Using Monte-Carlo Dropout in Deep Neural Networks for Interval Forecasting of Durian Export

PATCHANOK SRISURADETCHAI[1], WIKANDA PHAPHAN[2]
[1]Department of Mathematics and Statistics,
Thammasat University,
Khlong Luang, Pathum Thani, 12120,
THAILAND

[2]Department of Applied Statistics,
King Mongkut's University of Technology North Bangkok,
Bangsue, Bangkok 10800
THAILAND

Abstract: - Interval forecasting is essential because it presents predictions with associated uncertainties, which are not captured by point forecasts alone. In nature, data contain variability due to measurement and random noise. In machine learning, most research focuses on point forecasts, with relatively few studies dedicated to interval forecasting, especially in areas such as agriculture. In this study, durian exports in Thailand are used as a case study. We employed Monte Carlo Dropout (MCDO) for interval forecasting and investigated the impact of various hyperparameters on the performance of Monte Carlo Dropout Neural Networks (MCDO-NNs). Our results were benchmarked against traditional models, such as the Seasonal Autoregressive Integrated Moving Average (SARIMA). The findings reveal that MCDO-NN outperforms SARIMA, achieving a lower root mean squared error of 9,570.24 and a higher R-squared value of 0.4837. The interval forecast width obtained from the MCDO-NN was narrower compared to that of SARIMA. Also, the impact of hyperparameters was observed, and it can serve as guidelines for applying MCDO-NNs to other agricultural datasets or datasets with seasonal and/or trend components.

Key-Words: - Monte-Carlo dropout, SARIMA, interval forecast, neural networks, regularization, deep learning.

Received: February 23, 2023. Revised: November 25, 2023. Accepted: December 13, 2023. Published: February 28, 2024.

1 Introduction

The durian industry in Thailand has rapidly grown owing to the country's suitable climate and fertile soils, which are essential for agriculture. The area for growing durian has also increased, especially in important areas such as Rayong and Chanthaburi, [1]. In Thailand, the durian export has been driven by both governmental backing and the surging demand in global markets, particularly China, which serves as a major destination for exports, [2]. Thailand's durian, which has been a significant export since 2014, is the country's most profitable fruit export. In May 2021, it achieved exceptional sales of USD 934.9 million, setting a new record. China receives approximately 70% of Thailand's durian exports, [3]. In the third quarter of 2023, the export value of fresh, chilled, frozen, and dried fruits experienced a year-on-year growth rate of 61.4%. The heightened demand from Chinese consumers, which surged after the complete reopening of the country, contributed to a portion of this rise. The China-Laos high-speed railway also positively impacted fruit exports; this railway to Kunming in Yunnan Province, China, significantly reduced transportation costs and time, with the journey now taking approximately ten hours, [4].

This study aims to leverage the neural network architectures, specifically the one-head output model, for interval predicting the performance of durian in Thailand's exports because a single value of forecast cannot capture uncertainty in the future, [5]. In doing so, the study goes beyond conventional forecasting methods and incorporates Monte Carlo dropout (MCDO) techniques. This aligns with recent developments in neural network research, such as Bayesian deep learning and dropout techniques, [6], research into uncertainties in deep learning, [7], and MCDO by limiting dropout layers to later stages of the neural network and using this modified method for sorting radio frequency transmitters, [8]. Furthermore, some studies have

shown that using dropout as a Bayesian estimator will improve the ability of deep neural networks to generalize in the context of speech improvement, [9]. One study showed the effectiveness of MCDO in terms of precision and reliability for categorizing rock facies, [10]. Researchers evaluated the integration of MCDO with the residual network using the C-MAPSS dataset, [11]. Also, there is a study showing a workable way to use MCDO to approximate Bayesian neural networks and effectively capture the uncertainty in satellite telemetry time series, [12] and a new method for landmark detection employed the MCDO technique in a U-shaped convolutional neural network, [13].

Despite its success in various applications, time series forecasting has underutilized MCDO, particularly for interval forecasts in agricultural areas where time series data usually contains the seasonal component. We believe that MCDO has the potential to improve the accuracy of interval-predicting data with seasonality. In this paper, durian export values are used as a case study. In addition, we aim to explore the impact of hyperparameters on forecasting accuracy and to compare the accuracy of the MCDO-NN with that of the Seasonal Autoregressive Integrated Moving Average (SARIMA) model.

2 Related Theories

This section will describe the importance of uncertainty in forecasting, Bayesian neural networks, MCDO-NN, model topology, and the SARIMA model.

2.1 Uncertainty in Forecasting

In a one-head output MCDO-NN, aleatoric uncertainty is captured directly through the network's predictive variance. To do this, we train the network with dropout and then enable multiple forward passes with dropout at inference time. The variance observed in these predictions serves as an estimate of aleatoric uncertainty. Each forward pass with dropout will simulate a sample from the posterior predictive distribution, and the variance among these samples reflects the uncertainty inherent in the data, [14].

The mean squared error (MSE) is a standard loss function for regression tasks. It measures the average of the squares of the errors or the average squared difference between the estimated values and the actual value. In the context of an MCDO-NN, the MSE can be defined as:

$$L_{NN} = \frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{y}_i - f(\mathbf{x}_i)\right\|^2 . \quad (1)$$

For modeling aleatoric uncertainty, we can modify the MSE loss function to include a variance term. This will result in a weighted loss function where the inverse of the predicted variance weights the squared errors:

$$L_1 = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\sigma^2(\mathbf{x}_i)}\left\|\mathbf{y}_i - f(\mathbf{x}_i)\right\|^2 . \quad (2)$$

To prevent the network from predicting infinite variance, the following regularization term is included for the variance.

$$L_2 = \log\sigma^2(\mathbf{x}_i) \quad (3)$$

The total loss function combines the above two components:

$$L_{NN}(\theta) = \left(L_1 + L_2\right)/2 . \quad (4)$$

By minimizing this loss function during training, the neural network learns to predict both the mean and variance of the data, [15], [16], [17], [18].

