Detection and Classification of Transmission Line Faults Using Convolutional Neural Networks and Vision Transformer

ARTICLE INFO

Keywords:
Transmission line fault detection
Deep learning
Convolutional neural networks (CNN)
Vision transformers
GASF images
Fault classification
Smart grid reliability
Transfer learning
Power system protection
Phasor data analysis

ABSTRACT

Transmission lines are essential to power systems, necessitating the implementation of robust fault detection systems to maintain stability and operational efficiency, especially when contemporary infrastructure escalates demand. This paper presents an innovative method utilizing sophisticated deep learning techniques, such as Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs), to effectively identify and categorize transmission line faults. Phasor data for voltage and current across several fault scenarios is produced using Simulink and MATLAB, thereafter converted into two-dimensional Gramian Angular Summation Field (GASF) images to encapsulate spatial patterns for accurate fault identification. The amalgamation of transfer learning and fine-tuning markedly elevates the efficacy of pre-trained CNN models, while the inclusion of ViTs further augments both detection velocity and precision, rendering the methodology appropriate for real-time applications. Furthermore, a consolidated prediction model obviates the necessity for explicit phasor recognition, hence streamlining implementation in real-world applications. The results indicate an effective, scalable, and prompt method for fault detection and classification, providing a significant enhancement to the resilience and dependability of contemporary power grids.

1. Introduction

The transmission line network is one of the most important elements in the transportation of electricity from a generation source to the distribution system. They occupy a significant portion of the overall grid network area. Harsh weather conditions often make them vulnerable to faults. Therefore, detecting and classifying the faults in transmission lines is crucial for grid stability. In an electrical network, relays and circuit breakers collaborate to isolate and address faults. If fault detection and classification are not accurate and efficient, it can lead to damage to the connected power equipment. [1] An effective fault detection system ensures a reliable, fast, and secure relay operation.

Transmission line faults can lead to significant economic losses due to various factors. Power outages caused by transmission line faults is very common. A study by the Electric Power Research Institute (EPRI) estimated that power outages in the United States cost businesses between $104 billion to $164 billion annually in lost productivity and sales. [2] In developing countries, issues with transmission lines can make existing problems with infrastructure and economic stability worse. For instance, frequent power outages due to these problems can slow down industrial growth and economic productivity. A high level of accuracy in fault detection is demanded to reduce the effect of the abovementioned issues resulting from mistakenly detected or undetected faults.

Faults in a power system are often beyond human control and occur when conductors come into contact with each other or the ground. These faults include single line-to-ground faults (AG, BG, CG), line-to-line faults (AB, BC, CA), double line-to-ground faults (ABG, BCG, CAG), and three-phase faults (ABC, ABCG). Single line-to-ground, line-to-line, and double line-to-ground faults are considered unbalanced, while three-phase faults are balanced. High fault currents, frequently caused by short circuits, can result in overheating and mechanical stress on power system equipment. [3]

Pattern recognition techniques are effective in distinguishing between faulty and healthy electrical power systems. Numerous machine learning and signal processing methods have been developed in prior research to ensure fast and precise fault identification. These techniques are useful in finding the exact phase in a three-phase power system that is affected by a fault. Because of this capability, deep learning approaches are especially successful in fault detection and classification than the traditional methods. [4] Traditional methods of fault detection and classification mainly rely on complex algorithms, which may not always provide accurate results. The analyzing ability of images by deep learning models- Convolutional Neural Networks (CNNs) and Computer Vision Transformers (ViTs) has increased in recent years. These technologies excel in fault detection and classification. Both CNNs and ViTs offer a promising way to improve fault detection in transmission lines. CNNs are good at extracting patterns in images, while ViTs excel at capturing patterns in sequences. By using both of these technologies together, we can make fault detection systems more accurate, faster, and reliable.[5]

The proposed approach first uses a CNN model on the generated image dataset. This CNN model is developed from scratch. Later on, a pre-trained Keras CNN model is used to train on our dataset for better accuracy. This model uses transfer learning and fine-tuning to adapt for the specific purpose of fault detection and classification. Finally, the Hugging Face Vision Transformer model is used, which is mainly used for natural language processing (NLP), but it also has a growing collection of tools for computer vision tasks, such as image classification. Like the previous model, ViT is also a pre-trained model that is fine-tuned on a new, often smaller, dataset for a specific task.

ORCID(s):

The main contributions of this research are as follows:

- Creating a large dataset of voltage and current data of all phases that is suitable for deep learning models such as CNNs, pre-trained models, and Vision Transformers (ViT).
- Designing a novel CNN architecture to improve performance and boost evaluation accuracy.
- Using transfer learning and fine-tuning with pre-trained CNN models for comparative analysis.
- Applying Computer Vision Transformers (ViT) to enhance fault detection and classification capabilities.
- Developing a 'Single Prediction Model' that is capable of detecting faults without prior knowledge of voltage, current, or phasor data.

The rest of the paper is organized as follows. Section 2 reviews the literature on fault diagnosis in electrical power systems, classical techniques, recent advances in fault detection, and image encoding of time series data. Section 3 outlines the methodology, starting with the modeling of the transmission line system and the generation of time series data. It discusses the image encoding process for fault scenarios and explores the impact of varying parameters on the generated images. Section 4 details the design of various CNN models for fault detection and classification, including a single prediction model, and describes the simulation and experimental methods used. Section 5 presents the proposed pre-trained Vision Transformer (ViT) architecture, explaining its components such as patch embedding, transformer encoder, and classification head. Finally, Section **??** discusses the results and findings, focusing on the performance analysis of the CNN and ViT models on different datasets and their effectiveness in single fault prediction scenarios. The paper is concluded in Section 7.

2. Literature Review

Transmission line fault detection techniques can be divided into two main categories based on the number of terminals used for sensing and data acquisition: single-ended and end-to-end methods. Each category can be further divided depending on the data measurement type and the data analysis algorithm. The classification is described below: [6]

- Signal processing-based techniques mainly use three-phase current inputs. These techniques address the problem of fault detection and the prediction of system abnormalities. These methods also include various methods such as wavelet transform,[7] Fourier-Taylor transform[8] and power spectral density index.[9]
- Phasor-based methods work with phasor measurement units.[10] These techniques are good at detecting short circuit faults in both balanced and unbalanced systems.
- Machine learning-based methods include Neural Networks (NNs),[11] Support Vector Machines (SVMs),[12] decision trees,[13] summation wavelet and summation-Gaussian extremal learning machines,[14] Generalized Regression Neural Networks (GRNNs),[15] Feedforward Reural Networks (FNNs),[16] and Convolutional Neural Networks (CNNs). [17] [18] [19] These methods excel in detecting abnormalities and distinguishing between faulty and healthy systems.
- Traveling wave (TW)-based methods works in a way to detect and locate faults by analyzing the arrival time of transient waves produced by faults. These techniques are known for their speed and precision, necessitating measurement devices capable of high sampling rates. [20]

Some fault detection and classification methods have already achieved the state-of-the-art stage. Some examples are traditional circuit theory [21] and fuzzy logic methods [22]. These methods are device-based and are costly to implement.

Najafzadeh et al. (2024) proposed a fault detection, classification, and localization approach for smart grids using optimized machine learning algorithms. The method combines fuzzy logic and adaptive neural fuzzy networks with decision trees and random forests, achieving 98.1% and 100% accuracy, respectively, for fault classification. [23] Deep learning methods are cost-effective and reliable. In their paper, Alhanaf et al. (2023) presented a deep learning-based approach for fault detection, classification, and location in smart grids using Artificial Neural Networks (ANNs) and one-dimensional Convolutional Neural Networks (1D-CNNs). The models achieved high accuracy rates, with 99.99% for detecting faulty lines, 99.75% and 99.99% for fault classification for ANN and 1D-CNN, respectively.[24] This approach simplifies the process by eliminating pre-processing steps, improving efficiency for real-time fault diagnosis. Jamil et al. (2015) also worked with ANNs based on a feedforward neural network with a backpropagation algorithm. Their model successfully identifies faults using three-phase currents and voltages as inputs. The network achieved high accuracy, with a correlation coefficient of 0.99982 for fault detection and 78.1% accuracy in fault classification. [1] Zhang et al. (2019) used Convolutional Neural Networks (CNNs) which was applied to motor vibration signals. The CNN model automatically extracts features from time-frequency spectrum images. [25] The approach achieved high accuracy in fault classification, surpassing traditional machine learning techniques.

