Applied Intelligence (2021) 52:11489–11505
https://doi.org/10.1007/s10489-022-03251-7

A hybrid extreme learning machine model with harris hawks optimisation algorithm: an optimised model for product demand forecasting applications

Koushiki Dasgupta Chaudhuri[1] · Bugra Alkan[2]

Accepted: 15 January 2022 / Published online: 26 January 2022

Abstract

Accurate and real-time product demand forecasting is the need of the hour in the world of supply chain management. Predicting future product demand from historical sales data is a highly non-linear problem, subject to various external and environmental factors. In this work, we propose an optimised forecasting model - an extreme learning machine (ELM) model coupled with the Harris Hawks optimisation (HHO) algorithm to forecast product demand in an e-commerce company. ELM is preferred over traditional neural networks mainly due to its fast computational speed, which allows efficient demand forecasting in real-time. Our ELM-HHO model performed significantly better than ARIMA models that are commonly used in industries to forecast product demand. The performance of the proposed ELM-HHO model was also compared with traditional ELM, ELM auto-tuned using Bayesian Optimisation (ELM-BO), Gated Recurrent Unit (GRU) based recurrent neural network and Long Short Term Memory (LSTM) recurrent neural network models. Different performance metrics, i.e., Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE) and Mean Percentage Error (MPE) were used for the comparison of the selected models. Horizon forecasting at 3 days and 7 days ahead was also performed using the proposed approach. The results revealed that the proposed approach is superior to traditional product demand forecasting models in terms of prediction accuracy and it can be applied in real-time to predict future product demand based on the previous week's sales data. In particular, considering RMSE of forecasting, the proposed ELM-HHO model performed 62.73% better than the statistical ARIMA(7,1,0) model, 40.73% better than the neural network based GRU model, 34.05% better than the neural network based LSTM model, 27.16% better than the traditional non-optimised ELM model with 100 hidden nodes and 11.63% better than the ELM-BO model in forecasting product demand for future 3 months. The novelty of the proposed approach lies in the way the fast computational speed of ELMs has been combined with the accuracy gained by tuning hyperparameters using HHO. An increased number of hyperparameters has been optimised in our methodology compared to available models. The majority of approaches to improve the accuracy of ELM so far have only focused on tuning the weights and the biases of the hidden layer. In our hybrid model, we tune the number of hidden nodes, the number of input time lags and even the type of activation function used in the hidden layer in addition to tuning the weights and the biases. This has resulted in a significant increase in accuracy over previous methods. Our work presents an original way of performing product demand forecasting in real-time in industry with highly accurate results which are much better than pre-existing demand forecasting models.

Keywords Demand forecasting · Supply chain management · Extreme learning machines · Artificial neural networks · Optimisation · Harris hawks optimisation · Hyperparameter tuning · ARIMA

1 Introduction

In today's highly competitive manufacturing environment, product demand forecasting plays a critical role in the effective management of inventories. Companies that can accurately forecast market demand can take action to ensure that they hold the correct stocks to maximise sales and profit.

✉ Bugra Alkan
alkanb@lsbu.ac.uk

Extended author information available on the last page of the article

Demand forecasting refers to the process of making estimations about future customer demand over a defined period using historical demand data and other external factors [1]. Efficient and accurate forecasting of demand enables supply chain managers to make intelligent decisions about various aspects of their business. It helps them judge their business potential in the current scenario, optimise inventory, improve inventory turnover rates and decrease holding costs. Estimating future demand beforehand helps to identify possible upcoming difficulties and take steps to correct them such as increasing the number of workers during high sales periods. It allows companies to plan their future goals in a better way [2]. Hence, finding efficient and effective product demand forecasting methods is highly important right now. With better data acquisition and storage tools, a huge amount of historical sales data is available in all fields. Data mining approaches are needed to model and make sense of this data to make accurate future predictions about product demand.

Neural network models have been effectively used to capture and model non-linearity in product demand forecasting in the past. However, traditional backpropagated neural networks suffer from several drawbacks like slow computational speed, over-fitting, etc. ELMs have been shown to be more advantageous in efficiency and generalization performance over traditional FNN algorithms on a wide range of benchmark problems from different fields [3, 4]. Moreover, ELMs have also been shown to have universal approximation capabilities [5]. In this work, a novel approach to product demand forecasting - a hybrid extreme learning machine (ELM) model with Harris Hawks optimisation (HHO) algorithm is proposed. The proposed methodology retains the fast computational speed of ELMs while enhancing their ability to correctly predict product demands. The prediction capabilities of ELM are enhanced by auto-tuning its hyperparameters (number of hidden nodes, number of lags in input data, weights and biases of the hidden layer and the type of activation function) by the HHO algorithm. The superiority of the ELM-HHO approach was tested in comparison with traditional ELM and ELM-Bayesian Optimisation (ELM-BO) approaches. The ELM-HHO algorithm was also compared with GRU, LSTM and ARIMA models commonly used in industry to forecast product demand. Different performance metrics, i.e., Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE) and Mean Percentage Error (MPE) were used for model comparison. Moreover, horizon forecasting at 3 days ahead and 7 days ahead was performed to test how the ELM-HHO approach can be applied in real-time to predict future demand from the previous week's sales data. The findings indicated that the suggested model outperformed the traditional product demand prediction models in terms of prediction accuracy, making it a strong candidate for real-time product demand forecasting applications. To our knowledge, this is the first time extreme learning machine models optimised by Harris Hawks optimiser have been applied to product demand forecasting contexts. The novelty lies in the way the ELM and HHO approaches have been combined to ensure highly accurate product demand forecasting. Also, an increased number of hyperparameters are being optimised compared to traditional approaches to improve accuracy.

To reiterate, the main contributions of this paper involve :

1. Proposing a novel hybrid ELM optimised by HHO algorithm to perform product demand forecasting.
2. Optimising an increased number of hyperparameters in ELM to improve accuracy.
3. Combining the fast computational speed of ELMs with the accuracy gained by optimisation using HHO.
4. Showing that the ELM-HHO model has superior performance to state-of-the-art models used in the industry like ELM, ELM-BO, LSTM, GRU and ARIMA models considering RMSE of forecasting.
5. Performing horizon forecasting at 3 days ahead and 7 days ahead with good accuracy which confirms that our approach is useful for the industry.

Future avenues of research lie in trying to improve the optimisation part by exploring multi-objective optimisation or trying out newer optimization algorithms like that mentioned in [6]. Multi-variate demand forecasting can also be performed in cases where inputs other than historical sales data are available. Auto-encoders for automatic relevant feature extraction from input time series can be investigated.

The rest of the paper is organised as follows. Section 2 discusses the research background, related works and research gaps in the product demand forecasting area. Section 3 provides a brief introduction to some preliminaries required to understand the proposed model. Section 4 presents the proposed ELM-HHO approach. Section 5 describes the historical dataset used, the experimental setup, experimental results of the proposed model and comparison with the ARIMA, GRU, LSTM, ELM and ELM-BO approaches. Finally, in Section 6, the paper is concluded and future work is outlined.

2 Literature review

Supply chain management is the management of the flow of goods and services and includes all processes that transform raw materials into final products [7]. It involves the active streamlining of a business's supply-side activities to maximise customer value and gain a competitive advantage in the marketplace [8]. Demand forecasting forms an essential component of the supply chain process. The role of demand forecasting in supply chain management involves strategic business planning. Demand forecasting also sets in motion the push processes of supply chain management such as raw material planning, purchasing and logistics. It steers the pull processes including order management and distribution. While demand forecasting is undeniably important, it is also one of the most difficult aspects of supply chain planning. Demand is often volatile which makes demand forecasting both an art and a science. More than 74% of the respondents in a research survey indicated poor forecasting accuracy and demand volatility as the major challenges to supply chain flexibility [9]. Supply chain managers who learn from past experience can utilise demand forecasting to set better business goals and optimise returns [10].

An extensive literature survey has revealed that there are two main types of methodologies that are applied in the domain of product demand forecasting problems [11]. These are: *i*) stand-alone forecasting models and *ii*) hybrid forecasting models combining multiple models together. The standalone models can be statistical models or machine learning models. The most commonly used statistical models are simple moving average models [12], autoregressive integrated moving average (ARIMA) and autoregressive integrated moving average with explanatory variables models (ARIMAX). ARIMA models are used to forecast demand data from historical time series data, as in [13]. While ARIMA is a univariate method, ARIMAX uses multiple variables to incorporate external data (e.g., environmental factors) in addition to historical demand data to forecast demand [14]. A lot of regression models have also been used to predict demand, such as: Bayesian Linear Regression [15], Boosted Decision Tree Regression [15], Decision Forest Regression [15], Multiple Regression [12] and Symbolic Regression with Genetic Programming [12]. Recently, quite a few research projects using Artificial Neural Network (ANN) models were carried out to forecast product demand. ANN models are especially effective in forecasting demand when the time series data is non-linear. The properties of ANNs, such as self-learning ability, ability to perform problems with loss of data and ability to operate on multi-task problems, have made them especially useful in demand forecasting. An Integrated Case-Based Reasoning and ANN model was successfully implemented to predict product unit cost for mobile phone companies in Taiwan [16]. A similar study proposed by Islek et al. [11] successfully merged various models, i.e., multi-layer perceptrons with Bayesian networks, linear regression and support vector machines to predict product demands in a real data set derived from a dried fruits and nuts company from Turkey.

