www.rericjournal.ait.ac.th

Efficient Fault Classification in Distributed Generation Systems using M-KNN and Grid Search Techniques

Kamlesh S. Bisht*, [1] Nafees Ahamad*, and Saurabh Awasthi*

Abstract *–To meet the growing demand for electricity and ensure a sustainable future, there is a significant shift towards distributed generation (DG), including non-renewable energy sources. The stability and reliable operation of DGs, which involve non-uniform power generation, present challenging problems. Proper fault diagnosis and mitigation are crucial in these systems. Consequently, reliable fault identification and mitigation are essential to ensure the trustworthiness and functionality of DGs. Established mathematical models for fault identification, location, and system isolation can be time-consuming and inaccurate. With advancements in machine learning (ML) and artificial intelligence (AI), these technologies have found applications in distributed generation systems (DGS). Therefore, this article investigates the use of Mutual K-Nearest Neighbors (M-KNN) for fault identification. This study considers a 100 km grid-connected distributed generation system comprising two distributed generators (DGs), simulated using MATLAB® to obtain data for ten different faults at various locations spaced at 2 km intervals. Subsequently, M-KNN in Python is employed for fault classification to accurately determine the nature of faults. To enhance the robustness of the model, a grid search approach with and without cross-validation (CV) is utilized. The achieved training and testing accuracies approach 99%, surpassing the performance of Basic KNN (B-KNN) models.*

Keywords – Power System, Distributed Generation System (DGs), Renewable Energy, Fault Identification, K-Nearest Neighbors (KNN), Mutual KNN (M-KNN), Machine Learning (ML), Artificial Intelligence (AI).

1. INTRODUCTION

Distributed generation (DG) refers to the process of generating energy close to the point of consumption, typically at kilowatt (kW) or, in some cases, megawatt (MW) levels. It encompasses various forms of small-scale generation, including renewable energy sources. Integrating power from these dispersed generation stations with the conventional grid offers an economical solution. This integration achieves significantly reduced transmission expenses, minimal distribution outlay, enhanced efficiency, improved reliability, substantial reduction in maintenance costs, and promotes an environmentally friendly energy culture [1].

However, as the number of DG units increases, the probability of faults also rises. This necessitates the detection of fault locations with minimal error and, ideally, the prediction of fault types within the same or shorter time frame compared to existing methods. Thus, the incorporation of distributed generation in electricity generation and distribution has recently become a major focus for electrical engineers worldwide [2].

The implementation of these systems requires the detection of both the type and location of faults within the integrated distribution system. According to [3], [4], faults in DGs can result from various factors such as overloading, operational errors, short circuits, underground vegetation interference, and neglected maintenance due to aging. The most common faults in overhead distribution systems include line-to-ground (LG), double line (LL), double line-to-ground (LLG), and three-phase (LLL) faults. As per research published by [5], line faults are the most prevalent in distribution systems. Contemporary mathematical models with relays have been used for a long time to estimate and isolate faulty zones in power distribution systems. However, with advanced scientific tools like Artificial Intelligence (AI) and Machine Learning (ML), new and optimal methods are being researched globally.

For example, a method proposed by [6] using ML reduces the amount of dataset required for training and claims higher accuracy in detecting and classifying fault events. However, this method is limited to detecting and classifying LL and LG faults with a maximum accuracy of 96.66% and 91.66%, respectively. Another study [7] used discrete wavelet transform along with ML as a classifier to locate faults in both balanced and unbalanced radial systems, claiming 100% accuracy except for LLLG and LLL faults, which again limits the method's applicability. Therefore, a reliable tool for identifying all types of faults is required for grid systems.

Srinivasan et al. [8] employed ML-based Link Fault Identification and Localization (ML-LFIL), which analyzes data gathered from normal traffic flow, considering end-to-end delay, packet loss, and aggregate flow rate, achieving up to 97% performance. Many researchers [9] - [22] have worked globally on fault identification and classification using AI and ML. Their work and results are summarized in Table I.

In general, accuracy and computational complexity are key issues in fault classification and localization. Table I shows that various ML algorithms and networks are being researched for fault identification and precise

*Department of Electrical and Electronics and Communication Engineering.

[1] Corresponding author;
Tel: +9407389203.
E-mail: ksbisht99@gmail.com

fault classification. The literature review reveals that KNN is the most preferred ML algorithm by researchers. The simplicity of the algorithm and its better performance in specific fields, after modifications to the basic KNN, make it a popular research area. Various KNN variants have emerged, each with improved reliability in fault location, classification, or response time. Research is ongoing to develop a single algorithm that achieves all these parameters.

