DIGITAL SIGNAL PROCESSING AND MACHINE LEARNING APPLIED TO POWER LINE COMMUNICATIONS

by

Kushal Thapa, B.S.

A thesis submitted to the Graduate Council of
Texas State University in partial fulfillment
of the requirements for the degree of
Master of Science
with a Major in Engineering
August 2021

Committee Members:

Stan McClellan, Chair

Damian Valles

Semih Aslan

Andres Carvallo

COPYRIGHT

by

Kushal Thapa

2021

FAIR USE AND AUTHOR'S PERMISSION STATEMENT

Fair Use

This work is protected by the Copyright Laws of the United States (Public Law 94-553, section 107). Consistent with fair use as defined in the Copyright Laws, brief quotations from this material are allowed with proper acknowledgement. Use of this material for financial gain without the author's express written permission is not allowed.

Duplication Permission

As the copyright holder of this work I, Kushal Thapa, authorize duplication of this work, in whole or in part, for educational or scholarly purposes only.

ACKNOWLEDGEMENTS

I would like to thank Dr. McClellan, for constantly supporting me from the very first day I started working for him, believing in my abilities to provide me research opportunity in his lab, and trusting me in leading this thesis study. I am grateful for the time he has spent in guiding me through this research, and my graduate career as a whole, for the last two years. I would also like to thank Dr. Valles, first, for his class on 'Machine Learning' that provided me the inspiration and the technical skills needed for this study, and then, for his time and contributions beyond the class. I thank Dr. Aslan and Mr. Carvallo as well, for their input in this research. I also wish to thank Dr. Viswanathan for his continued guidance in my graduate career and life in general. His advice and sincere support throughout these last two years has made me feel right at home at Texas State.

I deeply appreciate my family- my wife, my parents back home, and my brother and sister- for their unwavering faith in me and their support in my academic ventures. Without them, none of this would be possible. Many thanks to my friends and relatives as well for their continued support in my life, I appreciate you all.

Finally, I want to acknowledge two people close to me who lost their life recently to Covid- Bikram dai and Bikash mama. I would not be who I am today without their love, support, and their inspiring life. Thank you.

TABLE OF CONTENTS

Page

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

LIST OF ABBREVIATIONS

Abbreviation	Description
AFRC	Audio Frequency Ripple Control
ANN	Artificial Neural Network
ASK	Amplitude Shift Keying
BER	Bit Error Ratio
BPL	Broadband over Power Line
CNN	Convolutional Neural Network
CT	Current Transformer
DAQ	Data Acquisition device
DNN	Deep Neural Network
DSP	Digital Signal Processing
FCC	Federal Communications Commission
FFT	Fast Fourier Transform
FSK	Frequency Shift Keying
HV	High Voltage
IG	Information Gain
LAN	Local Area Network
LR	Logistic Regression
LV	Low Voltage
ML	Machine Learning

MLP	Multi-Layer Perceptron
MV	Medium Voltage
NB-PLC	Narrow-Band Power Line Communications
OFDM	Orthogonal Frequency Division Multiplexing
OOK	On-Off Keying
PLC	Power Line Communications
PLM	Power Line Modem
PSK	Phase Shift Keying
QAM	Quadrature Amplitude Modulation
RMSE	Root Mean Square Energy
SBS	Sequential Backward Analysis
SNR	Signal to Noise Ratio
STFT	Short-Time Fourier Transform
SVM	Support Vector Machine
TREE	Decision Tree
ULF-PLC	Ultra-Low Frequency Power Line Communications

ABSTRACT

Power Line Communications (PLC) is a technology that uses power lines to transport communication data alongside the electric power signals. Due to the ubiquitous nature of pre-existing power grid infrastructure, PLC has a huge networking potential, especially in the implementation of smart grid technologies. However, the electrical architecture and function of distribution grid systems, which is specifically designed to carry power signals, poses a major hindrance to communication signals. This hindrance typically takes the form of poor signal propagation. Traditional signal processing measures may be neither sufficiently adaptable nor optimally effective in recovering communication signals at the receiver end. To overcome this challenge, this research investigates the use of machine learning techniques as a supplement to the traditional digital signal processing techniques. We focus on testing and comparing various supervised machine learning and deep learning algorithms for the purpose of signal demodulation and bit classification in ultra-low-frequency, baseband PLC systems operating in the electrical distribution grid.

