Comparative Load Flow Analysis Using Artificial Neural Networks and the Newton-Raphson Method

Abstract—**Accurate load flow analysis is crucial for power system studies. While traditional methods like the Newton-Raphson algorithm are widely used, deep learning (DL) techniques are emerging as promising alternatives. This paper presents a new approach that utilizes an Artificial Neural Network (ANN) model to predict load flow solutions, reducing the reliance on iterative calculations. Load flow analysis is first performed using the Newton-Raphson method to generate data for training the ANN, which is then used to predict system variables based on load inputs. The ANN's predictions are compared with those from the Newton-Raphson method to assess both accuracy and computational efficiency. Tested on the IEEE-4 bus system, the ANN model successfully replicates the load flow results, offering potential advantages in managing complex, nonlinear load patterns. This study illustrates the growing intersection between classical power system analysis methods and artificial neural network architectures.**

Keywords - Load Flow Analysis, Newton-Raphson Method, Deep Learning, IEEE 4 Bus System, Artificial Neural Network.

I. INTRODUCTION

The increasing complexity of modern power systems, driven by the integration of renewable energy sources, electric vehicles, and advanced load types, has created a need for more sophisticated tools to analyze system behavior and ensure reliable operation. Electrical distribution networks, which serve as the most extensive components in these systems, connect end-users to national or regional transmission systems via substations, playing a critical role in delivering electricity [1], [2]. These networks are undergoing rapid transformation due to the large-scale integration of distributed energy resources such as renewable generation and energy storage devices, turning traditional grids into active, dynamic systems [1].

As the power system evolves, optimal power flow (OPF) becomes increasingly crucial for maintaining system stability and efficiency [3]. OPF is a foundational element of power system operations, regularly solved in real-time markets every five minutes to ensure optimal decision-making. It also supports market-clearing processes and reliability optimizations that determine day-ahead and look-ahead commitments. However, the growing uncertainty from renewable generation, distributed energy resources, and extreme weather events has made it challenging to maintain the optimal operating point, especially during real-time operations. This dynamic and unpredictable environment demands advanced approaches to system analysis.

One of the key tools for assessing the performance and stability of power systems is load flow analysis, also known as power flow analysis. It provides essential insights into voltage levels, power losses, and overall system stability under various operating conditions [4]. Traditional load flow methods, such as the Newton-Raphson and Fast Decoupled algorithms, have been widely used due to their computational efficiency and robustness, particularly in large-scale systems. However, these classical methods face limitations in dealing with the nonlinearities and real-time demands of modern power systems, where load patterns can change unpredictably, and quick, adaptive decision-making is required. To address these challenges, machine learning (ML) and artificial intelligence (AI) have emerged as promising alternatives for power system analysis. These techniques are well-suited for handling complex, nonlinear relationships within large datasets and offer faster, more flexible solutions. Among these, deep learning (DL), a subset of ML, has shown exceptional potential in power systems due to its ability to generalize across different load conditions and manage vast amounts of data. By learning from historical load flow data, DL models can predict system behavior under new conditions, offering a powerful tool for improving both speed and adaptability in system operations.

This paper explores the application of deep learning to load flow analysis, specifically comparing its performance to the traditional Newton-Raphson method. The load flow solution is first obtained using the Newton-Raphson algorithm in MATLAB for the IEEE-4 bus system. A deep learning model is then trained on the same input data to predict system voltage and power flow values. The results of the deep learning model are evaluated against those of the Newton-Raphson method, focusing on both accuracy and computational efficiency. By integrating deep learning into load flow analysis, this approach presents an opportunity to enhance system operations, particularly in scenarios characterized by high load variability, uncertainty from distributed energy resources, and rapidly changing system conditions.

The remainder of this paper is organized as follows: Section II provides an overview of the traditional Newton-Raphson method and its application in load flow analysis. Section III discusses the architecture of the deep learning model and the training process employed in this study. In Section IV, the results from both the Newton-Raphson method and the deep learning model are compared and analyzed. Finally, Section V concludes the paper with a discussion of potential future applications of machine learning in power system analysis.