Therefore, in this article, Mutual K-NN (M-KNN) has been tested and validated for fault classification in a complex distributed power system. The system considered is a 100 km grid-connected distributed generation system comprising two distributed generators (DG) with loads, with ten faults considered at various locations, including data at every 2 km of the distributed line. The comparative results reveal that the proposed K-NN algorithm achieves a very high accuracy of more than 99%. This paper is organized as follows: Section II introduces Mutual KNN (M-KNN), Section III discusses the methodology of implementing the M-KNN algorithm, Section IV presents the results, and finally, Section V concludes the paper and highlights future research prospects.

Table 1: Overview of fault detection and identification.

Ref	Author	Method	Results	Pros	Cons
[9]	Omaer et al.	Extreme Learning Machine	No of Faults Considered: 10 Accuracy: 99.09%	Fast and Less computational complexity compared to the regular ANN model	Used with maximum of two generator not on DG
[10]	Tong et al.	CNN Model	No of faults considered: 5 Accuracy: 98.24%	High accuracy with fewer faults	Limited fault scenarios considered
[11]	Guo et al.	CNN Model with Hilbert-Huang Transform filter	No of faults considered: 10 Accuracy: 99.92%	High accuracy with advanced filtering techniques	Potentially high computational cost
[12]	Lee et al.	CNN Model	No of faults considered: 4 Accuracy: 98.00%	Reliable fault detection with limited faults	Limited fault scenarios considered
[13]	Tawfik, and Morcos	ANN Model Using Prony method	No of faults considered: 10 Accuracy: 98%	Effective for specific fault scenarios	Accuracy is limited to specific types of faults
[14]	Abdullah	ANN Model with Discrete Wavelet Transform	No of faults considered: 10 Accuracy: 98.00%	Enhanced fault detection using DWT	Computationally intensive
[15]	Fahim et al.	Spare filter	No of faults considered: 11 Accuracy: 99.72%	High accuracy with spare filter	Potentially high computational cost
[16]	Vyas et al.	Chebyshev Neural Network With Undecimated DWT	No of faults considered: 10 Accuracy: 98.69%	Improved accuracy with advanced methods	Computationally intensive
[17]	Mukerjee et al.	Probabilistic Neural Network	No of faults considered: 10 Accuracy: 99.33%	High accuracy with probabilistic approach	Complex implementation
[18]	Y. Zhang et al.	Deep Belief Network	No of faults considered: 9 Accuracy: 97.08%	Effective for complex systems	Lower accuracy compared to other methods
[19]	Majid et al.	KNN	No of faults considered: 10 Accuracy: 98.06%	Simple and effective	Lower accuracy compared to advanced methods
[20]	Dasgupta et al.	KNN with cross – correlation	No of faults considered: 10 Accuracy: 99.67%	High accuracy with cross-correlation	Potentially higher computational cost
[21]	Awasthi et. al	KNN	No of faults considered: 10 Accuracy: 98.98%	Effective for DG with renewable energy	Less Accuracy especially in testing data
[22]	Awasthi et. al	Shallow artificial neural network	No of faults considered: 10 Accuracy: 95.31 %	Effective for DG with renewable energy	Very less accuracy

2. FAULT DETECTION LOCATION METHODS

There are two primary methods for fault detection and location in power systems:

a) Mathematical Approach**:** This method involves calculating an approximate model of the power system and triggering various relays to isolate the fault [23]. A disadvantage of this method is that it may isolate healthy sections of the transmission network during fault occurrences. Additionally, there is potential to reduce the response time for detecting and classifying faults.

b) Machine Learning (ML) Approach**:** ML algorithms offer simplicity, quick response, and high reliability, making them a dependable system for fault location and identification. Although significant research has been conducted on conventional mathematical models, the emergence of ML as a potent technological tool presents great opportunities for further study. Various ML algorithms, including Decision Trees, Bagging

Classifiers, Random Forest, Naive Bayes, and Regression methods, have been extensively studied. Among these, K-Nearest Neighbors (K-NN) stands out due to its speed, simplicity, and reliability [24].

2.1 Mutual K-NN Algorithm

The superiority and efficacy of the K-NN-based method have been established by [25], proving it to be a potent tool for refining fault categorization and localization in distribution systems with numerous DGs. The simplicity of the K-NN algorithm, combined with slight tweaks to metric measurement techniques and parameter tuning, has demonstrated better performance in terms of fault location predictability and classification within a competitive timeframe. Research on different ML algorithms, particularly on plant species classification, has shown that the K-NN algorithm performs better than other algorithms [26]. K-NN does not require a training process or any prior information about the statistics of training instances; it can directly classify faults based on the training set [27]. As per [28], the asymptotic sample error of K-NN compared to the Bayes error rate is lesser by twice the value.