Image representation of 1D signals captures more complex patterns. Using images instead of raw time series data for fault detection enhances feature extraction. Additionally, image-based deep learning models are more robust to noise and variations, making them highly effective for real-world fault detection scenarios. [26] Several methods are available for encoding time series data as images. Gramian Angular Fields (GAFs) transform multivariate time series into images. [27] Markov Transition Fields (MTFs) capture data point transitions where Wavelet Scalograms visualize time

and frequency components.[28] Distance and Recurrence Plots analyze record similarities. [29] Gramian Angular Field (GAF) is better than the other image encoding techniques because in GAF the temporal dependencies are preserved by transforming data into a polar coordinate system. This method also enhances feature representation and allows the use of powerful image processing techniques, such as Convolutional Neural Networks (CNNs). Additionally, GAFs provide visual interpretability, making it easier to identify patterns and trends, and they are robust to noise, leading to more reliable analysis. [28]

CNNs are particularly effective in detection and classification application because of their multi-layered structure. Different layers capture complex features and preserve the spatial relationships of data, by considering pixel adjacency rather than analyzing them individually. [30] Their capacity to process large datasets and identify complex patterns makes CNNs suitable for applications such as fault detection and classification in power systems. Furthermore, CNNs are capable of identifying and optimizing features from raw data, thus improving overall accuracy and reliability. Similarly, pre-trained CNN models provide a significant advantage in power system fault detection and classification because of the prior training by large-scale datasets. These models can be fine-tuned for domain-specific tasks, such as identifying and classifying. By using this pre-training, the network avoids the need to learn feature extraction from scratch, leading to more efficient training and often improved accuracy. Furthermore, pre-trained models can mitigate issues like overfitting and enhance the model's generalization capabilities by utilizing well-established feature. Pre-trained models like EfficientNet, VGG16, ResNet, and Inception ResNet have been successfully applied to power system fault identification tasks and provided good results. [17]

While CNNs are good at extracting features from one-dimensional time-series data and two-dimensional images, they struggle to capture long-range dependencies. Increasing the number of convolutional layers does not completely resolve this issue when identifying patterns that needs a deeper contextual comprehension. [31] These limitations led to the proposal of transformer models, which have found significant applications in natural language processing and image recognition. The transformer methodology has also been applied to the fault diagnosis of mechanical equipment. However, transformers often struggle with low computational efficiency and high memory consumption. [32] To address these issues, the Vision Transformer (ViT) was introduced. It removes the decoder block from the original transformer model and specifically focuses on vision processing. Additionally, it uses parallel learning mechanisms, which enable the ViT to capture global spatiotemporal information effectively. Building on the advantages of ViT, a one-dimensional ViT architecture integrated with multiscale convolution fusion has been proposed. This architecture is designed to extract fault features across multiple temporal scales, thereby achieving high detection accuracy on bearing fault datasets. [33]

3. Modeling the System and Data Generation

In this paper a 221kV,50 Hz transmission line of 200 km is considered to model the 3-phase power system network. In the generation side, a swing type generator is connected while the other side is connected to a delta-wye transformer which is further connected to a distribution side load of 31kV. The network is modeled in the Simulink environment utilizing various built-in blocks provided by the software, such as the Three-Phase Source, Three-Phase V-I Measurement, Distributed Parameter Line (representing the transmission line), Three-Phase Transformer, Three-Phase Fault, Three-Phase Series RLC Load, Scopes, and additional blocks to export data to the workspace for further analysis. Later on, with the help of MATLAB, different scenarios were simulated to collect data. The network is simulated for all types of faults and for different system parameters like – source impedance, line impedances, fault impedance, fault distances etc. The collected data of raw voltage and current signals is in time series form. Then the time series data is converted to image. To train a deep learning model effectively, features must first be extracted from the image.

3.1. Transmission Line Network Modeling

Figure 1 shows a simplified model of a three-phase power system designed to analyze transmission line behavior under fault conditions. The key components are:

220 kV, 50 Hz transmission line: The transmission line is implemented by two 3-phases distributed parameter line model. This represents a transmission line with a length of 200 km .

Generating Unit: A swing-type generator unit is located at one end of the transmission line.

Delta-wye transformer: This transformer steps down the voltage from 220 kV to 33 kV for connection to the distribution network. The transformer also connects to a distribution side load.

Distribution load: It represents the demand for electricity at the end of the line, modeled as a constant load at 33 kV.

Three-phase V-I measurement units: Two three-phase V-I measurement units are positioned at each end of the transmission line – one connected to the generator and the other to the load. These units measure both voltage and current for all three phases of the system. The outputs of these units can be displayed on a scope or transferred to a software environment like MATLAB for further analysis.

Three-phase fault: A fault model is introduced at a specific location between the two distributed parameter line models. This allows for simulating various fault scenarios and analyzing their impact on the system.

This system model provides a controlled environment to study the behavior of the transmission line under different fault conditions, including variations in fault type, location, and resistance.

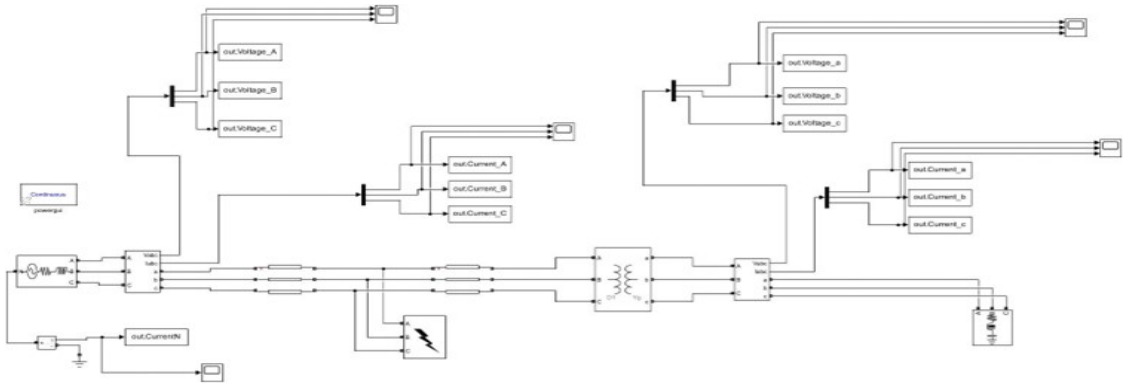


Figure 1: Transmission line electrical network modeling

Table 1
Varying the value of system parameters

System	Parameter	Value
Source	Resistance (Ω)	0.5–4
	Inductance (H)	0.008–0.0115
Fault	Resistance (Ω)	0.5–110
	Distance (km)	1–199
Trans. Line (before fault location)	Pos. seq. res. (Ω)	0.01273–0.0527
	Zero seq. resistance (Ω)	0.3864–0.5864
	Pos. seq. inductance (H)	0.0005337–0.0009337
	Zero seq. inductance (H)	0.0041264–0.0091
	Pos. seq. capacitance (F)	12.74e-9–62.74e-9
	Zero seq. capacitance (F)	7.751e-9–12.751e-9
Trans. Line (after fault location): same as before		

3.2. Generating Time Series Data

Different parameters of the components mentioned in the above segment are varied to collect data for different fault scenarios. The voltage and current time series data are obtained from the scope with the help of MATLAB workplace Some parameters of the network were kept the same. They are:-
Phase to phase voltage of source = 220 kV
Base Voltage of source = 220 kV
Base Power of source = 60 MVA

There is a possibility for a total of 11 types of transmission line fault:- A-g fault, B-g fault, C-g fault, AB-g fault, BC-g fault, CA-g fault, AB fault, BC fault, CA fault, ABC-g fault, and ABC fault. We have simulated all types of faults for data generation.

The following process generates data by simulating, in detail, the system model for different fault scenarios:
Data Preprocessing and Feature Extraction: The first step preprocesses the raw data-for example, p.Current A.da-with an optional wavelet denoising aiming at removing noise. In this way, the data are normalized within the interval of 0 to 1 in order to unify scaling. Then, for every sample that composes the data, the script calculates the angle theta and computes a radius value corresponding to its position or ranking in sequence.

Constructing Time Series Data: This is the crucial step done after repeatedly circulating through the runs of simulation. During each run, it will collect the relevant part of the preprocessed data of current and voltage associated with each phase. These parts should later be organized as time series variables such as Current A ts and Voltage C ts to go ahead with further techniques beyond Simulink.

