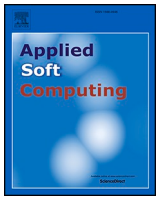
Applied Soft Computing Journal 149 (2023) 110939

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc



An ARIMA-LSTM model for predicting volatile agricultural price series with random forest technique

Soumik Ray [a,1,2], Achal Lama [b,*,1,3], Pradeep Mishra [c,4], Tufleuddin Biswas [d], Soumitra Sankar Das [e], Bishal Gurung [f]

[a] Department of Agricultural Economics and Statistics, Centurion University of Technology and Management, Paralakhemundi, Odisha, India
[b] Division of Forecasting and Agricultural Systems Modelling, ICAR-Indian Agricultural Statistics Research Institute, New Delhi, India
[c] College of Agriculture, Rewa, Jawaharlal Nehru Krishi Vishwa Vidyalaya, M.P., India
[d] Department of Agricultural Economics and Statistics, Centurion University of Technology and Management, Paralakhemundi, Odisha, India
[e] Faculty of Management and Commerce, The ICFAI University, Agartala, Tripura, India
[f] Department of Statistics, North-Eastern Hill University, Shillong, India

HIGHLIGHTS

- A novel hybrid of ARIMA-LSTM model using random forest algorithm for selecting suitable input lags for LSTM.
- The proposed model has the capability to deal with volatile data sets, with inherent skewedness.
- The model has been applied to 3 agricultural price series and superior results have been documented.
- This article establishes the superiority of the hybrid machine learning models over the statistical models.

ARTICLE INFO

Keywords:
Random forest
ARIMA
GARCH
LSTM
Hybrid
Volatile

ABSTRACT

Machine learning mechanism is establishing itself as a promising area for modelling and forecasting complex time series over conventional statistical models. In this article, focus has been made on presenting a machine learning algorithm with special attention to deep learning model in form of a potential alternative to statistical models such as Autoregressive Integrated Moving Average (ARIMA) and ARIMA-Generalised Autoregressive Conditional Heteroscedasticity (GARCH) models. Further, an improved hybrid ARIMA-Long Short-Term Memory (LSTM) model based on the random forest lag selection criterion has been introduced. ARIMA model has been used to estimate the mean effect and the GARCH model is employed with the residuals obtained from the ARIMA model to estimate the volatile behaviour of the series. ARIMA-GARCH models act as superior statistical models over ARIMA models based on the lowest AIC and BIC values. LSTM model is employed on all normalised training data series. After which we built a comparison scenario independently between ARIMA, ARIMA-GARCH, LSTM and ARIMA-LSTM models on forecasting accuracy in terms of the lowest RMSE, MAPE and MASE values. The proposed random forest-based ARIMA-LSTM model proved its superiority over the conventional statistical models with an improvement to the tune of 8–25% for RMSE, 2–28% for MAPE and 2–29% for MASE. The proposed hybrid model has been successfully applied to volatile monthly price indices of pulses namely gram, moong and urad. This piece of work will enrich the literature on machine learning and further intrigue researchers to apply it to various other volatile data sets.

Code metadata.

Permanent link to reproducible Capsule:https://doi.org/10.

* Corresponding author.
E-mail address: achal.lama@icar.gov.in (A. Lama).
[1] They have contributed equally and to be treated as joint first authors.
[2] ORCID ID: 0000–0003-2781–8913
[3] ORCID ID: 0000–0002-5376–3760
[4] ORCID ID: 0000–0003-4430–886X

https://doi.org/10.1016/j.asoc.2023.110939
Received 3 September 2022; Received in revised form 4 October 2023; Accepted 12 October 2023
Available online 20 October 2023

24433/CO.5856403.v1.

1. Introduction

Price forecasting plays a pivotal role in the efficient management of agricultural commodities; assists in the decision making process of different stakeholders and policymakers in the marketing chain *i.e.* farmers to consumers. A precise model for forecasting the price of agricultural commodities also contributes substantially to managing food security. In general, agricultural price data are considered to be a time series as the information is collected over time. Conventional Box-Jenkins' ARIMA model is used for modelling and forecasting a time series unless the series is volatile. It is well accepted that agricultural commodity price series are inconsistent and volatile because of various extraneous factors, like market fluctuation, demand-supply gap, seasonal weather conditions, low farm production, *etc.* Hence, most of the agricultural price series follow high volatility of unstable nature [1]. The volatile series can be assumed when a few error terms are larger than others and it gives a unique characteristic to the series, *i.e.*, heteroscedasticity. The volatility of time series can be modelled by the autoregressive conditional heteroscedasticity (ARCH) model [2]. The ARCH model is used to estimate the mean and conditional variance forecast from the recent past information. Bollerslev [3] generalized the ARCH model (GARCH), with maximum likelihood estimation making it parsimonious.

Owning to volatility characteristics of the most financial and economic time series, GARCH family of models are widely used to model and forecast the variance, *i.e.* volatility instead of the ARIMA model. Epaphra [4] employed the ARCH and GARCH models to capture the volatility clustering in the exchange rate series of Tanzania. Cheikh [5] examined the volatile nature of cryptocurrency markets by using the smooth transition GARCH (ST-GARCH) model. Also, agricultural commodity price series can be modelled with a hybrid ARIMA-GARCH model, a combination of the linear ARIMA model and the conditional variance of the GARCH model [6]. Bhardwaj [7] employed ARIMA and GARCH models to estimate the daily price of the gram series. The author examined that GARCH model is the most suitable for capturing volatility of time series. Yaun [8] modelled the co-movement amount of different agricultural commodity markets by using the copula-GARCH model. Shiferaw [9] used the Bayesian DCC-MGARCH model to capture the correlation dynamics between agricultural commodities and energy prices. Lin [10] estimated crude oil price forecast with novel hybrid long memory GARCH-M model. A comprehensive review of time series model development based on agricultural commodity series has been given by Majit and Mir [11].

In recent years, the machine learning technique is introduced for developing learning models and is used to forecast time series with deep learning algorithms. Deep learning algorithms such as Recurrent Neural Network (RNN) [12] and Long Short-Term Memory (LSTM) [13] have performed satisfactorily in many disciplines of science, especially in the financial sectors. Deep learning mechanism is used to visualize the pattern and structure of the series, mainly complexity and non-linearity by extracting hidden layers from the target network in time series forecasting. Generally, LSTM has been used for time series prediction, especially in economics and financial series [14]. Kurumatani [15] implemented RNN mechanism for forecasting agricultural product prices. He used two methods based on RNN, *i.e.* time alignment of time point forecast (TATP) and direct future time series forecast (DFTS) for forecasting. Om [16] modelled email traffic workloads using RNN and LSTM models and considered traffic as a time series function. Yoo and Oh [17] used a seasonal LSTM model to predict sales of agricultural products for stabilizing the demand-supply gap. Thus, studies pertaining to adequately comparing the accuracy and precision between conventional statistical forecasting models and deep learning algorithms seem to be a new promising area in this era of fast computation and complex data sets [18,19].

The other aspect in the domain of machine learning, which is fast gaining popularity is the hybridization of the algorithms with statistical or some other machine learning techniques. The concept of hybrid models is an age-old practice in time series literature. It allows the researchers to take advantage of each model and efficiently forecast the series in hand [20]. With similar intentions, we have hybridized the popular ARIMA and LSTM model, with input lags for LSTM have been obtained using random forest algorithm and thus, making this ARIMA-LSTM hybrid framework novel and efficient. The importance of lag selection and the appropriateness of random forest for doing so has been discussed subsequently.

As all the machine learning algorithms are highly data dependent, it is of utmost importance that appropriate quality data is fed to the system. In the context of time series analysis, for the data to be adequate, one needs to carefully determine the lag structure of it. Over the years several researchers have explored the possibility of either improving the existing criteria or proposing an alternative. Goutte [21] investigated relevant lag space *i.e.* input information for time series modelling using the Naïve generalization (NG) method. Scargle [22] developed Bayesian Blocks algorithm for estimating time series lags and structure of measurement which is a weighted integral over a finite range of the independent variable. Han [23] implemented dynamic panel lag order estimation using modified BIC for lag length selection. Cortez [24] investigated variable selection (lag selection) based on the backward algorithm by sensitivity analysis using NN and SVM to find the correct input lags for time series forecasting. Surakhi [25] applied the autocorrelation function and a heuristic algorithm with special attention to the LSTM approach for selecting optimal lag length for improving the efficiency of the time series forecasting approach. To this end, we have used the very powerful and widely celebrated random forest algorithm to determine the lag length as input nodes in the LSTM model. Random forest algorithm has been used very efficiently in time series literature in the context of forecasting [26,27]. But, to the best of our knowledge, we could not find any article that uses random forest algorithm for estimation of the lag length as an input to the LSTM model.

When highly volatile data or short data sequences are available, a hybrid approach combining non-stationary parametric models like GARCH with the nonlinear modelling potential of LSTM neural networks is required to demonstrate similar predictive performance in cryptocurrency price forecasting [28]. Li and Becker [29] used a state-of-the-art long short-term memory (LSTM) deep neural networks in conjunction with feature selection algorithms to handle nonlinear and challenging problems, analyse time series data, and deliver results that were surprisingly accurate for predicting electricity prices. Phuruan and Kasemset reported a better finding of hybrid ARIMA and LSTM model in forecasting shallot's price and reported minimum values of RMSE, MAE and MAPE as 10.275 Baht, 8.512 Baht, and 13.618%, respectively [30]. Kulshreshtha [31] reported a much better performance LSTM- ARIMA hybrid than Prophet for capturing the linear and non-linear portions of the time series. Exponentially weighted moving average, deep feedforward neural network (DFN), LSTM, and hybrid DFN models that combine a DFN with one GARCH-type model are examples of hybrid LSTM models with various generalized autoregressive conditional heteroscedasticity (GARCH)-type [32]. Hybrid LSTM-GJR-GARCH model (1,1) showed that use of hyperparameter modification boosts volatility predictions by an average of 40–70% compared to standard LSTM-GJR–GARCH [33] and thus, it was found that the degree of geopolitical risk uncertainty does not fully explain the interconnection of natural resource prices. It was suggested that GARCH family models, the CatBoost algorithm, CNN, and LSTM be used to display the variation in the daily load of natural gas through the traditional GARCH family model and a new gradient boosting method. This will demonstrate that the natural gas load prediction based on GARCH family-CatBoost-CNNLSTM is decreased by an average of 26.555%, 30.892%, and 26.283% in the three relative evaluation indicators of RMSE, MAE, and MAPE [34]. To produce superior copper price

volatility forecasts, [35] presented a novel hybrid GARCH- (LSTM) network with a conventional artificial neural network (ANN). They combined the advantages of these two approaches and found that they were particularly effective in predicting copper price volatility.

We are motivated by the above fact that the both deep learning mechanism and conventional statistical methods can be used for forecasting the volatile agricultural price series. Hence, in this paper, we attempt to compare the modelling and forecasting efficiencies of these two approaches using major pulses price indices of India. Driven strongly by a review of the literature, we selected traditional time series models ARIMA and GARCH and the deep learning model (LSTM) for the present investigation. First, the ARIMA model has been applied to check the mean behaviour of the series. Then, a hybrid ARIMA-GARCH model was introduced for modelling and forecasting volatility with residuals obtained from the ARIMA model. Then, we independently implement the deep learning mechanism with LSTM neural network for predicting the pulses price indices series. Finally, we sought the hybrid framework of the ARIMA-LSTM model based on the lag estimation technique using ACF and random forest. The proposed hybrid ARIMA-LSTM model was found to be superior to other conventional models based on random forest lag estimation. The complete framework of our study is depicted in Fig. 1. The remaining portion of the manuscript is broadly divided into data description, methodology, empirical results and discussion followed by conclusion section.

2. Data description

In this research, the volatile pulses price indices series was used to examine the forecasting ability of different models. Three pulses series *i.e.* Gram, Moong and Urad were considered. Monthly data series were collected from the Economic Adviser, Ministry of Commerce, Government of India, for the period April 1994 to March 2021. The series contained 324 data points, out of which 312 data points were used for the model development and the remaining 12 data points were used for validation purpose.

3. Methodology

3.1. ARIMA model

ARIMA Model is used to build a time series model from univariate time series observation; the model is also widely expressed as Box-Jenkins methodology [36]. The model consists of three terms; autoregressive (AR) term, integrated (I) term and moving average (MA) term. The AR term expressed autocorrelation between past and present observation while MA term denotes autocorrelation structure of residuals (error) and that most univariate time series data follows an increasing and decreasing trend, which is non-stationary; confirming integrated [37]. So the integrated (I) part is expressed by the differencing level of observations to transform a series from non-stationary to stationary [38].

Generally, an ARIMA model is expressed with three parameters *i.e.* (p, d, q); where p is denoted as the AR model, d is expressed as difference and q is denoted as the MA model. If the series is stationary, then the combination of AR and MA model, ARMA model $[X_t]$ is applied. The autoregressive, integrated and moving average term follows greater than or equal to zero of the model if.

$\nabla^d X_t = (1-\beta)^d \varepsilon_t$, where, β is denoted as backshift operator.

Then the model ARIMA (p, d, q) is denoted as

$$\varnothing(\beta)(1-\beta)^d X_t = \theta(\beta)\varepsilon_t \tag{1}$$

where $\varnothing(\beta) = 1 - \varnothing_1\beta - \varnothing_2\beta^2 - .. - \varnothing_p\beta^p$, (Order p is AR term),

$\theta(\beta) = 1 - \theta_1\beta - \theta_2\beta^2 - .. - \theta_q\beta^q$, (Order q is MA term).

and $\varepsilon_t \sim WN(0, \sigma^2)$, the difference $d \geq 0$.

