Transactions of NAMP 22, (2025) 21-30

Transactions of
The Nigerian Association of
Mathematical Physics
Journal homepage: https://nampjournals.org.ng

ENHANCING ECONOMIC FORECASTING WITH BAYESIAN NEURAL NETWORKS: A FOCUS ON GDP PREDICTION AND UNCERTAINTY QUANTIFICATION IN NIGERIA

[1]Chidiebere, N. P., [1]Olaniran, S. F., [2]Adeniran, U.A.

[1]Department of Mathematics and Statistics, Kwara State University, Malete, Nigeria.

[2]Department of Mathematics, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia.

ARTICLE INFO

Article history:
Received 14/5/2025
Revised17/5/2025
Accepted 28/6/2025
Available online 17/7/2025

Keywords:
Bayesian Neural Networks, Economic Forecasting, Deep Learning, Uncertainty Quantification

ABSTRACT

This study produces a Bayesian Neural Network (BNN) to forecast GDP growth while quantifying prediction uncertainty which is often overlooked in traditional neural networks. By integrating Bayesian inference with deep learning, the model generates prediction intervals rather than single-point estimates, offering a probabilistic perspective crucial for decision-making under uncertainty. The BNN was initially trained on simulated data to validate its architecture and subsequently tested on real-world quarterly GDP data (2010-2023), monthly inflation rates, and interbank interest rates sourced from the Central Bank of Nigeria. The model employs Guassian-distributed weights and biases, uses Rectified Linear Unit (ReLU) activation functions, and optimizes training through the Adam algorithm. Results demonstrates that the BNN achieves strong predictive performance, with its prediction interval providing actionable insights for scenarios where uncertainty quantification is paramount. This approach improves GDP forecasting accuracy and provides a robust framework for analyzing volatile economic metrics in emerging economies like Nigeria

1. INTRODUCTION

Accurate predictions of economic metrics including GDP growth rate, inflation rate, unemployment rate and interest rate, are a significant aspect of economic policy decisions. For businesses, financial institutions, and Governments, accurately predicting GDP growth is essential because it informs risk management, resource allocation, and decision-making,[1]. Traditional economic forecasting relies on statistical models that assume linear and stationary relationships between input variables and economic outcomes. Traditional economic forecasting methods, such as autoregressive integrated moving average (ARIMA) developed by [2] and vector autoregression (VAR) developed by [3], cannot accurately capture the complexities of economic data, including non-linearity, time-varying dynamics, and uncertainties.

*Corresponding author: CHIDIEBERE, N.P.
E-mail address: chidieberenwamaka0@gmail.com
https://doi.org/10.60787/tnamp.v22.547

