Research Article/Araştırma Makalesi

Gazi AKADEMİK YAYINCILIK

GAZİ
JOURNAL OF ENGINEERING SCIENCES

A Framework for Parametric Model Selection in Time Series Problems

Muhammed Abdulhamid Karabıyık[*a]

Submitted: 17.11.2023 Revised: 03.12.2023 Accepted: 04.12.2023 doi:10.30855/10.30855/gmbd.0705S09

ABSTRACT

People make future plans with the aim of simplifying their lives, and these plans are essential for preparing for forthcoming challenges. Forecasting methodologies take precedence in order to anticipate and plan for future events. Time series data stands out as a pivotal information type employed for predicting the future. This research introduces a framework for selecting the optimal model among classical artificial neural networks in time series forecasting. The classical artificial neural networks considered encompass the LSTM, CNN, and DNN models. The framework employs various parameters – including the dataset, model depth, loss functions, minimal success rate in model performance, epochs, and optimization algorithms – to determine the best-fitting model. Users have the flexibility to adjust these parameters to address specific issues. By default, the framework incorporates seven distinct loss functions and five optimization algorithms to facilitate model selection. The mean average error loss function is used as the metric for evaluating model performance. To validate the framework, Brent oil prices were utilized as the dataset in a series of tests, encompassing a total of 9000 daily price data points. The dataset was partitioned into 80\% for training and 20\% for testing purposes. The training iterations within the framework were 50 epochs. In the test scenarios, the price for the eighth day was predicted using price data from the preceding seven days. Consequently, a mean average error score of 1.1239657 was achieved. The results showed that the LSTM model, comprising two layers, the Adadelta optimization algorithm, and the mean square error loss function, emerged as the most successful configuration.

Keywords: Time series, lstm, cnn, dnn, forecasting.

[a,*] Niğde Ömer Halisdemir University,
Bor Vocational School,
Dept. of Computer Programming
51700 - Niğde, Türkiye
Orcid: 0000-0001-7927-8790
e mail: abdulhamidkarabiyik@ohu.edu.tr

*Corresponding author:
abdulhamidkarabiyik@ohu.edu.tr

Zaman Serisi Problemlerinde Parametrik Model Seçimi İçin Bir Çerçeve

ÖZ

İnsanlar yaşamlarını kolaylaştırmak için geleceğe yönelik planlamalar yapmaktadır. Bu planlamalar gelecekte karşılaşabilecek problemlere hazırlıklı olmak için önemlidir. Geleceğe yönelik hazırlıklar yapılabilmesi için de tahmin yöntemleri ön plana çıkmaktadır. Geleceğe yönelik tahminler için kullanılan verilerden birisi de zaman serileridir. Bu çalışmada zaman serisi tahminlerinde kullanılacak klasik yapay sinir ağları için en iyi modeli seçen bir çerçeve geliştirilmiştir. Klasik yapay sinir ağları olarak LSTM, CNN ve DNN modelleri kullanılmaktadır. Framework en iyi modeli seçmek için veri seti, model derinliği, kayıp fonksiyonları, model performansında minimum başarı oranı, tekrar sayısı ve optimizasyon algoritmalarını parametre olarak kullanmaktadır. Kullanıcılar bu parametreleri kendi problemlerine uygun güncelleyebilmektedir. Model seçimi içinse varsayılan olarak 7 farklı kayıp fonksiyonu ve 5 farklı optimizasyon algoritması kullanmaktadır. Model performansları Mean Avarage Error kayıp fonksiyonuyla belirlenmektedir. Framework deneylerinde, veri seti olarak Brent Ham Petrol fiyatları kullanılmış olup veri seti 9000 günlük fiyat bilgisi içermektedir. Veri seti %80 eğitim ve %20 test olarak iki bölünmüştür. Çerçeve testindeyse eğitimler 50 tekrar ile gerçekleştirilmiştir. Deneyde 7 günlük ardışık fiyat bilgisiyle 8. gündeki fiyat tahmin ettirilmiştir. Sonuç olarak 1.1239657 Mean Average Error skoru elde edilmiştir. En başarılı model, 2 katmanlı Adadelta optimizasyon algoritmasını ve Mean Square Error kayıp fonksiyonu kullanan LSTM modeli olmuştur.

Anahtar Kelimeler: Zaman serileri, lstm, cnn, dnn, tahminleme

To cite this article: M. A. Karabıyık, "A Framework for Parametric Model Selection in Time Series Problems," *Gazi Journal of Engineering Sciences,* vol.9, no.4, pp. 82-91, 2023. doi:10.30855/10.30855/gmbd.0705S09

1. Introduction

Deep learning has recently garnered considerable attention as a machine learning technique for addressing various issues. It has been effective in a variety of industries, attributed to its capability to discern complex data structures automatically. This approach is particularly employed due to its ability to handle huge datasets [1]. Nonetheless, an important factor, namely model selection, profoundly influences the application of deep learning. Choosing an appropriate deep learning model requires a careful consideration of both the problem's characteristics and the data [2]. Moreover, the performance of the chosen model is significantly impacted by the proper setup of its parameters. As a result, choosing the right model and the set of parameters for a given problem has become a key research domain for researchers.

