# A Hybrid Bayesian-Genetic Algorithm Based Hyperparameter Optimization of a LSTM Network for Demand Forecasting of Retail Products

Dr. Pravin Suryawanshi[1], Sandesh Gaikwad[1], Dr. Akansha Kumar[2], Akhil Patlolla[1] and Sai K. Jayakumar[3]

[1]*Data Scientist, Jio Platforms Limited, Navi Mumbai, Maharashtra, India*
[2]*Chief Data Scientist, Jio Platforms Limited, Hyderabad, Telangana, India*
[3]*Product Manager, Jio Platforms Limited, Hyderabad, Telangana, India*

Keywords: Hyperparameter Optimization, Demand Forecasting, Genetic Algorithm, Bayesian Optimization, LSTM Network.

Abstract: Demand forecasting is highly influenced by the non-linearity of time series data. Deep neural networks such as long short-term memory networks (LSTM) are considered better forecasters of such data. However, the LSTM network's performances are subject to hyperparameter values. This study proposes a hybrid approach to determine the optimal set of hyperparameters of an LSTM model using Bayesian optimization and genetic algorithm. Bayesian optimization explores the search space in the direction where the improvement over the existing solution is likely, based on a fitness function. At the same time, a genetic algorithm is an evolutionary approach that can achieve global convergence by using selection, crossover, and mutation operators. The proposed hybrid approach utilizes the strengths of both these algorithms to tune the values of the hyperparameter of the LSTM network to minimize the forecasting error. In the dataset considered, we found that the hybrid approach reduced the forecasting error by approximately 27% compared to the Bayesian optimization approach. Additionally, the proposed method is better than the genetic algorithm when performed independently, with a decrease in error value by approximately 13%.



## 1 INTRODUCTION

The era of globalization, market competition, and customer-centric businesses has made demand forecasting daunting. The accuracy of the forecast affects the planning cycle of any retail business. A better approach to forecasting can streamline the downstream supply chain operations and result in a better customer experience. Recently, many predictive approaches to forecasting have shown promising results. However, the non-linearity in demand, especially in the retail industry, multiplies the complexity during predicting the target variable (Kumar et al., 2020). Much work has been done on prescriptive models in the areas of edge computing infrastructure resource management (Viola et al., 2020), load forecasting in electricity supply (Johannesen et al., 2019), call center arrival calls (Taylor, 2008), forecasting of petroleum products (Sagheer and Kotb, 2019) and others. Unlike in these scenarios, demand forecasting in retail lacks a stable exogenous variable to guide the forecasting process (Carbonneau et al., 2008). Thus, it would be interesting to study and analyze the pattern of demand information and minimize forecasting errors while adopting advanced predictive analytics techniques. Demand forecasting for effective inventory optimization falls under the purview of time-series forecasting. Computational intelligence methods, like recurring neural network (RNN), have a special feature of short-term memory, which utilizes the prevailing input information to create effective future decisions in case of time-series data (Parmezan et al., 2019).

The prediction decisions with the memory cell are categorized as a long short-term memory (LSTM) network based on their strength of controlling information for future decisions. A few issues often observed in such models are variability in fitting the trend, training procedures, selection of algorithm, and, most importantly, the selection of the optimal set of hyperparameters. Often confused with internal model parameters, hyperparameters are learned be-



Suryawanshi, P., Gaikwad, S., Kumar, A., Patlolla, A. and Jayakumar, S.
A Hybrid Bayesian-Genetic Algorithm Based Hyperparameter Optimization of a LSTM Network for Demand Forecasting of Retail Products.
DOI: 10.5220/0012182900003595
In *Proceedings of the 15th International Joint Conference on Computational Intelligence (IJCCI 2023)*, pages 230-237
ISBN: 978-989-758-674-3; ISSN: 2184-3236

fore the training phase of the actual time-series model. They help in balancing the trade-off between model accuracy and model execution by providing the best-tuned parameters in a reasonable length of time.