Traditional techniques of economic forecasting frequently lack the flexibility to respond to new data and emerging trends. Big data and high-frequency financial and macroeconomic datasets have enabled advanced machine learning techniques, including deep learning, to improve GDP growth predictions [4], [5]. Traditional econometric models have limits in forecasting accuracy, as emphasized in comparative research [6], [7]. Deep learning models, with their capacity to understand complex patterns and extract significant information from vast, multidimensional datasets, have shown higher prediction accuracy in numerous financial and economic applications, [1], [8]. Deep neural networks' capacity to automatically learn hierarchical representations from data has result to remarkable breakthroughs in predictive modeling. Deep learning algorithms is able to recognize complicated patterns as well as connections in datasets that regular models might overlook in economic forecasting. Deep learning models, like Deep Neural Network (DNN), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), and others, have gained interest in economic applications, providing improved tools for predicting with vast data sets [9]. Deep learning models, on the other hand, frequently fail to produce valid estimates of uncertainties for tasks critical to the quantification of uncertainty in order to obtain an accurate evaluation of its influence on a particular economy, such as economic forecasting. Bayesian deep learning is a promising approach to improve accuracy and robustness while quantifying uncertainties. It combines the flexibility of deep learning with the probabilistic approach of Bayesian inference, resulting in better decision-making, [10], [11] By integrating the strengths of deep learning frameworks such as DNN, CNN, and RNN with Bayesian inference, researchers hope to deliver more accurate and probabilistic economic and financial predictions. Using Bayesian deep learning approaches in economic forecasting integrates deep learning and probabilistic graphical models (PGM) [12]. [13] book provides a theoretical framework for neural networks, including structures, learning techniques, and applications. It defines fundamental ideas like backpropagation and multilayer perceptron. [14] introduced the backpropagation algorithm, which significantly improved the training of multi-layer neural networks and enabled deep learning advancements. However, the algorithm suffers from slow convergence, vanishing gradient problems in deep networks, and high computational cost for large-scale datasets. [15] found that multi-layer feedforward neural networks with sufficient number of hidden units can estimate any continuous function, making them universal approximators. Deep learning has been increasingly popular for economic forecasting in recent years. In a thorough assessment of financial time series forecasting using deep learning applications from 2005 to 2019, [16] found that deep learning algorithms are increasingly effective in capturing financial patterns. [17] offered an overview of deep learning-based stock market prediction, highlighting significant models and their financial forecasting strengths. The intricate nature of economic data can lead to overfitting, particularly if the dataset is small or has a high number of dimensions. [18] proposed dropout as a regularization method to prevent overfitting in deep neural networks. [19] studied how regularization techniques improve model generalization and prevent overfitting in deep learning architectures. Bayesian techniques provide a powerful framework for economic analysis by quantifying uncertainty and updating predictions in response to new data. Bayesian methods have various advantages, including uncertainty quantification, integration of prior knowledge, and flexibility, as they can handle intricate data and hierarchical structures, which make them suitable for variety of applications. In [20], Bayesian strategies were proposed to address structural uncertainty in macroeconomic analysis using panel vector autoregressive (VAR) models. The study found that Bayesian techniques improve predicting ac curacy in panel VAR models. [21] proposed Bayesian Model Averaging (BMA) to improve forecast accuracy by accounting for model uncertainty. It was demonstrated in macroeconomic forecasting, where it outperformed single-model techniques. [22] proposed a Bayesian Regularized Neural Network (BRNN) model for predicting the Naira-USD exchange rate. It was demonstrated that BRNN outperforms typical machine learning models when dealing with exchange rate volatility and uncertainty.

MATERIALS AND METHODS

The Bayesian Neural Network (BNN) model was developed applying the Variational Inference (VI) method.

2.1 Data Source

The data used is a secondary data gotten from the Central Bank of Nigeria (CBN) database. The data used for the GDP growth forecast is the quarterly datasets of GDP at 2010 CONSTANT MARKET PRICES from quarter 1 of 2010 to quarter 4 of 2023, https://www.cbn.gov.ng/rates/RealGDP.html. For the inflation rate datasets, we used the monthly datasets of the ALL ITEMS (12 MONTHS AVG. CHANGE) from the month 1 of 2010 to month 12 of 2023, https://www.cbn.gov.ng/rates/inflrates.html. The interest rate datasets used is the monthly datasets of INTERBANK CALL RATE from month 1 of 2010 to month 12 of 2023, https://www.cbn.gov.ng/rates/interbankrates.html.

2.2 Data Processing

Upon data collection, the data was processed to verify quality and uniformity. The data was cleaned, which included handling missing value and screening for outliers. The data was also standardized to ensure that it is on comparable scales ap propriate for deep learning models.

2.3 Model Architecture

The input layer, hidden layers with the ReLU activation, and output layer make up the BNN model.

$$\hat{y} = f(x; W) = W_2 ReLU(W_1 . x + b_1) + b_2 \tag{1}$$

where;

x is the input to the network which is the vector of features.