Despite the growing applications and usefulness of neural network models in forecasting demand, the forecasts produced by these methods still suffer from several drawbacks such as over-fitting [17], slow learning speed [18], getting stuck at local minima etc. An emerging algorithm for Single Hidden Layer Feed-forward Networks (SLFNs) - the ELM model was proposed in 2004[19]. It overcomes many of the disadvantages of traditional neural network models. ELM models are being used for the past couple of years to forecast a variety of things. A modified teaching-learning based optimised ELM model was used to forecast solar power generation of Chattisgarh state, India [20]. An ELM model was used to forecast monthly stream-flow discharge rates in Tigris river, Iraq [21]. A hybrid model was developed for probabilistic forecasting of wind power generation using ELM and the pairs bootstrap method [22]. In a similar work, forecasting of photovoltaic power was performed using ELM [23]. Various optimisation algorithms have also been employed with ELM to improve its prediction accuracy, though not in a product demand forecasting context. A genetic algorithm optimised ELM model was proposed to optimise gear-blank preforms [24]. In another work, the gravity field of irregular small bodies was modelled using ELM and BO [25]. A number of variants of ELM have also come out, each with its own unique set of advantages. The Optimally Pruned ELM [26], for example, ranks the best-hidden neurons by multi-response sparse regression and then selects the optimal number of neurons using a leave-one-out validation method. The online sequential ELM or OS-ELM [27] can learn data one-by-one or chunk-by-chunk (a block of data) with fixed or varying chunk size, which makes it faster than other sequential algorithms. In voting based extreme learning machine [28], multiple independent ELM trainings are performed instead of a single ELM training and then the final decision is made based on the majority voting method. A detailed review of trends and variants of ELM can be found in [29].

The major issue with time-series approaches to demand forecasting is that a large number of observations is required to determine the best fit model for a time series [13]. Historical data is needed for such methods. In the beginning, there is no information about the past and estimation has to be made based on similar cases. This results in a large amount of uncertainty which can be avoided with time [13]. ARIMA models can only be applied to stationary time series, it fails to capture seasonality. Seasonal factors affecting demand like holidays, festivals, weather conditions and other environmental exogenous factors often influence demand and are not adequately represented in forecasting models. While neural networks help in capturing the non-linearity in data due to external factors, they suffer from certain problems like over-fitting, slow learning speed etc. ELMs help to overcome these difficulties as they have extremely fast learning speed, good generalisation performance and least human intervention [29].

3 Preliminaries

This section outlines some background knowledge required to understand the hybrid ELM-HHO approach proposed in this paper, namely the basics of extreme learning machines and the Harris Hawks optimisation algorithm. In addition, some other common state-of-the-art demand forecasting models are introduced whose results are compared with our proposed model's results later in the paper.

3.1 Extreme learning machine (ELM)

A single hidden-layer feed-forward neural network (ELM) with fast learning speed was put forward by Huang et al. [19] in 2004. The method does not require iterative tuning to reach a global minima [4]. It has been shown to have good generalisation performance, universal approximation capabilities [5], high learning accuracy and least human intervention [29]. The weights and biases of the hidden layer nodes are set randomly, thus saving time required in hyper-tuning by back-propagation. The weights of the output layer can be determined analytically by a Moore Penrose inverse operation. A generic ELM model is given in Fig. 1.

The output function of the ELM (a SLFN with L hidden nodes) can be represented as follows:

$$f_L(x) = \sum_{i=1}^{L} \beta_i h_i(x) = \sum_{i=1}^{L} \beta_i h(w_i \times x_j + b_i), j = 1, .., N \quad (1)$$

where N is the number of training samples, w_i and b_i are the weight and the bias of the i^{th} hidden node respectively, x_j is an input vector, h is an activation function (e.g., Relu, sigmoid, Gaussian etc.) and β_i is the weight vector between the i^{th} hidden node and the output. Equation (1) can be written as:

$$T = H\beta \quad (2)$$

where,

$$H = \begin{bmatrix} g(w_1 \times x_1 + b_1) & \dots & g(w_L \times x_1 + b_L) \\ \vdots & \dots & \vdots \\ g(w_1 \times x_N + b_1) & \dots & g(w_L \times x_N + b_L) \end{bmatrix} \quad (3)$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_L^T \end{bmatrix} \quad (4)$$

$$T = \begin{bmatrix} T_1^T \\ \vdots \\ T_N^T \end{bmatrix} \quad (5)$$

H is called the hidden layer output matrix and T is the training data matrix. To minimise the cost function ||Hβ - T||, the ELM theory claims that the parameters of the hidden nodes w_i and b_i can be assigned randomly without considering the input data. Since H is invertible, the output weights β can be calculated as follows.

$$\hat{\beta} = H^{\dagger} T \quad (6)$$

where $H^{\dagger}$ is the Moore-Penrose generalised inverse of H.

The main steps of the ELM algorithm are as follows: *i*) weights and biases of the hidden layer (w_i and b_i) are randomly assigned, i = 1,...,L, *ii*) the hidden layer output matrix H is calculated according to (3), *iii*) The output weight matrix $\hat{\beta}$ is calculated by $\hat{\beta} = H^{\dagger}T$, *iv*) $\hat{\beta}$ is used to make predictions on new test data $F = H\hat{\beta}$.

3.2 Harris hawks optimisation (HHO) algorithm

In 2019, Heidari et al. [30] proposed a novel nature-inspired metaheuristic optimisation algorithm - the Harris Hawks Optimiser. It is based on the cooperative predatory strategy of Harris hawks, one of the most intelligent birds in nature. A group of hawks living in the same family or the same stable group collaboratively attack a prey (usually a small rabbit) from different directions in a technique called "*surprise pounce*" or "*seven kills*". The way of pursuing may differ depending on the dynamically changing circumstances and the escape strategy of the prey. The HHO algorithm models this behaviour in two broad phases - an exploration and an exploitation phase. It is a population-based algorithm similar to algorithms such as Particle Swarm Optimisation (PSO) and Ant Colony Optimisation (ACO). Since its development, the HHO algorithm has been used in a variety of fields ranging from design and manufacturing [31], feature selection for data mining [32] and even for optimising deep learning models [33].

In the HHO algorithm, the Harris hawks are considered as candidate solutions, and in each iteration, the fitness value is computed for each solution based on the intended prey. The steps of the HHO algorithm are given in (Fig. 2) and explained as follows:

3.2.1 Exploration phase

In this phase, the initial iteration scheme for the solution is decided based on two perching strategies of Harris hawks :

$$X(t+1) = \begin{cases} X_{rand}(t) - r_1 |X_{rand}(t) - 2r_2 X(t)| & q \geq 0.5 \\ (X_{rabbit}(t) - X_m(t)) - C & q < 0.5 \end{cases} \quad (7)$$

$$C = r_3(LB + r_4(UB - LB)) \quad (8)$$

where $X_{rabbit}(t)$ is the current position of the rabbit, $X(t)$ is the current position vector of the hawks and $X(t+1)$ is their

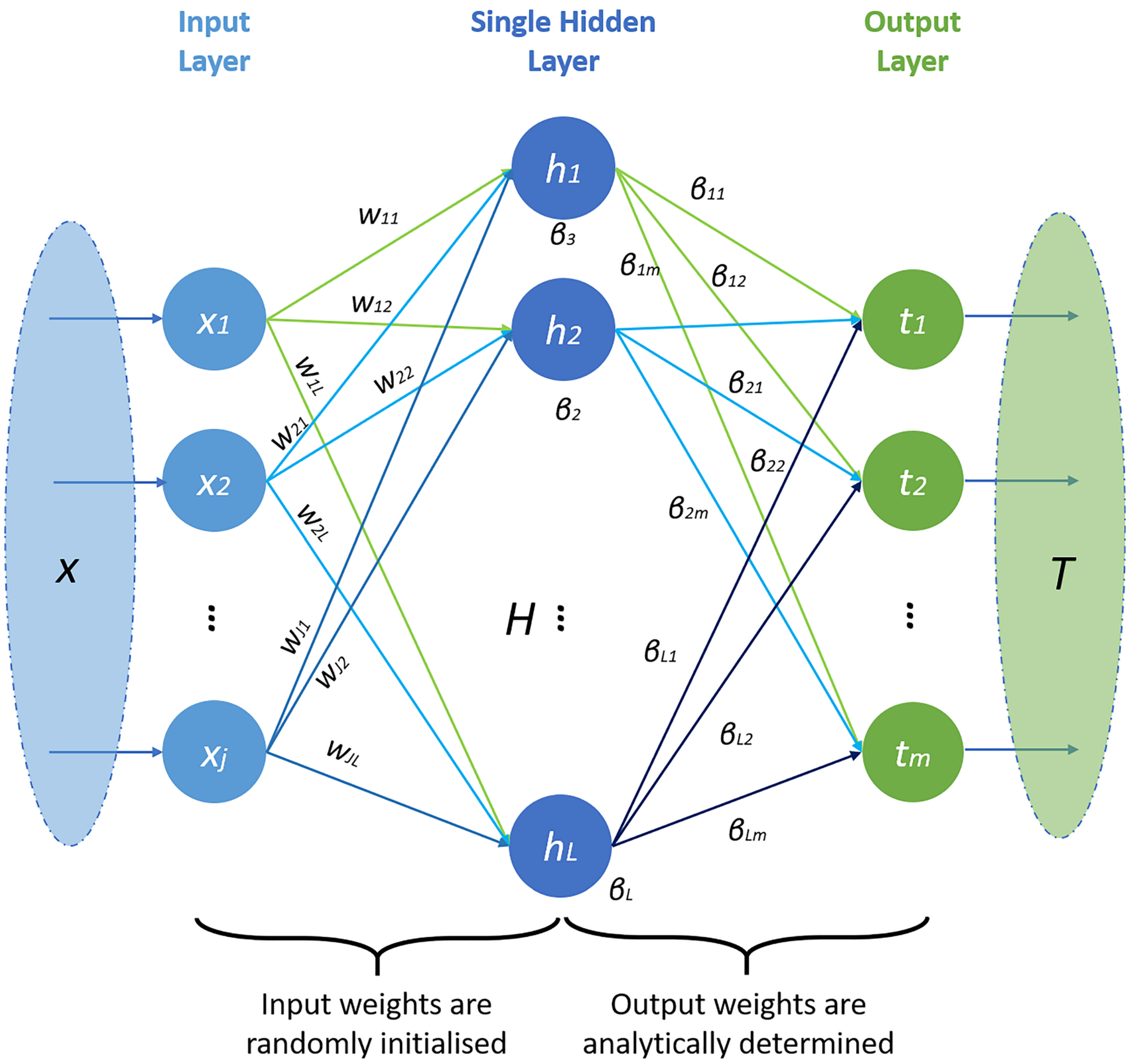

Fig. 1 The architecture of the single hidden layer ELM model

position vector in the next iteration, $X_{rand}(t)$ is a randomly selected hawk's position from the current population, $X_m(t)$ is the average position of the current population of hawks, r_1, r_2, r_3, r_4 and q are random numbers inside (0,1) which are updated in every iteration and LB and UB are the lower and upper bounds of variables respectively. $X_m(t)$ is obtained as follows:

$$X_m(t) = \frac{1}{N}\sum_{i=1}^{N} X_i(t) \tag{9}$$

where $X_i(t)$ is the location of the i^{th} hawk in the current iteration and N is the total number of hawks in the population.

3.2.2 Transition from exploration to exploitation

The energy of the prey or rabbit is modeled in this step :

$$E = 2E_0\left(1 - \frac{t}{T}\right) \tag{10}$$

where E indicates the escaping energy of the prey, E_0 is its initial energy and T is the maximum number of iterations.

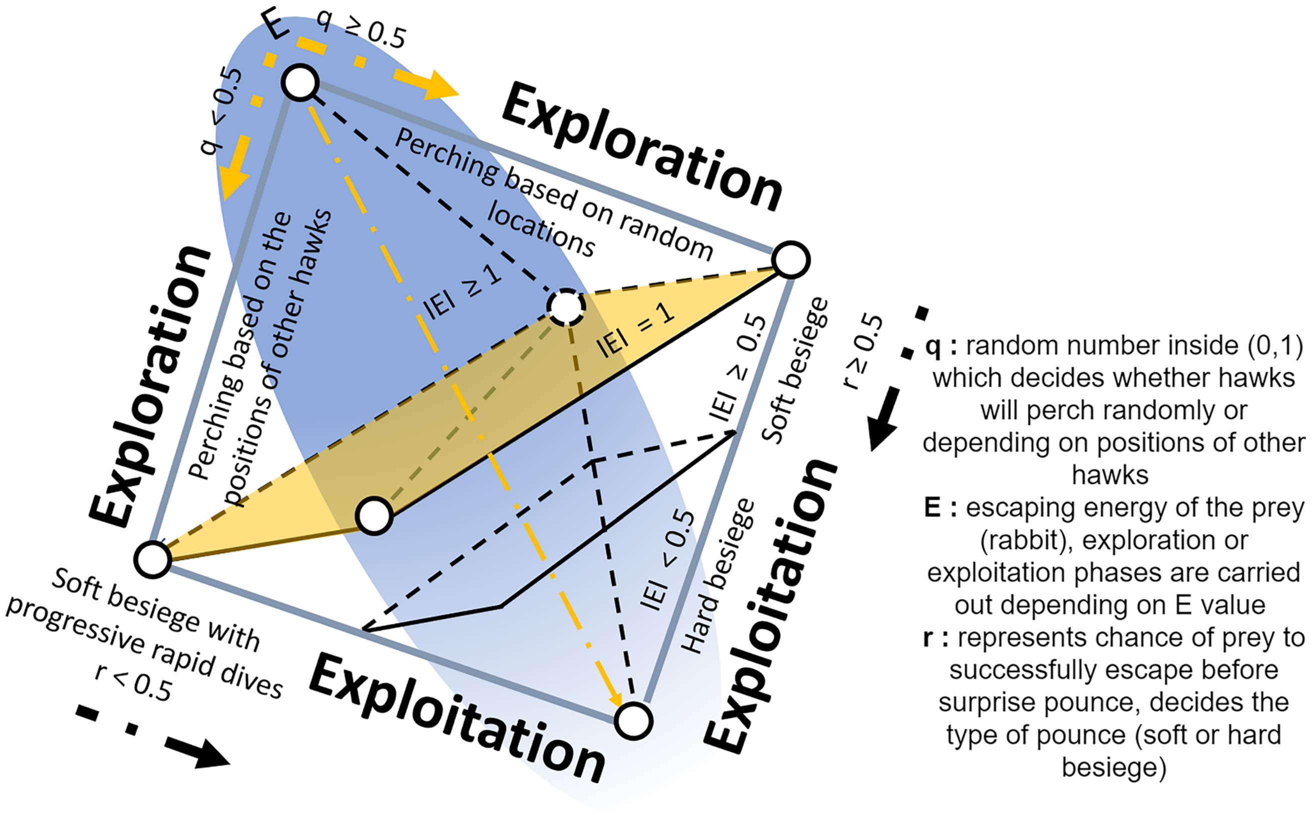
E
q ≥ 0.5
q < 0.5
Exploration
Perching based on random locations
Exploration
Perching based on the positions of other hawks
|E| ≥ 1
|E| = 1
|E| ≥ 0.5
Soft besiege
r ≥ 0.5
|E| < 0.5
Hard besiege
Exploitation
Soft besiege with progressive rapid dives
r < 0.5
Exploitation
q : random number inside (0,1) which decides whether hawks will perch randomly or depending on positions of other hawks
E : escaping energy of the prey (rabbit), exploration or exploitation phases are carried out depending on E value
r : represents chance of prey to successfully escape before surprise pounce, decides the type of pounce (soft or hard besiege)

Fig. 2 The steps of the HHO algorithm (adopted from [30])

3.2.3 Exploitation phase

Exploitation phase of the HHO algorithm includes the following steps :

- **Soft besiege:** This is the phase when the prey/rabbit still has enough energy and tries to escape by some random misleading jumps but finally fails ($r \geq 0.5$ and $|E| \geq 0.5$). The hawks encircle the rabbit softly to make it more exhausted before performing the surprise pounce. This behaviour is modelled by :

$$X(t+1) = \Delta X(t) - E|JX_{rabbit}(t) - X(t)| \tag{11}$$

$$\Delta X(t) = X_{rabbit}(t) - X(t) \tag{12}$$

$$J = 2(1 - r_5) \tag{13}$$

where r_5 is a random number from (0,1), $\Delta X(t)$ is the difference between the positions of the rabbit and the hawks in the t^th iteration and J indicates the rabbit's strength of jumping while it is trying to escape.

- **Hard besiege** This is the phase when the prey is very exhausted and has a low escaping energy.($r \geq 0.5$ and $|E| < 0.5$)

$$X(t+1) = X_{rabbit}(t) - E\|\Delta X(t)\| \tag{14}$$

- **Soft besiege with progressive rapid dives** When $r < 0.5$ but $|E| \geq 0.5$, a soft besiege is carried out before the *surprise pounce* although the rabbit still has sufficient energy to escape.

$$Y = X_{rabbit}(t) - E|JX_{rabbit}(t) - X(t)| \tag{15}$$

The hawks dive based on Levy flight-based patterns according to the following rule :

$$Z = Y + S \times LF(D) \tag{16}$$

where D is the dimension of the problem, S is a random vector of size 1 × D and LF(D) is the Levy flight function which is calculated using:

$$LF(x) = 0.01 \times \frac{u \times \sigma}{|v^{\frac{1}{\beta}}|} \tag{17}$$

$$\sigma = \left(\frac{\Gamma(1+\beta) \times sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times 2^{\left(\frac{\beta-1}{2}\right)}} \right)^{\frac{1}{\beta}} \tag{18}$$

where u, v are random variables inside (0,1), β is a constant = 1.5. So the final rule for updating the position of the hawks in this phase is :

$$X(t+1) = \begin{cases} Y \text{ if } LF(Y) < LF(X(t)) \\ Z \text{ if } LF(Z) < LF(X(t)) \end{cases} \tag{19}$$

where Y and Z are as calculated earlier in (15) and (16).

- **Hard besiege with progressive rapid dives** When $r < 0.5$ and $|E| < 0.5$, a hard besiege is performed before the *surprise pounce* as the rabbit does not have sufficient energy left to escape.

$$X(t+1) = \begin{cases} Y \text{ if } LF(Y) < LF(X(t)) \\ Z \text{ if } LF(Z) < LF(X(t)) \end{cases} \tag{20}$$

where Y and Z are calculated using new rules :

$$Y = X_{rabbit}(t) - E|JX_{rabbit}(t) - X_m(t)| \tag{21}$$

$$Z = Y + S \times LF(D) \tag{22}$$

3.3 Other methods utilised in the comparison study

There are many state-of-the-art-models which are utilised for product demand forecasting applications in the industry. We have used some of them in this paper to compare the results of our ELM-HHO approach with esixting approaches. This subsection provides an introduction to some of these comparative models we have used in our study.

3.3.1 ELM-BO

ELM-BO is a variant of ELM where the hyperparameters are optimised by Bayesian Optimisation (BO). BO uses Bayes Theorem to direct an efficient and effective search of a global optimisation problem. It constructs a surrogate function to represent the objective function probabilistically. The surrogate function is then explored with an acquisition function to choose potential solutions which are judged on the original objective function. BO is a popular choice of optimisation algorithm to tune hyperparameters in machine learning [34]. The same methodology is followed for building the ELM-BO model as has been described in Section 4 for the ELM-HHO forecasting model. ELM optimised by BO has been used in the past to model the gravity field of irregular small bodies [25].

