Predicting potato price series using N-BEATS deep learning architecture

Abstract

Agricultural commodity prices exhibit unique challenges due to seasonality, inelastic demand, and production uncertainty, leading to significant fluctuations in time series data. This paper explores these complexities by applying Deep Learning (DL) models to forecast agricultural prices, specifically focusing on potato prices. While DL models have excelled in domains like image processing and natural language processing, they require specialized architectures for effective time series forecasting. This study evaluates the Neural Basis Expansion Analysis for Interpretable Time Series Forecasting (N-BEATS) model, a novel DL architecture designed for time series datausing daily potato price data from the Azadpur market in Delhi, spanning January 1, 2018, to April 30, 2023.The performance of N-BEATS is compared with three baseline models: Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRU). Evaluation criteria include Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE). The results show that the N-BEATS model consistently outperforms the other models across all metrics. Additionally, the Diebold-Mariano (DM) test confirms the N-BEATS model's superior forecasting accuracy compared to the other models. This research highlights the potential of the N-BEATS model to significantly enhance the precision of agricultural price forecasting, providing valuable insights for farmers, planners, and other stakeholders in the agricultural sector.

Keywords: *Potato price, Basis expansion, Convolutional Neural Network (CNN), Deep learning, Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), N-BEATS*

1. Introduction

Vegetables play a pivotal role in global agriculture, contributing significantly to the nutritional needs of the population and the economy. Among them, the potato stands out as one of the most important crops, ranking as the third most essential food crop in the world after rice and wheat, with a rich history originating in the Andes Mountains of South America(Yadav et al. 2024). In the fiscal year 2023, India witnessed a noteworthy increase in potato production, reaching approximately 59.74 million metric tons, with Uttar Pradesh contributing significantly to this surge (Sahu et al. 2024). The potato is not only a staple in the Indian diet but also a key economic crop, providing livelihoods for millions of farmers and contributing substantially to the country's agricultural GDP. Given its widespread consumption and economic importance, the volatility in potato prices has significant implications for both producers and consumers. Accurate price forecasting of potatoes is essential to stabilize markets, guide farming decisions, and prevent economic losses. This need for precise forecasting is heightened by the crop's susceptibility to factors such as weather conditions, storage limitations, and market demand fluctuations(Jamuna et al. 2021).

Time series analysis has long been the cornerstone of forecasting in various domains, including agriculture. Over the decades, a range of techniques has been developed to improve the accuracy of predictions. The Autoregressive Integrated Moving Average (ARIMA) model, introduced in the mid-20th century, was among the first to systematically address time series forecasting(LIU 2024). ARIMA's strength lies in its ability to model linear relationships in time series data, but its limitations became apparent as it struggled with non-linear patterns and complex seasonal effects(Racocha 2020). To overcome these limitations, machine learning (ML) techniques emerged, offering more flexibility and the ability to capture non-linear dependencies. Models such as Random Forests, Support Vector Machines (SVM), and Gradient Boosting Machines (GBM) provided more accurate predictions by learning from vast amounts of data. However, these models still required manual feature engineering and were often limited in their ability to capture long-term dependencies.

The advent of deep learning (DL) revolutionized time series forecasting by introducing architectures capable of automatically extracting features and learning intricate patterns from data. Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks were among the first DL models to be applied to time series data, excelling in capturing sequential dependencies(Hewamalage et al. 2021). However, they were often prone to issues like vanishing gradients and required large computational resources. Building on these advancements, the N-BEATS (Neural Basis Expansion Analysis Time Series) deep learning architecture represents the latest innovation in time series forecasting. Designed specifically to address the shortcomings of earlier models, N-BEATS can deliver highly accurate forecasts without the need for extensive domain knowledge or feature engineering(Bulatov 2020). Its ability to model complex patterns and its hierarchical structure make it particularly suitable for applications such as potato price forecasting, where market dynamics are influenced by a multitude of factors.

Numerous studies have explored statistical, ML and DL algorithms for forecasting prices in diverse domains. Conejo et al. (2005) proposed a novel technique for day-ahead electricity price forecasting using the wavelet transform and ARIMA models. (an et al. (2010) introduced a novel price forecasting method based on wavelet transform combined with ARIMA and GARCH models. Moving on to agriculture, Paul et al. (2022) explored the effectiveness of four ML algorithms for forecasting the wholesale price of Brinjal in major markets of Odisha, India.Jaiswal et al. (2022)presented a deep long short-term memory (DLSTM) model for agricultural price forecasting, which outperforms traditional time-delay neural network (TDNN) and ARIMA models. For stock price prediction, Mehtab and Sen (2020)performed an agglomerative approach that combines statistical, ML and DL models. For financial TS forecasting, (Patarwal et al. 2018) proposed ELM-AE, a ML and DL-based method that outperforms existing methods based on MSE.Avinash et al. (2024)put forth Hidden Markov (HM) guided DL models for forecasting agricultural commodity prices. Tripathi and Sharma (2023)found that deep neural networks (DNNs) outperformed LSTM and CNN-LSTM models in predicting Bitcoin prices using technical indicators. Durairaj and Mohan (2022) proposed a CNN-based model for price forecasting incorporating chaos theory, 1D CNN and polynomial regression. These studies demonstrate the effectiveness of ML and DL learning algorithms for price forecasting in various domains and highlight the importance of selecting appropriate models based on the characteristics of the data.

