A PROJECT REPORT

on

"Dynamic Crop Price Prediction utilizing Deep Learning Techniques using Time Series Data"

Submitted to

KIIT Deemed to be University

In Partial Fulfillment of the Requirement for the Award of

BACHELOR'S DEGREE IN

COMPUTER SCIENCE & ENGINEERING

BY

SOHAM DAS	21051942
SHIVAM MISHRA	21051932
SHREYAS PANDE	21051936
GARVITA MISHRA	2105459

UNDER THE GUIDANCE OF

MR. PRADEEP KANDULA

Acknowledgements

We are profoundly grateful to **MR. PRADEEP KANDULA** of **Affiliation** for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion. ....

SOHAM DAS
SHIVAM MISHRA
SHREYAS PANDE
GARVITA MISHRA

ABSTRACT

Our project aimed to address the challenge of predicting crop prices dynamically by harnessing the power of deep learning algorithms. Through the utilization of recurrent neural networks (RNNs), long short-term memory networks (LSTMs), gated recurrent units (GRUs), and bidirectional variants, we sought to develop a robust model capable of accurate price forecasting.graph 1.

Traditional methods of crop price forecasting often struggle to capture the complexity and volatility of agricultural markets, leading to suboptimal decision-making for farmers, traders, and policymakers. Unlike static forecasting models, the dynamic approach continuously updates predictions in real-time, enabling stakeholders to respond swiftly to sudden price fluctuations, supply chain disruptions, and other external factors.

Contents

List of Figures

Chapter 1

Introduction

The project aims to develop a dynamic crop price prediction system based on time series data. This system addresses the pressing need for accurate forecasting in agricultural economics, particularly focusing on predicting crop prices. Current solutions often lack accuracy and fail to incorporate advanced Deep learning techniques, hence the need for a more robust predictive model. Predicting future crop prices can empower farmers to make informed decisions about planting, harvesting, and selling their crops.

Chapter 2

Basic Concepts/ Literature Review

2.1 NN Techniques

The set of models used below have been chosen due to the fact that the input data is a time series data. The models are based on RNN that has the property to preserve the contextual nature of the data, which the Normal Neural Networks can not do. The models that have been compared are as follows:

2.1.1 Recurrent Neural Networks (RNN): RNNs are specialized neural networks capable of handling sequential data by retaining memory of past inputs i.e. the data of the last step is used as input in the current step. They possess the ability to detect temporal dependencies making them good for time series prediction tasks. Thus using it was always the first choice in this study.

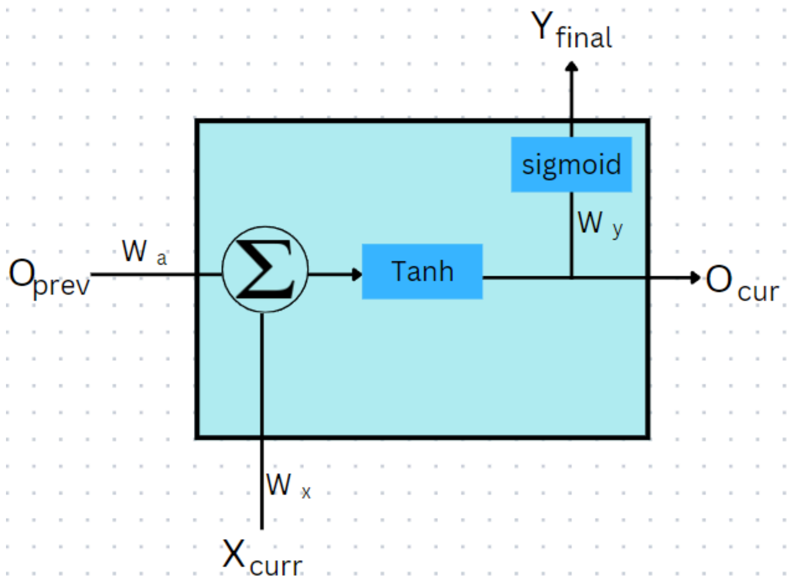


Fig1 Single cell of RNN

Let's assume there is a single processing unit in the RNN. This unit receives two inputs:

- Current Input (Xcur): This represents the data that has been feed to the network at the current time step.

- Previous Hidden State (Oprev): This comes from the previous processing unit in the RNN. It essentially captures the information the network processed earlier, acting like a kind of memory.

Weights are assigned to the inputs:

1. Wx: Weight for the current input (Xcurr).
2. Wa: Weight for the previous hidden state (Oprev).

These two inputs are then combined using a weightsum:

Equation 1: $a(t) = \tanh(Wx.Xcurr + Wa.Oprev + ba)$
Equation 2: $y(t) = \text{sigmoid}(Wy.a(t) + by)$

In equation 1, dot products of weights & inputs at two different time steps are summed up and adding bias. The linear equation is then passed through an activation function 'tanh' to squash the values between -1 and 1 which gives Ocurr. The second equation calculates the final output. It takes the function Ocurr and multiplies it by another set of weights (Wy). A bias term (by) is added, and the result is passed through the sigmoid function.

2.1.2 LSTM Sequential: They are a powerful type of recurrent neural network (RNN) architecture widely used in deep learning. They excel at processing sequential data, where the order of elements is important, and capturing long-term dependencies within those sequences. A Sequential model is a plain stack of layers where each layer has exactly one input tensor and one output tensor.

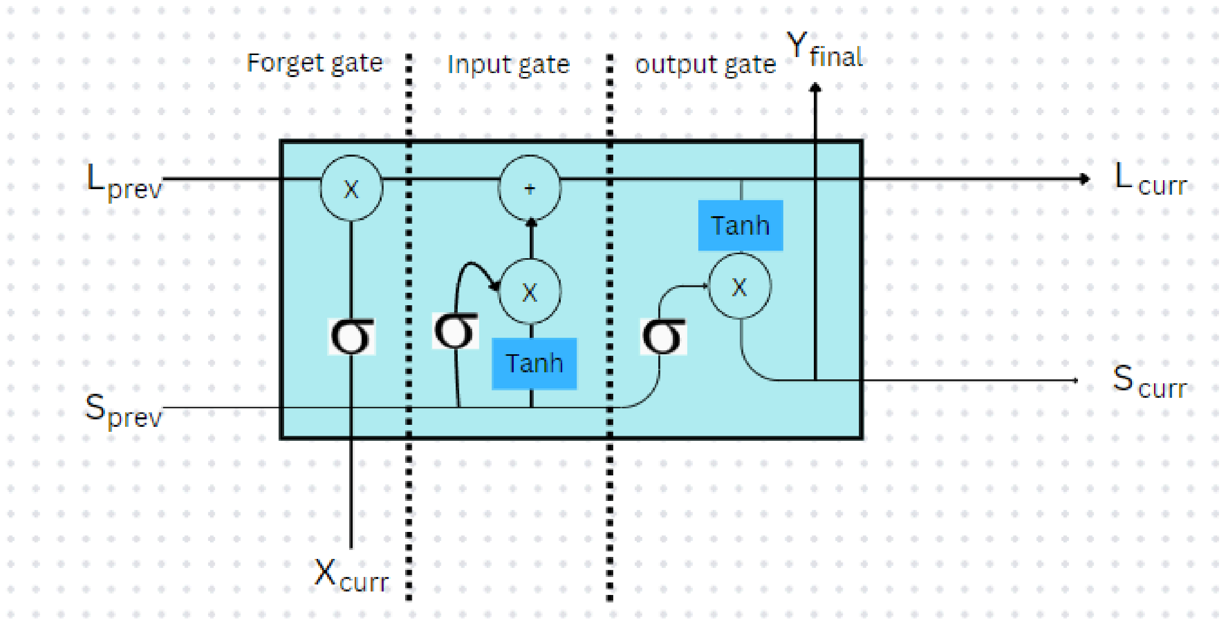


Fig2 Single cell of LSTM

Meaning of terms in Fig2:
Lprev: Previous state of the Long term memory
Sprev: Previous state of the Short term memory
Xcurr: Current Input
Lcurr: Current state of Long term memory
Scurr: Current state of Short term memory

LSTMs focusing on the cell state and gates:

- Cell State: It is the representation of Long Term Memory. It carries information across the entire network. (Lprev)
- Hidden State: It can be said as Short Term Memory. It is the working memory and is passed to the next state to retain the latest information.(Sprev)

- Forget Gate: This gate decides what to discard from the cell state. Like a filter, it examines the information and assigns a forgetfulness score between 0 and 1. A score of 0 tells the cell to completely forget that piece of information, while 1 signifies remembering it all.

