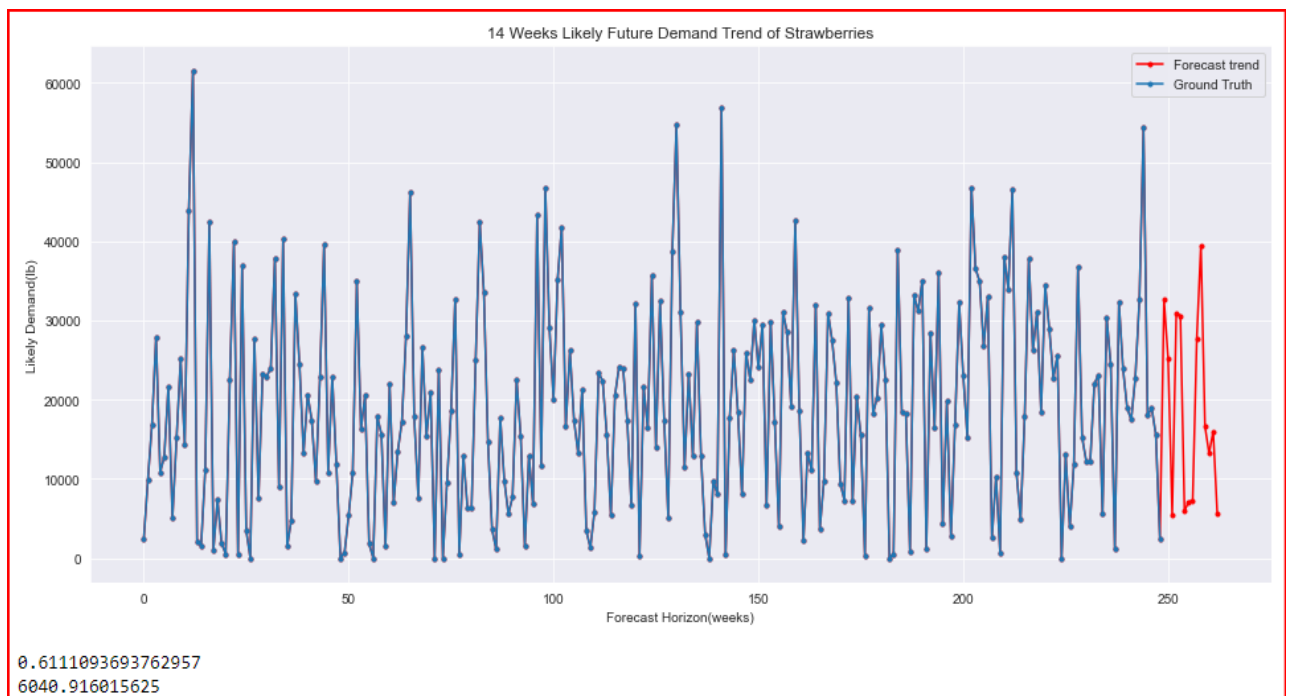


Figure 4.9(j): Graphical illustration of the LSTM 14weeks likely demand forecast

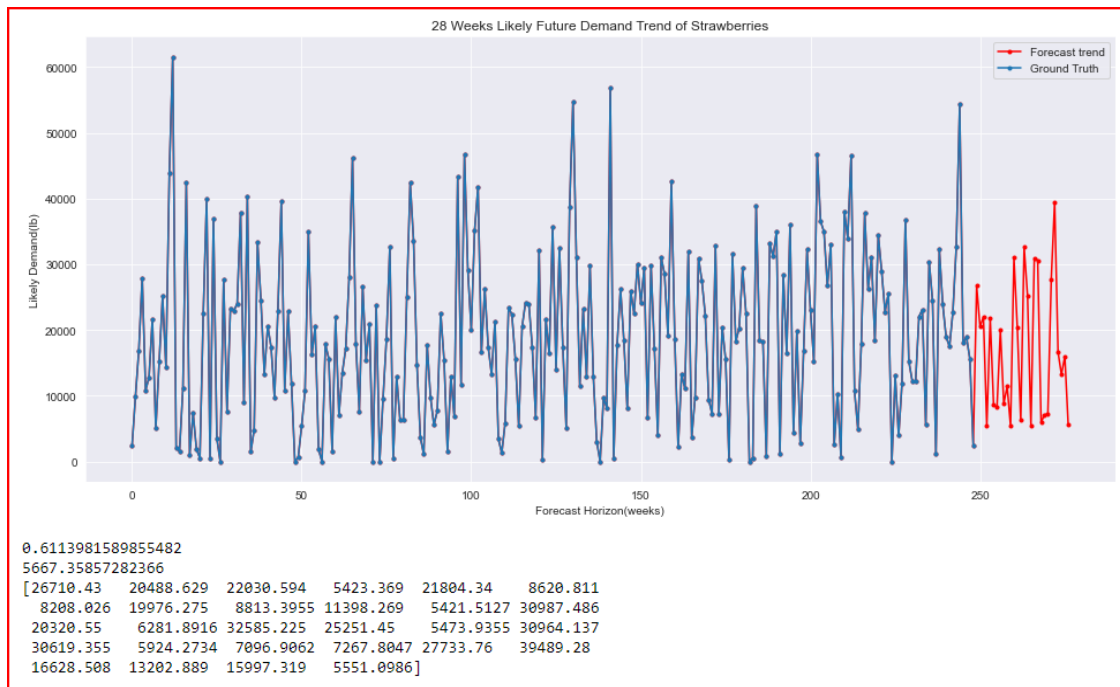


Figure 4.9(k): Graphical illustration of the LSTM 28 weeks likely demand forecast

The skill of the CNN-LSTM is then assessed over the same forecast horizon. This is depicted in Figures 4.(l), 4.9(m), and 4.9(n).

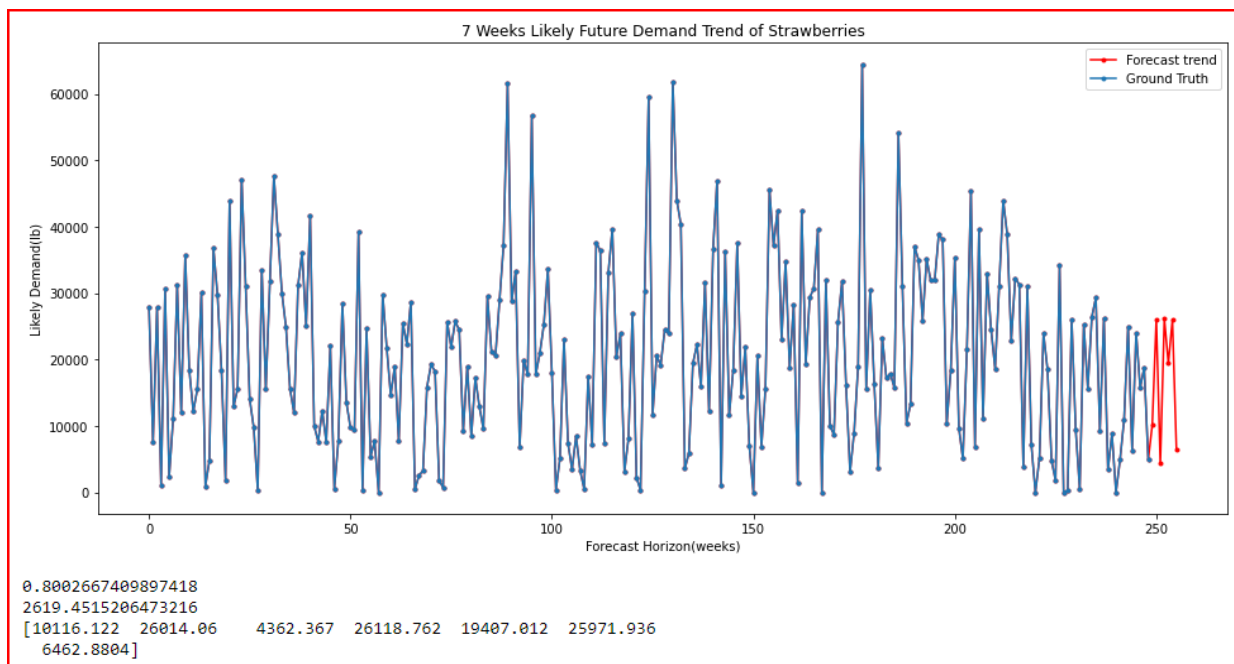


Figure 4.9(l): Graphical illustration of the CNN-LSTM 7weeks likely demand forecast.

