Nigerian Journal of Technology (NIJOTECH)
Vol. 44, No. 4, December, 2025, pp. 647 – 659
www.nijotech.com

Print ISSN: 0331-8443
Electronic ISSN: 2467-8821
https://dx.doi.org/10.4314/njt.v44i4.11

OPTIMAL DEEP NEURAL NETWORK PARAMETERS FOR POWER LOSS MINIMIZATION ANALYTICS

AUTHORS:
U. G. Inyang[1,5], I. J. Eyoh[2,5], U. A. Umoh[3,5], E. A. Ubong[1,5*], E. E. Ene[1,5], D. C. Obiyo[4,5], and B. E. Akponome[3,5]

AFFILIATIONS:
[1]Department of Data Science, Faculty of Computing, University of Uyo, Nigeria
[2]Department of Computer Science, Faculty of Computing, University of Uyo, Nigeria
[3]Department of Information Systems, Faculty of Computing, University of Uyo, Nigeria
[4]Department of Computer Science, College of Physical Sciences, Michael Okpara University of Agriculture, Umudike, Nigeria
[5]TETFund Center of Excellence in Computational Intelligence Research, University of Uyo, Nigeria

***CORRESPONDING AUTHOR:**
Email: emmanuelubong30@gmail.com

ARTICLE HISTORY:
Received: March 29, 2025.
Revised: October 15, 2025.
Accepted: October 23, 2025.
Published: January 03, 2026.

KEYWORDS:
DNN, Optimal Activation Function, Smart Grid, Power Loss, Model Performance Metrics, Parameter tuning

ARTICLE INCLUDES:
Peer review

DATA AVAILABILITY:
On request from author(s)

EDITORS:
Chidozie Charles Nnaji

FUNDING:
None

Abstract

Power loss remains a critical problem in power systems, leading to voltage instability, reduced transmission efficiency, equipment degradation, and significant financial losses for utility providers. These inefficiencies disrupt reliable electricity delivery, increase operational costs, and hinder the sustainability goals of smart grid infrastructure. Traditional analytical and optimization methods have often proven inadequate in accurately modeling the nonlinear and dynamic behaviors that characterize power networks. As a result, there is a growing need for advanced intelligent models capable of analyzing large-scale data, uncovering patterns, and predicting losses more effectively. This study addresses the power loss challenge by developing an optimized Deep Neural Network (DNN) framework for power loss minimization analytics in smart grids. Three separate power loss datasets were analyzed using the Orange data mining platform and the Python development environment. The DNN model was systematically tuned by optimizing key parameters such as activation functions, number of hidden layers, learning rates, and batch sizes. Among the configurations tested, the model employing the Rectified Linear Unit (ReLU) activation function with six hidden layers achieved the best performance. The optimized DNN produced a Mean Squared Error (MSE) of 1.0E-03, Root Mean Squared Error (RMSE) of 3.4E-02, Mean Absolute Error (MAE) of 1.9E-02, coefficient of determination (R^2) of 0.94, and Mean Absolute Percentage Error (MAPE) of 4.8%. When compared with conventional models, the optimized DNN demonstrated a 20–25% improvement in predictive accuracy. These results confirm that optimizing DNN parameters significantly enhances power loss analytics in smart grid systems. The proposed model offers a robust and intelligent solution for minimizing losses, improving energy efficiency, and supporting informed decision-making in modern power networks.

1.0 INTRODUCTION

The Smart Grid (SG) is an advanced and integrated power grid system that utilizes advanced technologies and communication infrastructures to enable real-time, bi-directional communication and control between power generation, transmission, distribution, and consumption systems [1]. Its primary objective is to enhance the power system efficiency, reliability, flexibility, and sustainability while ensuring the quality and security of electricity supply to consumers. The components of a smart grid

HOW TO CITE:
Inyang, U. G., Eyoh, I. J., Umoh, U. A., Ubong, E. A., Ene, E. E., Obiyo, D. C., and Akponome, B. E. "Optimal Deep Neural Network Parameters for Power Loss Minimization Analytics", *Nigerian Journal of Technology*, 2025. 44(4), pp. 647 – 659. https://dx.doi.org/10.4314/njt.v44i4.11

include sensors, communication networks, energy storage systems, and control systems. Smart grid technology finds applications in diverse areas, including electric vehicle charging, demand response, and microgrid systems. Despite its varied applications, it still faces a significant challenge: power loss in power transmissions. Electric power losses refer to the wasted energy that dissipates within the power system, primarily occurring along the transmission path from generation sources to load centers, resulting in reduced electricity availability for consumption [2]. Through this understanding, it becomes evident that reducing power loss is crucial. By focusing on minimizing this loss, the overall efficiency of the power grid can be significantly enhanced. Moreover, reducing power loss is beneficial for the grid's efficiency and plays a vital role in decreasing the environmental impact of power generation, leading to a more sustainable and eco-friendly energy system. Power losses in electrical networks are broadly classified into technical and non-technical losses.

Technical losses occur naturally due to energy dissipation in electrical components such as transmission lines, transformers, and conductors, and they can be reduced through engineering solutions like upgrading conductors, using shunt capacitors, and employing Flexible AC Transmission System (FACTS) devices.

Non-technical losses, on the other hand, result from human or operational factors such as electricity theft, meter tampering, inaccurate billing, or poor data management. Together, these two categories are often expressed as Aggregate Technical, Commercial, and Collection (ATC&C) losses, which utilities use to measure overall system efficiency. Minimizing both types of losses is essential for improving grid reliability, reducing financial losses, and enhancing sustainability in power delivery[3]. However, these traditional approaches have limitations as they often address power loss in isolated components rather than holistically [4]. To overcome these limitations, data analysis and machine learning techniques are essential. These techniques also give deeper insights into power loss patterns and causes, enabling more precise and effective interventions. By leveraging data analysis and machine learning, it is possible to optimize the entire power grid, further enhancing efficiency and sustainability. In the case of using smart facilities, techniques in data-driven models rely on analysis of data and machine learning (ML) to offer flexible methods for predicting building energy consumption [5]. There has been an intriguing interplay between deep learning techniques in power systems. Through this, DNNs benefit from parallel computation, which enhances their efficiency in processing vast amounts of data simultaneously [6].

By incorporating these advanced technologies, the limitations of traditional methods can be addressed, leading to a more efficient and sustainable power grid. In gradient descent-based training of DNN, only weights and biases are learned. Hence, parameter tuning is crucial in DNN as it significantly impacts model performance and efficiency. In DNNs, selecting appropriate hyperparameters is essential for optimizing performance, enhancing generalization, improving computational efficiency, and ensuring reproducibility. Well-tuned hyperparameters enable the model to deliver accurate and dependable predictions for the specific task [7]. This process enhances computational efficiency, which is vital when using large datasets with limited resources.