3.3. Image Encoding of Time Series

The model outlined in this paper takes the voltage and current signals of the three phases as input. GAF algorithm can transform these one-dimensional signal into two-dimensional image. If the time series data is presented in a polar coordinate system rather than the coordinate system, then it becomes the GAF representation. Each element of the gramian-matrix is cosine of the sum of the angles, which can be obtained from GAF representation. Each current or voltage information contains 3400 data samples.
For a time series data $X = \{x_1, x_2, ..., x_i, ..., x_N\}$, containing N observations. For our data total data samples, N=3400. Here the data, X is normalized first. For making all values of X is ranged between -1 to 1 or [-1,1] the expression is given below

$$\tilde{x}_i^{-1} = \frac{(x_i - \max(X)) + (x_i - \min(X))}{\max(X) - \min(X)} \tag{1}$$

For making all values of X range between 0 to 1 or [0,1] the expression is given below

$$\tilde{x}_i^{0} = \frac{x_i - \min(X)}{\max(X) - \min(X)}. \tag{2}$$

Next, to convert original one-dimensional time series data X from the cartesian coordinate system to the polar coordinate system the following expression is used

$$\phi_i = \arccos(\tilde{x}_i) \qquad -1 \le \tilde{x}_i \le 1, \tilde{x}_i \in \tilde{X}_i$$
$$r_i = \frac{i}{N}$$

Polar representation of time series is shown here. It is not necessary for conversion to GAF but understanding

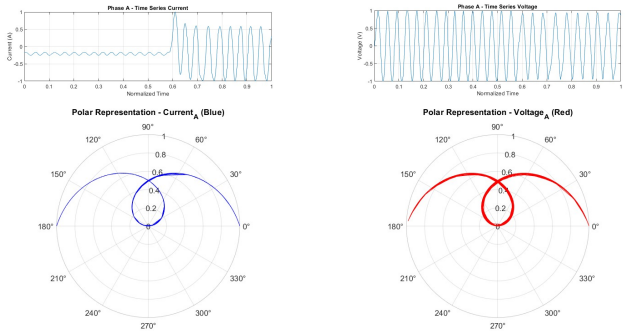


Figure 2: Time series and polar representation of Phase-A current and Phase-A voltage signal during 'a-g' fault at source resistance=0.5Ω and source inductance= 0.08mH

it is important to understand GAF. When representing the normalized observation x_i in a polar coordinate system, we can use its inverse cosine $\arccos(x_i)$ as the angle ϕ_i. The time label $\frac{i}{N}$ serves as the radius. This conversion process leads to different angular ranges depending on the normalization scheme used. For data normalized to the range [-1, 1], the cosine function maps values to the angle range $[0, \pi]$. On the other hand, for data normalized to the range [0, 1], the cosine function maps values to the angle range $[0, \frac{\pi}{2}]$. For our code we normalized the data between -1 to 1 or [-1,1] so the angle range is $[0, \pi]$.

This method utilizes the polar coordinate system to provide a new point of view for understanding time series data. It translates the data from its original amplitude-based changes to variations in angle as time progresses. By calculating trigonometric function relationships between data points, this method identifies the time correlation among them from the perspective of angle. This approach is particularly good at pointing out cyclical or repeating patterns within the time series. The formula of GAF can be written as:

$$GASF = \begin{pmatrix} \cos(\phi_1+\phi_1) & \cos(\phi_1+\phi_2) & \cdot & \cdot & \cos(\phi_1+\phi_N) \\ \cos(\phi_2+\phi_1) & \cos(\phi_2+\phi_2) & \cdot & \cdot & \cos(\phi_2+\phi_N) \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cos(\phi_N+\phi_1) & \cos(\phi_N+\phi_2) & \cdot & \cdot & \cos(\phi_N+\phi_N) \end{pmatrix} \quad (3)$$

$$= \tilde{T}'_{va} \cdot \tilde{T}_{va} - \sqrt{I - \tilde{T}'^2_{va}} \cdot \sqrt{I - \tilde{T}^2_{va}}$$

Here, I is a unit row vector [1,1, ...,1]. After converting the original time series into the polar coordinate system, each data point along the time series can be treated as a point within a 1-dimensional metric space. Equation (3) can be written as :

$$GASF = x.y - \sqrt{I - x^2}.\sqrt{1 - y^2} \quad (4)$$

So GAF can be written as:

$$GAF = \begin{pmatrix} \langle \tilde{x}_1, \tilde{x}_1 \rangle & \langle \tilde{x}_1, \tilde{x}_2 \rangle & \cdot & \cdot & \langle \tilde{x}_1, \tilde{x}_N \rangle \\ \langle \tilde{x}_2, \tilde{x}_1 \rangle & \langle \tilde{x}_2, \tilde{x}_2 \rangle & \cdot & \cdot & \langle \tilde{x}_2, \tilde{x}_N \rangle \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \langle \tilde{x}_N, \tilde{x}_1 \rangle & \langle \tilde{x}_N, \tilde{x}_2 \rangle & \cdot & \cdot & \langle \tilde{x}_N, \tilde{x}_N \rangle \end{pmatrix} \quad (5)$$

The GAF preserves temporal order. Analyzing it diagonally (from the top-left of the matrix to the bottom-right) reveals how data point relationships change over time. Here, the size of the matrix is n × n, where n is the size of the time series data.

Figure 3 shows the algorithm flow chart is given for the computation of GAF

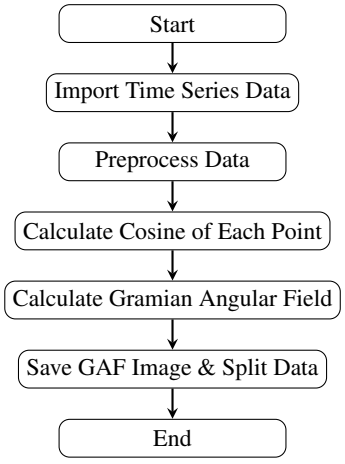


Figure 3: Algorithm flowchart for computation of GAF

Vortmann et all portrayed the following visualization on for GAF algorithm in Figure 4

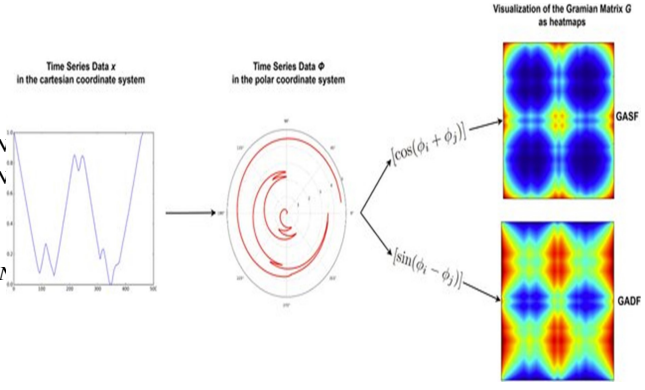


Figure 4: GAF algorithm for image encoding of time series data[34]

Both GASF and GADF images function in the same way. For the sake of simplicity, GASF is chosen to be used.

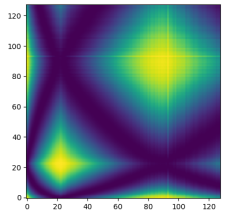
3.4. Effect of Varying Different Parameters on Images

Previously the source impedance, fault resistance, line impedance, and fault distance were varied, and the dataset

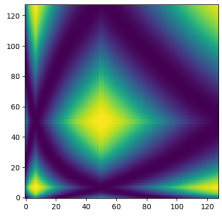
was generated. These variations lead to distinct and discernible effects on the generated images. The effect on images due to the variation of parameters is discussed here.

3.4.1. Effect of Fault Distance

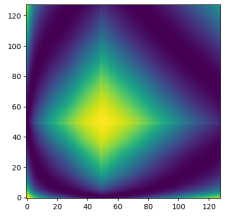
In a real-world scenario, a fault can happen anywhere in the transmission line. Here our transmission line is modeled as 200 km long. From the system, we can see two transmission line models: one is before the 'fault model' and the other is after that. Here both of the line lengths is varied in Simulink. In the specific scenario where the other parameters are kept constant, the fault distance is set to 1km from the generating unit when the first line is 1 km long and the subsequent line is 199km. Similarly, for a fault distance of 199km, the first line is 199km and the second line is 1km. This is how the fault distance ranges from 1km to 199km.
Fault distance affects voltage and current signals. Voltage tends to decrease near the fault and increase farther away. Conversely, current increases near the fault and decreases with distance due to less current flow. This relationship is important for detecting faults and locating them. Figure 5 shows the effect of fault distances on current and voltage images of phasor A.



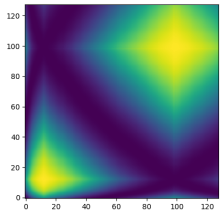
(a) Current-A during 'a-g' fault at 1km Fault Distance



(b) Voltage-A during 'a-g' fault at 1km Fault Distance



(c) Current-A during 'a-g' fault at 199km Fault Distance

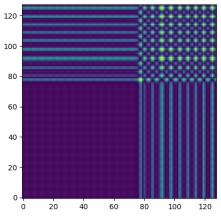


(d) Voltage-A during 'a-g' fault at 199km Fault Distance

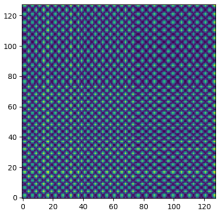
Figure 5: GASF image of Phase-A current and voltage signal during 'a-g' fault for different fault distance

3.4.2. Effect of Source Impedance

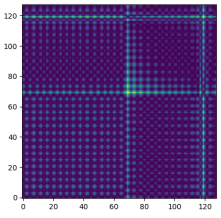
The internal source impedance is varied by changing source resistance and source inductance. Source resistance varies from 0.5 Ω to 4 Ω and source inductance varies from 0.08 mH to 0.011 H. In each step the source resistance is increased by 0.02 Ω and source inductance is increased by 0.55 mH. When analyzing the impact of source impedance on the signals, the fault distance is kept constant at 100 km from the measuring end, and other system parameters also remain constant.
When there is a higher source impedance, the fault current in a power system is lower. Higher source impedance increases the total impedance the fault current must overcome. This results in a lower fault current according to Ohm's Law (I = V/Z). The reduced fault current caused by higher source impedance also leads to a smaller voltage drop across the fault location. This results in a higher voltage remaining at the fault location compared to a scenario with low source impedance. The effect of source impedance on current and voltage images of phasor A is displayed on Figure 6.