The weight matrices are W_1 and W_2. W_1 connects the input layer to the hidden layer while W_2 connects the hidden layer to the output layer. The bias vectors are b_1 and b_2 which are added to the respective layers to shift the activation functions. b_1 is the bias for the hidden layer and b_2 is the bias for the output layer. hiddenlayerandb2isthebiasfortheoutputlayer. The non-linear activation function, ReLU, gives the network non-linearity so that it can represent complex patterns. Unlike traditional neural networks, BNNs view weight and bias as random variables with a prior distribution. The bias and weights are determined using the Guassian distribution, which assumes a normal distribution. The key to BNNs is Bayesian inference, which updates the weight and bias distributions based on observed data. The posterior distribution of the weights given the data, is computed as:

$$P\left(W^{(l)} \middle| D\right) = \frac{P\left(D \middle| W^{(l)}\right) P\left(W^{(l)}\right)}{P(D)} \propto P\left(D \middle| W^{(l)}\right) P\left(W^{(l)}\right) \tag{2}$$

Making predictions based on the posterior distribution of the model parameters is the main goal of the BNN model, rather than simply drawing inferences from it. This is achieved by integrating the posterior distribution to obtain the posterior predictive distribution:

$$P(y|x,D) = \int P(y|x,\theta)P(\theta|D)d\theta \tag{3}$$

Where y is the predicted output, x is the input, D is the observed data and θ=(W,b) is the model parameter. This distribution shows a comprehensive distribution of probable outcomes, including uncertainty.

2.4 Model Training and Evaluation

The posterior distribution in Eq. 2 is intractable, so we apply the Stochastic Variational Inference (SVI) approach. We use a more straightforward distribution to approximate the posterior distribution q(W) which belongs to Guassian distribution

$$q(W) = N(W|\mu,\Sigma) \tag{4}$$

And then minimizes the Kullback-Leibler (KL) divergence between q(W) and $P\left(W^{(l)}\middle|D\right)$.

This KL divergence minimization is the same as maximizing the Evidence Lower Bound (ELBO), that is, defining the loss function:

$$\mathcal{L}(\mu,\Sigma) = E_{q(w)}[\log p\,(D \mid w)] - \mathrm{KL}\big(q(w) \parallel p(w)\big) \tag{5}$$

We use the training data to update the parameters μ and Σ by minimizing $\mathcal{L}(\mu,\Sigma)$. The SVI used the variant of the Stochastic Gradient Descent (SGD) Adam optimizer to update the model parameters iteratively.

When considering the parameters, the loss function's gradient $\nabla_W L(W,b)$ and $\nabla_b L(W,b)$, represent the direction and the rate of change to decrease the loss function. The parameters need to be moved in the opposite direction of the gradient in order to minimize the loss. The Basic Gradient Descent (BGD) which updates the parameters by moving them in the opposite direction of the gradients is given as:

$$W \leftarrow W - \eta\nabla_W L(W,b) \tag{6}$$

$$b \leftarrow b - \eta\nabla_b L(W,b) \tag{7}$$

η is the learning rate, a hyperparameter that controls the size of the step we take in the parameter space during each update. The Adam optimizer improves upon BGD by adapting the learning rate for each parameter individually. The algorithm is given below:

1. Initialize the parameter: We initialize the weights W_0 using the prior knowledge. We also initialize the first moment vector $m_0 = 0$and the second moment vector$v_0 = 0$. We set the timestep $t = 0$ and the learning rate $\eta = 0.01$.

2. First Moment (Mean of the Gradients): Adam optimizer computes an exponentially moving average of the gradient mt, which helps in smoothing the update process:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla_W L(W,b) \tag{8}$$

β_1 is a hyperparameter that controls the decay rate of the moving average.

3. Second Moment (Variance of the Gradients): Adam optimizer also computes an exponentially moving average of the squared gradient v_t, which helps in scaling the learning rate:

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2)\left(\nabla_W L(W, b)\right)^2 \quad (9)$$

β_2 is another hyperparameter controlling the decay rate of this moving average.