3.3.2 Autoregressive integrated moving average (ARIMA)

An ARIMA model is a statistical model which is used frequently in the industry for time series forecasting. It is a generalisation of the simpler Autoregressive moving average or ARMA model and adds to it the notion of integration. The main features of an ARIMA (p,d,q) model are:

- Autoregression (AR): It uses the dependent relation between an observation and p number of lagged observations.
- Integrated (I): It differences (subtracts an observation from an observation at the previous time step) the raw time series data d times to make it stationary.
- Moving Average (MA): It uses the dependency between an observation and a residual error from a moving average model applied to q number of lagged observations.

ARIMA models are perhaps the most common models used in industry for time series forecasting [13, 14].

3.3.3 Long short term memory (LSTM)

Recurrent neural networks (RNNs) are a form of Artificial Neural Networks that can memorise arbitrary-length sequences of input patterns by capturing connections between sequential data types. However, owing to the failure of stochastic gradients, RNNs are unable to detect long-term dependencies in lengthy sequences [35]. To address this issue, a number of novel RNN models, notably LSTM, have been suggested. LSTM networks are RNN extensions designed to learn sequential (temporal) data and their long-term connections more precisely than standard RNNs. LSTM models are commonly used in deep learning applications such as stock forecasting, speech recognition, natural language processing and so forth. Some examples of LSTM in time-series forecasting can be found in [36, 37]

3.3.4 Gated recurrent unit (GRU)

A GRU is a RNN architectural variation that employs gating algorithms to regulate information flow among neural network cells. GRUs were initially developed in [38]. They enable the collection of relationships from massive data sequences in a flexible manner without losing information from previous portions of the sequence. This is accomplished through its gating units. These units govern whether or not information should be preserved or deleted at each time step. Some examples of GRU in time-series forecasting applications can be found in [36, 37]

4 Proposed ELM-HHO forecasting model

The main aim of this study is to obtain an accurate model for demand forecasting in an e-commerce company. The proposed methodology focuses on the optimisation of hyperparameters of ELM network using a single-objective

Harris hawks optimisation (HHO) algorithm. The model combines the fast computational speed of ELMs with the accuracy gained by optimising hyperparameters using HHO. HHO, being a population-based metaheuristic optimisation algorithm, finds optimum hyperparameters with much higher accuracy compared to solo-search algorithms like Bayesian optimisation [39] or stochastic algorithms like simulated annealing. Also, in most of the optimised ELM approaches mentioned in the literature review, only input weights and biases are optimised. In our method, we have optimised additional hyperparameters like the type of activation function, number of input lags and the number of hidden nodes which contribute greatly to the improved accuracy. To explore any instability arising due to stochasticity of the HHO algorithm, the ELM-HHO model was run a couple of times for each case and it was found that the optimization algorithm converges well in each case such that the standard deviation and the mean of the forecasts do not differ significantly between successive runs.

The proposed methodology for product demand forecasting is illustrated in Fig. 3. The methodological development of the proposed model can be described as follows: *i*) data cleansing and preparation, *ii*) selection of hyper-parameters to be optimised (refer to Table 1); *iii*) random generation of learning parameters within a specified range; *iv*) implementation of the HHO algorithm; *v*) iterative generation of model parameters using the HHO algorithm; *vi*) training ELM using the generated model parameters and calculating RMSE of forecasts with the test dataset; *vii*) checking fitness (i.e., RMSE of test dataset) at each iteration; *viii*) generation of optimised values of model parameters based on iterative performance; and *ix*) finally, validation of the ELM-HHO model using optimised model parameters for the best fitness (i.e., the least Test RMSE). The main steps of the methodology are described in more detail below.

4.1 Preparing data

Data cleansing and normalisation are the essential initial steps to be taken in any forecasting task. In our methodology, first the noisy values were smoothed and the missing values were replaced through linear interpolation. Then the time series data was normalised according to the following equation before being fed into the model:

$$x_{normalised} = \frac{x_i - \mu}{\sigma} \tag{23}$$

where x_i is i^{th} data, μ is the mean of all training data and σ is the standard deviation of all training data. Note that the predictions were transformed back to the original scale using the following denormalisation equation before making final predictions:

$$x_{denormalised} = (x_{normalised} \times \sigma) + \mu \tag{24}$$

4.2 Generating hyperparameters for ELM network

ELM models have achieved acceptable forecasting performance when applied on time-series data [26]. However, obtaining good performance with ELM is a tedious task as it involves the optimisation of various hyperparameters. In this study, the following hyperparameters were optimised.

1. **Lag size:** The number of input lags decides how many previous time steps data should be used to predict data at the next time step. The lag size has a significant influence on the performance of time series forecasting [40]. Hence, it is vital to select the optimal lag size in forecasting applications.
2. **Number of hidden nodes:** Similar to other artificial neural network methods, the number of hidden neurons in ELM has a significant effect on its performance [41].
3. **Input weights and biases:** The weights and the biases of the hidden layer of ELM are drawn from a truncated normal distribution [23]. This may lead to non-optimal performance and may result in ill-condition [42]. Random weights and biases may also result in over-fitting due to the ELM's excessive learning abilities [43]. In this study, due to these limitations, input weights and biases of the ELM network were optimally selected to achieve better generalisation performance and improved forecasting accuracy of the ELM model.
4. **Type of activation function:** The performance of the network varies according to the type of activation functions used in the ELM structure. The selection of the activation function generally depends on the problem, and this selection might have a significant impact on the performance of ELM networks [44]. In this research, (1) binary step function, (2) sigmoid, (3) hyperbolic tangent (TanH) and (4) rectified linear unit (ReLU) were considered.

4.3 Initialisation of the HHO algorithm

Single-objective Harris hawks optimisation was used to find optimal hyperparameter values. In the proposed model, each individual depicts a potential solution as a real vector *I*, where *I* consists of multiple variables to be optimised.

$$I = [x_1, x_2, x_3^1, x_3^2, ..., x_3^K, x_4^1, x_4^2, ..., x_4^{x_2}, x_5] \tag{25}$$

Here, *K* is the total number of weights (Table 1). The root mean squared error (RMSE) was used as the fitness function since it is the most widely utilised cost function in evolutionary ELMs. RMSE represents the square root of the mean

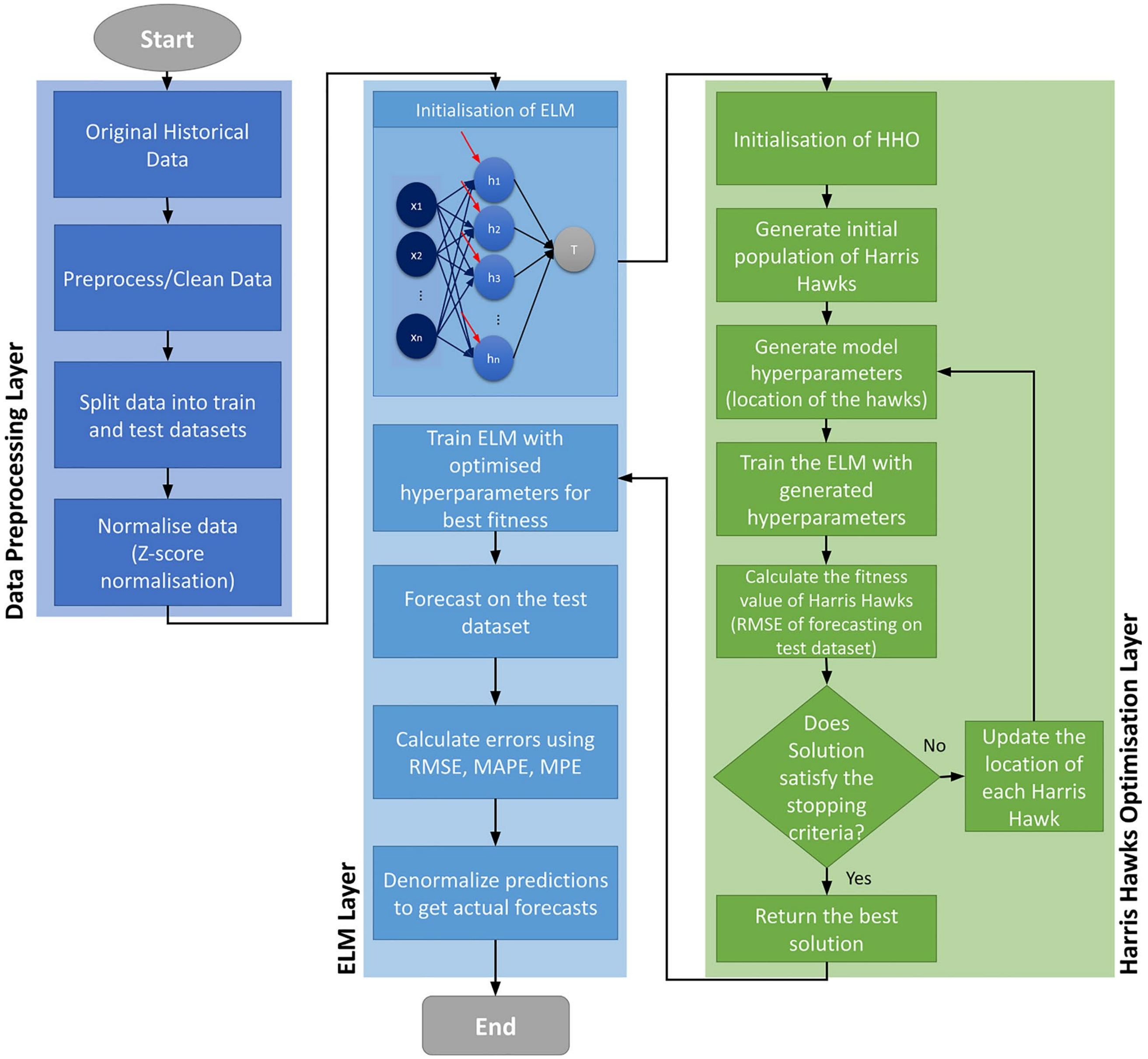
Start
Data Preprocessing Layer
Original Historical Data
Preprocess/Clean Data
Split data into train and test datasets
Normalise data (Z-score normalisation)
ELM Layer
Initialisation of ELM
x1
x2
xn
h1
h2
h3
hn
T
Train ELM with optimised hyperparameters for best fitness
Forecast on the test dataset
Calculate errors using RMSE, MAPE, MPE
Denormalize predictions to get actual forecasts
End
Harris Hawks Optimisation Layer
Initialisation of HHO
Generate initial population of Harris Hawks
Generate model hyperparameters (location of the hawks)
Train the ELM with generated hyperparameters
Calculate the fitness value of Harris Hawks (RMSE of forecasting on test dataset)
Does Solution satisfy the stopping criteria?
No
Update the location of each Harris Hawk
Yes
Return the best solution