There are several techniques available in the literature for the selection of deep learning models. Okewu et al. focused on enhancing a deep learning model by optimizing both loss functions and neural network parameters using a meta-heuristic search algorithm [3]. Meanwhile, Srivastaca et al. approached model selection by evaluating the performance of different architectures, such as ResNet and MobileNet, for the task of face recognition [4]. Kotthoff et al. conducted hyperparameter optimization for machine learning models using Bayesian optimization [5]. Taylor et al. introduced an approach for model selection that addresses the complexities of selecting deep learning models in embedded systems [6]. Additionally, Bertrand et al. explored Bayesian model selection through the integration of hyperparameter optimization and model selection [7]. Gharibi et al. developed a system called ModelKB and conducted model experiments in a self-sufficient environment [8]. Murdock et al. proposed a method for simultaneous regularization and model selection that involves teaching model architecture and parameters together [9].

In this study, we present a framework for objectively selecting the most suitable deep learning model. Abbreviations of technical terms are provided upon their first usage. This framework builds upon the foundational deep learning models such as Long Short-Term Memory (LSTM), Convolutional Neural Network (CNN), and Deep Neural Network (DNN) to facilitate well-informed decision-making. These models are chosen for their ability to producing successful results across diverse problem sets. Developed framework empowers users to tailor personalized models by using Tensorflow and Keras libraries [10], [11]. Moreover, it offers the flexibility to expand the model's structure based on user-specified parameters. The effectiveness of each model is evaluated using a variety of metrics.

The study focuses on time series analysis, a technique widely applied across various domains such as meteorology, medicine, and economics. This analytical approach serves the purpose of identifying evolving trends, aiding decision-making, and facilitating future forecasting. The dataset under examination pertains to Brent crude oil pricing, spanning a 9000-day period. This dataset was partitioned into 80% for training and 20% for testing. The experimental results are derived from this dataset, with the objective being the prediction of the 8th-day price based on a 7-day historical price sequence. The framework incorporates multiple parameters, providing users the ability to create personalized models. These parameters comprise of loss functions, optimization algorithms, training epochs, and minimum accuracy. Importantly, these parameters play a vital role in the model selection process within the framework.

As a result, a total of 120 distinct models were generated, with the most optimal model being identified as the 2-layer LSTM model. This particular model was trained by utilizing the Adadelta optimization algorithm, while the mean squared error was adopted as the designated loss function. With a training duration of 50 epochs, the resulting average absolute error score was calculated as 1.1239657. In conclusion, this study demonstrated the effectiveness of a framework for deep learning model selection and its application. The proposed framework allows users to select and customize models suitable for different problem sets, and the experimental results confirm the effectiveness and success of the proposed framework. Our approach holds the potential to enhance the efficiency of deploying deep learning models.

2. Methodology

In this study, the method of the developed framework comprises the subsequent steps: Initially, experiments are performed on diverse deep learning models, containing LSTM, CNN, and DNN. Throughout these experiments, the models undergo training for a designated number of training cycles employing distinct loss functions and optimization algorithms. This procedure produces a collection of models. From this collection, the model exhibiting the lowest mean absolute error (MAE) score is identified as the most effective model.

The employed parameters are presented in Table 1.

Table 1. Framework parameters and descriptions.

Parametres	Default Values	Descriptions
Dataset	-	Changeble for every problem so user must define Dataset.
Models	CNN, LSTM,DNN	It can be changed according to the type of problem. The users can use their cutom models.
Epochs	50	Users can adjust it according to their preferences.
Loss Functions	MAE,MSE, MASE	Users can adjust it according to their preferences.
Optimizers	Adam, Adadelta,Adamax	All optimizers supported by the Keras library can be used.
Success Rate	Avarage of Training Set Values	Users can use this parameter if they prefer models below a certain success rate.

2.1. Dataset

Brent crude oil prices serve as the dataset, containing a comprehensive collection of 9000 daily price data points. Upon analyzing the distribution of these prices, it becomes evident that the lowest price is 9.1 USD, while the highest price is 143.95 USD. The mean price is 48.421 USD. This considerable price range significantly influenced the choice of this dataset, ensuring the inclusion of diverse price trajectories over time. To facilitate effective modeling, the dataset was partitioned, allocating 80% for the training set and the remaining 20% for the test set. The graphical representation of price fluctuations within the training and test sets is illustrated in Figure 1.