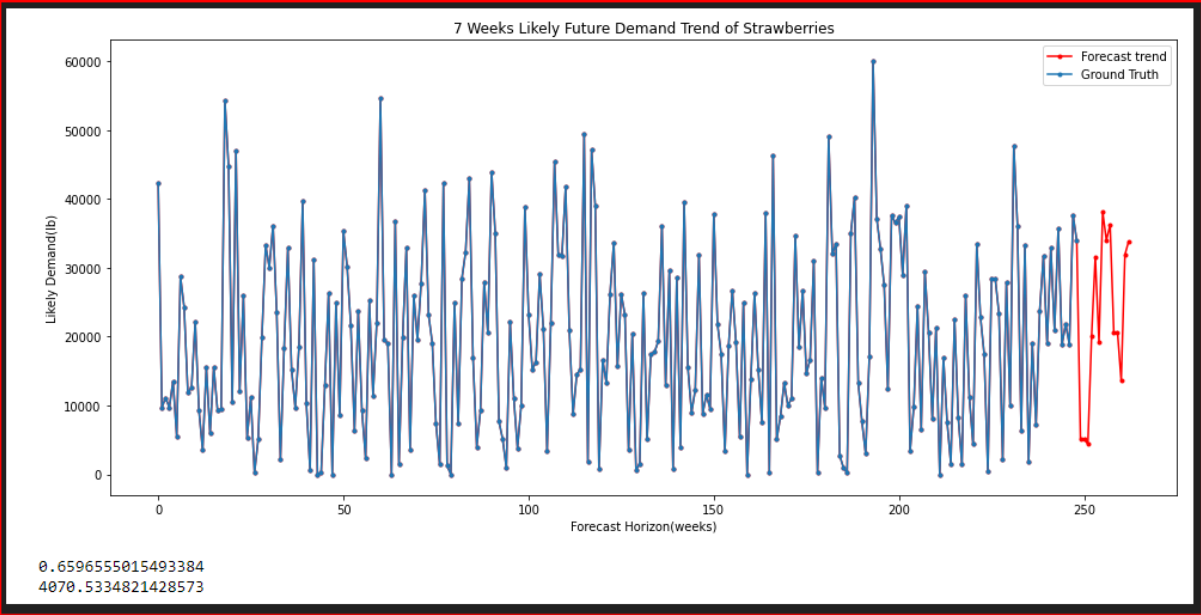


Figure 4.9(m): Graphical illustration of the CNN-LSTM 14weeks likely demand forecast

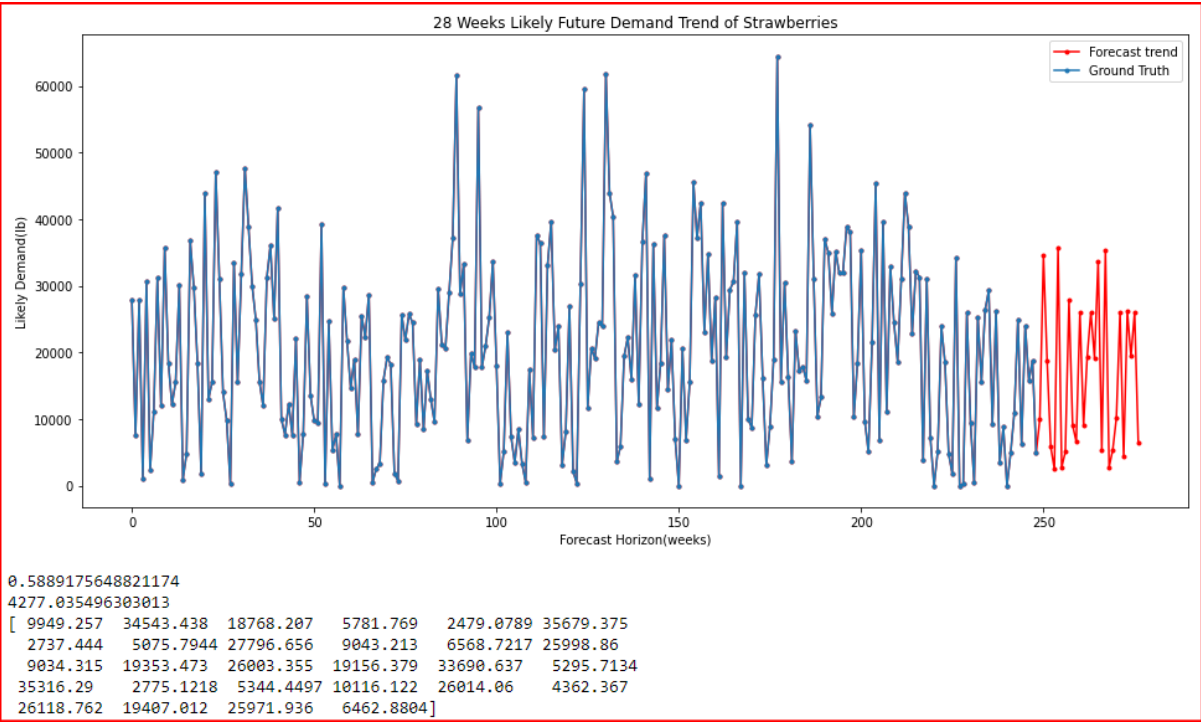


Figure 4.9(n): Graphical illustration of the CNN-LSTM 28 weeks likely demand forecast

The graphical illustration of each model’s forecast in steps of 7, 14, and 28 weeks alongside their respective R2-score and MAE has been shown for this experiment. The decision to increase the forecast horizon to 28 weeks adds a level of difficulty to the forecasting problem. Doing this would further showcase the

efficacy of the models. Table 4.5a, 4.5b and 4.5c show the tabulations of the key metrics of each model over the specified time horizons.

Model	R2-score(%)	MAE
LSTM	66.7	6551.7
CNN-LSTM	80	2619

Table 4.5a : Key metrics of LSTM and CNN-LSTM in 7 weeks forecast

Model	R2-score(%)	MAE
LSTM	61.1	6040.9
CNN-LSTM	65.9	4070

Table 4.5b : Key metrics of LSTM and CNN-LSTM in 14 weeks forecast

Model	R2-score(%)	MAE
LSTM	61.1	5667.3
CNN-LSTM	58.9	4277.0

Table 4.5c : Key metrics of LSTM and CNN-LSTM in 28 weeks forecast

Experiment ID: 2 (Self-Attention LSTM vs LSTM)

The experimentation procedure is the same as in the previous experiment. The difference is that the analysis would only be performed on the self-attention LSTM model and would not be repeated on the LSTM model. The first phase of the analysis presented was done to understand the training process, estimate the loss function and how well the self-attention model generalizes on the test data compared to the LSTM.