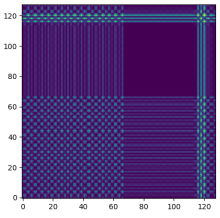
(a) Current-A during 'a-g' fault at source resistance=0.5Ω and source inductance= 0.08mH



(b) Voltage-A during 'a-g' fault at source resistance=0.5Ω and source inductance= 0.08mH



(c) Current-A during 'a-g' fault at source resistance=4Ω and source inductance= 0.0115H



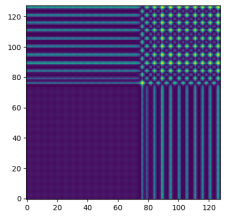
(d) Voltage-A during 'a-g' fault at source resistance=4Ω and source inductance= 0.0115H

Figure 6: GASF image of Phase-A current and voltage signal during 'a-g' fault at for different source impedance

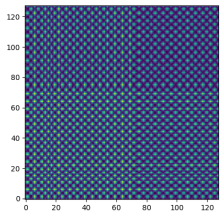
3.4.3. Effect of Fault Resistance

In this scenario, the resistance at the fault location is varied across various cases to examine its impact on the signals. The fault resistance will be varied from a low value of 0.1 Ω to a high value of 100 Ω, in different increment levels. This range enables us to investigate how different fault resistances influence the voltage and current signals at the measuring point.
Fault resistance, caused by imperfect contact or arcing during a fault, significantly influences electrical power systems. It can be analyzed using Ohm's Law (I = V/Z). Fault resistance restricts fault current flow, preventing equipment damage and safety hazards. A higher fault resistance leads to a lower fault current, crucial for protective relaying schemes. Additionally, fault resistance affects fault voltage: solid faults cause a significant voltage drop, safeguarding the system, while high resistance faults can maintain high voltage, posing a risk of electrical shock and equipment

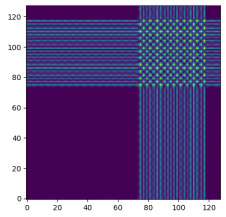
damage. Figure 7 illustrates the effect of fault resistance on current and voltage images of phasor A.



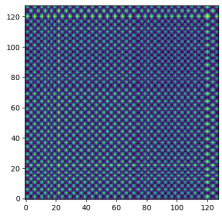
(a) Current-A during 'a-g' fault at fault resistance=2Ω



(b) Voltage-A during 'a-g' fault at fault resistance=2Ω



(c) Current-A during 'a-g' fault at fault resistance=4Ω



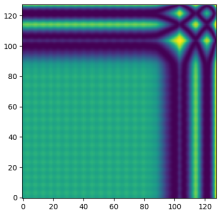
(d) Voltage-A during 'a-g' fault at fault resistance=4Ω

Figure 7: GASF image of Phase-A current and voltage signal during 'a-g' fault for different fault resistance

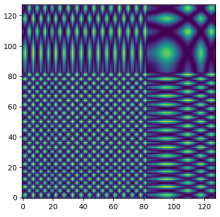
3.4.4. Effect of Transmission Line Impedance

Understanding the impact of transmission line impedance on fault current and fault voltage is important for fault detection. A higher transmission line impedance restricts the flow of current, leading to lower fault currents and reducing the risk of equipment damage and fire. The overall impedance consists of resistance and reactance, where reactance influences the phase angle of the current. Higher impedance also causes a more significant voltage drop, which can help isolate faulted sections and prevent excessive voltage from reaching healthy parts of the system. The specific influence of transmission line impedance on fault voltage depends on the fault type, location, and system configuration.

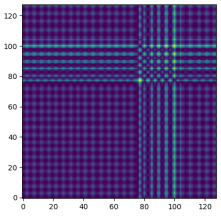
Positive sequence impedance typically results in reduced fault current, and a greater voltage drop at the fault location which minimizes fault damage and facilitates system isolation. On the other hand, zero sequence impedance has a more complicated impact. Lower zero sequence impedance can lead to higher fault currents, especially in ground faults, which can have adverse effects. So here both positive and zero sequence impedances are varied. The images are shown in Figure 8



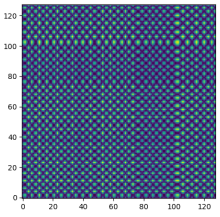
(a) Current-A during 'a-g' fault at lower transmission line impedance



(b) Voltage-A during 'a-g' fault at lower transmission line impedance



(c) Current-A during 'a-g' fault at higher transmission line impedance



(d) Voltage-A during 'a-g' fault at higher transmission line impedance

Figure 8: GASF image of Phase-A current and voltage signal during 'a-g' fault for different transmission line impedance

4. Proposed CNN and Pre-trained CNN Models for Image Classification

Convolutional Neural Networks are deep learning algorithms that take input images and convolve them with filters or kernels to extract features. The term kernels (filters) is used as a matrix in deep learning to extract the most significant features from every subspace of an image. The proposed architecture classifies the GASF image by using the sequential operation of convolution, pooling layer, and other layers and transfers the activation values in one volume to another volume by means of a differentiable function. The first layer consists of convolutional kernels that provide an output using the rectified linear unit (ReLu). In short, the polling layer shortens the amount of computation time and optimizes the parameters to acquire better accuracy. Besides, Batch Normalization was used in this model to improve the training process and increase the model's generalization capability.

4.1. Convolution Layer as Feature Extractor

Convolution layers extract features by applying 2D convolutional operations to the GASF input image. The convolutional operation shown in Figure 9 changes the stride number for the horizontal & vertical direction, filter exploration movement, and dimension size of the input image. This layer alters the filter dimension, size, padding operation, and stride operation. The layer generates the feature map by invoking the following steps:

1. It performs the dot product between the filters and the receptive field of input image.
2. It applies an activation function to the dot product results, introducing non-linearity and enabling the model to learn complex patterns.[35]

Table 2
CNN Layer Structure

Layer Name	Output Shape	Parameters
Conv2D	128,128,32	800
MaxPooling2D	64,64,32	0
Conv2D	64,64,64	18496
MaxPooling2D	32,32,64	0
Conv2D	32,32,128	73856
MaxPooling2D	16,16,128	0
Conv2D	16,16,256	295168
MaxPooling2D	8,8,256	0
Conv2D	8,8,512	1180160
MaxPooling2D	4,4,512	0
Conv2D	4,4,1024	4719616
MaxPooling2D	2,2,1024	0
Flatten	4096	0
Dense	1024	4195328
Dropout	1024	0
BatchNormalization	1024	4096
Dense	512	524800
Dropout	512	0
BatchNormalization	512	2048
Dropout	256	0
BatchNormalization	256	1024
Dense	128	32896
Dropout	128	0
BatchNormalization	128	512
Dense	64	8256
Dropout	64	0
BatchNormalization	64	256
Dense	12	715

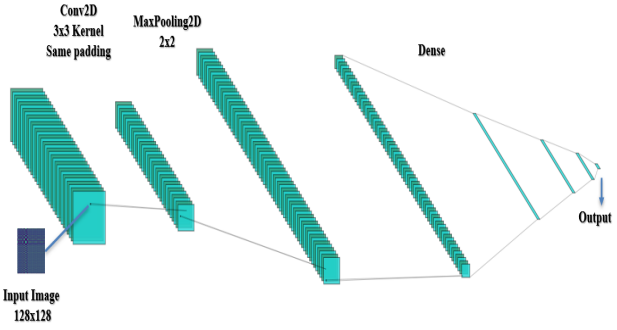


Figure 9: Proposed CNN Architecture as Feature Extractor

In the output, input image becomes from 128×128 pixels to feature-extracted $2 \times 2 \times 1024$ pixels. The stride = 1, padding='same', kernal size =3×3 for the convolution layer. The filter number differs from every forward propagation of the input image.The output of the convolution operation of each layer can be mathematically calculated as follows:

$$x_l^j = f\left(\sum_{i=1,2,\ldots,M} x_i^{l-1} \times k_{ij}^l + b_j^l\right), j = 1, \ldots, N \quad (6)$$

where x_l^j denotes the j_{th} output map of the convolution layer for the filter, k, and the number of inputs, M. Further, x_j^{l-1} represents the i_{th} input feature map of $(l-1)$ layer, b_j^l implies the bias value j_{th} filter, and f denotes the activation function.[36]

4.2. Pooling Layer

To reduce computational time, the pooling layer is connected after the convolution layer, which reduces the size of the feature map with a down-sampling operation without changing the variance of the distinguishing feature scale. In

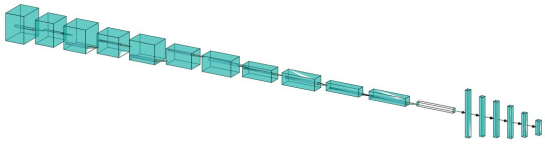


Figure 10: Visualization of Pooling Mechanism

Figure 10 when passing through multiple pooling layers, a large image will be scaled down but keep the features required for recognition. The max-pooling layer stores the maximum value. The stride = 2, pool size = 2×2, pool type ='max' for pooling layer. Calculate the output feature maps of the l_{th} layer as follows:

$$x_j^l = f\left(\beta_j^l . down(x_j^{l-1}) + b_j^l\right), j = 1, \ldots, M \quad (7)$$

Here, x_j^l and x_j^{l-1} are the j_{th} output and input map. f and down(.) denote the activation function and sub-sampling function, respectively. Two bias operations, multiplicative bias and additive bias for the j_{th} filter, are denoted by β_j^l and b_j^l[36].