4. Bias Correction: Since m_t and v_t are initialized to 0, Adam optimizer includes bias correction terms to counteract the initial bias:

$$\widehat{m_t} = \frac{m_t}{1 - \beta_1^t} \quad (10)$$

$$\widehat{v_t} = \frac{v_t}{1 - \beta_2^t} \quad (11)$$

5. Parameter Update: The parameters are updated as follows:

$$W_t \leftarrow W_{t-1} - \eta \frac{\widehat{m_t}}{\sqrt{\widehat{v_t}} + \epsilon} \quad (12)$$

$$b_t \leftarrow b_{t-1} - \eta \frac{\widehat{m_t}}{\sqrt{\widehat{v_t}} + \epsilon} \quad (13)$$

ϵ is a small constant added to prevent division by zero.

This process of iteration continues for multiple epochs or until convergence. We use the model to produce predictions on the test data after it has been trained. To make predictions, we sample weights from the posterior predictive distribution in Eq. 3 and average them.

$$\hat{y} = \frac{1}{N}\sum_{i=1}^{N} f\left(x; W^i\right) \quad (14)$$

where $W^i \sim q(W)$.

We evaluate the model's performance by assessing its predicted uncertainty. The predictive uncertainty is calculated by examining the variations in the predictions:

$$Var(\hat{y}) = \frac{1}{N}\sum_{i=1}^{N}\left(f\left(x; W^i\right) - \hat{y}\right)^2 \quad (15)$$

2.5 Hyperparameter Optimization

For the BNN model to perform better, different combinations of vital hyperparameters such as hidden dimensions and learning rate were evaluated. Table 1 shows the many combinations utilized during the tuning process. Each combination was assessed on the following metrics: Mean Square Error (MSE), Mean Absolute Error (MAE), and coverage.

Table 1: Performance Table with Hidden Dimensions and Learning Rates

Hidden Dimension	Evaluation Metrics	Learning Rate			
		0.001	**0.005**	**0.01**	**0.05**
	MSE	1.27	0.74	0.65	0.62

64	MAE	0.95	0.62	0.57	0.56
	COVERAGE	32.26%	100.00%	100.00%	100.00%
32	MSE	1.56	0.98	1.26	4.97
	MAE	1.07	0.78	0.93	2.01
	COVERAGE	19.35%	100.00%	100.00%	100.00%
16	MSE	1.64	0.99	1.11	1.35
	MAE	1.08	0.78	0.84	0.97
	COVERAGE	25.81%	90.32%	83.87%	96.77%
8	MSE	1.18	1.12	1.29	1.25
	MAE	0.81	0.85	0.95	0.93
	COVERAGE	29.03%	67.74%	61.29%	64.52%
4	MSE	1.39	1.14	1.21	1.12
	MAE	0.92	0.87	0.90	0.85
	COVERAGE	29.03%	54.84%	54.84%	54.84%

Table2 shows that the optimal hidden dimension and learning rate for BNN performance is (16,0.005), resulting in low MSE and MAE and high coverage.

The hyperparameters used for the BNN model is given in Table 2;

Table 2. Hyperparameters Table.

Hyperparameters	Value
Input Dim	10
Hidden Dim	16
Output Dim	1
Learning Rate	0.005
Batch Size	32
Epochs	300

An input dimension of 10 indicates that the model expects 10 features for each data point. The quantity of neurons in the hidden layers is the hidden dimension; our hidden layer contains 16 neurons. Because we have 2 hidden layers, each contains 8 neurons. The output dimension of 1 indicates that we expect only one output, which is GDP. The Adam optimization algorithm uses the learning rate to determine the step size. A learning rate of 0.005 indicates that the model will update the weights at a 0.005 step to help it converge on the optimal solution. The batch size of 32 indicates that the model processed 32 data points before performing the backpropagation step. Epochs of 300 means the model will make 300 passes over the data to optimize its performance.