Epistemic uncertainty arises from the model's lack of knowledge and can be due to limited data or an imperfect model. Within a one-head output MCDO framework, estimating epistemic uncertainty involves calculating the variance in the network's predictions across multiple stochastic forward passes. However, it is crucial to distinguish this from aleatoric uncertainty. Epistemic uncertainty reflects the variability in the model's predictions due to different "thinnings" (dropout configurations) of the network, essentially different sub-models being sampled each time, [19].

2.2 Bayesian Neural Networks

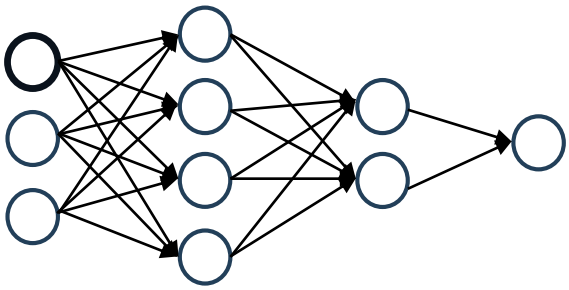
Bayesian Neural Networks (BNNs) combine Bayesian probability theory and neural networks. BNNs, in contrast to conventional NNs, consider weights as random variables with probability distributions. This enables the model to incorporate uncertainty into its predictions, [20]. In a BNN, each weight w is associated with a prior distribution $P(w)$, which captures our initial assumptions about the network's parameters. After observing data D, we update our beliefs to form the posterior distribution $P(w|D)$ using Bayes' theorem:

$$P(w|D)=\frac{P(D|w)P(w)}{P(D)}, \quad (5)$$

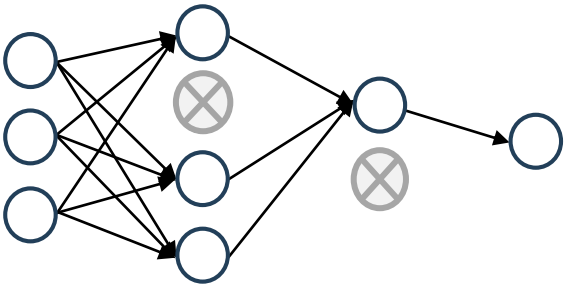
where $P(D|w)$ is the likelihood of the data given the weights, and $P(D)$ is the evidence or marginal likelihood of the data, which serves as a normalizing constant. The predictive distribution for a new data point x^* can then be obtained by marginalizing the weights:

$$P\left(y^*\middle|x^*,D\right)=\int P\left(y^*\middle|x^*,w\right)P(w|D)\,dw. \quad (6)$$

While BNNs approach this through the formal framework of Bayesian inference, MCDO provides a more computationally tractable approximation. MCDO provides a computationally tractable approximation for implementing a form of Bayesian inference in NNs, avoiding the computational complexity of true Bayesian methods like Markov Chain Monte Carlo (MCMC) sampling, commonly used in BNNs, [21].



(a) Original neural network.



(b) Dropout neural network.

Fig. 1: The original and dropout neural networks

2.3 Monte-Carlo Dropout

The MCDO helps approximate the posterior distribution $P(w|D)$. During training, dropout randomly sets a subset of the weights to zero, effectively sampling from a distribution over a possible sub-network. This process samples from an approximate posterior distribution, [6]. At test time, MCDO performs multiple forward passes with dropout enabled, each time randomly dropping out different neurons. These random forward passes result in different sets of active weights. This is similar to taking samples from the weights' posterior distribution, [22], [23]. By aggregating the outputs from multiple stochastic forward passes, MCDO estimates the predictive distribution $P\left(y^*\middle|x^*,D\right)$. It is thought that the empirical mean of these outputs is about the same as the predictive mean, and the empirical variance gives an idea of the predictive uncertainty, which includes both the aleatoric and epistemic uncertainty, [7].

Figure 1 presents the original and dropout neural networks. The top subfigure presents a fully connected 3-4-2-1 neural network architecture without dropout. The bottom subfigure illustrates the same network with dropout applied during two separate forward passes.

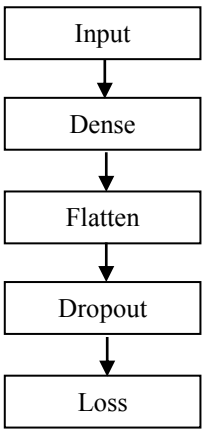


Fig. 2: Neural network architecture of a single-head output

2.4 Model Topology

In our exploration of neural network architectures for forecasting, we consider the single-output network, as illustrated in Figure 2. Each component is described here:

1) Input layer: The input layer takes the feature vector $\mathbf{x}_i$ for the i-th instance. If the dataset has p features, the input layer is mathematically represented as $\mathbf{X}\in\mathbf{R}^{n\times p}$, where n is the number of instances.
2) Dense layer: A dense layer with 100 neurons can be represented as $\mathbf{W}^{(1)}\in\mathbf{R}^{n\times 100}$ for the weights and $\mathbf{b}^{(1)}\in\mathbf{R}^{100}$ for the biases. The output of this layer for each instance is a 100-dimensional vector $\mathbf{h}_i^{(1)}$, computed as:

$$\mathbf{h}_i^{(1)}=\sigma\left(\mathbf{W}^{(1)T}\mathbf{x}_i+\mathbf{b}^{(1)}\right), \quad (7)$$

where σ is a non-linear activation function such as Rectified Linear Unit (ReLU) or sigmoid.

3) Flatten Operation: If the output from the previous layer is multi-dimensional, the flatten operation reshapes it into a one-dimensional array before it is fed into the next dense layer. For instance, if the output is a two-dimensional array from a convolutional layer, flattening converts it into a one-dimensional array $\mathbf{h}^{(flat)}$.
4) Dropout: Dropout is applied to the output of a layer, which can be represented by a diagonal matrix $\mathbf{D}$ with entries being 1 (neuron is kept) or 0 (neuron is dropped) with probability p_{drop}. The output after dropout is given by:

$$\mathbf{h}_i^{(dropout)} = \mathbf{D} * \mathbf{h}_i^{(1)} \tag{8}$$

where * denotes element-wise multiplication.
5) Loss Function: The loss function measures the discrepancy between the true labels $\mathbf{Y}$ and the predicted labels $\hat{\mathbf{Y}}$. For regression tasks, a common loss function is the MSE, which is defined as in (1), [24], [25].