4.3. Fully Connected Layer

Fully connected layers in a CNN play a vital role in learning high-level representations and making predictions based on the extracted features. A fully connected layer in Figure 11 refers to a neural network in which each neuron applies a linear transformation to the input vector through a weight matrix. After flattening the $2D$ image matrix, the $1D$ vector progresses through various dense layers having a different number of neurons. ReLU function ($f(x) = max(0, x)$) was used in each layer for the activation function.

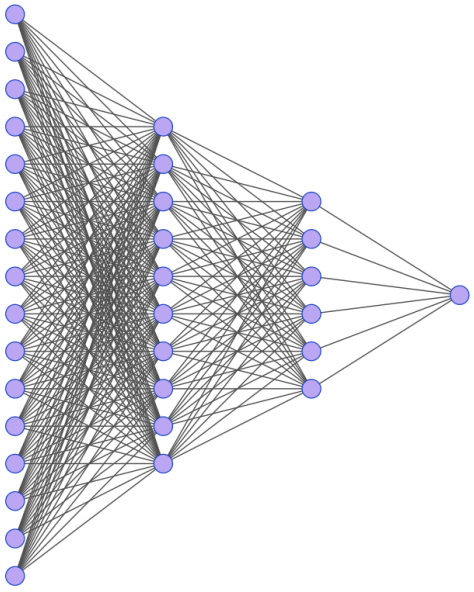


Figure 11: Feed Forward Network

For the input length, M, and total output vector length, N, the resultant output of the l_{th} layer can be calculated as follows:

$$x_j^l = f\left(\sum_{i=1,2,...M} x_i^{l-1} \times w_i^l j + b_j^l\right), j = 1, ..., N \quad (8)$$

where x_j^l denotes the j_{th} output value, x_j^{l-1} is the j_{th} input value, b_j^l and $w_i^l j$ are the bias and weight of the j_{th} output, and f indicates the activation function for the fully connected layer. In general, for the classification problem, the output of the fully connected layer is a probability for every class or category, computed by a SoftMax activation function [36].

4.4. Classifier, Optimizer and Loss Function

The Softmax classifier is suitable for multi-class classification, which outputs the probability for each of the classes. It is an important building block in deep learning networks.For each test input, the softmax classifier creates a K-dimensional vector whose elements sum to 1. Each element of the output vector represents the estimated probability of each class label as follows:

$$P(y_i = m|x_i; W) = a_i = \frac{e^{w_i^T x_i}}{\sum_{j=1}^{K} e^{w_j^T x_j}} \quad (9)$$

where W= $w_1, w_2, w_3, ..., w_k$ are the parameters that are learned by the back-propagation algorithm. The categorical cross-entropy loss function is used as the cost function for the Softmax classifier, and it can be calculated as,

$$J(W) = -\sum_{i=1}^{N}\sum_{j-1}^{K} y_{ij} log\left(\frac{e^{w_j^T x_i}}{\sum_{m=1}^{K} e^{w_m^T x_i}}\right) \quad (10)$$

With N is the number of data points in the training set. Then, the gradient descent method is applied to solve the minimum of the $J(W)$ as,

$$\Delta_w J(W) = \sum_{i=1}^{N} x_i (a_i - y_i)^T \quad (11)$$

Finally, the parameters are updated as,

$$W^{new} = W^{old} - \eta \Delta_w J(W) \quad (12)$$

where, η is the learning rate.[41]

4.5. Proposed Pre-trained CNN Architectures

The main benefits of pre-trained CNN models are that they have prior training knowledge on large prior datasets. This training knowledge helps to evaluate the project datasets efficiently. There are five different pre-trained CNN models that were used on the project datasets. There are pre-trained architectures in Figure 12,

Algorithm 1 Feature Extraction & Fine Tuning Algorithm

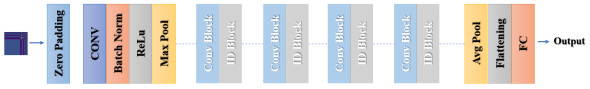
```
input ← GASFimage
preprocessed_image ← input
if Method ← Feature_Extraction then
    Freeze all Layers (Not Trainable).
    Perform training with moderate Lr rate.
    Result ← EvaluationandPrediction
    Output ← Result
else if Method ← Fine_Tuning then
    Unfreeze all Layers (Trainable).
    Perform training with very low Lr rate.
    Result ← EvaluationandPrediction
    Output ← Result
else if Method ← FeatureExtraction + FineTuning then
    Freeze all Layers (Not Trainable).
    Perform training with moderate Lr rate.
    Unfreeze all Layers (Trainable).
    Perform training with very low Lr rate.
    Result ← EvaluationandPrediction
    Output ← Result
else
    Invalid Method
end if
```

4.5.1. Transfer Learning

Transfer learning reduces the requisite computational costs to build models for new problems. By repurposing pre-trained models or pre-trained networks to tackle a different task, users can reduce the amount of model training time, training data, processor units, and other computational resources. In general, there are two different strategies for performing TL in deep CNN: feature extraction and fine-tuning. Both mechanisms are illustrated in Algorithm 1. [42]

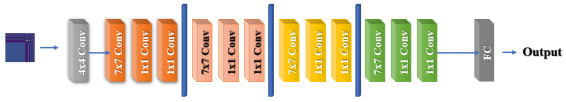
1. **Feature Extractor:** The feature extractor extracts features by performing a convolution operation once

(a) VGG16 architecture [37]



(b) ResNetv100 architecture [38]

(c) EfficientNet architecture [39]



(d) ConvNext architecture [40]

Figure 12: Proposed pre-trained CNN architectures.

in the feed-forward procedure. The advantage of implementing the feature extractor in Transfer Learning is that it greatly reduces the number of epochs and training time due to already extracted features. Instead of training from top to bottom through a large number of convolution layers multiple times, only features are fed into a shallow, fully connected neural network. Thus, all feature extractor procedures spend much less time than other methods.[42]

2. **Fine Tuning:** Compared to feature extraction, fine tuning involves retraining certain parts of the CNN. In some layers, parameters are kept fixed, meaning their gradients are not calculated during back-propagation. Only the parameters in the unfixed layers are updated by the gradient descent algorithm. [42] As a result, applying fine-tuning makes a pre-trained model much faster, more cost-effective, and more compute-efficient than training a model from scratch. However, curve overfitting, task mismatch, catastrophic forgetting, and limited interpretability disadvantages can harm the performance of the pre-trained CNN model.

5. Proposoed Pre-trained ViT Architecture

Vision Transformer was a pioneering innovation in the field of image classification. There are three main components of a vision transformer model.

5.1. Patch Embedding

At first, an input image of shape (height, width, channels) is embedded into a feature vector of shape $(n+1, d)$, following a sequence of transformations. The corresponding equation,

$$z_0 = [x_{class}; x_p^1 E; x_p^2 E; ...; x_p^N E] + E_{pos}, \quad E \in \mathbb{R}^{(p^2.C)\times D}, E_{pos} \in \mathbb{R}^{(N+1)\times D} \tag{13}$$

The image is split into n square patches of shape (p, p, c), where p is a pre-defined parameter. The patches are flattened

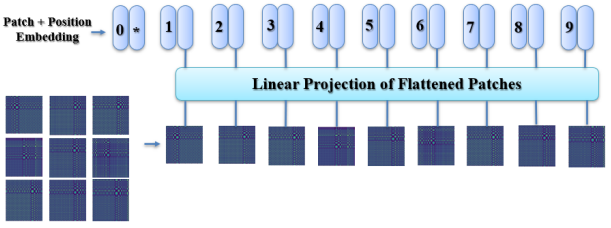


Figure 13: Patch embedding mechanism[43]

into n line vectors of shape $(1, p^{2*}c)$. The flattened patches are multiplied with a trainable embedding tensor of shape $(p^{2*}c, d)$, which learns to linearly project each flat patch to dimension d. The result is n embedded patches of shape $(1, d)$. A learnable [cls] token of shape $(1, d)$ is prepended to the sequence of patch embeddings. A trainable positional embedding tensor, E_{pos}, with the same shape, $(n+1, d)$, is added to the concatenated sequence of projections. This tensor gets 1D positional information for each of the patches. z_0 is the first input to the stacked transformer encoders. Each transformer takes input features represented by an $(n+1, d)$ tensor and produces an output of the same dimension. [43]