As high model accuracy is vital, there is a need to have the correct set of hyperparameters, the desirable evaluation metric, the right choice of initialization (with or without bias), etc., with reference to hyperparameters. Theoretically, identifying hyperparameters has seen much advancement with the evolution of different search techniques. For example, random or grid search approaches have been tested and proven to yield good results. With increasing network structure, leading to a larger number of parameters and a larger search space, the performance of such approaches has been observed to reduce (Feurer and Hutter, 2019). It is interesting to combine meta-heuristic approaches such as evolutionary-based ones with BO, considering their abilities to reduce search complexities, manage multimodal and nonlinear input information, local and global searching strategies, and achieve global optimum with fewer sets of tuneable LSTM parameters. This makes the problem interesting to study and motivates the research.

## 2 RELEVANT WORK

Several methods have been developed to address demand forecasting challenges in retail. Most of them rely on statistical intelligence methods. (Ramos et al., 2015) designed a forecasting model based on state space analysis and ARIMA (AutoRegressive Integrated Moving Average) for a retail network for the women's footwear industry. The authors found that state-space models outperform the ARIMA approach in the case of out-of-sample data at the cost of high computational efforts. It is also observed from past research that the performance of statistical methods such as ARIMA, moving averages, and exponential smoothing depreciates in the case of time series with irregular and highly random features due to nonlinearity and data leakage (Abbasimehr et al., 2020). Many studies have shown promising results considering advanced algorithms such as grid search, random search, BO approach, etc. However, each technique has disadvantages while training on large data sets. In the grid search approach, the number of evaluations increases exponentially with an increasing number of parameters making the grid search unproductive (Johnson, 2017). While in the random search, due to higher variability and no intelligent decision-making in selecting the optimal hyperparameters, the method suffers from fluctuations in the cost objective, resulting in relatively slower conversion (Kumar et al., 2021). Furthermore, gradient-based approaches are more likely to be trapped in a local optimum (Frazier, 2018). Additionally, such approaches are ineffective while handling categorical hyperparameters, which is hardly the case with the BO approach (Elsken et al., 2019).

On the contrary, the LSTM method under RNN can create memory and forget cells to improve forecasting accuracy by preserving required patterns from the past. (da Fonseca Marques, 2020) compared the LSTM model with the seasonality-based ARIMA approach on a fish market retail network, considering price, holidays, and whether the model features improved prediction accuracy. Similar findings were observed by (Abbasimehr et al., 2020) in the case of a furniture company with a relatively stable demand for real-time forecasting of time series data. Another advantage of LSTM models is that they effectively solve errors due to missing data and explore gradients using the built-in gates architecture that controls the flow of information among the cells (Cansu et al., 2023). Often, the design of the LSTM network and tuning of the hyperparameters is an intimidating task. Thus, (Johnson, 2017) suggested the implementation of hybrid approaches such as BO, evolutionary algorithms, swarm-based intelligence techniques, and others. Especially, evolutionary approaches have inherent qualities of not falling into the local optimality with gradient-free optimization features (Beheshti and Shamsuddin, 2013).

Few studies mention the use of meta-heuristic tools to create neural network infrastructure or speed up the architecture's performance by selecting optimal tuning parameters. (Kumar et al., 2021) trained a deep neural network model on stock market data using a genetic algorithm (GA) approach to find the optimal set of network hyperparameters and data subset selection. The main advantages of employing meta-heuristic approaches are tuning multiple hyperparameters and simultaneously providing near-optimal prediction performance. Specific to demand forecasting for retail goods, (Abbasimehr et al., 2020) designed an LSTM network model as a forecaster and compared the results with ARIMA and RNN approaches. The authors did not use any evolutionary approaches. Therefore, it will be interesting to study the design decisions that affect the performance of the LSTM network - more specifically, finding the architecture parameters of the LSTM network, identifying the hyperparameter tuning values, or reducing the dimensionality in the feature representation level of the LSTM network.

# 3 METHODOLOGY

## 3.1 Fundamentals of an LSTM Network

Lately, an LSTM model as a subset of RNN has been adopted in many studies as a sequence prediction approach considering their memory advantages and input-output handling capabilities (Greff et al., 2016). The advantages are evident with gates for input and output and cell memory. Typically, an LSTM has an internal storage system called a memory cell featured with an internal state, different gates, and a mechanism with which the internal state interacts with the different gates in place.