2.6 Evaluation Metrics

The BNN model was evaluated using the Mean Square Error (MSE), Mean Absolute Error (MAE), and coverage, which is the percentage of actual values that fall within the predicted confidence interval.

RESULTS

The evaluation table is given in Table 3;

Table 3: Evaluation Metrics

Metrics	Value
MSE	0.99
MAE	0.78
Coverage	90.32

MSE calculates the average squared difference between actual and predicted values. In this case, an MSE of 0.99 implies that the squared differences between predicted and actual GDP values are relatively small. The MAE calculates the average absolute difference between actual and predicted values. A value of 0.78 indicates that the average absolute difference between predicted and actual GDP values is around 0.78 units. This shows that the model is moderately accurate, with predictions deviating from true values by approximately 0.78 units on average. Coverage is the percentage of actual values falling within the prediction interval (specified by the lower and upper bounds of the 95% confidence interval). With a coverage of 90.32%, more than 90% of the actual values are within the predicted confidence interval, which is less than the expected 95%. This coverage indicates a minor under confidence, which means the model's intervals are a little too tight.

The evaluation plot is given in Fig. 1

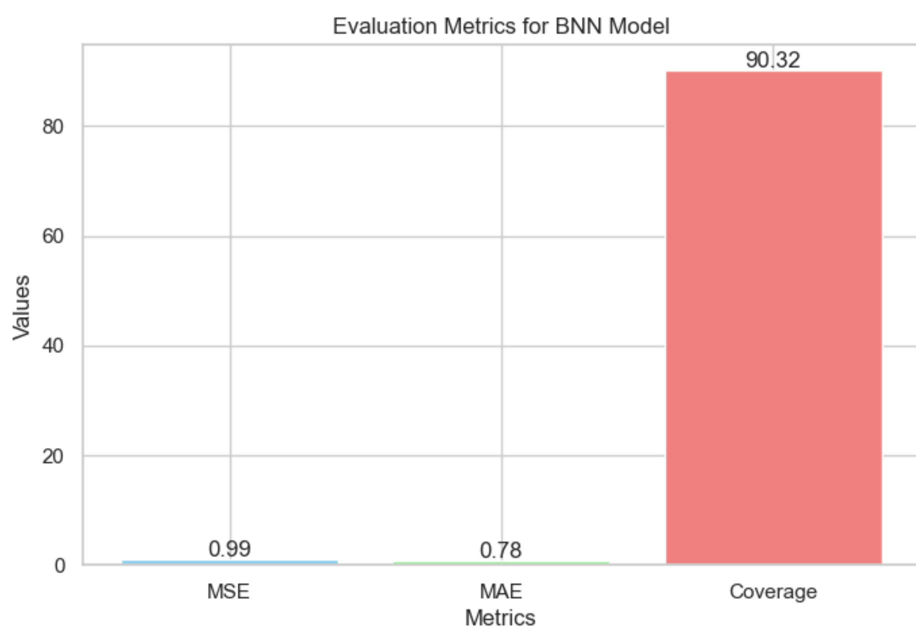

Figure 1. Plot of the Evaluation Metrics.

The low MAE and MSE indicates that the model have a high accuracy and the high coverage suggest that the approach is effectively capturing uncertainties.

The forecast table is given in Table 4;

Table 4: Forecast Table with Lower and Upper Bounds

Actual Value	Predicted Value	Lower Bound	Upper Bound
-0.107206	-0.202959	-1.050463	1.404272