5.2. Transformer Encoder

Second, the network acquires knowledge of more abstract features from the embedded patches using a stack of L transformer encoders. The corresponding equations,

$$z_l' = MSA(LN(z_l - 1)) + z_{l-1}, \qquad l = 1...L \tag{14}$$

$$z_l = MLP(LN(z_l')) + z_l', \qquad l = 1...L \tag{15}$$

The encoder component consists of a multi-headed attention (MHA) mechanism and a 2-layer MLP, with layer normalization and residual connections in between. Since the MLP has 2 layers (hidden and output), there will be two weight matrices:

- W_h of shape (d, d_{mlp})

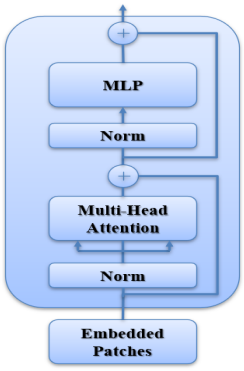


Figure 14: Transformer Encoder[43]

- W_0 of shape (d_{mlp}, d)

Layer normalization helps in reducing the training time and stabilizing the hidden state dynamics. Moreover, residual connections offer alternative paths for gradients in order to solve the problem of vanishing gradients in very deep architectures. Besides, the MHA mechanism follows the corresponding equations,

$$[q, k, v] = zU_{qkv} \qquad Uqkv \in \mathbb{R}^{D\times 3D_h}, \tag{16}$$

$$A = softmax(\frac{qk^T}{\sqrt{D_h}}) \qquad A \in \mathbb{R}^{N\times N}, \tag{17}$$

$$SA(z) = Av \tag{18}$$

$$MSA(z) = [SA_1(z); SA_2(z); ...; SA_k(z)]U_{msa} \qquad U_{msa} \in \mathbb{R}^{k.D_h\times D} \tag{19}$$

The hidden state from the previous encoder is split into K heads, resulting in K feature tensors of shape (n, d_h). Each is multiplied with 3 trainable matrices Q_i, K_i, V_i of shape (d_h, d_h). Q_i, K_i, and V_i represent the projection of the input in 3 sub-spaces. Each line in Q works as a learned projection of the patch, and lines in K as other patches are compared to Q. V and K are taught to express the importance of the features in V to measure the final **attention**. We compute the scaled dot-product attention tensor (A) on each head, normalizing it with the square root of the head's dimension. Self-attention is the product between A and v, which has the shape $(n + 1, d_h)$. The self-attention matrices are concatenated in the second dimension, resulting in a $(n + 1, d)$ tensor, which is then run through a single linear layer, effectively multiplying it with a (d, d) trainable tensor. The importance of this linear layer lies in its ability to learn features as aggregates from all the heads. [43]

5.3. Classification Head

The classification head uses the last representation of the $[cls]$ token. For pre-training, a 2-layer MLP is used; therefore, there are two weight matrices: W_h of shape (d, d_{mlp}) and W_0 of shape (d_{mlp}, d). For fine-tuning, a single linear layer is used; therefore, there is only a single tensor of shape (d, n_cls). In each case, the final output of the network is a vector of shape $(1, n\_cls)$, containing the probabilities associated with each of the n_cls classes. [43]

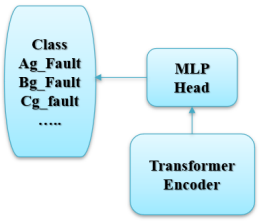


Figure 15: Classification Head[43]

6. Result and Analysis

6.1. Custom and Pretrained CNN Models

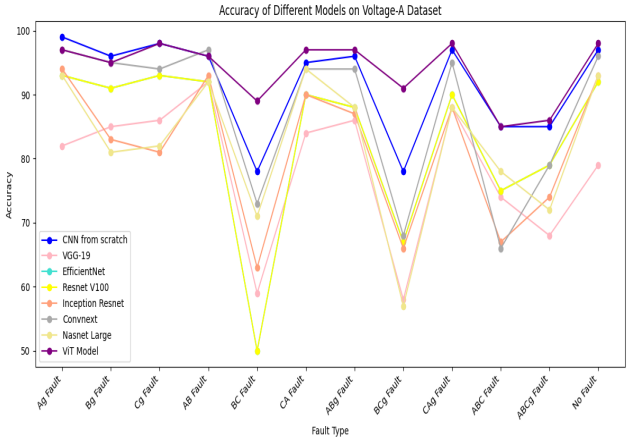


Figure 16: Accuracy of Different Models on Voltage-A Dataset

Both the custom CNN and pretrained CNN models demonstrated strong evaluation results on the voltage phase A dataset. As shown in Table 3, a comparative analysis reveals that the custom CNN achieved an accuracy of 91%, surpassing the other pretrained CNN models. Figure 16 illustrates the performance of the ViT and custom CNN models in detecting various fault types within the Voltage Phase A dataset. However, most models exhibited lower accuracy when detecting BC_Fault and BCg_Fault, likely due to the evaluation being limited to Voltage Phase A data. Since the ViT model produced the most promising results, subsequent evaluations on the Voltage Phase B, C, and Current Phase A, B, and C datasets were conducted using this model.

6.2. ViT Model

The proposed **ViT** model showcases promising results in evaluating other datasets. From Table 4, the performance label of the ViT model was prominent and noteworthy. Across all types of voltage and current phasor datasets, the accuracy

Table 3
Result Analysis of CNN and pretrained CNN Models

Model Performance Analysis											
	Cross Validation (1 of 5)	Cross Validation (1 of 5)	Cross Validation (2 of 5)	Cross Validation (2 of 5)	Cross Validation (3 of 5)	Cross Validation (3 of 5)	Cross Validation (4 of 5)	Cross Validation (4 of 5)	Cross Validation (5 of 5)	Cross Validation (5 of 5)	Test Accuracy (%)
Model Name	Accuracy (%)	Loss	Accuracy (%)	Loss	Accuracy (%)	Loss	Accuracy (%)	Loss	Accuracy (%)	Loss	
CNN	83.7865	0.5871	84.5562	0.594	83.274	0.806	85.428	0.5015	85.421	0.6474	91
ConvNext	80.627	0.4268	82.425	0.438	82.066	0.3908	83.0334	0.3908	81.593	0.427	87
EfficientV2 Large	76.618	0.5989	78.365	0.566	77.0812	0.658	77.995	0.612	81.593	0.664	82
NesNet Large	73.935	0.814	73.833	0.698	73.474	0.922	73.4735	0.711	75.308	0.705	82
ResNet 152V2	76.618	0.5989	78.365	0.566	77.081	0.6578	77.994	0.612	76.812	0.664	82
VGG19	72.909	0.5875	73.935	1.083	75.475	0.5386	69.625	1.0888	75.154	0.563	77
Inception Resnet	73.884	0.582	72.037	0.621	74.962	0.566	74.089	0.577	74.076	0.569	81

Table 4
Result Analysis of ViT Model

ViT Model Performance												
Dataset Type	Training Set				Validation Set				Test Set			
	Epoch	Loss	Samples per sec	Steps per sec	Accuracy (%)	Loss	Samples per sec	Steps per sec	Accuracy (%)	Loss	Samples per sec	Steps per sec
Voltage Phase A	100	0.17923	14.346	1.435	85.233	0.76224	50.845	8.483	93.945	.422	52.45	8.742
Voltage Phase B	100	0.1442	13.432	1.344	80.99	0.80815	34.456	5.752	90.598	0.3487	25.521	4.254
Voltage Phase C	100	0.0695	11.014	1.102	83.904	0.74144	35.982	6.007	93.078	0.2549	35.864	5.986
Current Phase A	100	0.0799	12.184	1.218	88.031	0.5511	42.634	7.113	94.485	0.215	32.276	5.385
Current Phase B	100	0.136	12.324	1.233	87.309	0.49495	42.157	7.037	93.058	0.2248	42.655	6.943
Current Phase C	100	0.0868	11.785	1.178	87.254	0.50948	29.497	4.921	92.911	0.2355	22.455	3.744

label was consistent and retained at 90-94%. Additionally, the accuracy of both voltage and current phasor datasets maintained the accuracy label above 95% across all types of fault classes. Figure 17 and Figure 18. However, because of the less variation nature of ABC and ABCg faults, the model struggles to evaluate any test datasets regarding dataset types (voltage and current). As a result, in Figure 17 and Figure 18 the accuracy label of these two faults is lower than usual.

6.3. Single Prediction Model

The creation of the GASF dataset and the evaluation of the phasor datasets in the ViT model added novelty to the research work. However, the development of the Single Prediction Model significantly elevated the level of novelty. The primary objective of this model is to identify fault types in lines that exhibit any phasor dependencies within the datasets. Given voltage and current phase A, B, and C data, the model can predict faults without needing to know the specific phase information. This feature makes the model versatile and highly applicable across diverse power system scenarios. In actual power system networks, an immediate fault diagnosis is required. Thus, immediate fault detection is necessary. If extracting specific phase data is not feasible, the model can still detect and classify faults using any available voltage and current phase data. This increases the detection efficiency and reduces the detection time as well, ensuring seamless operation and minimal downtime in critical systems.