The basic K-NN method revolves around two principles:

a) Search Stage**:** A unique and fixed value of K is used during the search stage, but the algorithm is sensitive to the value of K, and its selection significantly affects the results.
b) Decision Stage**:** Classification is based on a voting principle, which ignores next majority elements and disregards their significance.

Various modifications to the basic K-NN method, primarily by altering the distance metric to identify nearest neighbors, have been proposed. The basic K-NN method is based on finding the shortest distance using either the Euclidean or Manhattan distance method between the given samples of the dataset and the query [29]. However, if the distance metric is weighted and calculated using the following formula [30]:

$$W_i = \frac{1}{D(x_i, x_f)^2} \tag{1}$$

Where:

- x_i is vector *i*;
- x_f is fault vector to be classified;
- *D* is the distance;

By using this fact, the method becomes Weighted-KNN. Some scientists have proposed their own metric systems, such as Hassanat metric KNN (H-KNN), where the distance is given by:

$$D_{(hassanat)} = \sum_{1}^{m} (a_i , b_i) \tag{2}$$

K-NN acquires various forms depending on the distance matrices adopted, such as K-Means [31], Generalized Mean Distance KNN (GMD-KNN) [32], Ensemble Approach KNN (E-KNN) [33], Adaptive KNN [34], etc. The performance of basic K-NN depends heavily on the affinity to the nearest neighbor, disregarding that one or more of the nearest neighbors of a data point may be corrupt due to the wrong determination of K in K-NN. Tackling outliers, which are left over or missed even after data pre-processing, is crucial, especially with small training data, to make K-NN one of the most effective ML algorithms. Generally, a large value of K helps reduce noise, but this comes at the computational cost of the algorithm. Also, for uniformly distributed points, determining the appropriate pre-decided value of K is challenging.

Researchers have developed Mutual K-NN (M-KNN) to address these issues, using mutuality among nearest neighbors for query fault X_f to ascertain its label. The M-KNN algorithm searches for mutual nearest neighbors of X_f and concludes a label that is more authentic due to the exclusion of pseudo neighbors or outliers. Mathematically, for a given dataset, M-KNN for a new fault f is given by [34]:

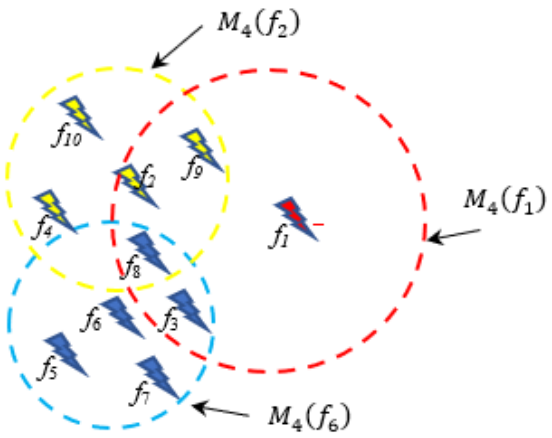
$$M_{k_1,k_2} f = x_i \in D | x_i \in N_{k_i}(x) \wedge x \in N_{k_2}(x_i) \tag{3}$$

where $N_k(x)$ is a set of K Nearest Neighbors (NN) of the query fault. Simply put, the above equation states that faults f_1 and f_2 are mutual neighbors when f_1 belongs to the K_1 NN of another fault f_2, and at the same time, f_2 is one of the K_1 NN of f_1. Figure 1 provides a better visualization for understanding the concept of mutual KNN [32]. It shows that when the value of K is taken as k = 4, the nearest neighbors for f_1 are f_2, f_3, f_8, and f_9. Similarly, for f_2, the nearest neighbors are f_4, f_8, f_9, and f_{10}, and for f_3, the nearest neighbors are f_3, f_5, f_7, and f_8 respectively.