II. NEWTON RAPHSON METHOD

The Newton-Raphson method is a widely used and accurate algorithm for solving nonlinear load flow equations in power systems [5]. It operates by iteratively solving for the unknown bus voltages, which include both voltage magnitudes and phase angles, based on specified real and reactive power injections

A. Method Overview

In load flow analysis, the real power P_i and reactive power Q_i at each bus are nonlinear functions of the bus voltages. These are expressed as:

$$P_i = V_i \sum_{j=1}^{n} V_j \left(G_{ij} \cos(\theta_i - \theta_j) + B_{ij} \sin(\theta_i - \theta_j)\right) \tag{1}$$

$$Q_i = V_i \sum_{j=1}^{n} V_j \left(G_{ij} \sin(\theta_i - \theta_j) - B_{ij} \cos(\theta_i - \theta_j)\right) \tag{2}$$

The Newton-Raphson method solves these nonlinear power equations by approximating the system using a Taylor series expansion around an initial guess and iteratively refining the solution.

B. Jacobian Matrix

At the heart of the Newton-Raphson method is the Jacobian matrix, which contains partial derivatives of the power equations with respect to the voltage magnitudes and phase angles. The power mismatch ΔP and ΔQ are given as:

$$\Delta P_i = P_i^{\text{spec}} - P_i^{\text{calc}} \tag{3}$$

$$\begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} = \begin{bmatrix} J_1 & J_3 \\ J_2 & J_4 \end{bmatrix} \begin{bmatrix} \Delta \theta \\ \Delta V \end{bmatrix} \tag{4}$$

where $J_1 = \frac{\partial P}{\partial \theta}$ (Partial derivative of real power w.r.t. voltage angles),
$J_2 = \frac{\partial P}{\partial V}$ (Partial derivative of real power w.r.t. voltage magnitudes),
$J_3 = \frac{\partial Q}{\partial \theta}$ (Partial derivative of reactive power w.r.t. voltage angles),
$J_4 = \frac{\partial Q}{\partial V}$ (Partial derivative of reactive power w.r.t. voltage magnitudes).

C. Iterative Process

- **Initial Guess:** Voltage magnitudes and phase angles are initialized, typically with a flat start.
- **Power Mismatch Calculation:**The difference between specified and calculated power values is computed.
- **Jacobian Update:**The Jacobian matrix is evaluated using the current voltage estimates.
- **Voltage Update:** Corrections to voltage magnitudes and angles are computed by solving the linearized system of equations.
- **Convergence:**The process is repeated until power mismatches fall below a set tolerance level.

D. Advantages and Application

The Newton-Raphson method is favored for its quadratic convergence, meaning that it rapidly approaches the solution with fewer iterations. It is robust for large, complex systems, but requires more computation due to repeated updates of the Jacobian matrix. In this study, the Newton-Raphson method is applied using MATLAB on the IEEE-4 bus system, serving as a benchmark to compare the results of a deep learning-based load flow model.

III. Dataset Generation And Preprocessing

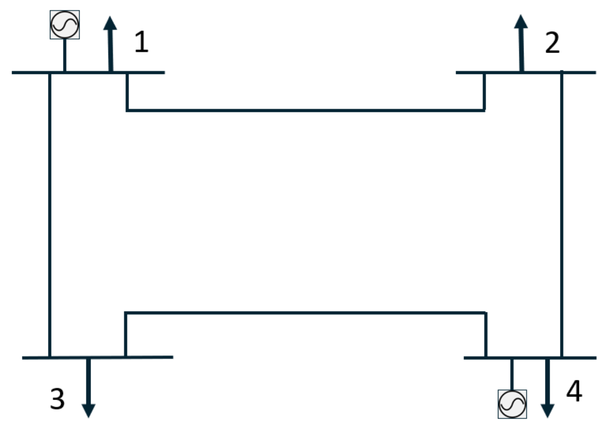


Fig. 1: Single line diagram of IEEE-4 bus system

The dataset was generated using the Newton-Raphson method, with two types of input data provided for its creation. These two categories of data are:

- Bus data
- Line data

These two tables were used as input for the Newton-Raphson model to generate the power flow dataset, as real-time data were unavailable for evaluation. After applying the bus and line data, the Newton-Raphson method output was used to train the ANN model. The data variables are shown in Figure 2.