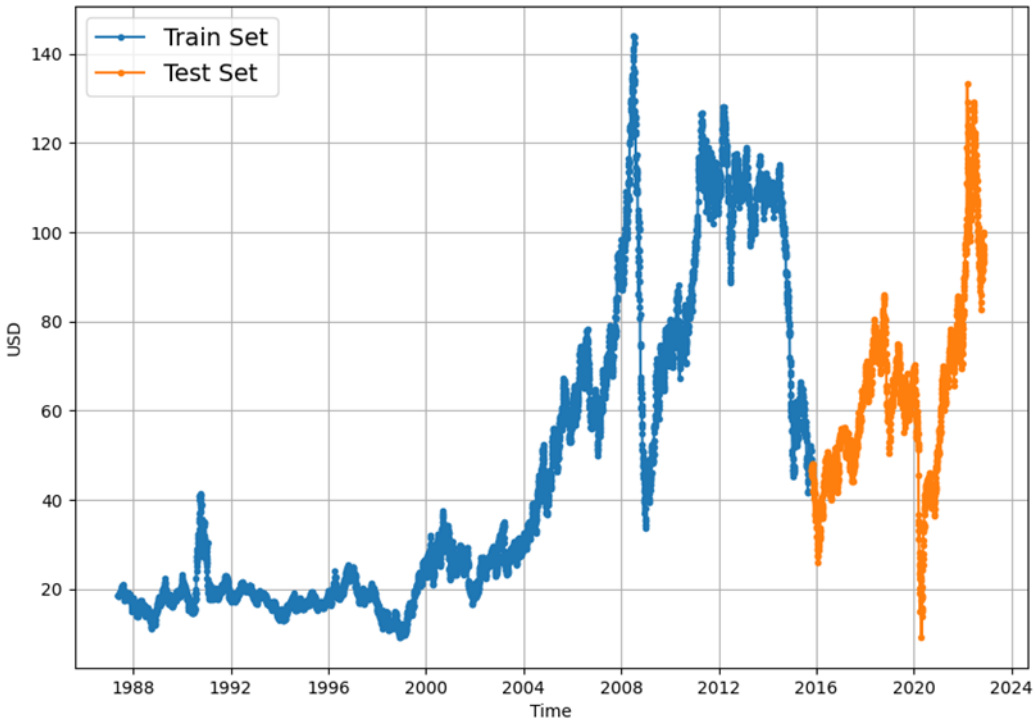

Figure 1. Training set and test set graph.

2.2. Deep learning models

Convolutional Neural Networks (CNN), a fundamental method in deep learning, is notably efficient for analyzing visual data. CNNs generate feature maps by employing filters in layers and achieve results by integrating these features in alignment with the desired output [12]. Long Short Term Memory (LSTM) is utilized for analyzing sequential data. The model operates through input, output, and forget gates, enabling it to scrutinize long-term connections and comprehend extended sequences of sequential data [13]. Deep Neural Networks (DNN) form the basis for general deep learning models. These models consist of layers of artificial neural network cells with connections established by weight values. The weight values are learned to address problems [14].

2.3. Loss functions

This study assesses the impact of loss functions on the training of models employed for regression problems. The selection of loss functions was aligned with the respective model and evaluated during the training of the models. They represent a crucial component in the training of deep learning models. Following loss functions were employed:

- Mean Squared Error
- Root Mean Squared Error
- Mean Absolute Error
- Mean Absolute Percentage Error
- Mean Squared Logarithmic Error
- Cosine Similarity
- Log Cosh Error

The MAE is a measure of success calculated by averaging the absolute differences between the actual values and model results [15]. Equation 1 demonstrates the formula for MAE.

$$\sum_{i=1}^{D} |X_i - Y_i| \tag{1}$$

The MSE is a measure of success calculated as the mean of the squares of the differences between the model results and the actual values [16]. Equation 2 demonstrates the formula for MSE.

$$\sum_{i=1}^{D} (X_i - Y_i)^2 \tag{2}$$

The RMSE is calculated by taking the square root of the MSE value and shows the average of the errors between the predicted values and the actual values [17]. Equation 3 demonstrates the formula for RMSE.