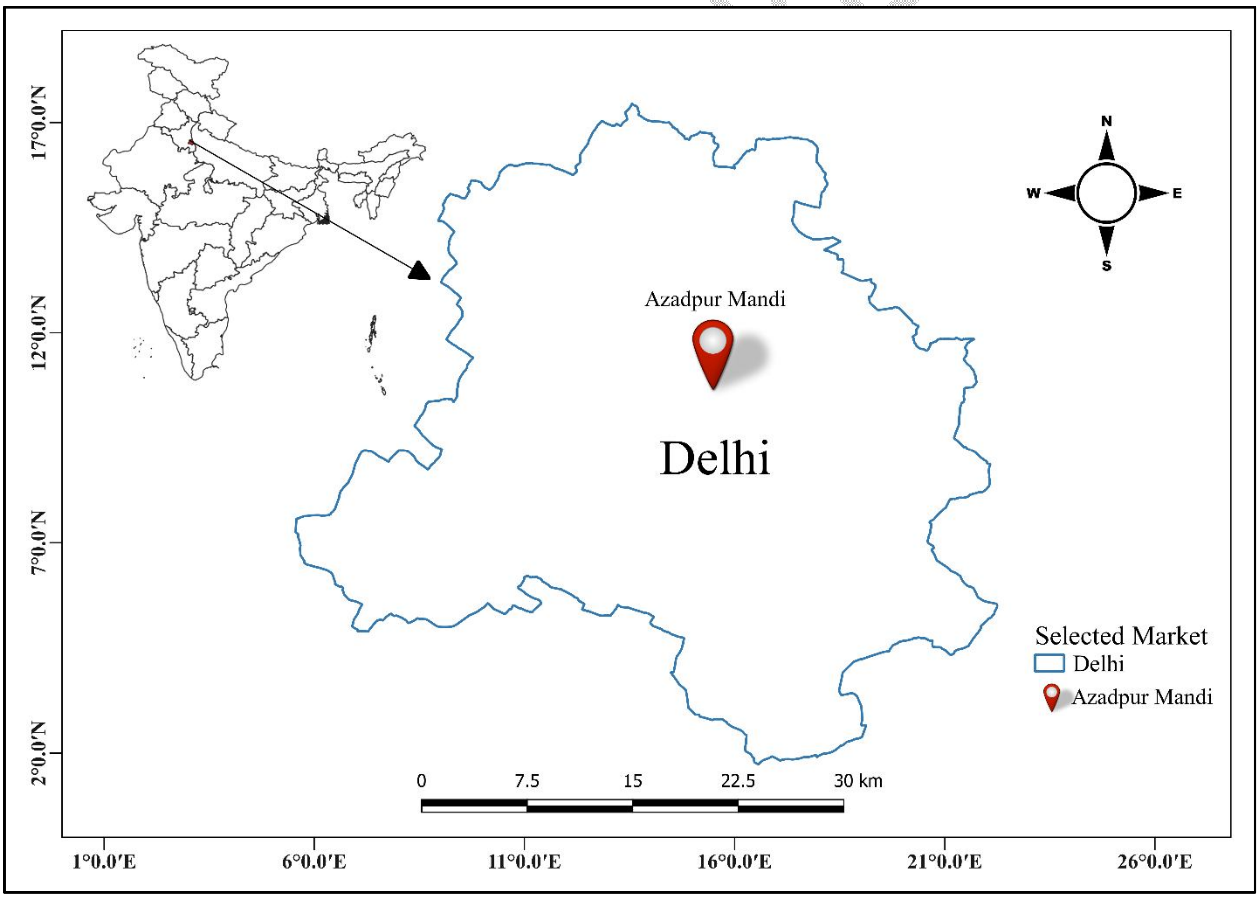
DL models initially found prominence in image processing, natural language processing (NLP) and computer vision (CV) applications, with later adaptations for TS forecasting. Oreshkin et al. (2021) introduced a specialized DL architecture for TS forecasting, named Neural Basis Expansion Analysis for Interpretable Time Series Forecasting (N-BEATS).Nayak et al. (2024a) utilized Deep learning techniques including NBEATS for improved forecasting of price of TOP crops in India.Jossou et al. (2022)proposed an N-BEATS-based model for predicting labour based on electro hysterography forecasting and Sbrana and Lima de Castro (2023) investigated its performance in forecasting cryptocurrency. This model aimed not only to deliver accurate predictions but also to enhance interpretability for end-users.

Hence, in this study, a novel deep learning approach, N-BEATS, was employed alongside baseline models such as CNN, LSTM, and GRU to forecast the potato price series in a key market in India. By leveraging these advanced models, the research aims to enhance the accuracy of price predictions, offering valuable insights into market dynamics. This comprehensive analysis not only highlights the effectiveness of N-BEATS in comparison to traditional models but also underscores its potential in agricultural price forecasting.

2. Materials and Methods

2.1Study Area and Data description

This study utilizes the daily potato price series data of Azadpur market, Delhi, which contains 1398 observations from January 1, 2018, to April 30, 2023. The data is sourced from the



'Agmarknet' website (https://agmarknet.gov.in/).

Fig. 1: Location of Azadpur market in Delhi,

2.2 Deep learning techniques used for forecasting price series

2.2.1 Convolutional Neural Network (CNN)

A convolutional neural network (CNN) is a specialized type of artificial neural network known for its success in identifying visual patterns. Typically, a CNN architecture includes three primary types of layers: convolutional layers, sub-sampling layers, and fully connected layers. In a standard configuration, multiple convolutional and sub-sampling layers are stacked together, followed by several fully connected layers (see Fig. 2). Convolutional layers work by receiving inputs from neighboring nodes in the previous layer, like the cells in the visual cortex of animals. These layers utilize shared local weights, which not only conserves memory but also enhances classification performance. The sub-sampling layers, which perform non-linear down-sampling, reduce data dimensionality. This reduction decreases local sensitivity and computational complexity, allowing the network to learn features and patterns more effectively(Sánchez-Reolid et al. 2022). Finally, the fully connected layers, akin to those in standard neural networks, conduct comprehensive matrix computations with all activations and nodes. Once the convolutional and sub-sampling layers have extracted features, the fully connected layers handle the reasoning and generate the model's output. CNNs are trained by optimizing the model to minimize the discrepancy between actual and target output values through backpropagation. The integration of convolutional, sub-sampling, and fully connected layers equips CNNs to proficiently recognize patterns and features in visual data, making them highly effective for various computer vision tasks.