Such functions are helpful to create the bounds on the output variables with set range values generally between 0 and 1. For every time step of the LSTM implementation, the forget gate determines whether to pass the current value of memory or completely discard it. In contrast, the output gate controls the influence of the memory cell on the output. An input node with an activation function is often attached to the gate. Primarily, the input gate advocates the addition of the input node's value in the current state of the memory cell. In our experimentation, the LSTM architecture consists of two hidden layers, a *tanh* activation function, and a single dense layer which is trained using Adam optimizer with mean squared error as a loss function.

## 3.2 A Hybrid Solution Strategy

In this section, we propose a learning algorithm that facilitates the execution of the LSTM model using a hybrid approach based on BO and GA. Unlike previously attempted approaches of combined strategies as in (Martinez-de Pison et al., 2019), the existing approach does not limit the number of model parameters to find the best features. Since most meta-heuristic methods require an initial solution, the output from the BO approach is fed as an initial solution to the second stage of hyperparameter optimization. The second stage uses GA with an initial population as obtained from the BO output. With advanced operators such as selection, crossover, and mutation, the best individual of some generations might be dropped during iterations. To avoid these, an optional elitism strategy is employed in the many GA-based approaches using a simple hall-of-fame concept (Wirsansky, 2020). As many best individuals as set by some constant integer (i.e., the hall of fame parameter) will always be kept in the mating pool of a population. We implement the above concept with motivation and explanation mentioned by authors (Fortin et al., 2012) and (Wirsansky, 2020). Such a strategy enhances the GA's performance by avoiding the wastage of time involved in rediscovering the potential solution. The central idea of the proposed hybrid strategy is depicted in Figure 1.

A mathematical description of the hyperparameter optimization process is described below, with the importance of the BO approach. Let $F(h)$ be a given loss function, i.e., Root Mean Square Error (RMSE). In our case it is represented by Equation 1 and is subjected to optimize over $h_1, h_2, \cdots, h_n$ hyperparameters and each of these hyperparameters $(h_i)$ have lower and upper bounds $[l_i, u_i]$ in a configuration or hyperparameter space $\Omega = [l_1, u_1] \times \cdots \times [l_n, u_n]$.

$$F(h) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2} \quad (1)$$

Where $x_i$ and $\hat{x}_i$ are the actual and forecast values of the series in time point i, respectively.

However, computing the true objective function is an expensive exercise. Thus, a surrogate model is built with the acquisition function mentioned in Equation 2, which is relatively cheaper to evaluate.

$$h^* = \underset{h \in \Omega}{\arg\min}\ f(h|\phi_{1:i-1}) \quad (2)$$

Where y is the true fitness value and $\phi$ is a search space of $h$ and $y$. With every iteration, more samples are added to the surrogate model with their respective acquisition function until a termination criterion is reached.

The fundamental understanding of every evolutionary algorithm follows the principle of the survival of the fittest individual in a search pool comprising feasible and infeasible candidate solutions. To evaluate the quality of the solution, a fitness measure in terms of the objective function is employed. Often, operators are employed in such meta-heuristics to improve the candidate solution to intensify (exploitation) and diversify (exploration). Some key terminologies in the evolutionary algorithm are elucidated as follows.

Selection operations are one of the fundamental tasks in the evolutionary approach. Many parents are selected depending on their fitness strength from a set of solution pools at each iteration. Often, this filtering is performed with the help of some set criteria. In the proposed study, we used tournament selection to find the best candidates, which will result in the next generation or be a part of it (Deb and Jain, 2013). The Crossover operator produces a diversified solution by searching different regions within the given solution space. In contrast, the mutation operator produces a high-quality solution by intensifying the search within the given region of solution space. We

