

**REPUBLIC OF TURKEY
ISTANBUL GELIŞİM UNIVERSITY
INSTITUTE OF GRADUATE STUDIES**

Department of Electrical-Electronic Engineering

**PARTIAL DISCHARGE DETECTION USING
CONVOLUTIONAL NEURAL NETWORK AND K-
NEAREST NEIGHBOR ALGORITHM**

Master Thesis

NOOR QASIM JEBUR AL ZAWI

Supervisor

Assoc. Prof. Dr. Indrit MYDERRIZI

Istanbul – 2022

THESIS INTRODUCTION FORM

Name and Surname : Noor Qasim Jebur AL ZAIDAWI

Language of the Thesis : English

Name of the Thesis : Partial Discharge Detection Using Convolutional Neural Network and K-Nearest Neighbor Algorithm

Institute : Istanbul Gelişim University Graduate Education Institute

Department : Electrical-Electronic Engineering

Thesis Type : Post Graduate

Date of the Thesis : 29.06.2022

Page Number : 82

Thesis Supervisors : Assoc. Prof. Dr. INDRIT MYDERRIZI

Index Terms : Convolutional Neural Network (CNN), K-nearest neighbor algorithm (KNN).

Turkish Abstract : Kısmi Deşarj, IEC-60270 standardına göre " İletkenler arasındaki yalıtımı kısmen kapatan ve bir iletkenin yakınında meydana gelebilecek veya oluşamayacak yerel bir elektrik boşalması" olarak tanımlanabilir. Orta gerilim elektrik hatları elektriği uzun mesafelere taşır, bunun sonucunda yetkililer bu alanlardaki kabloların kısmi boşalmasını izlemekte ve erken tespit etmekte zorlanırlar. Zorluk, uzun vadeli hasarı önlemek için kısmi deşarjları yeterince erken tespit etmektir, bu nedenle maliyetli onarımları ve önemli elektrik kesintilerini önlemek için erken kısmi deşarj tespiti önemlidir. Kısmi Deşarjın sınıflandırılması ve tanınması için geleneksel yöntemler, özelliklerin manuel olarak çıkarılmasına ve bir elektrik akımındaki çok özel darbeleri tanımlama uzmanlığına bağlıdır, bu nedenle, kısmi deşarjı yeterince hızlı tahmin etmek için özellikleri çıkarabilen ve bunları otomatik olarak sınıflandırabilen bir algılama mekanizmasına sahip

olmak esastır. Veri seti kaggle.com'dan toplanmıřtır. Ostrava Teknik Üniversitesi (VSB) ve Ostrava Teknik Üniversitesi, bařıboř elektrik alanlarının ve yalıtılmıř havai kabloların voltaj sinyalini belirlemek için özel bir sayaç tasarladı. Sınıflandırma modellerini oluřturmadan önce sinyaller için ön iřleme adımı olarak gürültüyü gidermek için hızlı fourier transformatör teknięi kullanılmıřtır. Dört sınıflandırma modeli oluřturulmuř ve birbirleriyle karřılařtırılarak modeller CNN-KNN, CNN, CNN+LSTM ve KNN, aęırlıkları güncellemek için Adam optimizasyon algoritması ve çıktıyı aralık içinde tutmak için sigmoid iřlevi kullanılmıřtır. 0_1, çünkü modelin eęitimi sırasında ařırı sıęmayı önlemek için katmanlar arasında ikili sınıflandırma ve bırakma teknięi. Model performansı birkaç metrik kullanılarak ölçüldü: F-score, recall, precision, accuracy ve confusion matrix, sonuçlar modellerin (CNN-KNN) modellerden (CNN, CNN-LSTM ve KNN) daha yüksek performans verdięini gösteriyor.

Distribution List

- : 1. To the Institute of Graduate Studies of Istanbul Geliřim University
2. To the National Thesis Center of YÖK (Higher Education Council)

Noor Qasim Jebur AL ZAIDAWI

**REPUBLIC OF TURKEY
ISTANBUL GELIŞİM UNIVERSITY
INSTITUTE OF GRADUATE STUDIES**

Department of Electrical-Electronic Engineering

**PARTIAL DISCHARGE DETECTION USING
CONVOLUTIONAL NEURAL NETWORK AND K-
NEAREST NEIGHBOR ALGORITHM**

Master Thesis

NOOR QASIM JEBUR AL ZAIDAWI

Supervisor

Assoc. Prof. Dr. Indrit MYDERRIZI

Istanbul – 2022

DECLARATION

I hereby declare that in the preparation of this thesis, scientific ethical rules have been followed, the works of other persons have been referenced in accordance with the scientific norms if used, there is no falsification in the used data, any part of the thesis has not been submitted to this university or any other university as another thesis.

NOOR QASIM JEBUR AL ZAIDAWI

...../...../2022



TO ISTANBUL GELISIM UNIVERSITY
THE DIRECTORATE OF INSTITUTE OF GRADUATE STUDIES

The thesis study of Noor Qasim ALzaidawi titled as Partial Discharge Detection Using Convolutional Neural Network and K-Nearest Neighbor Algorithm has been accepted as MASTER THESIS in the department of ELECTRICAL-ELECTRONIC ENGINEERING by out jury.

Director

Assoc. Prof. Dr. Indrit MYDERRIZI
(Supervisor)

Member

Asst. Prof. Dr. Khalid Q. Moh. YAHYA

Member

Asst. Prof. Dr. Hakan AKÇA

APPROVAL

I approve that the signatures above signatures belong to the aforementioned

faculty members

.... / ... / 2022

Signature

Prof. Dr. İzzet GÜMÜŞ

Director of the Institute

SUMMARY

A partial discharge (PD), according to IEC 60270, is a localized electrical discharge that only partially bridges the insulation between conductors. Medium voltage power lines transport electricity over long distances; as a result, officials have a difficult time monitoring and early detecting the PD of the cables in these areas. The challenge is to detect PDs early enough to prevent long-term damage; hence, early PD detection is essential to prevent costly repairs and substantial power outages. There are numerous approaches for measuring PDs online, but the most common method is done offline and includes an expert manually identifying acceptable features to classify PD type and severity, thus, it is essential to have a detection mechanism that can extract features and classify them automatically to predict PDs rapidly and accurately. The dataset was collected from kaggle.com. The Technical University of Ostrava (VSB) has designed a specialized meter to determine the voltage signal of stray electrical fields and insulated overhead cables. The fast Fourier transform technique was used to de-noise the signals as a preprocessing step before building the classification models. Four classification models were built and compared with each other; the models are CNN-KNN, CNN, CNN-LSTM, and KNN. The Adam optimization algorithm was used to update the weights to reduce the error between actual output and predicted output, and the sigmoid function was used in the output layer to keep the output within the range 0–1, and used dropout technique between layers to prevent overfitting during training of the model. Model performance was measured using several metrics: F-score, recall, precision, accuracy, and confusion matrix; the results show that the hybrid model (CNN-KNN) gives higher performance than the other models (CNN, CNN-LSTM, and KNN).

Keywords : Convolutional neural network (CNN), K-nearest neighbor algorithm (KNN), partial discharge (PD), and fast Fourier transform (FFT).

ÖZET

Kısmi Deşarj, IEC-60270 standardına göre " İletkenler arasındaki yalıtımı kısmen kapatan ve bir iletkenin yakınında meydana gelebilecek veya oluşamayacak yerel bir elektrik boşalması" olarak tanımlanabilir. Orta gerilim elektrik hatları elektriği uzun mesafelere taşır, bunun sonucunda yetkililer bu alanlardaki kabloların kısmi boşalmasını izlemekte ve erken tespit etmekte zorlanırlar. Zorluk, uzun vadeli hasarı önlemek için kısmi deşarjları yeterince erken tespit etmektir, bu nedenle maliyetli onarımları ve önemli elektrik kesintilerini önlemek için erken kısmi deşarj tespiti önemlidir. Kısmi Deşarjın sınıflandırılması ve tanınması için geleneksel yöntemler, özelliklerin manuel olarak çıkarılmasına ve bir elektrik akımındaki çok özel darbeleri tanımlama uzmanlığına bağlıdır, bu nedenle, kısmi deşarjı yeterince hızlı tahmin etmek için özellikleri çıkarabilen ve bunları otomatik olarak sınıflandırabilen bir algılama mekanizmasına sahip olmak esastır. Veri seti kaggle.com'dan toplanmıştır. Ostrava Teknik Üniversitesi (VSB) ve Ostrava Teknik Üniversitesi, başıboş elektrik alanlarının ve yalıtılmış havai kabloların voltaj sinyalini belirlemek için özel bir sayaç tasarladı. Sınıflandırma modellerini oluşturmadan önce sinyaller için ön işleme adımı olarak gürültüyü gidermek için hızlı fourier transformatör tekniği kullanılmıştır. Dört sınıflandırma modeli oluşturulmuş ve birbirleriyle karşılaştırılarak modeller CNN-KNN, CNN, CNN+LSTM ve KNN, ağırlıkları güncellemek için Adam optimizasyon algoritması ve çıktıyı aralık içinde tutmak için sigmoid işlevi kullanılmıştır. 0_1, çünkü modelin eğitimi sırasında aşırı sığmayı önlemek için katmanlar arasında ikili sınıflandırma ve bırakma tekniği. Model performansı birkaç metrik kullanılarak ölçüldü: F-score, recall, presision, accuracy ve confusion matrix, sonuçlar modellerin (CNN-KNN) modellerden (CNN, CNN-LSTM ve KNN) daha yüksek performans verdiğini gösteriyor.

Anahtar Kelimeler : evrişimli sinir ağı (CNN), KNN en yakın komşu algoritması (KNN), kısmi deşarj (PD), Hızlı Fourier dönüşümü (FFT).

TABLE OF CONTENTS

SUMMARY	I
ÖZET.....	II
TABLE OF CONTENTS	III
ABBREVIATIONS	VI
LIST OF TABLES	VII
LIST OF FIGURES	VIII
INTRODUCTION.....	1

CHAPTER ONE

PURPOSE OF THE THESIS

1.1. LITERATURE REVIEW	2
1.2. PROBLEM STATEMENT	6
1.3. PURPOSE OF THE STUDY	6
1.4 THE STUDY ORGANIZATION:.....	7

CHAPTER TWO

THEORETICAL AND BACKGROUND

2.1 PARTIAL DISCHARGE (PD)	8
2.2 AUTOMATIC ONLINE PD DETECTION BY DEEP LEARNING	9
2.3. SIGNALS PROCESSING	11
2.3.1 Savitzky-Golay Filter.....	12
2.3.2 Fast Fourier Transform (FFT).....	13
2.3.3 Low Pass Filter	13
2.3.4 High Pass Filter.....	14
2.3.5 Power Spectral Density Analysis.....	14
2.4 DEEP LEARNING FUNDAMENTALS	14
2.5 DEEP LEARNING MODELS.....	16
2.5.1. Convolutional Neural Network.....	16
2.5.2. Recurrent neural network (RNN)	17
2.6. MACHINE LEARNING:	20
2.6.1. K-Nearest Neighbor	20
2.7. TRAINING MODEL	22
2.7.1 Loss Function.....	22
2.7.2 Learning rate	23
2.7.3 Backpropagation	24

2.7.4. Optimization algorithm: Adaptive Moment Estimation (Adam).....	24
2.8 CHALLENGES IN TRAINING DEEP NETWORKS.....	25
2.8.1. Overfitting.....	25
2.8.2. Vanishing gradient.....	28
2.9. TYPES OF ACTIVATION FUNCTION	29
2.10. EVALUATION METRICS	31
2.10.1 Confusion Matrix.....	31
2.10.2 The receiver operating characteristic (ROC) curve	32
2.10.3. Accuracy	33
2.10.4. Precision/Recall	34
2.10.5. F1-Score.....	34

CHAPTER THREE

METHODOLOGY

3.1. SYSTEM OVERVIEW	35
3.2. THE WORKFLOW	35
3.3. DATASET	36
3.3.1. Signal data.....	36
3.3.2. Metadata.....	36
3.4 PREPROCESSING.....	37
3.5. FEATURES EXTRACTION.....	38
3.6. CLASSIFICATION	38
3.6.1. Construction of the CNN Model.....	39
3.6.2 Construction of the CNN-LSTM Model.....	41
3.6.3. Construction of the KNN model.....	43
3.6.4. Construction of the CNN-KNN Model.....	44

CHAPTER FOUR

RESULTS AND DISCUSSION

4.1. GENERAL.....	46
4.2. EXPERIMENTAL RESULTS OF DATASET	46
4.3 EXPERIMENTAL RESULTS OF PREPROCESSING.....	49
4.4. EXPERIMENTAL RESULTS OF FEATURES EXTRACTION.....	50
4.5. EXPERIMENTAL RESULTS OF CLASSIFICATION MODELS.....	50
4.5.1. First Experiment (CNN Model).....	50
4.5.2 Second Experiment (CNN-LSTM) model.....	52

4.5.3. Third experiment (CNN-KNN and KNN) models:	54
4.6. EVALUATION METRICS	56
4.6.1. Evaluation of the CNN, and CNN+ LSTM model	56
4.6.2 Evaluation of the KNN, and CNN+KNN model	57
4.7 RESULTS COMPARISON	58

CHAPTER FIVE

CONCLUSIONS AND FUTURE WORK

5.1 Conclusions	60
5.2. Future Works:	60
REFERENCES	61



ABBREVIATIONS

Adam	Adaptive moment estimation
AI	Artificial Intelligence
ANN	Artificial Neural Network
BCE	Binary Cross-Entropy
CNN	Convolution Neural Network
DL	Deep Learning
DNN	Deep Neural Network
FFT	Fast Fourier Transform
KNN	K-nearest neighbor
LSTM	Long Short-Term Memory
ML	Machine Learning
PD	Partial discharge
RELU	Rectified linear unit
RNN	Recurrent Neural Network
SVM	Support Vector Machine
SGD	Stochastic Gradient Descent

LIST OF TABLES

Table 1. Accuracy of sigmoid and RELU in deep networks (Catterson and Sheng 2015)	2
Table 2. Accuracy results comparison between SSAE model and SVM technique (Tang et al. 2017)	3
Table 3. The description parameters of the confusion matrix (Jehad & Yousif, 2020)	31
Table 4. Confusion Matrix Description (Čeponis, D., & Goranin, N. 2020)	32
Table 5. Trial and error method for choosing the best parameters.....	51
Table 6. Trial and error method for choosing the best parameters.....	51
Table 7. Evaluation performance of CNN+LSTM.....	53
Table 8. Evaluation performance of CNN+LSTM (2 conv layer, 2 max-pooling layer)	54
Table 9. Evaluation performance of CNN+KNN and KNN models.....	55
Table 10. Trial and error method for choosing the best parameters (CNN-KNN, KNN).....	55
Table 11. The final results of the CNN-LSTM and CNN models	56
Table 12. The final results of the CNN-KNN and KNN models	57
Table 13. Comparison of performance of different models	59

LIST OF FIGURES

Figure 1. LSTM fusion and the multi-resolution CNN (Li et al. 2018).....	4
Figure 2. The waveform of a partial discharge in the time domain (Barrios, et al. 2019).	9
Figure 3. Ultrasonic signals (Kong, et al., 2020)	11
Figure 4. The Savitzky-Golay filtering example (Samarkhanov, K et al. 2019) 12	
Figure 5. Artificial Intelligence (AI) taxonomy (Alom, et al.2019).	15
Figure 6. The difference between ML and DL (Bengio, 2009)	16
Figure 7. The basic structure of the artificial neural network (Ghosh, et al. 2014)	16
Figure 8. Architecture of a CNN (Barrios, et al. 2019).....	17
Figure 9. RNN architecture (Bhattacharyya, et al 2018).....	18
Figure 10. Architectures of LSTM cell and RNN (Li, Wang, Yang & Rong, 2018)	19
Figure 11. Example of KNN classification task with k=5 (Cheng et al. 2014)..	20
Figure 12. Euclidean distance and Manhattan distances (Ranjitkar, H. S., & Karki, S., 2016)	21
Figure 13. The early stopping technique (Gençay & Salih 2003).....	26
Figure 14. Standard NN and after adding the dropout technique (Alom, et al. 2019)	27
Figure 15. Bernoulli distribution (Starmans, et al. 2021).	28
Figure 16. Sigmoid function (Bahdanau et al. 2014)	29
Figure 17. ReLU function (Rysbek, D. 2019).....	30
Figure 18. Roc curve (Woods & Bowyer 1997)	33
Figure 19. Model building workflow	35
Figure 20. Denoising signals using the FFT technique.....	37
Figure 21. The general diagram of the DL proposed models.....	39

Figure 22. Construction of the CNN Model.....	40
Figure 23. Construction of the CNN-LSTM Model.....	41
Figure 24. Steps of the KNN algorithm	44
Figure 25. Construction of the CNN-KNN Model.....	44
Figure 26. Training dataset.....	46
Figure 27. The distribution of the sample	47
Figure 28. Testing dataset	47
Figure 29. The dataset parquet for 9 signals	48
Figure 30. The raw data: 3 phases over one period.....	48
Figure 31. Result of FFT filter	49
Figure 32. De-noised signals for three phases.....	49
Figure 33. Features extracted from the signals	50
Figure 34. The history of acc and loss of CNN model.....	52
Figure 35. The Roc curve of the CNN model.	52
Figure 36. The history of acc and loss of the (CNN+LSTM) model	53
Figure 37. The Roc curve of the CNN-LSTM model	54
Figure 38. The Roc curve for the CNN-KNN and KNN models.....	55
Figure 39. Confusion matrix of (CNN) and (CNN+LSTM) models after 15 epochs.....	56
Figure 40. Confusion matrix of (CNN+KNN) and (KNN) model.....	57

PREFACE

During the preparation and writing process of this thesis, I would like to thank my esteemed professor Assoc. Prof. Dr. Indrit MYDERRİZİ.