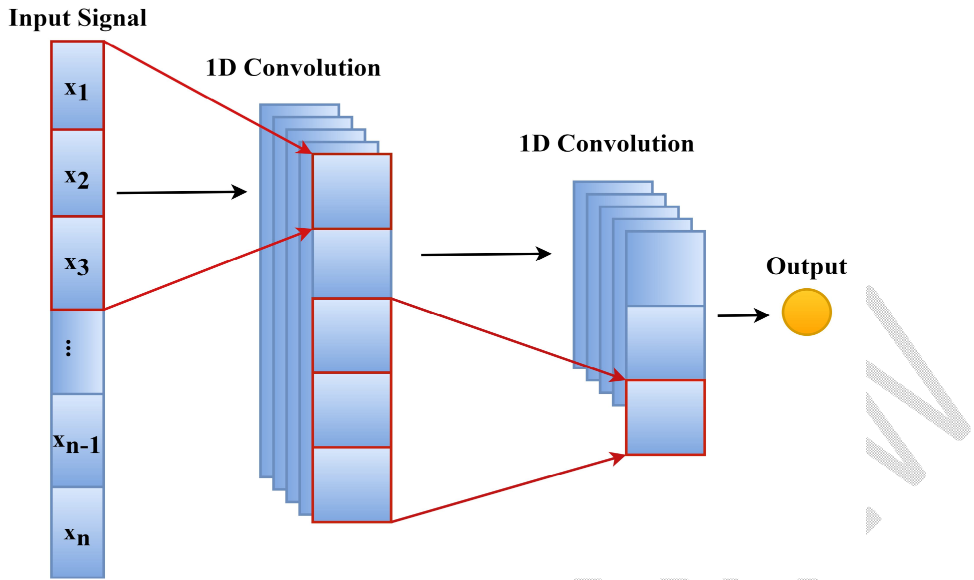


Fig. 2: One-dimension convolutional neural network (1D-CNN) architecture.

2.2.2 Long Short-Term Memory (LSTM)

In 1997, Hochreiter and Schmidhuber addressed a significant limitation in traditional Recurrent Neural Networks (RNNs)—their inability to maintain crucial historical information over long sequences. To overcome this, they introduced the Long Short-Term Memory (LSTM) model, which integrates specialized gate mechanisms into the RNN architecture. LSTMs employ three key gate structures: the forget gate, the input gate, and the output gate, all implemented as sigmoid layers (shown in Fig. 3)(Marino et al. 2016). These gates receive inputs from both the previous network output $(h_{t-1})$and the current input (x_t), and are designed to manage the retention or deletion of information from the previous cell state (C_{t-1}).

The forget gate determines the relevance of previously processed information, with its output $f_t$deciding whether to retain or discard this information. A value of 0 indicates complete discarding, while a value of 1 signifies full retention. The forget gate's output is calculated as:

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

Similarly, the input gate decides which values need updating by processing the previous output $(h_{t-1})$and current input (x_t)through a weight matrix (W_i), a sigmoid function (σ), and a bias term (b_i), resulting in a new candidate value for the current cell state (C_t):

$$i_t = \sigma(W_i . [h_{t-1}, x_t] + b_i) \qquad \dots [2]$$

The updated cell state value $(\widehat{C_t})$ is computed by applying the hyperbolic tangent $(tanh)$function to the weighted input and hidden node, yielding a value between -1 and +1:

$$\widehat{C_t} = tanh(W_c . [h_{t-1}, x_t] + b_c) \qquad \dots [3]$$

The final cell state (C_t) is updated by combining the forget gate's output $(f_t)$and the previous cell state (C_{t-1}), along with the input gate's output (i_t) and the new candidate state $(\widehat{C_t})$:

$$C_t = f_t * C_{t-1} + i_t * \widehat{C_t} \qquad \dots [4]$$

The output gate then determines which parts of the cell state should influence the current output. The output $(o_t)$is calculated using a sigmoid function, and the final output (h_t) is derived by applying the hyperbolic tangent function to the current cell state and multiplying it by the output gate value:

$$o_t = \sigma(W_o . [h_{t-1}, x_t] + b_o) \qquad \dots [5]$$

$$h_t = o_t * \tanh(C_t) \qquad \dots [6]$$

In the fully connected layer of the LSTM model, the Rectified Linear Unit (ReLU) activation function is employed, with the mean square error (MSE) used as the loss function for performance optimizationBakir et al. (2018).

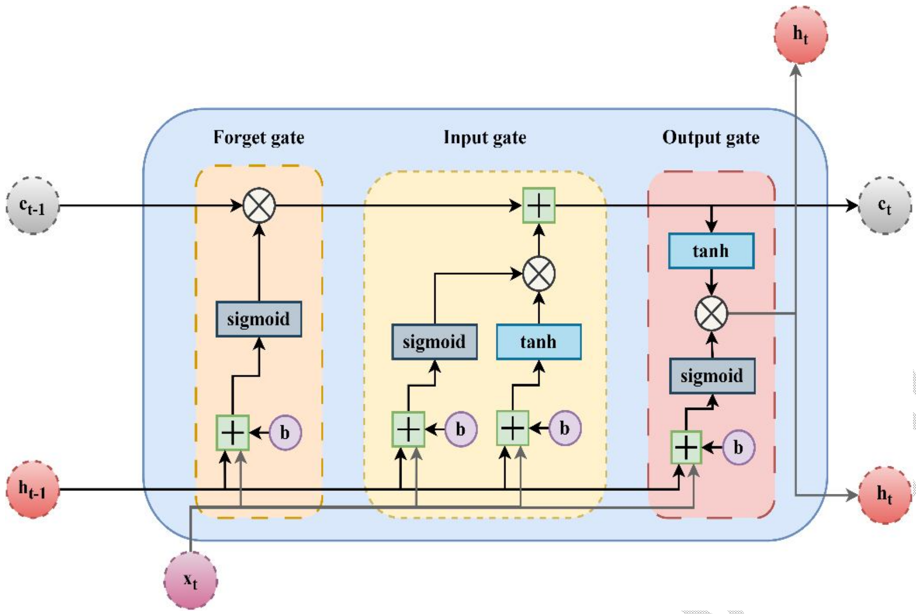


Fig. 3: LSTM architecture

2.2.3 Gated Recurrent Unit (GRU)

GRU layers were introduced in 2014 as a streamlined and more efficient alternative to LSTM layersChung Junyoung et al. (2014).GRUs streamline the architecture by combining the input and forget gates into a single update gate, and merging the hidden and cell states, resulting in fewer parameters compared to LSTM layers(Fig. 4). This simplification makes GRUs more efficient and cost-effective to process. GRUs are designed to prioritize recent events, which are typically more relevant for predicting future outcomes than older information. By efficiently retaining recent information, GRUs are better equipped to perform the current task. The reset gate in GRUs, composed of the hidden and cell states, determines how much past information should be forgotten, while the update gate retains useful information for predicting the presentNayak et al. (2024). The update gate determines the extent to which the GRU unit or cell will be updated and is given by:

$$Z_t(Update\ Gate) = \sigma(W_z.[h_{t-1}, x_t] + b_z) \quad \text{... [7]}$$

Similarly, the reset gate's operation is defined as:

$$R_t(Reset\ Gate) = \sigma(W_R.[h_{t-1}, x_t] + b_R) \quad \text{... [8]}$$

The new candidate state in GRUs is computed by applying the hyperbolic tangent $(tanh)$function to the reset gate, as shown in the equation:

$$\widehat{H}_t = tanh(W.[R_{t*}(h_{t-1}, x_t)] + b_z) \quad \text{... [9]}$$

This function helps the GRU control how much new information should be added to the current state, depending on what the reset gate decides to forget. The output of the tanh function, ranging between -1 and 1, allows the GRU to adaptively manage the flow of relevant information. The final hidden state is determined by the interaction between the previous hidden state and the new candidate state, modulated by the update gate:

$$h_t = (1 - Z_t) * h_{t-1} + Z_t * h_t) \quad \text{... [10]}$$

Here, the update and reset gates, controlled by the sigmoid activation function, manage the recurrent connections and inputs. Weight matrices (W_Z, W_Z and W) and bias terms (b_Z and b_R) regulate the input values in these gates. The final hidden state is a combination of the previous hidden state and the new candidate state, adjusted by the update gate.

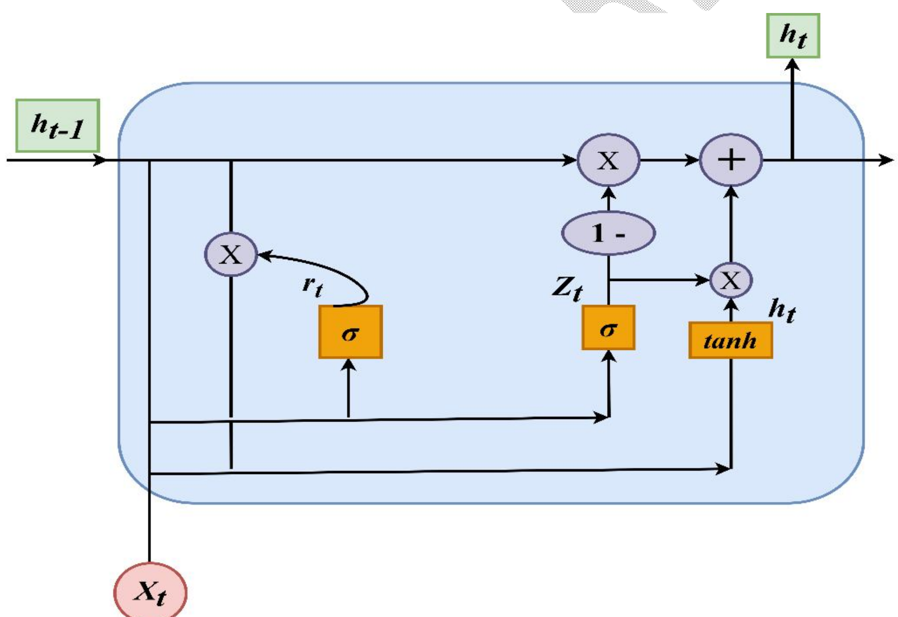


Fig. 4: GRU architecture

2.2.4 Neural Basis Expansion Analysis for Interpretable Time Series Forecasting (N-BEATS)

N-BEATS employs basis expansion as a key technique to enhance data interpretation and prediction accuracy(Oreshkin et al. 2021). This method transforms the original data into a higher-dimensional space by generating new features that capture non-linear relationships within the dataset. Unlike traditional methods where the basis expansion technique is manually selected, N-BEATS utilizes a neural network to automatically identify and optimize the most effective data augmentation strategy during training. This adaptive approach allows the model to tailor its basis expansion to the unique characteristics of the dataset, leading to improved interpretability and predictive performance.

2.2.4.1 The Architecture of N-BEATS

In tackling the univariate point forecasting problem in discrete time, the goal is to predict a future value vector $y \in R^H = [y_{T+1}, y_{T+2}, \dots, y_{T+H}]$ over a forecast horizon H (Figure 5). This prediction is derived from a historical time series $[y_1, \dots, y_T] \in R^T$, using a lookback window of length $t \leq T$, represented by $x \in R^t = [y_{T-t+1}, \dots, y_T]$, ending with the last observed value y_T. The predicted values are denoted as $\hat{y}$.

The design principles of N-BEATS are founded on three core tenets: constructing a simple, generic, yet expressive deep learning architecture; avoiding dependence on time-series-specific components like trend or seasonality; and ensuring the model's extendibility to enhance interpretability. The model takes the lookback period as input, with the forecast period containing actual values for evaluating the predictions. The input sequence length is typically a multiple of the forecast length, ranging from $2H$ to $7H$ for a forecast horizon H.The architecture is organized into layered stacks, each composed of multiple blocks, as illustrated in the left blue rectangle. Each block consists of four fully connected layers, producing two outputs: a forecast and a backcast. The forecast predicts future values, while the backcast allows for immediate comparison with the input sequence, aiding in assessing the model's fit. At the block level, the network calculates expansion coefficients $(\theta)$and performs basis expansion (g), which contribute to the model's adaptability and predictive accuracy (Nayak et al. 2024a).