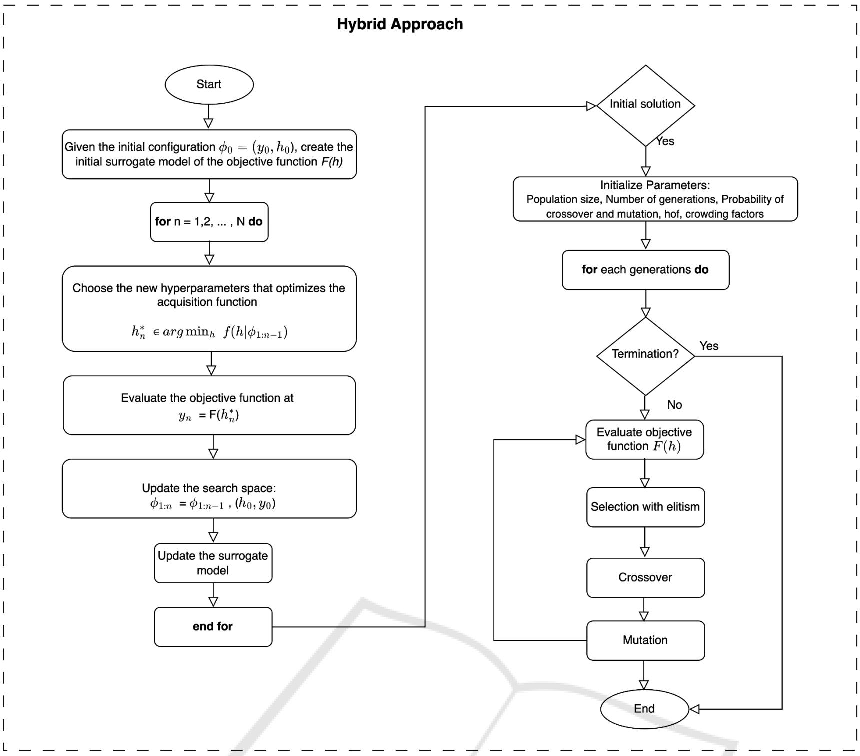


Figure 1: Flowchart for the proposed hybrid approach.

implemented a crossover option using simulated binary crossover and a mutation method based on polynomial mutation as popularly considered in NSGA II implementation (Deb and Jain, 2013).

# 4 PROBLEM DESCRIPTION, EXPERIMENTAL SET-UP AND RESULTS

Inventory optimization is a critical task across different sectors of the business. Especially, fast-moving goods pose alarming challenges in dealing with demand uncertainty as it involves a huge amount of monetary investment, time criticalities, and technology infrastructure to manage operational challenges and maintenance scenarios (Fildes et al., 2022). Additionally, the time series data features make forecasting efforts more challenging. First, a high-dimensionality problem is complex due to too many variables and too little data information. Second, the intermittent and promotion-driven episodes drive the random demand as completely non-stationary, exhibiting variable trends, i.e., the series' frequency, mean, and variance undergo several changes over time. The product properties and nature of the business model further complicate the problem and may result in nonlinearity and heterogeneity (Lang et al., 2015). Building a capability of predicting highly fluctuating demand data would be an interesting problem to study. However, in many situations, the results of such learning algorithms are governed by a set of hyperparameters. For example, some good examples of such top-level parameters are the number of hidden layers, dropout rate, epoch size, batch size, learning rate, etc. (Reimers and Gurevych, 2017). The optimal selection of such hyperparameters improves the model's performance. Therefore, choosing the right set of hyperparameters and their values is a prominent question to address before implementing the learning model.

The subsequent sections trigger the need to implement the hybrid approach to identify the optimal sets of hyperparameters for a better LSTM network prediction. We perform independent simulations for different approaches proposed in subsections to find the optimal values of the hyperparameters. It is also important to note that the experiments are performed on an Apple M1 Pro chip with 16 GB of RAM and a tencore CPU. The description is further categorized into sub-sections to explain preliminary results related to each approach. The basic information about the nonlinearity and causality in the uni-variate forecasting random variable is presented in Figure 2. The underlined time series demand data of retail products consists of trends, seasonalities, and errors in terms of residues. For example, an upward trend shows an increase in demand values. Seasonality explains the cyclic pattern occurring at regular intervals. In addition, a residue component is present in the time series data, which is neither systematic nor predictable (Parmezan et al., 2019).