Later, data from the *Bus data* table and *Power flow output data* were used to train the ANN architecture and build a complete data set. The goal was to see if the ANN model could forecast the output like the Newton-Raphson model. The project achieved accurate power flow measurement using simply *Bus data*, yielding encouraging findings in analysis. To improve the evaluation and performance of the model, the data set was normalized before feeding it into the ANN model. Normalization scales data between 0 and 1, preventing larger values from dominating learning. This pre-processing step improves model convergence speed and stability during training, resulting in more accurate and efficient outcomes. Implementing the Proposed ANN Model Deep learning affects power flow analysis. A variety of Artificial Neural Networks can solve power flow analysis issues.

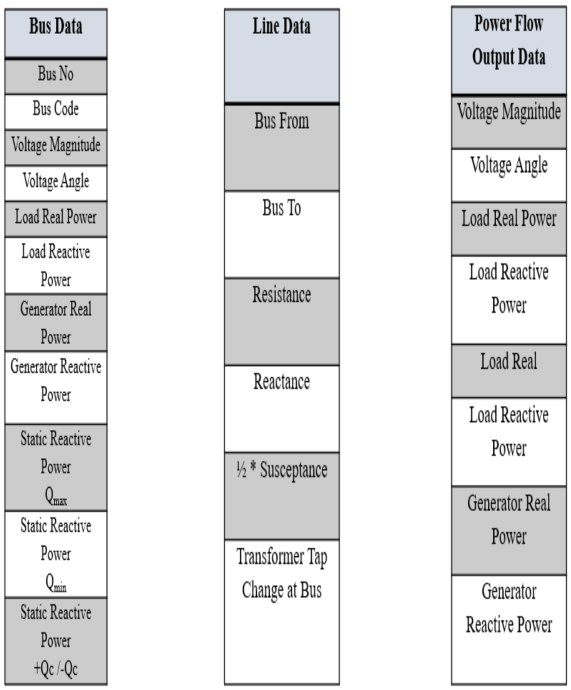


Fig. 2: Data variables for power flow analysis

IV. Proposed ANN Model Architecture

In order to conduct an accurate analysis of the power flow data set, the artificial neural network (ANN) model that has been presented includes dense, batch normalization, and dropout layers. Table I includes a representation of the usual architecture of the ANN model that has been proposed.

TABLE I: Proposed ANN model architecture

Layer	Output Shape	Parameters
Dense	250	3,000
BatchNormalization	250	1,000
Dropout	250	0
Dense	200	50,200
BatchNormalization	200	800
Dropout	200	0
Dense	150	30,150
BatchNormalization	150	600
Dropout	150	0
Dense	100	15,100
BatchNormalization	100	400
Dropout	100	0
Dense	1	101

A. Dense Layer

A dense layer, also known as a fully connected layer, is a fundamental building block in neural networks [6]. The dense layer consists of different architectures.

1) **Fully Connected Layer:** Neural networks consist of a set of dependent nonlinear functions. Each individual function comprises a neuron (or a perceptron).