Keywords: power line communications, ULF-PLC, machine learning, ML, smart grid

1. INTRODUCTION

1.1. Background and Motivation

The use of electrical wiring and power lines for network communication is not new. Since the early 1920s, this technology, commonly referred to as Power Line Communications (PLC), has been used to automate meter reading by utility companies [1]. Beyond this application, the potential of PLC was conceptualized as a universal networking solution mainly because of the pre-existing power-grid [2]. This power grid would obviate the need for building other types of dedicated communication infrastructures like phone lines and optical fibers, thereby saving billions in cost [2]. However, over the years, such high expectations of this technology have not been realized due to many factors. One of the primary culprits is signal propagation.

From generation in power plants to its use in a load (and loopback), the low-frequency power signal (50 Hz or 60 Hz depending on the country of origin) goes through many levels of voltage and current transformation. These transformations are facilitated by different electrical devices such as aptly-named transformers. Most of these devices were made specifically for power signals, and thus, pose propagation problems for higher-frequency signals traveling through the power lines [3]. This problem can be circumvented by excluding these devices altogether by bounding the PLC circuit within a "level" of the power grid. In this "intra-level" PLC, the system voltage and other properties of the power signal are the same throughout the path of the communication signal. Intra-level PLC has been developed into many applications such as powerline Local Area Network (LAN) [4] and baby monitors [5]. This solution, however, does not address the problem of signal propagation through the different levels of the power grid

i.e., “inter-level” PLC.

One existing solution to this propagation problem in inter-level PLC systems is the use of additional equipment like signal repeaters. These repeaters filter and catch the communication signal in one level and pass it to the next level so that the integrity of the signal is preserved [6]. Broadband over Power Line (BPL) technology, which is a duplex and high data-rate application of the inter-level PLC, is reliant on such devices [6]. However, the primary disadvantage of this solution is the space and cost associated with developing, deploying, and maintaining these devices. Another approach, which does not require the additional devices, is the transmission of ultra-low-frequency signals. Ultra-low-frequency signals do not attenuate as much compared to higher frequency signals, even after passing through distribution transformers, which have a fairly limited passband. Thus, Ultra-Low-Frequency (ULF) signals, typically in the range of 150 Hz-1350 Hz [7], can be injected into the power lines for PLC. This type of PLC is henceforth referred to as ULF-PLC in this paper. In fact, this approach is the main principle behind a simplex, low data-rate PLC application called “Audio Frequency Ripple Control” (AFRC) that has been used in Europe for many decades for load peak shaving, street lighting, etc. [7]. However, one major disadvantage of using ULF bands for communication is the low data rate. Consequently, this solution has been under-researched and mostly overlooked.

Nonetheless, ULF-PLC has applications in fields that do not require high-speed data transmission but prioritize reliability, simplicity, scalability, and ease of deployment. There is an urgent need for this kind of technology in the power sector. The existing power grids are failing because of the exponential increase in power demand over the last

few decades [8] . This problem is exacerbated by the disconnect between the power producers and consumers which leads to a huge waste of already depleted power supply. ULF-PLC can bridge this disconnect thereby becoming a valuable element in the communication infrastructure of a continuously sensing and self-monitoring power grid called “smart grid.”

ULF applications like AFRC employ a “downstream” communication, i.e., the directionality of signal is from High Voltage (HV) or Medium Voltage (MV) tier to Low Voltage (LV) tier. A different approach to communication is an “upstream” method, which injects a communication signal at the end-node in the LV tier of the distribution grid and extracts it from the variegated signal in the MV/HV tier at the destination, typically the distribution grid’s local substation. If the communication signal effectively transits between the tiers of the distribution grid, then various Digital Signal Processing (DSP) techniques can be utilized to recreate the signal at the substation. One flaw of this approach is that it would be unique to that source-to-destination power grid architecture at best, and every individual signal at worst. This is because every element in the power grid introduces its own time-varying interference which is coherent with the fundamental of the power signal. Thus, this approach, by itself, is not a ubiquitous or scalable solution. To make it more universally applicable, Machine Learning (ML) algorithms can be employed [9]. These algorithms will adjust the parameters of the signal processing programs to correctly extract and reconstruct the communication signal. Figure 1 shows the various types of PLC discussed above alongside some examples. The red circled branch in this figure highlights the scope of this research study in the grand scheme of PLC.