2.5 Seasonal Autoregressive Integrated Moving Average

The Seasonal Autoregressive Integrated Moving Average (SARIMA) model is an extension of the ARIMA (Autoregressive Integrated Moving Average) model that specifically addresses and models seasonality in time series data. SARIMA models are often denoted as SARIMA(p, d, q)(P, D, Q)$_S$, where: p, d, q are the non-seasonal components (AR order, differencing, and MA order); P, D, Q are the seasonal components; S represents the length of the seasonal cycle. The SARIMA model is defined as [5]:

$$\begin{aligned}(1-\textstyle\sum_{i=1}^{p}\phi_i L^i)(1-\textstyle\sum_{i=1}^{P}\Phi_i L^{is})(1-L)^d(1-L^s)^D y_t \\ = (1+\textstyle\sum_{i=1}^{q}\theta_i L^i)(1-\textstyle\sum_{i=1}^{Q}\Theta_i L^{is})\varepsilon_t.\end{aligned} \tag{9}$$

The parameters of SARIMA models are typically estimated using maximum likelihood estimation (MLE) or similar optimization techniques. The model's performance and accuracy depend on the correct specification of its parameters (p, d, q, P, D, Q, S). Researchers often carry out model selection using criteria like the Akaike Information Criterion (AIC) or the Bayesian Information Criterion (BIC).

3 Dataset

We obtained the monthly export figures for durian, expressed in millions of Thai Baht (THB), from the Office of Agricultural Economics of Thailand, [26]. The dataset is comprehensive and free from any missing data points. It spans the period from January 2015 to November 2023, including a total of 107 months. We partitioned the data into two distinct sets: a training set and a testing set. The testing set comprises the 12 most recent months, spanning from December 2022 to November 2023. The training set consists of 95 months, spanning from January 2015 to November 2022.

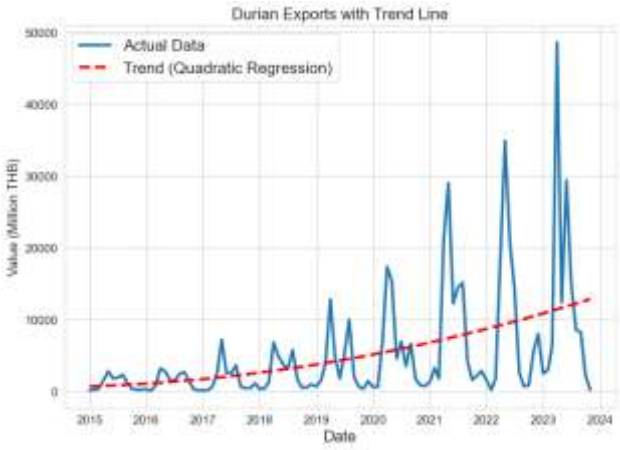


Fig. 3: Durian export with a quadratic trend line

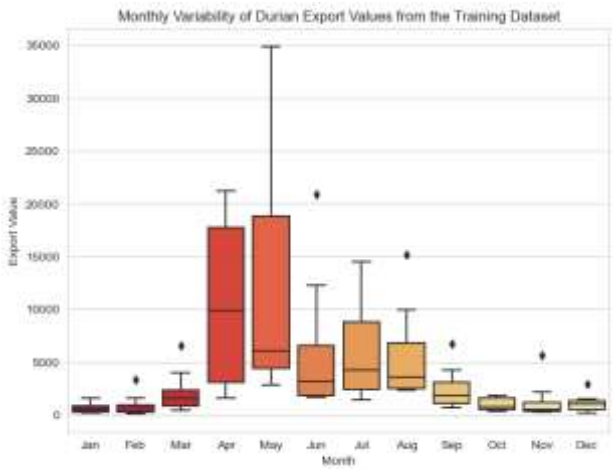


Fig. 4: Monthly variability of durian export values from the training dataset

Based on Figure 3, it appears that the dataset is most accurately represented by a multiplicative model, expressed by the formula $Y_t = T_t \times S_t \times I_t \times \varepsilon_t$. The regression line exhibits a distinct and consistent rising trajectory. Figure 4 indicates that April and May experience the highest variability in durian export values, suggesting significant fluctuations in the volume of durians exported due to factors like

seasonality, whereas months such as January, November, and December exhibit the lowest variability, as indicated by their much narrower boxplots. This variability presents challenges for forecasting, especially for high-variability months, while lower variability in other months implies more stable and predictable exports, yet it does not necessarily simplify forecasting due to potential underlying trends or shifts in the data not immediately apparent from the boxplot alone, necessitating robust forecasting models to capture any patterns or cyclicality in the export data.

4 Methodology

The methodology encompasses data preprocessing, model development, hyperparameter tuning, and validation to ensure accuracy and robustness in forecasting. The point and interval forecasts are compared to SARIMA.

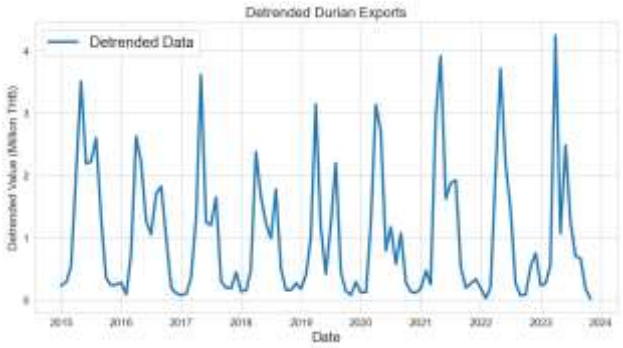


Fig. 5: Untrended durian export data

4.1 Data Preprocessing

The dataset, comprising monthly export values of durian in Thai Baht from January 2015 to November 2023, underwent preprocessing. Initial steps included parsing dates to ensure chronological consistency. To address the seasonality and trend components inherent in time series data, we applied regression techniques, thereby detrending the data as illustrated in Figure 5. Additionally, we generated lag features for up to 12 months to capture the seasonal component. This approach allowed us to integrate both recent and historical patterns into our analysis.