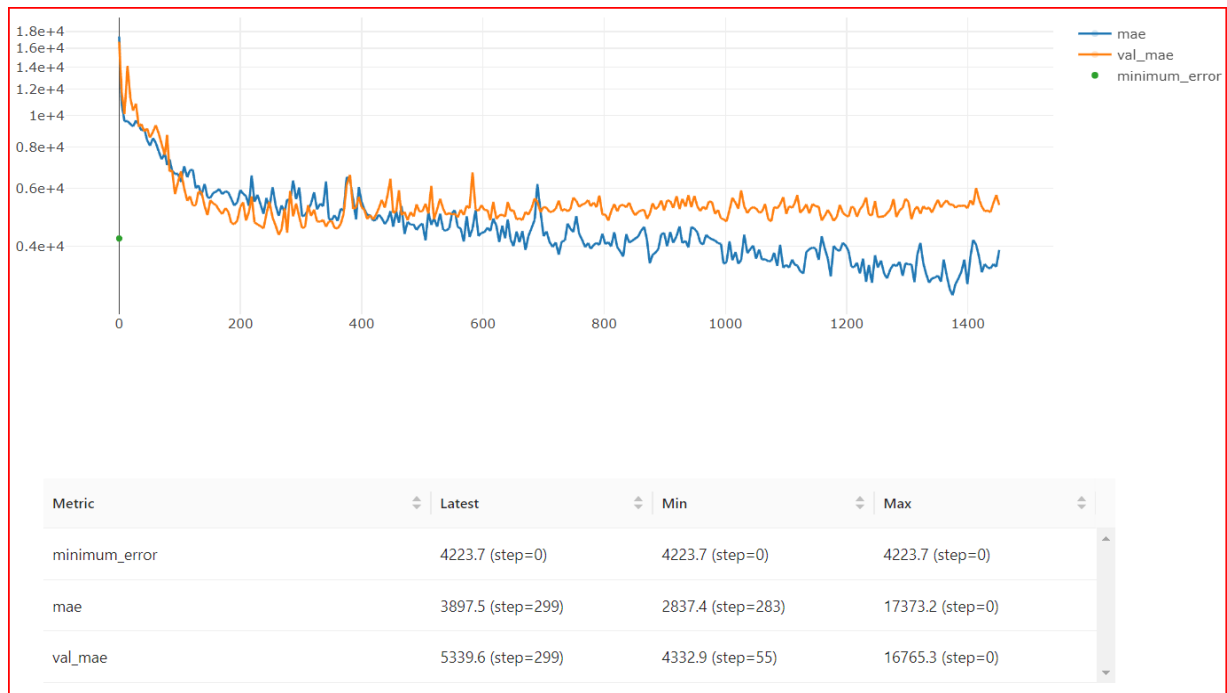


Figure 4.9(o): Graphical training trend of the Self-Attention based LSTM model over an epoch size of 300

The MAE on the test set for the under-review model is displayed above. In comparison to the LSTM model, it reveals a lower value. The requirement to optimize the LSTM led to the addition of the attention layer in this model. The optimized model was with the hyper-parameter combination of batch size (16), epochs (300), learning rate (0.01) and Adam as optimizer.

Furthermore, the predictive capacity (accuracy against error) of the self-attention LSTM was observed in figure 4.9(p). In comparison to the LSTM, table 4.6 shows the two models (LSTM and Self-attention LSTM) summarized parameters over the training process and predictive ability.

Model	Batch size	Epochs	Learning rate	Optimizer	R2-score (%)	Test MAE
Self-Attention-LSTM	16	300	0.01	Adam	80.5	4223.7
LSTM	50	100	0.001	Adam	78.2	4628.5

Table 4.6: Table showing the hyperparameter combinations and the key performance metrics of the two models



Figure 4.9(p): Log-scaled graphical representation of the relationship between Self-Attention LSTM MAE and R2-score on the test dataset.

Over the same forecasting horizons as previous models, the self-attention LSTM's forecasting power is evaluated. Figures 4.9(q), 4.9(r) and 4.9(s) illustrates these findings. The LSTM and self-attention LSTM's forecasting ability is summarized in Table 4.7.

Models	R2-score (%)	MAE	Forecasting Horizons(week)
LSTM	66.7	6551.7	7 weeks
LSTM with Attention	78.5	5804.4	7 weeks
LSTM	61.1	6040	14 weeks
LSTM with Attention	70.7	5524.8	14 weeks
LSTM	61.1	5667.3	28 weeks
LSTM with Attention	67.7	5253	28 weeks