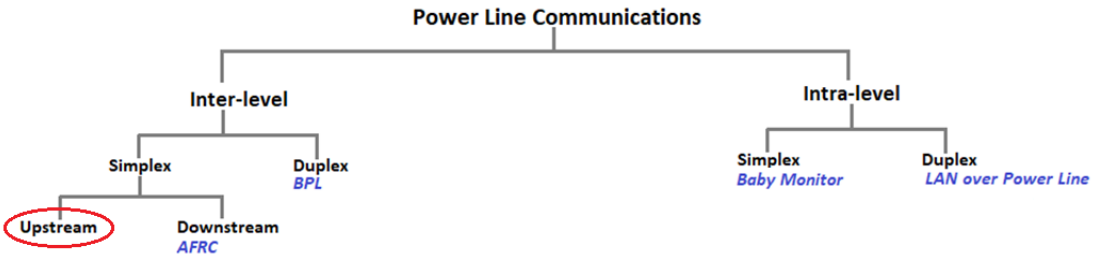


Figure 1. Chart showing the division of Power Line Communications based on signal pathway and directionality. Our research study is focused on the upstream, simplex, inter-level PLC as shown within the red circle.

1.2. Review of Relevant Literature

This section reviews the existing body of research related to PLC. Although the archival literature covers a wide variety of research in various facets of PLC, this review will focus mainly on low and ultra-low frequency, narrow-band PLC. The objective of this literature review is to survey the existing implementation method for ULF-PLC, which can be broadly categorized into 'top-down' and 'bottom-up' approach, to identify a suitable approach for our research.

Korki et al. [10] developed a ULF-PLC model for a LV distribution power grid network using a bottom-up approach to construct the model via mathematical analysis of the grid components. To evaluate the proposed model, Bit Error Ratio (BER) versus Signal-to-Noise Ratio (SNR) was investigated through numerical simulation. The simulation results showed that the model had acceptable BER (compared to other communication standards) for communication between neighboring nodes when SNR was above 15dB. They concluded that their analytical approach can satisfactorily model ULF-PLC in the LV region. This research applies a bottom-up approach to PLC since it builds a PLC model by incorporating the individual parts of the PLC network. Its counterpart, a top-down approach, uses measurement and data to estimate the nature of

the PLC network. The bottom-up approach is harder to generalize and scale to arbitrary PLC networks because of the dynamic nature of the PLC medium. This is shown by the highly specific scope (PLC in one-phase LV region between one-hop nodes) of the model proposed by this study. Therefore, in our research, we will use a top-down method, like ML, to generalize our ULF-PLC solution to a wider, less-specific grid domain.

Varadarajan et al. [11] investigated the channel and noise characteristics of the physical layer of PLC systems in the distribution grid. The characterization of channel and noise in the Federal Communications Commission (FCC) regulated PLC frequency band (9kHz to 150kHz) was done with empirical measurements from field tests using modem, utility meters, and measurement equipment such as spectrum analyzer and oscilloscope. The researchers observed that the transformers, located in the distribution grid, causes severe frequency-selective attenuation of the PLC signals. Hence, they concluded that focusing on the less attenuated sub-bands can increase the quality and the coverage of the transmitted PLC signals. In contrast to the research by Korki et al. [10], this study uses a top-down, empirical approach to characterize the PLC channel, which is simpler than an analytical bottom-up approach. However, the limitation of this research is that it does not directly provide a new technological solution to overcome the challenges of PLC, it merely makes the observations and provides suggestions for implementations. Our research will consider these observations, especially the negative effect of transformers in ULF-PLC, whilst also creating a solution framework which leverages and categorizes ML-based approaches.

Prasad et al. [12] evaluated the use of Power Line Modems (PLM) for detecting faults in the power grid. To do so, they collected raw data containing diagnostic

parameters from the modem, then, extracted suitable features from these raw data, and finally, used ML to identify and predict faults. The results of their experiment showed that they were able to detect faults with high accuracy and predict future faults with a lower, but satisfactory, accuracy. Similar to the research by Korki et al. [10], this study uses mathematical analysis to model PLC architecture. Additionally, similar to the research by Varadarajan et al. [11], this study uses a top-down approach to evaluate the fault through ML. In contrast to [10] and [11], [12] also presents a direct application of PLC i.e., grid diagnostics. However, this study is limited in scope because the associated technology, Broadband over Power line (BPL), is not widely available. Therefore, our research is not going to be focused on a specific PLC technology or hardware, but a more general implementation of ULF-PLC with a wider range of applications.