I would like to thank the thesis's jury members for their help in managing this thesis and taking it forward with their valuable comments and suggestions throughout the process. I would like to express my sincere gratitude to Assoc. Dr. Öğr. Üyesi Khalid and Dr. Öğr. Üyesi Hakan AKÇA , the staff of Istanbul Gelişim University Electrical and Electronics Engineering Department and the Institute of Science.

Finally, I want thanking my family for all of the assistance they have given me in my academic career by dedicating this work to them.

INTRODUCTION

A Partial discharge (PD) is defined as "A localized electrical discharge that just part-way bridges the insulation between conductors and can or cannot occur near to a conductor," according to the standard IEC-60270. PDs are caused by localized electrical stress concentrations in the insulation or on the insulation's surface. PDs can occur in overhead lines, underground cables, and transformers. In Northern Europe, fault classifications in medium voltage (MV) overhead distribution networks are classified into snow load 35%, trees falling on the lines 27%, tree branches on pole transformers 9%, lightning 6%, diggers 6%, and animals (Hanninen, Lehtonen, & Hakola, 2002). PD monitoring provides a caution for a power system component to be removed from service before a disastrous failure occurs because PD usually occurs before the total breakdown (Babnik, et al. 2003). The capacity to detect defects and act accordingly is essential for maintaining the electrical network system (Shafiq, et al. 2014). The detection of PDs is important because electrical energy has become an essential part of the global industrial revolution; hence, protecting and maintaining power sources is vital to ensure the sustainability and continuity of electrical energy reaching factories and cities. The most common causes of PDs are voids and cavities within solid insulators, bubbles within liquid insulators, and surface damage to insulators; PDs occur when these weaknesses are exposed to high voltages. Therefore, PD detection is essential because if the PD is ignored and allowed to continue, it will gradually have a cumulative harmful effect on the insulation material and eventually lead to the breakdown of the insulation material, resulting in service interruption (Babnik, et al. 2003). Several techniques are used for the online detection of PDs, including ultrasonic, ultraviolet, and radiofrequency methods; but the basic classification and recognition techniques are employed offline and require experts to manually extract features from the data and then utilize them to determine the type and severity of the PD. However, these techniques are incomplete and have drawbacks because they lack the necessary accuracy to ensure immediate repair and they place the user near potentially dangerous equipment. Machine learning, deep learning, and data analytics have opened up new possibilities for improving maintenance scheduling and system reliability (Kong, et al. 2020).

CHAPTER ONE

PURPOSE OF THE THESIS

1.1. LITERATURE REVIEW

Many researchers have used artificial intelligence including machine learning and deep learning models to address the problem of PD detection. Some of the studies are listed below:

Catterson and Sheng (2015) proposed deep neural networks (DNNs) for the diagnosis of phase-resolved partial discharge (PRPD) data. An ultra-high-frequency (UHF) sensor was used to collect data from fault samples in oil. Roughly 250–300 PRPD patterns were measured using the UHF sensor. The goal was to categorize the PD faults in oil into 6 categories. The number of layers used in these tests ranged from one to seven. They found that five hidden layers is a good number and increased the accuracy from (81%) to (86%) by using the rectified linear activation function (ReLU). The ReLU function was compared with the sigmoid function, as shown in the following table 1.

Table 1. Accuracy of sigmoid and RELU in deep networks (Catterson and Sheng 2015)

Number of hidden layers	1	2	3	4	5	6
Sigmoid accuracy	72%	41%	17%	18%	17%	18%
RELU accuracy	81%	80%	83%	84%	86%	81%

Lu et al. (2016) proposed a convolutional NN-based transient earth voltage sensor (TEV) detection method that eliminates the need for humans to create signal features and solves detection issues caused by a poorly selected feature. A five-layer convolutional NN was used with a softmax output function. Switchgear measurements were recorded as sound clips and then translated into an image using the TEV sensor data represented by time-frequency spectra. Three thousand pictures

with five hundred PD signals were used to train the CNN. The accuracy of the model was 95.58%.

Tang et al. (2017) developed a deep-learning NN model called a stacked sparse auto-encoder (SSAE) and used it to extract features from the intermediate layer with a limited number of nodes. The model produces an output feature that is nearly identical to the input PD data. The collected features from the PD data are then fed into a soft-max classifier, which classifies the features into one of four PD severity levels. The data came from experimental cells in a gas-insulated switchgear (GIS) enclosure that simulated four different forms of PD defects. The UHF PD signals were captured with an antenna, and a PRPD representation was obtained for each PD type at various voltages to illustrate its development. Also, nine statistical features were calculated and compared to the SSAE method as input for a support vector machine (SVM) algorithm, as shown in table 2. The SSAE method was accurate in recognizing the four defects.

Table 2. Accuracy results comparison between SSAE model and SVM technique (Tang et al. 2017)

Defect	SSAE(%)	SVM(%)
Protrusion	88	77
Particle	93	81
Contamination	88	80
Gap	90	83

Li et al. (2018) proposed a deep-learning-based PD classification algorithm that uses a multi-column conventional neural network (CNN), coupled with long short-term memory (LSTM) for PD detection. The input data type is a waveform spectrogram of artificial PD signals in a GIS tank recorded using UHF sensors. The proposed architecture includes three essential models: CNN, LSTM, and fully connected layers; for each UHF signal three separate short-time Fourier transforms were generated, each with various window lengths to represent the signal in three different spectrograms. These three spectrograms fed into three independent sub-

networks, merged using a fully-connected layer. Figure 1 depicts the planned multi-resolution network. Single-resolution embedding, multi-resolution fusion, and multi-sensor fusion are the three main components. The LSTM network merges the information signals that have been recorded by the sensors. The accuracy of the model is 98.2%.

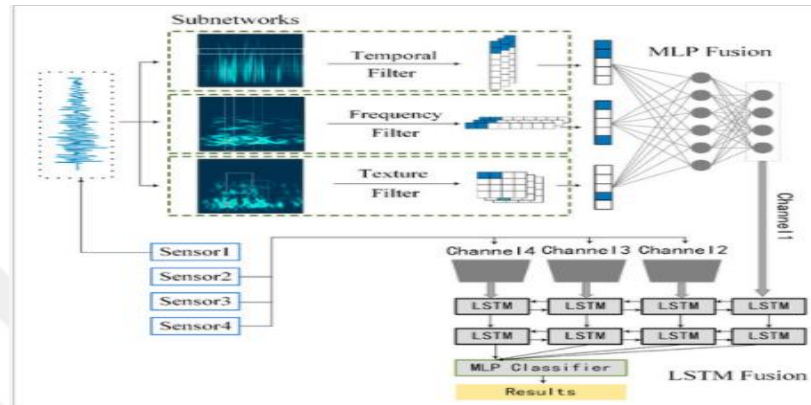


Figure 1. LSTM fusion and the multi-resolution CNN (Li et al. 2018)

Nguyen et al. (2018) proposed a recurrent neural network (RNN) structure with LSTM; the proposed model efficiently learns low-level characteristics and temporal dependencies of PRPDs using training data, based on a sequence of PRPDs in a GIS. Finally, a softmax layer was utilized to categorize four different PD types and an artificial noise source. The PD types (corona, floating, free particle, and void defect) were artificially created in the laboratory using pre-constructed cells mounted in a 345 kV GIS chamber. A data set for PRPD was collected and split into three sets using data augmentation. These three sets were used to establish training goals, conduct cross-validation, and evaluate performance. For categorizing PRPD in GIS, the suggested model is better than standard ANN and SVM techniques. The accuracy of this model was 96.62%.

Wang et al. (2018) proposed generative adversarial networks (GAN) that provide a unique data-generating model to improve PD Pattern recognition. For the GAN to categorize three PD sources, the artificial NN was trained with actual data and artificially created data, and then evaluated with real data only. When the model is trained with the same amount of real and fake data, the classification accuracy

improves, but when the fake data is larger than the actual data, the classification accuracy declines.

Duan, et al. (2019) proposed a deep learning network based on a sparse auto-encoder (SAE) and softmax that achieves a suitable classification result of more than 96 percent. Following a thorough selection of parameters. Furthermore, altering from sigmoid to the ReLU activation function increased the accuracy to 99.7 percent.

Dong, Sun, & Wang, (2019) used two main techniques in their proposed pattern recognition method: seasonal and trend decomposition (STL), and SVM. To recognize PD activities on insulated overhead conductors (IOCs), a new method was designed based on STL and SVM. STL is a technique for decomposing time series. SVM was used as a binary classifier, and various SVM kernels were evaluated and compared. The employment of a Gaussian radial basis kernel resulted in satisfactory classification rates on the VSB power line fault detection dataset. Actual PD signals may be recognized in 88 percent of cases, and 68 percent of observed PD signals are actual PD signals.

Qu, et al. (2020) proposed a method for fault detection on IOCs using a discrete wavelet transform (DWT) and LSTM. The original signal was de-noised by DWT. Next, DWT was employed to decompose the de-noised signal and extract characteristics on separate layers. Finally, the LSTM was used to detect the IOC problem; the VSB dataset of kaggle.com was used.

Samaitis et al. (2020) proposed a non-invasive PD detection and localization system that combines ultrasonic signal processing with machine learning to filter PD signals from background noise. Three machine learning classes were implemented and tested: SVM with a radial basis function kernel (RBF), naive Bayes (NB), and linear discriminant analysis. The partial discharge was recognized using CNNs after a dataset of discharge-induced signals had been collected using a laboratory corona discharge simulator.

Michau, Hsu, & Fink, (2021) proposed interpretable detection of PD in power lines with deep learning using the VSB dataset. Their proposed NN consisted of two blocks, the first block contains two convolutional layers with a filter length of 16. The second block contains two convolutional layers with 8 filter. A maximum 1D time-pooling layer follows each block, and then the global average pooling layer

with a size of 32 was calculated to determine which part of the input leads to the categorization results of the networks. The ReLU layer is the activation function except for the last output. Each phase is handled independently, and a high-pass filter processes the signal to reduce signal noise. The accuracy of the model has reached 0.967.

1.2. PROBLEM STATEMENT

A PD may occur in a power line because of a defect in the manufacture of the insulator or contact with a tree branch, for example. A PD will slowly damage the power line, so ignoring or not detecting this discharge and leaving it unrepaired will eventually destroy the insulator and cause a power outage. PD has been a very prevalent problem in recent years. This prompts us to focus on developing models for PD detection, using machine learning and deep learning classifiers to overcome the following problems:

- Medium-voltage power lines transmit power over long distances to supply electric power to factories and cities. Therefore, it is difficult for officials to monitor the cables in these areas.
- PD classification and recognition traditionally rely on manual and domain expertise extracting the features to recognize extremely precise pulses in an electrical current.

1.3. PURPOSE OF THE STUDY

The goal of this thesis is to provide a model for high-accuracy prediction of PDs, to detect PDs in three-phase MV overhead power lines using the hybrid CNN-KNN, and to compare the new model with other models that were implemented on the same dataset to understand the prediction performance of other models. The prediction performance of the models is evaluated using different hyper-parameters in several layers of the CNN until the optimized results are obtained. The scope of the thesis is:

- Using the FFT technique to reduce the noise of the original signal.
- Analyzing and classifying the data using machine learning and deep learning classifiers (CNN, CNN-LSTM, CNN-KNN, and KNN).
- Using the dropout technique to prevent overfitting during network training.
- Using Adam's optimization algorithm to update the network's weights to obtain high accuracy in prediction.
- Testing and evaluating the models to obtain the best implementation for the PD detection models.

1.4 THE STUDY ORGANIZATION:

This study consists of the following chapters:

- **Chapter One:** presents previous studies related to our topic, the statement of the problem, and the goal of the thesis.
- **Chapter Two:** presents a theoretical background for the topic. The first part sheds light on the partial discharge phenomenon. The second part deals with deep learning fundamentals, machine learning, strategies, classes, regulation, and optimization. The third part focuses on various performance measures to evaluate the performance of models.
- **Chapter Three:** presents the dependent methodology for the models.
- **Chapter Four:** presents the evaluation of models with experimental results.
- **Chapter Five:** presents the conclusions and future work.

CHAPTER TWO

THEORETICAL BACKGROUND

2.1 PARTIAL DISCHARGE (PD)

PD occurs because of robust and inhomogeneous electrical fields (Hashmi, 2008), if left without proper maintenance or repairs, PD problems can eventually develop into significant service interruptions and serious safety problems. The detection of this discharge provides information about the state of equipment insulation, which can aid in locating the source of the problem; therefore, detection of the PD with proper diagnosis of the problem is considered very important. Sensors based on inductive, electromagnetic detection, and capacitive methods, as well as near-field antennas, are used to measure the PD. The sensor's output signals are high-frequency damped oscillating pulses, and each PD pulse waveform can be represented in the time domain, as shown in the figure 2. Studies show that more than 85% of high-voltage (HV) and MV equipment faults are due to PD, and PDs eventually lead to failure. A PD can also be defined as a localized electrical discharge in a small insulator. A PD can occur at any point in the insulating system when the electric field strength exceeds the breakdown strength of that part of the insulator. A PD can also occur on the surface of the insulating material, within gas bubbles in liquid insulation materials, in voids within solid insulation materials, or around carriers when corona appears (Majidi, et al. 2015). Corona discharge is the result of the ionization of the air surrounding the transmission lines as a result of the presence of an irregular electric field in these lines or as a result of the ionization of air near the HV electrode (Shafiq, et al. 2014). Surface discharge occurs on the surface of a dielectric material or between two materials. In cables, especially cables made of several rigid insulating materials, internal PD can occur inside the insulation. During a PD, energy is emitted as electromagnetic emissions, acoustic emissions, or as ozone and oxides.

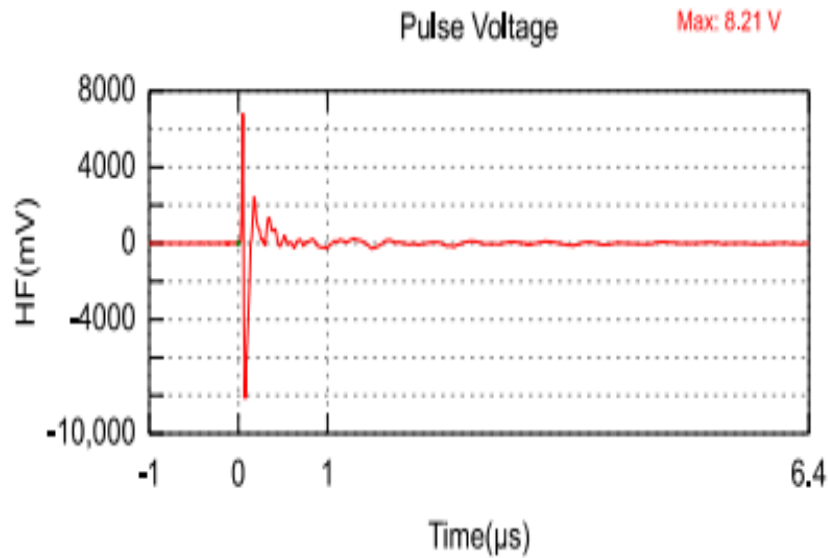


Figure 2. The waveform of a partial discharge in the time domain (Barrios, et al. 2019).

Several detection techniques are used to detect PDs:

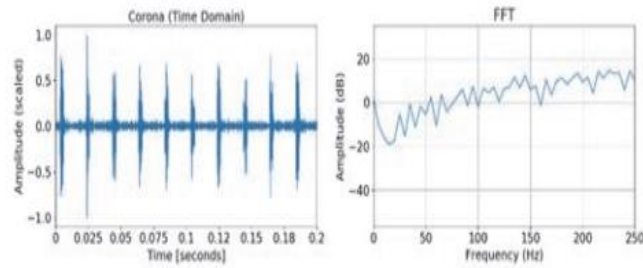
- The ultrasonic wave technique depends on detecting the sound emissions resulting from the discharge; using equipment to convert the ultrasonic frequency into a comfortable, audible frequency; and then sending it to headphones and measuring its intensity.
- The capacitive coupling technique is used for internal discharges. Energy is detected at the outer surface of the surrounding metal using the principle of capacitive coupling.

2.2 AUTOMATIC ONLINE PD DETECTION BY DEEP LEARNING

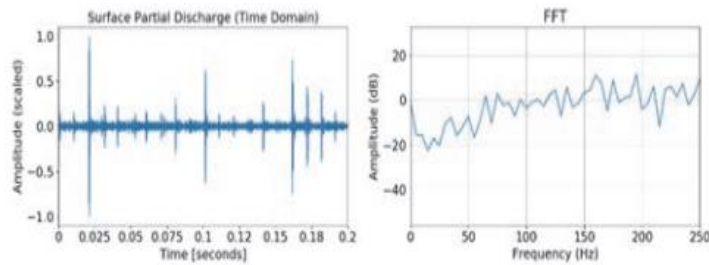
Automatic detection is now achievable because of new sensing techniques and data analytics methods. Online PD testing is less expensive than offline testing, which requires service and production interruption. Detection and interpretation of PD signals are the two main steps in online PD measurement (Stone, 2013). This testing can be implemented with various sensors. The PD signal is processed using

hardware and software methods. This includes signal processing with hardware and signal de-noising with software techniques, PD source identification by feature extraction, and pattern recognition techniques (Wu, et al. 2015). The rising popularity of automatic detection is due to improvements in detecting capabilities (Stone, 2013), along with the advancement of DSP techniques, such as software-based de-noising methods that improve the dependability of the measurement by overcoming noise and disturbances.