- Input Gate: The input gate controls what new information gets added to the cell state. It has a three-step process:
    1. Filtering Incoming Information
    2. Creating New Information Candidates:
    3. Merging Old and New

- Output Gate: This gate controls what information from the cell state is sent as output and contributes to the next cell's hidden state. It also follows a three-step process:
    1. Preparing the Information
    2. Filtering the Output:
    3. Sending Out the Filtered Information

2.1.3 Gated Recurrent Unit(GRU): GRU, which stands for Gated Recurrent Unit, is a type of recurrent neural network (RNN) architecture designed specifically for processing sequential data in deep learning.The basic idea behind GRU is to use gating mechanisms to selectively update the hidden state of the network at each time step. The gating mechanisms are used to control the flow of information in and out of the network. The GRU has two gating mechanisms, called the reset gate and the update gate.

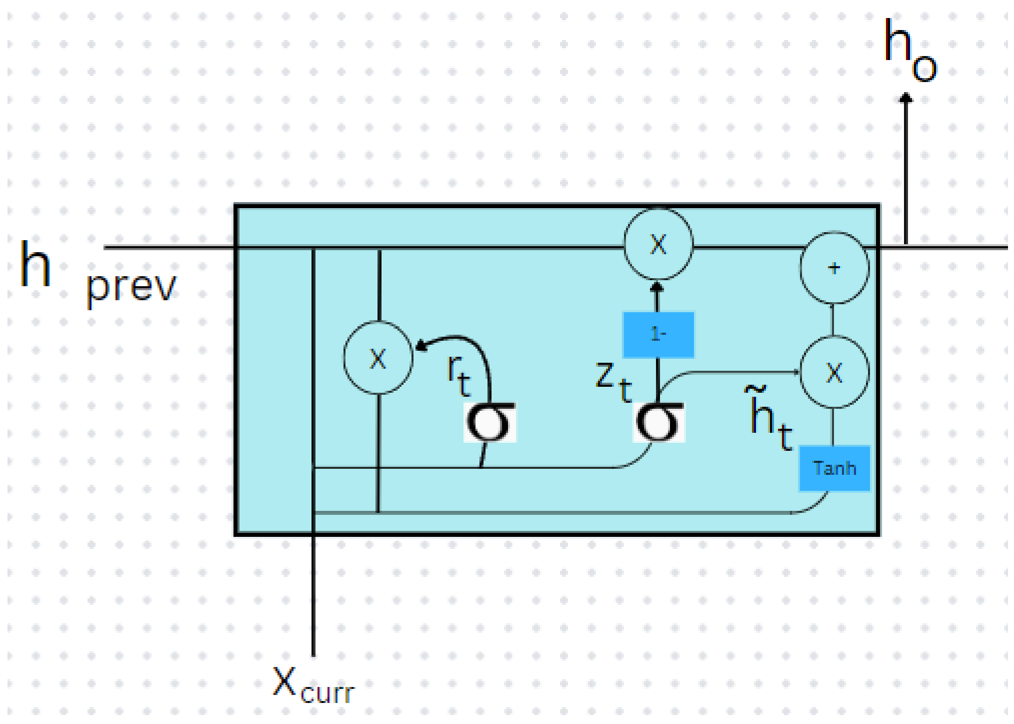


Fig3 Single cell of GRU

Meaning of terms in Fig3:
hprev: Previous hidden state
Xcurr: Current Input
rt: Reset Gate
Zt:Update gate
~ht: Candidate Hidden State
ho: Current Hidden State

GRUs are a simpler take on LSTMs, designed to avoid getting overwhelmed by short-term information. While LSTMs use three gates for memory control, GRUs achieve similar results with just two:

- Reset Gate: This gate acts like a filter for the GRU's memory. It analyzes the past information (hidden state) and decides how much to forget, discarding irrelevant short-term details.
- Update Gate: This gate controls the flow of new information. It considers both the filtered memory (thanks to the reset gate) and the current input to decide what gets incorporated into the GRU's memory.

2.1.4 **Bidirectional LSTM:** A Bidirectional Long Short-Term Memory (BiLSTM) network is a type of recurrent neural network (RNN) particularly well-suited for processing sequential data, especially in natural language processing (NLP). Unlike standard LSTMs which process information in one direction, BiLSTMs leverage two LSTM layers. One layer processes the sequence forward (left to right), while the other processes it backward (right to left). This allows the model to capture dependencies between elements at any position in the sequence, leading to a more comprehensive understanding of the relationships between words or elements within the sequence**.**

2.1.5 Bidirectional RNN: A special form of RNN that allows data flow bidirectionally i.e. the flow of data can be observed in both backward and forward direction thus enhancing the ability to read patterns. This is achieved by using two recurrent networks for each directional flow of data. They can be called an upgrade to the general model to which bidirectionality is applied, thus helping in efficiently analyzing the contextuality of data. We have chosen this or any bidirectional model, how much will the accuracy increase if we have higher computation(using two models at the same time).

2.1.6 Bidirectional GRU: Just like other bidirectional models, BiGRU is also a type of neural network where two GRUs are used in order to process data in normal and reverse.

2.2 Data Preprocessing Techniques

2.2.1 MinMax Scaling: It is a Normalization technique used over the data to ensure all features contribute equally to the model's learning process. It helps in taking the weights to a common scale by compressing the values in the range of [0,1].

2.3 Loss Function

2.3.1 Huber Loss: This is the loss function used in the study for checking the error. It has the ability to use both MSE(Mean Squared Error) and MAE(Mean Absolute Error) with the help of a parameter delta which acts as a threshold which helps in switching between the operations smoothly.

2.4 Optimizer used

2.4.1 Adam Optimizer: The optimizer used in the project is Adaptive Moment Estimation(Adam Optimizer). It is a derivative of the Root Mean Square Propagation(RMSprop) and Mini Batch Gradient Descent(Momentum). It is used in the gradient descent to minimize the loss function. One of the advantages with Adam is that it calculates a unique learning rate for each parameter in the neural network. This is very useful for noisy data.

Chapter 3

Problem Statement

The problem we are working on is Dynamic Crop Price Predictions. Our objective is to develop predictive models capable of forecasting crop prices accurately. In the problem we are asked to use historical price data for training predictive models, specifically RNN architectures as data being used will be in time series format. For evaluation of the models we shall be using standard regression metrics to ensure accuracy of the models.

To summarize above:

- Utilizing historical price data for training the predictive models.
- Implementing advanced Neural Network algorithms, particularly RNN-based architectures, to effectively capture the time-series nature of the data.
- Evaluating the performance of the models using standard regression metrics to ensure reliability and accuracy.