Additionally, parameter tuning makes models more robust to data variations, leading to reliable performance in real-world scenarios. Though a crucial role in determining model quality can be enhanced using hyperparameters. To optimize the Keras model’s hyperparameters, a combination of techniques was applied. Specifically, Randomized Search CV from scikit-learn was used to explore a defined search space, including hyperparameters such as learning rate, dropout rate, number of epochs, batch size, and activation function [8].This "activation" parameter specifies the activation function for network nodes (neurons). In TensorFlow, this is often set for each layer. The activation function determines how the input values of each node are transformed into output values, which can greatly affect model performance. Importantly, Activation Functions (AFs) play a critical role in NNs by introducing non-linearity, enabling the networks to learn complex patterns[9]. These optimum parameters vary after single unruly to another, even for results in the same domain [10]. There is a need for optimal DNN parameter tuning as it allows for a deeper exploration of model capabilities, optimizing specific performance metrics like Mean Square Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), coefficient of determination (R^2), Mean Absolute Percentage Error (MAPE), Accuracy, precision, and recall according to the task requirements; This work aims to provide optimal DNN parameters that can be

https://dx.doi.org/10.4314/njt.v44i4.11

easily applied to various dataset structures, thus contributing to the field of knowledge. This section is organized into five sub-sections. Sub-section A summarizes related works, while Sub-section B outlines the data-driven analytic framework. The experimental setup is detailed in Sub-section C. Finally, Sub-section D discusses Optimal Activation Function Selection.

2.0 RELATED WORKS

2.1 SG Power Loss

SGs are advanced power infrastructures that integrate modern communication, monitoring, and control technologies to optimize electricity generation, distribution, and consumption [11]. They effectively incorporate intermittent and unpredictable renewable energy sources, ensuring efficient power distribution. However, power loss remains a significant challenge, categorized into technical and non-technical losses (NTL). Technical losses, primarily caused by resistance in electrical components and infrastructure deficiencies, lead to substantial energy wastage, increasing operational costs for utilities and energy bills for consumers. Power loss presents economic, operational, and environmental challenges. Utilities must generate extra electricity to compensate for losses, raising fuel consumption and expenses.

Additionally, inefficiencies in power distribution contribute to increased greenhouse gas emissions and resource depletion. Effective power loss management improves grid efficiency, reduces operational costs, and enhances sustainability [12]. Strategies such as real-time monitoring, advanced grid analytics, and improved infrastructure design mitigate power losses, ensuring a more reliable, sustainable, and economically viable power system.

2.2 DNN models for power loss

A Deep Neural Network (DNN) is structured to emulate the human brain's learning mechanism, operating similarly to biological neurons [13]. It comprises an input layer, multiple hidden layers, and an output layer. Each neuron in a layer connects to neurons in the preceding layer through weighted connections learned during training. A DNN typically includes at least two hidden layers, with outputs computed iteratively as neuron values propagate through the network. The input layer first receives the data for processing [14]. The neurons used in the input layer receive external data, such as the grid1-load, grid1-temp, etc. This is shown in Figure 2, while the output neurons in the final layer provide the predictions made by the DNN, which are then compared with the true target data, such as power loss. Amongst the input and output layers are hidden layers, often more than one. Data enters the DNN through the input layer, undergoes transformations in the hidden layers, and emerges as predictions in the output layer. The values of all neurons in the hidden and output layers are calculated using the formula: (sum of the values in the previous neurons × weight + bias), where weights and biases are adjusted based on the errors between predictions and targets until these errors are minimized. This adjustment process is the DNN's "learning" phase. The "depth" and "width" of a DNN refer to layers and neurons, respectively. Deeper and wider DNNs, with more layers and neurons, often exhibit stronger fitting abilities and higher prediction accuracy. However, excessive layers and neurons may lead to overfitting, where the model executes as scheduled, the training set is poorly depicted on the test set. Therefore, the goal of model development is to find an optimal DNN architecture that balances fitting ability and generalization.

2.3 DNN Training and Tuning

DNNs are advanced neural networks with multiple hidden layers. They consist of three layers: Input, Hidden (responsible for computation), and Output [15]. Training involves two phases: the Forward pass (feed-forward calculation) and the Backward pass (backpropagation for weight updates). Hyperparameters like batch size and learning rate impact training efficiency.

DNN training is distributed using data and model parallelism [16]. In data parallelism, the model is replicated across processes, each handling a dataset portion, with gradients shared for synchronized updates. Model parallelism splits the model itself across processes, using communication operations for training.Challenges in DNN training include overfitting, where the model learns noise instead of patterns, and underfitting, where it fails to capture essential features [17]. Vanishing/exploding gradients hinder convergence, and computational complexity demands extensive resources. Effective hyperparameter tuning, sufficient data quality, and architectural design are crucial for robust and accurate DNNs.

2.4 DNN Driven Solutions

Deep learning has advanced rapidly in recent years due to the growing complexity and scale of neural networks [18]. DNN architectures excel at modeling

https://dx.doi.org/10.4314/njt.v44i4.11

complex, nonlinear relationships across multiple layers, enabling the extraction of meaningful insights from raw data. In power systems, deep learning has been widely applied in four key areas: load forecasting, renewable energy prediction (wind and solar), power quality disturbance detection and classification, and equipment fault detection. These applications support tasks such as predicting electricity demand, optimizing renewable generation, and monitoring system health. Additionally, deep learning contributes to predictive maintenance, energy optimization, and intelligent grid management, enhancing reliability and sustainability. DNN-based models are particularly valued for their accuracy in pattern recognition, scalability, adaptability, and ability to learn features autonomously. However, they require high computational resources, risk overfitting, and often lack interpretability due to their "black box" nature. Despite these challenges, when properly managed, DNNs offer powerful solutions for optimizing and securing modern power systems.

2.5 The Effectiveness of DNN Parameter Tuning in Diverse Domains

Several studies have explored deep learning model optimization, particularly in hyperparameter tuning for DNNs. [19] proposed a Bayesian optimization framework that outperformed random search in enhancing intrusion detection on the NSL-KDD dataset. [20] Bayesian neural networks provide a probabilistic approach to model uncertainty and make predictions more robust, enhancing decision-making in scenarios with incomplete information[21]. Hyperparameter Optimization Techniques like Grid Search, Random Search, Bayesian Optimization, Genetic Algorithm (GA), Particle Swarm Optimization (PSO), aim to find the optimal set of hyperparameters that maximize the performance of the predictive models, ensuring that the models generalize well to unseen data[22]. These findings emphasize the importance of automated hyperparameter tuning in improving DNN performance across various applications, from power systems, cybersecurity, to medical diagnostics.

3.0 METHODOLOGY

The dataset-driven analytical framework (Figure 1) follows a structured sequence comprising: (i) Knowledge Base, (ii) Data Preprocessing, (iii) Data Splitting, (iv) Model Building, (v) DNN Model, and (vi) Smart Grid (SG) Power Loss Detection. Knowledge representation is achieved using both Star and Snowflake schemas. The Star schema supports multidimensional modeling for the datasets, while the Snowflake schema extends it by normalizing dimension tables into sub-dimensions for improved efficiency and structure. Data preprocessing is a vital stage that transforms raw data into an analyzable format, involving dataset size reduction, normalization, and outlier detection. Three datasets are used for developing, testing, and deploying the power loss prediction model: the Grid Loss Prediction Dataset, the Smart Grid Stability Dataset, and the Solar Power Generation Dataset. The Grid Loss dataset (from Kaggle) links grid losses to grid loads, featuring hourly data on grid consumption, losses, weather forecasts, and seasonal variations. The Smart Grid Stability dataset captures consumer demand, supply dynamics, and price elasticity, labeling grid conditions as stable or unstable based on system equations. The Solar Power Generation dataset represents large-scale photovoltaic (PV) plants, including structural and operational parameters. For training and evaluation, each dataset is split into 80% for training and 20% for testing. This partitioning ensures that the Deep Neural Network (DNN) model is both effectively trained and rigorously validated for power loss minimization analytics.