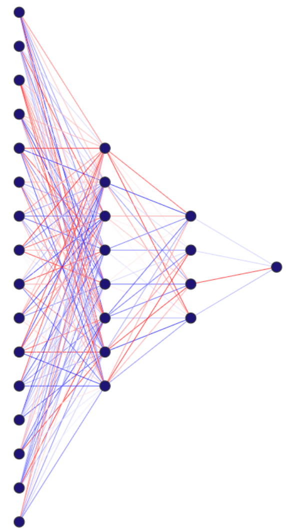


Fig. 3: ANN model architecture

In fully connected layers, each neuron applies a linear transformation to the input vector using a weight matrix. Then a non-linear transformation is applied to the product through a non-linear activation function f.

$$y_{jk}(x) = f\left(\sum_{i=1}^{n_H} w_{jk}x_i + w_{j0}\right) \tag{5}$$

In this context, the dot product is taken between the weight matrix W and the input vector x. The bias term W_0 can be incorporated into the non-linear function [6]. The Figure 3 represents the architecture of a typical feed-forward network.

2) **Activation Function:** Each neuron in a dense layer typically applies an activation function (such as ReLU, Sigmoid, or Tanh) to introduce nonlinearity into the model. This nonlinearity is crucial for learning complex patterns in the data.

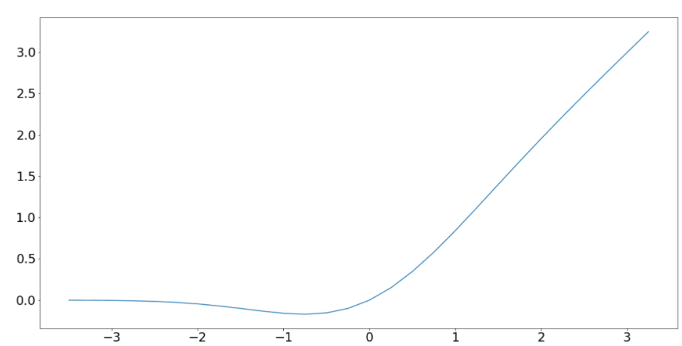


Fig. 4: Gelu activation function [7]

The project focuses on using the **GELU** (Gaussian Error Linear Unit). The key advantage of GELU lies in its smoothness and differentiability across the entire real line. The Gaussian Error Linear Unit (GELU) is—

$$GELU_{\tanh}(x) = 0.5x \left(1 + \tanh\left(\sqrt{\frac{2}{\pi}}\left(x + 0.044715x^3\right)\right)\right) \quad (6)$$

There is also a simpler sigmoid-based approximation:

$$GELU_{\text{sigmoid}}(x) = x \times \text{sigmoid}(1.702 \times x) \quad (7)$$

The benefits of the $tanh$ and sigmoid functions are that both the functions and their derivatives are easy to compute [8].

B. Batch-Normalization Layer

Batch normalization subtracts the batch mean and divides by the batch standard deviation to normalize prior activation layer output. This method reduces internal covariate shift to speed up training. It also regularizes by adding noise, reducing the requirement for dropout. High learning rates are possible using batch normalization, accelerating training as gradient descent converges faster [9].

C. Dropout Layer

Dropout is a technique that temporarily removes certain nodes from both input and hidden layers in a neural network, disabling all their connections. Nodes are dropped based on a predefined probability p, effectively creating a smaller network architecture. This process helps prevent over-fitting by encouraging the network to learn more robust features [10].

D. Optimizer and Loss Function

Optimizer and loss functions are essential for compiling a deep learning model.

- **Optimizer:** The optimizer used in this model is Adam (Adaptive Moment Estimation), an efficient algorithm for optimizing gradient descent, especially with large datasets or numerous parameters. The mathematical expression for the Adam optimizer is given below [11].

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1)\left[\frac{\delta L}{\delta w_t}\right] \quad (8)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2)\left[\frac{\delta L}{\delta w_t}\right]^2 \quad (9)$$

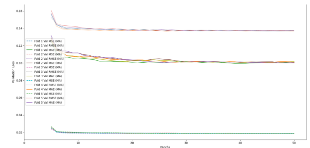
- **Loss Function:** Three loss functions were used in the project.