Fig. 3 Flowchart of the ELM-HHO approach

Table 1 Lower and upper bounds of the ELM-HHO model parameters

	Variable	Type	Lower bound	Upper bound
x_1	No. of input lags	Integer	1	275
x_2	No. of hidden nodes	Integer	1	1826
x_3	Hidden weights	Double	–5	5
x_4	Hidden biases	Double	–5	5
x_5	Type of activation function	Integer	1	4

squared differences between predicted values and actual values. The fitness function is given as follows.

$$Fitness = \sqrt{\frac{\sum_{i=1}^{N}(Y_i - \hat{Y}_i)^2}{N}} \tag{26}$$

where Y_i is the actual sales data,$\hat{Y}_i$ is the predicted sales data and N is the total number of days for which forecasting is done. It is a scale-dependent measure used to determine how far the absolute values of the forecasts are from the actual sales data.

4.4 Initialisation of the ELM algorithm

The HHO algorithm was used to find the optimal hyperparameters which gave the least RMSE of forecasting on the test dataset. The final chosen hyperparameters were then used to train ELM to obtain optimal forecasts. An ELM architecture with the optimum number of hidden nodes was used. If the optimum no. of input lags came out to be t, sales data at $(n-1), (n-2), (n-3) ... (n-t)$ days were used as inputs to predict the target sales data at the n^{th} day. The optimised weights and biases of the hidden layer with the optimised activation function were used to compute the hidden layer output matrix H. The output weight matrix was then calculated by $\hat{\beta} = H^{\dagger}T$ using the training data T. Finally $\hat{\beta}$ was used to make predictions on new test data using $F = H\hat{\beta}$.

5 Experimental results and discussion

5.1 Dataset

In this study, product demand forecasting was performed using historical sales data from the gaming industry. The global video game market has grown a lot in the past few years. Demand volatility is a particular characteristic of this market. Hence predicting future demand in advance can help gaming companies secure and retain new customers and get an edge over others in the market. Demand forecasting can enable these businesses to maintain an appropriate balance between their suppliers and their buyers. It can also help them ensure improved customer satisfaction at less cost to the supply chain. In this context, our article aims mainly to study the historical sales data of multiple products to provide reasonably accurate forecasts within a permissible error range. The dataset was attained from an e-commerce company based in Western Europe. It had daily sales data of 45 different products from multiple stores from January 2013 to December 2017. The original dataset had 4 columns in tabular form corresponding to date, store, item number and number of sales respectively. Data was present from 10 different stores and the daily sales number for each of the 45 products of a particular store was an integer roughly varying between 5 and 150. Data for each product and store showed a similar trend with some variations. Hence results for only Store 1 have been shown in our paper. We also summed over the daily sales data of all 45 products from Store 1 to consider the cumulative sales trend of video games from Store 1. When broken down into trend, seasonality and residual components (Fig. 4), the time series data showed an almost monotonically increasing trend and pronounced seasonality effects, with peaks in sales appearing at regular annual intervals. This showed that the time-series was not random, and hence could be forecasted. Results of Dickey-Fuller test revealed that the series was not stationary, but it became stationary on performing a first-order differencing operation. The statistical description of the utilised dataset (for Store 1) is given in Table 2.

5.2 Experimental setup

Data from January 2013 to September 2017 was used for training and October 2017 to December 2017 data was used for testing. For the ELM-BO and ELM-HHO approaches, an optimum number of lags or previous time steps data was chosen using the optimisation algorithm to be fed as input features into the ELM. The number of hidden nodes, values for hidden weights and biases in the ELM and the type of activation function were also optimally selected. Daily sales data for the last three months (October 2017 to December 2017) of the dataset was predicted. All computer operations were performed in Python environment, where the computer was configured with Intel(R) Core (TM) i7- 10750H CPU @2.60 GHz and 16 GB RAM.

5.3 Performance metrics

Along with RMSE, two additional performance metrics were used to evaluate the accuracy of forecasts by our approach. These metrics are:

- **Mean Absolute Percentage Error (MAPE)** : MAPE is calculated as the average absolute percentage error between the observed values and the predictions.

$$MAPE = 100 \times \frac{1}{N} \sum_{i=1}^{N} \left| \frac{(Y_i - \hat{Y}_i)}{Y_i} \right| \tag{27}$$

  Percentage errors have the advantage of being unit-free and are useful for comparing results among the different stores.

- **Mean Percentage Error (MPE)** : MPE is the same as MAPE, but instead of taking the absolute errors in forecasting, we consider the actual errors with sign.

$$MPE = 100 \times \frac{1}{N} \sum_{i=1}^{N} \frac{(Y_i - \hat{Y}_i)}{Y_i} \tag{28}$$

5.4 Parameter settings

The number of search agents and the maximum number of iterations are the two primary parameters of the HHO algorithm. The number of search agents has a direct relationship with the search stage's efficiency. The number of search agents or population size was taken as 15 in Harris hawks

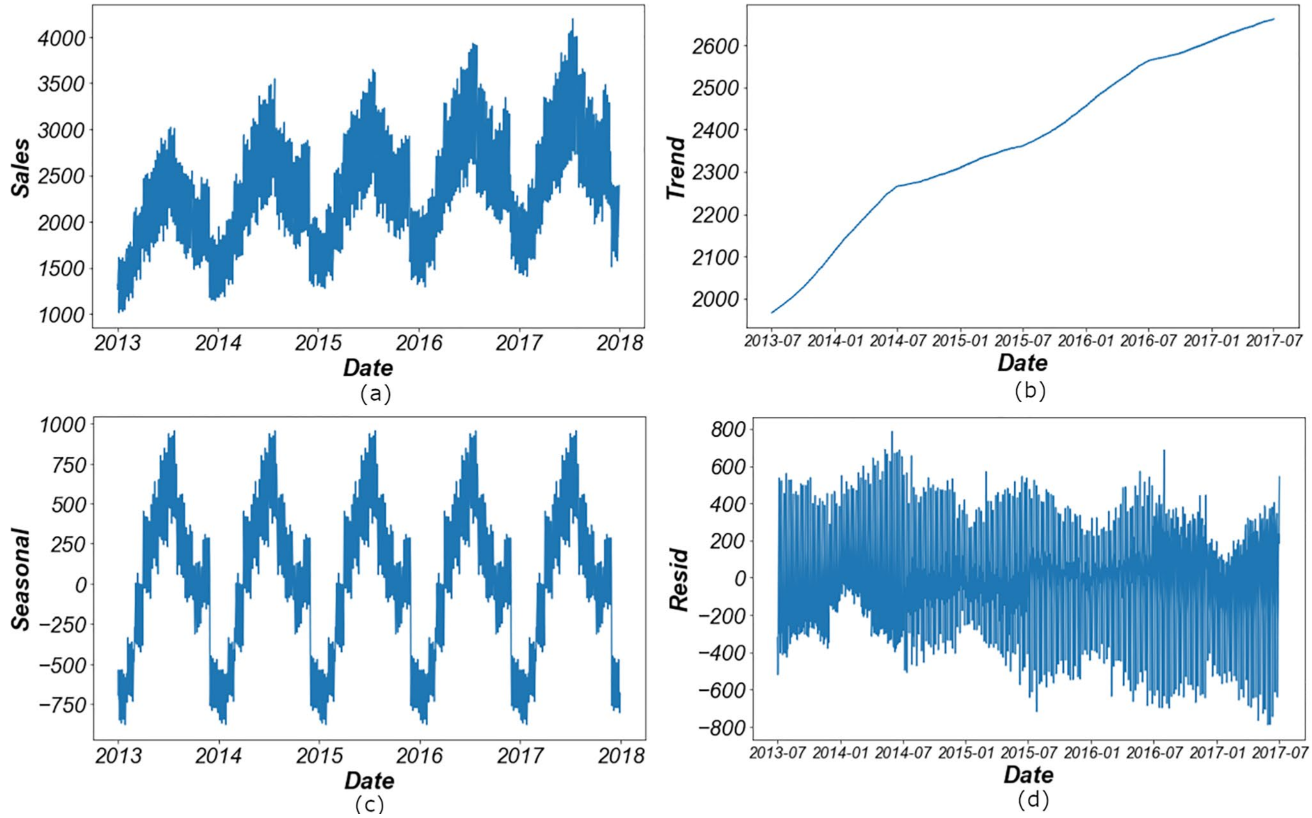

Fig. 4 Data visualisation : *a*) Daily sales data from January 2013 to December 2017, *b*) Trend component of data, *c*) Seasonal component of data, *d*) The residual values, excluding trend and seasonal components