Table 4.7: Summary of the LSTM and LSTM with Attention forecasting metrics

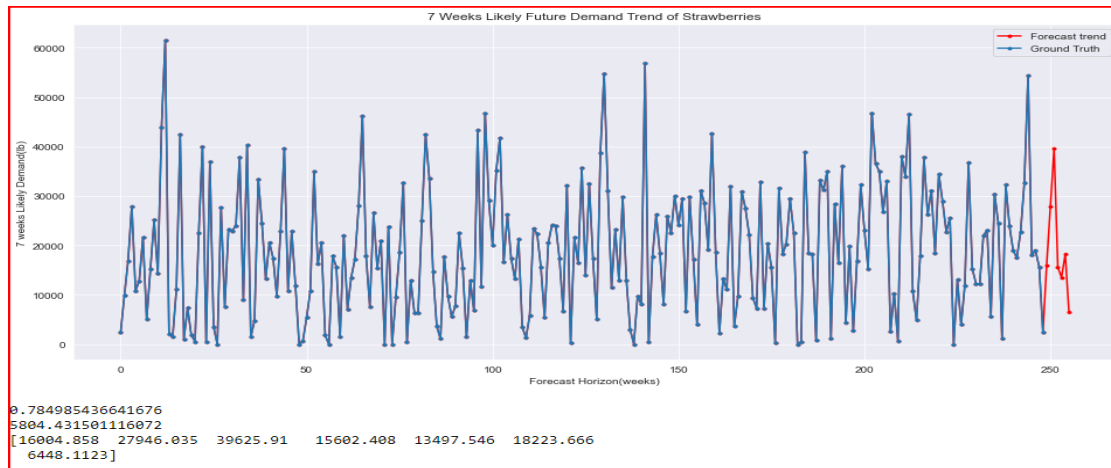


Figure 4.27 (q): 7 weeks forecast with Self-Attention LSTM

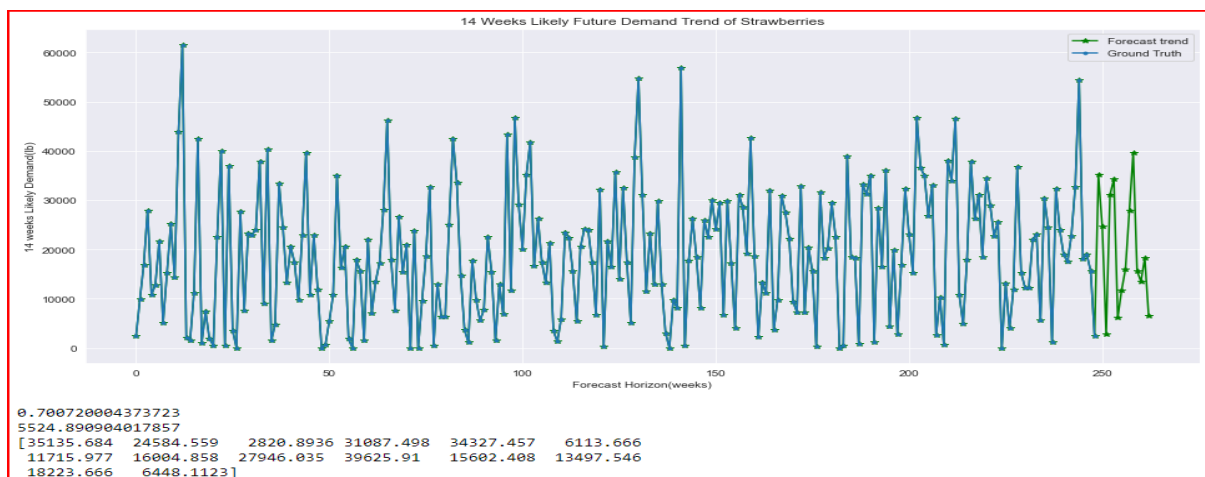


Figure 4.9 (r): 14 weeks forecast with Self-Attention LSTM

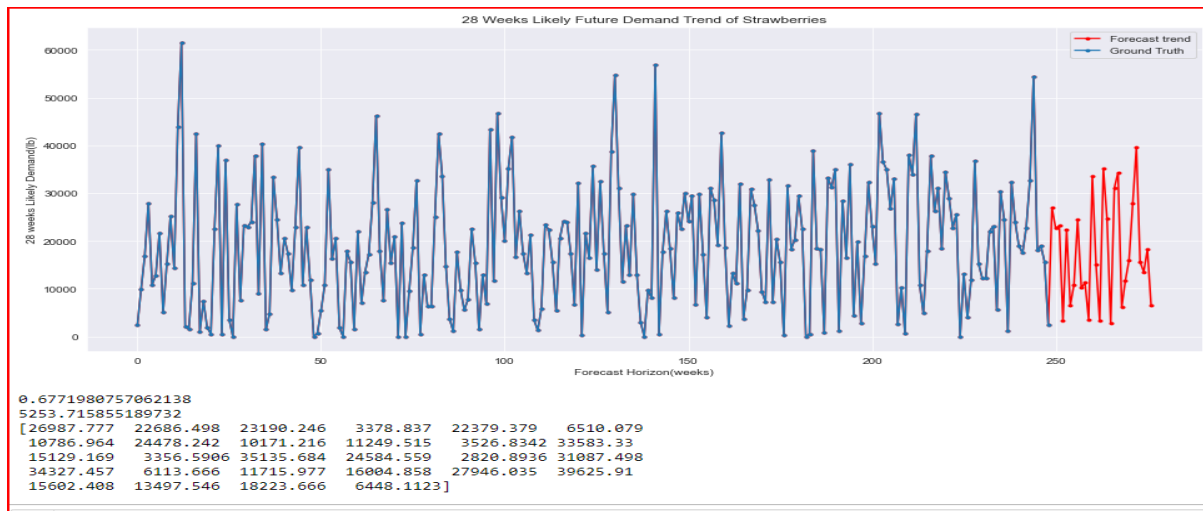


Figure 4.9(s): 28 weeks forecast with Self-Attention LSTM

Experiment ID: 3 (Self-Attention CNN-LSTM vs CNN-LSTM)

The thesis's final suggested experiment is this one. It compares the CNN-LSTM's performance to that of its enhanced version with the attention mechanism. To make the most efficient use of time, the outcomes of the three main procedures—training, prediction, and forecasting—are tabulated, and the graphs can be viewed in Appendix B (Figures B1, B2 and B3). The prediction capability of each model is shown in Table 4.8, and Table 4.9 displays the models' relative forecasting performance across a 28-week time horizon.

Model	Batch size	Epochs	Learning rate	Optimizer	R2-score (%)	Test MAE
Self-Attention-CNN-LSTM	50	400	0.009	Adam	70.8	5463.4
CNN-LSTM	32	300	0.01	Adam	79.7	4693.1

Table 4.8: Table showing the hyperparameter combinations and the key performance metrics of the two models.

Models	R2-score(%)	MAE	Forecasting Horizons(weeks)	
CNN-LSTM	80	2619	7	
CNN-LSTM with Attention	64.3	3771.0	7	
CNN-LSTM	65.9	4070	14	
CNN-LSTM with Attention	24.4	5400.6	14	
CNN-LSTM	58.9	4277	28	
CNN-LSTM with Attention	61.5	4366.8	28	

Table 4.9: Summary of the CNN-LSTM and CNN-LSTM with Attention forecasting metrics

4.3 Discussion

The previous sections of this chapter described the experimental setup and results presentation. This section concludes the chapter by interpreting and discussing the experimental findings. The sectional analysis presented in the results will guide the structural flow of the discussion.

To begin, both sides of the debate between manual and automatic hyperparameter tuning using BP were investigated, with the goal of reporting observations rather than determining the better process. The analysis of the training and validation losses based on the least MAE revealed a matching pattern in the training processes, indicating similarity in loss values (see figures 4.2 and 4.3). This could be a source of strength for BP proponents.

Furthermore, the BP (Bayesian Process) process's suggested model architectures (see figure 4.4), assumed to be the optimized architecture, revealed the addition of the attention mechanism to its LSTM model optimization process. This BP result is consistent with the findings of Bahdanu et al., who argue that the attention mechanism aids in the optimization of LSTM networks. The motivation

for optimizing the LSTM model with an attention layer in this work is based on Bahdanu et al., not on the assumed strength of the BP.

A discovery made during the BP-process in this work, on the other hand, revealed ambiguity in its process. Despite the fact that model structures and other key components are suggested, at least one or more hyperparameter ranges were set and a large number of runs were run, resulting in a list of possible runs that includes the potential low-performing recommendation. This could potentially strengthen BP's opponents' arguments.

The first experiment in this work sought to determine the better model for strawberry likely demand forecasting between LSTM and CNN-LSTM. Three distinct approaches were used, but the main yardstick is the ability of the two models to forecast using R2-score as the indicator. The other approaches were added to put the models' skill to the test.

With Adam as the optimizer, CNN-LSTM achieved 79.7% predictive accuracy on the test dataset with a batch size of 32, an epoch of 300, and a learning rate of 0.01. Figure 4.15 depicts the model's error level. The LSTM's predictive accuracy is 78.2% with a lower learning rate, which means convergence is slower, and a batch size of 50 with a lower training epoch of 100, which could translate to lower computational cost. Although the LSTM accuracy is slightly lower than that of the CNN-LSTM, this seemingly small margin can be useful when forecasting in tons. The difference in MAE indicates that the CNN-LSTM accumulated 64 units of error more than the LSTM.

The results of the forecasting skill test for both models over a forecast horizon (in weeks) of 7, 14, and 28 show that the CNN-LSTM outperforms the LSTM except for week 28, where the LSTM outperforms. The accuracy decreases as the forecasting horizon increases, which is ideal except for the LSTM, whose accuracy remained constant from week 14 to week 28. Surprisingly, as the CNN-LSTM accuracy decreases, so does its MAE, implying a logical correlation. It is the same for LSTM too. In general, this forecasting task demonstrates the increase in error as the forecasting horizon becomes longer.

The same procedure is repeated in the second experiment, but this time between the LSTM and its optimized variant with the Self-Attention layer (Self-Attention LSTM). The error level is lower when compared to the LSTM. This was accomplished using the hyperparameter combination shown in Table 4.6. The tabulated data revealed that the attention-based model had a predictive accuracy of 80.5%, which was 2.3% higher than the normal LSTM. Similarly, on the test data, the LSTM with attention mechanism has a lower error level, 404.8 unit lower, than the ordinary LSTM. Thus, across the stated metrics, attention-based LSTM shows a better prospects than the ordinary LSTM in terms of predictive skill and minimizing loss.

To further investigate, the forecasting abilities of the attention-based LSTM and the ordinary LSTM were assessed over the same forecast horizons as previously (see figures 4.27a, 4.27b, and 4.27c). The results, as shown in table 4.7, further validate the superiority of attention-based LSTM over ordinary LSTM. Again, as the forecasting horizon lengthens, both models' forecasting abilities deteriorate. However, there was an irregular pattern of error reduction as accuracy increased. Unlike the CNN-LSTM, which captures the logical trend.