4.2 Hyperparameters

The hyperparameters that impact the MCDO-NN are the following:

1) Number of Neurons: The structure of neural networks was progressively expanded to explore the effect of network depth. The simplest models began with two layers containing 60-30 neurons. For models with additional layers, the first layer's neuron count was increased in a specific pattern: adding 120 neurons for three-layer models, 180 for four-layer models, and so forth, up to a maximum configuration of 300-240-180-120-60-30 for six-layer models. The increment strategy aimed to systematically increase model capacity while maintaining a consistent structure in subsequent layers, [27].
2) Activation Functions: Common choices like ReLU and sigmoid have different characteristics. The ReLU is generally preferred for its computational efficiency and ability to mitigate the vanishing gradient problem. The sigmoid function is useful for binary classification in output layers and can also be tested in hidden layers for its nonlinear properties, [28]. In our study, models were tested with different activation functions: ReLU, sigmoid, and tanh. These approaches can be useful for modeling nonlinear relationships.
3) Dropout Rate: Dropout is a regularization technique to prevent overfitting. It randomly sets a fraction of input units to zero at each update during training. For instance, in simpler models with two layers, dropout rates are 0.2–0.3. For the most complex models, the dropout rates are 0.7–0.6–0.5–0.4–0.3–0.2. The range from 0.2 to 0.7 corresponds to a light to a more aggressive dropout.
4) Epochs: The models underwent training for 25, 50, or 100 epochs. The epoch count corresponds to the total number of iterations of the learning algorithm trained for the training dataset. Increasing the number of epochs enables the model to get a deeper understanding of the data, but excessive epochs might result in overfitting. In contrast, a limited number of epochs may lead to underfitting.
5) Batch Size: Batch sizes of 16, 32, 64, or 128 were used. The batch size determines the number of samples that will be propagated through the network at a time. Smaller batch sizes often lead to a more stable convergence but can increase the training time. Larger batch sizes can speed up the training process but might lead to less stable convergence and potentially poorer generalization performance.

Each model was iteratively evaluated using MCDO with 30,000 iterations to estimate the predictive uncertainty, capturing both aleatoric and epistemic uncertainties.

4.3 Model Development and Forecast Interval

In our study, we utilize TensorFlow along with the Keras library, combining TensorFlow's advanced computational power for data processing and model training with the user-friendly neural network API of Keras. This integration significantly enhances our application of complex machine learning techniques, particularly MCOD, [29], [30].

In the preparatory phase, we established a detailed grid of hyperparameters, which included varying layer depths, neuron counts, activation functions, dropout rates, epochs, and batch sizes. This framework enabled us to thoroughly explore a wide spectrum of models, ranging from simpler structures to more complex designs adept at capturing intricate data patterns. The models were fed time series data with lags ranging from 1 to 12 months, allowing the networks to learn from past trends and patterns in durian exports.

The activation functions we selected, namely ReLU, sigmoid, and tanh, are widely recognized in neural network modeling. Each of these functions offers distinct advantages when handling non-linear data. The utilization of dropout, a crucial regularization technique, has a significant impact on our models. To mitigate overfitting, neurons are randomly deactivated during the training process, [31].

In a conventional neural network, a point forecast $\hat{y}$ for a given input X is obtained through a single forward pass using the learned weights W, expressed as $\hat{y} = f(X;W)$. However, in an MCDO-NN, the point forecast is derived from the average output of N stochastic forward passes:

$$\hat{y} = \frac{1}{N}\sum_{i=1}^{N} f(X;W_i), \qquad (10)$$

This formulation, W_i denotes the set of weights used during the i-th forward pass, where dropout has been applied, effectively simulating a sample from the approximate posterior distribution of weights. This procedure approximates the expected value of the predictive distribution in a Bayesian framework, $E(Y \mid X)$, as [6], [32]:

$$E(Y \mid X) = \frac{1}{N}\sum_{i=1}^{N} f(X;W_i). \qquad (11)$$

The variability of outputs across these forward passes provides an empirical estimate of the standard error (S.E.):

$$S.E. = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(f(X;W_i) - \hat{y}\right)^2}, \qquad (12)$$

where $f(X;W_i)$ is the prediction from the i-th forward pass with dropout applied, $\hat{y}$ is the mean prediction from all forward passes. Using this standard error, the forecast interval at a given confidence level α can be calculated as:

$$\hat{y} \pm z_{\alpha/2} S.E., \qquad (13)$$

where $z_{\alpha/2}$ is the critical value from the standard normal distribution corresponding to the desired confidence level. The standard error represents aleatoric uncertainty as well as the epistemic. By capturing the variability of the outputs, the forecast interval can be constructed to reflect the confidence in the neural network's predictions, [23], [33]. It is observed that the forecast interval in (13) can be considered as Wald's confidence interval, [34], [35].

5 Results

The results will be separated into two subsections: the optimal model, the impact of the hyperparameters, and the forecast intervals from the proposed method and the SARIMA.

5.1 Optimal Model and Impact of Hyperparameters

Our detailed analysis of neural network hyperparameters for forecasting durian exports reveals a relationship between model complexity and RMSEs. Figure 6 presents the top 40 models with the lowest RMSE. The optimal model, featuring a deep layer configuration of 300-240-180-120-60-30, achieved an RMSE of 9,570.24. However, the RMSE values varied across configurations, suggesting potential overfitting in deeper networks. For instance, models with 120-60-30 layers averaged an RMSE of 11,343.47, as shown in Figure 7. Concerning the activation functions of our optimal model, the sigmoid function was the least effective, achieving an average RMSE of 11,907.44, compared to 11,822.70 for ReLU and 11,059.11 for Tanh, the latter being the most effective, as illustrated in

Figure 8. The length of training also had a big effect on accuracy.