ARIMA (p, d, q) model follows ARMA (p, q) model, when the difference $d = 0$.

3.2. GARCH model

The error series $\{\varepsilon_t\}$, is said to follow ARCH (q) process which can be defined by specifying the conditional distribution of ε_t given the information available up to time $t-1$ denoted by ψ_{t-1}. ARCH (q) model is denoted as

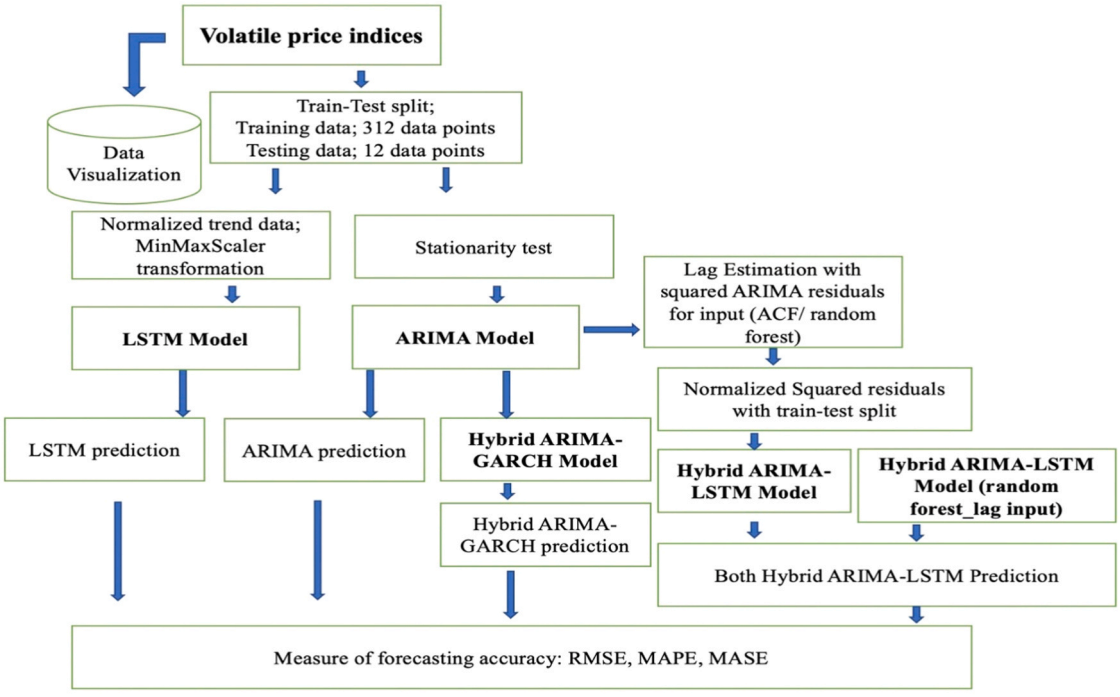


Fig. 1. The employed methodology framework.

$$\varepsilon_t|\psi_{t-1} \sim N(0, h_t) \tag{2}$$

$$h_t = a_0 + \sum_{i=1}^{q} a_i \varepsilon_{t-i}^2 \tag{3}$$

where, $a_0 > 0, a_i \geq 0$ for all i and $\sum_{i=1}^{q} a_i < 1$ are required to be satisfied to ensure nonnegativity and finite unconditional variance of stationary $\{\varepsilon_t\}$ series. Bollerslev [3] and Taylor [39] independently proposed the GARCH model, in which they assumed the conditional variance to be a linear function of its past values and takes the following form:

$$\varepsilon_t = \xi_t h_t^{1/2}$$

$$h_t = a_0 + \sum_{i=1}^{q} a_i \varepsilon_{t-i}^2 + \sum_{j=1}^{p} b_j h_{t-j} \tag{4}$$

where $\xi_t \sim N(0,1)$. A sufficient condition for the conditional variance to be positive is

$$a_0 > 0, a_i \geq 0, i = 1, 2, \ldots, q. b_j \geq 0, j = 1, 2, \ldots, p.$$

The GARCH (p, q) process is weakly stationary if and only if; $\sum_{i=1}^{q} a_i + \sum_{j=1}^{p} b_j < 1$..

The conditional variance defined by (3) has the property that the unconditional autocorrelation function of ε_t^2, if exists, has a slow decay pattern.

3.3. Recurrent Neural Network (RNN)

The Recurrent Neural Network (RNN) is a special neural network that is used to process past data sequences and predict the next time step from that sequence. This machine-learning mechanism can be used in different fields of science activities, such as language processing, voice recognition, data prediction, and signal processing [40]. Like Artificial Neural Network (ANN), RNN also has three layers *i.e.* input layer, hidden layer and output layer. The hidden layer of RNN acts as an internal storage memory to store the information, captured from sequential data processing in the previous stage. A typical RNN diagram structure is represented in Fig. 2.

3.3.1. Long short-term memory model

LSTM is an advanced algorithm of RNN with a special feature to memorise long-term data sequences. In the LSTM technique, the computational unit is associated with two states *i.e.* hidden state (h_t) and memory state (C_t). Also it has three different gates *i.e.* input gate $(\widehat{i}_t)$, forget gate $(\widehat{f}_t)$ and output gate $(\widehat{o}_t)$ to control the flow of data at different time epoch. The main function of these gates is regulating the data to remember or forget during the sequence processing and to overcome the vanishing gradient problem, which is most commonly observed in neural RNN. All three gates in LSTM are executed through an activation function *i.e.* sigmoid function and give the value within the range 0–1. Another activation function, the hyperbolic tangent function is applied in a temporary cell state (S_t), having the range -1 to $+1$. Input gate and cell state combinedly store the amount of information

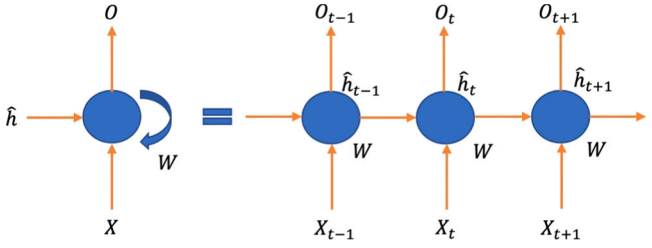


Fig. 2. Standard RNN architecture.

$(\widehat{i}_t * S_t)$, whereas forget gate keeps the amount of memory in the current memory state $(\widehat{f}_t * C_{t-1})$. So, for a sequence variable $(y_1, y_2, \ldots\ldots, y_n)$, where y_t is the feature vector and t indicates the corresponding time steps, all the parameter function is given below.

$$\widehat{i}_t = \sigma(W_{y_i} y_t + W_{h_i} h_{t-1} + B_i) \tag{5}$$

$$\widehat{f}_t = \sigma(W_{y_f} y_t + W_{h_f} h_{t-1} + B_f) \tag{6}$$

$$S_t = \tanh(W_{y_s} y_t + W_{h_s} h_{t-1} + B_s)$$

$$C_t = (\widehat{i}_t * S_t) + (\widehat{f}_t * C_{t-1})$$

$$\widehat{o}_t = \sigma(W_{y_o} y_t + W_{h_o} h_{t-1} + B_o) \tag{7}$$

$$h_t = \widehat{o}_t * \tanh(C_t) \tag{8}$$

where, W and B are the weights and biases of each gate.

3.4. Proposed Model (ARIMA-LSTM)

The proposed model was constructed by a combination of ARIMA and LSTM models. First, the residuals from the best ARIMA model were used to predict volatility using a deep learning LSTM model, say the ARIMA-LSTM model.

The residuals should follow a normal distribution with zero mean and constant variance.

$\varepsilon_t \sim WN(0, \sigma^2)$, the difference $d \geq 0$.

For employing the LSTM model, the residuals normalized with minmax scaling

$$\varepsilon_{scaled} = \frac{e_i - \min(\varepsilon)}{\max(\varepsilon) - \min(\varepsilon)} \sim iidN(0,1).$$

The main purpose of our proposed model was to estimate the optimal input lag to get the best output from the deep learning mechanism. The proposed model is depicted in Fig. 3 and the other relevant information are given in appendix B [Pseudocode 3; Figure B3]. Autocorrelation function and random forest were used for lag estimation, *i.e.*, to estimate the value of n.

3.4.1. Autocorrelation function

The autocorrelation function at lag$j$is denoted as$\rho_j = \frac{\beta_j}{\beta_0}$where, $\beta_j = \text{cov}(x_i, x_{i+j}) = \frac{1}{n}\sum_{i=1}^{n+j}(x_i - \bar{x})(x_{i+j} - \bar{x})$.

β_0 is the stochastic variance process.

A plot of β_j at j^{th} lag is known as correlogram. The correlogram provides information about the standard error and confidence interval at β_j. The maximum significant lag was considered as input lag to predict the output, the details of which are provided in appendix B [Pseudocode 1 (Figure B1)].

3.4.2. Random forest

Random forest is a flexible, easy-to-use machine learning method that, in most cases, delivers good results even without hyper-parameter tuning. Because of its simplicity and diversity, it is also one of the most often used algorithms. Random forest is supervised machine learning algorithm. It builds a "forest" out of an ensemble of decision trees, which are generally trained using the "bagging" process. The bagging method's general concept is to combine several learning models that improves the final outcome.

3.4.2.1. Procedure.

- Pick k data points at random from the training dataset.

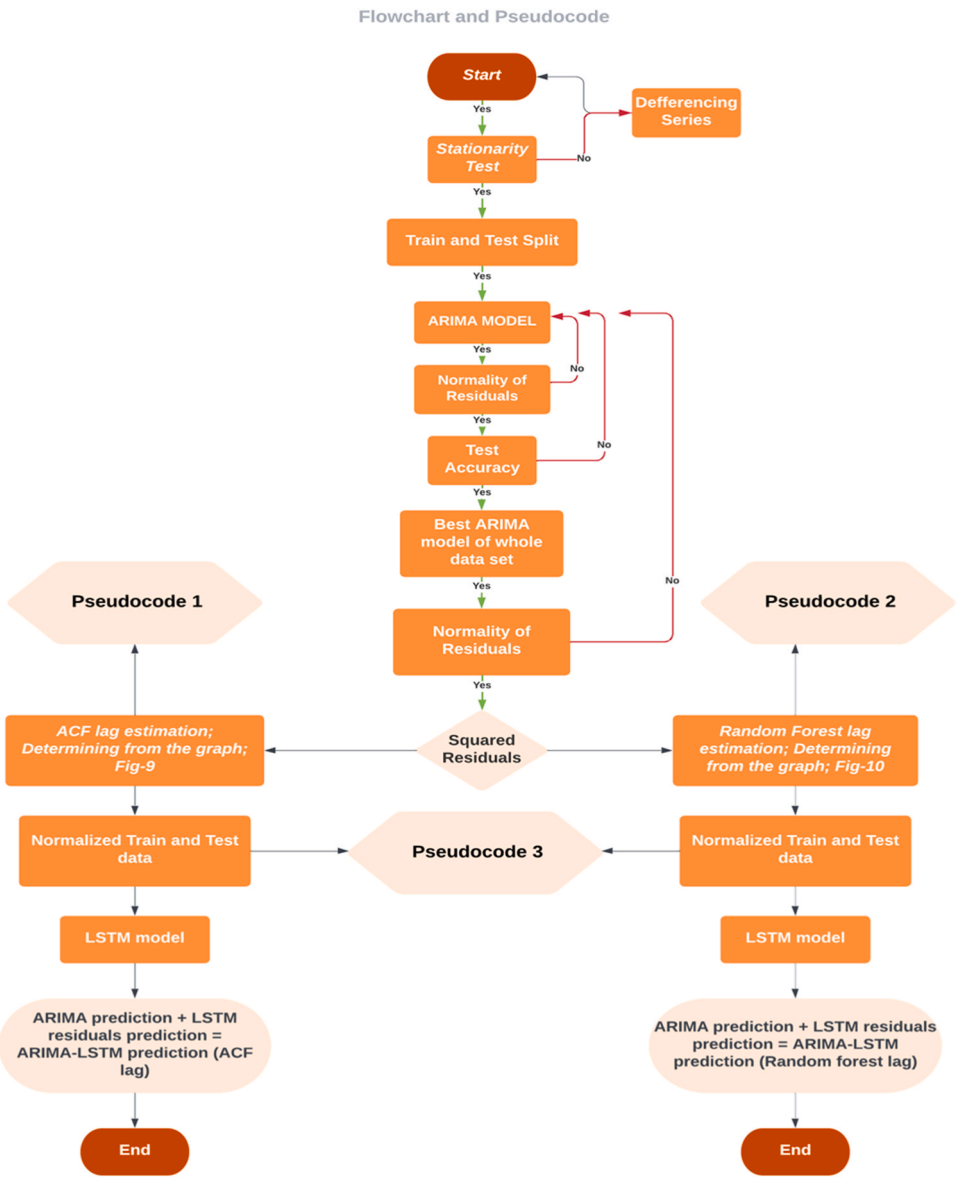


Fig. 3. The proposed methodology flowchart with pseudocode.

- Create a decision tree based on these k points.
- Select the number N of trees one wish to construct and repeat the procedures above.
- Make each of the N-tree trees predict the value of y for a new data point, then assign the new data point to the average of all the predicted y values.

Form of the regression trees model, where y takes the form:

$$f(X) = \sum_{m=1}^{M} c_m . 1_{(X \in R_m)}$$

where, R_1, R_2, …, R_M represents partition of feature space (*i.e.*, optimal lag length).