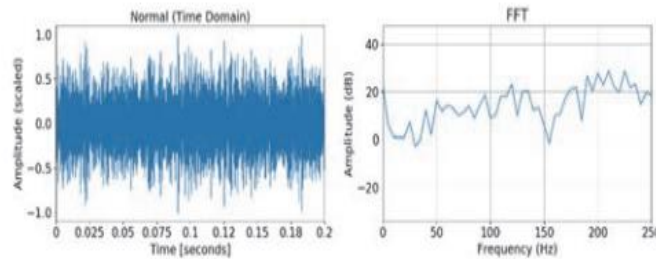
Furthermore, advances in feature extraction and single PD source categorization have been achieved rapidly (Wu, et al. 2015). It has been found that AI—specifically deep NN models—is much better than a traditional diagnosis for early detection and diagnosis of defects in terms of improving maintenance scheduling and improving system reliability. Figure 3 presents some of the signals that PD detection experts depend on as they manually search thousands of signals to identify the defect. Moreover, applying ultrasonic PD detection and diagnosis technologies includes manual labor and expertise from experts. Human experts in PD diagnosis, in particular, rely on FFT features to assist them in classifying ultrasonic signals (Kong, et al 2020). However, with the development of science and AI in partial discharge detection and fault type diagnosis, the problem of partial discharge is detected and diagnosed online using deep NNs where features can be extracted. The type of defect is detected and diagnosed through these NNs, which can reduce the cost of searching for PD defects.



Ultrasonic signals for corona discharge and the FFT analysis



Ultrasonic signals for surface discharge and the FFT analysis



Ultrasonic signals for normal condition and the FFT analysis

Figure 3. Ultrasonic signals (Kong, et al., 2020)

2.3. SIGNALS PROCESSING

It is impossible to remove all noise using conventional hardware-based noise reduction techniques. Improvements in computer science, combined with recent advances in signal processing methods, have resulted in a wide range of software-based noise reduction techniques being explored and developed (Wu, et al., 2015). De-noising the signal is the initial step before building the classifications model. The appearance of noise is a severe constraint during online PD testing. Furthermore, online monitoring suffers from significant electromagnetic interference, making it difficult to obtain a pure PD signal in real-world situations. A small, weak PD signal

can sometimes be drowned out by background noise. Recent advances in digital signal processing (DSP) have made it easy to extract PD signals. There are thirty-two different de-noising methods in common use, such as low-pass filtering, FFT filtering, wavelet de-noising, etc (Sriram, et al 2005).

2.3.1 Savitzky-Golay Filter

The Savitzky-Golay filter is a technique of smoothing curves (signals) that increases the signal-to-noise ratio (SNR) while retaining the peak value and other important characteristics of the original signal (Samarkhanov, et al 2019), as shown in the figure 4. Abraham Savitzky and Marcel J.E. Golay pioneered this technique in 1964. The data utilized have been compared before and after using the least exponent technique. For example, assume the data contains n , x_j , and y_j points ($j = 1, \dots, n$). The independent variable is x , and the dependent variable is y_j ; if there are m convolution coefficients, the following equation applies:

$$Y_j = \sum_{i=\frac{m-1}{2}}^{\frac{m-1}{2}} \frac{m-1}{2} \leq j \leq n - \frac{m-1}{2} \quad (1)$$

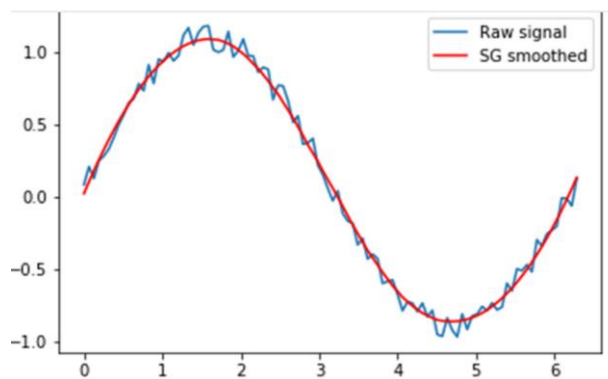


Figure 4. The Savitzky-Golay filtering example (Samarkhanov, K et al. 2019)

2.3.2 Fast Fourier Transform (FFT)

The FFT was first introduced in 1965 and is still considered an excellent method for de-noising signals (Santoso, Powers, & Grady, 1997). The concept of the FFT is to break the input signal into blocks, execute a FFT on each block, multiply by a frequency-domain filter function, and then return the filtered signal to the time domain using an inverse FFT (IFFT). The FFT converts the original time-domain signal to its frequency domain to differentiate between the desired signals and the noise signals by knowing the frequency bands of their respective spectra. To de-noise, it is necessary to recognize the periodicity of the signal. The FFT is a set of algorithms used to increase the efficiency of computing a discrete Fourier transform (DFT) for a series of samples (Cochran, et al.1967); the DFT of a time series can be quickly computed using the FFT. This means that the FFT technique saves time when transforming large sequences. Its inverse FFT enables a one-to-one connection between the time and frequency domains (Cochran, et al.1967), as shown in Eq.(2):

$$[x(t)] \longleftrightarrow [X(\omega)] \quad (2)$$

Assume that $F(\omega)$, as indicated in Eq.(3), is the FFT of the signal $f(t)$.

$$F(\omega) = \iint_{-\infty}^{\infty} f(t)e^{-j\omega t} dt \quad (3)$$

2.3.3 Low Pass Filter

A low-pass filter is used to filter signals to obtain more precise results. A low-pass filter was used to pass a signal with a lower frequency than the cut-off frequency (Africa, et al. 2020). A Low-pass filter is used in various applications, including audio hiss filters, digital data smoothing filters, acoustic barriers, and picture blurring.

2.3.4 High Pass Filter

Frequencies below a cutoff frequency (the stopband) are attenuated by a high-pass filter, while signals above the cutoff frequency are allowed through (the passband). High-pass filters remove low-frequency noise from audio signals. Higher-frequency signals can be routed to appropriate speakers in sound systems and low-frequency trends can be removed from time-series data, accentuating the high-frequency trends (Metwalli, et al. 2009).

2.3.5 Power Spectral Density Analysis

The PSD of a signal helps to separate the power distribution into different frequency components (McNamara, & Boaz, 2006). The PSD helps to analyze the distribution of frequency components of a signal, but unlike spectrograms, it does not provide time-resolved information. The PSD can be of particular interest when analyzing time-independent characteristics of music signals like timbre or chroma. PSD can be applied to stationary or quasi-stationary signals.

2.4 DEEP LEARNING FUNDAMENTALS

Since 1950, a small subset of AI has been referred to as Machine Learning (ML); moreover, neural networks (NNs) came to be regarded as a sub-field of ML. In turn, NNs gave rise to Deep Learning (DL) (Alom, et al 2019). Almost immediately, DL created massive disruptions in the technology because of its significant efficiency in most applications. Artificial NNs that closely mimic natural NNs are known as spiking NNs (SNNs). In addition to neuronal and synaptic states, SNNs incorporate the concept of time into their operating model. Figure 5 shows the Artificial Intelligence taxonomy.

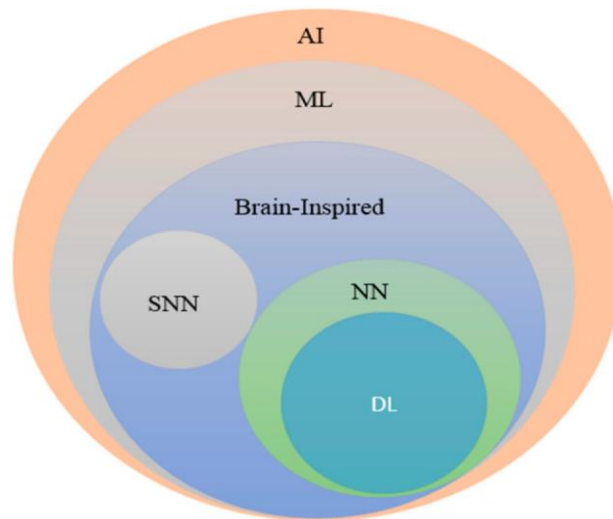


Figure 5. Artificial Intelligence (AI) taxonomy (Alom, et al.2019).

DL applies hierarchical learning methods or deep architectures of learning; it is one of the ML classes primarily developed from 2006 onward. In addition, DL is a process that consists of evaluating the model; thus, the learned model (algorithm) will have the ability to execute a specified mission. For instance, in Artificial Neural Networks (ANNs), the parameters are weight matrices. At the same time, DL includes many layers between the output and input layers, allowing various stages that include nonlinear information processing units with hierarchical architecture types to exist; these can be used for pattern classification and feature learning (Bengio, Schmidhuber, 2015). The learning approaches based on data representations might be described as representation learning. The latest studies indicated that DL-based representation learning includes a hierarchy of concepts or features, in which high-level concepts might evolve from concepts of low-level concepts (Bengio, et al 2013). A few articles have described DL as a universal learning method to solve all issues in various application domains (Bengio, 2009). DL is a subtype of ML in which data is transported from input to output using NNs with multiple layers, as shown in the figure 6.

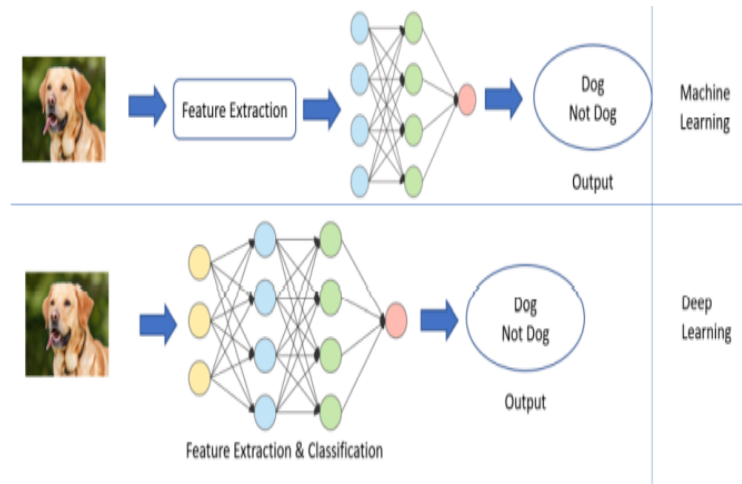


Figure 6. The difference between ML and DL (Bengio, 2009)

2.5 DEEP LEARNING MODELS

The models studied in this thesis are listed in the following sections:

2.5.1. Convolutional Neural Network

CNN is a kind of deep learning (Albawi, et al. 2017). However, the difference lies in what the network learns, how it builds, and the goal. ANN is described as a biologically-inspired programming paradigm that allows computers to learn from data. It includes many interconnected processing elements, neurons, and operations to solve problems. ANNs were tailored for a specific purpose, such as data classification or pattern recognition. Three layers are present in ANNs: the input layer, hidden layer, and output layer (Barrios, et al 2019), as shown in Figures (7) and (8).

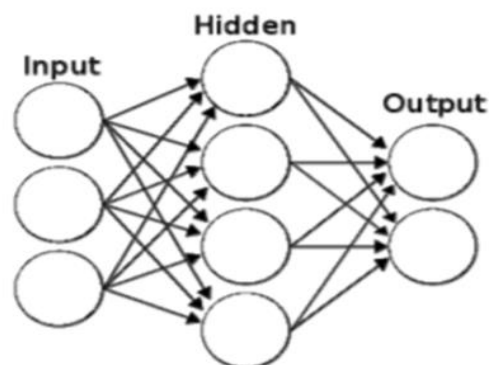


Figure 7. The basic structure of the artificial neural network (Ghosh, et al. 2014)

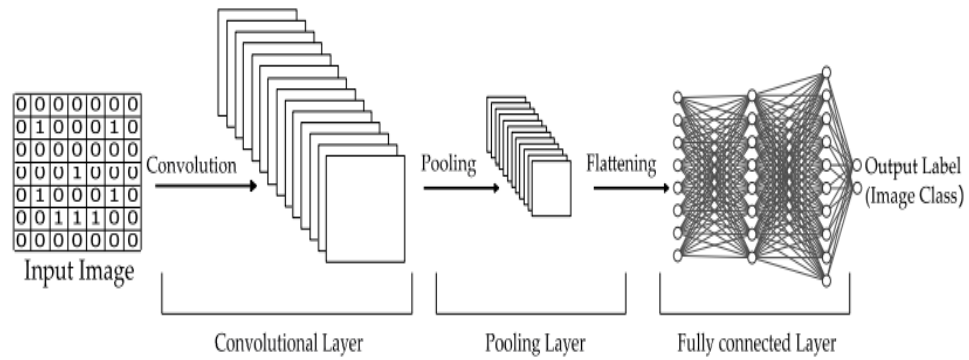


Figure 8. Architecture of a CNN (Barrios, et al. 2019)

CNNs are comparable to traditional NNs. Also, they comprise neurons with learnable biases and weights. The difference between them is in the number of layers. In the case of NNs, each input neuron is connected to the output neurons in the next layer; this is referred to as a fully-connected layer structure. However, in CNNs, convolution layers are utilized different filters in each convolution layer to extract the features and compute the outputs, which leads to local connections. then results are combined using pooling layers, pooling layer is one of the significant features of a CNN; generally, the pooling layers follow the convolutional layers. There are significant reasons for performing the pooling one feature of pooling is that it reduces the dimensionality of the features map while maintaining the important salient features (Barrios, et al. 2019); each filter might be thought of as identifying particular features. Throughout the training phase, a CNN will learn (automatically) the values related to its filters according to the task it will perform. For instance, when used for image classification, a CNN learns to detect the edges from raw pixels in the first layer. After that, the edges are used to detect simple shapes in the second layer, and then the shapes will be utilized to detect high-level features, like facial shapes, in the highest layers. The final layer is a classifier that applies high-level features (Lopez, & Yu, 2017).

2.5.2. Recurrent neural network (RNN)

RNN is considered a class of NNs, that make use of sequential information, RNNs are referred to as recurrent since they perform the same set of operations in each element as shown in figure 9. The information that the RNN has seen

previously can be memorized in the same sequence. In practice, they can only look back on a few stages, making them ineffective for modeling long-term dependencies; they also have a problem with vanishing gradients. To overcome this limitation, a specially designed RNN called a LSTM network is used in most applications; LSTMs are far less susceptible to these issues (Bishop, & Nasrabadi, 2006).

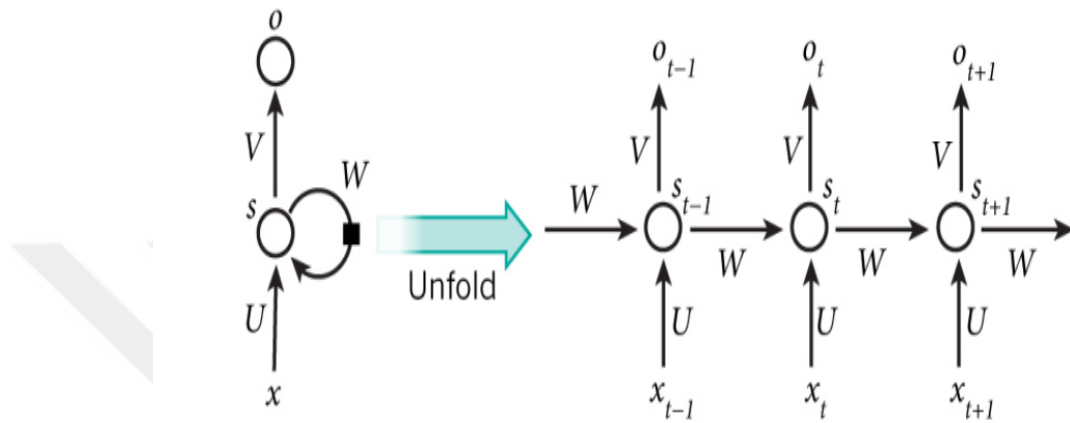


Figure 9. RNN architecture (Bhattacharyya, et al 2018)

LSTM is a recently developed of RNN that is less susceptible to the vanishing gradient issue, while it is better able to model long-distance dependencies in sequence. LSTM is designed only to memorize a portion of the sequence shown thus far. LSTM cells are depicted in figure 10. The use of gates in the network enables this behavior, as the network determines what to keep in memory based on the current input and hidden state (Li, et al 2018).

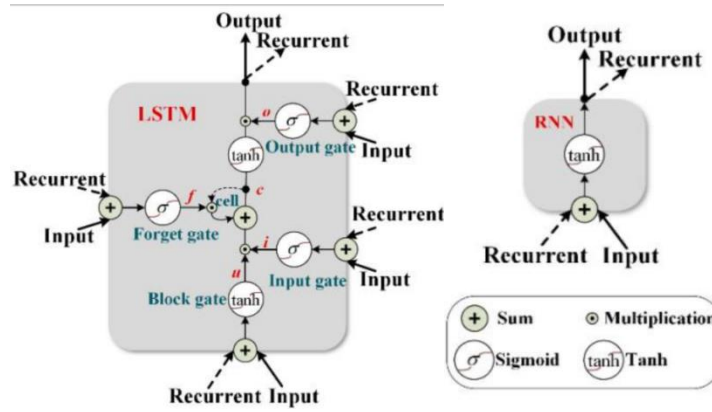


Figure 10. Architectures of LSTM cell and RNN (Li, Wang, Yang & Rong, 2018)

The equations of the LSTM below:

$$i_t = \sigma(W_i x_t + w_i h_{t-1} + b_i) \quad (4)$$

$$o_t = \sigma(W_o x_t + w_o h_{t-1} + b_o) \quad (5)$$

$$f_t = \sigma(W_f x_t + w_f h_{t-1} + b_f) \quad (6)$$

$$C_{in} = \tanh(W_c x_t + w_c h_{t-1} + b_c) \quad (7)$$

$$C_t = (i_t * C_{in} + f_t * C_{t-1}) \quad (8)$$

$$h_t = o_t * \tanh(C_t) \quad (9)$$

Where, i_t : input gate, f_t : forget gate, o_t : output gate, C_{in} : memory cell, and h_t : output (Chatterjee, Gerdes, & Martinez, 2020). The gate values are based on the current input and output of the preceding cell. This indicates that it considers both previous information and the current information for making decisions about what to keep in memory. A new cell memory is formed in Eq.(8) by forgetting the portion of the present memory and entering some new input x_t . The significance of Eq.(9) is that part of the new memory is output by the LSTM cell, while the final cell output is typically taken as the final output, representing the entire sequence. In a few applications, a variant related to LSTM has been utilized that is referred to as

peephole LSTM. The difference is that the gates function on the basis of the value of the last cell rather than the output of the previous cell (Verner, 2019).