1) MSE Loss Function
2) RMSE Loss Function
3) Mean Absolute Error

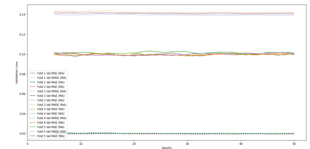
These error functions help to evaluate the dataset efficiently.

V. Result and Discussion

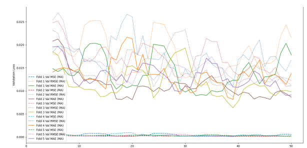
The power flow solution dataset contained voltage magnitude, voltage angle, PV and PQ bus real and reactive power. The ANN model took these characteristics as inputs to estimate voltage magnitude (pu), angle, and power flow at buses and branches.



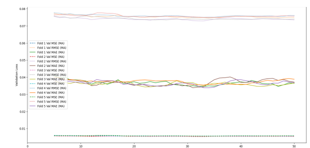
(a) Voltage magnitude validation losses



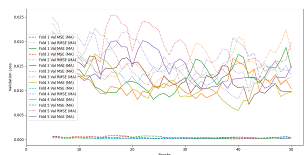
(b) Voltage angle validation losses



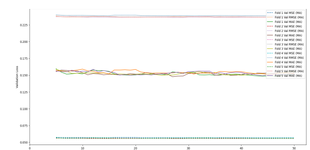
(c) Generation real power validation losses



(d) Generation reactive power validation losses

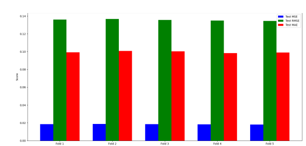


(e) Load real power validation losses

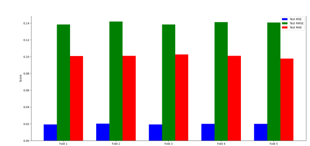


(f) Load reactive power validation losses

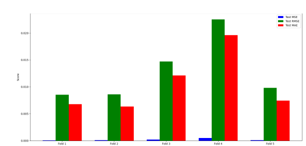
Fig. 5: Different power flow variables validation losses