Nassar et al. [13] reviewed the signal processing techniques used to model 3-500 kHz range narrowband PLC (NB-PLC) and presented the local utility applications of NB-PLC. They modeled the channel using multi-path modeling and transmission-line modeling. They characterized the PLC noise in terms of its periodicity and uncoordinated interference. Then, they provided the framework for Orthogonal Frequency-Division Multiplexing (OFDM) implementation of NB-PLC. They concluded that channel and noise modeling can be used to make robust PLC for two-way communication between the customers and the local utility. This study uses the channel and noise modeling approach similar to Korki et al. [10], while focusing on NB-PLC above 3kHz for utility application. One limitation of this study is that, although the research paper is comprehensive, no novel techniques or experimentation were introduced. Further, the applicability of spectrum above 3kHz is suspect in the distribution grid due to the large,

series inductance presented by transformers. However, the focus on low-frequency narrow band and low data-rate for utility application is interesting. As a result, we focus on ULF-PLC and leverage findings from the various approaches described in the literature.

This literature review revealed the two main methods of PLC implementation- a bottom-up approach through mathematical modeling and a top-down approach through empirical measurement. The bottom-up approach is network and application specific, cannot be easily scaled, and is usually more complex. The top-down approach, on the other hand, has the potential to overcome the limitations of bottom-up approach by being easily scalable and simpler to implement. However, the literature survey showed this potential is largely untapped. Therefore, the objective of our research is to employ a top-down approach in ULF-PLC, including a combination of DSP techniques and machine-learning models to aid in discriminating communication signals in the distribution grid.

1.3. Thesis Problem Statement

There is a huge, largely untapped potential of PLC, especially in the implementation of smart grid technologies [14]. Although some advancement has been made for high-frequency and intra-level PLC, research and development are still lacking, particularly in ultra-low frequency, inter-level PLC. Further, use of modern computational tools like ML is still uncommon in this domain, even though such tools can efficiently handle the problem of dynamic and highly correlated noise in the PLC channel. For this reason, the objective of this research is to utilize DSP methods alongside ML techniques to perform classification of communication signal components in an inter-level, simplex, and upstream ULF-PLC transmission.

1.4. Assumptions, Limitations, Delimitations

1.4.1. Assumptions

Some of the assumptions we made in carrying out this research and their brief justification are listed below:

- Ultra-low-frequency signals will pass upstream through the transformers without substantial degradation. We can safely make this assumption because of the existing downstream ULF-PLC applications like AFRC and 'Smart Meters' as well as previous work in this space [15].
- Features extracted using DSP techniques can be used in ML for PLC signals. This is a safe assumption since DSP feature extraction has been used for ML and Neural Networks (NN) in other domains such as music, radio frequency signals, seismic signals, etc.
- Use of ML will make the PLC demodulation technique generalizable and scalable for various PLC architectures. This assumption is supported by ML implementation in other applications which tend to be generalizable and scalable.

1.4.2. Limitations

Limitations are the potential weaknesses of our study which were beyond our control. Some of the major limitations of our research and their cause are listed below:

- The frequency of our communication signal is limited to be below 10 kHz. This is because our data acquisition device had maximum sampling rate of 20 kHz and going above 10 kHz will violate Nyquist rule causing under-sampling errors. However, the presence of large, series inductance in the distribution grid (transformers) limits frequency content to well below 5kHz.

- ❖ The generalizability and scalability of the proposed ML solution cannot be tested. This is because our data were collected only from one substation and a single feeder. However, future work will directly address this testing limitation.
- ❖ The data we collected have huge dependence on the status of the grid, and thus, can show variability at different times even when the controllable parameters are kept identical. This could be overcome by testing on an artificial or controlled, closed grid which we do not have access to. However, such testing would only be applicable in the artificial environment, and so is not particularly useful in practical application.

1.4.3. Delimitations

Delimitations are the boundaries or scope of a study and are controllable. The delimitations of our research and their brief reasoning are listed below:

- ❖ Only the physical layer of the PLC medium will be investigated because the succeeding layers are similar to other communication media. In particular, OSI Layer 2, or medium access control (MAC) or multi-user sequencing /reservation/ discrimination is not a consideration in this research.
- ❖ The traditional demodulation techniques will not be employed because they are well-documented in literature. Further, our approach uses direct modulation in the baseband with no upconverting or carrier considerations.
- ❖ ML techniques are applied to the signal identification/bit classification problem of PLC to bound operational scope of the project.

1.5. Research Questions

The objective of our research is to use the inter-level PLC data to answer the following questions:

1. What are the optimal amplitude and frequency for ULF-PLC signals?
2. How should the raw ULF-PLC data be processed to use in ML/NN?
3. What ML/NN algorithms perform well for the ULF-PLC data?