The last experiment compares the Self-Attention CNN-LSTM to the CNN-LSTM. Similar procedures were used, and CNN-LSTM accuracy of 79.7% outperformed the Self-Attention CNN-LSTM accuracy of 70.8%. (see table 4.8). Furthermore, CNN-LSTM reduces errors better than the CNN-LSTM with the self-attention mechanism, with an error unit of 770 lower. Furthermore, the CNN-LSTM was optimized at 300 epochs and a 0.01 learning rate, resulting in a shorter computation time than the other model with 400 iterations and a 0.009 learning rate.

The models' propensity for forecasting was also assessed. This validates the CNN-LSTM's predictive capability to attention-based alternatives (see table 4.9). The CNN-LSTM outperforms its attention-based variations in terms of accuracy over all forecasting horizons. The models display similar trends in that their forecasting accuracy is inversely correlated with their errors. That is, the forecasting accuracy increases as the MAE decreases. Because of this, CNN-LSTM has lower error rates over all forecasting horizons, making it even preferable.

The experimental objectives in this section have been met, and the choice about the hypothesis is shown in table 4.9(a).

ID	Experiment Type	Null hypothesis (H_0) Based on R2-Score	Alternate hypothesis (H_1) Based on R2-Score	Decision
1	LSTM vs CNN-LSTM	CNN-LSTM > LSTM	LSTM > CNN-LSTM	Accept H_0
2	Self-attention(LSTM) vs LSTM	Attention(LSTM)>LSTM	LSTM > Attention(LSTM)	Accept H_0
3	Attention(CNN-LSTM) vs CNN-LSTM	Attention(CNN-LSTM) > CNN-LSTM	CNN-LSTM > Attention(CNN-LSTM)	Reject H_0

Table 4.9(a) : Table showing the hypothesis and decision made

In general, the importance of the attention mechanism became apparent when it was included in the LSTM model, but the CNN-performance LSTM's was not enhanced by the attention effect. Additionally, attention layers were added to the top three purportedly optimized LSTM model architectures produced by the Bayesian approach (these architectures are logged in Mlflow)..

The argument made by Bahdanu et al. that the attention mechanism can be employed to optimize the LSTM model serves as a possible illustration of this. The self-attention LSTM performs best for these experiments and the data used, followed closely by the CNN-LSTM model, the LSTM, and the attention-based CNN-LSTM. In alignment with this dissertation's goal, an attention mechanism optimized LSTM model performs better than the attention mechanism optimized CNN-LSTM.

4.3 Summary

The proposed models were put to the test to see how well they could predict and forecast. In order to describe the observation on the Bayesian process for LSTM model tuning, the two sides of the arguments were looked at. at the end, it was discovered that upon optimization of the two major models (LSTM and CNN-LSTM) with the attention mechanism, the LSTM model performs better than the CNN-LSTM. The chapter came to a close with a thorough analysis of the results.

Chapter 5

5.1 Conclusion

In order to estimate likely demand, this thesis built, assessed, and determined which of the LSTM and CNN-LSTM models performed better following optimization on a strawberry dataset. The self-attention-based mechanism was used to optimize the models. Due to this, it was discovered that the LSTM (after optimization) performs better than the CNN-LSTM (after optimization). Prior to optimization, CNN-LSTM performed better.

The minimization of errors throughout the training phase, the predictive abilities using the R²-score on the test dataset, and the forecasting abilities across a maximum forecasting horizon of 28 weeks were all used to evaluate each model's performance both before and after optimization. Overall, it was found that the optimized LSTM was better.

Even though the Bayesian technique was used to identify optimized hyperparameter combination for the LSTM only, decision was not made based on it but rather gave the opportunity to explore the two sides of an argument. That is, observing the pattern between manual and automatic hyperparameter tuning.

These decisions were the result of thorough and organized experimental approaches. Machine learning flow (Mlflow), which enables logging of all model parameters, artefacts, and hyper-parameters at scale during each run, was used to set up the experiment. This prompted the development of data, which is a crucial component of the ML process. Due to the pipeline established, new data were generated in addition to the original strawberry data.

The CNN-LSTM outperforms the LSTM before optimization (using attention mechanisms), indicating that it is more accurate at prediction than the LSTM, which relies on the temporal relationship between data. After optimization, the LSTM with attention outperformed CNN-LSTM.

The outcome after optimization provides avenue to further explore, in future research, the predictive ability of a temporal model, when the focus (attention in the context of this research) is on the most important input out of very many others.

The spatio-temporal relationship based CNN-LSTM has the capacity to observe and extract the most crucial information from the input for LSTM prediction. This can imply that when a model that leverages temporal relationship is focuses on the most crucial feature (as is the case when CNN and attention were added to the LSTM) it performs better.

The results of the attention-based CNN-LSTM are intriguing. Even when compared to the standard LSTM model, the performance was poor. Although it is acknowledged that this thesis did not examine all possible optimization strategies, it would be beneficial to investigate the impact of feeding the LSTM model with ‘overly focused or too important input’ in light of the findings of this dissertation.

The underlying concept of this proposition is that with CNN-LSTM, key characteristics are extracted and input into the LSTM. As seen in this work, the performance was comparable to that of the attention-based LSTM. After the significant feature has been retrieved, the LSTM model is supplied with it, along with an attention mechanism that will further weight the input that is even more significant. The observed performance of the attention-based CNN-LSTM was poor in this aspect, as demonstrated in this work. Could this excessively narrow emphasis on a crucial LSTM model input be a benefit or a drawback? Future research projects can examine these observations.

5.2 Limitations:

The following are the limitations of this study:

- 1.) The direct quantity of strawberries purchased was unable to be accessed due to the decline in requests made to connect to the USDA API. Thus, the number of advertising stores, which also reflects consumer purchasing patterns for at least a unit of strawberry, was used as the likely demand.
- 2.) Only a few model optimization techniques were considered, the effect of which could perhaps be the reason of some underperforming models.
- 3.) Demand forecasting is a means to an end, not the end, in the intricate food supply chain network. As a result, this dissertation did not consider how forecasting the demand and the modelling of all other constraints, such as the procurement of materials and other climatic conditions, affect or influence business decisions, profit, or loss in a supply chain network.
- 4.) The error metrics used in this study have their own limitations. The model selection criteria were based on the R2-score only. Perhaps, the criteria for selection could have been better done using approaches like the Bayesian information criterion, Akaike information criterion, etc.
- 5.) The models' forecasting and predicting abilities may have been evaluated on a different dataset in order to evaluate their effectiveness in applications other than the strawberry dataset.
- 6.) Inability to gain access to the USDA API makes the forecasting not real-time.
- 7.) The DL network's stochastic nature causes the performance score to vary slightly during the course of separate runs.
- 8.) A what-if analysis tool could have been used to examine to a greater extent, the impact of each input variables.