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{d_i - f_i}{\partial_i}\right)^2} \tag{3}$$

MAPE calculates the error between model predictions and actual values as a percentage [18]. Equation 4 demonstrates the formula for MAPE.

$$\frac{100\%}{N}\sum_{i=0}^{N-1}\frac{y_i - \hat{y}_i}{y_i} \tag{4}$$

MASE assesses model effectiveness using MAE values that have been normalized to the dataset's characteristics. MASE enables the assessment of errors in the training history [19]. Equation 5 demonstrates the formula for MASE.

$$\frac{\frac{1}{N}\sum_{i=0} N - 1|y_i - \hat{y}_i|}{\frac{1}{N-1}\sum_{i=1}^{N-1}|y_i - y_{i-1}|} \tag{5}$$

2.4. Optimization algorithms

Choice of the optimization algorithm is a crucial step for effective training of models. Following optimization algorithms were considered:

- Adam
- Stochastic Gradient Descent
- Adadelta
- Nadam
- Adamax

These optimization algorithms were applied to mitigate the error rates derived from the loss functions. The selection of the appropriate optimization algorithm was determined with consideration of the problem nature and the underlying data structure. This selection aimed to enhance the optimization of deep learning models, thereby facilitating improved understanding and prediction of price fluctuations within the dataset.

Adam is a gradient-based optimisation algorithm commonly utilised in deep learning models. It was initially proposed by Kingma and Ba in 2014. Adam adjusts the learning rate through the use of moving averages of the gradient momentum and the square of the gradient, facilitating accelerated learning and decreasing the necessity for hyperparameter tuning [20].

Stochastic Gradient Descent (SGD) is a fundamental optimisation algorithm widely used in machine learning and deep learning models. SGD learns by updating the weights for each data sample. However, this can cause fluctuations and may require some modifications to ensure faster convergence [21].

Adadelta is an optimisation algorithm that attempts to overcome the disadvantages of SGD without the need to specifically set the learning rate hyperparameter. Adadelta automatically adjusts the learning rate using moving averages of the gradients. This can provide better convergence to the model [22].

Nadam is an optimisation approach that combines the Nesterov Momentum and Adam algorithms. Nesterov Momentum employs a technique that encourages swift convergence in momentum-based optimisation algorithms, while also maintaining the adaptive properties of Adam. Consequently, Nadam frequently attains rapid and steady convergence [23].

Adamax is a variant of the Adam optimization algorithm. It works similarly to Adam but employs the unbounded norm of the gradient values instead of moving averages. This approach can result in superior performance, particularly when the L2 norm is significant [24].

2.5. The Framework

The primary objective of the framework is to generate diverse models by replicating the layers of the input model. These generated models undergo training phase using a variety of loss functions and optimization algorithms. Finally, the framework assesses the performance of these models resulting from different parameter combinations.

The framework systematically gathers and compares the performance scores of the models generated through varied training processes. These scores effectively gauge the predictive capability of each model, aiding in the identification of the most suitable one. Thus, the framework identifies and presents the model that aligns best with the user-specified criteria.

The fundamental workflow of the framework is illustrated in Figure 2. It depicts the creation of diverse models by replicating input model layers and employing diverse loss functions and optimization algorithms. Subsequently, these models are evaluated during the training processes to determine the optimal choice. The framework's objective is to attain optimal results by promoting model diversity through systematic experimentation with distinct parameters.