As shown in Figure 2, "n" signifies the neurons within the hidden layer, while n stands for the number of inputs, and m pertains to the count of outputs. In addition, x and P_L denote the input and output vectors of the Multi-Layer Perceptron (MLP), which encompass optimization variables and objective functions (or constraints) relevant to the smart grid.

These components are interconnected by specific weights, represented by the vector as shown in Equations (1) and (2).

$$'w_{11}^{1} \ldots\ldots w_{mN}^{1} \quad (1)$$

$$b_1^1 \ldots . \ b_m^3 \quad (2)$$

denotes the bias vector associated with all neurons except those in the input layer. The inputs are derived from the smart grid profile, like the load, temperature, and network, and they are represented as x_1 to x_n. The DNN further comprises 2 – 20 hidden layers, generating one distinct output: P_L, signifying power loss. Finally, the DNN output is calculated based on $\boldsymbol{x}$ and $\boldsymbol{w}$ according to the following formula in Equation (3):

https://dx.doi.org/10.4314/njt.v44i4.11

$P_L = f_{DNN}(x, w)$ (3)

where f_{DNN} Represents a non-linear activation function applied to all neurons. The inherent input-output correlation of power loss in the smart grid can be articulated as follows in Equation(4):

$P_L = f(x)$ (4)

The objective of MLP training is to determine an approximation, denoted as f_{DNN} Of the original relationship (*f*). This is achieved by employing the provided input-output training data and the weight factors of the MLP (w). The initial stage of the training process involves constructing a neural network comprising input-output pairs and an adequate count of hidden neurons. The data essential for training the neural network is typically acquired from smart grid profiles. Considering that the input values (x) and output values (P_L) might be of varying magnitudes, it becomes necessary to implement a scaling process before training in order to attain a more accurate model. This procedure is known as data scaling. The study employs a widely used scaling method known as linear scaling, which is outlined as follows in Equation (5):

$$\tilde{x} = \tilde{x}_{min} + \frac{x - x_{min}}{x_{max} - x_{min}} (\tilde{x}_{max} - \tilde{x}_{min}) \quad (5)$$

The corresponding formula for descaling is defined as follows in Equation (6):

$$x = x_{min} + \frac{\tilde{x} - \tilde{x}_{min}}{\tilde{x}_{max} - \tilde{x}_{min}} (x_{max} - x_{min}) \quad (6)$$

Here, $\tilde{x}$ Represents the scaled data vectors. A pivotal aspect of the model development process is the training of the DNN. This training process essentially involves an optimization procedure, commencing with initial weights and persisting until optimal weights are achieved through the minimization of the following Equation (7):

$$E_{D_{tr}}(w) = \frac{1}{2} \sum_{k \in Dtr} \sum_{j=1}^{m} |y_j(x_k, w) - d_{jk}|^2 \quad (7)$$

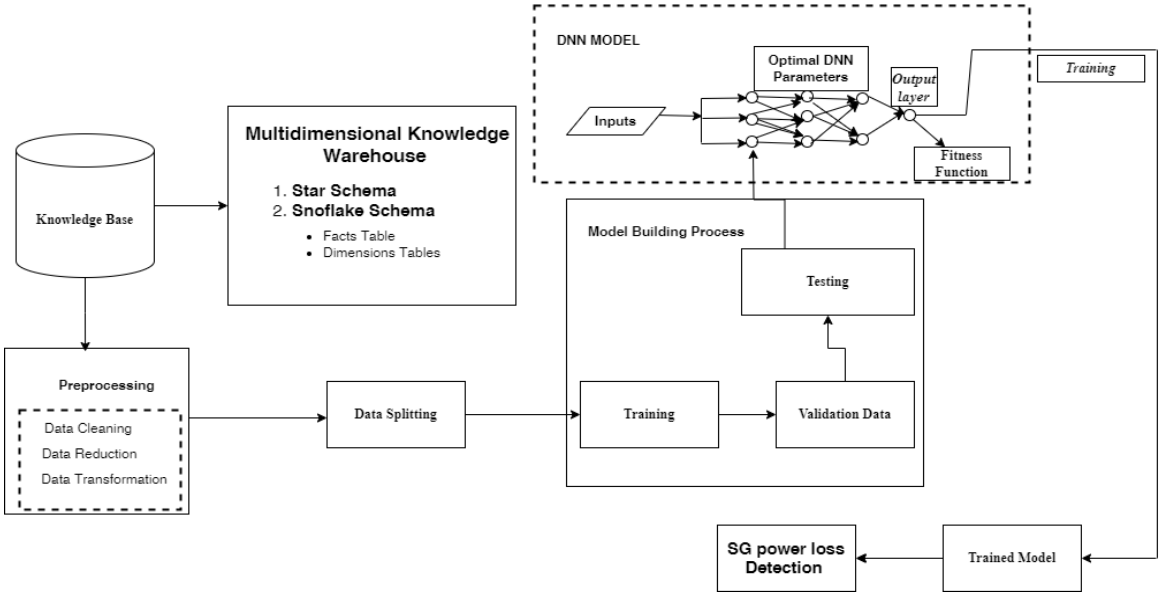

Figure 1: Dataset-driven analytic workflow for sg modeling

where $E_{D_{tr}}$ Represents the training error, in which ***x*** and ***d*** denote inputs and desired outputs of the model, and D_{tr} is the number of training samples. Also, $y_j(x_k, w)$ Is the j^{th} neural network output for the *k*th training sample. To minimize error, the gradient-based technique has been applied to update ***w***, according to the following formula, as shown in Equation (8).

$$w = w - \eta_{tr}(\partial E_{D_{tr}} / \partial w) \quad (8)$$

https://dx.doi.org/10.4314/njt.v44i4.11
http://creativecommons.org/licenses/by-nc-nd/4.0/

where η_{tr} denotes learning rate and $\partial \boldsymbol{E}_{D_{tr}} / \partial \mathrm{w}$ Is the gradient vector of the error concerning each weight? We used the backpropagation (BP) method to calculate the gradient, which is described here. Per-sample error function (E_k) is determined as follows in Equation (9):

$$\mathrm{E_k} = \frac{1}{2} \sum_{\mathrm{j=1}}^{\mathrm{m}} |\, \mathrm{y_j(x_k, w) - d_{jk}}|^2 \qquad (9)$$