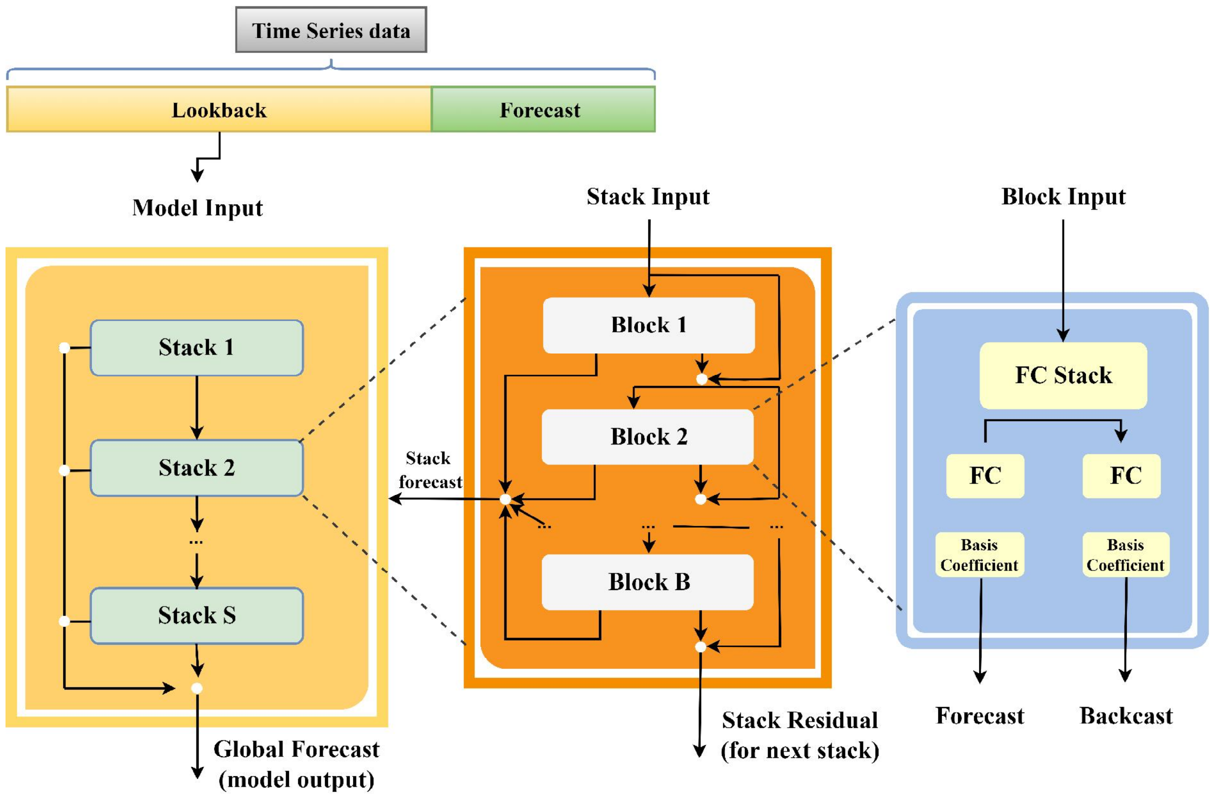


Fig. 5: The architecture of N-BEATS

In this architecture, the actual input sequence is only provided to the first block. The subsequent blocks receive the residuals generated by the previous block. Consequently, only the information not captured by the first block is passed on to the next block. This sequential processing ensures that each block attempts to capture the information missed by the previous one. The basic building block, which has a fork architecture, is depicted in Fig. 5. The l^{th}block accepts its input $x_l$and outputs two vectors, $\hat{x}_l$ and $\hat{y}_l$. For the first block, x_l corresponds to the overall model input a history lookback window of a specific length ending with the last observed value. The input window length is a multiple of the forecast horizon H, typically ranging from $2H$ to $7H$. For subsequent blocks, x_l consists of the residual outputs from the previous blocks. Each block generates two outputs: $\hat{y}_l$, the block's forward forecast of length H; and $\hat{x}_l$, the block's best estimate of x_l, referred to as the 'backcast,' within the constraints of the block's functional space.

Internally, the basic building block has two components. The first part is a fully connected network that produces the forward θ_l^f and the backward θ_l^b expansion coefficients. The second part includes the backward $g_l^b$and the forward g_l^f basis layers, which accept the respective expansion coefficients θ_l^f and θ_l^b, project them onto a set of basis functions, and generate the backcast$\hat{x}_l$ and the forecast outputs $\hat{y}_l$.

The operation of the first part of the l^{th}block is described by the following equations:

$$h_{l,1} = FC_{l,1}(x_l), \qquad h_{l,2} = FC_{l,2}\left(h_{l,1}\right), h_{l,3} = FC_{l,3}\left(h_{l,2}\right), \qquad h_{l,4} = FC_{l,4}\left(h_{l,3}\right)$$

$$\theta_l^b = Linear_l^b\left(h_{l,4}\right), \qquad \theta_l^f = Linear_l^b\left(h_{l,4}\right)$$

where the Linear layer is a simple linear projection, i.e., $\theta_l^f = W_l^f h_{l,4}$. The FC layer is a standard fully connected layer with ReLU non-linearity. This part of the architecture aims to predict the forward expansion coefficients θ_l^f to optimize the accuracy of the partial forecast $\hat{y}_l$ by appropriately combining the basis vectors from g_l^f. Additionally, it predicts the backward expansion coefficients θ_l^b, which are used by g_l^b to produce an estimate of x_l, helping the downstream blocks by removing unhelpful components from their input for forecasting.

The second part of the network maps the expansion coefficients θ_l^f and θ_l^b to outputs via basis layers:

$$\hat{y}_l = g_l^f\left(\theta_l^f\right), \hat{x}_l = g_l^b\left(\theta_l^b\right)$$

This operation is further defined by:

$$\hat{y}_l = \sum_{i=1}^{dim\left(\theta_l^f\right)} \theta_{l,i}^f v_i^f, \qquad \hat{x}_l = \sum_{i=1}^{dim\left(\theta_l^b\right)} \theta_{l,i}^b v_i^b$$

where v_i^f and v_i^b are the forecast and backcast basis vectors, respectively, and $\theta_{l,i}^f$ is the i^{th} element of θ_l^f.

The classical residual network architecture adds the input of a stack of layers to its output before passing the result to the next stack. In contrast, a novel hierarchical doubly residual topology is proposed. This architecture introduces two residual branches: one over the backcast prediction of each layer and the other over the forecast branch. Its operation is described by the following equations:

$$x_l = x_{l-1} - \hat{x}_{l-1}, \qquad \hat{y} = \sum_l \hat{y}_l$$

For the first block, the input is the model-level input x, so $x_1 \equiv x$. In all other blocks, the backcast residual branch $x_l$performs a sequential analysis of the input signal. The previous block removes the signal portion $\hat{x}_{l-1}$ that it can approximate well, simplifying the forecasting task for downstream blocks. This structure also facilitates smoother gradient backpropagation. More importantly, each block outputs a partial forecast $\hat{y}$, which is aggregated first at the stack level and then at the overall network level, providing a hierarchical decomposition. The final forecast $\hat{y}$is the sum of all partial forecasts. In a generic model context, allowing arbitrary g_l^b and g_l^f for each layer makes the network more transparent to gradient flows. In a special case with structured g_l^b and g_l^f shared across a stack, this enables interpretability through the aggregation of meaningful partial forecasts. The residual connections in the network help capture information missed by previous blocks. Combining different blocks forms a stack, which produces a partial prediction. The final forecast is obtained by combining all these partial predictions.