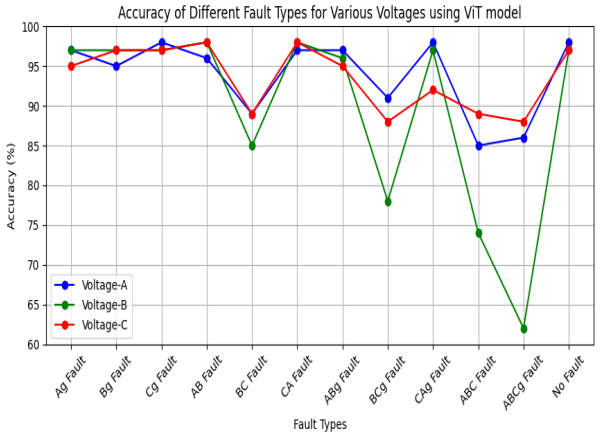


Figure 17: Accuracy vs Fault Types on voltage pahse A, B and C datasets

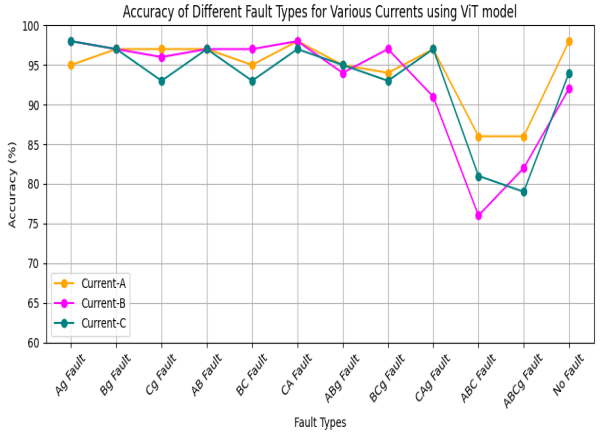


Figure 18: Accuracy of Different Fault Types for Various Currents using ViT model

The dataset for this model was created by collecting datasets for both current and voltage across phases A, B, and C. After tuning the ViT model to suit the dataset, it became capable of detecting faults without relying on phasor information. The Figure 19 showcases a good performance overview of this Single Prediction Model. The Figure 19, also highlights a low accuracy level in specific ABC and ABCg faults. The single prediction model also failed to distinguish between these two faults. Approximately 20, 329 GASF images were evaluated with the model (Figure 20), which achieved around 94% accuracy.

7. Conclusion

This paper presents a standard approach for detecting and classifying transmission line faults, utilizing a custom CNN architecture and then pre-trained CNN models combined with transfer learning techniques. By transforming voltage and current phasor time-series data into GASF images, the proposed method significantly improves fault classification accuracy. Comparative studies show that

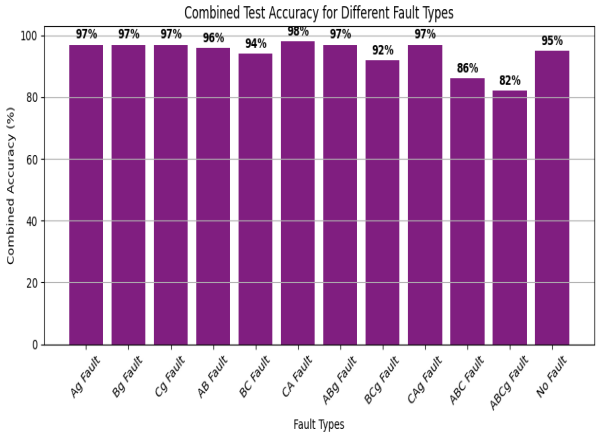


Figure 19: Combined Test Accuracy for Different Fault Types

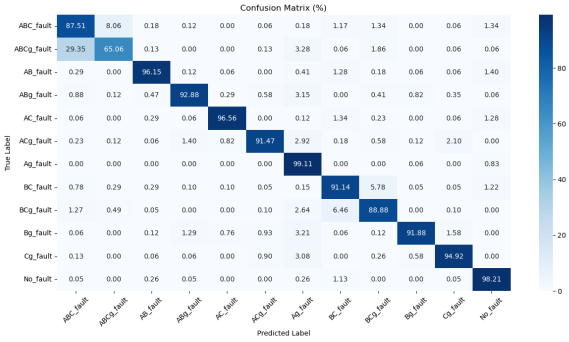


Figure 20: Confusion Matrix of Single Prediction Model

the custom CNN model outperforms traditional pre-trained models, achieving an accuracy rate of 91-94% across various fault types. Additionally, the Vision Transformer (ViT) architecture demonstrates superior performance in handling complex fault scenarios, further enhancing detection efficiency. Moreover, the introduction of a Single Prediction Model provides a flexible solution for real-time fault detection without relying on specific phasor information. Overall, the proposed method contributes to more reliable and timely fault identification and classification.

References

[1] Majid Jamil, Sanjeev Kumar Sharma, and Rajveer Singh. Fault detection and classification in electrical power transmission system using artificial neural network. *SpringerPlus*, 4:1–13, 2015.

[2] EPRI Home. URL https://www.epri.com/research/products/3002000476.

[3] J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye. *Power System Analysis and Design*. Cengage Learning, 2017.

[4] V. Rizeakos, A. Bachoumis, N. Andriopoulos, M. Birbas, and A. Birbas. Deep learning-based application for fault location identification and type classification in active distribution grids. *Applied Energy*, 338:120932, 2023. ISSN 0306-2619. doi:https://doi.org/10.1016/j.apenergy.2023.120932. URL https://www.sciencedirect.com/science/article/pii/S0306261923002969.

[5] Anam Abid, Muhammad Tahir Khan, and Javaid Iqbal. A review on fault detection and diagnosis techniques: basics and beyond. *Artificial Intelligence Review*, 54(5):3639–3664, 11 2020. doi:10.1007/s10462-020-09934-2. URL https://doi.org/10.1007/s10462-020-09934-2.

[6] Fatemeh Mohammadi Shakiba, Milad Shojaee, S. Mohsen Azizi, and Mengchu Zhou. Real-Time sensing and fault diagnosis for transmission lines. *International journal of network dynamics and intelligence*, pages 36–47, 12 2022. doi:10.53941/ijndi0101004. URL https://doi.org/10.53941/ijndi0101004.

[7] Hanif Livani and C. Yaman Evrenosoglu. A machine learning and wavelet-based fault location method for hybrid transmission lines. *IEEE Transactions on Smart Grid*, 5(1):51–59, 2014. doi:10.1109/TSG.2013.2260421.

[8] Jalal Khodaparast and Mojtaba Khederzadeh. Three-phase fault detection during power swing by transient monitor. *IEEE Transactions on Power Systems*, 30(5):2558–2565, 2015. doi:10.1109/TPWRS.2014.2365511.

[9] Daniel Guillen, Mario R. Arrieta Paternina, Jose Ortiz-Bejar, Rajesh Kumar Tripathy, Alejandro Zamora-Mendez, Ruben Tapia-Olvera, and Eric S. Tellez. Fault detection and classification in transmission lines based on a PSD index. *IET generation, transmission distribution*, 12(18):4070–4078, 9 2018. doi:10.1049/iet-gtd.2018.5062. URL https://doi.org/10.1049/iet-gtd.2018.5062.

[10] Abdul Qayyum Khan, Qudrat Ullah, Muhammad Sarwar, Sufi Tabassum Gul, and Naeem Iqbal. Transmission Line Fault Detection and Identification in an Interconnected Power Network using Phasor Measurement Units. *IFAC-PapersOnLine*, 51(24):1356–1363, 1 2018. doi:10.1016/j.ifacol.2018.09.558. URL https://doi.org/10.1016/j.ifacol.2018.09.558.

[11] Weilin Li, Antonello Monti, and Ferdinanda Ponci. Fault detection and classification in medium voltage dc shipboard power systems with wavelets and artificial neural networks. *IEEE Transactions on Instrumentation and Measurement*, 63(11):2651–2665, 2014. doi:10.1109/TIM.2014.2313035.

[12] Nassim Laouti, Nida Sheibat-Othman, and Sami Othman. Support vector machines for fault detection in wind turbines. *IFAC 18th World Congress*, 2011.

[13] Ye Zhao et al. Decision tree-based fault detection and classification in solar photovoltaic arrays. In *2012 Twenty-Seventh Annual IEEE Applied Power Electronics Conference and Exposition (APEC)*. IEEE, 2012.

[14] Y. Q. Chen, O. Fink, and G. Sansavini. Combined fault location and classification for power transmission lines fault diagnosis with integrated feature extraction. *IEEE Transactions on Industrial Electronics*, 65:INSPEC Accession Number: 7962173, 2018.