2.6. MACHINE LEARNING:

2.6.1. K-Nearest Neighbor

The k-nearest neighbors (KNN) algorithm is a supervised machine learning approach that is straightforward to use that predicts the test point based on the surrounding training points. It is algorithm nonparametric; it saves the training data and uses it during the prediction process for the test point. The smaller the distance between two points, the more remarkable their similarity (Gou, et al 2019). The K-Nearest Neighbor approach seeks to discover the shortest path between the data to be assessed and the k closest neighbors in the training data. The (k) stands for the number of closest neighbors, choosing the correct value for K is essential to get the best accuracy, as shown in figure 11. The k value is determined after trying several values and testing the classification result each time. The KNN is an example of a technique that, despite its simplicity, outperforms other methods on big training sets.

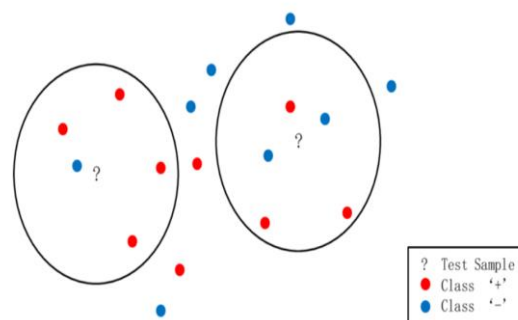


Figure 11. Example of KNN classification task with $k=5$ (Cheng, et al. 2014).

Several measures of distance can be used:

Manhattan distance: (also known as the city-mass distance) is the distance between two points calculated by summing the absolute values of line segments, each of which is parallel to a Cartesian coordinate (Ranjitkar, & Karki, 2016), as in the equation:

$$d(x, y) = \sum_{i=1}^n |x_i - y_i| \quad (10)$$

The total x and y distances are used to get a city block distance comparable to how we travel in a city (e.g. Manhattan) where you have to walk around buildings rather than go straight through.

Euclidean distance: Euclidean distance is the direct line between 2 data points that calculates the root squared difference between the coordinates (Ranjitkar, & Karki, 2016), as in the equation:

$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (11)$$

Manhattan distances as in the equation:

$$d(x, y) = \sum_{i=1}^k |x_i - y_i| \quad (12)$$

In figure 12, the green line represents the Euclidean distance, while the blue, red, and yellow lines represent Manhattan distances.

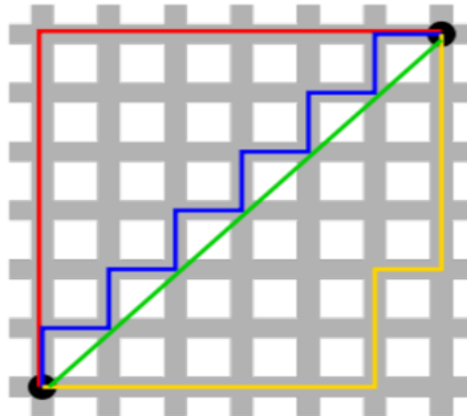


Figure 12. Euclidean distance and Manhattan distances (Ranjitkar, H. S., & Karki, S., 2016)

Minkowski distance: The Minkowski distance is the general form of the Euclidean and Manhattan distances, as in the equation:

$$d(x, y) = (\sum_{i=1}^n |x_i - y_i|^c)^{\frac{1}{c}} \quad (13)$$

Hamming distance: It is possible to compare a series of binary data using the Hamming distance statistic (Apostolico, Guerra, Landau, & Pizzi, 2016). To measure how far apart two binary strings are, the Hamming distance between them must be calculated.

2.7. TRAINING MODEL

Training is the process of teaching a NN to perform the required task such as classification; the NN training with given a labeled dataset using supervised learning finds network parameters that minimize the error rate to enable more accurate predictions. The goal is that the learned function can be used for mapping additional unknown inputs; this is called generalization, and it requires the function to model the fundamental relationship in the labeled examples from the training dataset (Nguyen, 2020).

2.7.1 Loss Function

A loss function known as the cost or objective function supports the capabilities of a ML prediction or statistical model. In a NN, the loss function quantifies the difference between the expected output and the output generated by the DL model. The output of the loss function should be minimized to obtain the best-performing ML model. There are various types of loss functions, as listed below:

Mean Squared Error (MSE): is a commonly used loss function for determining the average squared difference between actual and projected target values:

$$MSE(\hat{y}_i, y_i) = \frac{1}{N} \sum_{i=1}^N (\hat{y}_i - y_i)^2 \quad (14)$$

Here, N represents the number of training samples, \hat{y}_i represents the output generated by the deep learning model, and y_i represents the expected output. The MSE is commonly employed in conjunction with the hyperbolic tangent and linear activation functions. Furthermore, it is assumed that the errors are normally distributed.

CE (Cross-Entropy): it is another loss function that is often used in statistical models to compute the error rate; this function is described by Eq.(15):

$$CE \left(y_i, \hat{y}_i \right) = \frac{1}{N} \sum y_i \log \left(\hat{y}_i \right) \quad (15)$$

Binary Cross-Entropy (BCE): compares each of the predicted values to the real category output, which can be 0 or 1.

$$BCE \left(y_i, \hat{y}_i \right) = -\frac{1}{N} \sum_{i=1}^N \left(y_i \log \left(\hat{y}_i \right) + (1 - y_i) \log \left(1 - \hat{y}_i \right) \right) \quad (16)$$

BCE is also known as loss of sigmoid entropy. Because it is used with the sigmoid activation function, in classification error rates, CE typically leads to faster convergence and better outcomes than MSE (Rysbek, 2019).

2.7.2 Learning rate

The learning rate (α) is a tuning parameter in the optimization algorithm that selects the step size in each repetition as it moves toward the minimum loss function with a value that is typically in the range of 0 to 1. The learning rate determines how quickly the model adapts to the issue (Buduma & Locascio, 2017). Low learning rates require more training epochs, while higher learning rates require lesser training. Therefore, determining the learning rate is a challenge because too high a learning rate can cause the model to converge too quickly without reaching the goal, while too low a learning rate can cause training to stop.

2.7.3 Backpropagation

Backpropagation is a supervised learning algorithm for NN training (Rysbek, 2019). The backpropagation algorithm is used to modify the values of the weights to minimize the error rate between the predicted and the actual target values in the NN using the optimization algorithm. After randomly initializing the weights, the backpropagation algorithm consists of two steps: forward pass and backward pass; the forward pass process starts by computing the output and then ends in calculating the loss function. (Rysbek, 2019). The computed loss function is then reduced using derivatives throughout the backward pass procedure, then the weights are updated via optimization algorithms, and the forward propagation and backpropagation are repeated to increase the accuracy of NN predictions by optimizing weights. Gradient descent (GD) is one of the most commonly used optimization algorithms; it uses derivatives to follow the negative gradient of the loss function. Since this function is an activation function $y^{\wedge} = \sigma(\varepsilon)$, the chain rule is utilized for calculating the derivative of the loss function. The gradient of the loss function L , in terms of the specific weight w_i is given by the following equation:

$$\frac{\partial L}{\partial w_i} = \frac{\partial L}{\partial y^{\wedge}} \frac{\partial y^{\wedge}}{\partial \varepsilon} \frac{\partial \varepsilon}{\partial w_i} \quad (17)$$

The following equation is used to update the weight:

$$\dot{w}_i = w_i - \alpha \frac{\partial L}{\partial w_i} \quad (18)$$

Where (α) is the learning rate. The central concept of this algorithm is used also in NNs with more hidden layers and many inputs (Kimashev, 2017).

2.7.4. Optimization algorithm: Adaptive Moment Estimation (Adam)

An optimization algorithm, as previously stated, is used to update the weights. Several optimization algorithms have been introduced, such as Adam, RMSProp, Adagrad, and others (Kingma, & Ba, 2014). The Adam optimizer will train the NN model because of its rapid convergence (Duchi, Hazan, & Singer, 2011). The empirical results show that Adam works effectively in practice and compares favorably to other stochastic optimizations (Kingma, & Ba, 2014). The Adam

algorithm can be defined as an extension of the traditional stochastic gradient descent algorithm. In stochastic gradient descent, there is just one learning rate that updates all weights and remains constant during the training phase (Zheng, et al 2019). In contrast, the Adam optimization algorithm calculates adaptive learning rates for all parameters. Along with storing exponentially decaying averages regarding previously squared gradients vt such as RMSprop, Adam keeps an exponentially decaying average of the previous gradients mt , comparable to momentum as in the equation :

$$mt = \beta_1 mt - 1 + (1 - \beta_1)gt \text{ and } vt = \beta_2 vt - 1 + (1 - \beta_2)gt^2 \quad (19)$$

In which mt and vt estimate the first moment (mean) and second moment (uncentered variance), respectively, of the gradients (Ruder, 2016). The Adam algorithm has the advantages of AdaGrad, RMSProp, and momentum at the same time. The Adam algorithm updates weights according to the equation:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{vt+\epsilon}} mt \quad (20)$$

2.8 CHALLENGES IN TRAINING DEEP NETWORKS

2.8.1. Overfitting

NNs are powerful computational models used extensively; nevertheless, nonlinear hidden layers make deep networks very expressive models, prone to overfitting. The foremost issue in deep learning is overfitting (Srivastava, 2013). It happens when the model memorizes data rather than learning from it (Srivastava, 2013), if the training accuracy is much higher than the test accuracy, then it can be concluded that the model has overfitting. Several techniques have been developed that can be used in deep learning models to reduce overfitting, For example:

- Early stopping

Since it is difficult to determine when to stop training, this can lead to overfitting in the model if training does not stop at the correct point (Gençay, & Salih, 2003). The early stopping technique is an improvement technique used to

reduce overfitting without affecting model accuracy. Early stopping has been used primarily because it is simple to understand and implement. In most preliminary research regarding supervised NNs training, the purpose of this technique is to stop training before the overfitting begins, as shown in figure 13.

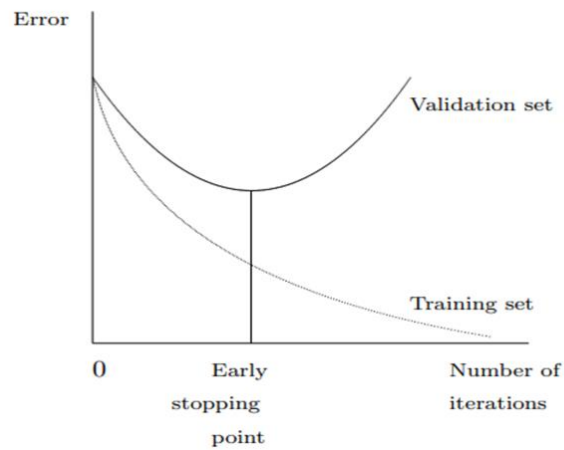
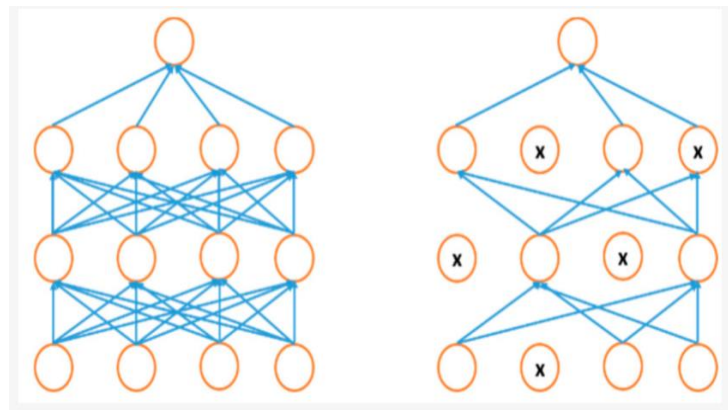


Figure 13. The early stopping technique (Gençay & Salih 2003)

- Dropout technique

DNN consists of several hidden layers, complicating learning between inputs and outputs. A dropout is the regularized technique that randomly removes neurons during the model training process (Srivastava, et al. 2014), dropping a unit out means briefly separating, it from the network, as shown in figure 14. A dropout applies to a sample in a thinned network, and the thinned comprises the considerable number of units that survived from the dropout (Alom et al. 2019). This technique allows the deep model to adapt progressively robust attributes, which help with other neurons' random subsets (Srivastava et al. 2014).



(a) Standard Neural Net

(b) After applying dropout.

Figure 14. Standard NN and after adding the dropout technique (Alom, et al. 2019)

Bernoulli's gate is the basis for the majority of DNN dropout techniques. Bernoulli distribution can be described as separate distribution, including two potential values, such as $x=0$ and $x=1$ in which $x=1$ represents success occurs that has probability P , and $x=0$ is failure occurs that has probability $1-p$, surly $0 \leq p \leq 1$. Then x is called a Bernoulli random variable, defined as $x \sim \text{Bernoulli}(p)$, where p is a probability of success. It, therefore, has a probability density function.

$$p(x) = \begin{cases} 1 - p & \text{for } x = 0 \\ p & \text{for } x = 1 \end{cases} \quad (21)$$

Example: The probability of a disappointment is named on the x -axis as 0, and achievement is labelled 1. In the accompanying Bernoulli distribution, the probability of accomplishment (1) is 0.4, and the probability of disappointment (0) is 0.6 (Starmans, et al. 2021), as shown in figure 15.

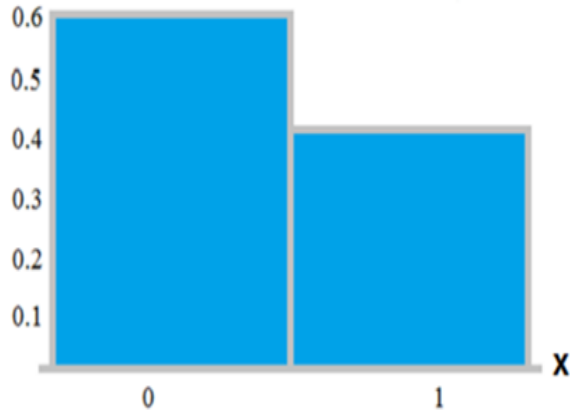


Figure 15. Bernoulli distribution (Starmans, et al. 2021).

2.8.2. Vanishing gradient

All DNNs with an activation function, such as tanh, suffer from vanishing gradient problems. The vanishing gradient makes it extremely difficult to train and update parameters in the initial network layers. Such an issue will worsen with an increase in the number of layers. Back-propagation in NNs updates the parameters to reduce the network error while the actual output is getting closer to the target (Huang, et al 2019). Throughout the back-propagation, weights update utilizing gradient descents (rates of change in total error E in terms of weights (w)). In the case of deep networks, such gradients determine how much each of the weights might change. Also, the gradients will be small as they propagate via various layers. The sigmoid function equation is:

$$f(x) = \frac{1}{1 + e^{-x}} \quad (22)$$

And the sigmoid function's derivative equation is:

$$f'(x) = \frac{1}{1 + e^{-x}} \left(1 - \frac{1}{1 + e^{-x}} \right) \quad (23)$$

2.9. TYPES OF ACTIVATION FUNCTION

The activation function is of high importance in the model architecture of ANNs. Without an activation function, a NN is essentially just a linear regression model. The activation function performs a non-linear adjustment on the input, allowing it to learn and execute more difficult tasks. The most commonly used activation functions are sigmoid functions, the rectified linear unit (ReLU), and the hyperbolic tangent (tanh) (Nwankpa, et al. 2018).

The Sigmoid Function, a widely used activation function known as the logistic function, is nonlinear. It exists in the range [0,1] as shown in figure 16, and it gives clear predictions for binary classifications.

$$\sigma(x) = \text{sigmoid}(x) = \frac{1}{1 + e^{-x}} \quad (24)$$

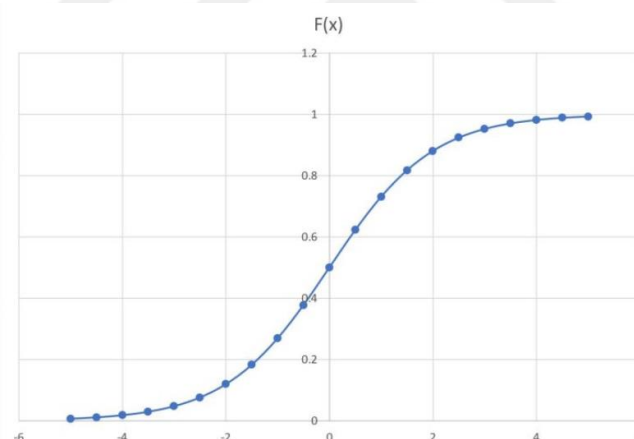


Figure 16. Sigmoid function (Bahdanau et al. 2014)

The domain of the sigmoid function is the set of all real numbers, and its range is (0, 1), as shown in figure 16. However, the vanishing of small gradient values at saturation points occurs because the sigmoid function saturates at (0) for large negative values and at (1) for large positive values.