Table 2 Descriptive statistics for product sales between Jan. 2013 and Dec. 2017

Partition	Time Period	No. records	Mean	St. dev.	Median	Min.	Max.
Training	Jan. 2013 - Sept. 2017	1736	2357.47	588.32	2325.00	1012	4199
Testing	Oct. 2017 - Dec. 2017	90	2478.14	460.90	2429.50	1510	3485
Total	Jan. 2013 - Dec. 2017	1826	2363.42	583.18	2339.50	1012	4199

optimisation technique. Preliminary testing was performed to choose this number which presented an optimal balance between accuracy and computational costs. The optimisation algorithms for ELM (i.e., BO and HHO) were run for 200 iterations each. Both the algorithms converged well within this number of iterations for our dataset. Bayesian optimisation was performed with the open-source Python library HyperOpt using a Tree of Parzen Estimator (TPE) algorithm to construct the surrogate function. Although the same methodology was followed by both ELM-BO and ELM-HHO, the optimised hyperparameters came out different for both (Table 3). In the traditional ELM approach, none of the hyperparameters were optimised. ELM models with 50 hidden nodes, 100 hidden nodes and 500 hidden nodes respectively with randomly assigned hidden weights and biases and 14 lags as input features were used for the forecasting. 14 lags was chosen as the standard in unoptimised ELM, LSTM and GRU models because the dataset showed a sharp autocorrelation peak at this lag.

ARIMA models commonly used in time series forecasting were also applied to our dataset. Autocorrelation and partial autocorrelation plots of the daily sales data were used to choose the parameters of the ARIMA model. The auto-regressive order was chosen as 7 (from the ACF and PACF plots) and the integrated order as 1 (Dickey-Fuller's test revealed that the first-order differenced time series was stationary). The predictions of ARIMA (7,1,0) model were compared with our ELM approaches. For fair comparison, a few other ARIMA models: ARIMA(5,1,0), ARIMA(10,1,0), ARIMA(7,1,1) and SARIMAX[(7, 1, 0) x (1, 1, 0, 12)] models were also used to perform forecasting.

A LSTM model was also used for the forecasting. It was a sequential model with the following layers: i) Input layer, ii) LSTM layer of 128 units, iii) LSTM layer of 64 units,

Table 3 Optimised hyperparameters for ELM chosen by BO and HHO algorithms

Algorithm	No. of input lags	No. of hidden nodes	Activation function	Weight range	Bias range
ELM-BO	158	661	Relu	[–2.478, 3.016]	[–4.690, 3.049]
ELM-HHO	194	656	Relu	[–0.588, 3.719]	[–0.612, 3.507]

iv) Dropout layer with a dropout rate of 0.5, v) LSTM layer of 32 units, vi) Dropout layer with a dropout rate of 0.25, vii) Fully connected layer, viii) Regression layer. A lag of 14 days was used as input features at each time step. The mini batch size was taken as 64 with an initial learning rate of 0.00202 and learn drop factor of 0.25. The L2 regularisation metric was 1e-3 and the optimiser was Adam. The model was trained over 200 epochs with shuffling after every epoch. A GRU model was also used for the forecasting. It was a sequential model with the following layers: i) Input layer, ii) GRU layer of 128 units, iii) GRU layer of 64 units, iv) Dropout layer with a dropout rate of 0.5, v) GRU layer of 32 units, vi) Dropout layer with a dropout rate of 0.25, vii) Fully connected layer, viii) Regression layer. A lag of 14 days was used as input features at each time step. The mini batch size was taken as 64 with an initial learning rate of 0.00202 and learn drop factor of 0.25. The L2 regularisation metric was 1e-3 and the optimiser was Adam. The model was trained over 200 epochs with shuffling after every epoch.

5.5 Comparison of ELM-HHO results with ELM, ELM-BO, LSTM, GRU and ARIMA models

Figure 5 plots the forecasting results for all the models with the error values given in Table 4.

RMSE values of the test dataset were used for the comparison. Among the ARIMA models, ARIMA (7,1,0) gave the best results since the order was chosen from the trends observed in the auto-correlation and partial auto-correlation plots of the dataset. The ACF plot was found to be strong to a lag of 7, the inertia of that relationship carried on to subsequent lag values and then trailed off. So 7 was a good choice for the auto-regressive order. The integrated order of 1 was justified by a Dickey-Fuller's test which showed that the first-order differenced time series was stationary. ARIMA(7,1,0) performed 135.62% better than ARIMA(5,1,0), 1.57% better than ARIMA(10,1,0), 23.38% better than SARIMAX [(7,1,0) × (1,1,0,12)] and 1.46% better than ARIMA(7,1,1) in terms of test RMSE of forecasting.

The ELM models, ELM, ELM-BO and ELM-HHO performed better than the ARIMA models in making future demand predictions. ELM-BO showed an improvement in forecasting accuracy of 45.78% over ARIMA(7,1,0) while our proposed approach ELM-HHO was better than ARIMA(7,1,0) by 62.73%. The ELM models performed better than the ARIMA models because ELM, being a neural network model, can capture the non-linearity in sales data better than a statistical model like ARIMA. ARIMA models need to convert the time series to a stationary one, so first order differencing was performed on our dataset. However the underlying process used in ARIMA is linear, so it fails to capture nonlinear models from the time series. Historical sales data like our dataset is often highly non-linear and chaotic because a lot of external environmental factors affect it. ELM models are hence better forecasting models for such datasets.

Table 4 Comparison of performance of different models in forecasting sales data for the last 3 months

Prediction Model	Test RMSE	Test MAPE	Test MPE
ARIMA(5,1,0)	355.306	10.101%	–2.584%
ARIMA(10,1,0)	153.164	4.123%	–0.707%
SARIMAX	186.052	5.887%	0.266%
ARIMA(7,1,1)	152.994	4.116%	–0.694%
ARIMA(7,1,0)	150.797	4.228%	–0.868%
GRU	130.411	3.281%	–1.022%
LSTM	124.217	3.109%	–1.094%
ELM(50 nodes)	133.091	3.379%	–1.240%
ELM(100 nodes)	117.835	2.996%	–0.763%
ELM(500 nodes)	129.647	3.708%	–0.312%
ELM-BO	103.442	3.604%	–0.063%
ELM-HHO	92.665	2.950%	0.153%

The LSTM and GRU models also performed better than the statistical ARIMA models since they are also neural network models like ELM. While the performance of LSTM and GRU models was comparable to that of the traditional ELM models without any optimisation, they did not perform as well as the optimised ELM models - ELM-BO and ELM-HHO. The ELM-BO model was 26.07% better than GRU and 20.08% better than the LSTM model while the ELM-HHO model showed an improvement in forecasting accuracy of 40.73% over GRU and 34.05% over the LSTM model. This might be attributed to the fact that the GRU and LSTM architectures were not tuned to achieve optimal performance.

Test RMSE values of ELM with 100, 500 and 1000 hidden nodes respectively confirmed that the performance of ELM depends on the number of hidden nodes. Hence, the ELMs in which no. of hidden nodes and other parameters were optimised - ELM-BO and ELM-HHO