In our study, we have used the above regression tree model, to identify the optimal lag length (*i.e.*, feature). Here, input (X) are the different number lagged input of the series which varies between 1 and 12 and the target variable (y) is the original series itself.

After estimating the input lag, the LSTM model was carried out with the activation function 'Rectified Linear Unit', optimizer 'Adam' and the loss function 'MSE'. The final prediction was evaluated by adding ARIMA prediction and LSTM residual prediction. The entire proposed methodology framework was depicted in Fig. 3.

3.5. Statistical test

3.5.1. Grubbs' test

In case of a univariate data set, approximately following a normal distribution, a single outlier can be found using Grubbs' test (Grubbs 1969 and Stefansky 1972) [41,42]. The biggest absolute difference from the sample mean expressed in units of the sample standard deviation is known as the Grubbs' test statistic.

Grubbs' test is defined for the hypothesis:

$\mathbf{H_0}$: There are no outliers in the data set.

$\mathbf{H_1}$: There is exactly one outlier in the data set.

Test Statistic: The Grubbs' test statistic is defined as:

$$G = \frac{\max|Y_i - \overline{Y}|}{s}$$

with $\overline{Y}$ and s denoting the sample mean and standard deviation, respectively.

Test whether the minimum value is an outlier $G = \frac{\overline{Y} - Y_{\min}}{s}$; with $Y_{\min}$ denoting the minimum value. Test whether the maximum value is an outlier; with $Y_{\max}$ denoting the maximum value.

$$G > \frac{N-1}{\sqrt{N}} \sqrt{\frac{\left(t_{\alpha/(2N),N-2}\right)^2}{N-2+\left(t_{\alpha/(2N),N-2}\right)^2}}$$

with $t_{\alpha/(2N),N-2}$denoting the critical value of the t distribution with (N-2) degrees of freedom and a significance level of α/(2 N). For one-sided tests, we use a significance level of α/N.

3.5.2. Augmented Dickey Fuller (ADF) test

Dickey and Fuller (1979) [43] established the Augmented Dickey Fuller test and takes the following form

$$\Delta Y_t = \alpha + \beta_t + \delta Y_{t-1} + \sum \lambda_i \Delta_{t-i} + e_i$$

where ΔY_t is the first difference of Y and α allows for a non-zero intercept or drift component *i.e.*, constant *t* is included to allow for a deterministic trend as the data may be trend stationary. The null hypothesis here is Y_t has a unit root (H_0: δ = 0) against δ is negative. Thus, the test consists of testing the negativity of δ in the above equation. The test statistic is given by $DF = \frac{\delta}{SE(\delta)}$.

It can be compared to the relevant critical value for the Dickey-Fuller Test. If the test statistic is less than the critical value, then the null hypothesis of δ = 0 is rejected and the data is stationary.

3.5.3. Phillips and Perron (PP) Test

Phillips and Perron (1988) [44] developed several unit root tests that have become popular in the analysis of financial time series. The PP unit root tests differ from the ADF tests primarily in how they handle serial correlation and heteroskedasticity in the errors. The PP tests specifically ignore any serial correlation in the test regression whereas the ADF tests employ a parametric autoregression to simulate the ARMA structure of the errors in the test regression. The test regression for the PP tests is

$$\Delta y_t = \beta' D_t + \pi y_{t-1} + u_t$$

where u_t is I(0) and may be heteroskedastic. The PP tests correctly for any serial correlation and heteroskedasticity in the errors u_t of the test regression by directly modifying the test statistics $t_{\pi=0}$ and $T\hat{\pi}$. These modified statistics, denoted Zt and Zπ, are given by

$$Z_t = \left(\frac{\hat{\sigma}^2}{\hat{\lambda}^2}\right)^2 . t_{\pi=0} - \frac{1}{2}\left(\frac{\hat{\lambda}^2 - \hat{\sigma}^2}{\hat{\lambda}^2}\right)\left(\frac{T.SE(\hat{\pi})}{\hat{\sigma}^2}\right)$$

$$Z_\pi = T_{\hat{\pi}} - \frac{1}{2}\frac{T^2 SE(\hat{\pi})}{\hat{\sigma}^2}\left(\hat{\lambda}^2 - \hat{\sigma}^2\right).$$

The term $\hat{\sigma}^2$and $\hat{\lambda}^2$are consistent estimates of the variance parameters

$$\sigma^2 = \lim_{T\to\infty} T^{-1} \sum_{t=1}^{T} E[u_t^2]$$

$$\lambda^2 = \lim_{T\to\infty} T^{-1} \sum_{t=1}^{T} E[T^{-1} S_T^2]$$

where $S_T = \sum_{t=1}^{T} u_t$. The sample variance of the least squares residual $\widehat{u}_t$. is a consistent estimate of σ^2, and the Newey-West long-run variance estimate of u_t using $\widehat{u}_t$ is a consistent estimate of λ^2. Under the null hypothesis that π = 0, the PP Z_t and Z_π statistics have the same asymptotic distributions as the ADF t-statistic and normalized bias statistics. One advantage of the PP tests over the ADF tests is that the PP tests are robust to general forms of heteroskedasticity in the error term u_t. Another advantage is that the user does not have to specify a lag length for the test regression.

3.5.4. Ljung–Box test

The Ljung-Box test is a sort of statistical analysis that determines which autocorrelations in a group of time series are different from zero. It is named after Greta M. Ljung and George E. P. Box [45]. This test is a portmanteau because it examines randomness overall rather than examining it at each lag.

The Ljung–Box test may be defined as:

H_0: The data are independently distributed (*i.e.* the correlations in the population from which the sample is taken are 0, so that any observed correlations in the data result from the randomness of the sampling process).

H_1: The data are not independently distributed; they exhibit serial correlation.

$$Q = n(n+2)\sum_{k=1}^{h} \frac{\widehat{\rho}_k^2}{n-k}$$

where *n* is the sample size, $\widehat{\rho}_k$ is the sample autocorrelation at lag *k*, and *h* is the number of lags being tested.

3.5.5. Diebold-Mariano (DM) test

The Diebold-Mariano (DM) [49] test is probably the most commonly used tool to evaluate the significance of differences in forecasting accuracy. It is an asymptotic z-test of the hypothesis that the mean of the loss differential series,

$$\Delta_k^{A,B} = L(\varepsilon_k^A) - L(\varepsilon_k^B)$$

where $\varepsilon_k^Z = p_k - \widehat{p}_k$ is the prediction error of model Z for time step k and L (·) is the loss function. For point forecasts, we usually take $L(\varepsilon_k^Z) = \left|\varepsilon_k^Z\right|^p$ with p = 1 or 2, which corresponds to the absolute and square losses.

3.6. Model Selection and Forecasting accuracy criteria

$$AIC = 2k - 2\ln(L)$$

$$BIC = -2 \times \ln \quad L + k \times \ln(n)$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (X_i - \widehat{X}_i)^2}{n}}$$

$$MAPE = \frac{\sum_{i=1}^{n} \left|\frac{X_i - \widehat{X}_i}{X_i}\right|}{n} \times 100$$

$$MASE = mean\left(\frac{|e_j|}{\frac{1}{T-1}\sum_{T=2}^{n} |X_t - X_{t-1}|}\right).$$

The entire analysis was performed in Python; Jupiter platform by using popular packages like pmdarima (https://pypi.org/project/pmdarima/), arch (https://pypi.org/project/arch/), Keras Tensorflow (https://keras.io).

4. Empirical results and discussion

4.1. Basic statistics

Various performance metrics such as the mean, standard deviation,

Table 1
Descriptive Statistics of Pluses price index series.

	Gram	Moong	Urad
Mean	82.484	81.893	90.579
Standard Deviation	50.801	45.988	48.660
Skewness	1.143	0.499	0.953
Kurtosis	0.915	-1.187	0.227
Min	22.070	23.660	28.640
Max	278.900	179.700	247.600

skewness, kurtosis, minimum, and maximum are employed in the data series visualization are given in Table 1 for the analysis of the data. The average gram price index is found to be 82.484, with a range of 22.07–278.9, which is the highest among the other series. Positive skewness is shown by all the pulse series, indicating an upward trend in prices during the study period from April 1994 to March 2021. Asymmetrical and platykurtic distributions are exhibited by all the series. To gain a further understanding of the monthly price range behaviour of the series, a visualization in Fig. 4 was created. This analysis revealed that the highest price range for gram occurs in November and for urad in June, while the price range of moong remains relatively steady throughout the year. The *p*-value obtained from the Grubbs' test, which was less than 0.05, confirmed the presence of extreme values of the series in appendix A (Table A1).

4.2. Model development

The objective of the analysis is to determine the best model based on its prediction accuracy through the evaluation of different models. Traditional time series models such as ARIMA and GARCH, a deep learning neural network model (LSTM), and an ARIMA-LSTM hybrid model were employed to achieve this. Selection of these models was made according to the methodology outlined in the methodology section and a comparison was made in terms of the accuracy of the models in predicting the data series. The objective is to identify the model that provides the most accurate predictions and is best suited for the data series under analysis.

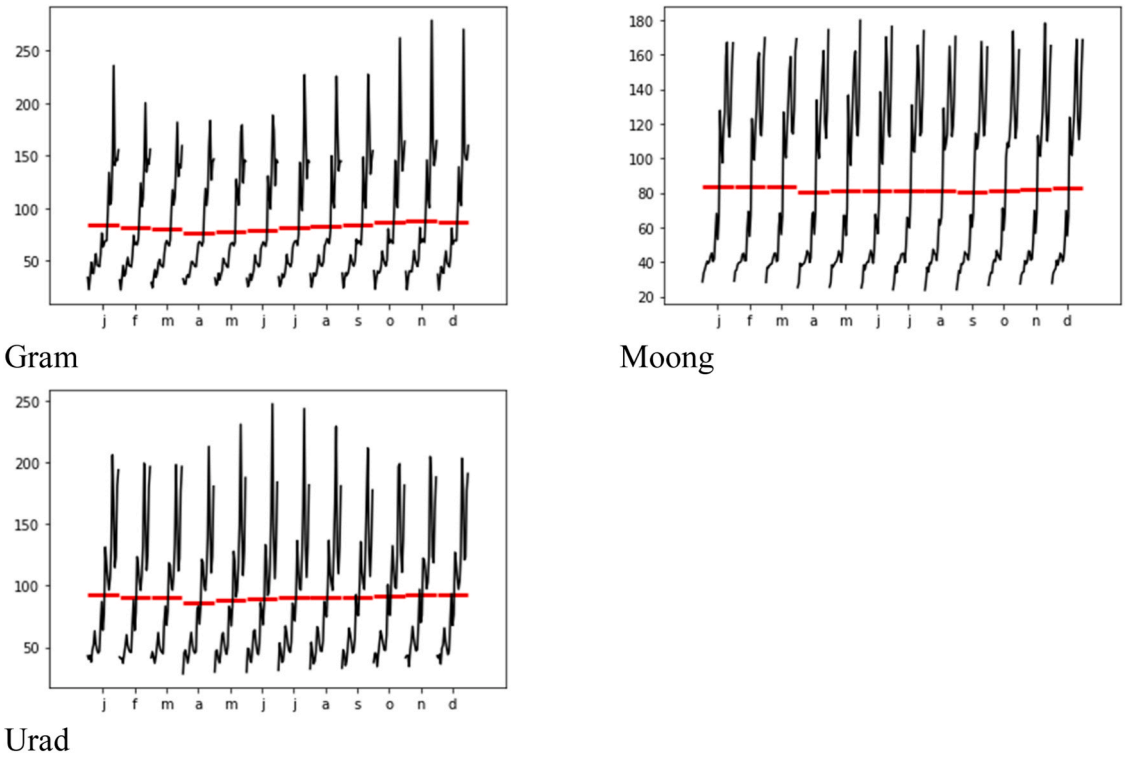


Fig. 4. Monthly plot of different price index series.

Table 2
Model parameter and Ljung-Box Q test for best ARIMA model.

	Model	Parameter Estimation			Ljung-Box Q test		
		Coefficients	Estimates	S.E.	Residuals	Statistic	p-value
Gram	ARIMA(4,1,3)	ar1	0.044	0.056	upto 5 lag	4.902	0.428
		ar2	0.687	0.043	upto 10 lag	19.787	0.052
		ar3	0.633	0.040			
		ar4	-0.485	0.052			
		ma1	0.519	0.030			
		ma2	-0.537	0.021			
		ma3	-0.982	0.033			
Moong	ARIMA(1,1,0)	ar1	0.412	0.052	upto 4 lag	1.992	0.737
					upto 10 lag	15.212	0.125
					upto 15 lag	20.189	0.165
Urad	ARIMA(2,1,0)	ar1	0.644	0.056	upto 5 lag	6.969	0.223
		ar2	-0.140	0.056	upto 15 lag	24.435	0.058

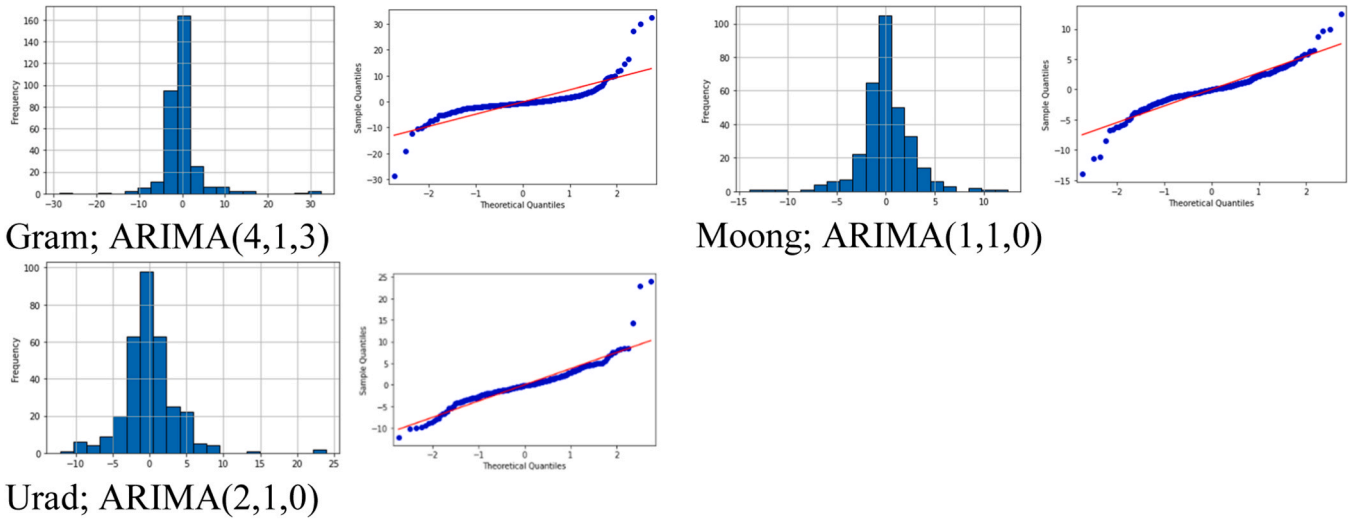


Fig. 5. Diagnostic Check of ARIMA model in different series.