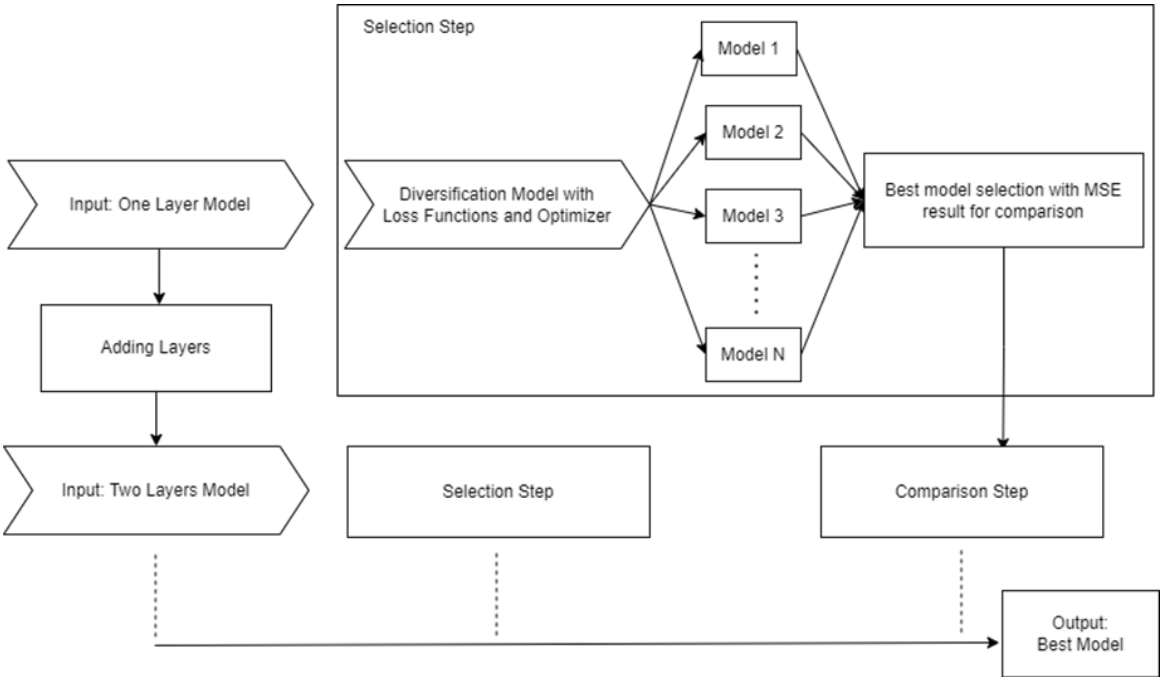

Figure 2. Workflow of the framework.

The framework presents a structured approach for the selection and development of models. Users have the flexibility to input models of their preference into the framework, thereby optimizing training costs for models tailored to specific problems. Developed by utilizing the Tensorflow and Keras libraries, the framework accepts models that are created using Keras' layer class as its input. By default, it employs CNN, DNN, and LSTM models, chosen based on their proven efficacy across diverse problem domains.

Multiple versions of the model can be provided to the framework via its parameters. These variations are created by implementing different loss functions and optimization algorithms. To determine the total number of models, the product of NL (number of layers), NLF (number of loss functions), and NOA (number of optimization algorithms) must be calculated. The number of divergent models generated can be determined using the formula outlined in Equation 6.

$$Number\ of\ Models = NL * NOA * NLF \tag{6}$$

A distinctive diversification process is executed for each model and layer provided as a parameter, considered a measure towards achieving diversity. The results of diversification are evaluated based on the Mean Absolute Error (MAE) results for the respective model. The framework utilises the early stopping function from the Keras library to address overfitting and underfitting situations that may arise during model training. Therefore, the outcomes acquired from the models are prepared for comparison. The most suitable model is retained in memory and compared with subsequent diversification processes. The inclusion of layers is halted if the success rate falls below that of the previous step. The framework promotes variation based on the input models, achieved through various combinations of loss functions and optimization algorithms. MAE results are utilized within the framework to assess the effectiveness of diversified versions of the models. As a result, the most optimal model is selected.

The framework selects the optimal outcome among various models and presents it to the user. Simultaneously, upon the user's request, a compilation can display optimal outcomes alongside additional successful examples. This level of flexibility empowers users to showcase and select the most suitable models.

3. Results and Discussion

The framework was tested using Brent Crude Oil prices. To introduce model diversity in the tests, a range of loss functions including MSE, MAE, and MAPE, along with optimization algorithms such as Adam, Adadelta, Nadam, and Adamax, were employed. The process of adding layers was implemented on the default LSTM, CNN, and DNN models, resulting in a foundational model comprising ten distinct layers. This foundational

model was assessed in 120 variations, encompassing different loss functions and optimization algorithms. The training process of the models involved 50 epochs. The results of the most successful model, employing 50 epochs for the ten base models, are presented in Table 2.

Table 2. The most successful model with 50 epochs for the base 10 models.

Model	Loss Function	Optimizer	MAE	MSE	RMSE	MAPE	MASE
CNN-1L	MAE	ADAMAX	1.1951666	3.2317772	1.7977145	2.0679417	1.0574669
CNN-2L	MSE	ADAMAX	1.1716549	3.0966759	1.7597374	2.020125	2.020125
CNN-3L	MSE	ADADELTA	1.2711904	3.4629989	1.8609134	2.1794298	1.1247317
DNN-1L	MSE	ADAMAX	1.2403895	3.4186473	1.8489584	2.151392	1.0974795
DNN-2L	MSE	ADADELTA	1.2403287	3.2778876	1.8104937	2.138884	1.0974256
DNN-3L	MSE	ADAMAX	1.1692218	3.074043	1.753295	2.0136263	1.0345112
DNN-4L	MSE	ADADELTA	1.1800587	3.036366	1.7425171	2.03344	1.0440996
LSTM-1L	MSE	ADADELTA	1.1503365	2.9347835	1.7131209	1.9893798	1.0178018
LSTM-2L	**MSE**	**ADADELTA**	**1.1239657**	**2.840379**	**1.6853424**	**1.9417524**	**0.99446934**
LSTM-3L	MSE	ADADELTA	1.1298846	2.903611	1.7039986	1.9573519	0.99970627

From the 120 models created, the optimal performance was exhibited by the LSTM model with two layers. The most successful model employed MSE as its loss function and Adadelta as its optimization algorithm. The results are shown in Figure 3.