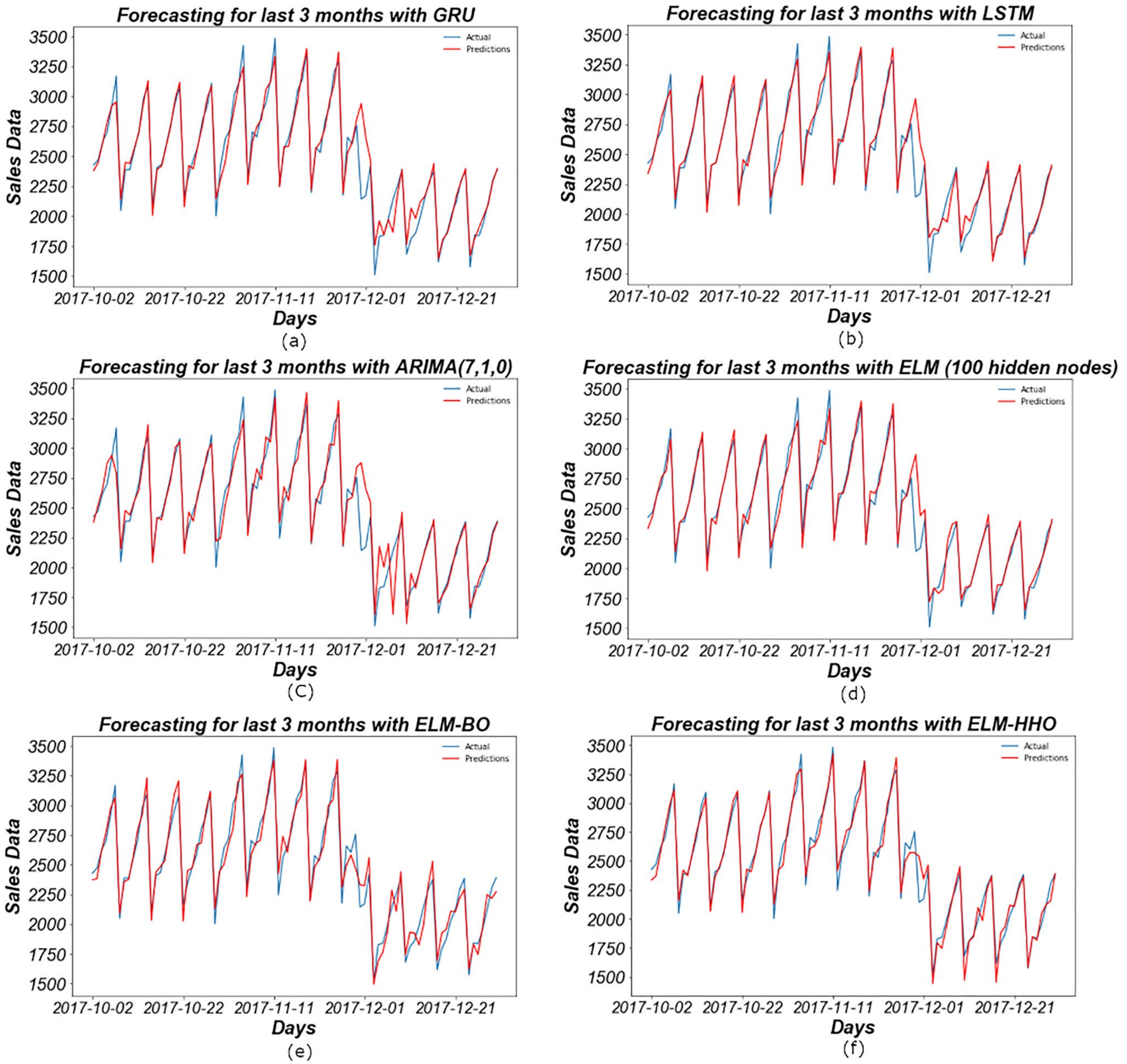

Fig. 5 Forecasting sales data for the last 3 months with *a*) GRU, *b*) LSTM, *c*) ARIMA(7,1,0), *d*) ELM with 100 hidden nodes, *e*) ELM-BO, *f*) ELM-HHO

performed better forecasting than traditional ELM models in which all parameters were chosen randomly. HHO is a meta-heuristic population based optimisation technique and has been shown to have excellent performance on benchmark problems compared to other well-regarded optimisers [30]. For our problem, ELM-HHO performed better than ELM-BO in forecasting last three months sales data by 11.63%. The superior performance of ELM-HHO to ELM-BO can be explained by the fact that BO learns about the underlying objective function through a trial and error approach which might not always result in finding the exact optimum. The problem may also lie in our choice of algorithm to construct the surrogate function, TPE was used in our case instead of Gaussian process or random forest. There are also a number of mechanisms which benefit population-based algorithms like HHO over solo-search ones like BO [39], such as putting together blocks from different solutions (search agents/ hawks), a population's ability to search different parts of the fitness landscape to avoid getting stuck in a local optima due to bad initialisation etc.

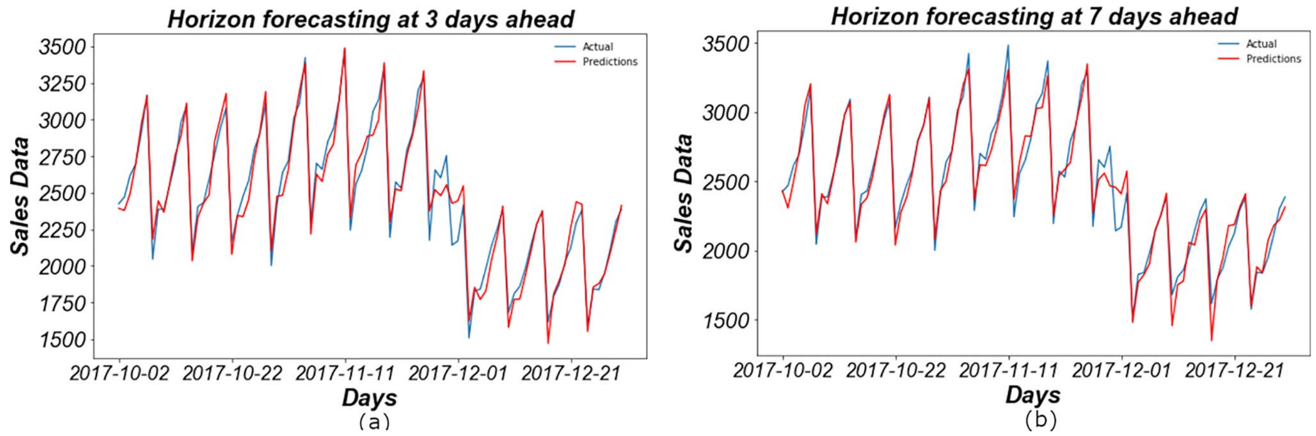

Fig. 6 Horizon forecasting for last 3 months with ELM-HHO at *a*) 3 days ahead, *b*) 7 days ahead

5.6 Horizon forecasting results

Operation managers need long-range as well as short-range demand forecasts to make strategic decisions about products, processes and facilities. Time horizon of forecasts is an integral part of making efficient supply chain decisions and hence needs to be clearly established. Short range forecasts help to decide how much raw material should be ordered for delivery for the next week, how much of each product should be scheduled for production for the next week, how many workers should be scheduled to work next week etc. In this context, forecasting was performed using a 3 day ahead and a 7 day ahead horizon with ELM-HHO to predict the daily sales data for the last 3 months of the dataset. Figure 6 shows horizon forecasting results for the last 3 months, with error values given in Table 5. As expected, forecasting error increases with the increase in horizon length. Even then, ELM-HHO approach resulted in good horizon forecasting accuracy at 3 days and 7 days ahead, thus making it a promising approach to be applied in real time in companies to predict future demand from last week's sales data.

5.7 Discussion

Detailed comparisons with traditional forecasting models revealed that the proposed approach ELM-HHO is a highly accurate model for demand forecasting problems. Although the population-based HHO algorithm is computationally costlier than the solo-search Bayesian Optimisation, the rise in computational cost is offset by the huge improvement in accuracy. To summarise, the proposed ELM-HHO model performed 62.73% better than the statistical ARIMA(7,1,0) model, 40.73% better than the neural network based GRU model, 34.05% better than the neural network based LSTM model, 27.16% better than the traditional non-optimised ELM model with 100 hidden nodes and 11.63% better than the ELM-BO model in forecasting demand for last 3 months. Most importantly, our approach is practical and industrially useful as it can predict sales data accurately over a time horizon. The key strength of the proposed approach lies in combining the fast computational speed of ELMs with the accuracy gained by optimising hyperparameters using HHO. Optimising the number of input lags using HHO yields more accurate results than choosing them by seeing ACF and PACF plots, which is prone to human error. Usually, only input weights and biases are optimised in optimised ELM algorithms. In our approach, more hyperparameters like the type of activation function, number of input lags and number of hidden nodes were optimised which contributed to the highly improved forecasting accuracy.

Table 5 Performance assessment of horizon forecasting for last 3 months with ELM-HHO

No. days ahead	Test RMSE	Test MAPE	Test MPE
3 days	94.214	3.207%	0.183%
7 days	99.884	3.210%	0.445%

6 Conclusion and future work

There are numerous challenges involved in performing accurate and timely product demand forecasting [45–47]. Traditional time series forecasting models like ARIMA can only be applied on stationary time series, and fail to capture non-linearity introduced in sales data due to market volatility, seasonal factors etc. In this context, we introduced a hybrid neural network based demand forecasting model - ELM tuned by Harris hawks optimiser which is able to accurately forecast non-linear sales data of products belonging to an e-commerce store. ELM, being a single layer feed forward neural network, has been shown to be very effective in forecasting non-linear time series data at fast computational speed. We improved the accuracy of ELM further by tuning its hyperparameters with Harris hawks optimisation technique. The ELM-HHO model had better forecasting accuracy than ELM, ELM-BO, GRU, LSTM and ARIMA models, thus making it a great choice for product demand forecasting. The ELM-HHO model also performed good horizon forecasting at 3 days and 7 days ahead, which makes it a good model to be applied by companies in predicting sales data over a horizon. This work can be extended in future by adding auto-encoders for time series feature extraction which might improve accuracy further. CNN or LSTM auto-encoders which can choose the most important time series features to be passed as inputs to the ELM layer, can be analysed. This will help in better identification of the most important input lags and reduce the computational complexity incurred in optimising the number of input lags using an optimisation algorithm. The optimisation part in our approach can be tuned further and multi-objective optimisation can be investigated. The population-based HHO algorithm suffers from certain drawbacks like immature convergence and getting stuck in local optima. Hence, improved versions of HHO, like that mentioned in [48], which uses the Salp Swarm Algorithm (SSA) to enhance the robustness of the HHO algorithm can be explored in future work. The HHO algorithm can even be hybridized with simulated annealing to accelerate its global convergence performance, as shown in [49] and [50]. Newer optimization algorithms like that mentioned in [6] can also be explored. Multivariate demand forecasting which uses inputs other than just the historical sales data can also be explored. For instance, the effect of external factors like seasonal holidays, market volatility etc. can be introduced in our approach to improve robustness.

Author Contributions Koushiki Dasgupta Chaudhuri: Conceptualisation of this study, Methodology, Software, Writing - Original draft preparation. Bugra Alkan: Data curation, Research design, Supervision, Writing and editing of manuscript.

Funding This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Data availability The data that support the findings of this study are available on request from the corresponding author B.A.

Code availability Researchers or interested parties are welcome to contact the corresponding author B.A. for further explanation, who may also provide the Python codes upon request.

Declarations

Conflicts of interest The authors declare that they have no conflict of interest.