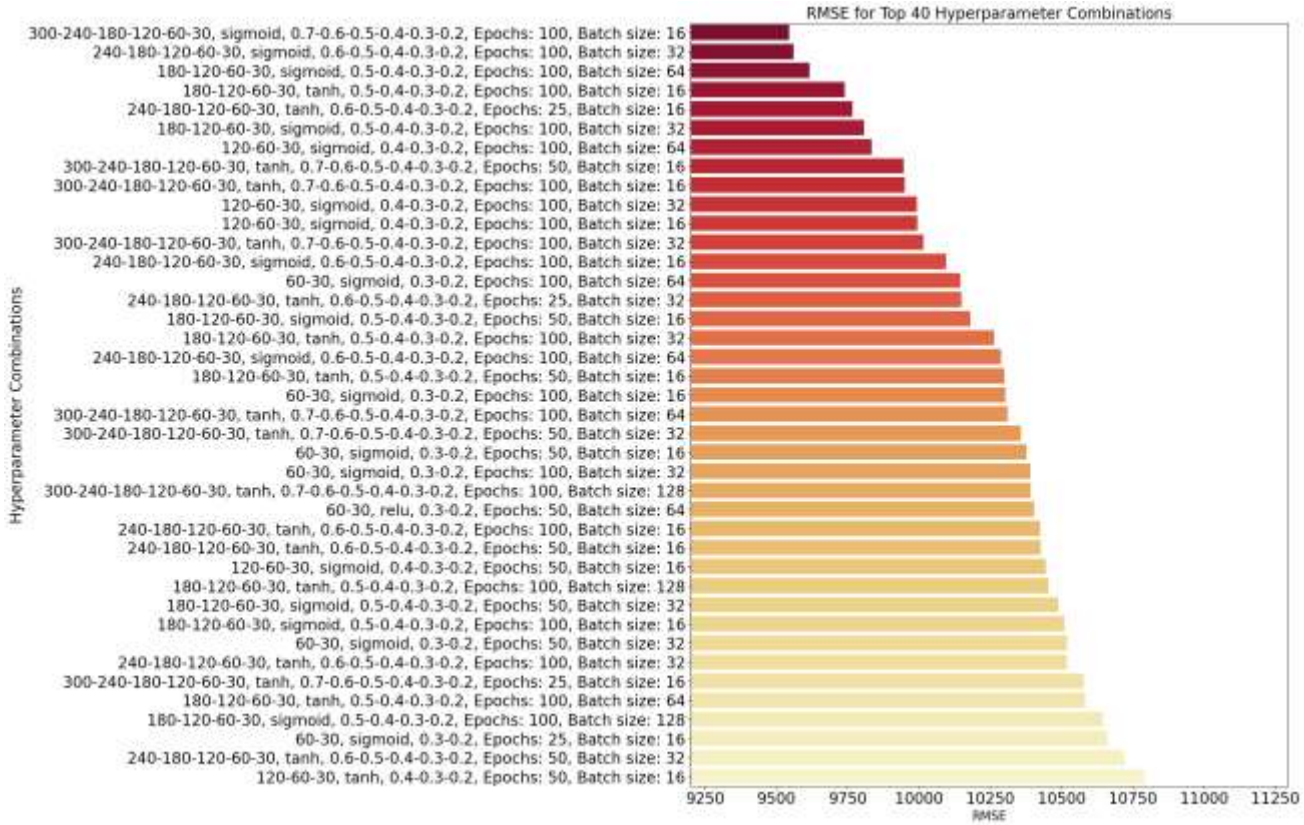


Fig. 6: Top 40 neural network models with the lowest RMSE values

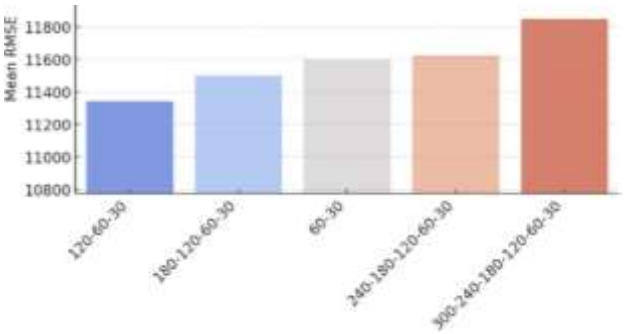


Fig. 7: Average RMSE for various layer configurations

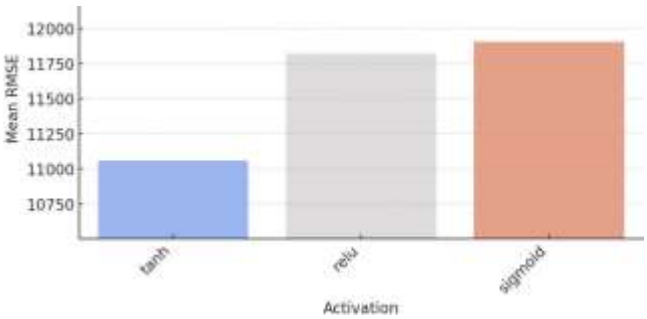


Fig. 8: Average RMSE for different activation functions

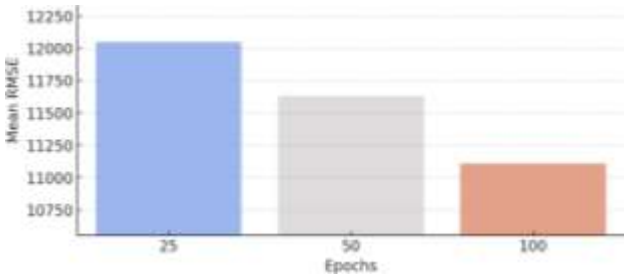


Fig. 9: Average RMSE for different numbers of Epochs

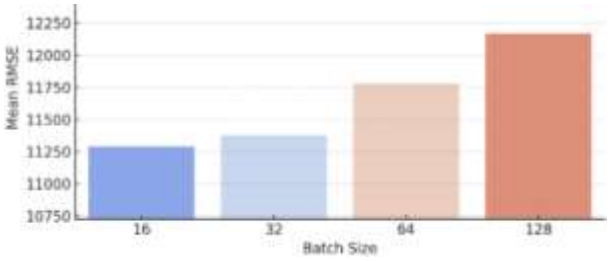


Fig. 10: Average RMSE for different numbers of batch sizes

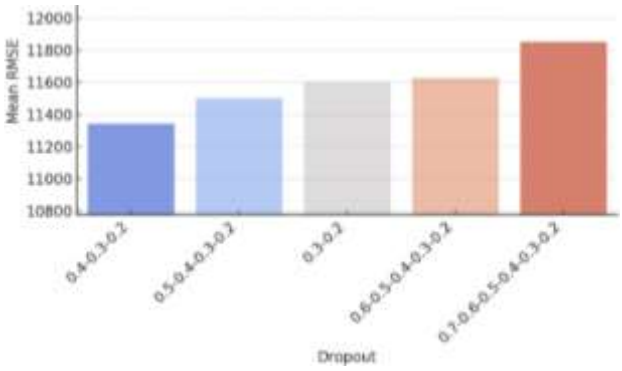


Fig. 11: Average RMSE for different dropout patterns

For example, Figure 9 shows that neural network models trained for 100 epochs had the lowest average RMSE of 11,110.30, which shows how important it is to have enough training time. Smaller batch sizes improved performance, with the optimal model using a batch size of 16 reaching an RMSE of 11,328.21, while larger sizes resulted in higher RMSEs, as shown in Figure 10. Figure 11 shows that models with dropout rates between 0.4 and 0.3 to 0.2 did better than models with higher dropout rates, with an average RMSE of 11,343.47. This shows how important it is to have balanced regularization.

5.2 Interval Forecasting of Durian Export

The study investigated the efficacy of a MCDO-NN model for interval forecasting durian exports and compared its performance with a benchmark SARIMA(1, 1, 2)(0, 1, 2)$_{12}$ model which gives the lowest AIC from the grid search. The SARIMA model, was chosen for its popularity and widespread use in time series forecasting. However, when subjected to out-of-sample testing, the SARIMA model presented an RMSE of 11,581.04, an MSE of 134,120,379.12, and an R-squared value of 0.2439. In contrast, the MCDO-NN model demonstrated superior predictive performance with an RMSE of 9,570.24, an MSE of 91,589,426.42, and an R-squared value of 0.4837. These metrics indicate that the MCDO-NN model not only provided better point estimates but also explained a greater variance in the data, thus offering a more accurate forecast.