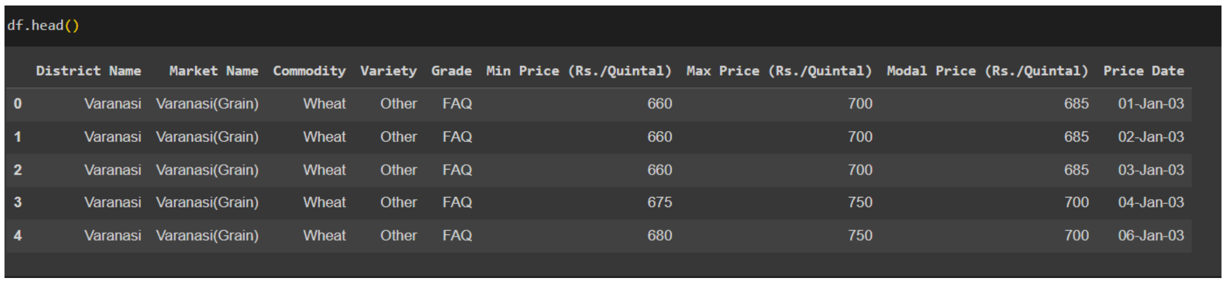
Chapter 4

Implementation

In this section we shall be discussing the methodology we used while making the program and how we have tested it.

4.1 Data Set Analysis

The dataset we have used is of Varanasi district. The dataset was extracted from a government website (agmarket.gov.in). The head of the file can be seen in the image below. As described before the data is time dependent i.e. while train-test-split we have to remember to preserve the time dependency. Also the price predicted will be in the unit Rs/Quintal. The price used in the modal is Modal price. The commodity whose price is being predicted here is wheat.



```
df.head()
```

	District Name	Market Name	Commodity	Variety	Grade	Min Price (Rs./Quintal)	Max Price (Rs./Quintal)	Modal Price (Rs./Quintal)	Price Date
0	Varanasi	Varanasi(Grain)	Wheat	Other	FAQ	660	700	685	01-Jan-03
1	Varanasi	Varanasi(Grain)	Wheat	Other	FAQ	660	700	685	02-Jan-03
2	Varanasi	Varanasi(Grain)	Wheat	Other	FAQ	660	700	685	03-Jan-03
3	Varanasi	Varanasi(Grain)	Wheat	Other	FAQ	675	750	700	04-Jan-03
4	Varanasi	Varanasi(Grain)	Wheat	Other	FAQ	680	750	700	06-Jan-03

Fig4. First five entries of data set

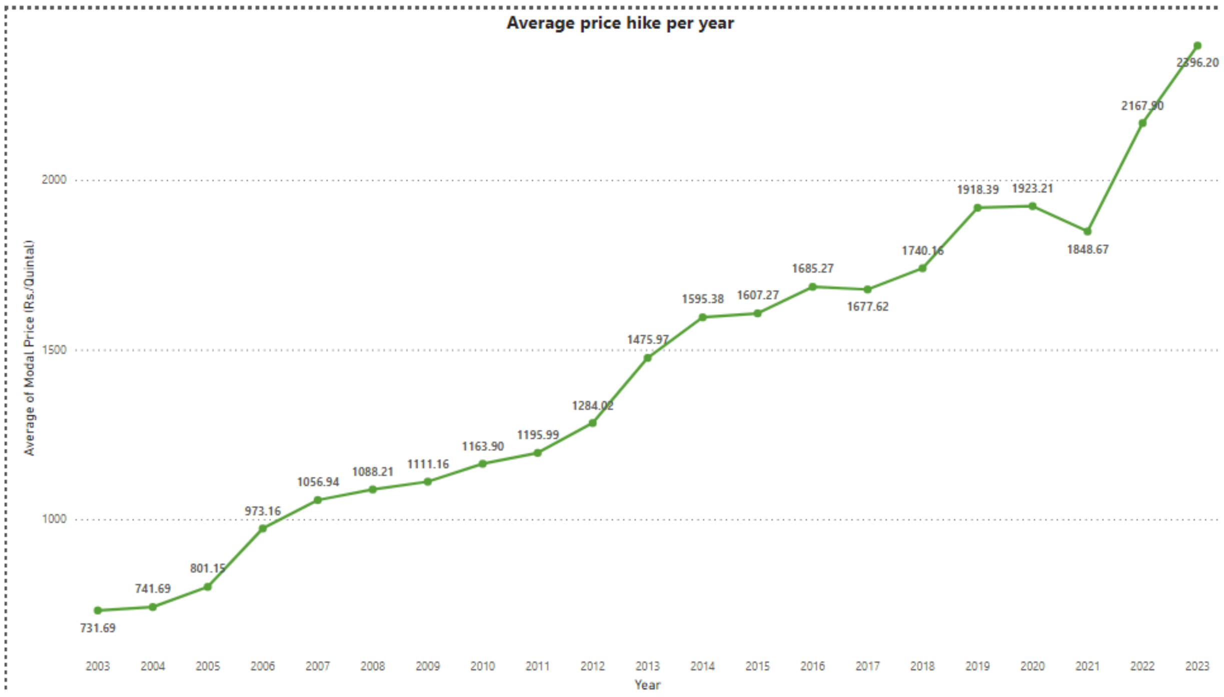


Fig5. Average price hike per year

Some observations can be made even before we use models. The data is of 20 years from 2003 to 2023. Firstly it is observed that every year the average price of wheat is increasing. This can be seen clearly in Fig5.

except for the year 2021 when the pandemic hit India. Also it is observed that the prices are on a huge hike after 2020. After the years 2005, 2006, 2012, 2013 and 2020, the prices of the grains suddenly hiked, maybe because of the elections which were held during that time.

If we go deeper and look into the graphs each year, it would look as it is seen in the Fig6. From these graphs, we see that the prices start to fall during the time of May till June.