Table 3
Model parameter and Ljung-Box Q test for best ARIMA-GARCH model.

	Model	Parameter Estimation				Ljung-Box Q test		
		Coefficients	Estimates	S.E.	t-statistic	Residuals	Statistic	p-value
Gram	ARIMA(4,1,3)-GARCH(1,1)	mu	-0.723	0.292	-2.476	upto 5 lag	4.391	0.495
		omega	0.428	1.128	0.379	upto 9 lag	16.412	0.059
		alpha	0.219	0.177	1.866	upto 30 lag	37.921	0.152
		beta	0.781	0.122	6.418			
Moong	ARIMA(1,1,0)-GARCH(1,1)	mu	-0.099	0.082	-1.213	upto 5 lag	3.065	0.547
		omega	0.086	0.059	1.450	upto 10 lag	12.559	0.244
		alpha	0.173	0.063	2.728	upto 15 lag	16.725	0.241
		beta	0.828	0.058	14.314			
Urad	ARIMA(2,1,0)-TARCH(1,1) with t dist.	mu	-0.132	0.131	-1.010	upto 5 lag	5.455	0.363
		omega	0.077	0.074	1.045	upto 10 lag	14.356	0.157
		alpha	0.187	0.072	2.597	upto 15 lag	17.891	0.268
		gamma	-0.187	0.040	-4.694			
		beta	0.906	0.075	12.039			
		nu	4.565	1.122	4.054			

4.2.1. *ARIMA model*

Before the development of the ARIMA model, it was deemed necessary to ensure the stationarity of all data series. The Augmented Dickey-Fuller (ADF) test [43] and the Phillips-Perron (PP) test [44] were employed for this purpose, as outlined in appendix A (Table A2). The results indicated that all data series become stationary at their level point after undergoing the first differencing. Based on this evidence, the order of d was set to be 1 in the ARIMA model. The ARIMA model was then constructed to capture the mean effect of all the data series, as outlined in Table 2. The best-fit ARIMA models for the gram, moong and urad series, based on the estimated parameters and standard errors were found to be ARIMA(4,1,3), ARIMA(1,1,0), and ARIMA(2,1,0) respectively. The autocorrelation of the residuals at zero at different lags was confirmed by the p-value (greater than 0.05) of the Ljung-Box Q test [45]. This indicated that the residuals followed a normal distribution with zero mean and constant variance, which was further confirmed through the visualization of histograms and Q-Q plots in Fig. 5.

The histograms and Q-Q plots also show evidence of the presence of volatility in residuals, implying that the GARCH model needs to be applied.

4.2.2. *ARIMA-GARCH model*

An ARCH LM test was performed on the residuals obtained from the ARIMA model to detect the presence of volatility in appendix A (Table A3). All the p-values obtained were less than 0.05 from the test at different lags (10, 15 and 20), which confirmed the presence of volatility in the residuals. So the GARCH model was employed for the trained residuals of the ARIMA model. The model was developed as the ARIMA-GARCH model. The best models were determined to be ARIMA(4,1,3)-GARCH(1,1) for gram, ARIMA(1,1,0)-GARCH(1,1) for moong, and ARIMA(2,1,0)-TARCH(1,1) with a t distribution for urad, with the normality of the residuals again confirmed by the Ljung-Box Q test as p-value was greater than 0.05 (Table 3). The normally distributed nature of residuals can also be followed in Fig. 7. Also we have displayed the conditional volatility plot and residuals plot obtained from best fitted ARIMA-GARCH model to each series (Fig. 6). Higher conditional variance was observed in those scenarios where volatility behaviour was present in the series. The mean effect of the series (ARIMA) and volatile effect (ARIMA-GARCH) was compared with model selection criteria, the lowest value of AIC [46,47] and BIC [48] in appendix A (Table A4). From the table, ARIMA-GARCH models obtained lower AIC and BIC values than the ARIMA models for all the series in training data. This confirmed that the ARIMA-GARCH model was superior for capturing volatility leading to a better prediction for the test data series.

4.2.3. *LSTM model*

After implementing ARIMA and ARIMA-GARCH, the LSTM model was developed with a deep learning algorithm (Fig. 8). Before developing LSTM models, all the training data series were normalized using 'MinMaxScaler', available in the Python platform. A 'TimeseriesGenerator' imported from the 'Keras' library was used to transform the data series for supervising the neural network learning problems.

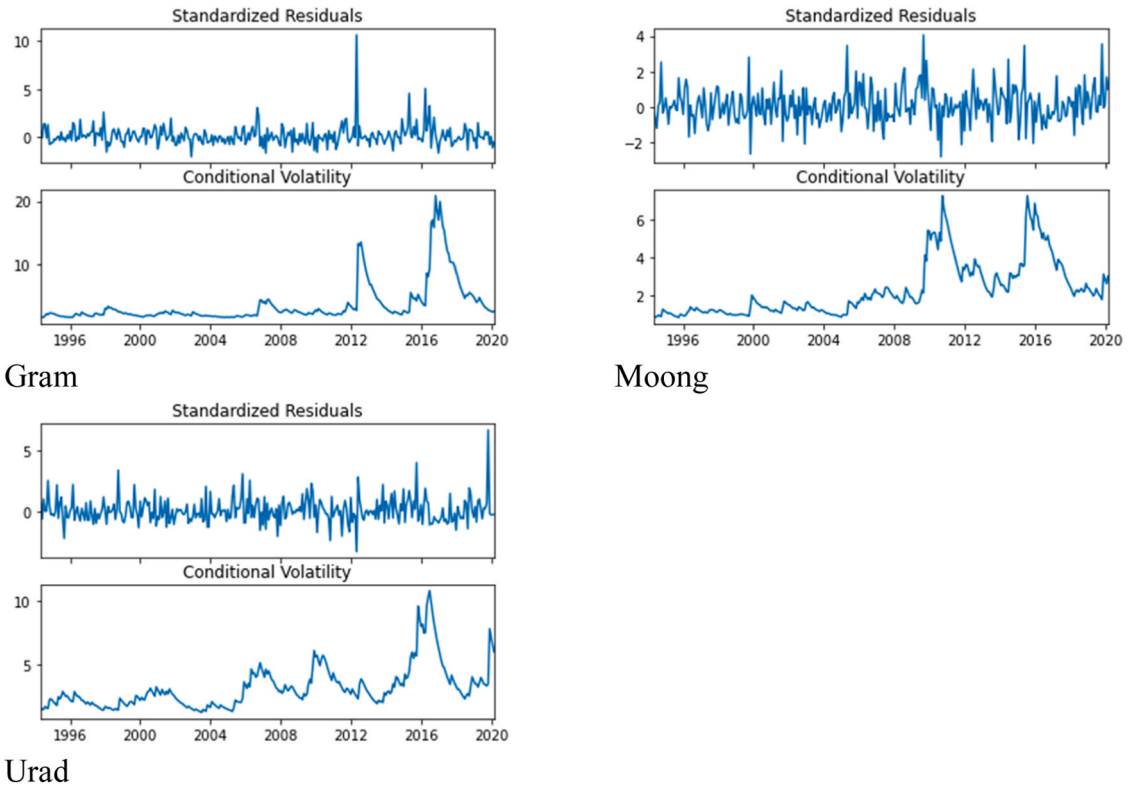


Fig. 6. Standardized residuals and conditional volatility of the ARIMA-GARCH model.

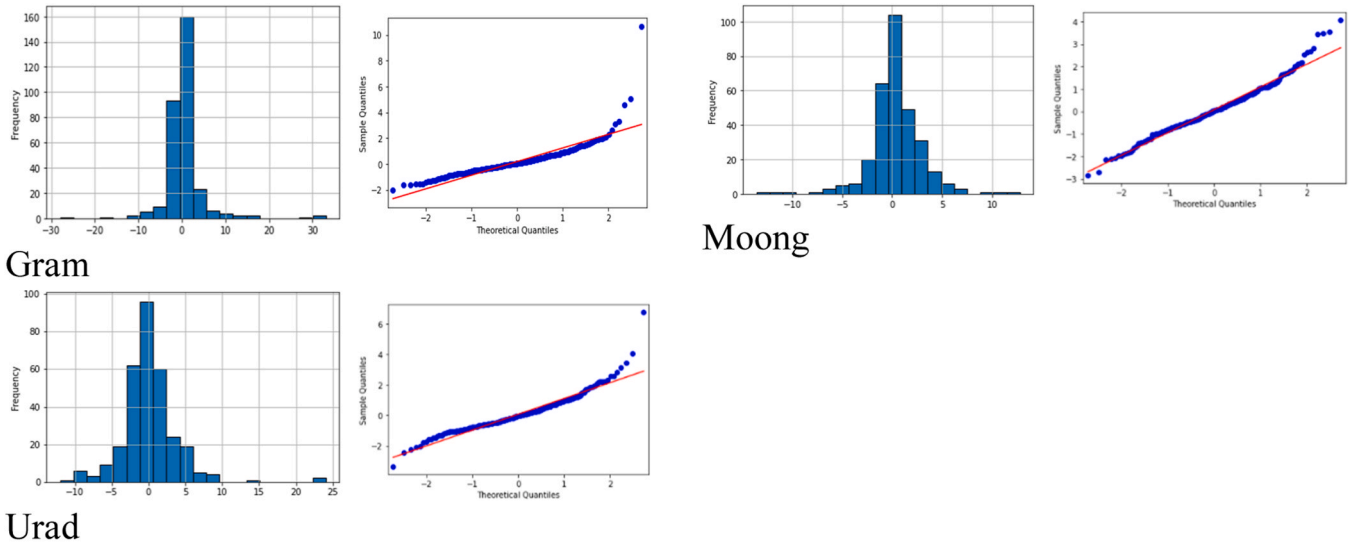


Fig. 7. Diagnostic check of residuals obtained from ARIMA-GARCH model.

LSTM structural parameter estimation entails determining the number of LSTM layers and units to capture complicated temporal patterns while avoiding overfitting. During training, tuning of hyperparameters such as batch size, learning rate, and dropout rate ensures adequate convergence and regularization. As the series were volatile, the one step cross validation technique was employed to evaluate the model's performance over multiple time periods. The grid search method was then used to estimate the appropriate hyperparameter tuning. Various combinations of LSTM models were tried by a number of hidden layers to obtain appropriate forecasting models, using the Rectified Linear Unit as activation function, Adam as optimizer and the MSE as loss function. The Adam optimizer's purpose is to iteratively alter the model's parameters (weights and biases) based on the backpropagation gradients. These gradients represent the size and direction of the changes required to improve the model's performance on the provided training data. The random initial weights offer the requisite variation and symmetry-breaking for effective learning. The Adam optimizer can alter the weights as the model trains based on the data and the optimization objective, progressively improving the model's performance [49]. The best models for all the series are presented in Table 4. The input lag was selected as 12 for the monthly base series. In case of gram, the LSTM (66, 1,1) model with one hidden and output layer consisting of 66 LSTM blocks, 'adam' optimizer obtained MSE loss function value between 0.0081 and 5.5396e-04 in 25 epochs. For the moong series, the LSTM

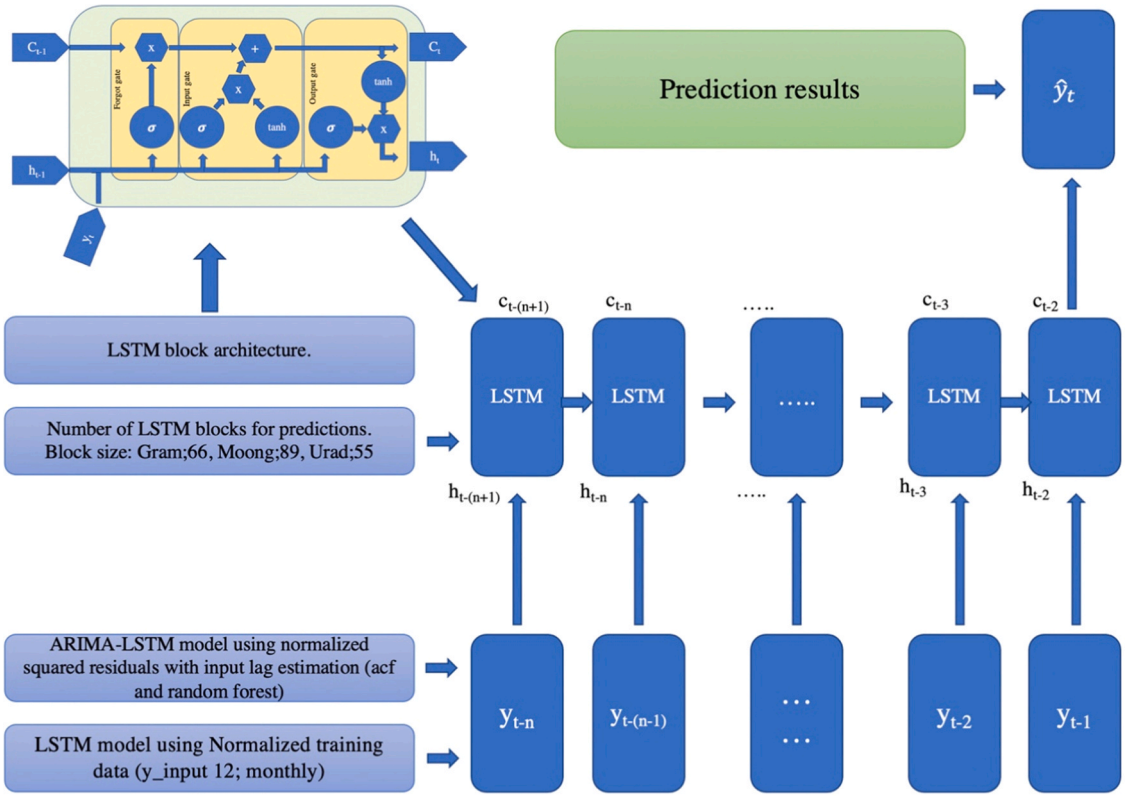


Fig. 8. The framework of the LSTM model applied.