Rectified Linear Unit (ReLU): as shown in figure 17, sets negative input values to zero (Nwankpa, et al, 2018), squashing the net input to a number larger than or equal to zero. ReLU computations are easier to calculate than those of the sigmoid function; there is no calculation of the exponential function in activations, and sparsity can be utilized. Because not all neurons are active, the network is sparse, making the function quick and efficient. The advantages of using ReLU in NNs are that it converges faster and avoids gradient vanishing.

$$\sigma(x) = \text{ReLU}(x) = \max(0, x) \quad (25)$$

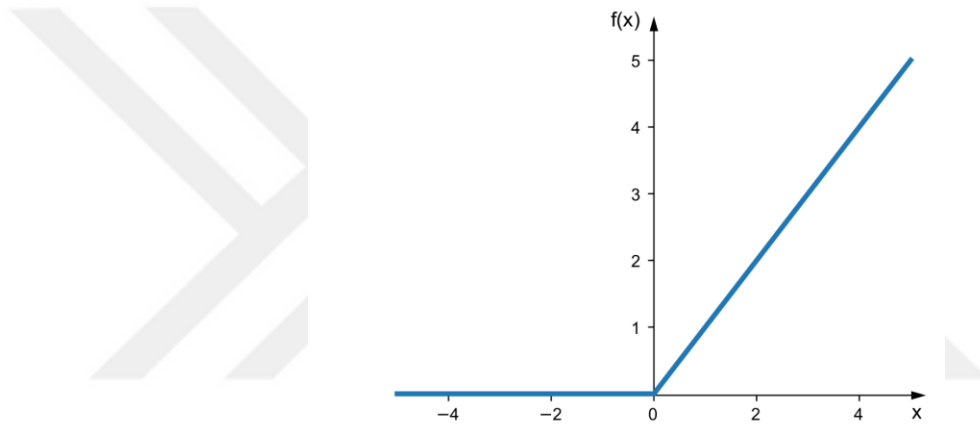


Figure 17. ReLU function (Rysbek, D. 2019)

Softmax (i.e., exponential) function: the softmax function is considered a logistic activation function that is more generalized and is utilized for multi-class classifications (Nwankpa, et al. 2018). This function is sometimes utilized in the output layer of NNs for classifications and is mathematically specified as:

$$\sigma(x_j) = \frac{e^{x_j}}{\sum_{k=1}^n e^{x_k}} \quad (26)$$

The softmax function is considered a logistic activation function that is generalized and can be utilized for multi-class classifications (Nwankpa, et al. 2018).

2.10. EVALUATION METRICS

After building the model for the classification, it is essential to evaluate and test its accuracy and quality.

2.10.1 Confusion Matrix

A confusion matrix is a technique by which model performance can be evaluated. Using the classification accuracy alone is not sufficient to evaluate a model, especially if we have more than two classes in the dataset (Čeponis & Goranin, 2020). Computing the confusion matrix gives us a better idea of how to improve our model and shows the kinds of errors that the model generates, as shown in the Table 3.

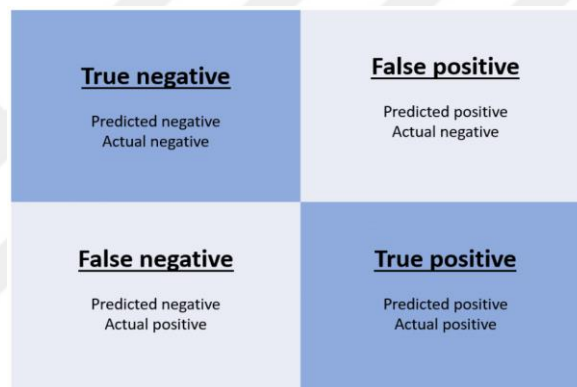
Table 3. The description parameters of the confusion matrix (Jehad & Yousif, 2020)

Parameters	Description
True Positive TP	Number of records which correctly classified
True Negative TN	Number of the correct rejection of records which have been classified
False Positive FP	The number of records incorrectly classified
False Negative FN	Number of the incorrect rejection of records which have been classified

In a confusion matrix, each row and column represent one possible classification. N stands for the number of categories. The matrix is $N \times N$ in size, with N being the number of class values. Table 4 shows a confusion matrix.

Table 4. Confusion Matrix Description (Čeponis, D., & Goranin, N. 2020)

		Predicted label	
		Benign	Malware (Intrusion)
True label	Benign	TN	FP
	Malware (Intrusion)	FN	TP



2.10.2 The receiver operating characteristic (ROC) curve

The ROC curve is a graph of sensitivity versus 1-specificity that is used to assess the diagnostic test's effectiveness (Woods & Bowyer 1997). When talking about the ROC curve, the subject is accompanied by a discussion of the area under the curve (AUC), which is an abbreviation for words that explain themselves. The area under the curve also reflects the quality of the test; the more the curve is close to the upper left corner, the more powerful, and the area under it is larger (close to 1); but, on the other hand, if the test is weak, the curve is close to the reference line and the area under it is close to 0.5, as shown in figure 18. On the ROC curve, two parameters are plotted:

- Rate of True Positives (**TPR**)

$$TPR = \frac{TP}{TP+FN} \quad (27)$$

- Rate of False Positives (**FPR**)

$$FPR = \frac{FP}{FP+TN} \quad (28)$$

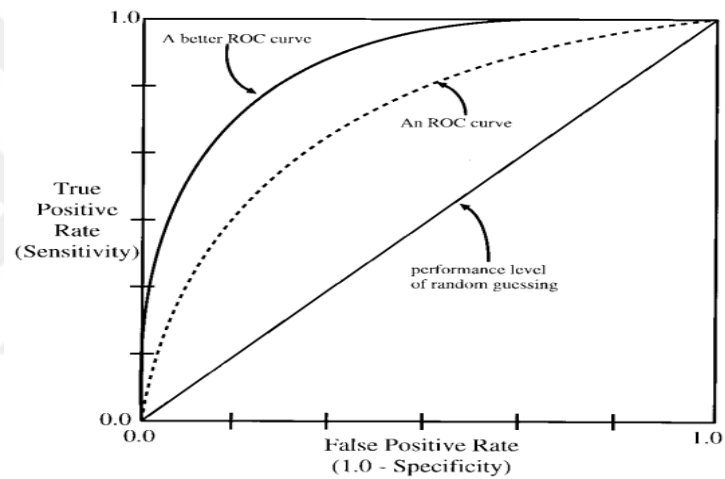


Figure 18. Roc curve (Woods & Bowyer 1997)

2.10.3. Accuracy

Accuracy is a measure that describes how the model performs in general across all classes. A classification accuracy rate is calculated by dividing the number of correct predictions by the total number (N) of predictions obtained (Gilda & Rumman, 2017) as in the equation:

$$Accuracy = \frac{TP}{N} \quad (29)$$

2.10.4. Precision/Recall

Precision and recall are metrics that allow evaluating the predicted performance of a classification model on a certain class of interest (Boyd, Eng, & Page, 2013), also known as the positive class Precision: how many are actual positive predictions from all positive predictions? Recall: how many are predicted positive from all actual positive cases? (Gilda, 2017). These concepts are expressed by the equations:

$$precision. = \frac{Tp}{Tp + Fp} \quad (30)$$

$$Recall. = \frac{Tp}{Tp + FN} \quad (31)$$

2.10.5. F1-Score

The F1-score is defined as the harmonic mean of precision and recall. It uses the following formula to combine precision and recall into a single number: It is worth noting that the F1-score considers both precision and recall, which implies it accounts for both FP and FN (Gilda, 2017).

:

$$F1 = \left(\frac{recall^{-1} + precision^{-1}}{2} \right)^{-1} = 2 \cdot \frac{Precision \cdot recall}{precision + recall} \quad (32)$$

CHAPTER THREE

METHODOLOGY

3.1. SYSTEM OVERVIEW

The data obtained from Kaggle.com was processed using the FFT technique, and then the important features of the signals were extracted by Using the data compression method and extracting the important features from it, and then used to train DL models for PD detection. The dropout method was used to prevent over-allocation in the DL models. Furthermore, Adam optimization was used to update the weights of the backpropagation method.

3.2. THE WORKFLOW

The models were constructed in five steps, as shown in Figure 19. The first step was to choose the VSB dataset from kaggle.com and preprocess the dataset using the FFT filter. After that, the spectra method was applied to extract the features from the dataset. Then, the data set was split using the hold-out method, and the classification models (CNN-KNN, CNN-LSTM, CNN, and KNN) were constructed. In the last step, the actual performance of the model was evaluated using performance metrics.



Figure 19. Model building workflow

3.3. DATASET

Kaggle.com provided the data to detect the presence of the PD patterns. This data was acquired directly from the overhead power lines with the new meter designed at ENET Centre (ENET Centre at VSB, 2019). Therefore, this is real-life data and provides a viable solution to detect PD problems.

3.3.1. Signal data

The signal data were obtained straight from the ENET Center's new meter. There are 800 000 floating-point data points in each signal; the training data contains 8 712 signals, whereas the testing data contains 20 337 signals. The signal data is stored as a **Parquet file**, with each column representing a single signal. As a result, the training signal data is $800\,000 \times 8\,712$ in size.

3.3.2. Metadata

It consists of four columns:

The signal ID: Each signal is identified by its signal ID, which is a unique number. Column '0' in the signal data corresponds to a signal ID of '0.' Three conductors carry electricity from one region to another in phase ID. Each conductor carries a signal with a different phase.

The phase: Each signal has three phases (a problem on the line may or may not affect all of the phases).

Measurement ID: the identification code for three signals recorded at the same time (The ID of different phases of the same signal is the same).

Target: this parameter indicates whether or not a signal has a PD pattern. A value of 0 indicates that the PD pattern is not present, and a value of 1 indicates that the PD pattern is present for the relevant signal ID.

$$target = \begin{cases} 0 & \text{No PD} \\ 1 & \text{PD} \end{cases} \quad (33)$$

3.4 PREPROCESSING

Signal data cannot be used because extracting any useful feature from a noisy signal is challenging. Noise removal is one of the most critical aspects of the problem. Therefore, the noise was removed from the signals by filters in our work. The FFT (Fast Fourier Transform) technique was used. This method, which detects peaks in the frequency domain, essentially removes sinusoidal noise. The FFT converts the noise-accompanied signal from the time domain to the frequency domain to separate the noise from the signal, then, by applying the inverse FFT, returns the filtered signal to the time domain; figure 20 shows a flowchart of the steps were used in the de-noising process.

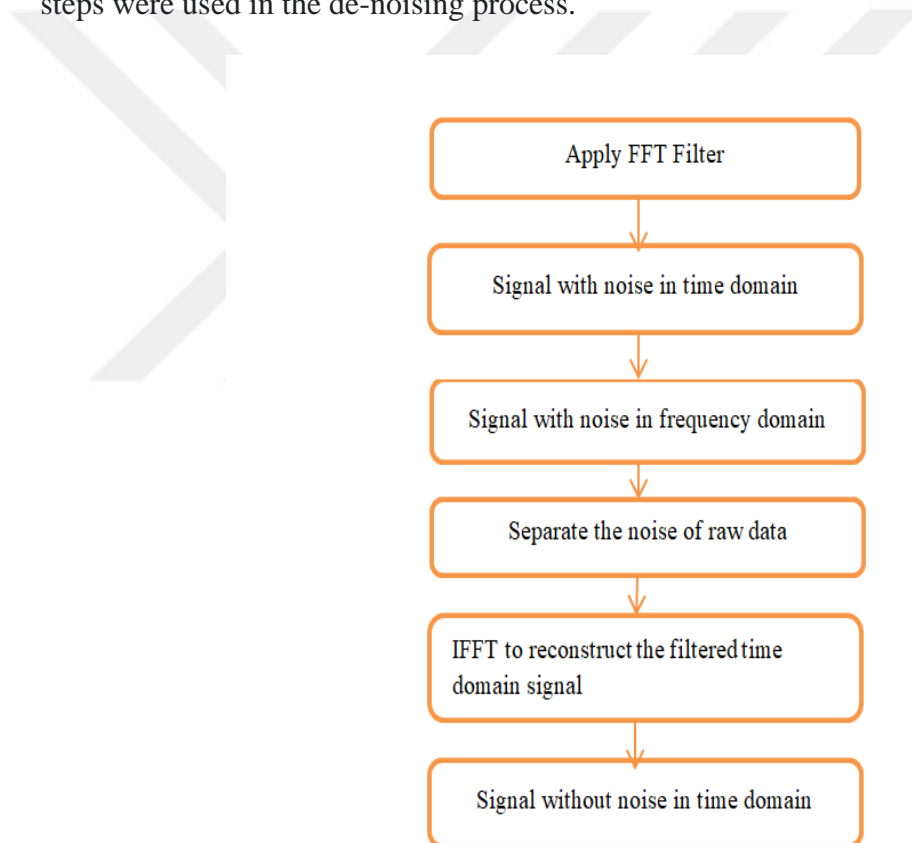


Figure 20. Denoising signals using the FFT technique

3.5. FEATURES EXTRACTION

The term "features extraction" refers to the process of converting raw data into numerical features that may be processed while retaining the information of the original dataset. It yields better results than applying machine learning and deep learning directly to the raw data. Therefore feature extraction remains the first challenge that requires significant expertise before building effective predictive models. In our work, the features extraction method was used in the following steps:

Step 1: The spectra, mean, and percentile for each signal were computed for each chunk of size 1000 (each signal was divided into equal parts with a size of 1000 for data compression).

Step2: The peak interval of width = 150 was computed which has the max deviation in the max-mean spectrum within the 800 chunks of the spectra.

Step3: The features like mean and max were extracted from the peak interval calculated above.

Step4: All three phases of a signal were merged instead of considering each phase separately.

Step5: All features extracted from signals were saved in the form of a CSV file.

3.6. CLASSIFICATION

This section will illustrate the most critical part of the proposed models. It explains how to train models and develops these NNs to help in classifying the PD with high accuracy as shown in the figure 21. The following sections explain the proposed models (CNN-KNN, hybrid CNN-LSTM, CNN, and KNN).

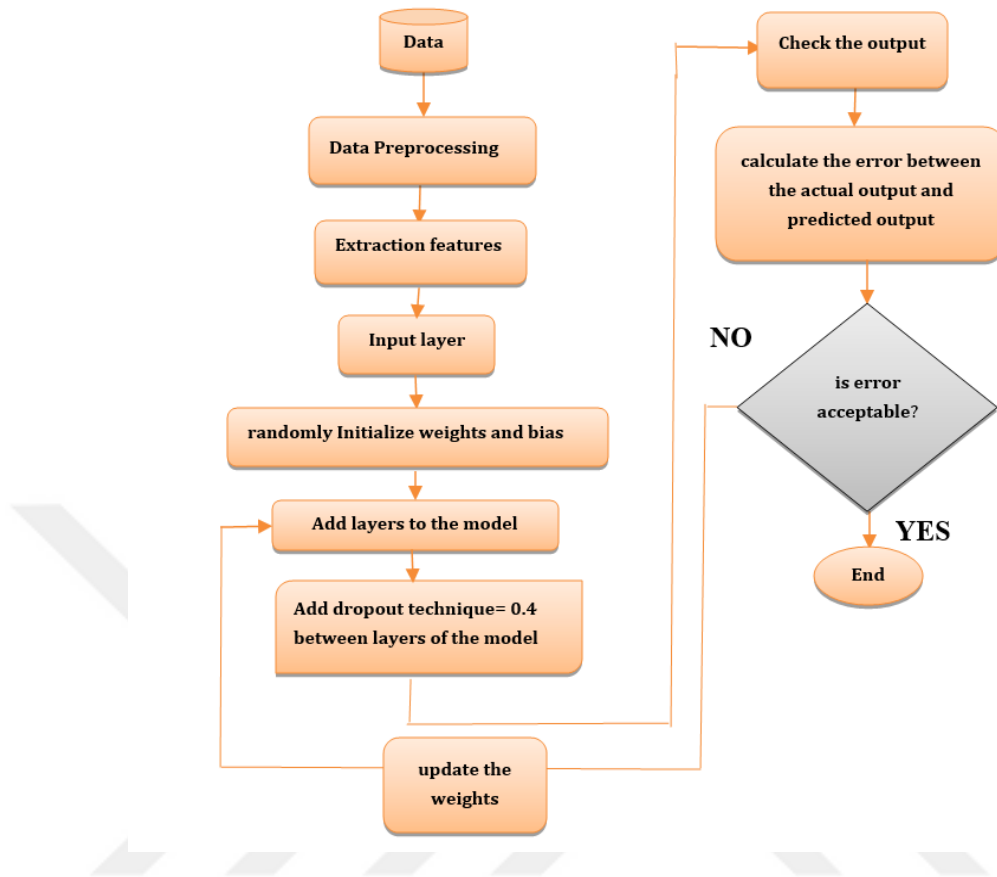


Figure 21. The general diagram of the DL proposed models

3.6.1. Construction of the CNN Model

The CNN model in Figure 22 shows an overview of the convolution NN for PD detection. After the model had been constructed, samples of signals were trained and tested in phases.