References

1. Fotios Petropoulos, Spyros Makridakis, Vassilios Assimakopoulos, Konstantinos Nikolopoulos, 'Horses for Courses' in demand forecasting, *European Journal of Operational Research*, Volume 237, Issue 1, 2014, Pages 152-163, ISSN 0377-2217, <https://doi.org/10.1016/j.ejor.2014.02.036>.
2. FAO (2011). *Global food losses and food waste - Extent, causes and prevention*. United Nations Food & Agriculture Organization, Rome.
3. Ventour L. (2008). *The food we waste*. WRAP, Banbury, UK
4. Buzby J.C., Wells H.F., Axtman B. & Mickey J. (2009). Supermarket loss estimates for fresh fruit, vegetables, meat, poultry, and seafood and their use in the ERS loss-adjusted food availability data. *Economic Information Bulletin* 44, March.
5. Sahara Sahara, Nicholas Minot, Randy Stringer & Wendy J. Umberger (2015). Determinants and effects of small chilli farmers' participation in supermarket channels in Indonesia. *Bulletin of Indonesian Economic Studies* 51(3): 445-460
6. Juning Su, Jiebing Wu and Chenguang Liu, 2013. In: "Research on Coordination of Fresh Produce Supply Chain in Big Market Sales Environment". Hindawi Publishing Corporation. *The Scientific World Journal*. Volume 2014, Article ID 873980. Available at: <http://dx.doi.org/10.1155/2014/873980>.
7. Willem A. Rijpkema, Robert Rossi and Jack G.A.J, 2013. In: "Effective sourcing strategies for perishable product supply chains". *International Journal of Physical Distribution & Logistics Management*. Vol.44 No. 6, 2014. Pp. 494-510.
8. Gustavsson, J., Cederberg, C., Sonesson, U., van Otterdijk, R. and Meybeck, A. (2011), *Global Food Losses and Food Waste*, Food and Agriculture Organisation of the United Nations (FAO), Rome.
9. Karanam, M.; Krishnanand, L.; Manupati, V.K.; Antosz, K.; Machado, J. Identification of the Critical Enablers for Perishable Food Supply Chain Using Deterministic Assessment Models. *Appl. Sci.* 2022, 12, 4503. <https://doi.org/10.3390/app12094503>.
10. Hsiao, Y.-H.; Chen, M.-C.; Lu, K.-Y.; Chin, C.-L. Last-mile distribution planning for fruit-and-vegetable cold chains. *Int. J. Logist. Management*. 2018, 29, 862-886.
11. Leng, L.; Zhang, J.; Zhang, C.; Zhao, Y.; Wang, W.; Li, G. Decomposition-based hyperheuristic approaches for the bi-objective cold chain considering environmental effects. *Comput. Oper. Res.* 2020, 123, 105043.
12. Awad, M.; Ndiaye, M.; Osman, A. "Vehicle routing in cold food supply chain logistics: A literature review". *Int. J. Logist. Manag.* 2020, 32, 592-617.

13. Stellingwerf, H.M.; Kanellopoulos, A.; van der Vorst, J.G.; Bloemhof, J.M. "Reducing CO₂ emissions in temperature-controlled road transportation using the LDVRP model". *Transp. Res. Part D Transp. Environ.* 2018, 58, 80-93.
14. Samir Gokan, Thyagaraj S. Kuthambalayan. "Creating sustainable fresh produce supply chains by managing uncertainties". In: *Journal of Cleaner Production* 207 (2019) 908-919.
15. Ali, J., Kapoor, S., Moorthy, J., 2010. "Buying behaviour of consumers for food products in an emerging economy". *Br. Food J.* 112 (2), 109e124.
16. L. Jaxsens, P.A. Luning et al., 2009. "Simulation modelling and risk assessment as tools to identify the impact of climate change on microbiological food safety— The case study of fresh produce supply chain". In: *Food Research International* 43 (2010) 1925-1935.
17. Miraglia, M., Marvin, H. J. P., Kleter, G. A., Battilani, P., Brera, C., Coni, E., et al. (2009). "Climate change and food safety: An emerging issue with special focus on Europe". In: *Food and Chemical Toxicology*, 47, 1009-1021.
18. Soysal, M.; Bloemhof-Ruwaard, J.M.; Haijema, R.; van der Vorst, J.G. "Modelling a green inventory routing problem for perishable products with horizontal collaboration". In: *Comput. Oper. Res.* 2018, 89, 168-182.
19. Bortolini, M.; Faccio, M.; Ferrari, E.; Gamberi, M.; Pilati, F. "Fresh food sustainable distribution: Cost, delivery time and carbon footprint three-objective optimization". In: *J. Food Eng.* 2016, 174, 56-67.
20. Solina, V.; Mirabelli, G. Integrated production-distribution scheduling with energy considerations for efficient food supply chains. *Procedia Comput. Sci.* 2021, 180, 797-806.
21. E. Simangunsong, L.C. Hendry & M. Stevenson (2012) Supply-chain uncertainty: a review and theoretical foundation for future research, *International Journal of Production Research*, 50:16, 4493-4523, DOI: [10.1080/00207543.2011.613864](https://doi.org/10.1080/00207543.2011.613864).
22. J. Cuaresma, J. Hlouskova, S. Kossmeier, and M. Obersteiner "Forecasting electricity spotprices using linear univariate time-series models", *Appl. Energy*, vol. 77, pp. 87-106, 2004.
23. J. Contreras, R. Espinola, F. J. Nogales and A. J. Conejo, "ARIMA models to predict next-day electricity prices," *IEEE Transactions on Power Systems*, vol. 18, no. 3, pp. 1014-1020, Aug. 2003.
24. A.T. Jebb and L. Tay, "Introduction to time series analysis for organizational research: Methods for longitudinal analyses". *Organizational Research Methods*, vol. 20, no. 1, pp. 61-94, 2016.

25. E. S. Gardner, and Ed. McKenzie, "Forecasting Trends in Time Series" Management Science Vol. 31, No. 10, pp. 1237-1246, Oct., 1985.
26. A. M. Davey and B. E. Flores, "Identification of seasonality in time Series: A note". Mathl. Comput. Modelling vol. 18, no. 6, pp. 73-81, 1993.
27. D. Kwiatkowski, P. C. B. Phillips, P. Schmidt and Y. Shin, "Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root?" Journal of Econometrics, vol. 54, no. 1-3, pp. 159-178, 1992.
28. ES. Gardner, "Exponential smoothing: the state of the art", Journal of Forecasting, vol. 4, no. 1, pp. 1-28, 1985.
29. ES. Gardner, "Exponential smoothing: The state of the art-Part II". International Journal of Forecasting, vol. 22, no. 4, pp. 637-666, 2006.
30. N. H. Chan, "Time series co-integration". International Encyclopaedia of the Social & Behavioural Sciences, pp. 15709-15714, 2001.
31. J. H. Stock, "Time series: Economic forecasting". International Encyclopaedia of the Social & Behavioural Sciences, pp. 15721-15724, 2001.
32. Teja, Kalakuntla. (2019). Re: Difference between "Forecasting" and "Predicting" a hydrological variable? Retrieved from: www.researchgate.net/post/Difference-between-Forecasting-and-Predicting-a-hydrological-variable.
33. H. Demuth, M. Beale, and M. Hagan, "Neural network toolbox user's guide". The MathWorks Inc, MA, USA, 2009.
34. I. Goodfellow, Y. Bengio and A. C. Courville. "Deep Learning", MIT Press, 2016.
35. Y. Bengio, A. Courville and P. Vincent, "Representation Learning: A Review and New Perspectives," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 8, pp. 1798-1828, Aug. 2013.
36. Nielson, M.A., 2015. Neural Networks and Deep Learning. Determination Press.
37. Faruk, D.O., 2010. A hybrid neural network and Arima model for water quality timeseries prediction. Eng. Appl. Artif. Intell. 23 (4), 586-594.
38. Y.-Y. Chen, Y.-H. Lin, C.-C. Kung, M.-H. Chung, and I.-H. Yen, "Design and implementation of cloud analytics-assisted smart power meters considering advanced artificial intelligence as edge analytics in demand-side management for smart Homes," Sensors, vol. 19, no. 9, p. 2047, May 2019.
39. Diego Manfre. Title: Do Artificial Neural Network Really Learn? Available at: Towards Data Science. [Accessed: 28-Augut,2022].
40. P. Murugan, "Learning the sequential temporal information with recurrent neural networks", arXiv:1807.02857v1 [cs.LG] Jul 2018.