It is evident that f_1, though not an immediate neighbor of f_2, f_3, f_8, and f_9, treats them as nearest neighbors when k = 4. In the simplest terms, for f_1, NN can be f_2 and f_3, but the reverse is not true when seen from the perspective of f_2 and f_3, respectively. Now, if we assign K_1 and K_2 as K for understanding the concept, then equation 3 along with NN become

$$\begin{aligned} M_4(f_1) &= \{f_2, f_3, f_8, f_9\}; \\ M_4(f_2) &= \{f_4, f_8, f_9, f_{10}\}; \\ M_4(f_6) &= \{f_3, f_5, f_7, f_8\}; \end{aligned} \tag{4}$$

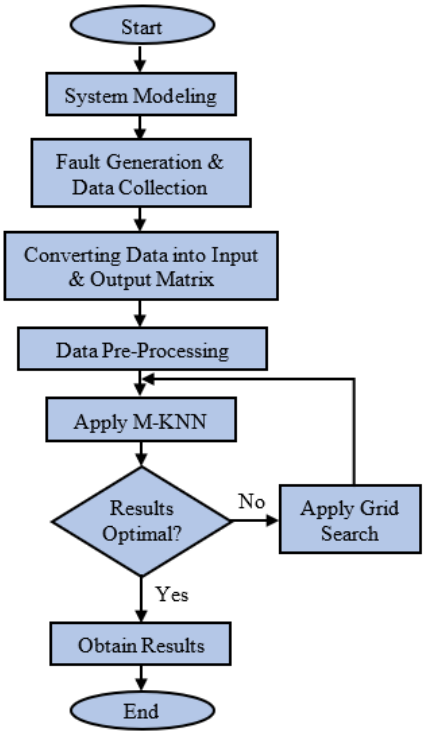
It is quite apparent why f1 should not be discarded since it is acting like an outlier and will adversely affect the outcome. This is addressed by the ML algorithm and is the basic concept of M-KNN for such types of noisy data sets, making M-KNN one of the most robust and easier algorithms for implementation concerning inconsistent and noisy data. Additionally, adopting some pre-processing steps before subjecting the data to the M-KNN machine learning algorithm turns out to be a very effective tool, providing precision within a minimal time frame.



3. PROPOSED METHODOLOGY

The flow chart for the proposed method is given in Figure 2 and describes the entire process of the methodology implemented in the M-KNN algorithm. The process is divided into four main sections:

a) Data Collection**:** Due to the scarcity of ideal data in the real world, simulation is necessary. Fault data from a substation should ideally be collected for all types of faults and at all possible distances, but this is a time-consuming and difficult process, making real-world data acquisition practically impossible. Therefore, simulations are conducted in MATLAB©, and data is collected for all types of faults at equal intervals of 2 km each along a 100 km line. A near-practical 132 kV, 50 Hz, star-grounded system with two distributed generation systems connected via a 100 MVA, 132/66 kV step-down transformer and terminated by two distributed generators of 100 kVA at 3.3 kV is simulated. Fault parameters are observed at equal intervals of 2 km. A 10 kW load dissipates power.



b) Data Pre-Processing: The data collected in the previous step is in raw form, containing many outliers and extreme values that can skew the results and provide incorrect outcomes. Therefore, pre-processing of the collected data is essential before training and testing. The pre-processing process is executed in four stages:

- Refining Data: This step involves eliminating incorrect or noisy data and addressing any missing data.
- Transforming Data: Using various processes such as the selection of specific attributes and normalization, data is made suitable for the mining process.
- Optimization of Data: The processing time of any ML algorithm depends on the volume of data to be processed. Hence, redundant data is eliminated to handle less data for classification. It is vital to optimally select data features and eliminate unnecessary data without compromising the overall effect.
- Integration of Data: This involves combining data obtained from various sources into a single data warehouse with larger storage capacity.

The entire process is depicted diagrammatically in Figure 3.

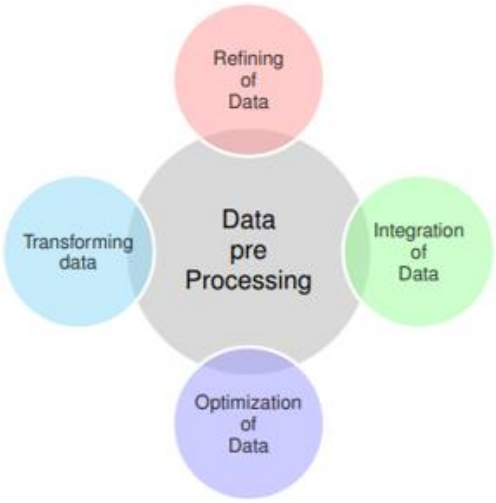


Fig. 1 Data Pre-Processing for M-KNN.

c) ML Algorithm Realization: Once data is processed and converted into an input-output matrix, an ML algorithm for M-KNN is created in Python, and results are tabulated.

d) Applying Grid Search: Grid searching is a tool available in Python to tune the hyper parameters for complex mathematical models to achieve optimal data pre-processing. It involves tuning the optimal combination of hyper parameters at which the algorithm has maximum efficacy using a hit-and-trial method. In our study, better results were achieved after tuning the hyper parameters to $n = 5$ instead of $n = 3$ in the basic KNN, with all other parameters kept the same.