Table 1 and Table 2 summarize the point, lower, and upper forecasts generated by the SARIMA and MCDO-NN models, respectively. Both models can generate negative forecasts for lower bounds, but the SARIMA model tends to produce higher negative values. Furthermore, the average width of the forecast intervals—an indicator of the model's confidence in its predictions—was narrower for the MCDO-NN model (13,492.32) than for the SARIMA model (16,006.12). This indicates that the MCDO model was not only more accurate but also more precise, offering tighter confidence intervals. Figure 12 and Figure 13 illustrate the interval forecasts.

Table 1. Comparison of point, lower, and upper forecasts from the SARIMA model with actual values in the test dataset

Month	Actual values	SARIMA Model		
		Lower Forecast	Point Forecast	Upper Forecast
Dec-22	8,016.03	-2,734.33	4,715.00	12,164.33
Jan-23	2,508.98	-5,174.65	2,861.73	10,898.11
Feb-23	2,955.26	-5,906.12	2,139.58	10,185.28
Mar-23	6,376.79	-6,038.98	2,009.08	10,057.15
Apr-23	48,583.21	12,862.15	20,912.25	28,962.36
May-23	12,397.42	30,432.27	38,484.38	46,536.50
Jun-23	29,361.91	14,605.78	22,659.91	30,714.04
Jul-23	15,200.59	8,804.16	16,860.31	24,916.46
Aug-23	8,604.99	-1,157.88	6,900.28	14,958.45
Sep-23	8,241.27	-6,118.60	1,941.58	10,001.75
Oct-23	2,323.94	-5,976.57	2,085.62	10,147.81

Table 2. Comparison of point, lower, and upper forecasts from the MCDO-NN model with actual values in the test dataset

Month	Actual values	MCDO-NN Model		
		Lower Forecast	Point Forecast	Upper Forecast
Dec-22	8,016.03	-1,844.70	2,443.93	6,732.56
Jan-23	2,508.98	-1,766.84	2,641.08	7,049.00
Feb-23	2,955.26	-1,790.59	2,914.34	7,619.28
Mar-23	6,376.79	-2,255.36	7,108.63	16,472.62
Apr-23	48,583.21	11,613.31	19,424.14	27,234.97
May-23	12,397.42	12,417.06	20,298.94	28,180.81
Jun-23	29,361.91	10,910.00	19,345.95	27,781.89
Jul-23	15,200.59	10,573.10	19,375.07	28,177.03
Aug-23	8,604.99	-2,573.09	7,379.21	17,331.52
Sep-23	8,241.27	-2,064.47	3,001.72	8,067.90
Oct-23	2,323.94	-2,141.27	2,930.45	8,002.17