The local error at the i^{th} neuron in the l^{th} layerδ_i^1 It is calculated and can be backpropagated from the output layer to hidden layers according to the following Equation (10):

$$\delta_i^1 = \frac{\partial \mathrm{E}^1}{\delta z_i^1} \left(\sum_{j=1}^{N_{l+1}} \delta_i^{1+1} w_{ji}^{l+1}\right) z_i^l (1 - z_i^l);$$
$$l = L-1, L-2, \ldots, 3, 2 \qquad (10)$$

where w_{ji}^lrepresents the weight between the j^{th} neurone of the l^{th} *layer* and the i^{th} neurone of $(l-1)^{\text{th}}$ layer and z_i^l Represents the output of the i^{th} neuron in the l^{th} layer. δ_i^1 Is the BP formulation supposed to be initialized from the last layer as (Equation 11):

$$\delta_i^1 = y_j(x_k, w) - d_k \qquad (11)$$

The derivative of the error E_k in Equation (3) concerning each weight parameter w_{ji}^l Is computed as follows:

$$\frac{\partial \boldsymbol{E}_k}{\partial w_{ji}^l} = \delta_i^l z_i^{l-1}, l = L, L-1, \ldots, 2 \qquad (12)$$

At long last, the derivative of all errors in Equation (7) concerning w_{ji}^l is given by

$$\frac{\partial \boldsymbol{E}_{Dtr}}{\partial w_{ji}^l} = \sum_{k T_r} \frac{\partial \boldsymbol{E}_k}{\partial w_{ji}^l} \qquad (13)$$

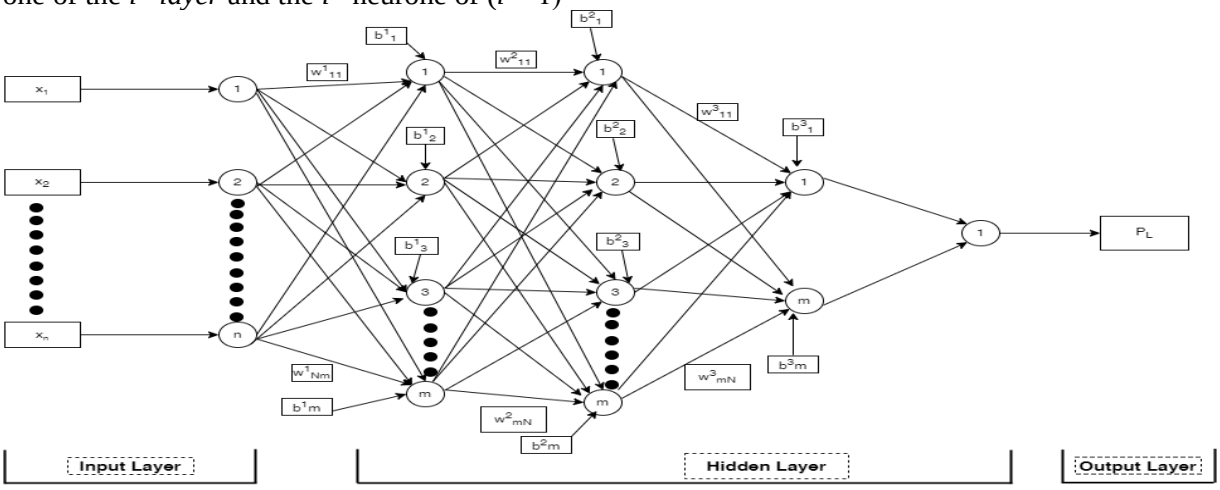

Figure 2: Layered DNN model for SG analytics

The gradient $\frac{\partial \boldsymbol{E}_{Dtr}}{\partial w_{ji}^l}$ It can be computed by using the BP algorithm and passed to the gradient-based training algorithm to get the weight update $\Delta \boldsymbol{w}$. Minimizing the training error ($\boldsymbol{E}_{Dtr}$) is the goal of a neural network training algorithm by update the weights. In this process, $\boldsymbol{w}$ is updated in each iteration of training that includes passing all training data to the
neural network simultaneously. The operation of this neural network transpires across two primary phases: training and testing. In the training phase, the architecture depicted in Figure 2 is employed.

Through the utilization of the backpropagation algorithm, the network's internal weights are iteratively adjusted to enhance performance. Following the completion of the training phase, the testing stage ensues. During this stage, when the smart grid profile serves as input to the neural network, the network undertakes information processing and subsequently provides estimations for power loss within the smart grid as outputs. This capability facilitates the evaluation of power loss

https://dx.doi.org/10.4314/njt.v44i4.11
http://creativecommons.org/licenses/by-nc-nd/4.0/

within the smart grid, leveraging the supplied profile and the trained DNN model.

3.1 DNN Activation Function Tuning

This work discusses several commonly used activation functions in neural networks, including Linear, Sigmoid, Tanh, ReLU, Softmax, and SELU. The **Linear** activation function directly scales the input but lacks complexity in learning nonlinear relationships. The **Sigmoid** function maps inputs to a range between 0 and 1, ideal for binary classification, but its lack of zero-centering can hinder learning. The **Tanh** function is similar but symmetric around zero, improving training by providing both positive and negative outputs, and is preferred for its steeper gradients. **ReLU** (Rectified Linear Unit) allows for faster training by activating only a subset of neurons, though it can lead to "dead neurons" when gradients become zero. The **Softmax** function generalizes Sigmoid for multi-class classification, outputting probabilities for each class. Finally, the **SELU** (Scaled Exponential Linear Unit) introduces self-normalization, enhancing performance in deep networks by preventing dying neurons [23].

[9] highlighted that various activation functions (AFs) have been explored in recent years for deep learning to achieve specific desirable properties. Key properties of effective AFs include a) introducing non-linear curvature in the optimization landscape to enhance training convergence; b) avoiding significant increases in the model's computational complexity; c) maintaining the gradient flow during training; and d) preserving data distribution to improve network training. This study adopts the ReLU activation function due to its beneficial characteristics, such as good sparsity, fast convergence, simple computation, and effective resolution of the gradient vanishing problem associated with sigmoid and tanh [24].

4.0 EXPERIMENTATION AND ANALYSIS

The Orange platform, Python, and Jupyter Notebook provided an intuitive environment for machine learning model development, helping grid operators utilize insights effectively [25][26]. DNN experiments varied the number of layers ranging between 2 and 20 hidden layers and activation functions namely linear, sigmoid, Tanh, Relu, SoftMax and Selu with datasets pre-processed by removing irrelevant features and missing records. Performance was evaluated using MSE, RMSE, MAE, R2, and MAPE, with results shown in Tables 4 to 7. Dimension reduction through PCA identified significant variables, with eigenvalues ≥1 indicating key features. In the Grid Loss Prediction dataset, grid-temp and grid-load explained 4.1E+01 and 1.6E+01 of variance. In SG Stability, tau1 to tau4 and p1 explained 5.1E+01, and in Solar Power Generation, the DATE_TIME attribute explained 5.0E-01 of variance. PCA involved calculating the covariance matrix, obtaining eigenvalues/eigenvectors, ordering eigenvalues, selecting relevant eigenvectors, and identifying key factors for each dataset.