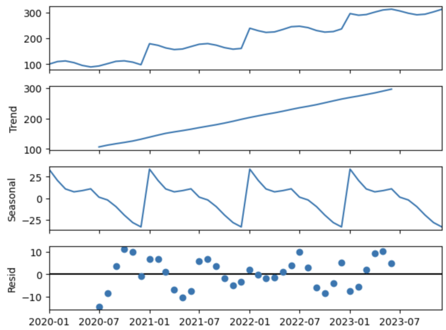


Figure 2: Decomposition of time series demand data.

Additional information related to the parameter setting is as follows. Five major hyperparameters related to LSTM are considered for our experimentation. The details of which, as mentioned in the Keras documentation (Chollet et al., 2015), is as follows:

1. Units of layer: This represents the dimensionality of the output space and is a positive integer. We have taken two hidden layers in the LSTM network with units varying from 10 to 25.
2. Dropout: This hyperparameter decides the fraction of the units to drop for the linear transformation of the inputs and takes a continuous value between 0 and 1.
3. Batch size: This defines the number of samples per gradient update. We took a lower and upper bound for the batch size of 16 and 64, respectively.
4. Epochs: This decides the number of epochs to train over the LSTM model, which is a positive integer. Epochs are between 5 and 15 during simulation.
5. Learning rate: This hyperparameter decides how fast the LSTM model updates its parameters. This parameter takes a value between 0 and 1. With a very high learning rate value, the model may not converge, and a very low learning rate will slow down the learning process.

GA requires a few parameters such as population size, probabilities for crossover and mutation, maximum number of generations, and population size for the Hall of Fame. For our experiments, we have fixed the values of these parameters with well-known standard values as described by (Fortin et al., 2012). For example, probabilities of crossover and mutation are taken as 0.5 each, respectively. Similarly, the crowding factor for mutation is 15, and the same is 10 for crossover operations. Additionally, an integer value of 2 for the Hall of Fame is considered throughout the simulations.

## 4.1 BO for LSTM Network Implementation

BO is used as a hyperparameter optimization tool in various machine learning models and in well-known Python libraries for building neural networks. The methodology section fairly explains the execution of the BO approach. This section highlights some key computation implications related to the methodology. The BO implementation is as per (Balandat et al., 2019). The recorded objective (RMSE) shows a declining trend with increasing execution time as the number of iterations is increased (Figure 3).

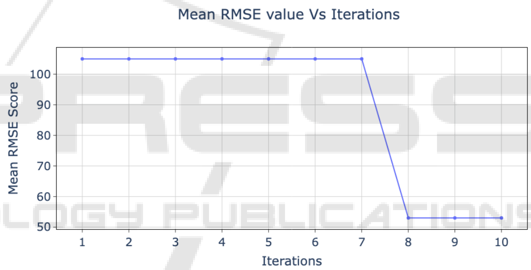


Figure 3: Iteration vs. Mean RMSE score of BO approach for one of the simulations.

## 4.2 GA with Elitism

The implementation is based on the DEAP library by (Fortin et al., 2012). One of the major decisions on population size (P1) and number of generations (P2) is decided by a trial-and-error approach to understand the implementation of the GA with elitism approach.

Different combinations of P1 and P2 are taken to identify their best values based on fair number of fitness evaluations (population size multiplied by number of generations). We elucidate the behavior of the RMSE score in Figure 4. We observed that a population size of 20 and a number of generations of 10 has given better performance compared to other sets of combinations. Thus, we fix these values throughout experimentation. Fixing the values of P1 and P2 is a subjective question and depends on the dataset and search strategy employed within the optimization al-

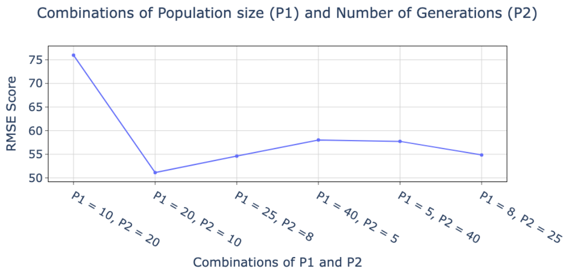


Figure 4: Trial-and-error approach for fixing values of population size (P1) and number of iterations (P2).

gorithm. Therefore, the approach is sensitive toward the optimal set of both these parameters.