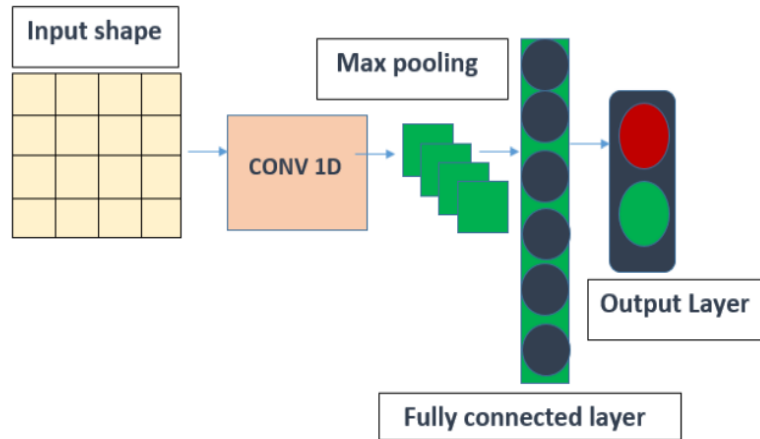


Figure 22. Construction of the CNN Model

Training phase:

After processing the data, the feature extraction method was used to extract important features from the raw dataset, and all features extracted from signals were saved in the form of a CSV file. The architecture of the model consists of an input layer followed by a convolutional layer. In the input layer, the inputs are multiplied by the weights; these weights are randomly initialized and then modified automatically during the training. In the convolutional layer, the extraction of features is implemented by passing filters over the input data; also in the convolutional layer, the ReLU function is used to convert negative numbers in feature maps to zero while preserving positive values. The resulting feature maps are then passed from the convolutional layer to the max-pooling layer. The purpose of the max-pooling layer is to reduce the size of the feature maps, reduce the number of computations, and prevent overfitting. Next, the output of the max-pooling layer is flattened to one vector and passes through a fully connected layer and the sigmoid function in the output layer is used to predict the final output results of 0 or 1. The drop-out technique is added between layers to avoid overfitting. Finally, the error between the predicted output and the actual output is calculated by using binary cross-entropy to check whether these errors are acceptable or not. The Adam optimization algorithm was used in the backpropagation step to reduce these errors and to update the weights.

Testing phase:

When building the CNN model and optimizing its fit to the training dataset, the model is evaluated on the test dataset. The evaluation is based on the correct prediction of a PD (0 or 1). The evaluation metrics were used to determine the actual performance of the final model, relying on a trial and error method that does not stop trying to choose the best hyper-parameters until success has been achieved. The following parameters were tested: several filters (64, 16, 256), learning rate (0.01, 0.001), dropout rate (0.2, 0.4, 0.5), and epochs (40, 15, 50).

3.6.2 Construction of the CNN-LSTM Model

A CNN-LSTM model in Figure 23 shows an overview of the convolution neural network and LSTM for partial discharge detection. After the model had been constructed, samples of signals were trained and tested in phases.

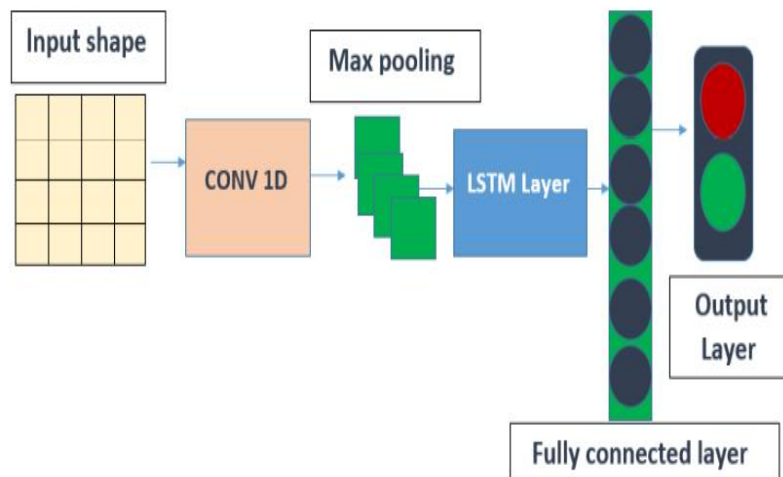


Figure 23. Construction of the CNN-LSTM Model

Training phase

This section describes the second model of classification: hybrid (CNN-LSTM). The CNN-LSTM model was trained by forwarding the (train-x) data into the model;

Step 1: FFT Technique was used to reduce the noise of the original signals.

Step 2: The spectral feature extraction method was used, and all features extracted from signals were saved in the form of a CSV file.

Step 3: The dropout technique was added between layers to prevent overfitting and the dropout was set to a probability of 0.4 when trained.

Step 4: In this step, the convolutional neural network layers were added, the first layer consisted of a convolutional layer1D with 64 filters, each of which was size 3. Each filter extracts several features with the ReLU activation function to represent them in the rectifier features map. Subsequently, the max-pooling layer was used; the reason behind choosing the maximal value is to capture the most significant feature and reduce the calculation in advanced layers, then the flatten layer to feed the next layer. The output vectors of the max-pooling layer become inputs to a fully connected layer. Finally, the sigmoid function was used in the output layer (it gives clear predictions for binary classifications).

Step 5: The layer of the LSTM has a set number of units, and the input to every cell is the output of the preceding max-pooling layer. The output vectors of the max-pooling layer become inputs to the networks of the LSTM for measuring long-term feature sequence dependencies. The output of the LSTM layer passes through a fully connected layer with sigmoid function prediction of the final output results (the sigmoid function keeps the output within a 0–1 range). A binary cross-entropy loss function was used to calculate the error to measure the distinction of the actual distributions. Finally, the error between the predicted output and the actual output is calculated by using binary cross-entropy to check whether these errors are acceptable or not. The Adam optimization algorithm was used in the backpropagation step to reduce these errors and to update the weights.

Testing phase

When building the CNN-LSTM model and optimizing its fit to the training dataset, the model is evaluated on the test dataset. The evaluation is based on the correct prediction of a PD (0 or 1). The evaluation metrics were used to determine the actual performance of the final model, relying on a trial and error method that does not stop trying to choose the best hyper-parameters until success has been achieved. The following parameters were tested: several filters (64, 16, 256), learning rate (0.01, 0.001), dropout rate (0.2, 0.4, 0.5), LSTM cell (100,200,300), and epochs (40, 15, 50). The results will be presented in the next chapter.

3.6.3. Construction of the KNN model.

This section describes the K-nearest neighbor (KNN) prediction algorithm and its steps, which do not require any training to build a model. (The data was divided into 70% training data and 30% test data.) As shown in Figure 24, these steps are followed:

Step1: The distance between the new point and the old points is calculated. The assumption is that points that are close to each other are similar, and those that are far from each other are not.

Step2: The neighbors closest to the new point are determined after selecting the value of K, to predict the class.

Step3: The final results are calculated using performance metrics.

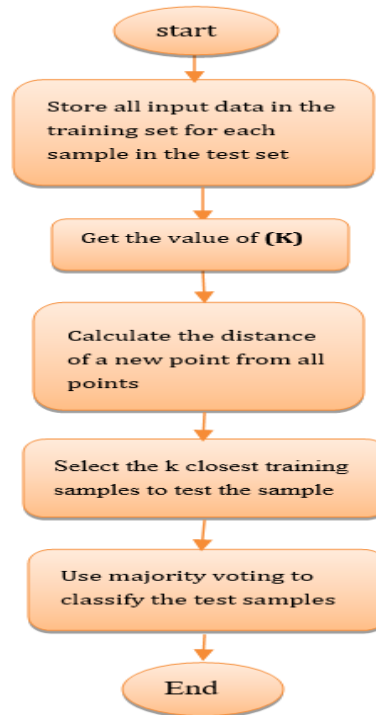


Figure 24. Steps of the KNN algorithm

3.6.4. Construction of the CNN-KNN Model

A CNN-KNN model in Figure 25 shows an overview of the convolution neural network and KNN for partial discharge detection. After the model had been constructed, samples of signals were trained and tested in phases.

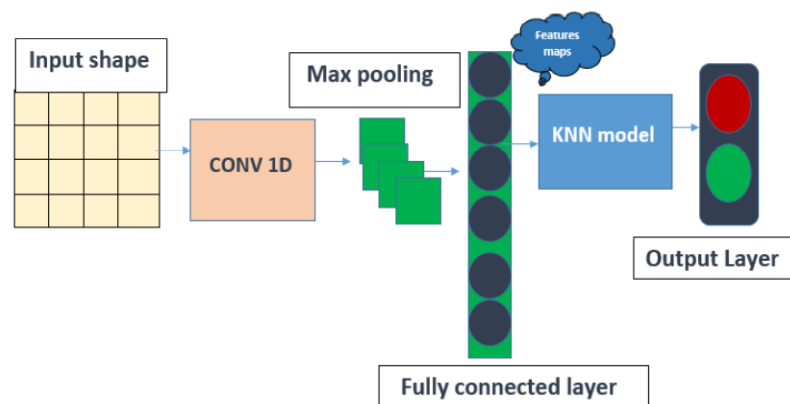


Figure 25. Construction of the CNN-KNN Model

Training phase

This section describes hybrid CNN-KNN. The CNN-KNN model was trained by forwarding the (train-x) data into the model;

Step 1: FFT Technique was used to reduce the noise of the original signals.

Step 2: The spectral feature extraction method was used, and all features extracted from signals were saved in the form of a CSV file.

Step 3: The dropout technique was added between layers to prevent overfitting and the dropout was set to a probability of 0.4 when trained.

Step 4: In this step, the convolutional neural network layers were added, the first layer consisted of a convolutional layer1D with 64 filters, each of which was size 3. Each filter extracts several features with the ReLU activation function to represent them in the rectifier features map. Subsequently, the max-pooling layer was used; the reason behind choosing the maximal value is to capture the most significant feature and reduce the calculation in advanced layers, then the flatten layer to feed the next layer. The output vectors of the max-pooling layer become inputs to a fully connected layer. Finally the sigmoid function was used in the output. The Adam optimization algorithm was used in the backpropagation step to reduce these errors and to update the weights.

Step 5: After the CNN model was built, the features extracted from the CNN model were used. Then, the extracted features were used as a dataset for the KNN algorithm by splitting the data into training and testing data to predict the category. As explained earlier, training is not required in the K-nearest neighbor (KNN) prediction algorithm.

Testing phase

When building the model and optimizing its fit to the training dataset, the model is evaluated on the test dataset. The evaluation is based on the correct prediction of a PD (0 or 1). The evaluation metrics were used to determine the actual performance of the final model, relying on a trial and error method that does not stop trying to choose the best hyper-parameters until success has been achieved: filters (64, 16, 256), dropout rate (0.2, 0.4,0.5), learning rate (0.1, 0.01, 0.001), epochs (40, 15, 50), K (3,5,8,9).

CHAPTER FOUR

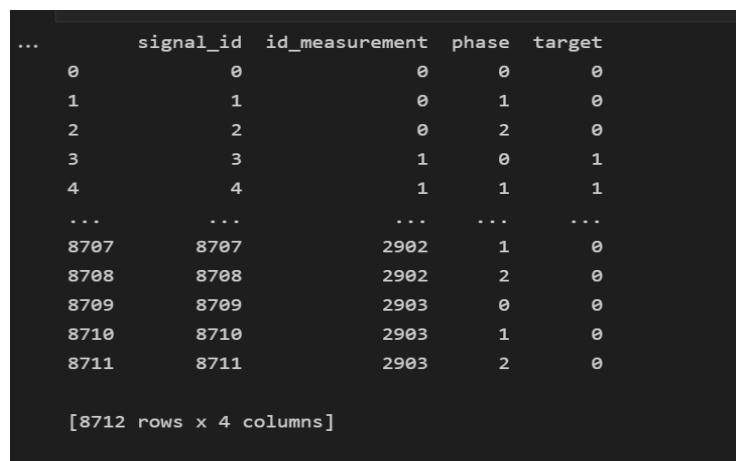
RESULTS AND DISCUSSION

4.1. GENERAL

This chapter explains the requirements needed to build the proposed partial discharge (PD) classification models. Additionally, the experimental results and evaluations for each model will be presented.

4.2. EXPERIMENTAL RESULTS OF DATASET

Electrical transmission line faults may cause a harmful phenomenon known as PD. If left unchecked, PDs may cause damage to equipment to the point that it stops working. It is essential to find any discharges to make repairs before serious harm occurs. This study used the VSB dataset (open source) created by the Technical University of Ostrava, each signal contains 800000 voltage measurements. Because the underlying electric grid works at 50 Hz, each signal spans a single grid cycle. The grid itself is a three-phase electrical system, with all three phases being monitored simultaneously. The training data contains 8712 samples, with three labels: measurement ID, phase, and target, as shown in Figure 26.



```
...      signal_id  id_measurement  phase  target
0         0         0         0         0
1         1         1         0         1
2         2         2         0         2
3         3         3         1         0
4         4         4         1         1
...      ...         ...         ...         ...
8707      8707      2902         1         0
8708      8708      2902         2         0
8709      8709      2903         0         0
8710      8710      2903         1         0
8711      8711      2903         2         0

[8712 rows x 4 columns]
```

Figure 26. Training dataset

Here the training data consists of 8187 samples of an undamaged power line (0), and 525 samples of a damaged power line (1), as shown in Figure 27.

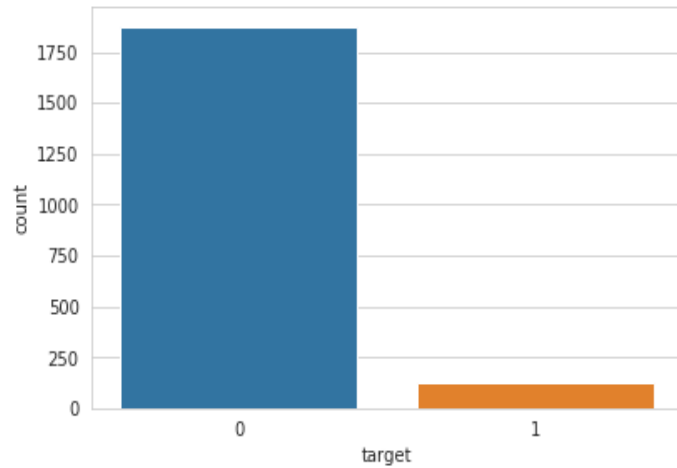


Figure 27. The distribution of the sample

The testing data contains 20337 samples, with two labels: measurement ID and phase, as shown in Figure 28.

```
signal_id  id_measurement  phase
0          8712      2904    0
1          8713      2904    1
2          8714      2904    2
3          8715      2905    0
4          8716      2905    1
...        ...        ...    ...
20332     29044      9681    1
20333     29045      9681    2
20334     29046      9682    0
20335     29047      9682    1
20336     29048      9682    2

[20337 rows x 3 columns]
```

Figure 28. Testing dataset

Nine signals from the Parquet dataset are displayed for illustration, as shown in Figure 29. Figure 30 shows the raw data for all the three phases of a signal, the Matplotlib library was used to help visualize the data and to see the data in the form of signals.

	0	1	2	3	4	5	6	7	8
0	18	1	-19	-16	-5	19	-15	15	-1
1	18	0	-19	-17	-6	19	-17	16	0
2	17	-1	-20	-17	-6	19	-17	15	-3
3	18	1	-19	-16	-5	20	-16	16	0
4	18	0	-19	-16	-5	20	-17	16	-2
...
799995	19	2	-18	-15	-4	21	-16	16	-1
799996	19	1	-19	-15	-4	20	-17	15	-3
799997	17	0	-19	-15	-4	21	-16	14	-2
799998	19	1	-18	-14	-3	22	-16	17	-1
799999	17	0	-19	-14	-4	21	-17	14	-4

800000 rows × 9 columns

Figure 29. The dataset parquet for 9 signals

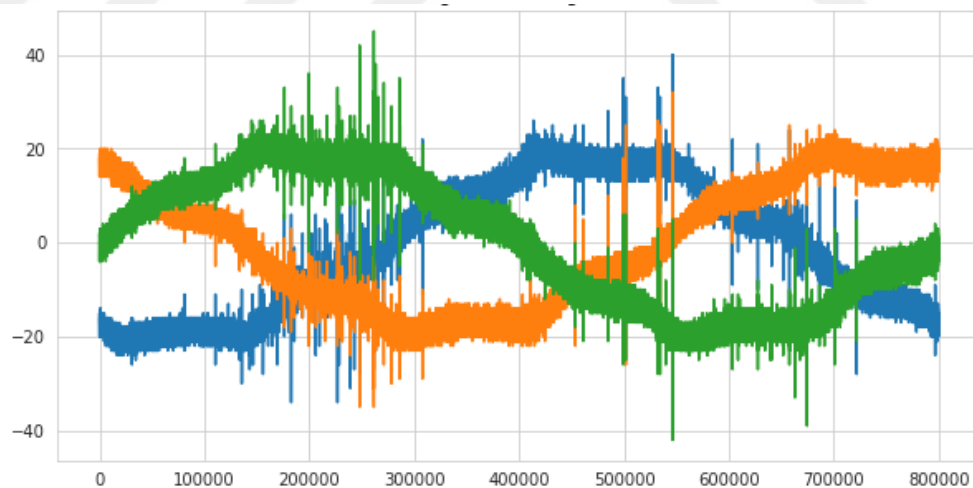


Figure 30. The raw data: 3 phases over one period

4.3 EXPERIMENTAL RESULTS OF PREPROCESSING

To overcome the noise, threshold filtering was added to the traditional FFT code. FFT converts a signal from its original time domain to its frequency domain to differentiate between the actual signal and the noise. The orange-colored curve in Fig. 31 is a filtered signal, and the blue lines are noise. The blue color in the figure represents the signal before using the filter, and the orange color represents the signal after the FFT filter use.