Table 1: Eigenvalues and ratio of explained variance for grid loss prediction dataset

S/N	Attribute	Eigenvalue	Explained Proportion Ratio	proportion	Commutative
1	grid-temp	4.5E+00	4.0E-01	4.1E+01	4.1E+01
2	grid-load	1.8E+00	1.5E-01	1.6E+01	5.7E+01
3	month_x	1.2E+00	1.0E-01	1.1E+01	6.7E+01
4	grid1-loss	1.0E+00	9.0E-02	9.1E+00	7.6E+01
Total		8.4E+00	7.6E-01	7.6E+01	2.4E+02

Table 2: Eigenvalues and ratio of explained variance for SG stability dataset

S/N	Attribute	Eigenvalue	Explained Proportion Ratio	Proportion	Commutative
1	tau1	2.0E+00	1.6E-01	1.7E+01	1.7E+01
2	tau2	1.0E+00	8.0E-02	8.5E+00	2.5E+01
3	tau3	1.0E+00	8.0E-02	8.5E+00	3.4E+01
4	tau4	1.0E+00	8.0E-02	8.4E+00	4.2E+01
5	p1	1.0E+00	8.0E-02	8.4E+00	5.1E+01
Total		6.1E+00	5.0E-01	5.1E+01	1.7E+02

https://dx.doi.org/10.4314/njt.v44i4.11
http://creativecommons.org/licenses/by-nc-nd/4.0/

Table 3: Eigenvalues and ratio of explained variance for solar power generation dataset

S/N	Attribute	Eigenvalue	Explained Proportion Ratio	Proportion	Commutative
1	DATE_TIME	1.0E+00	5.0E-01	5.0E+01	5.0E+01
Total		1.0E+00	5.0E-01	5.0E+01	5.0E+01

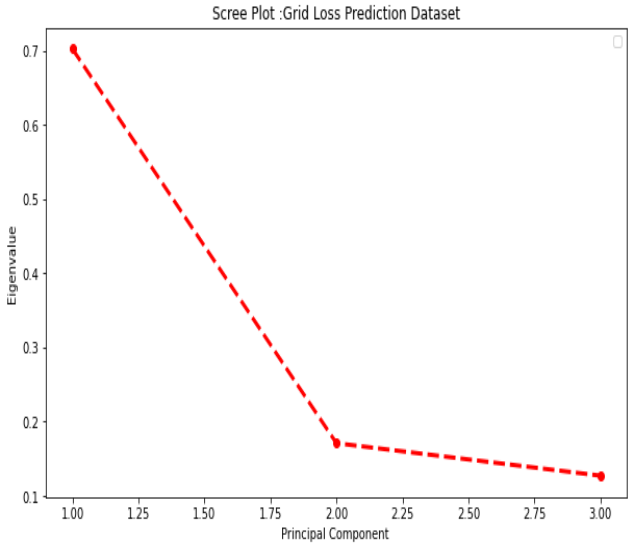

Figure 3: Eigenvalue representation for grid loss prediction dataset

Figure 3 shows an eigenvalue drop from 0.7 at the first component to 0.2 by the third, indicating that the first two to three principal components capture most of the data variance. This suggests that dimensionality reduction to a few key components enhances computational efficiency and optimizes the neural network's performance for power loss minimization.

Figure 4: The eigenvalue trend decreases from 0.204 to 0.198 by the second component before stabilizing, forming an elbow pattern. This implies that the first two principal components hold most of the variance, allowing the model to reduce dimensionality effectively while maintaining accuracy and efficiency in power loss prediction.

Eigenvalues decline sharply from 0.510 to 0.490 between the first two components and then level off, showing that these components capture the majority of the variance, as shown in Figure 5. Using them ensures efficient dimensionality reduction, improving neural network accuracy and speed in solar power loss minimization

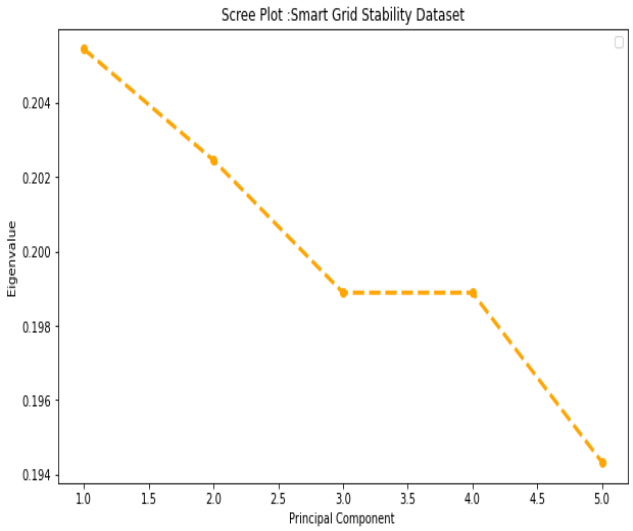

Figure 4: Eigenvalue representation for sg stability dataset

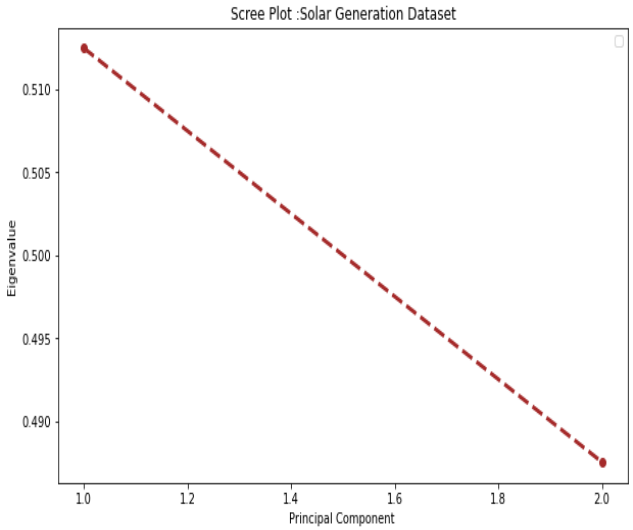

Figure 5: Eigenvalue representation for smart solar power generation dataset

4.1 Optimal Activation Function Selection

Tables 4 and 5 compare the performance of various activation functions in DNNs across multiple metrics with different hidden layer configurations. For the two-layer setup, six activation functions, Linear, Sigmoid, Tanh, ReLU, SoftMax, and SeLU, were evaluated using MSE, RMSE, MAE, R^2, MAPE, and computation time. Sigmoid, Tanh, and ReLU showed strong performance with low error values and high

https://dx.doi.org/10.4314/njt.v44i4.11
http://creativecommons.org/licenses/by-nc-nd/4.0/

R^2 (≈0.9), indicating a good fit, while SoftMax performed poorly with higher errors and negative R^2. Overall, Sigmoid slightly outperformed others due to its consistently low MSE and RMSE

The three-hidden-layer evaluation, shown in Table 5 and illustrated in Figure 6, produced results consistent with the two-layer setup. Sigmoid, Tanh, and ReLU remained the top performers, achieving low error values and high R^2 scores, while SoftMax again showed the weakest results with higher errors and negative R^2. Training time decreased overall, with Sigmoid and Tanh delivering the best balance of accuracy and computational efficiency.