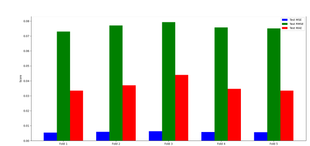
(a) Voltage magnitude test losses



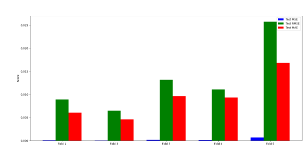
(b) Voltage angle test losses



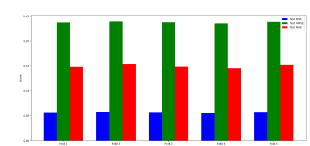
(c) Generation real power test losses



(d) Generation reactive power test losses



(e) Load real power test losses



(f) Load reactive power test losses

Fig. 6: Different power flow variables test losses

TABLE II: Loss Metrics for Different Power Systems Variables

	Fold Number	Train			Validation		
		MSE (loss)	RMSE (loss)	MAE (loss)	MSE (loss)	RMSE (loss)	MAE (loss)
Voltage Magnitude	Fold-1	0.0183	0.1352	0.0981	0.0187	0.1369	0.0981
	Fold-2	0.0184	0.1355	0.0999	0.0189	0.1373	0.0999
	Fold-3	0.0183	0.1354	0.0999	0.0189	0.1374	0.0999
	Fold-4	0.0185	0.1361	0.0998	0.0189	0.1375	0.0998
	Fold-5	0.0183	0.1354	0.0997	0.0188	0.1371	0.0997
	Fold Number	**Train**			**Validation**		
		MSE (loss)	**RMSE (loss)**	**MAE (loss)**	**MSE (loss)**	**RMSE (loss)**	**MAE (loss)**
Voltage Angle	Fold-1	0.0196	0.1399	0.1015	0.0200	0.1414	0.1015
	Fold-2	0.0193	0.1390	0.0994	0.0193	0.1388	0.0994
	Fold-3	0.0198	0.1409	0.1040	0.0198	0.1409	0.1040
	Fold-4	0.0193	0.1390	0.0996	0.0196	0.1400	0.0996
	Fold-5	0.0199	0.1411	0.0981	0.0203	0.1425	0.0981
	Fold Number	**Train**			**Validation**		
		MSE (loss)	**RMSE (loss)**	**MAE (loss)**	**MSE (loss)**	**RMSE (loss)**	**MAE (loss)**
Generation Real Power	Fold-1	0.0001	0.0085	0.0068	0.0001	0.0086	0.0068
	Fold-2	0.0001	0.0088	0.0065	0.0001	0.0088	0.0065
	Fold-3	0.0002	0.0146	0.0121	0.0002	0.0146	0.0121
	Fold-4	0.0005	0.0224	0.0194	0.0005	0.0224	0.0194
	Fold-5	0.0001	0.0097	0.0074	0.0001	0.0098	0.0074
	Fold Number	**Train**			**Validation**		
		MSE (loss)	**RMSE (loss)**	**MAE (loss)**	**MSE (loss)**	**RMSE (loss)**	**MAE (loss)**
Generation Reactive Power	Fold-1	0.0058	0.0763	0.0354	0.0056	0.0747	0.0354
	Fold-2	0.0057	0.0753	0.0362	0.0055	0.0730	0.0362
	Fold-3	0.0058	0.0759	0.0350	0.0056	0.0750	0.0350
	Fold-4	0.0185	0.1361	0.0998	0.0189	0.1375	0.0998
	Fold-5	0.0057	0.0755	0.0332	0.0056	0.0746	0.0332
	Fold Number	**Train**			**Validation**		
		MSE (loss)	**RMSE (loss)**	**MAE (loss)**	**MSE (loss)**	**RMSE (loss)**	**MAE (loss)**
Load Real Power	Fold-1	0.0001	0.0089	0.0060	0.0001	0.0088	0.0060
	Fold-2	0.0000	0.0065	0.0046	0.0000	0.0064	0.0046
	Fold-3	0.0001	0.0110	0.0092	0.0001	0.0110	0.0092
	Fold-4	0.0001	0.0080	0.0050	0.0001	0.0083	0.0050
	Fold-5	0.0000	0.0042	0.0023	0.0000	0.0043	0.0023
	Fold Number	**Train**			**Validation**		
		MSE (loss)	**RMSE (loss)**	**MAE (loss)**	**MSE (loss)**	**RMSE (loss)**	**MAE (loss)**
Load Reactive Power	Fold-1	0.0016	0.0402	0.0256	0.0015	0.0388	0.0256
	Fold-2	0.0018	0.0422	0.0292	0.0016	0.0392	0.0292
	Fold-3	0.0015	0.0387	0.0252	0.0014	0.0374	0.0252
	Fold-4	0.0016	0.0402	0.0260	0.0015	0.0388	0.0260
	Fold-5	0.0014	0.0372	0.0249	0.0013	0.0356	0.0249

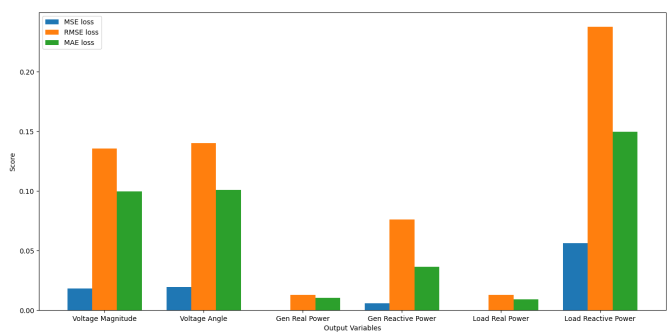


Fig. 7: Power flow output variable test losses

The Figure 5 and Figure 6 showcase three different losses (MSE, RMSE, MAE) for the validation and test datasets in evaluating the output variables in power flow analysis. The errors are minimal for each output variable measurement. Furthermore, Figure 6 depicts the training and validation losses for each fold to demonstrate the model's performance.