4. RESULTS AND DISCUSSION

The grid-connected 100 km distributed power system was simulated in MATLAB©, generating different faults at 2 km intervals. Voltage and current data from both the distributed generators and the grid were recorded under all fault conditions, resulting in a dataset of size 19x490 [35]. This data was then pre-processed to identify and address any outliers, although none were observed due to the nature of the simulation data. The input data consisted of grid voltage phases A, B, and C, as well as voltage and current measurements for DG1 and DG2 across their respective phases. This resulted in an input matrix of fault voltage and current with dimensions 18x490, which was used to train both the Mutual KNN (M-KNN) model and the basic KNN (B-KNN) model for comparison.

To ensure the robustness of the system, the model underwent cross-validation (CV), and a grid search (GS) was employed to determine the optimal parameters for the M-KNN model. Models were created with and without grid search, resulting in different configurations, which are tabulated in Table II.

Table 2. Different KNN models with their hyper-parameters and accuracy.

Algorithm/Parameters	Hyper Parameters	Training Accuracy	Test Accuracy	No of Wrongly Classified Faults
B-KNN CV = 5, Without GS	Algorithm: auto, K = 5, weights: uniform, metric: minkowski	0.8879	0.9694	3
B-KNN CV = 0, Without GS	Algorithm: auto, K = 5, weights: uniform, metric: minkowski	0.9694	0.9694	3
Basic KNN, CV = 5, With GS	Algorithm: auto, K = 1, weights: uniform, metric: minkowski	0.9617	0.9898	1
B-KNNCV = 0, With GS	Algorithm: auto, K = 1, weights: uniform, metric: minkowski	0.9617	0.9898	1
M-KNN, CV = 5, Without GS	Algorithm: brute, K = 3, weights: uniform, metric: cosine	0.9465	0.9898	1
M-KNN, CV = 0, Without GS	Algorithm: brute, K = 3, weights: uniform, metric: cosine	0.9898	0.9898	1
M-KNN, CV = 5, With GS	Algorithm: brute, K = 5, weights: distance, metric: cosine	0.9694	0.9898	1
M-KNN, CV = 0, With GS	Algorithm: brute, K = 5, weights: distance, metric: cosine	0.9694	0.9898	1

Table 3: Classification report of the best performing B-KNN and M-KNN models.

Fault Type	B-KNN CV = 0 with GS				Mutual KNN CV = 0, without GS			
	Precision	Recall	f1-score	Support	Precision	Recall	f1-score	Support
ABC	1	0.92	0.96	13	1	1	1	13
A to G	1	1	1	11	1	1	1	11
B to G	1	1	1	10	1	1	1	10
C to G	1	1	1	11	1	1	1	11
AB	1	1	1	11	0.92	1	0.96	11
AC	0.91	1	0.95	10	1	1	1	10
BC	1	1	1	7	1	1	1	7
AB to G	1	1	1	7	1	0.86	0.92	7
AC to G	1	1	1	9	1	1	1	9
BC to G	1	1	1	9	1	1	1	9

In Table II, various hyper parameters, training and test accuracies, and the number of wrongly classified faults are documented. It is evident that the training and test accuracies of B-KNN with CV = 0 & 5 without grid search are comparatively lower than those of other algorithms, as illustrated in Figure 4. This indicates poor performance of these two methods. Conversely, B-KNN with CV = 0 & 5 with grid search demonstrates improved performance.

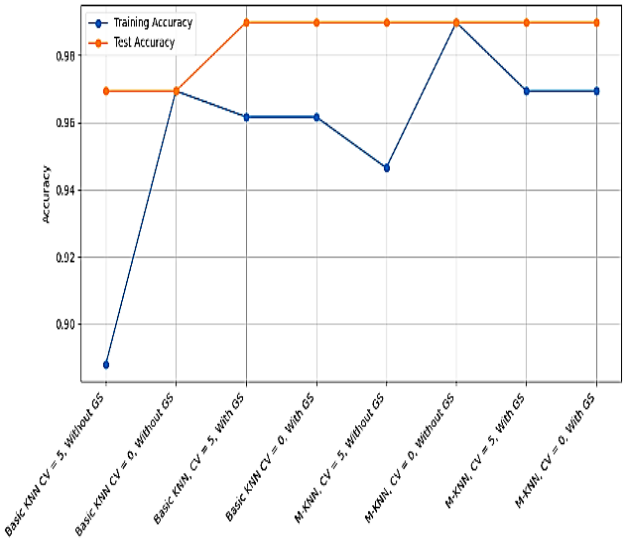


Fig. 2. Training and test accuracy of different KNN models.