Table 4
LSTM model for pulses price index series.

Series	Input	Neuron	Dense	Total Param	Epochs
Gram	12	66	1	18019	25
Moong	12	89	1	32486	20
Urad	12	55	1	12596	25

(89,1,1) model with one hidden and output layer, 89 LSTM blocks or neurons, 'adam' optimizer obtained the loss function MSE value ranging from 0.0183 to 6.9796e-04 in 20 epochs. And for the urad series, the model LSTM(55,1,1) with one hidden and output layer, 55 LSTM neurons, 'adam' optimizer obtained the loss function MSE in 25 epoch which ranged from 0.0101 to 7.1788e-04. The visual depiction of the LSTM model obtained for each data series was depicted in appendix B (Figure B4). The loss function MSE of LSTM was displayed in appendix B (Figure B5).

4.2.4. *Proposed Model (ARIMA-LSTM based on random forest)*

Based on the objective of the study, the hybrid ARIMA-LSTM model was developed by estimating the input lag length selection using traditional ACF and random forest approaches. The total process of the proposed model is depicted in Fig. 3. The pseudocode of our proposed model was given in the appendix B (Figure B1, B2, B3). ARIMA model squared residuals were used to estimate input lag through ACF and random forest technique (Figs. 9 and 10). From the correlogram, maximum significant lags were observed up to 6, 12, and 6 for the gram, moong and urad series respectively. From the random forest technique, the maximum lag correlation was observed up to 3, 5, and 5 for the same series. Based on these two techniques, we chose the relevant input lag and then implement the LSTM model for residual prediction. Table 5 presents the different lags selected for each series using ACF and random forest techniques. It is noteworthy that for each series the number of lags selected from the random forest technique was smaller as compared to that of ACF. This directly reduces the number of parameters to be estimated and the computational time as well.

Further, the hybrid models implemented are graphically depicted along with their MSE loss function in appendix B (Figure B6 and Figure B7) for ACF based hybrid and in a similar manner, in appendix B (Figure B8 and Figure B9) represent the prosed random forest based approach. The MSE values for the proposed model have reduced significantly for gram and moong, and in case of urad series the values are statistically at par.

After successful model development (ARIMA-LSTM), we intended to study the appropriateness of the models using run test on the residuals. This has been done with the assumption that if the residuals are random in nature, it indicates that the employed model has satisfactorily captured all the patterns present in the data set. For all the three pulses series under consideration, we obtained p value of the test statistic to be > 0.05. This supports our assumption of the randomness of the residuals and model to be a good fit to the data set.

4.3. *Forecasting accuracy*

Finally, we investigated the prediction behaviour by combining ARIMA prediction with LSTM residual prediction. Table 6 represent all the five different models prediction for the 12-point forecast horizon. In addition to the five models considered primarily for our study, we also include the Naïve method and ETS model for the sake of comprehensive comparison. Thus, a total of seven models are compared based on their

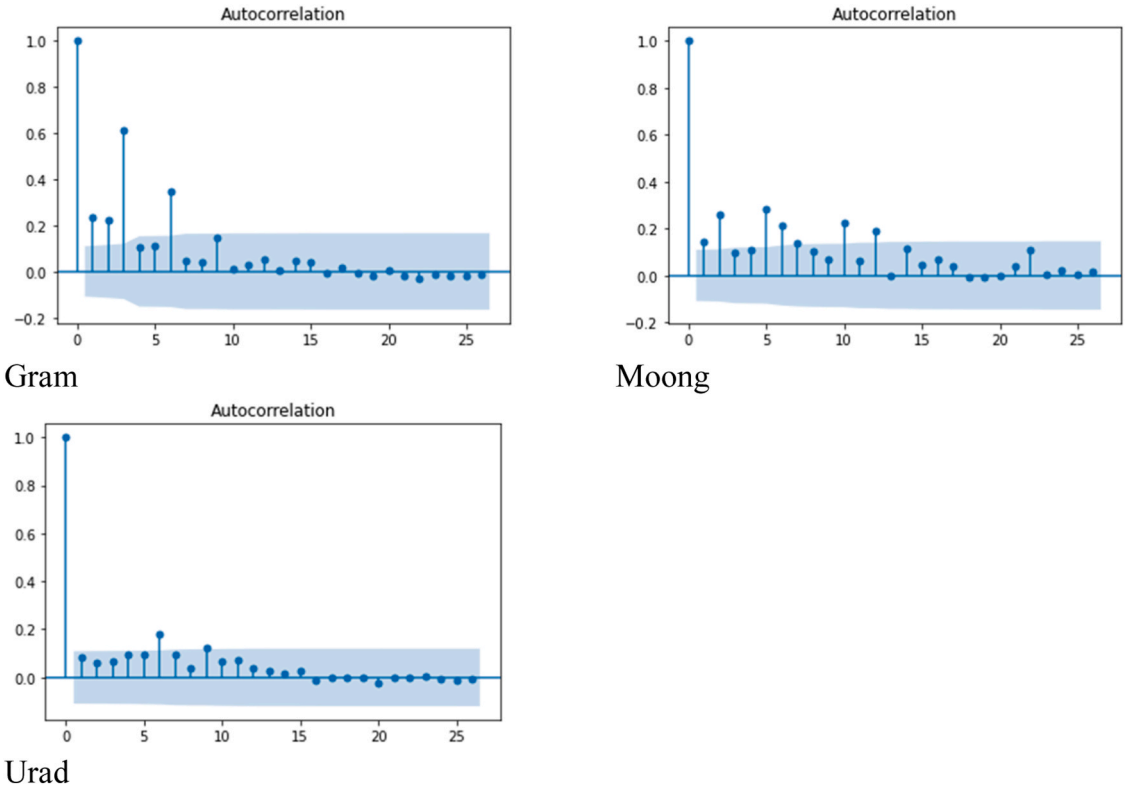


Fig. 9. Autocorrelation of Squared residuals obtained from ARIMA model.

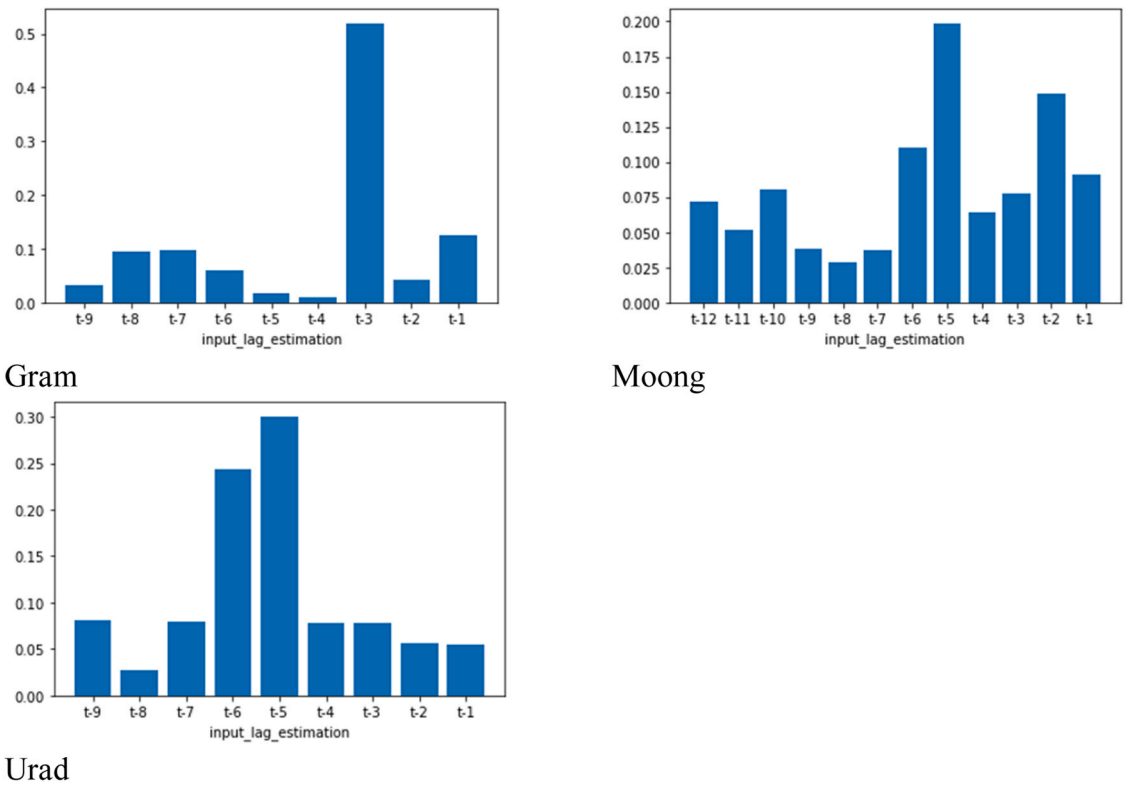


Fig. 10. Input Lag estimation using random forest regression.

Table 5
Hybrid ARIMA-LSTM model after Input lag estimation.

Series	Input_lag		Neuron	Dense	Total Param	Epochs
	ACF	random forest				
Gram	6	3	66	1	18019	25
Moong	12	5	89	1	32486	20
Urad	6	5	55	1	12596	25

forecasting accuracy. The forecasting was carried out at 12 points horizon, using one step ahead approach. This one step ahead approach was selected intentionally to address the issue of cross validation. Table 7 presents the different values obtained for measures of accuracy considered namely RMSE, MAPE and MASE. Also, we indicated the MASE ratio against Naïve method to the other models.

The results have indicated that the ARIMA-LSTM hybrid model based on the random forest approach has the lowest RMSE, MAPE and MASE values for all the series followed by ARIMA-LSTM based on the ACF approach except for the gram series, where the LSTM model showed superior performance. To ascertain our findings statistically, we have used Diebold-Mariano (DM) test [50] to determine whether the forecast has the same accuracy or not (Table 8). The test was conducted between the best fit statistical model and LSTM and ARIMA-LSTM hybrid models. Test results revealed that for gram and moong series proposed ARIMA-LSTM model (random forest) outperformed the other competing

Table 6
Prediction obtain from different models.

Date	Gram						Moong					
	Actual	ARIMA	ARIMA-GARCH	LSTM	ARIMA-LSTM	ARIMA-LSTM (random forest)	Actual	ARIMA	ARIMA-GARCH	LSTM	ARIMA-LSTM	ARIMA-LSTM (random forest)
Apr-20	146.700	136.669	139.034	149.912	142.550	147.059	174.200	164.489	167.539	163.077	171.439	176.695
May-20	144.800	136.558	141.498	141.035	142.506	147.273	179.700	165.779	170.835	165.412	172.283	168.677
Jun-20	143.800	137.536	142.781	143.013	143.531	148.213	176.100	166.575	171.132	166.968	172.593	170.399
Jul-20	144.300	138.094	142.824	145.431	144.088	148.965	173.600	167.168	172.053	168.095	172.660	173.011
Aug-20	144.800	139.662	143.929	148.101	145.695	150.552	170.400	167.677	172.122	168.893	172.793	170.648
Sep-20	154.800	140.841	144.717	150.902	146.843	151.735	164.300	168.151	172.301	169.435	172.938	170.585
Oct-20	163.600	141.903	147.635	153.716	147.921	152.807	162.600	168.612	172.919	169.799	173.120	169.481
Nov-20	164.300	143.535	148.844	156.487	149.555	154.441	165.100	169.066	172.989	170.006	173.377	169.844
Dec-20	159.800	144.377	149.242	159.177	150.397	155.283	168.500	169.518	173.313	170.101	173.107	170.121
Jan-21	155.600	145.690	150.545	161.763	151.711	156.596	166.700	169.969	173.543	170.119	173.531	169.928
Feb-21	156.300	146.898	151.325	164.216	152.919	157.804	169.700	170.420	173.971	170.068	173.993	170.089
Mar-21	159.600	147.649	152.055	166.544	153.669	158.555	169.100	170.871	174.444	169.966	174.082	170.224
Date	Urad											
	Actual	ARIMA	ARIMA-GARCH	LSTM	ARIMA-LSTM	ARIMA-LSTM (random forest)						
Apr-20	180.600	175.578	181.123	179.585	194.015	181.736						
May-20	187.700	175.341	181.397	182.500	185.034	181.915						
Jun-20	184.000	175.607	181.961	185.509	185.291	182.187						
Jul-20	181.700	176.046	181.887	188.146	185.717	182.630						
Aug-20	180.900	176.525	182.064	190.362	186.199	183.112						
Sep-20	177.600	177.007	182.103	192.192	186.727	183.596						
Oct-20	181.500	177.484	182.178	193.610	186.806	184.073						
Nov-20	188.100	177.957	183.360	194.567	187.270	184.546						
Dec-20	191.000	178.429	184.057	195.196	187.735	185.019						
Jan-21	194.000	178.901	184.082	195.620	188.201	185.490						
Feb-21	196.600	179.373	184.511	195.892	188.669	185.962						
Mar-21	196.700	179.845	184.762	196.007	189.129	186.434						

Table 7
The measure of forecasting accuracy of different Models.