Figure 5: Iteration vs. RMSE score of GA approach for one of the simulations.

Figure 5 shows three series plots for maximum, minimum, and averages of RMSE score evaluated for the population during each iteration of the GA approach. For example, at every iteration, the algorithm selects some set of parents out of a population pool of 20. It is observed that over the iterations, the fitness value shows a declining trend. In the instance above, the minimum value of RMSE achieved is 46.9765.

## 4.3 Proposed Hybrid Strategy for Hyperparameters Tuning

The current section mentions results related to the combined strategy proposed in the paper. Primarily, we highlight the need and advantages of adopting such approaches to model LSTM networks subject to optimal configurations of hyperparameters. It is important to note that most of the meta-heuristics are given an initial solution to start with. We utilize the surrogate output from the BO to warm start the search space for a hybrid approach. This was primarily implemented with the motivation of early termination and improving the prediction strategy during algorithm implementation. In the current simulation experimentation, we carried out multiple sets of independent simulations by fixing the total computation time assigned to each approach.

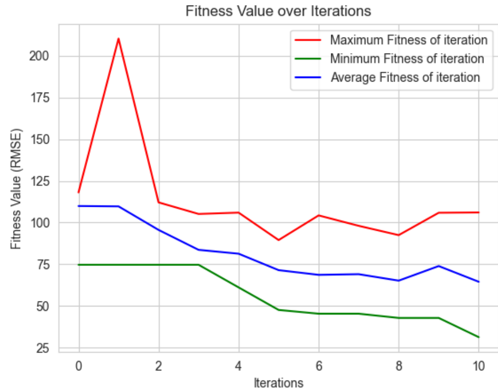


Figure 6: Iteration vs. RMSE score of Hybrid approach for one of the simulations.

The Figure 6 represents one of the simulation results for which the RMSE score achieved was 31.3041. The number of chromosomes evaluated at every iteration during the hybrid approach might not be equal to the number evaluated during the GA approach. Further, we performed a set of simulations to understand the performances of each of the proposed algorithms considering a similar execution time. The complete simulation experiments are mentioned in Table 1.

Table 1: Comparison of BO, GA, and hybrid approach based on an independent set of simulations.

| Simulation No. | RMSE Score | | |
|---|---|---|---|
| | BO | GA | Hybrid Approach |
| Simulation 1 | 49.29 | 43.77 | 30.90 |
| Simulation 2 | 44.78 | 43.72 | 35.70 |
| Simulation 3 | 49.45 | 30.04 | 31.30 |
| Simulation 4 | 48.26 | 41.10 | 35.40 |
| Simulation 5 | 51.77 | 54.11 | 40.84 |
| Simulation 6 | 47.99 | 46.98 | 40.31 |
| Simulation 7 | 50.50 | 35.95 | 35.90 |
| Simulation 8 | 47.52 | 41.37 | 36.05 |
| Simulation 9 | 45.13 | 42.25 | 36.37 |
| Simulation 10 | 52.84 | 32.03 | 35.06 |

In most of the simulations, the hybrid approach records a lower RMSE score, which is also highlighted in Figure 7. The information about hyperparameters obtained from three algorithms is mentioned in Table 2.

Other key findings from our analysis are as follows:

- We employed GA with elitism approach. Since meta-heuristics efficiently reach feasible solutions with faster conversion, we observed an average of