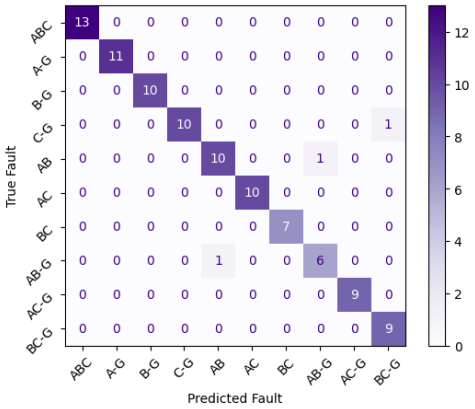


Fig. 3. Confusion matrix of worst performing B-KNN.

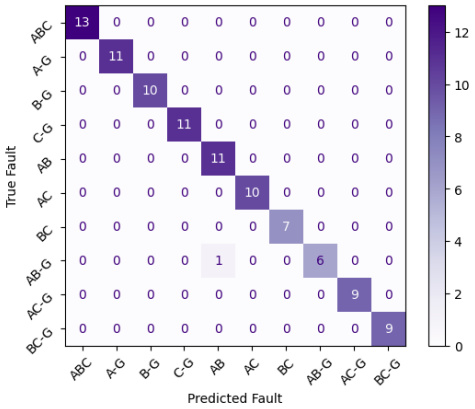


Fig. 4. Confusion matrix of worst performing M-KNN.

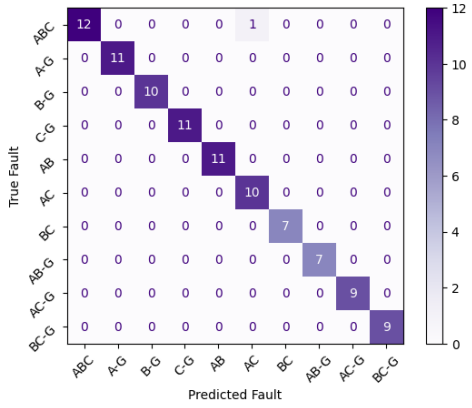


Fig. 5. Confusion matrix of best performing B-KNN.

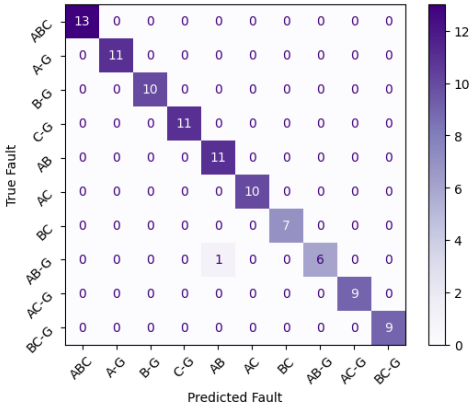


Fig. 6. Confusion matrix of best performing M-KNN.

Furthermore, from Table II and Figure 4, it is evident that M-KNN outperforms B-KNN. Specifically, the results of M-KNN with CV = 0 and without grid search exhibit the highest performance, achieving training and test accuracies of 99% with only one wrongly classified fault.

Additionally, a comparison in terms of different parameters like precision, recall, f1-score, and support for various faults is made between the best performing M-KNN and the best performing B-KNN, as shown in Table III. From this table, it can be observed that M-KNN outperforms B-KNN for ABC faults, while B-KNN demonstrates better performance for AB and AB to G faults. Both models perform equally well for single phase-to-ground faults (A to G, B to G, C to G), AC and BC faults, and AC to G and BC to G faults.

A comparison of the worst-performing KNN and M-KNN models is shown in Figure 5 and Figure 6, respectively, in the form of confusion matrices. It is evident that even in its worst performance, M-KNN has only one wrong classification compared to three in B-KNN. Figure 7 and Figure 8 show the confusion matrices for the best-performing KNN models. In B-KNN, a line-to-line (AC) fault is misclassified as a three-line (ABC) fault, whereas in M-KNN, a line-to-line (AB) fault is misclassified as a line-to-line-to-ground (AB-G) fault, which is a closer approximation.