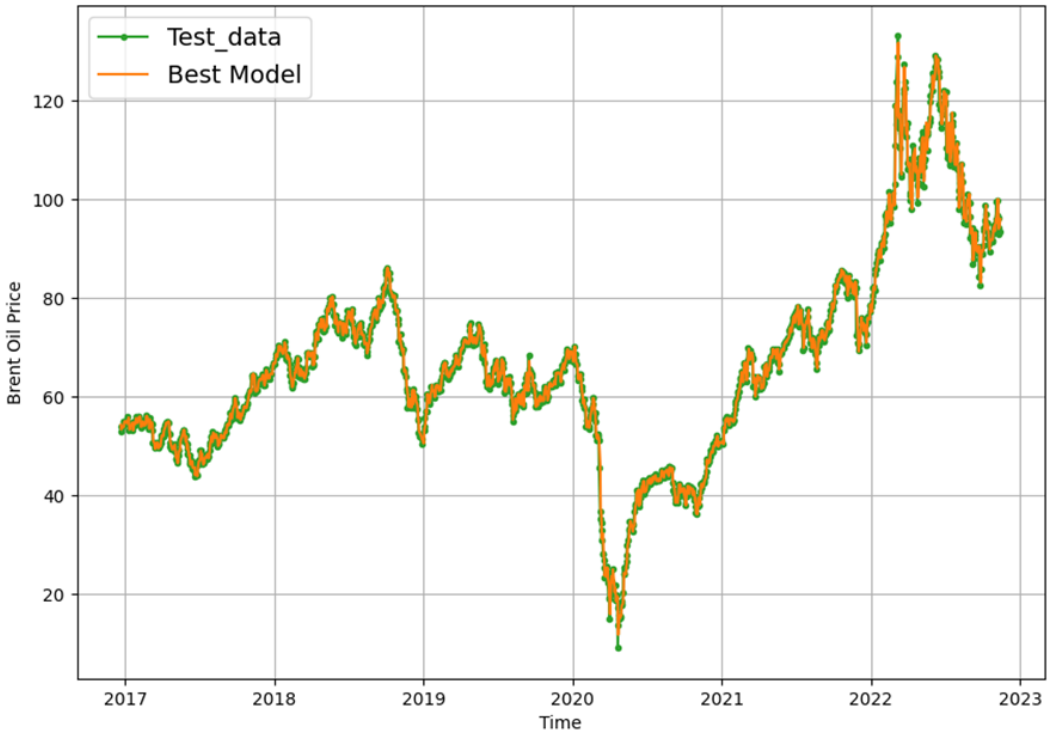

Figure 3. Best model results for Windows size 7.

The experiment entailed a window size of 7 for the series in the data set. To view the outcomes from an alternative viewpoint, the experiment was re-executed with a window size of 15. Table 3 exhibits the effects of the experiment carried out with a window size of 15.

Table 3. The most successful model with 50 epochs for the base 7 models.

Model	Loss Function	Optimizer	MAE	MSE	RMSE	MAPE	MASE
CNN-1L	MSE	ADAMAX	1.2158406	3.2258146	1.7960552	2.0856576	1.07523
CNN-2L	MSE	ADAMAX	1.470338	4.511765	2.1240916	2.5233996	1.300295
DNN-1L	MSE	ADAMAX	1.3783844	3.6747472	1.916963	2.3201127	1.2189758
DNN-2L	MSE	ADADELTA	1.2984885	3.767063	1.9408923	2.236332	1.1483197
DNN-3L	MAE	ADADELTA	1.4727191	4.197971	2.048895	2.4877105	1.3024007
LSTM-1L	**MSE**	**ADAMAX**	**1.2086239**	**3.277059**	**1.810265**	**2.0864668**	**1.0688479**
LSTM-2L	MSE	ADAMAX	1.2292047	3.4163816	1.8483456	2.1175768	1.0870485

In the second experiment, from the 96 models created, the optimal performance was exhibited by the LSTM

model with one layer. The most successful model employed MSE as its loss function and Adamax as its optimization algorithm. The results are shown in Figure 4.