The performance of the DNN using the sigmoid activation function across different numbers of hidden layers, ranging from 2 to 20, is depicted in Table 6. The key insights are as follows: the performance metrics (MSE, RMSE, MAE, R^2, MAPE) remained stable and optimal with up to 10 hidden layers. However, beyond 10 layers, there was a significant deterioration in performance, with increased errors and negative R^2 values, indicating overfitting or model instability. This is depicted in Figure 7 for visualization.

Table 4: Performance activation function with two hidden layers

	Activation Function	**MSE**	**RMSE**	**MAE**	**R^2**	**MAPE**	**Time**
1	Linear	2E-03	4E-02	2E-02	9E-01	7E+00	0:07:34
2	Sigmoid	1E-03	4E-02	2E-02	9E-01	5E+00	0:09:13
3	Tanh	1E-03	4E-02	2E-02	9E-01	6E+00	0:10:12
4	Relu	1E-03	4E-02	2E-02	9E-01	5E+00	0:11:10
5	SoftMax	2E-01	5E-01	5E-01	-1E+01	1E+02	0:04:47
6	Selu	1E-03	4E-02	2E-02	9E-01	6E+00	0:05:23

Table 5: DNN Performance with different activation functions in three hidden layers

S/N	**Activation Function**	**MSE**	**RMSE**	**MAE**	**R^2**	**MAPE**	**Time**
1	Linear	2E-03	4E-02	3E-02	9E-01	7E+00	0:03:21
2	Sigmoid	1E-03	4E-02	2E-02	9E-01	5E+00	0:03:25
3	Tanh	1E-03	3E-02	2E-02	9E-01	5E+00	0:03:46
4	Relu	1E-03	4E-02	2E-02	9E-01	5E+00	0:04:22
5	SoftMax	2E-01	5E-01	5E-01	-1E+01	1E+02	5:26:03
6	Selu	1E-03	4E-02	2E-02	9E-01	5E+00	0:03:11

Table 6: DNN performance comparison with sigmoid activation in different layers

PERFORMANCE METRICS	**NUMBER OF HIDDEN LAYERS**										
	2	**3**	**4**	**6**	**8**	**10**	**12**	**14**	**16**	**18**	**20**
MSE	1E-03	1E-03	1E-03	1E-03	1E-03	1E-03	2E-02	2E-02	2E-02	2E-02	2E-02
RMSE	4E-02	4E-02	4E-02	4E-02	4E-02	4E-02	1E-01	1E-01	1E-01	1E-01	1E-01
MAE	2E-02	2E-02	2E-02	2E-02	2E-02	2E-02	1E-01	1E-01	1E-01	1E-01	1E-01
R^2	0.9	0.9	0.9	0.9	0.9	0.9	-2.0	-1.0	-8.0	-6.0	-7.0
MAPE	5.0	5.0	5.0	5.0	5.0	5.0	30	30	30	30	30
TIME	0:09:13	0:03:25	0:05:00	0:05:31	0:08:13	0:03:26	0:05:27	0:17:47	0:03:51	0:04:41	0:04:02

The comparative analysis of DNNs using the ReLU activation function, illustrated in Figure 8 and detailed in Table 7, evaluates models with different hidden layer counts across metrics such as MSE, RMSE, MAE, R^2, MAPE, and training time. Networks with 2, 3, 4, and 6 layers performed well, while the 12-layer model showed increased errors and lower R^2, indicating overfitting. As shown in Figure 9, training loss continues to drop while validation loss rises, confirming poor generalization. MAPE fluctuations reveal varying prediction accuracy, and training time grows with depth,

https://dx.doi.org/10.4314/njt.v44i4.11

emphasizing the trade-off between complexity and efficiency.

Figure 7 provides a detailed assessment of the Sigmoid activation function's performance across metrics such as Time, MAPE, R^2, MAE, RMSE, and MSE for optimizing deep neural networks in power loss reduction. The Time metric peaks around the 10th point, indicating higher computational effort during intensive learning. Despite this, MAPE and other error metrics remain low and stable, reflecting reliable prediction accuracy. However, the negative dip in R^2 suggests that Sigmoid struggles to capture the full variance in power loss data, implying that further tuning or alternative activation functions may improve the model's predictive effectiveness.

Table 7: DNN performance comparison with relu activation in different layers

PERFORMANCE METRICS	HIDDEN LAYERS										
	2	**3**	**4**	**6**	**8**	**10**	**12**	**14**	**16**	**18**	**20**
MSE	1E-03	1E-03	1E-03	1E-03	1E-03	1E-03	2E-02	2E-02	2E-02	2E-02	2E-02
RMSE	4E-02	4E-02	4E-02	4E-02	4E-02	4E-02	1E-01	1E-01	1E-01	1E-01	1E-01
MAE	2E-02	2E-02	2E-02	2E-02	2E-02	2E-02	1E-01	1E-01	1E-01	1E-01	1E-01
R2	0.9	0.9	0.9	0.9	0.9	0.9	-2.0	-1.0	-8.0	-6.0	-7.0
MAPE	5.0	5.0	5.0	5.0	5.0	5.0	3.0	3.0	3.0	3.0	3.0
TIME	0:04:38	0:03: 25	0:05: 00	0:05: 31	0:08: 13	0:03: 26	0:05: 27	0:17: 47	0:03: 51	0:04: 41	0:04: 02

Figure 6 compares the performance of different activation functions, Linear, Sigmoid, Tanh, ReLU, Softmax, and SeLU, across metrics such as MSE, RMSE, MAE, MAPE, and computation time for optimizing deep neural network parameters in power loss minimization. The SeLU function, shown by the green line, exhibits a sharp drop at the MAPE metric, indicating its effectiveness in minimizing prediction errors with three hidden layers. Linear, Sigmoid, and Tanh show stable but less adaptable performance, while ReLU and Softmax maintain low metric values. SeLU's fluctuations highlight its sensitivity to network depth, making it crucial for achieving optimal power loss reduction.

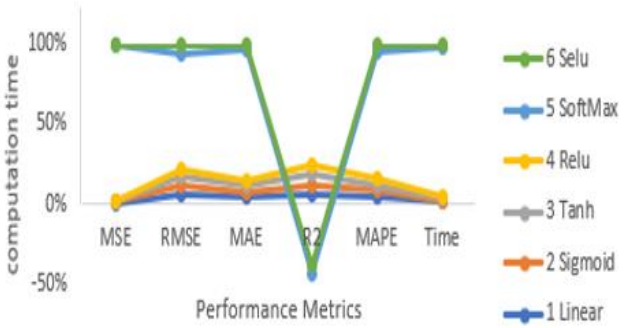

Figure 6: Graph comparing activation in different layers on SG loss prediction dataset (three hidden layers)

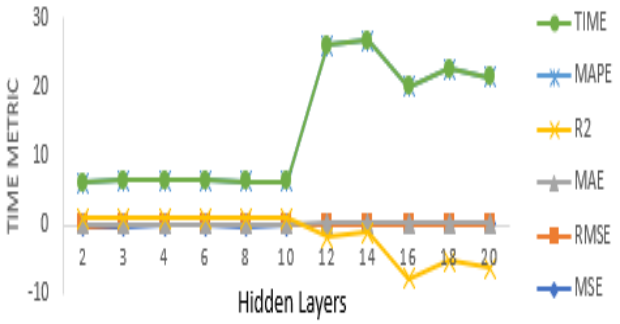

Figure 7: Graph comparing sigmoid activation in different layers on SG loss prediction dataset (sigmoid comparison)