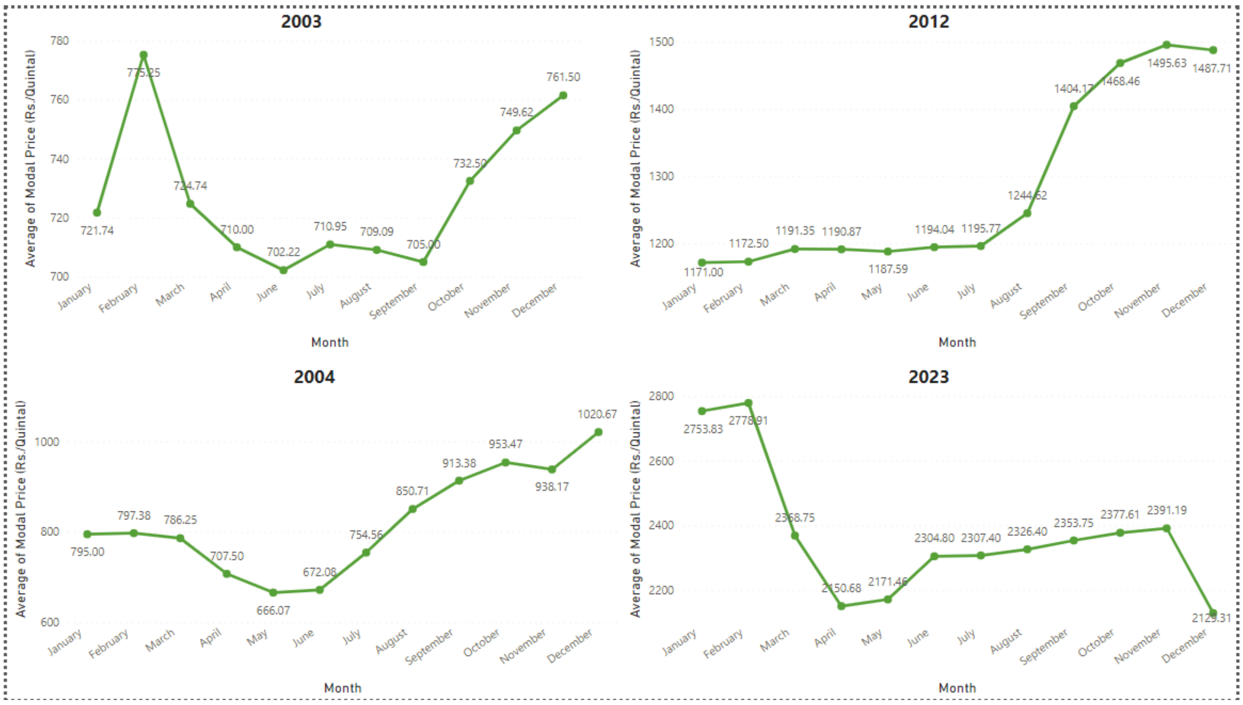


Fig6. Price pattern over an year, for four different years

4.2 Methodology

The project employs various RNN architectures, including classical RNN, LSTM, GRU, Bidirectional RNN, Bidirectional LSTM, and Bidirectional GRU, for price prediction. *The number of epochs used for training is 50 and data is sent in batches of 1 year.* The implementation encompasses cleaning the data first before uploading the dataset which includes removal of the 0 values. After cleaning Data is preprocessed in which features are scaled using MinMaxScaler. Lastly, we convert the data in time series format by converting it into vectors. After this comes the computing part, but before that we have to split the data into training and testing sets. Here we have specially defined the function *train_test_split()*. This is because the data that we are working on is in time series format and in-order to preserve the order we have created this function. Now we have trained six models on the basis of the historical price data(training set). After training each model we calculate the predicted value and after inverse transformation model performance is evaluated using standard regression metrics eg. Mean Absolute Error(MAE) and Root Mean Square Error(RMSE). At the end we have plotted two graphs in which we have compared the original values with the predicted values which we obtained from the model after using training and testing data.

4.2 Testing

Testing involves validating the trained models on unseen data to assess their generalization capabilities. Before testing we have to unscale the scaled predicted values.The verification criteria include:

MAE- Mean Absolute Error: it is a metric used to evaluate performance of a regression model. MAE calculates the average of the absolute differences between the predicted values by your model and the actual values you're trying to predict.

MSE- Mean Square Error :When building statistical models, assessing their performance is essential. Mean Squared Error (MSE) plays a vital role in this evaluation. It measures the average squared difference between the predicted values by our model and the actual values we're trying to estimate. Here, lower values indicate better accuracy.

Root Mean Square Error(RMSE) : is the method of finding residuals in the data set. Here we take the square of the difference in values(Actual - Predicted) and take the mean of all the values. Lastly, we take the square root of the final value. Here lower values indicate better accuracy.

R2 Score: it tells us about how well the model expresses the variance in the dependent variable based on independent variables. R-squared represents the proportion of variance in the dependent variable that can be attributed to the independent variable included in the model. 0 indicates the model explains none of the variance(terrible fit), while 1 indicates the model explains all the variance(perfect fit).Higher values indicate better fit to the data.

Chapter 5

Standards Adopted

In order to have a structured view and working we have defined certain features to make sure the project goes on track.

5.1 Code Standards

Certain coding standards are maintained throughout the project to ensure readability and maintainability. We have tried to use a consistent use of appropriate naming conventions throughout the code. We used segmentation or block format of execution where we have divided the code into different cells for better readability. Proper indentation and documentation have been done for better understanding. Code reusability for decreasing the number of lines of code can be seen while we try to test the prediction.

5.2 Testing Methods

The project follows industry-standard testing practices to ensure the reliability and quality of the predictive models. Testing standards include adherence to ISO and IEEE guidelines for quality assurance.

Chapter 6

Results:

Overall, the project lays a solid foundation for leveraging Neural Network techniques to address challenges in agricultural economics, with potential applications beyond crop price prediction. For each model we have two graphs: one for training data and one for testing data. Each graph is the comparison of Actual data vs Predicted data.