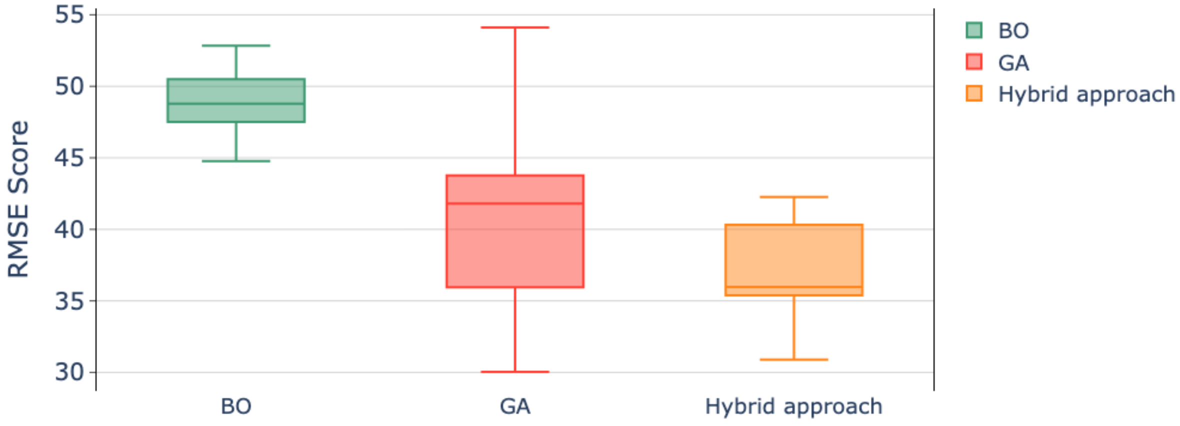


Figure 7: Boxplot of RMSE scores for BO, GA, and Hybrid approach.

Table 2: Optimal values of hyperparameters for one of the simulations.

| Hyperparameters | BO | GA | Hybrid Approach |
|---|---|---|---|
| layer1 units | 23 | 23 | 22 |
| layer1 dropout | 0.6 | 0.59 | 0.33 |
| layer2 units | 23 | 24 | 16 |
| layer2 dropout | 0.6 | 0.44 | 0.20 |
| epochs | 7 | 8 | 10 |
| batch size | 16 | 16 | 17 |
| learning rate | 0.94 | 0.71 | 0.76 |

160 seconds of convergence time to reach the desired RMSE value after a trial-and-error simulation.

- A major observation during experimentation is that BO's computational time is relatively higher compared to the hybrid approach presented in the paper. The same case is true when compared with the GA with elitism approach.
- In the hybrid method, when the first stage output from BO is provided as the initial solution to the second stage, the mean RMSE is 30.90, while the BO approach attains a mean RMSE of 49.29 when run for the same duration (see Table 1).
- Table 1 and Figure 7 represent independent set of simulations for the three approaches implemented in the proposed study. The result highlights that the hybrid approach shows a significant difference in terms of RMSE score with a relatively smaller mean and lower variability compared to BO and GA approaches.

## 5 CONCLUSIONS

The proposed study configures the hyperparameters of the LSTM network for demand forecasting of retail products. The methodology can effectively set up the LSTM network to learn patterns of the time series data and generate the forecast. To further improve forecasting accuracy and network performance, we have incorporated a hybrid BO and GA with elitism for hyperparameter optimization. We combined the learning strengths of two well-known approaches within the optimization domain. These observations necessitate the significance of the second stage in the hybrid approach to configuring the LSTM network for error minimization objectives. Other meta-heuristics approaches, such as ant colony optimization, particle swarm intelligence, etc., can be explored. The hybrid strategy can be extended to hyperparameter optimization of machine learning objectives other than retail demand forecasting algorithms and stochastic learning methods. Although the LSTM network acts as a benchmark model with promising results, the optimal design of the neural network architecture is still an appealing research direction to explore.

## REFERENCES

Abbasimehr, H., Shabani, M., and Yousefi, M. (2020). An optimized model using lstm network for demand forecasting. *Computers & industrial engineering*, 143:106435.

Balandat, M., Karrer, B., Jiang, D. R., Daulton, S., Letham, B., Wilson, A. G., and Bakshy, E. (2019). Botorch: Programmable bayesian optimization in pytorch. *arXiv preprint arXiv:1910.06403*, 117.

Beheshti, Z. and Shamsuddin, S. M. H. (2013). A review of population-based meta-heuristic algorithms. *Int. J. Adv. Soft Comput. Appl*, 5(1):1–35.

Cansu, T., Kolemen, E., Karahasan, Ö., Bas, E., and Egrioglu, E. (2023). A new training algorithm for long short-term memory artificial neural network based on particle swarm optimization. *Granular Computing*, pages 1–14.