Series	Model	RMSE	MAPE	MASE	MASE ratio against Naïve method
Gram	ETS	24.18101	14.2426	7.8355	1.5123
	Naïve method	16.5273	9.3749	5.1813	1.0000
	ARIMA	12.704	7.431	4.0824	0.7879
	ARIMA-GARCH	8.601	4.44	2.4664	0.4760
	LSTM	5.746	2.957	1.6283	0.3143
	ARIMA-LSTM	7.638	3.61	2.021	0.3901
	ARIMA-LSTM(random forest)	**5.231**	**2.657**	**1.4522**	0.2803
Moong	ETS	8.1186	4.4465	3.8822	0.9431
	Naïve method	9.3779	4.6276	4.1165	1.0000
	ARIMA	6.517	3.043	2.6981	0.6554
	ARIMA-GARCH	6.471	3.509	3.05501	0.7421
	LSTM	6.839	3.153	2.7893	0.6776
	ARIMA-LSTM	6.099	3.224	2.7943	0.6788
	ARIMA-LSTM(random forest)	**4.868**	**2.167**	**1.9008**	0.4618
Urad	ETS	17.4952	8.3075	5.4906	1.6065
	Naïve method	11.6841	5.14	3.4178	1.0000
	ARIMA	10.746	4.924	3.2639	0.9550
	ARIMA-TARCH	6.608	2.658	1.7736	0.5189
	LSTM	6.976	2.917	1.8605	0.5444
	ARIMA-LSTM	6.516	2.981	1.9331	0.5656
	ARIMA-LSTM(random forest)	**5.956**	**2.605**	**1.7261**	0.5050

Table 8
Diebold-Mariano Test for accuracy prediction.

Series	Model compared	DM test Statistic	P-value
Gram	(ARIMA-GARCH)-LSTM	2.013	0.044
	(ARIMA-GARCH)-(ARIMA-LSTM (random forest))	2.356	0.018
Moong	(ARIMA-GARCH)-LSTM	0.485	0.628
	(ARIMA-GARCH)-ARIMA	0.623	0.533
	LSTM-ARIMA	0.691	0.489
	ARIMA-(ARIMA-LSTM)	1.910	0.056
	(ARIMA-GARCH)-(ARIMA-LSTM)	3.966	0.049
	LSTM- (ARIMA-LSTM(random forest))	2.203	0.027
Urad	LSTM-(ARIMA-TARCH)	0.115	0.909
	(ARIMA-TARCH)-(ARIMA-LSTM (random forest))	0.394	0.693
	LSTM-(ARIMA-TARCH)	0.207	0.835

models. But, for the urad series we could not statistically differentiate among the competing models. Such results from the urad series can be attributed to the fact that not much change in the lag structure was observed between the two approaches using ACF (6 lags) and random forest (5 lags).

5. Conclusion

In this present investigation, we have introduced a novel and efficient random forest-based ARIMA-LSTM hybrid model. Further, we attempted to empirically document the comparative performance of traditional time series models with deep learning algorithms in case of forecasting volatile price series. The models selected were ARIMA, ARIMA-GARCH, LSTM and ARIMA-LSTM owing to their wide acceptability in literature for satisfactorily modelling and forecasting financial time series. Our study was conducted on three major pulses price series in India which had inherent volatility. For the statistical model, we identify ARIMA-GARCH as better suited to model the data sets than ARIMA due to lower AIC and BIC values. For forecasting volatility of the price series, ARIMA-GARCH model performed uniformly superior to its competitor ARIMA model in terms of lower RMSE, MAPE and MASE values. Further, the results obtained from LSTM were compared with that of the statistical models. We also document an interesting and encouraging finding that for those series in which LSTM performed superior, the tune of improvement ranged between 8% and 25% for RMSE, 2–28% MAPE and 2–29% MASE respectively (Table 9).

This manuscript has successfully revealed the importance of efficient lag length estimation for machine learning models (LSTM) for time series forecasting. We also have satisfactorily established the superiority of hybrid models for volatile series. We see our proposed model as a promising alternative to traditional statistical models, which has stringent assumptions and are hard to satisfy at times. We are also hopeful that this study will certainly benefit different stakeholders and policymakers by providing accurate price forecast. Further, we strongly believe that our study has contributed to the vast and rapidly growing literature on machine learning models in the domain of time series.

The present investigation was carried out primarily to make use of feature selection algorithm for obtaining optimal lag length, which

Table 9
Forecast accuracy improvement percentage.

Series	Model	RMSE	RMSE improvement percent	MAPE	MAPE improvement percent	MASE	MASE improvement percent
Gram	LSTM	5.746	8.964	2.957	10.145	1.6283	10.815
	ARIMA-LSTM(RF)	5.231		2.657		1.4522	
Moong	ARIMA	6.517	25.303	3.043	28.787	2.6981	29.550
	ARIMA-LSTM(RF)	4.868		2.167		1.9008	
Urad	ARIMA-TARCH	6.608	9.867	2.658	1.994	1.7736	2.678
	ARIMA-LSTM(RF)	5.956		2.605		1.7261	

would improve the forecasting efficiency of hybrid ARIMA-LSTM model. Hence, we made use of random forest algorithm to achieve our objective. But, it would be interesting to document the findings when information theory based feature selection methods such as forward selection minimal redundancy maximal relevance (FSMRMR) [51] and conditional mutual information maximization (CMIM) [52] are used. As future endeavour, authors are working in the direction of extending the proposed framework with genetic algorithm (GA) and particle swarm optimization (PSO) based LSTM model.

Avail ability of data and material

The datasets used and analysed in this study are available in the public domain.

Funding

Not applicable.

CRediT authorship contribution statement

Achal Lama, Soumik Ray: Conceptualization, Methodology, Software, Data curation, Validation, Writing – original draft. **Bishal Gurung**: Visualization, Investigation. **Pradeep Mishra, Tufleuddin Biswas, Soumitra Sankar Das**: Writung – review & editing. **Achal Lama, Soumik Ray**: Have contributed equally and to be treated as joint first authors.

Declaration of Competing Interest

The authors declared that they have no conflicts of interest.

Data availability

Data will be made available on request.

Appendix A

Table A1
Grubbs's test for detecting Outliers.

	Test Statistic	P- Value	Outlier
Gram	3.86635	0.01499	278.9; Nov-2016
Moong	4.1268	0.0105	179.7; May-2020
Urad	3.9622	0.01254	247.6; June-2016

Table A2
Stationarity Test of Pulses price index series.

		ADF Test			PP Test		
Series		Statistic	Lag Order	P-Value	Statistic	Truncation lag parameter	P-Value
Gram	Level	-1.444	11	0.561	-1.182	17	0.681
	Diff	-4.606	10	< 0.01	-9.438	17	< 0.01
Moong	Level	-0.606	11	0.87	-0.507	17	0.891
	Diff	-4.606	10	< 0.01	-12.093	17	< 0.01
Urad	Level	-1.432	11	0.567	-1.072	17	0.726
	Diff	-4.013	10	< 0.01	-9.764	17	< 0.01

Table A3
ARCH LM Test for ARIMA residuals.

		ARCH LM test		
		Residual	Statistic	P-Value
Gram	ARIMA(4,1,3)	upto 10 lag	129.760	< 0.01
		upto 15 lag	128.780	<0.01
		upto 20 lag	127.990	<0.01
Moong	ARIMA(1,1,0)	upto 10 lag	56.224	< 0.01
		upto 15 lag	60.301	<0.01
		upto 20 lag	60.764	<0.01
Urad	ARIMA(2,1,0)	upto 10 lag	19.123	0.038
		upto 15 lag	29.438	<0.01
		upto 20 lag	40.922	<0.01

Table A4
AIC and BIC value comparison of ARIMA and ARIMA-GARCH.

Series	Model	AIC	BIC
Gram	ARIMA(4,1,3)	1863.927	1897.585
	ARIMA(4,1,3)-GARCH(1,1)	1574.950	1589.910
Moong	ARIMA(1,1,0)	1500.979	1512.198
	ARIMA(1,1,0)-GARCH(1,1)	1322.260	1337.220
Urad	ARIMA(2,1,0)	1713.005	1727.965
	ARIMA(2,1,0)-TARCH(1,1) with t dist.	1534.870	1557.310

Appendix B

```
1 acf(FinalResult_model.resid**2,nlags=12,qstat=True)
2 plot_acf(FinalResult_model.resid**2)
```

Fig. B1. (Pseudocode 1): ACF input lag estimation.

.

```
1 mylist= ['resid']
2 df1 = df[mylist][1:]
3 dataframe = DataFrame()
4 for i in range(12,0,-1):
        dataframe['t-'+str(i)] = df1.shift(i).values[:,0]
   dataframe['t'] = df1.values[:,0]
print(dataframe.head(12))
dataframe = dataframe[12:]

5 array = dataframe.values
6 X = array[:,0:-1]
7 y = array[:,-1]
8 model = RandomForestRegressor(n_estimators=100, random_state=1)
9 model.fit(X, y)
10 print(model.feature_importances_)
11 names = dataframe.columns.values[0:-1]
12 ticks = [i for i in range(len(names))]
   pyplot.bar(ticks, model.feature_importances_)
   pyplot.xticks(ticks, names)
   pyplot.xlabel('input_lag_estimation')
   pyplot.show()
```

Fig. B2. (Pseudocode 2): Random forest input lag estimation.

.

```
1 train_resid = df1.iloc[:311]
2 test_resid = df1.iloc[311:]
3 scaler = MinMaxScaler()
4 scaler.fit(train_resid)

5 scaled_train = scaler.transform(train_resid)
6 scaled_test = scaler.transform(test_resid)

# Define the generator
7 n_input = ?
# How many data will estimate the next time stand value based on the lag estimation.
8 n_features = 1
9 train_generator = TimeseriesGenerator(scaled_train,scaled_train,length=n_input,batch_size=1)

10 model = Sequential()
    model.add(LSTM("sequence_number", activation='relu', input_shape =(n_input,n_features)))
    #relu = rectified linear unit
    model.add(Dense(1))
    model.compile(optimizer='adam',loss='mse')
    model.summary()
11 plot_model(model,show_shapes=True)
12 model.fit(train_generator, epochs= 'number of epochs' )
13 myloss = model.history.history['loss']
   plt.plot(range(len(myloss)),myloss)

# Holding the prediction
14 test_prediction = []
# Last n_input point from the train set
15 first_eval_batch= scaled_train[-n_input:]
# Reshape this to the RNN format. same as TimeSeriesGenarator
16 current_batch = first_eval_batch.reshape((1,n_input,n_features))
# Prediction range
17 for i in range(len(test)):
        current_pred = model.predict(current_batch)[0]
        test_prediction.append(current_pred)
        current_batch= np.append(current_batch[:,1:],[[current_pred]],axis=1)
18 true_prediction = scaler.inverse_transform(test_prediction)
   true_prediction
```

Fig. B3. (Pseudocode 3): Constructing LSTM model after finding the input lag.

.