6.1 RNN

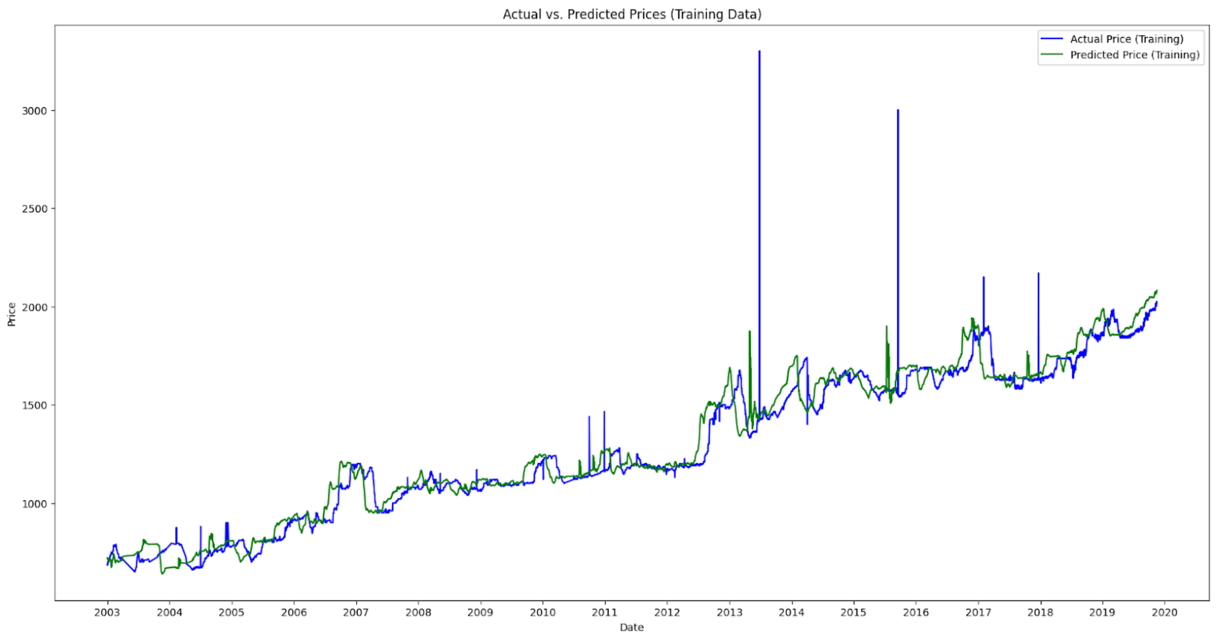


Fig7. Training data graph for RNN

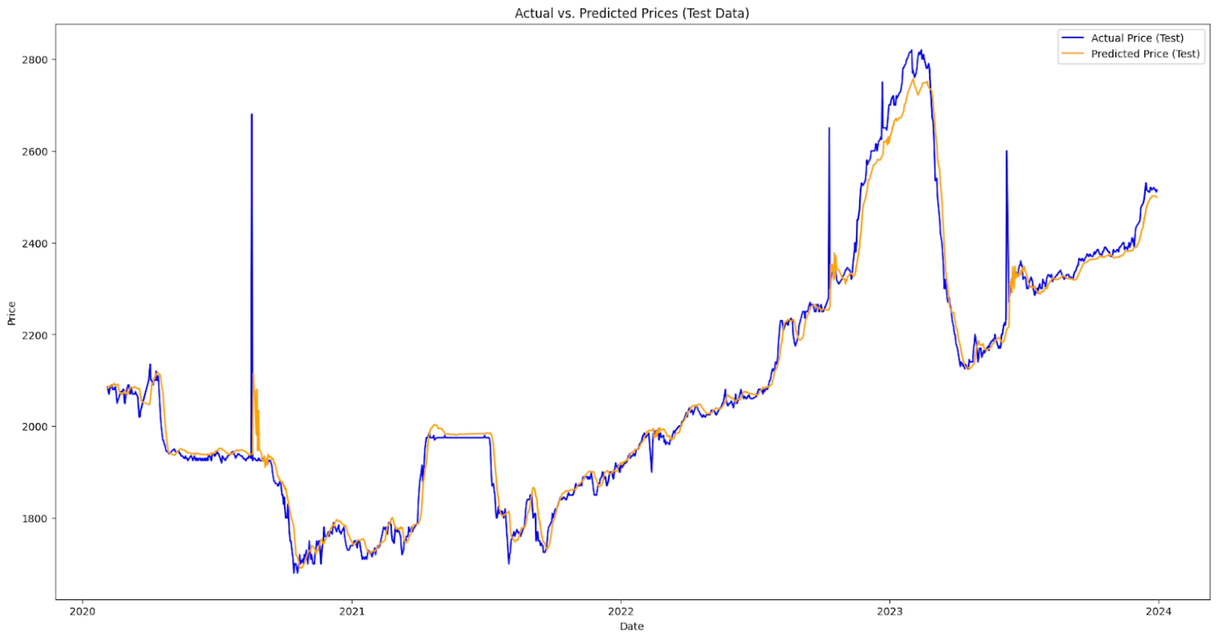


Fig8. Testing data graph for RNN

6.2 Bidirectional RNN:

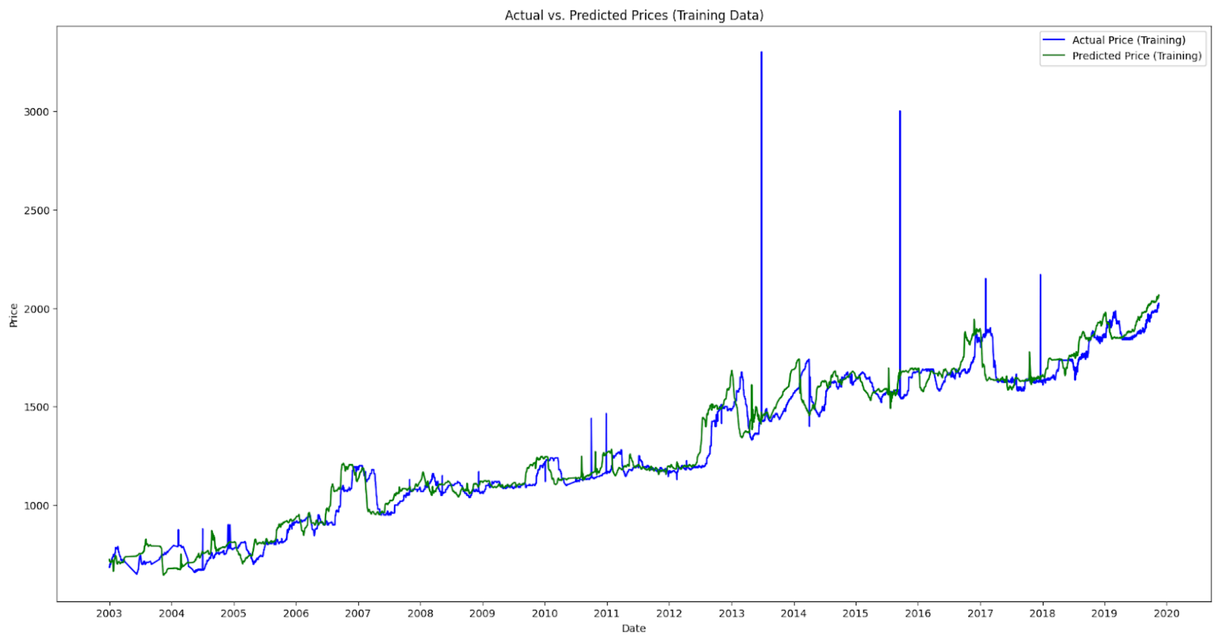


Fig9 Training data graph for Bi RNN

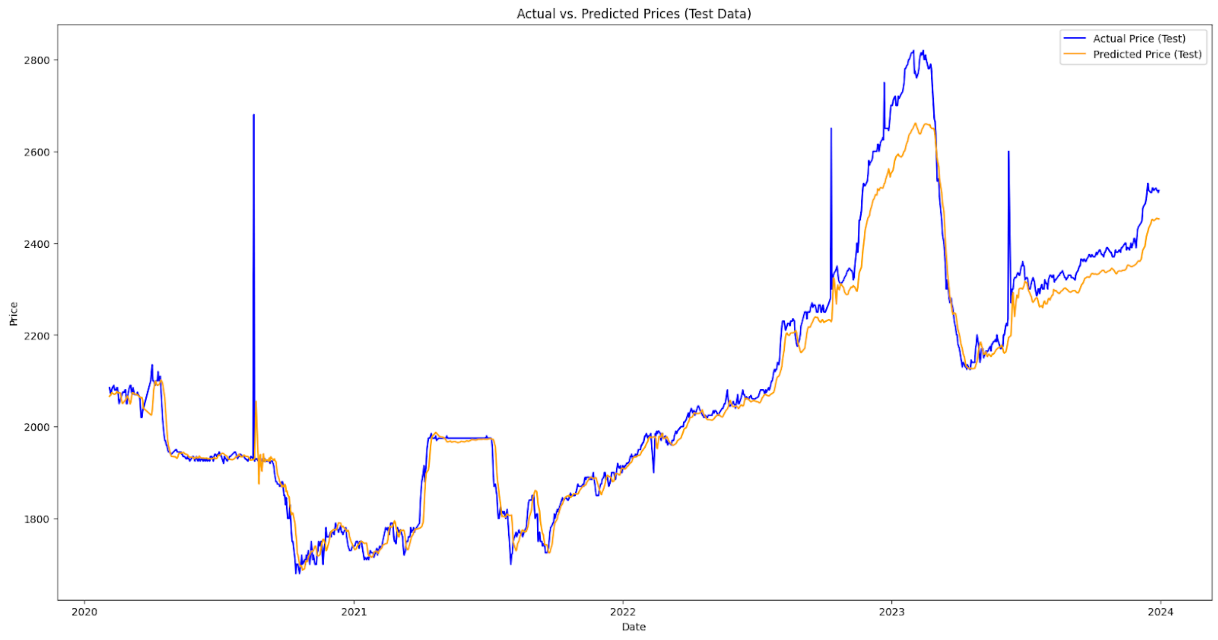


Fig10. Testing Data for Bi RNN

6.3 LSTM:

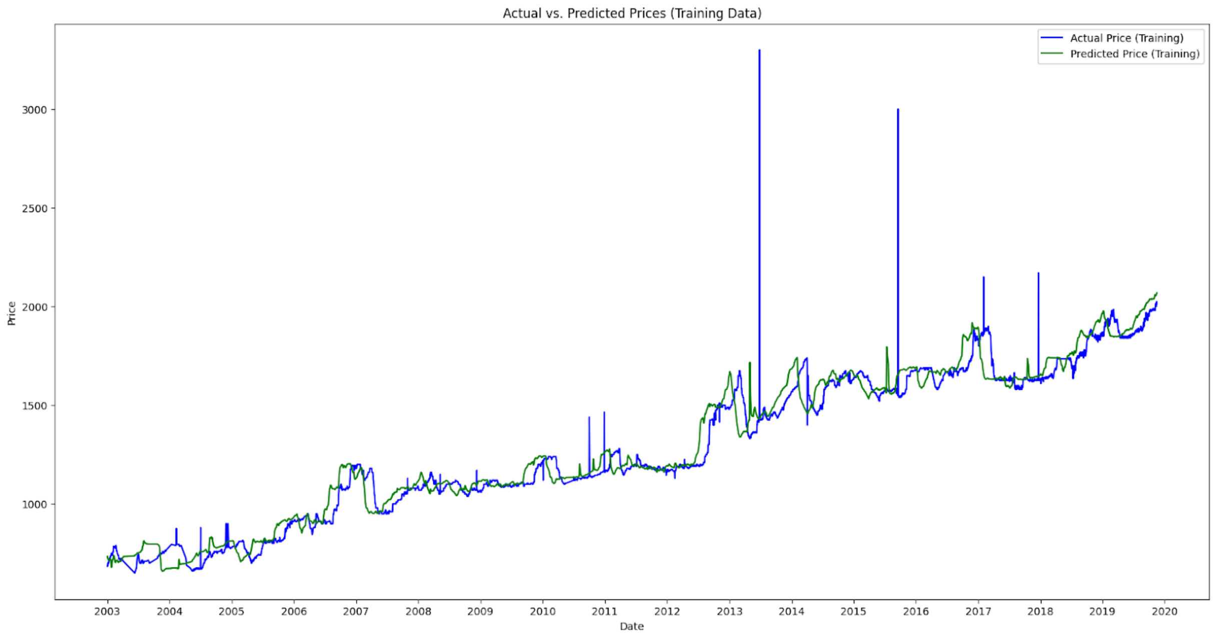


Fig11. Training data graph for LSTM

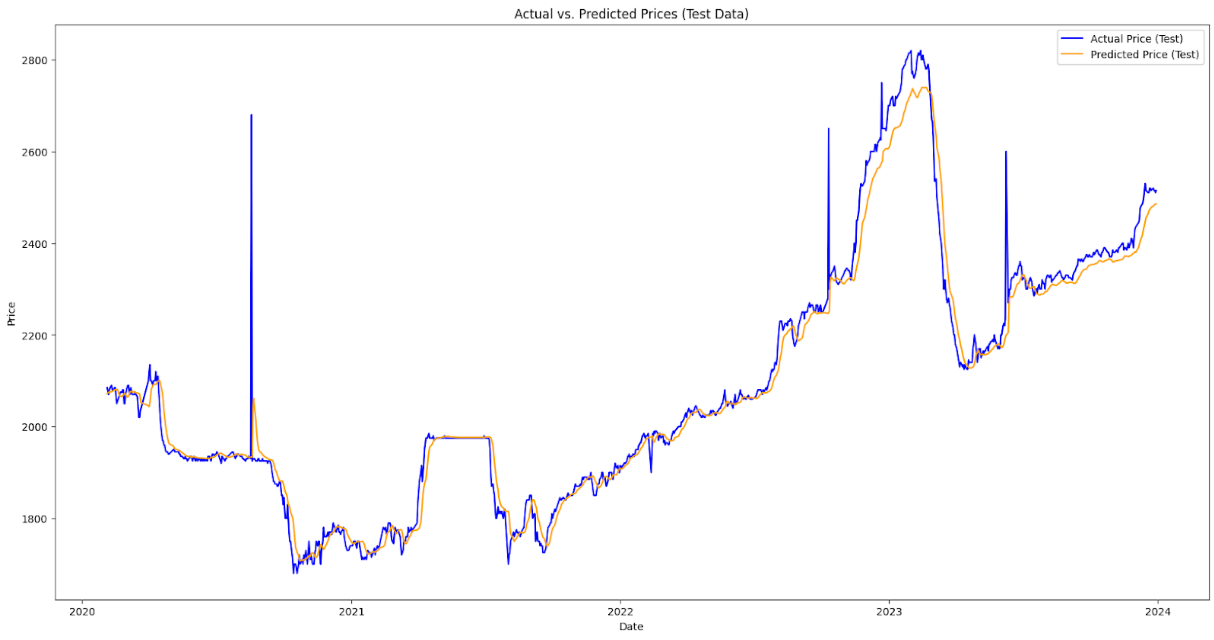


Fig12. Testing data graph for LSTM

6.4 Bidirectional LSTM:

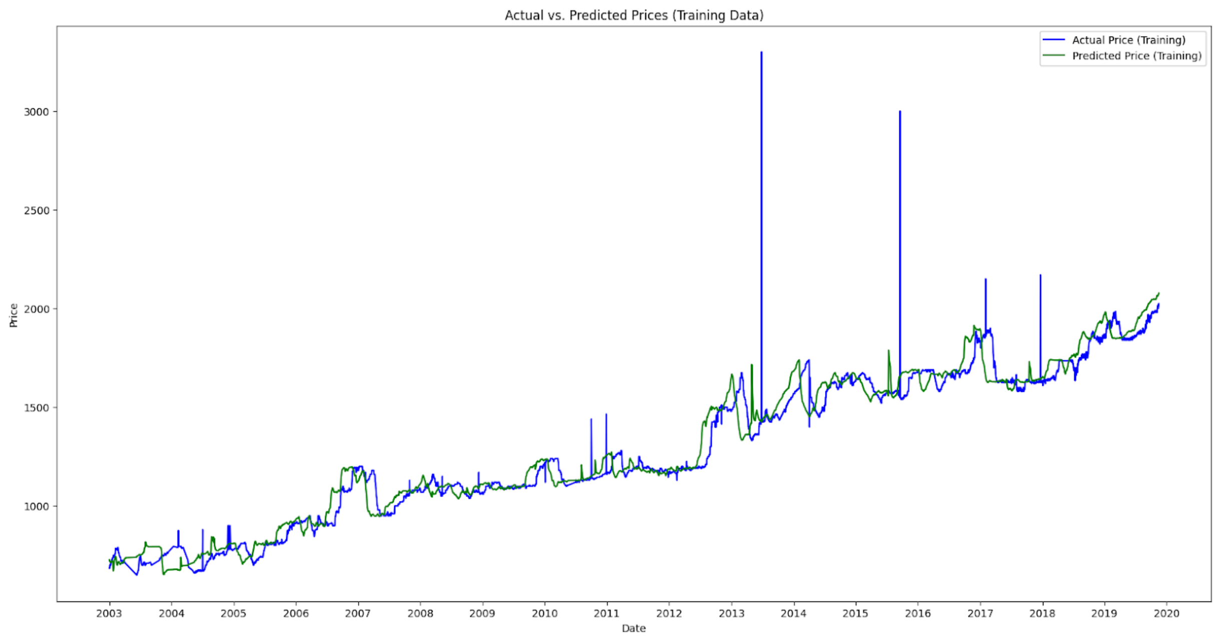


Fig13. Training data graph for Bi LSTM

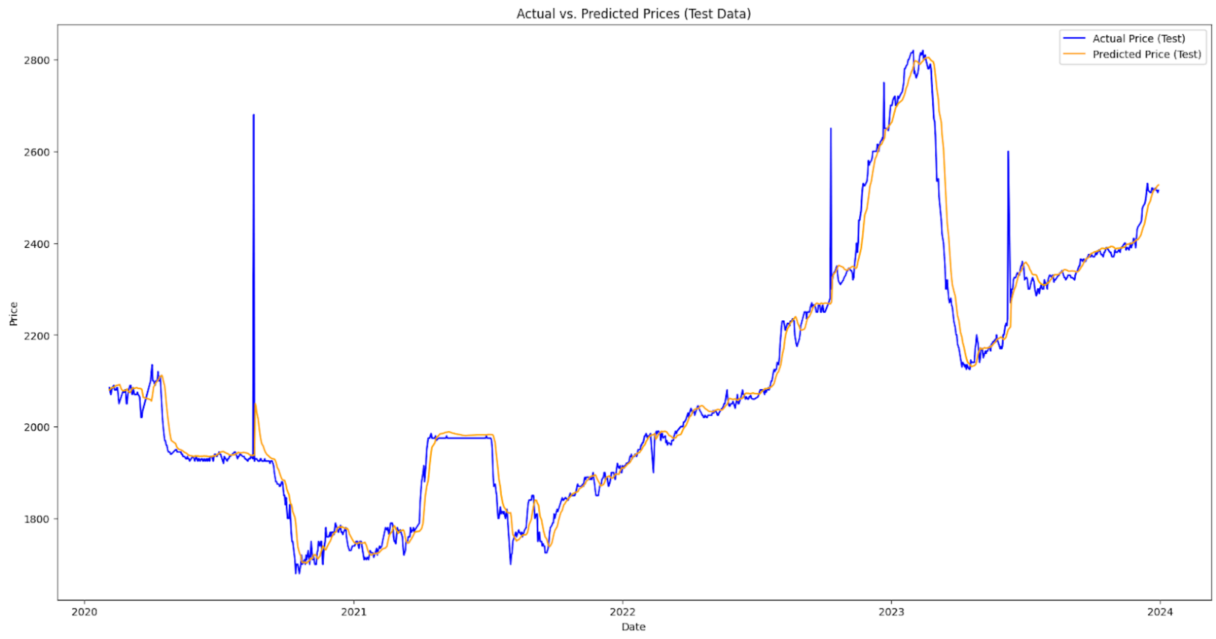


Fig14. Testing data graph for Bi LSTM

6.5 GRU:

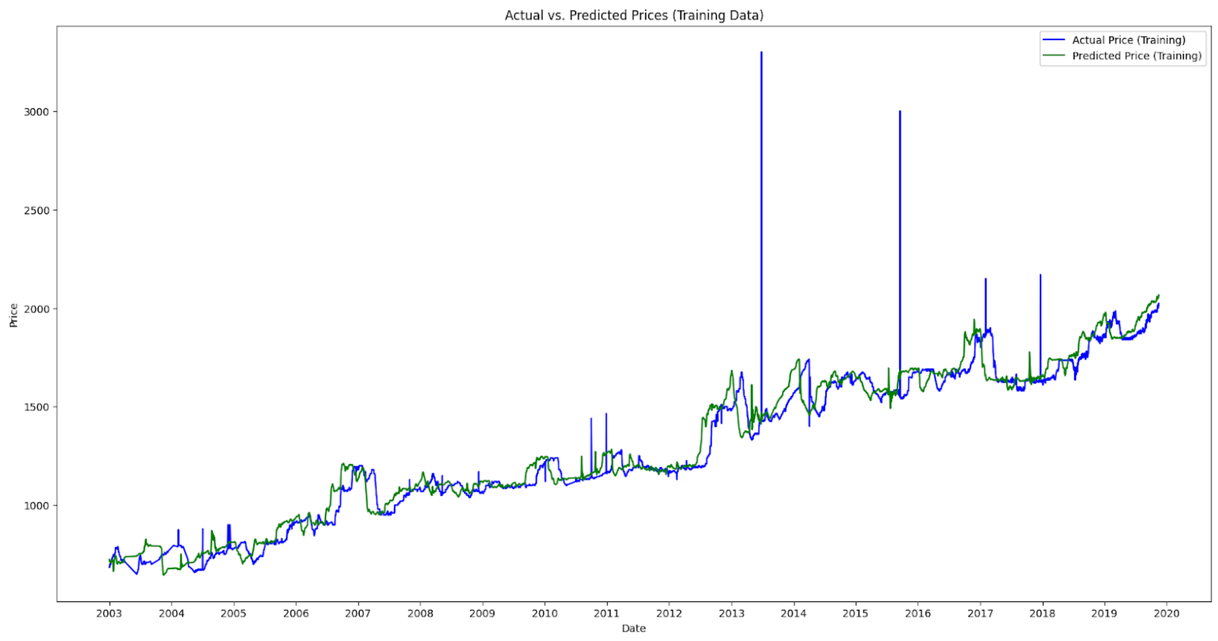