2.3Data Pre-processing and Normalization

To ensure effective fitting of deep learning models and unbiased extrapolation, it is essential to preprocess and normalize the data series. Normalization rescales the values of both series to a range between 0 and 1 while preserving their inherent shape. This standardized approach enhances model training robustness and improves the model's ability to generalize patterns from the data. The normalization process is defined by the following equation:

$$X'_t = \frac{X_t - X_{min}}{X_{max} - X_{min}}$$

where X_{min}, $X_{max}$and $X_t$are the minimum, maximum and observation at time t, respectively and X'_t is the rescaled value.

2.4Hyperparameter tuning:

Here, we provide a comprehensive overview of the hyperparameters used in developing the various forecasting models. The fine-tuning process utilized the random search method for hyperparameter optimization. Specifically, random search was employed to optimize hyperparameters for the DL models. Training accuracy was evaluated across a range of randomly selected hyperparameter combinations, with the final configuration selected based on the highest achieved accuracy (as detailed in Table 1). In our study, we implemented four different algorithms: CNN, LSTM, GRU, and N-BEATS to forecast the potato price series.

Table 1: The hyperparameters and their values of different models used for comparison

Models	**Hyperparameters**	**Values**
N-BEATS	Fully connected layers	4
	Lookback	7
	Horizon	1
	Stacks	30
	Neurons per layer	512
	Epochs	500
	Loss function	MAE
	Optimizer	Adam
CNN	Filters	128
	Kernel size	5
	Batch size	128
	Epochs	100
	Loss function	MAE
	Optimizer	Adam
LSTM	Inputs	128
	Activation function	ReLU
	Batch size	128
	Epochs	100
	Loss function	MAE
	Optimizer	Adam
GRU	Inputs	128
	Activation function	ReLU
	Batch size	128
	Epochs	100

	Loss function	MAE
	Optimizer	Adam

2.5Performance measure:

a) Mean Absolute Percentage Error (MAPE)

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{y_i - \hat{y}}{y_i}\right|$$

b) Mean Absolute Error (MAE)

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}|$$

c) Root Mean Squared Error (RMSE)

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

where,N is the number of observations in the dataset, $y_i$is the true values of the variable being predicted and $\hat{y}$ is the predicted values of the variable.

3. Results and Discussion

3.1 Descriptive statistics

The descriptive statistics of the experimental datasets used in this study is displayed in Table 2, while Fig. 6 illustrate the actual time series plot of the Potato price series. It is evident from the graphs that data exhibit non-stationarity. This observation is further validated through statistical tests.

Table 2: Descriptive statistics of Potato price series data.

Descriptive Statistics	**Price (Rs. /Quintal)**
Minimum	335

Mean	1132.55
Maximum	4600
Standard Deviation	569.47
Coefficient of Variation (%)	50.28
Skewness	1.53
Kurtosis	3.16
Jarque-Bera test	1280.78 (<0.0001)
Shapiro-Wilk's test	0.87 (<0.0001)

The value in the parentheses indicates *p*-value

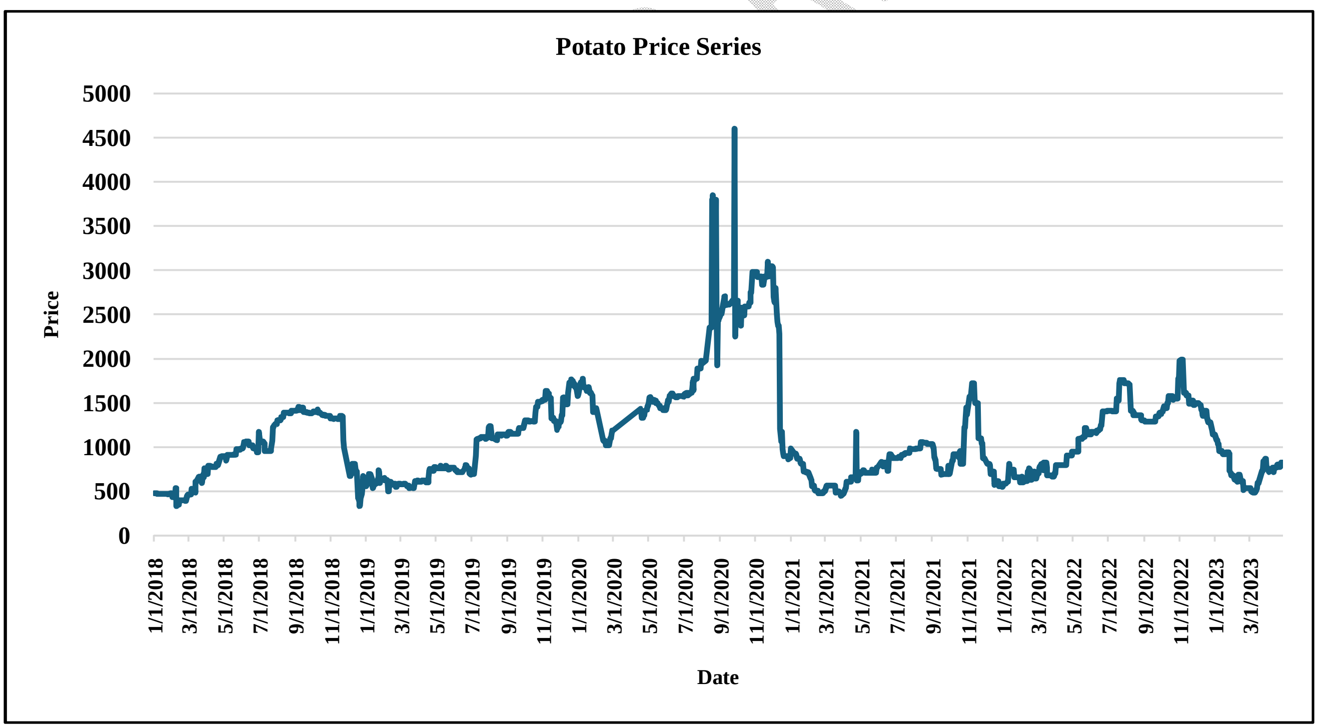


Fig. 6: Time series plot of Potato price series

The potato price series data, with a price range between Rs. 335 and Rs. 4600 per quintal, exhibits significant volatility, as indicated by the standard deviation shown in Table 2. The data displays positive skewness and leptokurtic characteristics, suggesting a non-normal distribution. This non-normality is confirmed by the Shapiro-Wilk and Jarque-Bera tests. The dataset contains a total of 1398 observations, which are divided into training (80%) and testing (20%) sets. The training set, consisting of 1198 observations, is used for model development, while the testing set, with 300 observations, is used for model validation and post-sample prediction. This approach ensures a thorough analysis of potato price dynamics by incorporating both training and testing subsets.