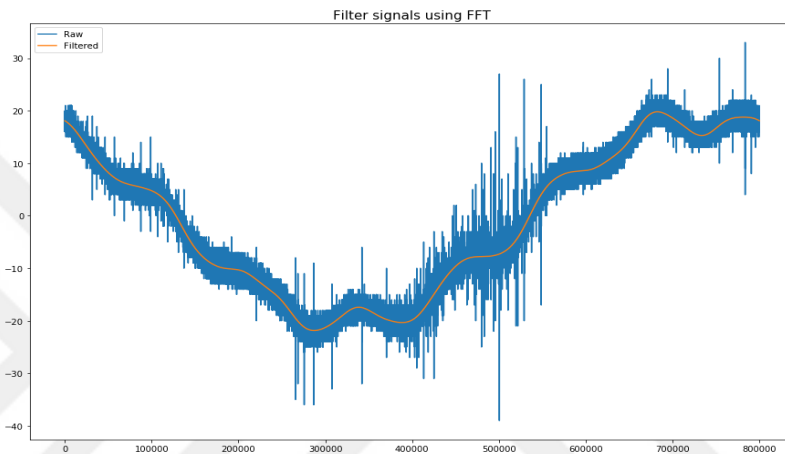


Figure 31. Result of FFT filter

Then, the filtered signal is returned to the time domain using the inverse fast Fourier transform (IFFT), as shown in Fig. 32.

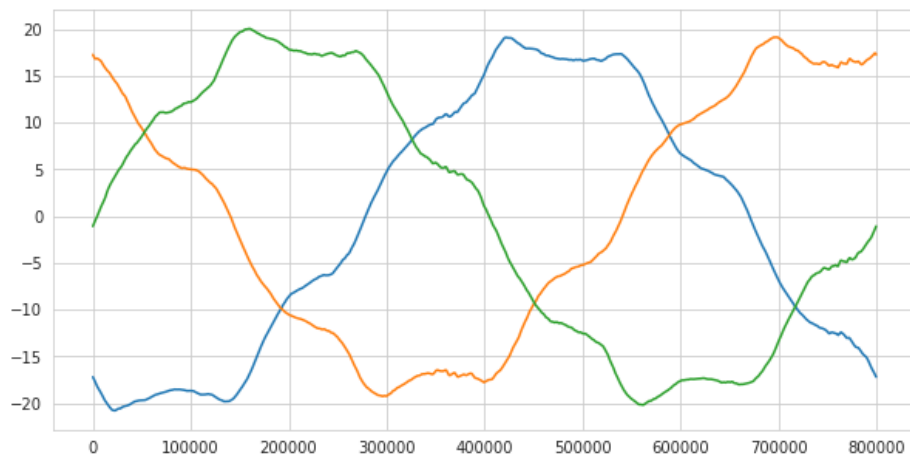


Figure 32. De-noised signals for three phases

4.4. EXPERIMENTAL RESULTS OF FEATURES EXTRACTION

After the processing step, the features from the signals through the max peak interval were extracted. The orange line in Figure 33 is the smoothed max-mean spectrum (sum of three phases); and the red band is the peak interval, i.e., the interval that gives the maximum smoothed line (orange). Next, the features in the peak interval were computed.

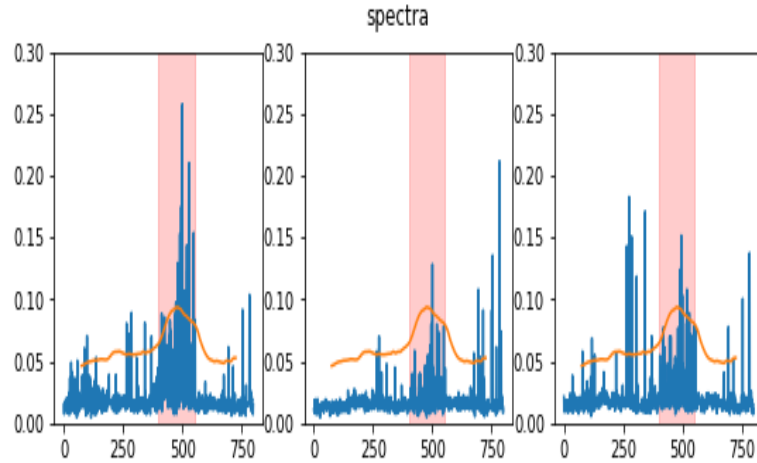


Figure 33. Features extracted from the signals

4.5. EXPERIMENTAL RESULTS OF CLASSIFICATION MODELS

The experimental setups of the classification models are listed in the following sections:

4.5.1. First Experiment (CNN Model) :

In this section, the performance of the CNN model was evaluated using evaluation metrics; the CNN model was built using one convolutional layer, one max pooling layer, a fully connected layer, and a flattening layer. The optimization was tested (Adam, SGD, **RMSPROP**), for several epochs, as shown in table 5.

Table 5. Trial and error method for choosing the best parameters

Filters	Optimization	Epoch	Batch size
64	Adam	15	64
64	RMSPROP	35	64
64	SGD	25	64

The best hyper-parameters of the CNN model are listed in Table 6. The best accuracy was obtained using filter 64, optimization algorithm Adam, and epoch 15.

Table 6 Trial and error method for choosing the best parameters

Filters	Optimization	Epoch	Accuracy
64	Adam	15	96.44%
64	RMSPROP	35	95.86%
64	SGD	25	93.92%

Figure (34) shows the history of accuracy and loss to diagnose learning issues such as underfitting or overfitting problems. The ROC curve in figure (35) shows the performance of the CNN model

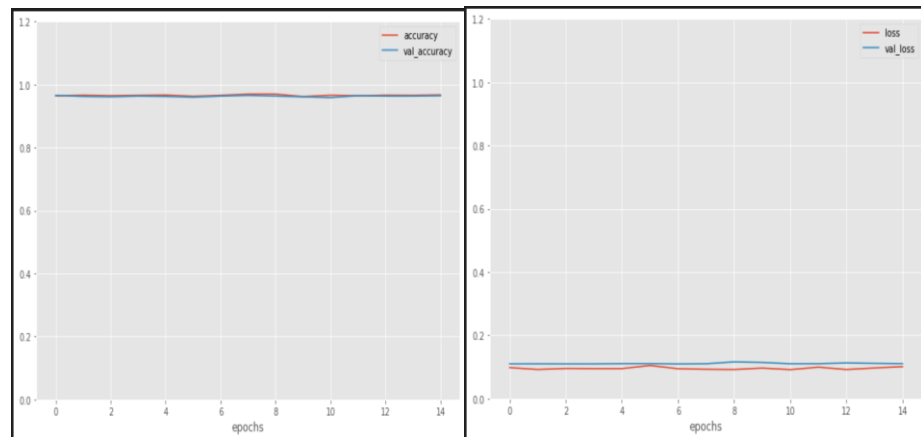


Figure 34. The history of acc and loss of CNN model

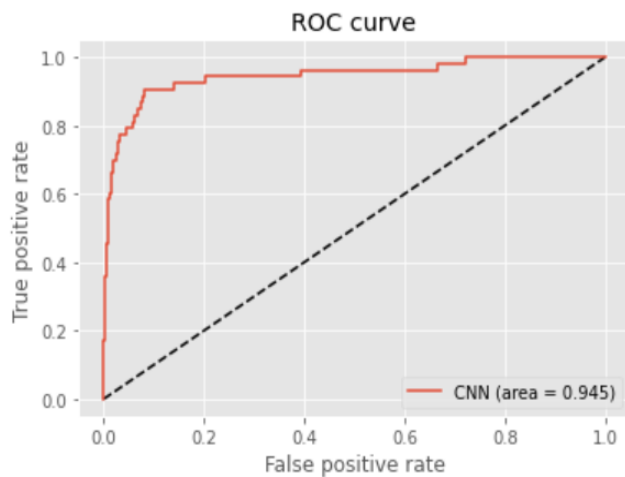


Figure 35. The Roc curve of the CNN model.

4.5.2 Second Experiment (CNN-LSTM) model

In the second experiment, the performance of the CNN-LSTM model was evaluated. The hybrid CNN-LSTM was used with one convolutional layer, a max-pooling layer, a fully connected layer, a flattening layer, and LSTM. The experiment also investigated LSTM (100), LSTM (200), and LSTM (300), as shown in table 7.

Table 7. Evaluation performance of CNN+LSTM

Classification model	Accuracy
CNN+LSTM(100)	95.76%
CNN+LSTM(200)	89.45%
CNN+LSTM(300)	78.23%

The best accuracy of the CNN-LSTM model was obtained using filter 64, Adam's optimization algorithm, epoch 15, and LSTM (100). Figure (36) shows the history of accuracy and loss, and Figure (37) shows the ROC curve of the CNN-LSTM model.

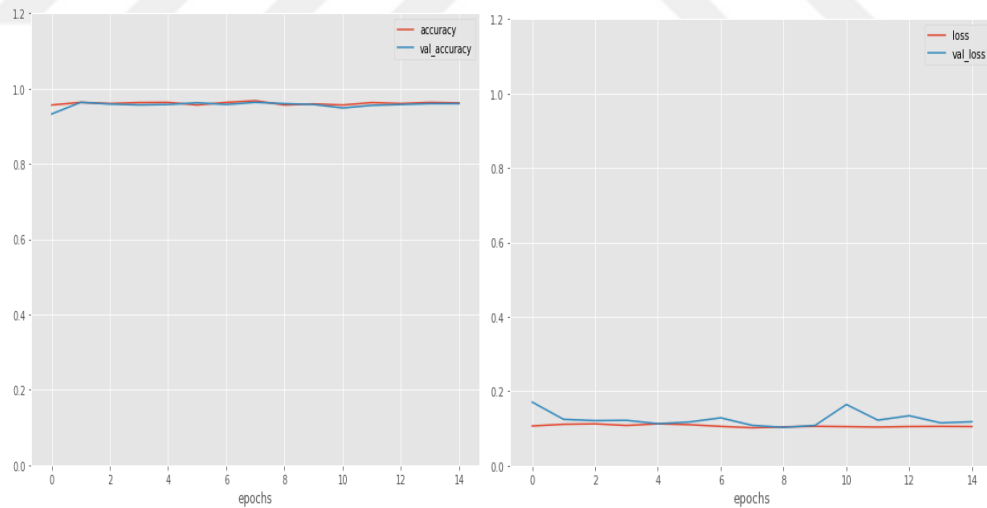


Figure 36. The history of acc and loss of the (CNN+LSTM) model

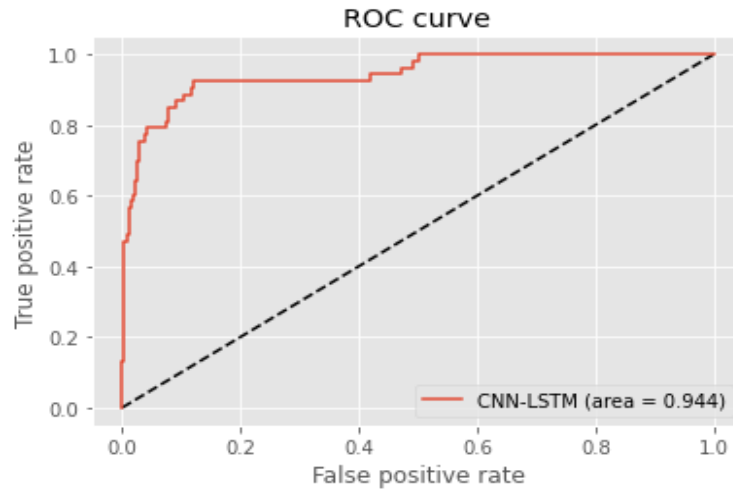


Figure 37. The Roc curve of the CNN-LSTM model

The experiment was repeated, and table (8) shows the results using two convolutional layers and two layers of max-pooling, with LSTM (100 cells).

Table 8. Evaluation performance of CNN+LSTM (2 conv layer, 2 max-pooling layer)

Classification model	Accuracy
CNN+LSTM(100)	88.55%
CNN+LSTM(200)	86.81%
CNN+LSTM(300)	73.28%

4.5.3. Third experiment (CNN-KNN and KNN) models:

In the third experiment, the hybrid model (CNN-KNN) was built using one convolutional layer, one max pooling layer, a fully connected layer, and one flat layer. After building the CNN model, the features extracted from this model were used. Then, the extracted features were used as a dataset for the KNN algorithm by splitting the data into training and testing data. Also, another experiment was conducted that used just the KNN algorithm to predict 0, or 1, as shown in table (9). The best accuracy was obtained when using $k = 3$, as shown in table (10). Figure (38) shows the **ROC** curve for the **CNN-KNN** and **KNN** models.

Table 9. Evaluation performance of CNN+KNN and KNN models

Classification model	Accuracy %
CNN +KNN	99.89%
KNN	98.74%

Table 10. Trial and error method for choosing the best parameters (CNN-KNN, KNN)

Classification model	Accuracy % K=3	Accuracy %, K=5	Accuracy %, K=9
CNN-KNN	99.98	99.20	98.67
KNN	98.74	98.12	97.66

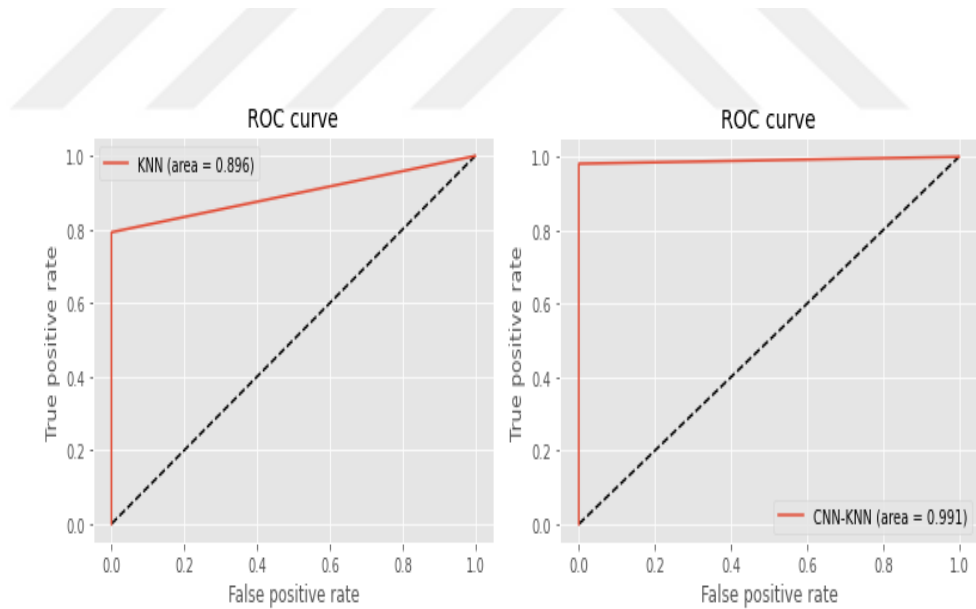


Figure 38. The Roc curve for the CNN-KNN and KNN models

4.6. EVALUATION METRICS

The evaluations metrics of the models are listed in the following sections:

4.6.1. Evaluation of the CNN, and CNN+ LSTM model

The performance of the model (CNN, and CNN-LSTM) was evaluated after 15 epochs. Figure (39) shows the confusion matrix that displays the values of TP, TN, FP, and FN; and table (11) illustrates all the results of the evaluation metrics that were used for evaluating the (CNN, and CNN-LSTM) models.

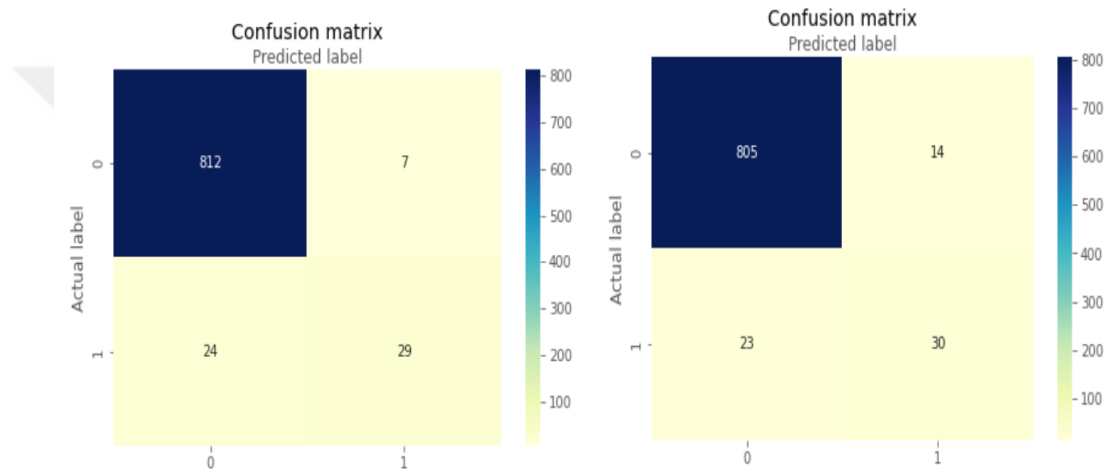


Figure 39. Confusion matrix of (CNN) and (CNN+LSTM) models after 15 epochs

Table 11. The final results of the CNN-LSTM and CNN models

Model	Accuracy %	Recall %	Precision%	F1-score %
CNN-LSTM	95.79	57	68	62
CNN	96.45	55	81	66

4.6.2 Evaluation of the KNN, and CNN+KNN model

The performance of the model (KNN, and CNN-KNN) was evaluated after 40 epochs. Figure (40) shows the confusion matrix that displays the values of TP, TN, FP, and FN; and table (12) illustrates all the results of the evaluation metrics that were used for evaluating the (KNN, and CNN-KNN) models.