Fig15. Training data graph for GRU

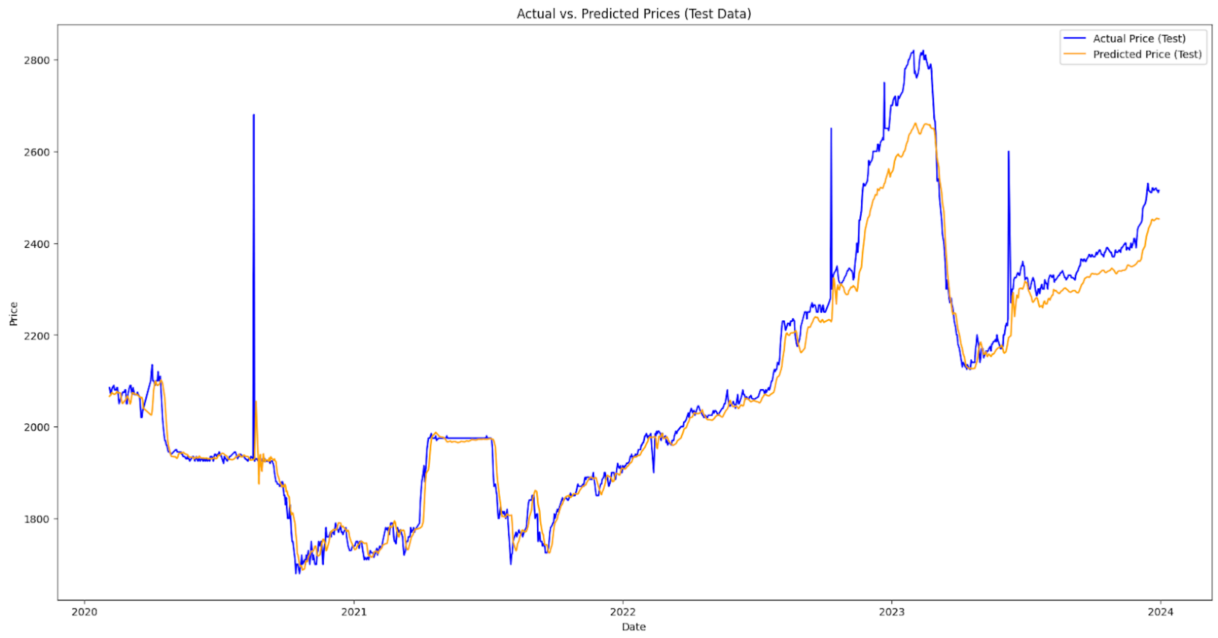


Fig16. Testing data graph for GRU

6.6 Bidirectional GRU:

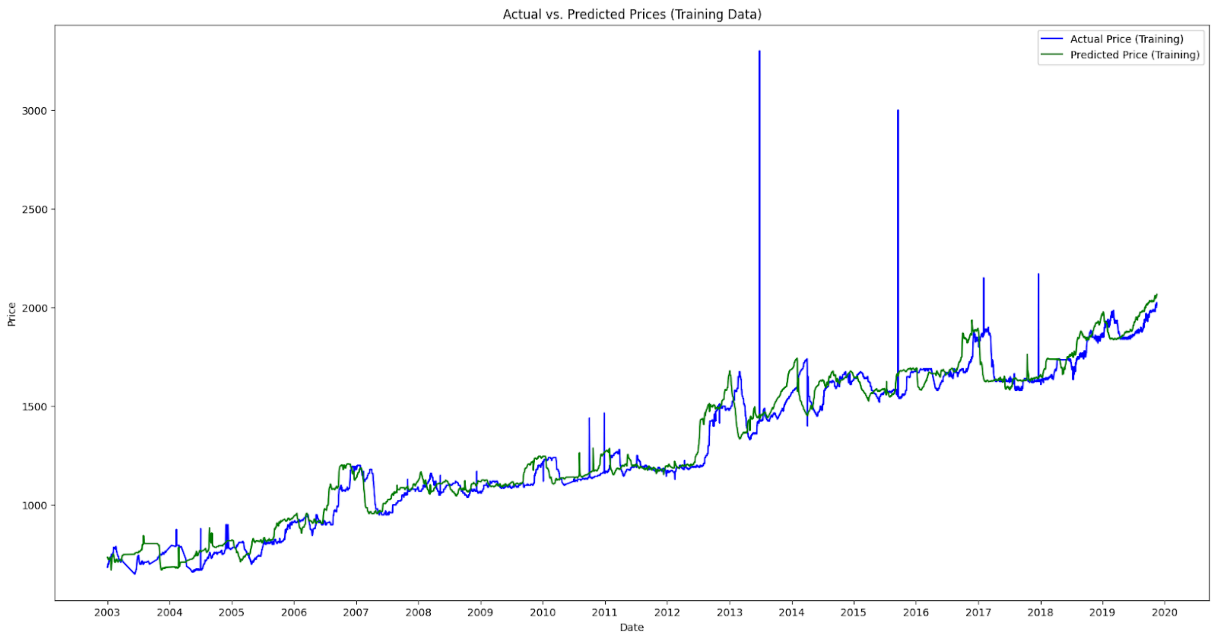


Fig17. Training data graph for Bi GRU

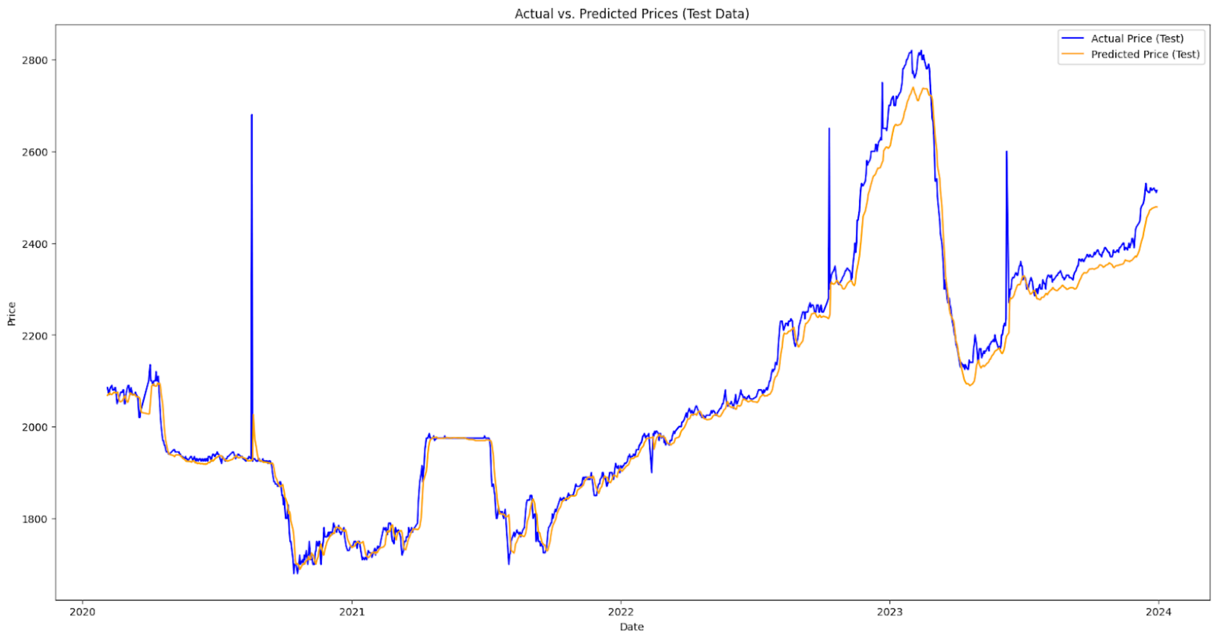


Fig18. Training data graph for Bi GRU

After testing the predicted data through all the parameters Mean Absolute Error, Mean Square Error, Root Mean Square Error, and R2 Score we have received values which have been compiled in the table Fig 19. For better understanding we have used graphs to compare different models for each of the parameters above.

Model	MAE (Train)	MAE (Test)	MSE (Train)	MSE (Test)	RMSE (Train)	RMSE (Test)	R2 Score (Train)	R2 Score (Test)
RNN	12.2	20	1849.12	1557.09	43	39.46	0.9853	0.9791