Ethical approval Not applicable

Consent to participate Not applicable

Consent for publication Not applicable

References

1. Archer B et al (1987) Demand forecasting and estimation. Demand Forecasting and Estimation:77–85
2. Hamiche K, Abouaïssa H, Goncalves G, Hsu T (2018) A robust and easy approach for demand forecasting in supply chains. IFAC-PapersOnLine 51(11):1732–1737
3. Huang G-B, Zhu Q-Y, Siew C-K (2006) Extreme learning machine: theory and applications. Neurocomputing 70(1–3):489–501
4. Huang G-B, Zhou H, Ding X, Zhang R (2011) Extreme learning machine for regression and multiclass classification. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 42(2):513–529
5. Huang G-B, Chen L, Siew CK et al (2006) Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Trans. Neural Networks 17(4):879–892
6. Zhang Z, Ding S, Jia W (2019) A hybrid optimization algorithm based on cuckoo search and differential evolution for solving constrained engineering problems. Engineering Applications of Artificial Intelligence 85:254–268
7. Christopher M (2016) Logistics & supply chain management. Pearson UK, London
8. Alkan B, Bullock S, Galvin K (2021) Identifying optimal granularity level of modular assembly supply chains based on complexity-modularity trade-off. IEEE Access 9:57907–57921
9. Lu D (2011) Fundamentals of supply chain management. Bookboon, London
10. Gabbott M, Hogg G (1994) Consumer behaviour and services: a review. Journal of Marketing Management 10(4):311–324
11. Islek I, Ögüdücü SG (2017) A decision support system for demand forecasting based on classifier ensemble. In: FedCSIS (Communication Papers), pp 35–41
12. Merkuryeva G, Valberga A, Smirnov A (2019) Demand forecasting in pharmaceutical supply chains: A case study. Procedia Computer Science 149:3–10
13. Fattah J, Ezzine L, Aman Z, El Moussami H, Lachhab A (2018) Forecasting of demand using arima model. International Journal of Engineering Business Management 10:1847979018808673
14. Anggraeni W, Vinarti RA, Kurniawati YD (2015) Performance comparisons between arima and arimax method in moslem kids clothes demand forecasting: Case study. Procedia Computer Science 72:630–637

15. Tanizaki T, Hoshino T, Shimmura T, Takenaka T (2019) Demand forecasting in restaurants using machine learning and statistical analysis. Procedia CIRP 79:679–683
16. Chang P-C, Lin J-J, Dzan W-Y (2012) Forecasting of manufacturing cost in mobile phone products by case-based reasoning and artificial neural network models. Journal of Intelligent Manufacturing 23(3):517–531
17. Lawrence S, Giles CL (2000) Overfitting and neural networks: conjugate gradient and backpropagation. In: Proceedings of the IEEE-INNS-ENNS international joint conference on neural networks. IJCNN 2000. neural computing: new challenges and perspectives for the new millennium. vol 1, IEEE, pp 114–119
18. Wilson D, Martinez T (2001) The need for small learning rates on large problems. In: IJCNN'01. International joint conference on neural networks. proceedings (Cat. No.01CH37222). vol 1, pp 115 – 119
19. Huang G-B, Zhu Q-Y, Siew C-K (2004) Extreme learning machine: a new learning scheme of feedforward neural networks. In: 2004 IEEE international joint conference on neural networks (IEEE Cat. No. 04CH37541). vol 2, Ieee, pp 985–990
20. Sahu RK, Shaw B, Nayak JR et al (2021) Short/medium term solar power forecasting of chhattisgarh state of india using modified tlbo optimized elm. Engineering Science and Technology, an International Journal 24(5):1180–1200
21. Yaseen ZM, Jaafar O, Deo RC, Kisi O, Adamowski J, Quilty J, El-Shafie A (2016) Stream-flow forecasting using extreme learning machines: a case study in a semi-arid region in iraq. Journal of Hydrology 542:603–614
22. Wan C, Xu Z, Pinson P, Dong ZY, Wong KP (2013) Probabilistic forecasting of wind power generation using extreme learning machine. IEEE Transactions on Power Systems 29(3):1033–1044
23. Teo TT, Logenthiran T, Woo WL (2015) Forecasting of photovoltaic power using extreme learning machine. In: 2015 IEEE innovative smart grid technologies-asia (ISGT ASIA). IEEE, pp 1–6
24. Cao Z, Xia J, Zhang M, Jin J, Deng L, Wang X, Qu J (2015) Optimization of gear blank preforms based on a new r-gplvm model utilizing ga-elm. Knowledge-Based Systems 83:66–80
25. Furfaro R, Barocco R, Linares R, Topputo F, Reddy V, Simo J, Le Corre L (2021) Modeling irregular small bodies gravity field via extreme learning machines and bayesian optimization. Advances in Space Research 67(1):617–638
26. Miche Y, Sorjamaa A, Bas P, Simula O, Jutten C, Lendasse A (2009) Op-elm: optimally pruned extreme learning machine. IEEE Transactions on Neural Networks 21(1):158–162
27. Liang N-Y, Huang G-B, Saratchandran P, Sundararajan N (2006) A fast and accurate online sequential learning algorithm for feedforward networks. IEEE Transactions on Neural Networks 17(6):1411–1423
28. Cao J, Lin Z, Huang G-B, Liu N (2012) Voting based extreme learning machine. Information Sciences 185(1):66–77
29. Huang G, Huang G-B, Song S, You K (2015) Trends in extreme learning machines: A review. Neural Networks 61:32–48
30. Heidari AA, Mirjalili S, Faris H, Aljarah I, Mafarja M, Chen H (2019) Harris hawks optimization: Algorithm and applications. Future Generation Computer Systems 97:849–872
31. Yıldız AR, Yıldız BS, Sait SM, Bureerat S, Pholdee N (2019) A new hybrid harris hawks-nelder-mead optimization algorithm for solving design and manufacturing problems. Materials Testing 61(8):735–743
32. Zhang Y, Liu R, Wang X, Chen H, Li C (2020) Boosted binary harris hawks optimizer and feature selection. Eng Comput
33. Rammurthy D, Mahesh P (2020) Whale harris hawks optimization based deep learning classifier for brain tumor detection using mri images. Journal of King Saud University - Computer and Information Sciences
34. Joy TT, Rana S, Gupta S, Venkatesh S (2016) Hyperparameter tuning for big data using bayesian optimisation. In: 2016 23rd International conference on pattern recognition (ICPR). IEEE, pp 2574–2579
35. Bengio Y, Simard P, Frasconi P (1994) Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks 5(2):157–166
36. Kumar S, Hussain L, Banarjee S, Reza M (2018) Energy load forecasting using deep learning approach-lstm and gru in spark cluster. In: 2018 Fifth international conference on emerging applications of information technology (EAIT). IEEE, pp 1–4
37. Fu R, Zhang Z, Li L (2016) Using lstm and gru neural network methods for traffic flow prediction. In: 2016 31st youth academic annual conference of chinese association of automation (YAC). IEEE, pp 324–328
38. Cho K, Van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y (2014) Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv:1406.1078
39. Prügel-Bennett A (2010) Benefits of a population: Five mechanisms that advantage population-based algorithms. IEEE Transactions on Evolutionary Computation 14(4):500–517
40. Ribeiro GH, Neto PSDM, Cavalcanti GD, Tsang R (2011) Lag selection for time series forecasting using particle swarm optimization. In: The 2011 International joint conference on neural networks. IEEE, pp 2437–2444
41. Eshtay M, Faris H, Obeid N (2019) Metaheuristic-based extreme learning machines: a review of design formulations and applications. International Journal of Machine Learning and Cybernetics 10(6):1543–1561
42. Han F, Yao H-F, Ling Q-H (2013) An improved evolutionary extreme learning machine based on particle swarm optimization. Neurocomputing 116:87–93
43. Silva DN, Pacifico LD, Ludermir TB (2011) An evolutionary extreme learning machine based on group search optimization. In: 2011 IEEE congress of evolutionary computation (CEC). IEEE, pp 574–580
44. Li B, Li Y, Rong X (2013) The extreme learning machine learning algorithm with tunable activation function. Neural Computing and Applications 22(3):531–539
45. Hosking JRM (2011) Demand forecasting problems in production planning. International Series in Operations Research & Management Science. In: Kempf KG, Keskinocak P, Uzsoy R (eds) Planning Production and Inventories in the Extended Enterprise, chapter 0, pp 103–117, Springer
46. Alkan B, Vera D, Ahmad M, Ahmad B, Harrison R (2018) Complexity in manufacturing systems and its measures: A literature review. European J of Industrial Engineering 12:116–150
47. Alkan B, Bullock S (2021) Assessing operational complexity of manufacturing systems based on algorithmic complexity of key performance indicator time-series. Journal of the Operational Research Society 72:2241–2255
48. Abd Elaziz M, Heidari AA, Fujita H, Moayedi H (2020) A competitive chain-based harris hawks optimizer for global optimization and multi-level image thresholding problems. Applied Soft Computing 95:106347
49. Kurtuluş E, Yıldız AR AR, Sait SM, Bureerat S (2020) A novel hybrid harris hawks-simulated annealing algorithm and rbf-based metamodel for design optimization of highway guardrails. Materials Testing 62(3):251–260
50. Elgamal ZM, Yasin NBM, Tubishat M, Alswaitti M, Mirjalili S (2020) An improved harris hawks optimization algorithm with simulated annealing for feature selection in the medical field. IEEE Access 8:186638–186652

Authors and Affiliations

Koushiki Dasgupta Chaudhuri[1] · **Bugra Alkan**[2]

Koushiki Dasgupta Chaudhuri
koushiki@iitkgp.ac.in

[1] Department of Mathematics, IIT Kharagpur,
721302 Kharagpur, West Bengal, India

[2] School of Engineering, London South Bank University, 103
Borough Rd, SE1 0AA London, United Kingdom