41. Z. C. Lipton, J. Berkowitz and C. Elkan, "A critical review of recurrent neural networks for sequence learning". 2015. [Online]. Available: arXiv:1506.00019.
42. C. Olah, "Understanding LSTM networks". Available online at: <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>. Viewed: 28-08-2022.
43. S. Hochreiter and J. Schmidhuber, "Long short-term memory", Neural Computation, vol. 9, pp. 1735-1780, 1997.
44. F. A. Gers, N. N. Schraudolph, J. Schmidhuber, "Learning precise timing with LSTM recurrent networks", Journal of Machine Learning Research, vol. 3, pp. 115-143, 2002.
45. C. Olah, "Understanding LSTM networks". Available: <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>. Viewed: 28-08-2022.
46. A. Vaswani et al., "Attention is all you need," In 31st Conference on Adv Neural Inf Process Syst, 2017, pp. 6000-6010.
47. Anusha Lihala., 2019. "Attention and its Different Forms". Available [here](#). Viewed: 28-08-2022.
48. J. Wang and Y. Hu, "An Improved Enhancement Algorithm Based on CNN Applicable for Weak Contrast Images," in IEEE Access, vol. 8, pp. 8459-8476, 2020.
49. R. Collobert, J. Weston, "A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning", Proceedings of the 25th International Conference on Machine Learning. ICML '08. New York, NY, USA: ACM. pp. 160-167, 2008.
50. A. Vandenoord, S. Dielema and B. Schrauwe, "Deep content-based music recommendation", Proceedings on the 26th international conference on Neural Information Processing Systems, vol. 2, pp. 2643-2651, 2013.
51. A. Tsantekidis, N. Passalis, A. Tefas, J. Kannianen, M. Gabbouj and A. Iosifidis, "Forecasting Stock Prices from the Limit Order Book Using Convolutional Neural Networks," 2017 IEEE 19th Conference on Business Informatics (CBI), Thessaloniki, 2017, pp. 7-12.
52. "Translation invariance". Viewed online on 29-Aug-2022. Link [here](#)
53. Vincent Tatan. "Understanding CNN (Convolutional Neural Network). Viewed online on 29-Aug-2022. Link [here](#)
54. N. K, Ahmed, A. F, Atiya, N. E. Gayar, H. El-Shishiny, "An Empirical Comparison of Machine Learning Models for Time Series Forecasting". Econometric Reviews, vol. 29, no. 5-6, pp. 594- 621, 2010.

55. S. Jharkharia and M. Shukla, "Agri-fresh produce supply chain management: A state-of-the-art literature review," *International Journal of Operations & Production Management*, vol. 33, no. 2, pp.114-158, 2013.
56. R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *Int. J. Forecast.*, vol. 22, pp. 679-688, 2006.
57. D. L. Alexander, A. Tropsha, and D. A. Winkler, "Beware of $R(2)$: Simple, unambiguous assessment of the prediction accuracy of QSAR and QSPR models," *J. Chem. Inf. Model.*, vol. 55, no. 7, pp. 1316-1322.
58. C. J. Willmott and K. Matsuura, "Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance". *Climate Research*, vol 30, pp. 79- 82, 2005.
59. R. G. D. Steel and J. H. Torrie, "Principles and procedures of statistics with special reference to the biological sciences", McGraw Hill, 1960.
60. N. R. Draper, and H. Smith, "Applied Regression Analysis", Wiley-Interscience, 1998.
61. T. O. Kvalseth, "Cautionary note about R^2 ", *The American Statistician*, vol. 39, no. 4, pp. 279- 285, 1985.
62. M. Ray, A. Rai, V. Ramasubramanian and K. N. Singh, " ARIMA-WNN hybrid model for forecasting wheat yield time-Series data", *Journal of the Indian Society of Agricultural Statistics*, vol 70, no. 1, pp. 63-70, 2016.
63. L. Michel and D. Makowski, "Comparison of statistical models for analysing wheat yield time series", *PLOS One*, vol. 8, no. 10, pp. 1371, 2013.
64. A. Shabru, R. Samsudin and I. Zuhaimy, "Forecasting of the rice yields time series forecasting using artificial neural network and statistical model", *Journal of Applied Sciences*, vol. 9, no. 23, 2009.
65. K. Yamamoto, K. Seida, J. Nishiyama, K. Hayashi, S. Daishi and M. S. Tanaka, "Predicting System of Harvest Time and Yield of Tomato," *IEEE 7th Global Conference on Consumer Electronics (GCCE)*, Nara, pp. 437-439, 2018.
66. Ifeanyi Okwuchi, Lobna Nasar et al.,2020. "Deep Learning Ensemble Based Model for Timeseries Forecasting Across Multiple Applications",2020 *IEEE International Conference on Systems, Man, and Cybernetics (SMC)* October 11-14, 2020. Toronto, Canada.
67. RAPLANG, L., KHADE, S.D. and ROY, T.N., 2021. Factors Affecting Marketing of Vegetables among Small-Scale Farmers in West Bengal: An Analysis. *Economic Affairs*, 66(3), pp. 407-412.
68. I. Goodfellow, Y. Bengio and A. C. Courville. "Deep Learning", MIT Press, 2016.

69. Yugesh Verma. "Neural Network Hyperparameter Tuning using Bayesian Optimization. Viewed on September 1, 2022. Link [here](#)
70. Asmaa Fahim, Qingmei Tan, Mouna Mazzi, Md Sahabuddin, Bushra Naz, and Sibghat Ullah Bazai. "Hybrid LSTM Self-Attention Mechanism Model for Forecasting the Reform of Scientific Research in Morocco". Computational Intelligence and Neuroscience Volume 2021.
71. Y. Wu et al., "Demystifying Learning Rate Policies for High Accuracy Training of Deep Neural Networks", IEEE Big Data, arXiv:1908.06477 [cs.LG], 2019.
72. Chen, R., Wang, X., Zhang, W., Zhu, X., Li, A. and Yang, C. (2019) 'A hybrid CNN-LSTM model for typhoon formation forecasting', *Geoinformatica*, 23(3), 375+, available:<https://link.gale.com/apps/doc/A593707405/AONE?u=southampton&sid=bookmark-AONE&xid=6dffd49d> [accessed 03 Sep 2022].
73. D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4945-4949, IEEE, Shanghai, China, March 2016.
74. J. Snoek, Larochelle and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms", *Proc. Adv. Neural Inf. Process. Syst.*, pp. 2951-2959, 2012.
75. H. Cho, Y. Kim, E. Lee, D. Choi, Y. Lee and W. Rhee, "Basic Enhancement Strategies When Using Bayesian Optimization for Hyperparameter Tuning of Deep Neural Networks," in *IEEE Access*, vol. 8, pp. 52588-52608, 2020, doi: 10.1109/ACCESS.2020.2981072.

Appendix A

Model Artifact and Requirement

```
1 import mlflow
2 import mlflow.keras
3 mlflow.set_experiment("strawberry-modelling-experiment")
4 mlflow.set_tracking_uri("http://127.0.0.1:5000")
5 mlflow_experiment_id = 1
6
7 mlflow.keras.autolog(log_models=True)
8 with mlflow.start_run(experiment_id=mlflow_experiment_id):
```

Figure A.1: A screenshot of the MLFlow experiment set up and runs

```
##Create Metrics
r_score = r2_score(y2_test, y_prediction)
minimum_error = mean_absolute_error(y2_test, y_prediction)
## Log metrics to MLflow
mlflow.log_metric("r_score",r_score)
mlflow.log_metric("minimum_error",minimum_error)
#mlflow.log_metric("validation_split",validation_split)

mlflow.keras.log_model(model,"model")

with open("output.txt", "w") as f:

    f.write("Strawberry demand forecasting!")
    log_artifact("output.txt")
```