LSTM	10.83	23.67	1722.02	1891.45	41.5	43.49	0.9863	0.9746
GRU	14.37	32.05	1726.49	2137.62	41.55	46.23	0.9862	0.9713
Bi RNN	10.41	29.67	1620.93	2787.53	40.26	52.8	0.9871	0.9625
BiLSTM	10.56	20.58	1721.9	1769.51	41.5	42.07	0.9863	0.9762
BiGRU	9.94	22.67	1574.19	1718.84	39.68	41.46	0.9874	0.9769

Fig19. Total compilation of all the testing

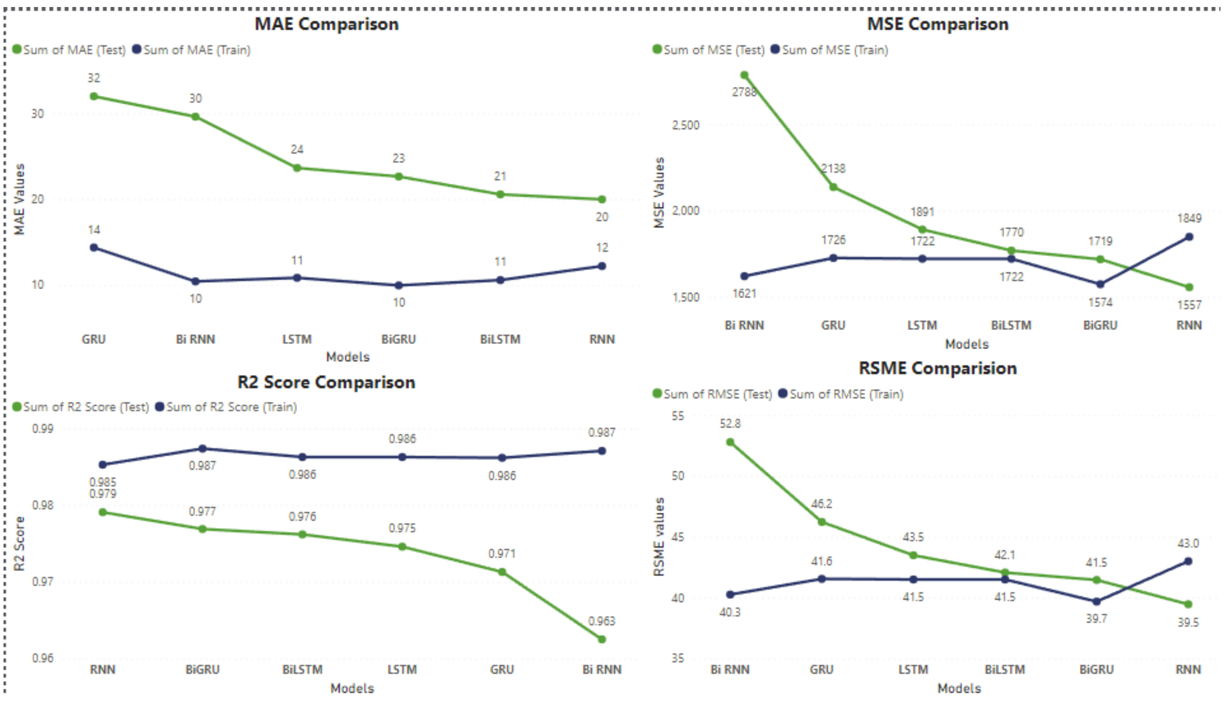


Fig20. Graphical Comparison of all the models on basis of testing techniques

Chapter 7

Conclusion and Future Scope

Analyzing the Fig19*,* it is evident that the Bidirectional LSTM model outperforms other models across various evaluation metrics. It achieves the lowest MAE and MSE values on both training and testing data, indicating better accuracy and predictive capability.

Additionally, it exhibits the highest R2 score on both training and testing data, signifying a better fit to the underlying data distribution. Therefore, the Bidirectional LSTM model can be considered the best model for dynamic crop price prediction in this scenario.

7.1 Conclusion

The project demonstrates the effectiveness of RNN-based architectures in dynamic crop price prediction. The implemented models exhibit promising performance metrics, indicating their potential utility in agricultural economics for decision support.

7.2 Future Scope

Some models that can be used for prediction of crops, withstanding the dynamicity of the data, CNN, Sequence to sequence and Transformers can be used for future work purposes.