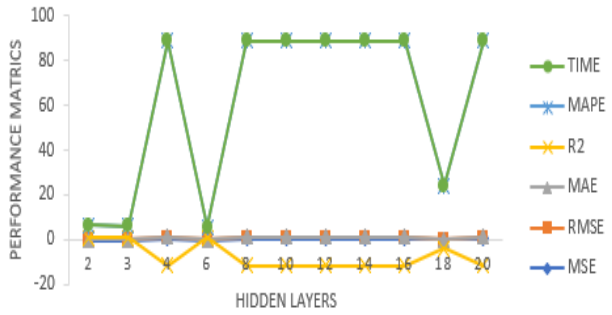

Figure 8: Graph comparing ReLU activation in different layers on SG loss prediction dataset (Relu comparison)

Figure 8 evaluates the ReLU activation function's performance across metrics such as Time, MAPE, R^2, MAE, RMSE, and MSE with different hidden layer counts. The Time metric peaks at about 80% for models with 3–10 hidden layers, indicating higher computational demand due to complexity. MAPE

https://dx.doi.org/10.4314/njt.v44i4.11
http://creativecommons.org/licenses/by-nc-nd/4.0/

and other error metrics remain low and stable, showing strong predictive accuracy. However, R^2 drops sharply when layer numbers fall outside the optimal 6–10 range, suggesting ReLU performs best in moderately deep architectures. Proper tuning of hidden layers is therefore crucial for achieving optimal power loss minimization.

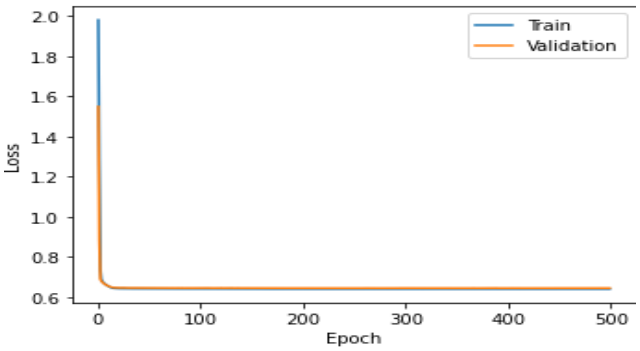

Figure 9: Train and validation loss for smart grid loss prediction dataset overfitting

4.2 Layer Adoption to the Dataset

The performance of DNN models using the ReLU activation function was tested on two datasetssmart grid stability and solar power generation, as shown in Tables 8 and 9. For the smart grid stability dataset (Table 8), the model achieved MSE = 6E-03, RMSE = 8E-02, MAE = 6E-02, R^2 = 0.8, and MAPE = 2E+00, with a training time of 0:51:07, indicating accurate predictions and strong variance explanation. For the solar power generation dataset (Table 9), results showed MSE = 1E-02, RMSE = 4E-01, MAE = 3E-01, R^2 = 0.9, and MAPE = 1E+00, with the same training duration. These findings highlight the robustness and efficiency of ReLU in modeling complex energy data for both stability assessment and generation forecasting.

Table 8: Relu activation in smart grid stability dataset

S/N	Activation Function	MSE	RMSE	MAE	R^2	MAPE	Time
1	Relu	6E-03	8E-02	6E-02	8E-01	2E+00	0:51:07

Table 9: Relu activation in solar power generation dataset

S/N	Activation Function	MSE	RMSE	MAE	R^2	MAPE	Time
1	Relu	1E-02	4E-01	3E-01	9E-01	1E+00	0:51:07

5.0 RESULTS AND DISCUSSION

The results and discussion section evaluates DNN performance using different activation functions (AFs) across multiple datasets and hidden layer configurations to assess accuracy and efficiency. Tables 4 and 5 compare six AFS Linear, Sigmoid, Tanh, ReLU, SoftMax, and SeLU using two and three hidden layers. Sigmoid, Tanh, and ReLU achieved strong results with low MSE (1E-03), RMSE (4E-02), MAE (2E-02), and high R^2 (0.9), while SoftMax performed poorly with high errors and negative R^2 (−1E+01). Sigmoid showed a slight advantage due to consistent accuracy and shorter training times. Table 6 analyzes DNNs using the Sigmoid function with 2–20 hidden layers. Performance remained stable up to 10 layers (MSE = 1E-03, R^2 = 0.9) but declined beyond that point, showing higher errors (MSE = 2E-02) and negative R^2 values, indicating overfitting and reduced generalization. Table 7 examines ReLU across varying layers, showing strong performance up to 6 layers (MSE = 1E-03, R^2 = 0.9) but degradation with deeper networks (MSE = 2E-02, negative R^2), suggesting overfitting and higher computational cost. Training time increased with network depth, highlighting the need to balance complexity and efficiency. Overall, ReLU proved advantageous with six hidden layers due to its nonlinear mapping, computational speed, and ability to mitigate the vanishing gradient problem. As shown in Tables 8 and 9, ReLU's efficiency and stability make it effective for both smart grid stability and solar power generation datasets, reinforcing its suitability for power loss minimization analytics.

6.0 CONCLUSION

The study highlights the significant impact of using the ReLU activation function in optimizing DNN for power loss analytics in smart grids. ReLU outperforms other AFS such as sigmoid, linear, softmax, SELU, and tanh by enhancing sparsity, accelerating convergence, and effectively addressing the gradient vanishing issue. The DNN model, featuring six hidden layers with ReLU activation, achieved exceptional performance metrics, including an MSE of 1.0E-03, RMSE of 3.4E-02, MAE of 1.9E-02, R2 of 9.4E-01, and MAPE of 4.8E+00. These results underscore the model's high accuracy

https://dx.doi.org/10.4314/njt.v44i4.11

and reliability in predicting power loss. The study's rigorous approach, encompassing a comprehensive literature review, meticulous data collection, and advanced preprocessing techniques, ensures robust and effective model tuning. This optimized framework provides a powerful tool for enhancing power loss analytics in smart grids, demonstrating its practical value in improving grid management. Future research may benefit from exploring additional activation functions, deeper network configurations, and real-world applications to further refine and validate the model's performance.

ACKNOWLEDGMENT

We express our gratitude to the Tertiary Education Trust Fund (TETFund) for supporting this research through the TETFund Centre of Excellence in Computational Intelligence at the University of Uyo. We also thank the management of the University of Uyo for providing a conducive environment.

REFERENCES

[1] Agarwala, A. Tahsin, T. Ali, M. F. Sarker, S. K. Abhi, S. H. Das, S. K. and Ahamed, M. H. "Towards Next Generation Power Grid Transformer for Renewables: Technology Review," *Engineering Reports*, 6(4), p. e12848, 2024.