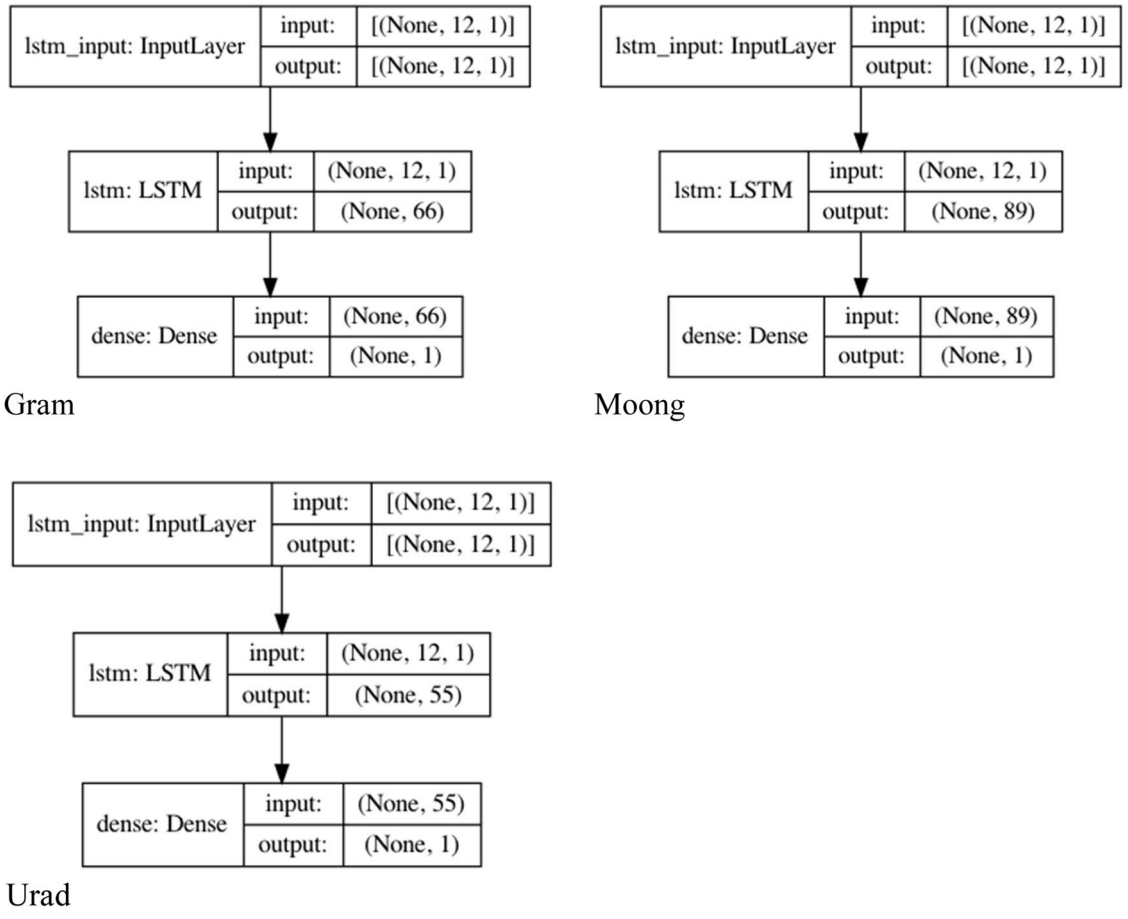


Fig. B4. LSTM architecture using normalised training data.

.

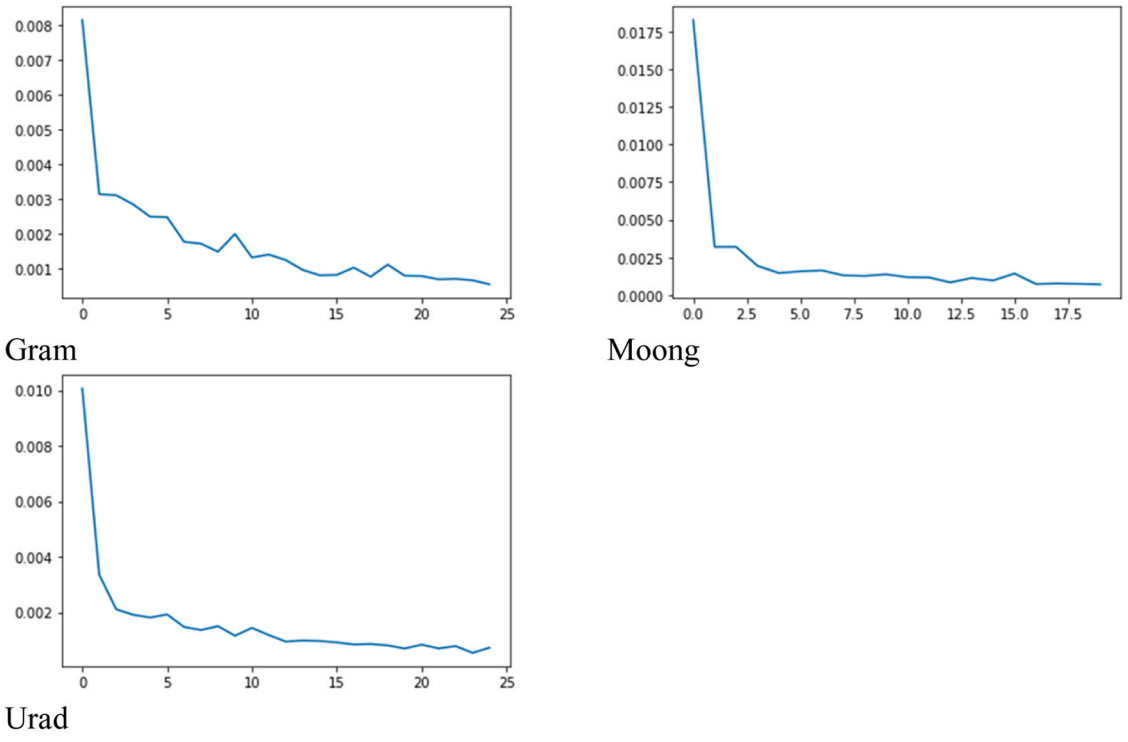


Fig. B5. Mean square error (MSE) loss function in LSTM.

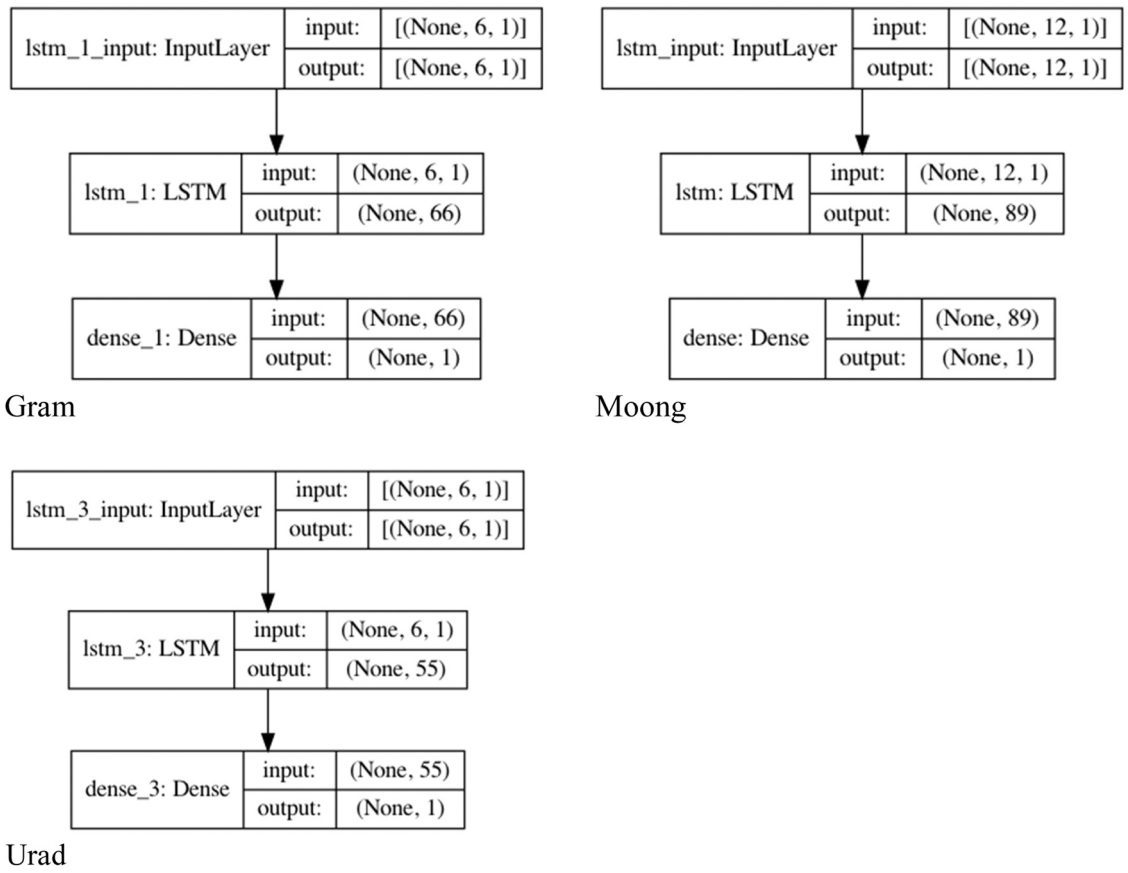


Fig. B6. LSTM architecture using normalised squared residuals obtained from ARIMA with ACF lag selection.

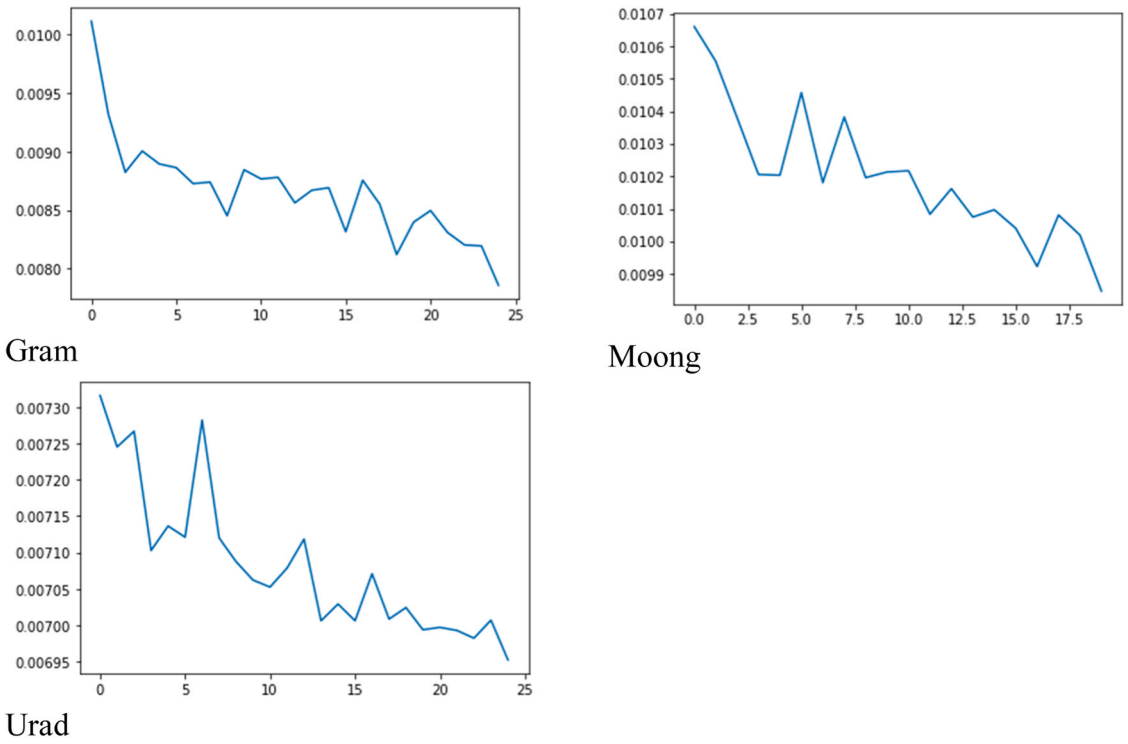


Fig. B7. Mean square error (MSE) loss function in hybrid ARIMA-LSTM with ACF lag selection.

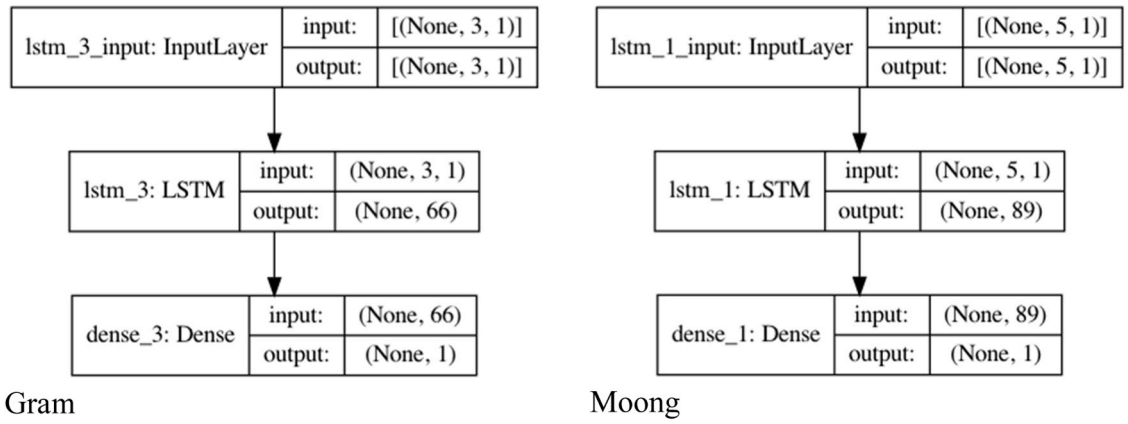


Fig. B8. LSTM architecture using normalised squared residuals obtained from ARIMA with random forest lag selection.

.