5. CONCLUSION

This article investigated fault classification techniques using machine learning in a 100 km grid-connected distributed power system with two distributed generators. The system was simulated in MATLAB, generating various faults, and capturing corresponding fault voltages and currents. These parameters served as the input matrix for training, testing, and validating Mutual K-nearest neighbours (M-KNN) models in Python. Different configurations, including cross-validation (CV) with and without grid search (GS), were explored and compared with basic K-Nearest Neighbors (B-KNN).

The following conclusions are drawn:

- M-KNN consistently outperforms B-KNN across all scenarios.
- B-KNN without grid search yielded suboptimal results, whereas M-KNN without grid search demonstrated better performance.
- Both KNN models exhibited improved results when grid search was employed.
- M-KNN with CV = 0 and without grid search achieved the highest training and test accuracy of nearly 99%, which remained consistent even with CV = 5.
- Notably, one of the cross-validation scores for M-KNN with CV = 5 and grid search reached 100%

In conclusion, M-KNN emerges as the preferred method for fault classification in power systems, given its superior performance. Further research can explore its application to real-world power system data and investigate additional machine learning techniques to enhance results.

REFERENCES

[1] P. Chiradeja and R. Ramakumar, "An approach to quantify the technical benefits of distributed generation," IEEE Transactions on Energy Conversion, vol. 19, no. 4, pp. 764-773, Dec. 2004.

[2] J. Rocabert, A. Luna, F. Blaabjerg, and P. Rodríguez, "Control of power converters in AC microgrids," IEEE Transactions on Power Electronics, vol. 27, no. 11, pp. 4734-4749, Nov. 2012.

[3] C. Okoli, B. Anyaka, C. Nwokedi, and V. Anya, "Application of Composite Method for Determining Fault Location on Electrical Power Distribution Lines," Journal of Electrical and Computer Engineering, vol. 2020, pp. 1-9, Jan. 2020.

[4] T. Covington, T. Stankiewicz, R. Anderson, L. Wright, and B. Cockerham, "A simple method for determining fault location on distribution lines," in Proc. 71st Annu. Georgia Tech Protective Relaying Conf., May 2017.

[5] M. Kezunovic, "Smart fault location for smart grids," IEEE Transactions on Smart Grid, vol. 2, no. 1, pp. 11-22, Mar. 2011.

[6] J. Hosseinzadeh, F. Masoodzadeh, and E. Roshandel, "Fault detection and classification in smart grids using augmented K-NN algorithm," SN Applied Sciences, vol. 1, no. 12, p. 1627, Dec. 2019.

[7] M. Hojabri, S. Nowak, and A. Papaemmanouil, "ML-based intermittent fault detection, classification, and branch identification in a distribution network," Energies, vol. 16, no. 16, p. 6023, Aug. 2023.

[8] S. M. Srinivasan, T. Truong-Huu, and M. Gurusamy, "Machine learning-based link fault identification and localization in complex networks," IEEE Internet of Things Journal, vol. 6, no. 4, pp. 6556-6566, Aug. 2019.

[9] M. O. F. Goni et al., "Fast and accurate fault detection and classification in transmission lines using extreme learning machine," e-Prime - Advances in Electrical Engineering, Electronics and Energy, vol. 3, p. 100107, Feb. 2023.

[10] H. Tong et al., "Detection and classification of transmission line transient faults based on graph convolutional neural network," CSEE Journal of Power and Energy Systems, vol. 7, no. 3, pp. 456-471, Sep. 2021.

[11] M. F. Guo, N. C. Yang, and W. F. Chen, "Deep-learning-based fault classification using Hilbert–Huang transform and convolutional neural network in power distribution systems," IEEE Sensors Journal, vol. 19, no. 16, pp. 6905-6913, Aug. 2019.

[12] H. J. Lee et al., "Convolutional neural network-based false battery data detection and classification for battery energy storage systems," IEEE Transactions on Energy Conversion, vol. 36, no. 4, pp. 3108-3117, Dec. 2021.

[13] M. M. Tawfik and M. M. Morcos, "ANN-based techniques for estimating fault location on transmission lines using Prony method," IEEE Transactions on Power Delivery, vol. 16, no. 2, pp. 219-224, Apr. 2001.

[14] A. Abdullah, "Ultrafast transmission line fault detection using a DWT-based ANN," IEEE Transactions on Industry Applications, vol. 54, no. 2, pp. 1182-1193, Mar. 2017.