The Figure 7 illustrates all the evaluation results for the power flow output variables. The model effectively measures these variables using the given input. The minimal error demonstrates the effectiveness of the model. Beside, the Table III highlights a comparative analysis on measuring voltage and angle using Newton-Raphson and ANN model. [12] The low convergence values demonstrate that the proposed ANN model can efficiently perform power flow analysis, comparable to the Newton-Raphson method and other traditional approaches.

TABLE III: Result Analysis

Bus_No	Analytical Solution		ANN Solution		Convergence	
	Voltage	Angle	Voltage	Angle	Voltage	Angle
1	.9904	-2.1295	.9858	-1.7688	.0046	.3607
2	0.9801	-2.799	0.9862	-2.7497	-0.0061	-0.0493
3	1	0	0.9993	.0249	0.0007	-.0249
4	0.9798	-2.15708	0.9972	-2.3784	-0.0174	.2213
Bus_No	**Analytical Solution**		**ANN Solution**		**Convergence**	
	Voltage	Angle	Voltage	Angle	Voltage	Angle
1	0.9693	-0.5778	0.9955	-1.0451	-.0262	.4673
2	1.0000	2.0092	1.0053	.8847	-0.0053	1.1245
3	1	0	0.9992	.0923	0.0008	-0.0923
4	0.9778	-2.7714	0.9984	-2.9326	-.0206	.1612
Bus_No	**Analytical Solution**		**ANN Solution**		**Convergence**	
	Voltage	Angle	Voltage	Angle	Voltage	Angle
1	0.9696	-3.033	0.9836	-3.0578	-0.014	.0248
2	1.0117	0.0175	1.0177	.9552	-0.006	-.9377
3	1	0	.9992	.0825	0.0008	-0.0825
4	0.9937	-1.1103	.9958	-1.7866	-0.0021	.6763
Bus_No	**Analytical Solution**		**ANN Solution**		**Convergence**	
	Voltage	Angle	Voltage	Angle	Voltage	Angle
1	1	0	0.9991	.0765	0.0009	-0.0765
2	0.9724	-2.4305	0.9865	-2.0735	-0.0141	-.357
3	1	0	0.9992	.0491	0.0008	-.0491
4	1.0165	-1.15441	.9975	-2.1891	0.019	1.034

Table IV compares the Newton-Raphson method and ANN-based techniques for power flow analysis, using randomly selected data to demonstrate the ANN model's efficiency. While deviations in power flow results are slightly higher, these can be reduced with a larger dataset and more train-

TABLE IV: Comparison Between Newton-Raphson and ANN based Power Flow Analysis

Newton-Raphson based Power Flow Analysis						
Line		**Power at Bus and Line Flow**			**Line loss**	
From	To	MW	MVAR	MVA	MW	MVAR
1		239.000	274.429	363.912		
	2	32.901	179.663	182.651	13.989	14.105
	3	31.427	179.663	182.391	11.265	14.105
2		152.652	-8.311	152.878		
	1	-18.912	-165.557	166.634	13.989	14.105
	4	93.013	-28.927	97.407	8.688	4.426
3		184.335	-65.181	195.519		
	1	-20.162	-165.557	166.781	11.265	14.105
	4	95.513	-28.927	99.797	13.721	4.426
4		-220.000	-89.797	237.620		
	2	-84.325	33.353	90.681	8.688	4.426
	3	-81.791	33.353	88.330	13.721	4.426
ANN based Power Flow Analysis						
Line		**Power at Bus and Line Flow**			**Line loss**	
From	To	MW	MVAR	MVA	MW	MVAR
1		235.562	281.801	367.289		
	2	42.624	182.222	187.141	14.118	14.735
	3	39.441	182.499	186.713	11.392	14.726
2		181.229	7.511	181.384		
	1	-28.507	-167.487	169.896	14.118	14.735
	4	91.512	-14.649	92.677	8.530	3.987
3		174.132	12.066	174.550		
	1	-28.050	-167.773	170.102	11.392	14.726
	4	95.591	-14.831	96.735	13.548	4.131
4		-216.791	-90.932	235.089		
	2	-82.982	18.636	85.049	8.530	3.987
	3	-82.043	18.962	84.206	13.548	4.131
Average Deviations		4.536	24.827	2.096	0.0673	0.0347