3.2 Test for Stationarity

Stationarity is a critical consideration in forecasting models and is evaluated in this study using the Augmented Dickey-Fuller (ADF) test(Avinash et al. 2024). The null hypothesis of the ADF test posits that the series is non-stationary or contains a unit root. The results, detailed in Table 3, confirm that the series is stationary.

Table 3: ADF test result of potato price series

Data	**Augmented Dickey-Fuller**		**Remarks**
	Statistic	***p* - value**	
Potato	-2.63	0.08	Stationary

3.3 Performance Evaluation

Performance evaluation of each model on the potato price series dataset utilized metrics including MAPE, MAE, and RMSE, as reported in Table 4. The N-BEATS model consistently demonstrated superior performance, achieving the lowest MAPE of 2.09, MAE of 22.48, and RMSE of 49.68. This highlights the N-BEATS model's exceptional ability to capture the underlying patterns and dynamics of the potato price series, leading to more accurate forecasts.

Table 4: Results obtained by different models on the testing dataset for potato price series

Models	**MAPE (%)**	**MAE**	**RMSE**
CNN	2.15	23.44	49.88

LSTM	2.82	31.35	55.98
GRU	2.62	28.94	53.26
N-BEATS	**2.09**	**22.48**	**49.68**

Table 5: Diebold–Mariano test results of potato price series

Models	CNN	LSTM	GRU	N-BEATS
CNN	-	-70.96 (0.98)	-61.34 (0.92)	57.71 (0.99)
LSTM	-70.96 (0.98)	-	-4.95 (0.94)	3.99 (0.98)
GRU	-61.34 (0.92	-4.95 (0.94)	-	0.31 (0.92)
N-BEATS	57.71 (0.99)	3.99 (0.98)	0.31 (0.92	-

Values in the parentheses indicates *p*-value

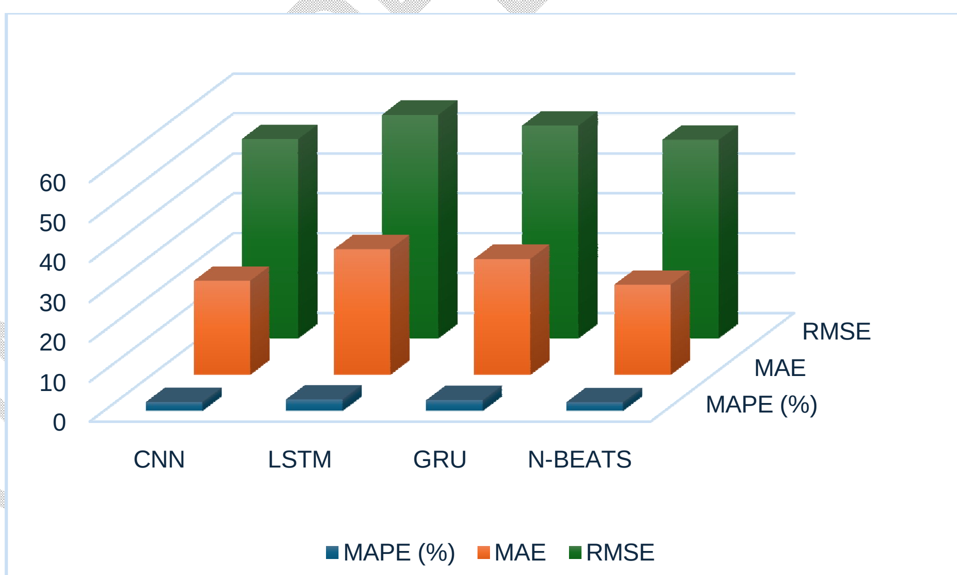


Fig. 7:3D Bar diagram of evaluation criteria of Potato price series on the testing dataset

Traditional models such as ARIMA struggle with capturing nonlinear patterns, while parametric nonlinear models like GARCH face limitations due to their rigid assumptions. To address these challenges, researchers have increasingly turned to machine learning (ML) methods. However, for large datasets, ML's predictive accuracy can diminish due to the need for manual feature extraction. Consequently, deep learning (DL) architectures, including CNN, LSTM, and GRU, have become popular for modeling price data. This study presents the N-BEATS algorithm for forecasting daily potato prices from the Azadpur market, Delhi. Comparative analysis shows that N-BEATS outperforms CNN, LSTM, and GRU in predictive accuracy, achieving the lowest values for key performance metrics such as MAPE, MAE, and RMSE, which signifies its effectiveness in capturing trends and patterns in time series data (Figure 7).

N-BEATS' success is attributed to its unique architecture, which employs stacked blocks of fully connected layers for both backcasting and forecasting. The inclusion of residual links and double residual stacking enhances the model's learning capacity and prediction refinement. N-BEATS demonstrates remarkable adaptability and flexibility, efficiently handling diverse time series patterns and various data types. The model's performance is further validated by the Diebold-Mariano (DM) test, which confirms its superior forecasting accuracy compared to other benchmark models (Table 5). This highlights N-BEATS' ability to capture complex patterns and its suitability for different forecasting tasks, making it a promising tool for time series analysis. Although the current N-BEATS architecture does not incorporate spatiotemporal modeling, future adaptations designed for such data could offer even greater utility, especially in contexts involving spatial-temporal dynamics. Overall, the study establishes N-BEATS as a highly effective and superior approach for time series forecasting, with significant implications for practitioners and researchers in the field.