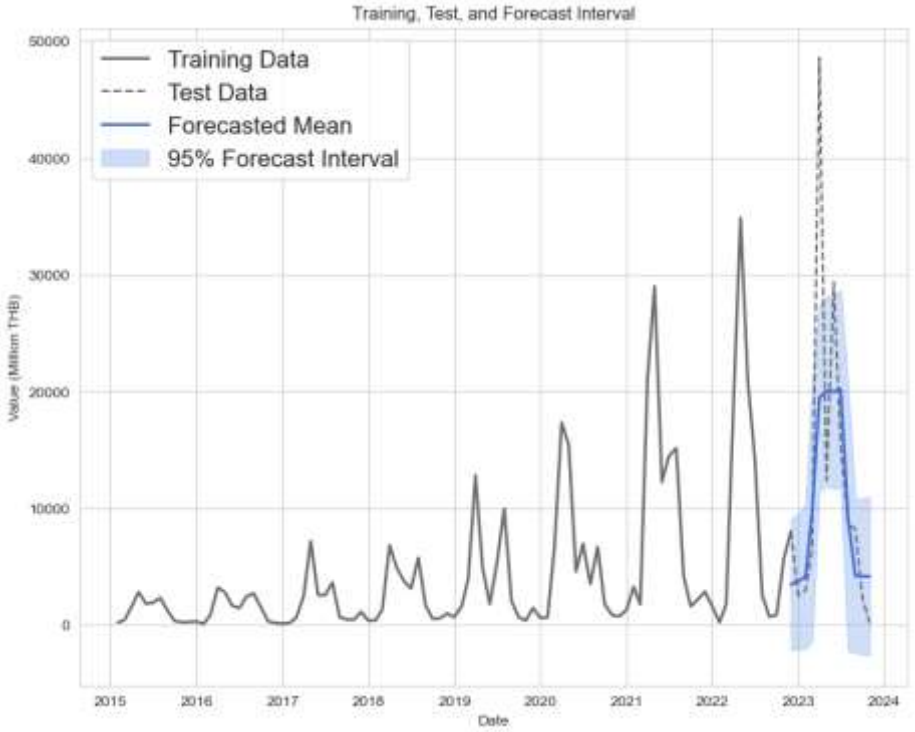


Fig. 12: Time series representation of durian export volumes, highlighting the division of data into training (solid line), test (dashed line), and the forecasted mean with a 95% forecast interval (shaded area)

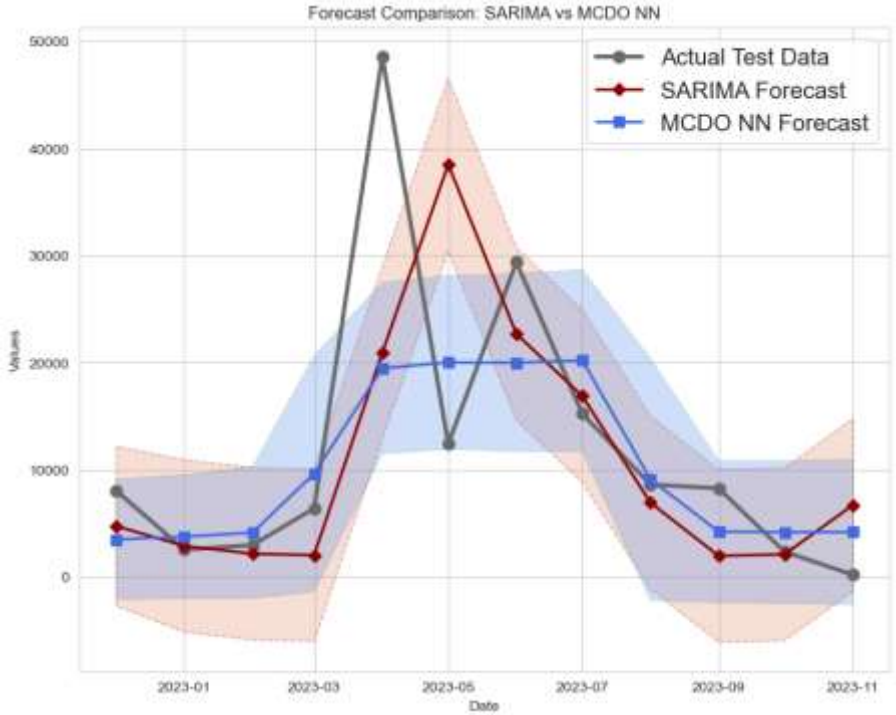


Fig. 13: Forecast visualization between the SARIMA and MCDO-NN models against actual test data from December 2022 to November 2023 (the shaded areas represent the prediction intervals)

6 Conclusions and Future Work

This study has shown the application of MCDO-NNs to create forecast intervals for durian exports in Thailand. The effects of various hyperparameters on the MCDO-NN models were explored. The result can be a guideline for data with seasonality, like the durian dataset. The point and interval forecasts were compared to those from the SARIMA model, a widely used statistical approach for time series data.

For future research, incorporating external factors, such as economic indicators, weather conditions, or market trends, could further improve the accuracy of the MCDD-NN. Additionally, the potential of combining MCDO-NN with other statistical methods can potentially improve both point and interval forecasts. This ensemble approach could leverage the strengths of various forecasting methodologies.

References:

[1] S. Thongkaew, C. Jatuporn, P. Sukprasert, P. Rueangrit, and S. Tongchure, "Factors affecting the durian production of farmers in the eastern region of Thailand," *Int. J. Agric. Ext.,* vol. 9, no. 2, pp. 285–293, 2021. DOI: 10.33687/ijae.009.02.3617.

[2] O. Rattana-amornpirom, "The Impacts of ACFTA on Export of Thai Agricultural Products to China," *J. ASEAN Plus+ Stud.*, vol. 1, no. 1, pp. 44-60, 2020.

[3] Kasikorn Research Center, "Durian: Record high export value of USD 934.9 million in May 2021", [Online]. https://www.kasikornresearch.com/en/analysis/k-econ/business/Pages/Durian-z3233.aspx (Accessed Date: November 12, 2023).

[4] S. Chaisayant, K. Chindavong, P. Wattananusarn, and A. Sittikarn, "Krungthai COMPASS," efinancethai, [Online]. https://www.efinancethai.com/LastestNews/LatestNewsMain.aspx?ref=A&id=WEZlTDM4eW5RaTQ9 (Accessed Date: December 9, 2023).

[5] P. Srisuradetchai, "A Novel Interval Forecast for K-Nearest Neighbor Time Series: A Case Study of Durian Export in Thailand," in *IEEE Access*, vol. 12, pp. 2032-2044, 2024, doi: 10.1109/ACCESS.2023.3348078.

[6] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," in *Proc. of The 33rd International Conference on Machine Learning*, PMLR, vol. 48, pp. 1050-1059, 2016.