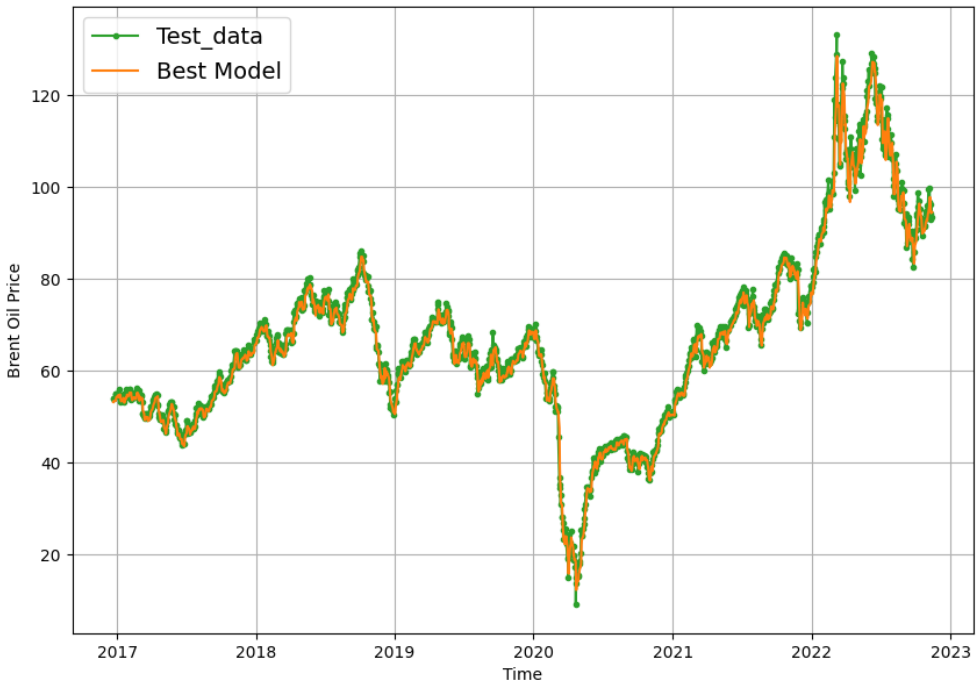

Figure 4. Best model results for Windows size 15.

4. Conclusions

In this study, we have designed a framework for the development and selection of models in the context of time series analysis. User-specified models are input into the framework, which subsequently evaluates their performance by constructing a diverse collection of models. These models, generated through a variety of loss functions and optimization algorithms, are then assessed based on their Mean Absolute Error (MAE) results. These findings offer valuable insights to users facilitating the process of model selection and refinement.

The framework has been implemented using the Tensorflow and Keras libraries, supporting tailored to diverse problem domains. By default, a comprehensive collection of models, including Convolutional Neural Networks (CNN), Deep Neural Networks (DNN), and Long Short-Term Memory (LSTM) networks, is available. The determination of diverse models depends on the combination of distinct loss functions and optimization algorithms, with the most proficient model selected the optimal choice.

This study affords users substantial ease in both model selection and performance assessment. The framework empowers users to identify appropriate models for specific problems and attain optimal results by iteratively experimenting with various loss functions and optimization algorithms. Furthermore, it offers support for different models. In particular, the default CNN, DNN, and LSTM models can be applied to wide range of problem domains.

The future perspective of the study is to extend the framework by incorporating additional models and diversification methods. Furthermore, extending the framework's domain to ensemble learning will pave the way for exploration of different research studies. In deep learning problem-solving, an array of techniques is implemented to enhance result success. However, the success strategies presented within the framework only encompass methods that cover all problems. In forthcoming research, problem-specific approaches could be integrated into the framework. To reduce the runtime cost of the framework, search algorithms that are effectively used in parameter selection can be integrated into the model in future studies. In this way, some steps that can be defined as unnecessary during model diversification can be removed from the process.

Conflict of Interest Statement

The authors declare that there is no conflict of interest.

References

[1] P. Wang, E. Fan, and P. Wang, "Comparative analysis of image classification algorithms based on traditional machine learning and deep learning," *Pattern Recognit Lett*, vol. 141, pp. 61–67, Jan. 2021, doi: 10.1016/j.patrec.2020.07.042.

[2] U. Michelucci, "A Case-Based Approach to Understanding Deep Neural Networks," in *Applied Deep Learning*, Berkeley, CA: Apress, 2018. doi: 10.1007/978-1-4842-3790-8.

[3] E. Okewu, P. Adewole, and O. Sennaike, "Experimental Comparison of Stochastic Optimizers in Deep Learning," in *Computational Science and Its Applications – ICCSA 2019*, 2019, pp. 704–715. doi: 10.1007/978-3-030-24308-1_55.

[4] Y. Srivastava, V. Murali, and S. R. Dubey, "A Performance Comparison of Loss Functions for Deep Face Recognition," *ArXiv*, Dec. 2018, doi: 10.48550/arXiv.1901.05903.

[5] L. Kotthoff, C. Thornton, H. H. Hoos, F. Hutter, and K. Leyton-Brown, "Auto-WEKA: Automatic Model Selection and Hyperparameter Optimization in WEKA," in *Automated Machine Learning*, 2019, pp. 81–95. doi: 10.1007/978-3-030-05318-5_4.

[6] B. Taylor, V. S. Marco, W. Wolff, Y. Elkhatib, and Z. Wang, "Adaptive deep learning model selection on embedded systems," *ACM SIGPLAN Notices*, vol. 53, no. 6, pp. 31–43, Dec. 2018, doi: 10.1145/3299710.3211336.