[15] S. R. Fahim, S. K. Sarker, S. M. Muyeen, S. K. Das, and I. Kamwa, "A deep learning based intelligent approach in detection and classification of transmission line faults," International Journal of

Electrical Power & Energy Systems, vol. 133, p. 107102, Nov. 2021.
[16] B. Vyas, B. Das, and R. P. Maheshwari, "An improved scheme for identifying fault zone in a series compensated transmission line using undecimated wavelet transform and Chebyshev Neural Network," International Journal of Electrical Power & Energy Systems, vol. 63, pp. 760-768, Dec. 2014.
[17] A. Mukherjee, K. Chatterjee, P. K. Kundu, and A. Das, "Probabilistic neural network-aided fast classification of transmission line faults using differencing of current signal," Journal of The Institution of Engineers (India): Series B, pp. 1-14, Jan. 2021.
[18] Y. Zhang et al., "A cable fault recognition method based on a deep belief network," Computers & Electrical Engineering, vol. 71, pp. 452-464, Oct. 2018.
[19] A. A. Majd, H. Samet, and T. Ghanbari, "k-NN based fault detection and classification methods for power transmission systems," Protection and Control of Modern Power Systems, vol. 2, no. 4, pp. 1-11, Dec. 2017.
[20] A. Dasgupta, S. Debnath, and A. Das, "Transmission line fault detection and classification using cross-correlation and k-nearest neighbor," International Journal of Knowledge-based and Intelligent Engineering Systems, vol. 19, no. 3, pp. 183-189, Sep. 2015.
[21] S. Awasthi, G. Singh, and N. Ahamad, "Classifying electrical faults in a distribution system using k-nearest neighbor (knn) model in presence of multiple distributed generators," Journal of The Institution of Engineers (India): Series B, pp. 1-14, Feb. 2024.
[22] S. Awasthi, G. Singh, and N. Ahamad, "Identification of type of a fault in distribution system using shallow neural network with distributed generation," Energy Engineering, vol. 120, no. 4, pp. 811-829, Dec. 2023.
[23] G. Saleem, M. Akhtar, N. Ahmed, and W. S. Qureshi, "Automated analysis of visual leaf shape features for plant classification," Computers and Electronics in Agriculture, vol. 157, pp. 270-280, Nov. 2019.
[24] A. Yadav and A. Swetapadma, "Fault analysis in three phase transmission lines using k-nearest neighbor algorithm," in Proc. 2014 Int. Conf. Adv. Electron. Comput. Commun., pp. 1-5, Oct. 2014.
[25] L. Jun-li, "Research and application on KNN method based on cluster before classification," in Proc. Int. Conf. Mach. Learn. Cybern., Jul. 2008.
[26] W. Cherif, "Optimization of K-NN algorithm by clustering and reliability coefficients: application to breast-cancer diagnosis," Procedia Computer Science, vol. 127, pp. 293-299, Oct. 2018.
[27] J. Dhar, A. Shukla, M. Kumar, and P. Gupta, "A weighted mutual k-nearest neighbour for classification mining," arXiv preprint arXiv:2005.08640, May 2020.
[28] M. Alkasassbeh, G. A. Altarawneh, and A. Hassanat, "On enhancing the performance of nearest neighbour classifiers using hassanat distance metric," arXiv preprint arXiv:1501.00687, Jan. 2015.
[29] Cristianleo, "The math and code behind k-means clustering," Medium, Jan. 2024.
[30] J. Gou et al., "A generalized mean distance-based k-nearest neighbor classifier," Expert Systems with Applications, vol. 115, pp. 356-372, Jan. 2019.
[31] A. B. Hassanat, M. A. Abbadi, G. A. Altarawneh, and A. A. Alhasanat, "Solving the problem of the K parameter in the KNN classifier using an ensemble learning approach," arXiv preprint arXiv:1409.0919, Sep. 2014.
[32] L. Baoli, Q. Qin, and Y. Shiwen, "An adaptive k-nearest neighbor text categorization strategy," ACM Transactions on Asian Language Information Processing (TALIP), vol. 3, no. 4, pp. 215-226, Dec. 2004.
[33] Y. Gao et al., "Efficient mutual nearest neighbor query processing for moving object trajectories," Information Sciences, vol. 180, no. 11, pp. 2176-2195, Jun. 2010.
[34] H. Liu and S. Zhang, "Noisy data elimination using mutual k-nearest neighbor for classification mining," *Journal of Systems and Software*, vol. 85, no. 5, pp. 1067-1074, May 2012.
[35] S. Awasthi, G. Singh, and N. Ahamad, "Identification of type & location of a fault in a distributed generation system," *International Journal of Computing and Digital Systems*, vol. 14, no. 1, pp. 1-1, 2023.