Figure A.2: A screenshot of the logged metrics with MLFlow

Appendix B

Graphs and chart

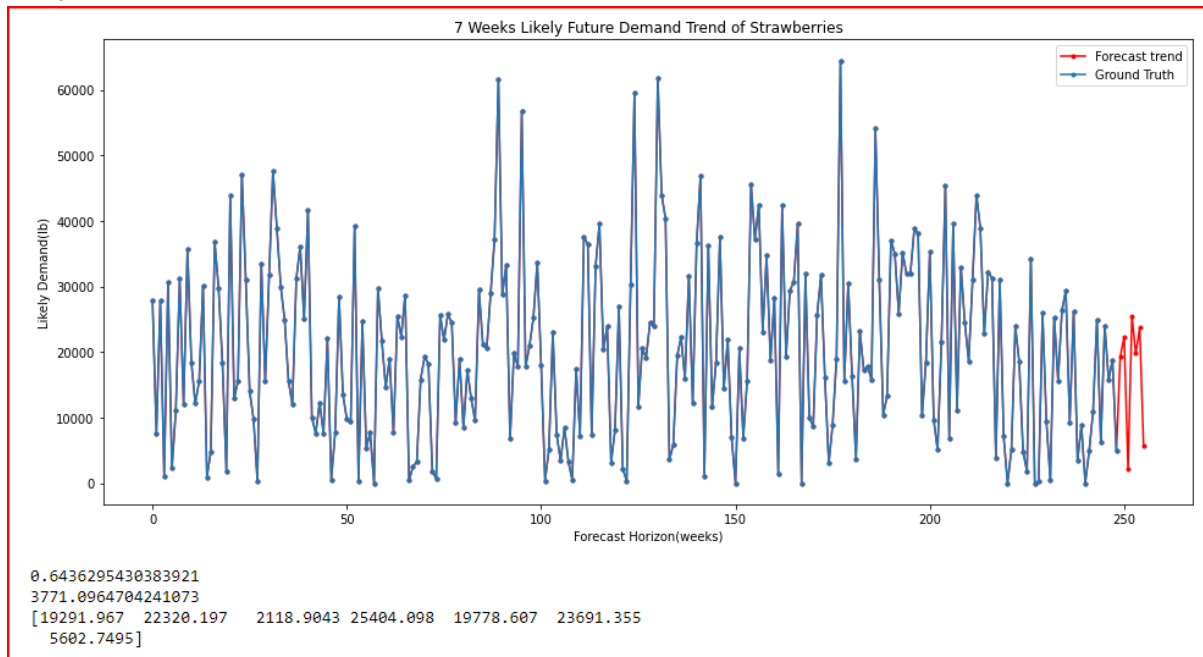


Figure B1: Attention-based CNN-LSTM 7 weeks forecast

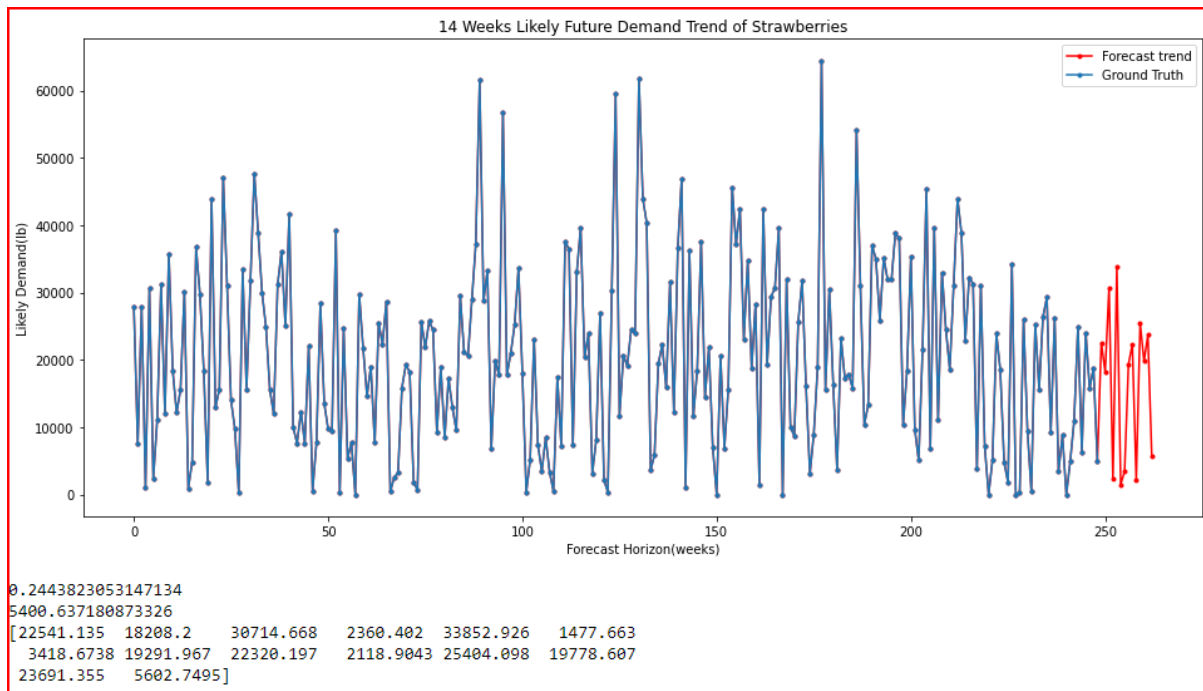


Figure B2: Attention-based CNN-LSTM 14 weeks forecast

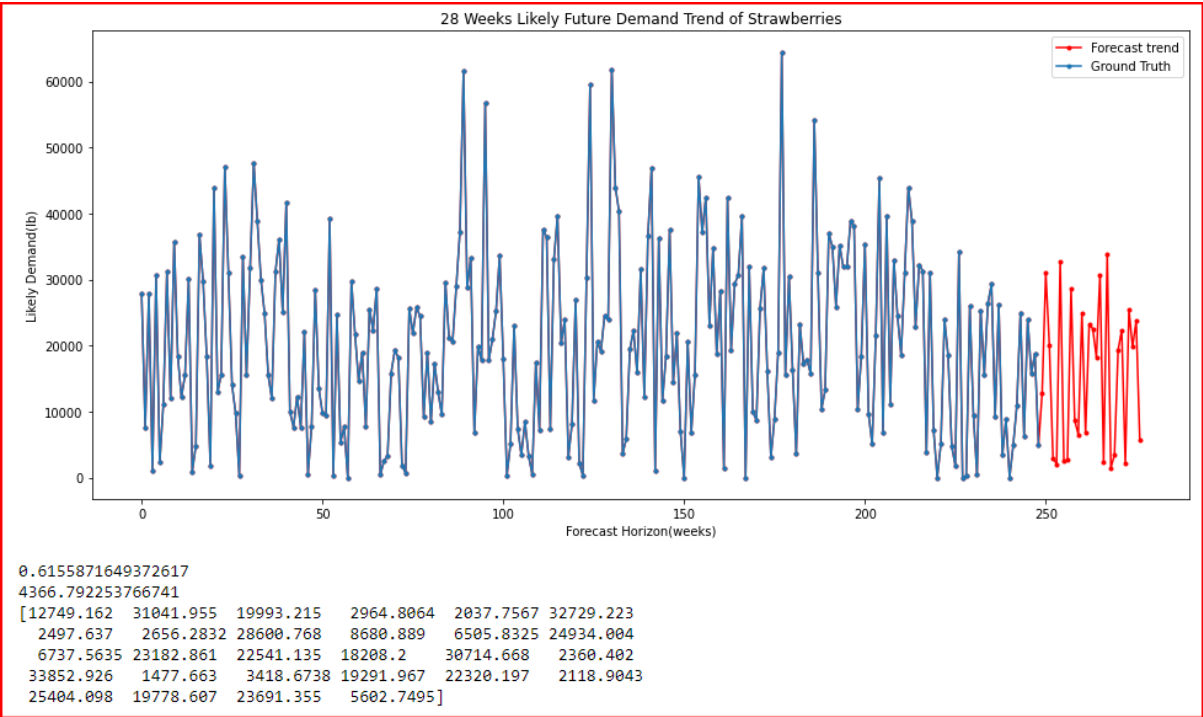


Figure B3: Attention-based CNN-LSTM 28 weeks forecast