-0.107206	0.224955	-0.353073	1.156345
-0.131346	-0.235192	-1.335047	0.864664
-0.131346	0.244618	-1.387753	0.930206
0.722205	0.225144	-1.387753	0.632976
0.722205	-0.229072	-1.449252	0.689124
0.722205	0.159671	-1.050278	1.203022
0.722205	-0.193950	-1.560273	1.229458
1.523102	0.240734	-1.053524	1.237345
1.523102	0.160408	-1.264354	1.072118
1.523102	-0.105702	-1.376738	1.013945
0.162268	0.149976	-2.120406	1.520661
0.162268	0.148667	-1.762581	1.464365
0.162268	0.146317	-1.758120	1.463385
0.121148	-0.211961	-1. 571137	1.147203

The forecast plot is given in Fig.2;

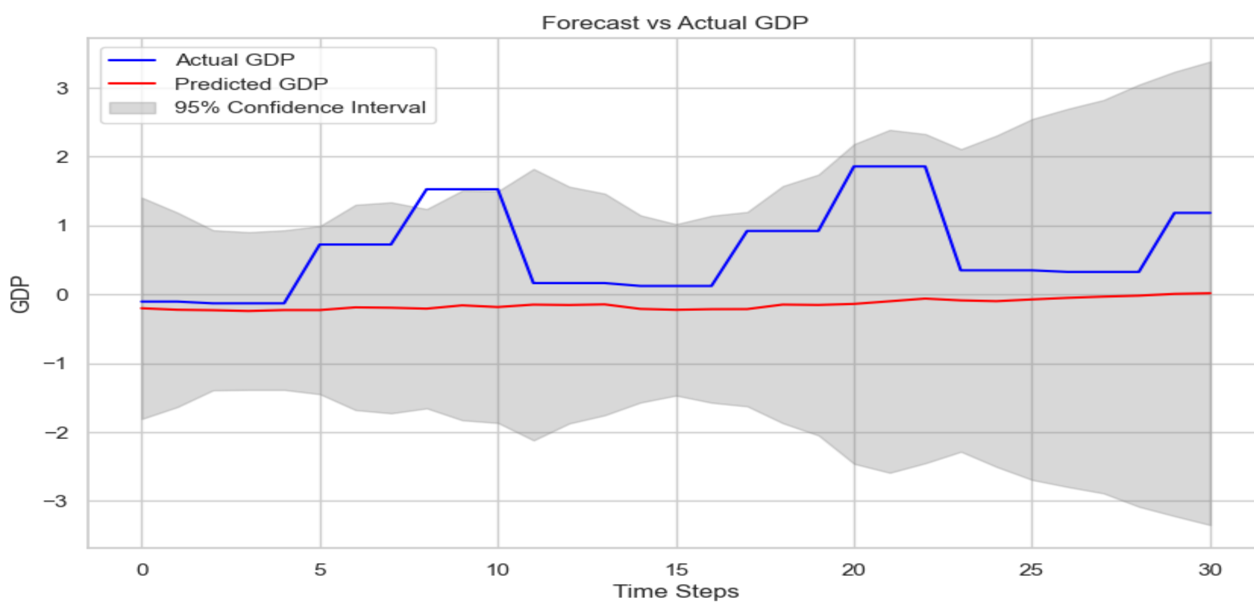

Figure 2. Forecast plot with coverage.

The actual GDP fluctuates which shows high variability in the data while the predicted GDP does not capture the fluctuation of the actual GDP. The 95% confidence interval is wide which indicate high uncertainties in the prediction.

CONCLUSION

In this work, BNN is demonstrated to be a highly successful technique for economic forecasting. The BNN model presented in this research presents a strategy for forecasting economic variables that gives both accuracy and a measure of uncertainty. This was accomplished by integrating Bayesian inference with neural network flexibility. The BNN model built also demonstrates that BNN may be applied to small datasets as well as large datasets. The BNN model's uncertainty quantification is a highly successful technique for improving the dependability of economic projections and providing decision-makers with critical insights into risk management and capitalizing on opportunities in ever-changing economic scenarios.

REFERENCES

[1] J. Heaton, N. G. Polson, and J. H. Witte, "Deep Learning for Finance: Deep Portfolios," *Appl. Stoch. Models Bus. Ind.*, vol. 32, no. 3, pp. 220–234, 2016, doi: 10.1002/asmb.2200.