[7] A. Kendall and Y. Gal, "What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?" in *Proc. of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*, Long Beach, California, USA, 2017, pp. 5580-5590. Curran Associates Inc., Red Hook, NY, USA, ISBN: 9781510860964.

[8] L. Ma and J. Kaewell, "Fast Monte Carlo Dropout and Error Correction for Radio Transmitter Classification," *2020 IEEE International Workshop on Information Forensics and Security (WIFS)*, New York, NY, USA, 2020, pp. 1-5, doi: 10.1109/WIFS49906.2020.9360887.

[9] P. M. Nazreen and A. G. Ramakrishnan, "DNN Based Speech Enhancement for Unseen Noises Using Monte Carlo Dropout," *2018 12th International Conference on Signal Processing and Communication Systems (ICSPCS)*, Cairns, QLD, Australia, 2018, pp. 1-6, doi: 10.1109/ICSPCS.2018.8631773.

[10] T. M. Hossain, M. Hermana, and S. J. Abdulkadir, "Epistemic Uncertainty and Model Transparency in Rock Facies Classification Using Monte Carlo Dropout Deep Learning," in *IEEE Access*, vol. 11, pp. 89349-89358, 2023, doi: 10.1109/ACCESS.2023.3307355.

[11] H. Zhai, Y. Hou, Q. Li, K. Chang, W. Zhang, and Q. Cheng, "Prediction of Remaining Useful Life Uncertainty Based on Monte Carlo Dropout and ResNet," *2022 China Automation Congress (CAC)*, Xiamen, China, 2022, pp. 2188-2193, doi: 10.1109/CAC57257.2022.10055388.

[12] M. Amin Maleki Sadr, Y. Zhu and P. Hu, "An Anomaly Detection Method for Satellites Using Monte Carlo Dropout," in *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 2, pp. 2044-2052, April 2023, doi: 10.1109/TAES.2022.3206257.

[13] J. Chen, H. Che, J. Sun, Y. Rao and J. Wu, "An automatic cephalometric landmark detection method based on heatmap regression and Monte Carlo dropout," 2023 *45th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, Sydney, Australia, 2023, pp. 1-4, doi: 10.1109/EMBC40787.2023.10341102.

[14] E. Hüllermeier and W. Waegeman, "Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods," in *Machine Learning*, vol. 110, no. 3, pp. 457-506, 2021, doi: 10.1007/s10994-021-05946-3.

[15] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles," in *Advances in Neural*

Information Processing Systems, vol. 30, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, [Online]. https://proceedings.neurips.cc/paper_files/paper/2017/file/9ef2ed4b7fd2c810847ffa5fa85bce38-Paper.pdf (Accessed Date: December 1, 2023).

[16] Y. Gal, "*Uncertainty in Deep Learning*," University of Cambridge, PhD thesis, 2016.

[17] N. Shahroudi, *"Probabilistic Forecasting with Monte-Carlo Dropout in Neural Networks,"* Master's thesis, Dept. Comput. Sci., Univ. Tartu, Tartu, Estonia, 2019.

[18] D. A. Nix and A. S. Weigend, "Estimating the mean and variance of the target probability distribution," *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, Orlando, FL, USA, 1994, pp. 55-60 vol.1, doi: 10.1109/ICNN.1994.374138.

[19] O. F. Tuna, F. O. Catak, and M. T. Eskil, "Exploiting epistemic uncertainty of the deep learning models to generate adversarial samples," in *Multimedia Tools and Applications*, vol. 81, no. 8, pp. 11479-11500, 2022, doi: 10.1007/s11042-022-12132-7.

[20] C. M. Bishop, "*Pattern Recognition and Machine Learning*," Springer, 2006.

[21] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling, "Semi-Supervised Learning with Deep Generative Models," in *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, Montreal, 2014, [Online]. http://arxiv.org/abs/1406.5298 (Accessed Date: November 9, 2023).

[22] M. Teye, H. Azizpour, and K. Smith, "Bayesian Uncertainty Estimation for Batch Normalized Deep Networks," in *Proc. of the 35th International Conference on Machine Learning (ICML)*, vol. 80, pp. 4907-4916, 2018.

[23] A. Graves, "Practical variational inference for neural networks," in *Advances in Neural Information Processing Systems 24 (NIPS)*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2011, pp. 2348-2356.

[24] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[25] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[26] Office of Agricultural Economics. "Agricultural Statistics of Thailand", [Online]. https://www.oae.go.th/view/1/Home/EN-US (Accessed Date: November 15, 2023).

[27] G. E. Hinton, "Learning multiple layers of representation," *Trends in Cognitive Sciences*, vol. 11, no. 10, pp. 428-434, 2007, ISSN: 1364-6613, doi: 10.1016/j.tics.2007.09.004.

[28] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in Proceedings of the *14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011, vol. 15, pp. 315-323.

[29] F. Chollet, *Deep Learning with Python*, 2nd ed. Manning, 2021, pp. 361–365.

[30] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: A System for Large-Scale Machine Learning," *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation (OSDI '16)*, Savannah, GA, USA, 2016, pp. 265-283.

[31] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, June 2014.

[32] D. P. Kingma, T. Salimans, and M. Welling, "Variational dropout and the local reparameterization trick," in *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 2015, vol. 28, pp. 2575-2583.

[33] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," in *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, Barcelona, Spain, Dec. 5-10, 2016, pp. 1019-1027.

[34] U. Kummaraka and P. Srisuradetchai, "Interval Estimation of the Dependence Parameter in Bivariate Clayton Copulas," *Emerg. Sci. J.*, vol. 7, no. 5, pp. 1478-1490, 2023. doi: 10.28991/ESJ-2023-07-05-02.

[35] P. Srisuradetchai, A. Niyomdecha, and W. Phaphan, "Wald Intervals via Profile Likelihood for the Mean of the Inverse Gaussian Distribution," *Symmetry*, vol. 16, no. 1, p. 93, 2024. doi: 10.3390/sym16010093.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

Both authors made equal contributions to the current study.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The authors have no conflicts of interest to declare.