[15] M.F. Othman, M. Mahfouf, and D.A. Linkens. Transmission lines fault detection, classification and location using an intelligent power system stabiliser. In *2004 IEEE International Conference on Electric Utility Deregulation, Restructuring and Power Technologies. Proceedings*, volume 1, pages 360–365 Vol.1, 2004. doi:10.1109/DRPT.2004.1338522.

[16] Jianguo Wang. Fault diagnosis of underwater vehicle with fnn. In *Proceedings of the 10th World Congress on Intelligent Control and Automation*, pages 2931–2934, 2012. doi:10.1109/WCICA.2012.6358371.

[17] Kalanidhi K, Baskar D, and Vinod Kumar D. Transmission power line fault detection using convolutional neural networks. EAI, 6 2021. doi:10.4108/eai.7-6-2021.2308661.

[18] Ruojun Zheng, Li Zhu, Tao Hu, and Jun Li. Detection of fault insulator of power transmission line based on region-cnn. In *2020 35th Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pages 73–76, 2020. doi:10.1109/YAC51587.2020.9337692.

[19] Subhrajit Mitra, Rajarshi Mukhopadhyay, and Paramita Chattopadhyay. Pso driven designing of robust and computation efficient 1d-cnn architecture for transmission line fault detection. *Expert Systems with Applications*, 210:118178, 2022.

[20] F. B. Costa, A. Monti, F. V. Lopes, K. M. Silva, P. Jamborsalamati, and A. Sadu. Two-terminal traveling-wave-based transmission-line protection. *IEEE Transactions on Power Delivery*, 32(3):1382–1393, 2017. doi:10.1109/TPWRD.2016.2574900.

[21] D. Binu and B.S. Kariyappa. A survey on fault diagnosis of analog circuits: Taxonomy and state of the art. *AEU - International Journal of Electronics and Communications*, 73:68–83, 2017. ISSN 1434-8411. doi:https://doi.org/10.1016/j.aeue.2017.01.002. URL https://www.sciencedirect.com/science/article/pii/S1434841116308445.

[22] Anamika Yadav and Aleena Swetapadma. Enhancing the performance of transmission line directional relaying, fault classification and fault location schemes using fuzzy inference system. *IET Generation, Transmission Distribution*, 9(6):580 – 591, 2015. doi:10.1049/iet-gtd.2014.0498.

[23] Masoud Najafzadeh, Jaber Pouladi, Ali Daghigh, Jamal Beiza, and Taher Abedinzade. Fault detection, classification and localization along the power grid line using optimized machine learning algorithms. *International Journal of Computational Intelligence Systems*, 17(1), 3 2024. doi:10.1007/s44196-024-00434-7. URL https://doi.org/10.1007/s44196-024-00434-7.

[24] Ahmed Sami Alhanaf, Hasan Huseyin Balik, and Murtaza Farsadi. Intelligent fault detection and classification schemes for smart grids based on deep neural networks. *Energies*, 16(22), 2023. ISSN 1996-1073. doi:10.3390/en16227680. URL https://www.mdpi.com/1996-1073/16/22/7680.

[25] Wei Zhang, Xiang Li, and Qian Ding. Deep residual learning-based fault diagnosis method for rotating machinery. *ISA Transactions*, 95:295–305, 2019. ISSN 0019-0578. doi:https://doi.org/10.1016/j.isatra.2018.12.025. URL https://www.sciencedirect.com/science/article/pii/S0019057818305202.

[26] Vaishnavi V. Kulkarni, Vishwanath R. Hulipalled, Mayuri Kundu, Jay B. Simha, and Shinu Abhi. *Thermal Image-Based Fault Detection using Machine learning and Deep Learning in Industrial Machines: Issues-Challenges and Emerging Trends*. 1 2023. doi:10.1007/978-981-99-7093-3{_39. URL https://doi.org/10.1007/978-981-99-7093-3_39.

[27] Abdullah Al Safi, Christian Beyer, Vishnu Unnikrishnan, and Myra Spiliopoulou. *Multivariate time series as images: Imputation using convolutional denoising Autoencoder*. 1 2020. doi:10.1007/978-3-030-44584-3{_1. URL https://doi.org/10.1007/978-3-030-44584-3_1.

[28] Zhiguang Wang and Tim Oates. Imaging time-series to improve classification and imputation. *arXiv preprint arXiv:1506.00327*, 2015.

[29] Stephan Spiegel, Johannes-Brijnesh Jain, and Sahin Albayrak. *A Recurrence Plot-Based distance measure*. 1 2014. doi:10.1007/978-3-319-09531-8{_1. URL https://doi.org/10.1007/978-3-319-09531-8_1.

[30] Jiyoung Song, Young Chul Lee, and Jeongsu Lee. Deep generative model with time series-image encoding for manufacturing fault detection in die casting process. *Journal of Intelligent Manufacturing*, 34(7):3001–3014, 7 2022. doi:10.1007/s10845-022-01981-6. URL https://doi.org/10.1007/s10845-022-01981-6.

[31] David W. Romero, David M. Knigge, Albert Gu, Erik J. Bekkers, Efstratios Gavves, Jakub M. Tomczak, and Mark Hoogendoorn. Towards a general purpose CNN for long range dependencies in *N*D. *arXiv (Cornell University)*, 1 2022. doi:10.48550/arxiv.2206.03398. URL https://arxiv.org/abs/2206.03398.

[32] Xinyu Liu, Houwen Peng, Ningxin Zheng, Yuqing Yang, Han Hu, and Yixuan Yuan. Efficientvit: Memory efficient vision transformer with cascaded group attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Computer Vision Foundation, 2023. This CVPR paper is the Open Access version, provided by the Computer Vision Foundation. Except for this watermark, it is identical to the accepted version; the final

published version of the proceedings is available on IEEE Xplore.

[33] C. Weng, B. Lu, and J. Yao. A one-dimensional vision transformer with multiscale convolution fusion for bearing fault diagnosis. In *Proceedings of the 2021 Global Reliability and Prognostics and Health Management (PHM-Nanjing)*, pages 1–6, Nanjing, China, October 2021. IEEE, IEEE.

[34] Lisa-Marie Vortmann, Jannes Knychalla, Sonja Annerer-Walcher, Mathias Benedek, and Felix Putze. Imaging time series of eye tracking data to classify attentional states. *Frontiers in neuroscience*, 15, 5 2021. doi:10.3389/fnins.2021.664490. URL https://doi.org/10.3389/fnins.2021.664490.

[35] S. Shajun Nisha and M. Nagoor Meeral. 9 - applications of deep learning in biomedical engineering. In Valentina Emilia Balas, Brojo Kishore Mishra, and Raghvendra Kumar, editors, *Handbook of Deep Learning in Biomedical Engineering*, pages 245–270. Academic Press, 2021. ISBN 978-0-12-823014-5. doi:https://doi.org/10.1016/B978-0-12-823014-5.00008-9. URL https://www.sciencedirect.com/science/article/pii/B9780128230145000089.

[36] Rafia Nishat Toma, Farzin Piltan, Kichang Im, Dongkoo Shon, Tae Hyun Yoon, Dae-Seung Yoo, and Jong-Myon Kim. A bearing fault classification framework based on image encoding techniques and a convolutional neural network under different operating conditions. *Sensors*, 22(13), 2022. ISSN 1424-8220. doi:10.3390/s22134881. URL https://www.mdpi.com/1424-8220/22/13/4881.

[37] Tanmay Thaker. Vgg 16 easiest explanation, 2021. URL https://medium.com/nerd-for-tech/vgg-16-easiest-explanation-12453b599526.

[38] Suvaditya Mukherjee. The annotated resnet-50 explaining how resnet-50 works and why it is so popular, 2022. URL https://towardsdatascience.com/the-annotated-resnet-50-a6c536034758.

[39] Wisdomml. Efficientnet and its performance comparison with other transfer learning networks, 2023. URL https://wisdomml.in/efficientnet-and-its-performance-comparison-with-other-transfer-learning-networks/.

[40] Atakan Erdoğan. Convnext — next generation of convolutional networks, 2023. URL https://medium.com/@atakanerdogan305/convnext-next-generation-of-convolutional-networks-325607a08c46.

[41] Shahriar Rahman Fahim, Yeahia Sarker, Subrata K. Sarker, Md. Rafiqul Islam Sheikh, and Sajal K. Das. Self attention convolutional neural network with time series imaging based feature extraction for transmission line fault detection and classification. *Electric Power Systems Research*, 187:106437, 2020. ISSN 0378-7796. doi:https://doi.org/10.1016/j.epsr.2020.106437. URL https://www.sciencedirect.com/science/article/pii/S037877962030242X.

[42] Yuqing Gao and Khalid Mosalam. Deep transfer learning for image-based structural damage recognition. *Computer-Aided Civil and Infrastructure Engineering*, 33, 04 2018. doi:10.1111/mice.12363.

[43] Andrei-Cristian Rad. Understanding the vision transformer and counting its parameters, 2020. URL https://medium.com/analytics-vidhya/understanding-the-vision-transformer-and-counting-its-parameters-988a4ea2b8f3.