Carbonneau, R., Laframboise, K., and Vahidov, R. (2008).

Application of machine learning techniques for supply chain demand forecasting. *European journal of operational research*, 184(3):1140–1154.

Chollet, F. et al. (2015). Keras. https://keras.io.

da Fonseca Marques, R. A. (2020). A comparison on statistical methods and long short term memory network forecasting the demand of fresh fish products.

Deb, K. and Jain, H. (2013). An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: solving problems with box constraints. *IEEE transactions on evolutionary computation*, 18(4):577–601.

Elsken, T., Metzen, J. H., and Hutter, F. (2019). Neural architecture search: A survey. *The Journal of Machine Learning Research*, 20(1):1997–2017.

Feurer, M. and Hutter, F. (2019). *Hyperparameter Optimization*, pages 3–33. Springer International Publishing, Cham.

Fildes, R., Ma, S., and Kolassa, S. (2022). Retail forecasting: Research and practice. *International Journal of Forecasting*, 38(4):1283–1318.

Fortin, F.-A., De Rainville, F.-M., Gardner, M.-A. G., Parizeau, M., and Gagné, C. (2012). Deap: Evolutionary algorithms made easy. *The Journal of Machine Learning Research*, 13(1):2171–2175.

Frazier, P. I. (2018). A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*.

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., and Schmidhuber, J. (2016). Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232.

Johannesen, N. J., Kolhe, M., and Goodwin, M. (2019). Relative evaluation of regression tools for urban area electrical energy demand forecasting. *Journal of cleaner production*, 218:555–564.

Johnson, A. (2017). Common problems in hyperparameter optimization. *Blog. sigopt. com*.

Kumar, A., Shankar, R., and Aljohani, N. R. (2020). A big data driven framework for demand-driven forecasting with effects of marketing-mix variables. *Industrial marketing management*, 90:493–507.

Kumar, P., Batra, S., and Raman, B. (2021). Deep neural network hyper-parameter tuning through twofold genetic approach. *Soft Computing*, 25:8747–8771.

Lang, S., Steiner, W. J., Weber, A., and Wechselberger, P. (2015). Accommodating heterogeneity and nonlinearity in price effects for predicting brand sales and profits. *European Journal of Operational Research*, 246(1):232–241.

Martinez-de Pison, F., Gonzalez-Sendino, R., Aldama, A., Ferreiro-Cabello, J., and Fraile-Garcia, E. (2019). Hybrid methodology based on bayesian optimization and ga-parsimony to search for parsimony models by combining hyperparameter optimization and feature selection. *Neurocomputing*, 354:20–26.

Parmezan, A. R. S., Souza, V. M., and Batista, G. E. (2019). Evaluation of statistical and machine learning models for time series prediction: Identifying the state-of-the-art and the best conditions for the use of each model. *Information sciences*, 484:302–337.

Ramos, P., Santos, N., and Rebelo, R. (2015). Performance of state space and arima models for consumer retail sales forecasting. *Robotics and computer-integrated manufacturing*, 34:151–163.

Reimers, N. and Gurevych, I. (2017). Optimal hyperparameters for deep lstm-networks for sequence labeling tasks. *arXiv preprint arXiv:1707.06799*.

Sagheer, A. and Kotb, M. (2019). Time series forecasting of petroleum production using deep lstm recurrent networks. *Neurocomputing*, 323:203–213.

Taylor, J. W. (2008). A comparison of univariate time series methods for forecasting intraday arrivals at a call center. *Management Science*, 54(2):253–265.

Viola, R., Martin, A., Morgade, J., Masneri, S., Zorrilla, M., Angueira, P., and Montalbán, J. (2020). Predictive cdn selection for video delivery based on lstm network performance forecasts and cost-effective trade-offs. *IEEE Transactions on Broadcasting*, 67(1):145–158.

Wirsansky, E. (2020). *Hands-on genetic algorithms with Python: applying genetic algorithms to solve real-world deep learning and artificial intelligence problems*. Packt Publishing Ltd.