able parameters. Despite the current differences, the ANN model shows potential for outperforming the Newton-Raphson method due to its focus on pattern recognition.

VI. Conclusion

We have investigated the use of deep learning for load flow analysis and compared its performance with the traditional Newton-Raphson method. The IEEE-4 bus system was employed as a test case to assess the accuracy and computational efficiency of both approaches. The Newton-Raphson method, implemented in MATLAB, provided a reliable and accurate solution for load flow analysis, serving as a benchmark for comparison. Given the increasing complexity of modern power systems, machine learning techniques such as deep learning offer a promising alternative for more adaptive load flow solutions. The deep learning model developed in this study was able to almost accurately predict load flow results similar to those obtained through the Newton-Raphson method, demonstrating its potential to model complex system behavior. The deep learning approach offers advantages in terms of adaptability, making it well-suited for real-time applications [4]. This study demonstrates that deep learning can be effectively integrated with traditional load flow methods to enhance system analysis in modern power grids. Future work could focus on extending the model to larger power systems, incorporating diverse load profiles, and exploring other advanced machine learning algorithms to further improve prediction accuracy and computational performance.

References

[1] A. Garces, "A quadratic approximation for the optimal power flow in power distribution systems," *Electric Power Systems Research*, vol. 130, pp. 222–229, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S037877961500276X

[2] A. Marini, S. Mortazavi, L. Piegari, and M.-S. Ghazizadeh, "An efficient graph-based power flow algorithm for electrical distribution systems with a comprehensive modeling of distributed generations," *Electric Power Systems Research*, vol. 170, pp. 229–243, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378779618304231

[3] P. V. Hentenryck, "Machine learning for optimal power flows," 2021. [Online]. Available: https://pubsonline.informs.org/doi/10.1287/educ.2021.0234

[4] A. Abdelaziz, "A neural network-based model for load flow solution," *Scientific Bulletin, Faculty of Engineering, Ain Shams University*, 06 2002.

[5] T. Kulworawanichpong, "Simplified newton–raphson power-flow solution method," *International Journal of Electrical Power Energy Systems*, vol. 32, no. 6, pp. 551–558, 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142061509001847

[6] D. Unzueta, "Fully connected layer vs. convolutional layer: Explained," https://builtin.com/machine-learning/fully-connected-layer#:~:text=Fully%20Connected%20Layer%3F-,A%20fully%20connected%20layer%20refers%20to%20a%20neural%20network%20in,output%20of%20the%20output%20vector., accessed: 2024-09-23.

[7] S. Goel, "Gelu activation," https://medium.com/@shauryagoel/gelu-gaussian-error-linear-unit-4ec59fb2e47c, accessed: 2024-09-23.

[8] M. S. Thomas Carr, "Gelu explained," https://www.baeldung.com/cs/gelu-activation-function, accessed: 2024-09-23.

[9] S. Saxena. (2021) Introduction to batch normalization. Accessed: 2024-09-23. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/03/introduction-to-batch-normalization/

[10] H. Yadav, "Dropout in neural networks," https://towardsdatascience.com/dropout-in-neural-networks-47a162d621d9, accessed: 2024-09-23.

[11] (2024) What is adam optimizer? Accessed: 2024-09-23. [Online]. Available: https://www.geeksforgeeks.org/adam-optimizer/

[12] S. Efe, "Power flow analysis by artificial neural network," *International Journal of Energy and Power Engineering*, vol. 2, p. 204, 11 2013.