[7] H. Bertrand, R. Ardon, M. Perrot, and I. Bloch, "Hyperparameter optimization of deep neural networks: combining Hperband with Bayesian model selection," in *Conférence sur l'Apprentissage Automatique*, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:10497518.

[8] G. Gharibi, V. Walunj, R. Alanazi, S. Rella, and Y. Lee, "Automated Management of Deep Learning Experiments," in *Proceedings of the 3rd International Workshop on Data Management for End-to-End Machine Learning*, New York, NY, USA: ACM, Jun. 2019, pp. 1–4. doi: 10.1145/3329486.3329495.

[9] C. Murdock, Z. Li, H. Zhou, and T. Duerig, "Blockout: Dynamic Model Selection for Hierarchical Deep Networks," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jun. 2016, pp. 2583–2591. doi: 10.1109/CVPR.2016.283.

[10] T. Developers, "TensorFlow," *Zenodo*, May 2021, doi: 10.5281/zenodo.4758419.

[11] N. Ketkar, "Introduction to Keras," in *Deep Learning with Python*, Berkeley, CA: Apress, 2017, pp. 97–111. doi: 10.1007/978-1-4842-2766-4_7.

[12] L. O. Chua and T. Roska, "The CNN paradigm," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 40, no. 3, pp. 147–156, Mar. 1993, doi: 10.1109/81.222795.

[13] A. Sherstinsky, "Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network," *Physica D*, vol. 404, p. 132306, Mar. 2020, doi: 10.1016/j.physd.2019.132306.

[14] G. Li *et al.*, "Understanding error propagation in deep learning neural network (DNN) accelerators and applications," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, New York, NY, USA: ACM, Nov. 2017, pp. 1–12. doi: 10.1145/3126908.3126964.

[15] J.-H. Lin, T. M. Sellke, and E. J. Coyle, "Adaptive stack filtering under the mean absolute error criterion," *IEEE Trans Acoust*, vol. 38, no. 6, pp. 938–954, Jun. 1990, doi: 10.1109/29.56055.

[16] A. A. Poli and M. C. Cirillo, "On the use of the normalized mean square error in evaluating dispersion model performance," *Atmospheric Environment. Part A. General Topics*, vol. 27, no. 15, pp. 2427–2434, Oct. 1993, doi: 10.1016/0960-1686(93)90410-Z.

[17] W. Wang and Y. Lu, "Analysis of the Mean Absolute Error (MAE) and the Root Mean Square Error (RMSE) in Assessing Rounding Model," *IOP Conf Ser Mater Sci Eng*, vol. 324, p. 012049, Mar. 2018, doi: 10.1088/1757-899X/324/1/012049.

[18] J. McKenzie, "Mean absolute percentage error and bias in economic forecasting," *Econ Lett*, vol. 113, no. 3, pp. 259–262, Dec. 2011, doi: 10.1016/j.econlet.2011.08.010.

[19] P. H. Franses, "A note on the Mean Absolute Scaled Error," *Int J Forecast*, vol. 32, no. 1, pp. 20–22, Jan. 2016, doi: 10.1016/j.ijforecast.2015.03.008.

[20] Z. Zhang, "Improved Adam Optimizer for Deep Neural Networks," in *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*, IEEE, Jun. 2018, pp. 1–2. doi: 10.1109/IWQoS.2018.8624183.

[21] S. Ruder, "An overview of gradient descent optimization algorithms," *ArXiv*, Sep. 2016, doi: 10.48550/arXiv.1609.04747.

[22] E. M. Dogo, O. J. Afolabi, N. I. Nwulu, B. Twala, and C. O. Aigbavboa, "A Comparative Analysis of Gradient Descent-Based

Optimization Algorithms on Convolutional Neural Networks," in *2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS)*, IEEE, Dec. 2018, pp. 92–99. doi: 10.1109/CTEMS.2018.8769211.

[23] Q. Zhang *et al.*, "Boosting Adversarial Attacks with Nadam Optimizer," *Electronics (Basel)*, vol. 12, no. 6, p. 1464, Mar. 2023, doi: 10.3390/electronics12061464.

[24] R. Llugsi, S. El Yacoubi, A. Fontaine, and P. Lupera, "Comparison between Adam, AdaMax and Adam W optimizers to implement a Weather Forecast based on Neural Networks for the Andean city of Quito," in *2021 IEEE Fifth Ecuador Technical Chapters Meeting (ETCM)*, IEEE, Oct. 2021, pp. 1–6. doi: 10.1109/ETCM53643.2021.9590681.

** This paper was presented at the 5th International Conference on Artificial Intelligence and Applied Mathematics in Engineering (ICAIAME 2023) and the abstract was published as an e-book.*