[2] Olanite, O. Nwohu, M. N. Adegboye, B. A. and Tola, O. J. "Large Scale Penetration Impact of PMS-WTGS on Voltage Profile and Power Loss for 5-Bus, 330kV System of the Nigerian Grid," *Nigerian Journal of Technology*, 43(1), pp. 115–122, 2024.

[3] Sadiq, E. H. and Antar, R. K. "Minimizing Power Losses in Distribution Networks: A Comprehensive Review," *Chinese Journal of Electrical Engineering*, 10(4), pp. 20–36, 2024.

[4] Ghahramani, M. Habibi, D. Ghamari, S. Soleimani, H. and Aziz, A. "Renewable-Based Isolated Power Systems: A Review of Scalability, Reliability, and Uncertainty Modeling," *Clean Technologies*, 7(3), p. 80, 2025.

[5] Strielkowski, W. Vlasov, A. Selivanov, K. Muraviev, K. and Shakhnov, V. "Prospects and Challenges of the Machine Learning and Data-Driven Methods for the Predictive Analysis of Power Systems: A Review," *Energies*, 16(10), p. 4025, 2023.

[6] Douglass, F., "Deep Learning and Neural Networks: Methods and Applications," *Online Publication*, 2023. DOI: 10.59646/csebookc8/004.

[7] Tiep, N. H. Jeong, H. Kim, K. Mung, N. X. Dao, N. Tran, H. Hoang, V. Anh, N. N. and Vu, M. "A New Hyperparameter Tuning Framework for Regression Tasks in Deep Neural Network: Combined-Sampling Algorithm to Search the Optimized Hyperparameters," *Mathematics*, 12(24), p. 3892, 2024. DOI: 10.3390/math12243892.

[8] Al-Jamimi, H. A. BinMakhashen, G. M. Worku, M. Y. and Hassan, M. A. "Advancements in Household Load Forecasting: Deep Learning Model with Hyperparameter Optimization," *Electronics*, 12(24), p. 4909, 2023. DOI: 10.3390/electronics12244909.

[9] Hammad, M. M. "Deep Learning Activation Functions: Fixed-Shape, Parametric, Adaptive, Stochastic, Miscellaneous, Non-Standard, Ensemble," arXiv preprint arXiv:2407.11090, 2024.

[10] Inyang, U. G. Ijebu, F. F. Osang, F. B. Afoluronsho, A. A. Udoh, S. S. and Eyoh, I. J. "A Dataset-Driven Parameter Tuning Approach for Enhanced K-Nearest Neighbour Algorithm Performance," *International Journal of Advanced Science, Engineering and Information Technology*, 13(1), pp. 380–388, 2023. DOI: 10.18517/ijaseit.13.1.16706.

[11] Al-Shetwi, A. Q. Hannan, M. A. Al-Masri, H. M. and Sujod, M. Z. "Latest Advancements in Smart Grid Technologies and Their Transformative Role in Shaping the Power Systems of Tomorrow: An Overview," *Progress in Energy*, 2024.

[12] Kanimozhi, K. Neelaveni, P. Seethalakshmi, K. Rao, N. Prabhu, M. and Naganathan, S. B. T. "Implementing Real-Time Analytics for Enhanced Energy Efficiency in IoT-Integrated Smart Grid Systems," Proceedings of the IEEE ICCSP Conference, pp. 762-766, 2024. DOI: 10.1109/iccsp60870.2024.10543583.

[13] Naik, M. A. and Singh, E. J. "Optimization of Deep Neural Networks Using DAPP (DNN Acceleration Using Ping-Pong) Approach," *International Journal for Research in Applied Science and Engineering Technology*, 11(1), pp. 1099–1108, 2023. DOI: 10.22214/ijraset.2023.48771.

[14] Duong, H. Xu, D. Nguyen, T. and Dwyer, M. B. "Harnessing Neuron Stability to Improve DNN Verification," *Proceedings of the ACM on Software Engineering*, 1(FSE), pp. 859–881, 2024.

[15] Edwin Singh, C. and Celestin Vigila, S. M. "WOA-DNN for Intelligent Intrusion Detection and Classification in MANET Services," *Intelligent Automation & Soft Computing*, 35(2), pp. 1–10, 2023.

[16] Al Maruf, M. Azim, A. Auluck, N. and Sahi, M. "Optimizing DNN Training with Pipeline Model Parallelism for Enhanced Performance in Embedded Systems," *Journal of Parallel and Distributed Computing*, 190, p. 104890, 2024.

[17] Nie, C. Maghakian, J. and Liu, Z. "Training DNN Models Over Heterogeneous Clusters with Optimal Performance," arXiv preprint arXiv:2402.05302, 2024. DOI: 10.48550/arxiv.2402.05302.

[18] Aktemur, E. Zorlutuna, E. Bilgili, K. Bok, T. E. Yanikoglu, B. and Mutluergil, S. O. "Going Forward-Forward in Distributed Deep Learning," arXiv preprint arXiv:2404.08573, 2024.

[19] Fristiana, A. H. Alfarozi, S. A. I. Permanasari, A. E. Pratama, M. and Wibirama, S. "A Survey on Hyperparameters Optimization of Deep Learning for Time Series Classification," *IEEE Access*, 2024.

[20] Li, X. Tai, X. Shi, H. Liu, L. and Xu, J. "DNN-Based Method to Solve Absolute Value Equations Derived from Signal Processing," Proceedings of the IEEE ICMMT Conference, 2023. DOI: 10.1109/icmmt58241.2023.10277495.

[21] Ruan, J. Liang, G. Zhao, J. Zhao, H. Qiu, J. Wen, F. and Dong, Z. Y. "Deep Learning for Cybersecurity in Smart Grids: Review and Perspectives," *Energy Conversion and Economics*, 4(4), pp. 233–251, 2023.

[22] Malakouti, S. M. Menhaj, M. B. and Suratgar, A. A. "Applying Grid Search, Random Search, Bayesian Optimization, Genetic Algorithm, and Particle Swarm Optimization to Fine-Tune the Hyperparameters of the Ensemble of ML Models Enhances Its Predictive Accuracy for Mud Loss," *Engineering Reports*, 2024.

[23] Çatalbaş, B. and Morgül, Ö. "Deep Learning with Extended Exponential Linear Unit (DELU)," *Neural Computing and Applications*, 35(30), pp. 22705–22724, 2023.

[24] Waqas, N. Islam, M. Yahya, M. Habib, S. Aloraini, M. and Khan, S. "Med-ReLU: A Parameter-Free Hybrid Activation Function for Deep Artificial Neural Network Used in Medical Image Segmentation," *Computers, Materials & Continua*, 84(2), pp. 1–10, 2025.

[25] Domingues, I. and Rasteiro, D. M. L. D. "A Paradigm Shift in Teaching Machine Learning to Sustainable City Management Master Students," Proceedings of the 16th International Conference on Education Technology and Computers, pp. 413–417, 2024.

[26] Venkatesh, A. P. S. Wang, J. Li, L. and Bodden, E. "Enhancing Comprehension and Navigation in Jupyter Notebooks with Static Analysis," *Proceedings of the IEEE SANER Conference*, pp.391-401, 2023. DOI: 10.1109/saner56733.2023.00044.