[2] G. Box, "Box and Jenkins: Time Series Analysis, Forecasting and Control," in *A Very British Affair: Six Britons and the Development of Time Series Analysis During the 20th Century*, London: Palgrave Macmillan UK, 2013, pp. 161–215. doi: 10.1057/9781137291264_6.

[3] C. A. Sims, "Macroeconomics and Reality," *Econometrica*, vol. 48, no. 1, pp. 1–48, 1980.

[4] Y. Huang and E. S. Yan, "Economic Recession Forecasts Using Machine Learning Models Based on the Evidence from the COVID-19 Pandemic," *Mod. Econ.*, vol. 14, no. 7, pp. 899–922, 2023.

[5] J. Prosky, X. Song, A. Tan, and M. Zhao, "Sentiment predictability for stocks," *ArXiv Prepr. ArXiv171205785*, 2017.

[6] J. H. Stock and M. W. Watson, "Vector autoregressions," *J. Econ. Perspect.*, vol. 15, no. 4, pp. 101–115, 2001.

[7] M. Banbura, D. Giannone, and L. Reichlin, "Large Bayesian vector auto regressions," *J. Appl. Econom.*, vol. 25, no. 1, pp. 71–92, 2010.

[8] J. Barunik and L. Hanus, "Learning probability distributions in macroeconomics and finance," *ArXiv Prepr. ArXiv220406848*, 2022.

[9] Y. Zheng, Z. Xu, and A. Xiao, "Deep learning in economics: a systematic and critical review," *Artif. Intell. Rev.*, vol. 56, no. 9, pp. 9497–9539, 2023.

[10] H. Wang and D.-Y. Yeung, "A survey on bayesian deep learning," *ACM Comput. Surv. CSUR*, vol. 53, no. 5, pp. 1–37, 2020.

[11] J. Geweke and C. Whiteman, "Bayesian forecasting," Elsevier, Amsterdam, The Netherlands, Tech. Rep., 2006.

[12] A. Brusaferri, M. Matteucci, P. Portolani, and A. Vitali, "Bayesian deep learning based method for probabilistic forecast of day-ahead electricity prices," *Appl. Energy*, vol. 250, pp. 1158–1175, 2019.

[13] S. Haykin, *Neural networks: a comprehensive foundation*. Prentice hall PTR, 1994.

[14] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by backpropagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[15] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, no. 5, pp. 359–366, 1989.

[16] O. B. Sezer, M. U. Gudelek, and A. M. Ozbayoglu, “Financial time series forecasting with deep learning: A systematic literature review: 2005–2019,” *Appl. Soft Comput.*, vol. 90, p. 106181, 2020.

[17] P. Raj, A. Mehta, and B. Singh, “Stock Market Prediction Using Deep Learning Algorithm: An Overview,” in *International Conference on Innovative Computing and Communications: Proceedings of ICICC 2022, Volume 2*, Springer Nature, 2022, p. 327.

[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, 2014.

[19] M. D. Liman, S. I. Osanga, E. S. Alu, and others, “Regularization effects in deep learning architecture,” *J. Niger. Soc. Phys. Sci.*, pp. 1911–1911, 2024.

[20] G. Koop and D. Korobilis, “Model uncertainty in panel vector autoregressive models,” *Eur. Econ. Rev.*, vol. 81, pp. 115–131, 2016.

[21] M. F. Steel, “Bayesian model averaging and forecasting,” *Bull. EU US Inflat. Macroecon. Anal.*, vol. 200, pp. 30–41, 2011.

[22] O. R. Olaniran, S. F. Olaniran, and J. Popoola, “Bayesian regularized neural network for forecasting naira-USD exchange rate,” in *International Conference on Soft Computing and Data Mining*, Springer, 2022, pp. 213–222.