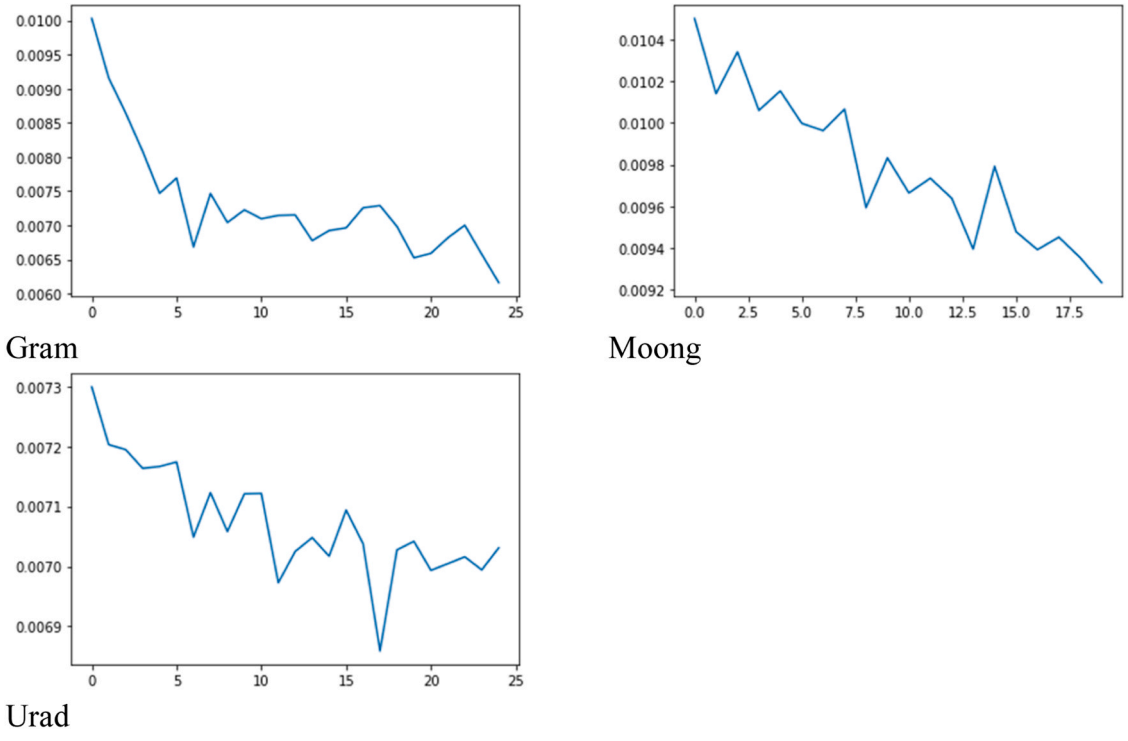


Fig. B9. Mean square error (MSE) loss function in hybrid ARIMA-LSTM with random forest lag selection.

.

References

[1] A. Lama, G.K. Jha, R.K. Paul, B. Gurung, Modelling and forecasting of price volatility: An application of GARCH and EGARCH models, Agric. Econ. Res. Rev. 28 (2015) 73–82, https://doi.org/10.5958/0974-0279.2015.00005.1.

[2] R.F. Engle, Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation, Econometrica 50 (1982) 987–1007, https://doi.org/10.2307/1912773.

[3] T. Bollerslev, Generalized autoregressive conditional Heteroskedasticity, J. Econ. 31 (1986) 307–327, https://doi.org/10.1016/0304-4076(86)90063-1.

[4] M. Epaphra, Modeling exchange rate volatility: Application of GARCH and EGARCH models, J. Mathe Financ. 7 (2017) 121–143, https://doi.org/10.4236/jmf.2017.71007.

[5] N.B. Cheikh, Y.B. Zaied, J. Chevallier, Asymmetric volatility in crytocurrency markets: New evidence from smooth transition GARCH models, Financ. Res. Lett. 35 (2019) 1–9, https://doi.org/10.1016/j.frl.2019.09.008.

[6] C. Dritsaki, The performance of hybrid ARIMA-GARCH modeling and forecasting oil price, Int. J. Energy Econ. Policy 8 (3) (2018) 14–21.

[7] S.P. Bhardwaj, R.K. Paul, D.R. Singh, K.N. Singh, An empirical investigation of arima and garch models in agricultural price forecasting, Econ. Aff. 59 (2014) 415–428, https://doi.org/10.5958/0976-4666.2014.00009.6.

[8] X. Yuan, J. Tang, W.K. Wong, S. Sriboonchitta, Modeling Co-Movement among different agricultural commodity markets: A copula-GARCH approach, Sustainability 12 (1) (2020) 17, https://doi.org/10.3390/su12010393.

[9] Y.A. Shiferaw, Time-varying correlation between agricultural commodity and energy price dynamics with bayesian multivariate DCC-GARCH models, Phys. A: Stat. Mech. its Appl. 526 (2019), https://doi.org/10.1016/j.physa.2019.04.043.

[10] L. Lin, Y. Jiang, H. Xiao, Z. Zhou, Crude oil price forecasting based on a novel hybrid long memory GARCH-M and wavelet analysis model, Phys. A: Stat. Mech. its Appl. 543 (2020), https://doi.org/10.1016/j.physa.2019.123532.

[11] R. Majid, S.A. Mir, Advances in statistical forecasting methods: An overview, Econ. Aff. 63 (2018) 815–831, https://doi.org/10.30954/0424-2513.4.2018.5.

[12] K. Funahashi, Y. Nakamura, Approximation of Dynamical systems by continuous time recurrent neural networks, Neural Netw. 6 (1993) 801–806, https://doi.org/10.1016/S0893-6080(05)80125-X.

[13] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.

[14] S.S. Namini, N. Tavakoli, A.S. Namin, A comparison of ARIMA and LSTM in forecasting time series, Pap. 17th IEEE Int. Conf. Mach. Learn. Appl. (2018), https://doi.org/10.1109/icmla.2018.00227.

[15] K. Kurumarani, Time series forecasting of agricultural product prices based on recurrent neural networks and its evaluation method, SN Appl. Sci. 2 (2020), 1434, https://doi.org/10.1007/s42452-020-03225-9.

[16] K. Om, S. Boukoros, A. Nugaliyadde, T. McGill, M. Dixon, P. Koutsakis, K.W. Wong, Modelling email traffic workload with RNN and LSTM models, Hum. -Centr Comput. Inf. Sci. 10 (1) (2020) 16, https://doi.org/10.1186/s13673-020-00242-w.

[17] T.W. Yoo, I.I.S. Oh, Time series forecasting of agricultural products sales volumes based on seasonal long short-term memory, Appl. Sci. 10 (2020) 1–15, https://doi.org/10.3390/app10228169.

[18] D. Yi, S. Bu, I. Kim, An enhanced algorithm of RNN using trend in time series, Symmetry 11 (2019) 1–14, https://doi.org/10.3390/sym11070912.

[19] K. Zhou, W. Wang, T. Hu, K. Deng, Time series forecasting and classification models based on recurrent with attention mechanism and generative adversarial networks, Sensors 20 (2020) 1–20, https://doi.org/10.3390/s20247211.

[20] H. Tsangari, An alternative methodology for combining different forecast models, J. Appl. Stat. 34 (2007) 403–421, https://doi.org/10.1080/02664760701231633.

[21] C. Goutte, Lag space estimation in time series modelling, 1997 IEEE International Conference on Acoustics, Speech, Signal Process. (1997), https://doi.org/10.1109/ICASSP.1997.595502.

[22] J.D. Scargle, Bayesian estimation of time series lags and structure, AIP Conf. Proc. 617 (2002) 23, https://doi.org/10.1063/1.1477036.

[23] C. Han, P.C.B. Phillips, D. Sul, Lag length selection using panel autoregression, Econom. Rev. 36 (2016) 225–240, https://doi.org/10.1080/07474938.2015.1114313.

[24] P. Cortez, Sensitivity analysis for time lag selection to forecast seasonal time series using neural networks and support vector machines, 2010 Int. Jt. Conf. Neural Netw. (IJCNN) (2010), https://doi.org/10.1109/IJCNN.2010.5596890.

[25] O. Surakhi, M.A. Zaidan, P.L. Fung, N. Hossein Motlagh, S. Serhan, M. AlKhanafseh, R.M. Ghoniem, T. Hussein, Time-lag selection for time-series forecasting using neural network and heuristic algorithm, Electronics 10 (2021) 2518, https://doi.org/10.3390/electronics10202518.

[26] M.J. Kane, N. Price, M. Scotch, et al., Comparison of ARIMA and Random Forest time series models for prediction of avian influenza H5N1 outbreaks, BMC Bioinforma. 15 (2014), 276, https://doi.org/10.1186/1471-2105-15-276.

[27] M. Markiewicz, A. Wyłomańska, Time series forecasting: problem of heavy-tailed distributed noise, Int J. Adv. Eng. Sci. Appl. Math. 13 (2021) 248–256, https://doi.org/10.1007/s12572-021-00312-x.

[28] Z. Gao, Y. He, E.E. Kuruoglu, A Hybrid Model Integrating LSTM and Garch for Bitcoin Price Prediction, 2021 IEEE 31st Int. Workshop Mach. Learn. Signal Process. (MLSP) (2021) 1–13, https://doi.org/10.1109/MLSP52302.2021.9596429.

[29] W. Li, D.M. Becker, Day-ahead electricity price prediction applying hybrid models of LSTM-based deep learning methods and feature selection algorithms under consideration of market coupling, Energy 237 (2021), 121543, https://doi.org/10.1016/j.energy.2021.121543.

[30] K. Phurun, C. Kasemset, Shallot Price Forecasting Model Using Hybrid ARIMA-LSTM Model, Data Sci. Eng. (DSE) Rec. 3 (1) (2020) 35–43.

[31] S. Kulshreshtha, An ARIMA-LSTM hybrid model for stock market prediction using live data, J. Eng. Sci. Technol. Rev. 13 (4) (2020).

[32] H.Y. Kim, C.H. Won, 'Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models', Expert Syst. Appl. 103 (2018) 25–37, https://doi.org/10.1016/j.eswa.2018.03.002.

[33] M. Srivastava, A. Rao, J.S. Parihar, S. Chavriya, S. Singh, What do the AI methods tell us about predicting price volatility of key natural resources: Evidence from hyperparameter tuning, Resour. Policy 80 (2023), 103249, https://doi.org/10.1016/j.resourpol.2022.103249.

[34] H. Zeng, B. Shao, H. Dai, Y. Yan, N. Tian, Prediction of fluctuation loads based on GARCH family-CatBoost-CNNLSTM, Energy 263 (2023), 126125, https://doi.org/10.1016/j.energy.2022.126125.

[35] Y. Hu, J. Ni, L. Wen, A hybrid deep learning approach by integrating LSTM-ANN networks with GARCH model for copper price volatility prediction, Phys. A: Stat. Mech. its Appl. 557 (2020), 124907, https://doi.org/10.1016/j.physa.2020.124907.

[36] G.E. Box, G.M. Jenkins, 1976, Time series analysis: forecasting and control. vol 2nd Edition. John Wiley & Sons, Holden-Day, San Francisco, USA.

[37] S. Ray, S.S. Das, P. Mishra, A.M.G. Al-Khatib, Time series SARIMA modelling and forecasting of monthly rainfall and temperature in the South Asian countries, Earth Syst. Environ. 5 (2021) 531–546, https://doi.org/10.1007/s41748-021-00205-w.

[38] S. Ray, B. Bhattacharyya, S. Pal, Statistical modeling and forecasting of food grain in effects on public distribution system: an application of ARIMA model, Indian J. Econ. Dev. 12 (2016) 739–744, https://doi.org/10.5958/2322-0430.2016.00199.2.

[39] S.J. Taylor, 2007, Modeling financial time series. 2nd edition. Wiley, New York. https://doi.org/10.1142/6578.

[40] E. Ndikumana, D.H.T. Minh, N. Baghadadi, D. Courault, L. Hossard, Deep recurrent neural network for agricultural classification using multitemporal SAR sentinel-1 for camargue, France, Remote Sens. 10 (8) (2018) 1217, https://doi.org/10.3390/rs10081217.

[41] F. Grubbs, Procedures for detecting outlying observation in samples, Technometrics 11 (1969) 1–21.

[42] W. Stefansky, Rejecting outliers in factorial design, Technometrics 14 (1972) 469–479.

[43] D.A. Dickey, W.A. Fuller, Distribution of the estimators for autoregressive time series with a unit root, J. Am. Stat. Assoc. 74 (1979) 427–431, https://doi.org/10.2307/2286348.

[44] P.C. Phillips, P. Perron, Testing for a unit root in time series regression, Biometrika 75 (1988) 335–346, https://doi.org/10.2307/2336182.

[45] G.M. Ljung, G.E. Box, On a measure of lack of fit in time series models, Biometrika 65 (1978) 297–303, https://doi.org/10.1093/biomet/65.2.297.

[46] H. Akaike, A new look at the statistical model identification, IEEE Trans. Autom. Control 19 (1974) 716–723, https://doi.org/10.1109/TAC.1974.1100705.

[47] P. Mishra, A. Matuka, M.S.A. Abotaleb, et al., Modeling and forecasting of milk production in the SAARC countries and China, Model. earth Syst. Environ. (2021), https://doi.org/10.1007/s40808-021-01138-z.

[48] G. Schwarz, Estimating the dimension of a model, Ann. Stat. 6 (1978) 461–464, https://doi.org/10.1214/aos/1176344136.

[49] D.P. Kingma, J. Ba, 2014, Adam: A method for stochastic optimization. 3rd International Conference for Learning Representations, San Diego. https://doi.org/10.48550/ARXIV.1412.6980, https://arxiv.org/abs/1412.6980.

[50] F.X. Diebold, R.S. Mariano, Comparing predicting accuracy, J. Bus. Econ. Stat. 13 (1995) 253–265.

[51] H. Peng, F. Long, C. Ding, Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy, IEEE Trans. Pattern Anal. Mach. Intell. 27 (2005) 1226–1238. ⟨https://ieeexplore.ieee.org/document/1453511⟩.

[52] X.V. Nguyen, J. Chan, S. Romano, J. Bailey, Effective global approaches for mutual information based feature selection, KDD'14 (2014) 512–521, https://doi.org/10.1145/2623330.2623611.