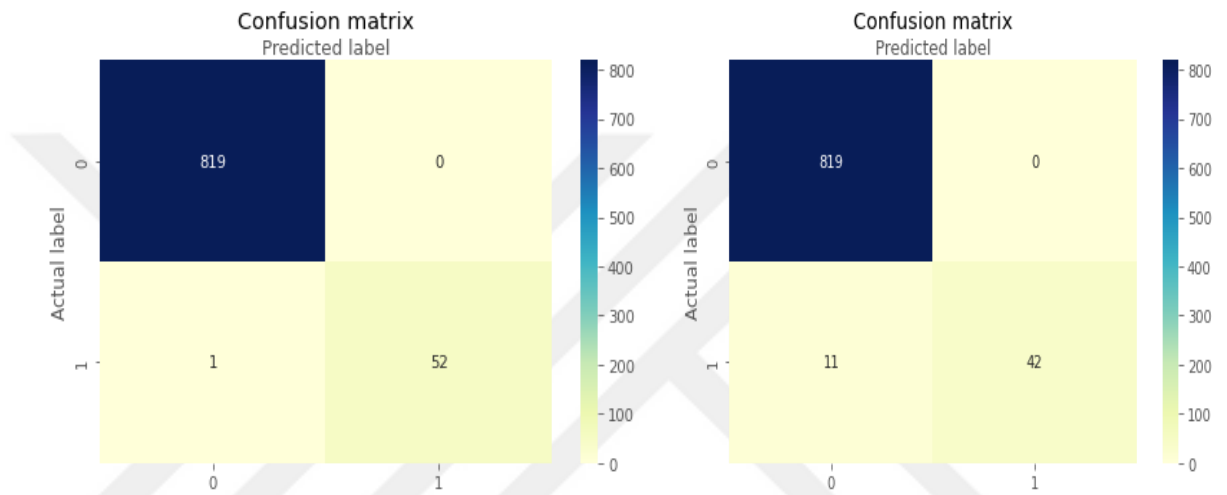


Figure 40. Confusion matrix of (CNN+KNN) and (KNN) model

Table 12. The final results of the CNN-KNN and KNN models

Model	Accuracy %	Recall %	Precision %	F1-score %
CNN- KNN	99.89	98	100	99
KNN	98.74	79	100	88

The CNN-KNN model combines the benefits of both methods (Srinivas, Sasibhushana, 2019). The benefits of CNN are the connection between neurons at consecutive layers and weight sharing among layers. Moreover, the ReLU activation function in the convolution layer reduces the complexity of the model computations

better than in the LSTM layer, which has three gates that are very complex in the computations and take a long time in their complex computations. The KNN algorithm differs from training DNNs that require numerous hyper-parameters to be set; KNN only requires one hyper-parameter (K) to be set. Moreover, because KNN does not require training on the dataset, modifications to the dataset (for example, the addition of some new samples) would not impair the performance of the algorithm. As a result, when compared to other models, the proposed model performs better.

4.7 RESULTS COMPARISON

The proposed models were compared with previous studies that used the ENET VSB dataset.

In 2021, Michau et al. used a high-pass filter to reduce the noise of the signals and a convolutional NN for PD detection; each phase was handled independently. The Adam optimization algorithm was used in this study.

In 2019, Dong et al. used two main techniques to determine PD activities on insulated overhead conductors (IOCs): **STL**, which is a technique for decomposing time series, and **SVM**, which is used as a binary classifier.

In 2020, Qu et al. used a discrete wavelet transform (**DWT**) and (**LSTM**). The original signal was first de-noised by the DWT. Then the DWT was used to decompose the de-noised signal and extract the characteristics on separate layers. Finally, the LSTM was employed to detect the problem(s) with the insulated overhead conductors (IOCs).

The proposed models were compared with previous studies regarding the classification models, size of datasets, and the accuracy in detecting PDs as shown in table 13.

Table 13. Comparison of performance of different models

Ref	Size of dataset	Classification	Accuracy	Recall	Precision	F1-score
Proposed Model 2022	800000x8712	CNN-LSTM	95.75%	57%	68%	62%
	800000x8712	CNN	96.44%	55%	81%	65%
	800000x8712	CNN- KNN	99.89%	98%	100%	99%
	800000x8712	KNN	98.74%	79%	100%	88%
Michau, et al. (2021)	800000x8712	CNN	96.7%	95.7%	72.6%	-
Dong, et al. (2019)	800000x8712	STL+SVM	-	88%	68%	77%
Qu, et al. (2020)	800000x8712	LSTM	-	81%	81%	83%

CHAPTER FIVE

CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

The proposed models were developed to detect PDs using the ENET dataset; DL and ML were employed. The FFT technique was used to reduce the noise of the original signal. After that, the spectral method was used to extract features like mean and max from the peak interval; then the model was trained to distinguish between PD and the absence of PD, and the models were validated using performance evaluation metrics.

Many points can be concluded from the results of the proposed models to classify PDs:

- The results show that the hybrid model (CNN-KNN) gives higher accuracy than the hybrid and machine learning models (CNN, CNN-LSTM, and KNN).
- The FFT technique can effectively reduce the noise of the original signal.
- The dropout technique is effective for reducing overfitting during model training.
- Adam's optimization method used with this dataset plays a significant role in improving classification accuracy.

5.2. Future Works:

- Using other classification models such as the hybrid CNN-SVM model to check their impacts on the classification accuracy.
- Using another type of dataset such as images of PD patterns.
- Using another technique to prevent overfitting to check its impact on classification accuracy.

REFERENCES

- Africa, A. D. M., Evidente, R., Piamonte, C., Sayoc, V., & Uy, J. (2020). Development of low pass filter simulation models. *International Journal of Advanced Trends in Computer Science and Engineering*, 9(4), 5894.
- Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017, August). Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)* (pp. 1-6). IEEE.
- Alom, M. Z., Aspiras, T., Taha, T. M., Asari, V. K., Bowen, T. J., Billiter, D., & Arkell, S. (2019). Advanced deep convolutional neural network approaches for digital pathology image analysis: A comprehensive evaluation with different use cases. *arXiv preprint arXiv:1904.09075*.
- Apostolico, A., Guerra, C., Landau, G. M., & Pizzi, C. (2016). Sequence similarity measures based on bounded hamming distance. *Theoretical Computer Science*, 638, 76-90.
- Babnik, T., Aggarwal, R. K., Moore, P. J., & Wang, Z. D. (2003, June). Radio frequency measurement of different discharges. In *2003 IEEE Bologna Power Tech Conference Proceedings*, (Vol. 3, pp. 5-pp). IEEE.
- Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, Accepted at ICLR 2015 as oral presentation.
- Barrios, S., Buldain, D., Comech, M. P., Gilbert, I., & Orue, I. (2019). Partial discharge classification using deep learning methods—Survey of recent progress. *Energies*, 12(13), 2485
- Bekdas, G., Nigdeli, S. M., & Yucel, M. (2020). *Artificial Intelligence and Machine Learning Applications in Civil, Mechanical, and Industrial Engineering*. IGI Global.
- Bengio, Y. (2009). *Learning deep architectures for AI*. Now Publishers Inc.
- Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8), 1798-1828.
- Bhattacharyya, V. Sasel, A. E. Hassanien, S. Saha, and B. K. Tripathy, 2018, "Deep learning", IEEE, vol.3, pp.122-124
- Bishop, C. M., & Nasrabadi, N. M. (2006). *Pattern recognition and machine learning* (Vol. 4, No. 4, p. 738). New York: Springer.
- Boyd, K., Eng, K. H., & Page, C. D. (2013, September). Area under the precision-recall curve: point estimates and confidence intervals. In *Joint European conference on machine learning and knowledge discovery in databases* (pp. 451-466). Springer, Berlin, Heidelberg

- Buduma, N., & Locascio, N. (2017). *Fundamentals of deep learning: Designing next-generation machine intelligence algorithms*. " O'Reilly Media, Inc."
- Catterson, V. M., & Sheng, B. (2015, June). Deep neural networks for understanding and diagnosing partial discharge data. In 2015 IEEE Electrical Insulation Conference (EIC) (pp. 218-221). IEEE.
- Čeponis, D., & Goranin, N. (2020). Investigation of dual-flow deep learning models LSTM-FCN and GRU-FCN efficiency against single-flow CNN models for the host-based intrusion and malware detection task on univariate times series data. *Applied Sciences*, 10(7), 2373.
- Chatterjee, A., Gerdes, M. W., & Martinez, S. G. (2020). Statistical explorations and univariate timeseries analysis on COVID-19 datasets to understand the trend of disease spreading and death. *Sensors*, 20(11), 3089
- Cheng, D., Zhang, S., Deng, Z., Zhu, Y., & Zong, M. (2014, December). kNN algorithm with data-driven k value. In *International Conference on Advanced Data Mining and Applications* (pp. 499-512). Springer, Cham.
- Cochran, W. T., Cooley, J. W., Favin, D. L., Helms, H. D., Kaenel, R. A., Lang, W. W., ... & Welch, P. D. (1967). What is the fast Fourier transform?. *Proceedings of the IEEE*, 55(10), 1664-1674.
- Dong, M., Sun, Z., and Wang, C. (2019, May). A pattern recognition method for partial discharge detection on insulated overhead conductors. In 2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE) (pp. 1-4). IEEE.
- Duan, L., Hu, J., Zhao, G., Chen, K., He, J., and Wang, S. X. (2019). Identification of partial discharge defects based on deep learning method. *IEEE Transactions on Power Delivery*, 34(4), 1557-1568
- Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7).
- ENET Centre at VSB—Technical University of Ostrava. VSB Power Line Fault Detection 2019. Available online: <https://www.kaggle.com/c/vsb-power-line-fault-detection>
- Gençay, R., & Salih, A. (2003). Degree of mispricing with the Black-Scholes model and nonparametric cures. *Economics and Finance. Annals*, 4, 73-101.
- Ghosh, T., Sengupta, S., Doloi, B., & Dan, P. K. (2014). AI-based techniques in cellular manufacturing systems: a chronological survey and analysis. *International Journal of Industrial and Systems Engineering*, 17(4), 449-476.
- Gilda, S. (2017, December). Notice of Violation of IEEE Publication Principles: Evaluating machine learning algorithms for fake news detection. In 2017 IEEE 15th student conference on research and development (SCOREd) (pp. 110-115). IEEE.

- Gou, J., Ma, H., Ou, W., Zeng, S., Rao, Y., & Yang, H. (2019). A generalized mean distance-based k-nearest neighbor classifier. *Expert Systems with Applications*, 115, 356-372.
- Hanninen, S., Lehtonen, M., & Hakola, T. (2002). Earth faults and related disturbances in distribution networks. *IEE Proceedings-Generation, Transmission and Distribution*, 149(3), 283-288.
- Hashmi, G. M. (2008). Partial discharge detection for condition monitoring of covered-conductor overhead distribution networks using Rogowski coil. Teknillinen korkeakoulu.
- Huang, K., Hussain, A., Wang, Q. F., & Zhang, R. (Eds.). (2019). *Deep learning: fundamentals, theory and applications (Vol. 2)*. Springer.
- Jehad, R., & Yousif, S. A. (2020). Fake News Classification Using Random Forest and Decision Tree (J48). *Al-Nahrain Journal of Science*, 23(4), 49-55.
- Kimashev, A. (2017). *Deep Learning for text data mining: Solving spreadsheet data classification (Master's thesis, University of Stavanger, Norway)*
- Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980. Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015
- Kong, W., Men, K., Guo, Z., Dong, Z. Y., Zhang, R., & Jia, Y. (2020, February). Automatic Online Partial Discharge Diagnosis via Deep Learning. In *2020 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)* (pp. 1-5). IEEE
- Li, G.; Wang, X.; Li, X.; Yang, A.; Rong, M. (2018) Partial Discharge Recognition with a Multi-Resolution Convolutional Neural Network. *Sensors* 2018, 18, 3512
- Lopez, M., & Yu, W. (2017, October). Nonlinear system modeling using convolutional neural networks. In *2017 14th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)* (pp. 1-5). IEEE.
- Lu, Y.; Wei, R.; Chen, J.; Yuan, J. Convolutional Neural Network Based Transient Earth Voltage Detection. In *Proceedings of the 2016 15th International Symposium on Parallel and Distributed Computing (ISPDC)*, Fuzhou, China, 8–10 July 2016; pp. 386–389.
- Majidi, M., Fadali, M. S., Etezadi-Amoli, M., & Oskuoee, M. (2015). Partial discharge pattern recognition via sparse representation and ANN. *IEEE Transactions on Dielectrics and Electrical Insulation*, 22(2), 1061-1070.
- McNamara, D. E., & Boaz, R. I. (2006). *Seismic noise analysis system using power spectral density probability density functions: A stand-alone software package*. Reston, Virginia: US Geological Survey.

- Metwalli, M. R., Nasr, A. H., Allah, O. S. F., & El-Rabaie, S. (2009, December). Image fusion based on principal component analysis and high-pass filter. In 2009 International Conference on Computer Engineering & Systems (pp. 63-70). IEEE.
- Michau, G., Hsu, C. C., & Fink, O. (2021). Interpretable detection of partial discharge in power lines with deep learning. *Sensors*, 21(6), 2154.
- Ghahramani, 2016, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning", in international conference on machine learning, pp. 1050-1059.
- Nguyen, J., 2020, "Using Deep Learning and Linguistic analysis to Predict to predict fake news within text fake news within text ", vol 2, pp. 1137-1145.
- Nguyen, M.-T.; Nguyen, V.-H.; Yun, S.-J.; Kim, Y.-H. (2018) Recurrent Neural Network for Partial Discharge Diagnosis in Gas-Insulated Switchgear. *Energies* 2018.
- Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). Activation functions: Comparison of trends in practice and research for deep learning. arXiv preprint arXiv:1811.03378.
- Qu, N., Li, Z., Zuo, J., & Chen, J. (2020). Fault detection on insulated overhead conductors based on DWT-LSTM and partial discharge. *IEEE Access*, 8, 87060-87070.
- Ranjitkar, H. S., & Karki, S. (2016). Comparison of A*, Euclidean and Manhattan distance using Influence map in MS. Pac-Man., Faculty of Computing Blekinge Institute of Technology SE-371 79 Karlskrona Sweden.
- Ruder, S. (2016). An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747.
- Rysbek, D. (2019). Sentiment analysis with recurrent neural networks on turkish reviews domain (Master's thesis, Middle East Technical University).
- Samaitis, V., Mažeika, L., Jankauskas, A., & Rekuviene, R. (2020). Detection and localization of partial discharge in connectors of air power lines by means of ultrasonic measurements and artificial intelligence models. *Sensors*, 21(1), 20.
- Samarkhanov, K., Abuduwaili, J., Samat, A., & Issanova, G. (2019). The Spatial and Temporal Land Cover Patterns of the Qazaly Irrigation Zone in 2003–2018: The Case of Syrdarya River's Lower Reaches, Kazakhstan. *Sustainability*, 11(15), 4035,
- Santoso, S., Powers, E. J., & Grady, W. M. (1997). Power quality disturbance data compression using wavelet transform methods. *IEEE Transactions on Power Delivery*, 12(3), 1250-1257.
- Schmidhuber, 2015, " Deep Learning in Neural Networks: An Overview. *Neural Netw*", 61, 85–117. <http://www.elsevier.com/locate/neunet>.

- Shafiq, M.; Hussain, G.A.; Kütt, L.; Lehtonen, M. Effect of geometrical parameters on high frequency performance of Rogowski coil for partial discharge measurements. *Measurement* 2014, 49, 126–137.
- Srinivas B., Sasibhushana Rao, 2019, A Hybrid CNN-KNN Model for MRI brain Tumor Classification, *International Journal of Recent Technology and Engineering (IJRTE)*.
- Sriram, S., Nitin, S., Prabhu, K. M. M., & Bastiaans, M. J. (2005). Signal denoising techniques for partial discharge measurements. *IEEE Transactions on Dielectrics and Electrical Insulation*, 12(6), 1182-1191.
- Srivastava, N. (2013). Improving neural networks with dropout. *University of Toronto*, 182(566), 7
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1), 1929-1958.
- Starmans, M., van der Voort, S. R., Phil, T., Timbergen, M. J., Vos, M., Padmos, G. A., ... & Klein, S. (2021). Reproducible radiomics through automated machine learning validated on twelve clinical applications. *arXiv preprint arXiv:2108.08618*.
- Stone, G. C. (2013). Condition monitoring and diagnostics of motor and stator windings—A review. *IEEE Transactions on Dielectrics and Electrical Insulation*, 20(6), 2073-2080.
- Tang, J.; Jin, M.; Zeng, F.; Zhang, X.; Huang, R. 2016 Assessment of PD severity in gas-insulated switchgear with an SSAE. *IET Sci., Meas. Technol.* 2017, 11, 423–430.
- Verner, 2019, "LSTM Networks for Detection and Classification of Anomalies in Raw Sensor Data ", *Journal of Machine Learning*, 2, 21. Volume 2, Issue 1, July 2012.
- Wang, X.; Huang, H.; Hu, Y.; Yang, Y. Partial Discharge Pattern Recognition with Data Augmentation based on Generative Adversarial Networks. In *Proceedings of the 2018 Condition Monitoring and Diagnosis (CMD)*, Perth, WA, Australia, 23–26 September 2018; pp. 1–4.
- Woods, K., & Bowyer, K. W. (1997). Generating ROC curves for artificial neural networks. *IEEE Transactions on medical imaging*, 16(3), 329-337.
- Wu, M.; Cao, H.; Cao, J.; Nguyen, H.L.; Gomes, J.B. An overview of state-of-the-art partial discharge analysis techniques for condition monitoring. *IEEE Electr. Insul. Mag.* 2015, 31, 22–35.
- Zheng, Q., Tian, X., Jiang, N., & Yang, M. (2019). Layer-wise learning based stochastic gradient descent method for the optimization of deep convolutional neural network. *Journal of Intelligent & Fuzzy Systems*, 37(4), 5641-5654.