4. Conclusion

The study highlights the crucial role of accurate forecasting in agricultural commodity prices due to its substantial impact on India's economy. Traditional models like ARIMA are limited in their ability to capture nonlinear patterns, prompting researchers to explore machine learning (ML) methodologies. However, the manual feature extraction required for ML, especially with large datasets, presents challenges. The advent of deep learning (DL) architectures, such as CNN, LSTM, GRU, and particularly N-BEATS, has transformed price data modeling. N-BEATS stands out with its unique architecture, which includes stacked blocks and innovative features like residual links and double residual stacking, demonstrating superior performance in capturing complex time series patterns. Its adaptability and flexibility are evidenced by its strong performance in comparative analyses and the Diebold-Mariano test. Although the current N-BEATS architecture does not incorporate spatiotemporal modeling, future adaptations designed for such data may offer even greater potential. Overall, N-BEATS proves to be a powerful and advanced tool for time series forecasting, with important implications for both practitioners and researchers in the field.

Availability of data and material

Data will be available based on the request with the corresponding author.

Code availability

Code will be available on request to the corresponding author.

Declarations

Ethics approval, consent to participate, and consent for publication

The manuscript does not report on or involve the use of any animal or human data and "not applicable" in this section.

Reference

Avinash, G., V. Ramasubramanian, M. Ray, R. K. Paul, S. Godara, G. H. H. Nayak, R. R. Kumar, B. Manjunatha, S. Dahiya, and M. A. Iquebal. 2024. Hidden Markov guided Deep Learning models for forecasting highly volatile agricultural commodity prices. Applied Soft Computing 158 (June 2024):111557.

Bakir, H., G. Chniti, and H. Zaher. 2018. E-Commerce Price Forecasting Using LSTM Neural Networks. International Journal of Machine Learning and Computing 8(2):169–174.

Bulatov, A. 2020. Forecasting bitcoin prices using n-beats deep learning architecture (2020).

Chung Junyoung, Gulcehre Caglar, Cho KyungHyun, and Bengio Yoshua. 2014. Evaluation of datasets. In: NIPS 2014 Workshop on Deep Learning (2014).

Conejo, A. J., M. A. Plazas, R. Espinola, and A. B. Molina. 2005. Day-Ahead Electricity Price Forecasting Using the Wavelet Transform and ARIMA Models. IEEE Transactions on Power Systems 20(2):1035–1042.

Durairaj, Dr. M., and B. H. K. Mohan. 2022. A convolutional neural network based approach to financial time series prediction. Neural Computing and Applications 34(16):13319–13337.

Hewamalage, H., C. Bergmeir, and K. Bandara. 2021. Recurrent neural networks for time series forecasting: Current status and future directions. International Journal of Forecasting 37(1):388–427.

Jaiswal, R., G. K. Jha, R. R. Kumar, and K. Choudhary. 2022. Deep long short-term memory based model for agricultural price forecasting. Neural Computing and Applications 34(6):4661–4676.

Jamuna, C. J., C. Patil, and R. A. Kumar. 2021. Forecasting the Price of Potato Using Time Series ARIMA Model. In Proceedings of International Conference on Communication and Computational Technologies: ICCCT 2021, 493–518. Springer.

Jossou, T. R., Z. Tahori, G. Houdji, D. Medenou, A. Lasfar, F. Sanya, M. H. Ahouandjinou, S. M. Pagliara, M. S. Haleem, and A. Et-Tahir. 2022. N-Beats as an EHG Signal Forecasting Method for Labour Prediction in Full Term Pregnancy. Electronics 11(22):3739.

LIU, H. 2024. Time Series Predictive Control in Robotics. EDP Sciences.

Marino, D. L., K. Amarasinghe, and M. Manic. 2016. Building energy load forecasting using Deep Neural Networks. In IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, 7046–7051. IEEE.

Mehtab, S., and J. Sen. 2020. A time series analysis-based stock price prediction using machine learning and deep learning models. International Journal of Business Forecasting and Marketing Intelligence 6(4):272.

Nayak, G. H. H., M. W. Alam, K. N. Singh, G. Avinash, R. R. Kumar, M. Ray, and C. K. Deb. 2024a. Exogenous variable driven deep learning models for improved price forecasting of TOP crops in India. Scientific Reports 14(1):17203.

Nayak, G. H. H., W. Alam, K. N. Singh, G. Avinash, M. Ray, and R. R. Kumar. 2024b. Modelling monthly rainfall of India through transformer-based deep learning architecture. Modeling Earth Systems and Environment 10(3):3119–3136.

Oreshkin, B. N., G. Dudek, P. Pełka, and E. Turkina. 2021. N-BEATS neural network for mid-term electricity load forecasting. Applied Energy 293 (July 2021):116918.

Patarwal, P., A. Dagar, R. Bala, and R. Singh. 2018. Financial Time Series Forecasting Using Deep Learning Network. In 23–33.

Paul, R. K., Md. Yeasin, P. Kumar, P. Kumar, M. Balasubramanian, H. S. Roy, A. K. Paul, and A. Gupta. 2022. Machine learning techniques for forecasting agricultural prices: A case of brinjal in Odisha, India. PLOS ONE 17(7):e0270553.

Racocha, K. 2020. Impact of Drought on Commodity Market Forecasting (2020).

Sahu, P. K., M. Das, B. Sarkar, A. VS, S. Dey, L. Narasimhaiah, P. Mishra, R. K. Tiwari, and Y. S. Raghav. 2024. Potato production in India: a critical appraisal on sustainability, forecasting, price and export behaviour. Potato Research (2024):1–37.

Sánchez-Reolid, R., F. López de la Rosa, M. T. López, and A. Fernández-Caballero. 2022. One-dimensional convolutional neural networks for low/high arousal classification from electrodermal activity. Biomedical Signal Processing and Control 71 (January 2022):103203.

Sbrana, A., and P. A. Lima de Castro. 2023. N-BEATS Perceiver: A Novel Approach for Robust Cryptocurrency Portfolio Forecasting. Computational Economics (September 22, 2023).

Tan, Z., J. Zhang, J. Wang, and J. Xu. 2010. Day-ahead electricity price forecasting using wavelet transform combined with ARIMA and GARCH models. Applied Energy 87(11):3606–3610.

Tripathi, B., and R. K. Sharma. 2023. Modeling Bitcoin Prices using Signal Processing Methods, Bayesian Optimization, and Deep Neural Networks. Computational Economics 62(4):1919–1945.

Yadav, S., A. M. G. Al Khatib, B. M. Alshaib, S. Ranjan, B. Kumari, N. A. Alkader, P. Mishra, and P. Kapoor. 2024. Decoding potato power: a global forecast of production with machine learning and state-of-the-art techniques. Potato Research (2024):1–22.