1.6. Thesis Structure

This thesis is organized into five chapters. Chapter 1: 'Introduction' presented the background on PLC and provided the motivations for our research in this area. This chapter also included review and evaluation of some relevant research papers. All this background information culminated in identifying the objective and scope of our thesis research study. Chapter 2: 'Machine Learning and Neural Network' provides an overview of the ML/NN concepts and terminologies relevant to our research. Chapter 3: 'Research Methods' describes the data processing and ML/NN framework used in our experimental study. The results of these experiments, alongside their corresponding discussions are presented in Chapter 4: 'Results and Discussions.' Finally, the overarching conclusions of the research and the potential next steps in this PLC project are presented in Chapter 5: 'Conclusions and Future Works.'

2. MACHINE LEARNING AND NEURAL NETWORK

Machine learning (ML) is a data-driven algorithm optimization technique. The algorithms learn from the data to optimize themselves to perform a specific task more effectively, faster, or with better outcomes. ML has applications in various fields such as healthcare, finance, social media, transportation, and other applications. In our area of interest (PLC), ML may provide an alternative approach to conventional DSP algorithms. Rather than using mathematical models to analyze and predict characteristics of a signal as it passes through various grid components, which can be very challenging, ML aggregates the grid or the channel as a black box. The input and output are on either side of this black box, and the collection of ML algorithms tries to find a relationship. In doing so, the goal is to find some intrinsic knowledge of the channel as a whole and use it to classify or predict the outcome. In our research, the input is the transmitted ULF-PLC signal, the output is the received ULF-PLC signal, and the outcome is the binary information ("bit state") contained within this ULF-PLC signal.

2.1. Types of Machine Learning

ML algorithms can be categorized based on data, the learning process, the desired outcome, and other requirements. Some of these categories and their respective sub-categories that are relevant to this research are described in more detail in subsequent sections.

2.1.1. Based on the Forms of Learning

The training or learning process of ML can be performed in a few different ways. Based on this process, there are three main types of ML: Supervised, Unsupervised, and

Reinforcement Learning. These types are described in subsequent sections.

2.1.1.1. Supervised Learning

The training input data are labeled with their corresponding desired, observed, or measured outputs in supervised learning. These known outputs for the training data are called labels. The supervised algorithms form a model that maps the inputs to the labels with direct feedback between each training step. This model can then be used with unlabeled data (similar to the training data) to perform some tasks such as a classification, in which the training labels are discrete, or regression, in which the training labels are continuous [16], [17].

2.1.1.2. Unsupervised Learning

Unlike supervised learning, the training input data in unsupervised learning do not have labels, and thus, the training process does not use feedback [18]. Therefore, the outcome of the learning process, even for training data, is not known beforehand. The function of unsupervised algorithms is not to map inputs to outputs, but to find the hidden structure or meaningful information within the data. An example of this type of learning is clustering, where the data are grouped into clusters based on some similarity measure [19].

2.1.1.3. Reinforcement Learning

In reinforcement learning, a system or an agent is trained to maximize a reward through interaction with the environment [20]. An example of such reinforcement learning is a chess engine where the system is trained by associating the reward, i.e., win or lose at the end of the game, with each move depending on the environment, i.e., state of the board [16]. There is feedback in this type of learning since the reward is connected

to the training; however, it is more complex than feedback for supervised learning.

Depending on the application, PLC can benefit from the full spectrum of ML types and techniques. However, for this research, we focus primarily on supervised algorithms with classification tasks. Supervised learning provides several benefits over unsupervised or reinforcement algorithms for our ULF-PLC application, including the bit classification objective of the research, the availability of the training labels, and ease of implementation.

2.1.2. Based on the Data Processing Required

Although not a traditional category for classifying ML, the data preprocessing requirement is an important distinction because of its implication in realistic applications. Under this criterion, ML can be broadly divided into two sub-categories: feature-based learning and featureless learning, as described in subsequent sections. In this research, we implement both feature-based and featureless ML on our PLC data, comparing and contrasting the performance of these two methods based on various metrics.

2.1.2.1. Feature-based Learning

In this type of ML, valuable features are extracted from the raw data in the processing step. These features are then compiled into a dataset before being fed into the ML algorithms. The advantage of this type of ML is that large datasets with lots of noise and irrelevant features can be transformed and concentrated into smaller datasets, reducing computing time and complexity. However, the ML performance largely depends on the quality of the extracted features, and different people can get widely different results from the same raw dataset. The study of the data and the knowledge of the domain is vitally important in this type of ML, and thus, substantial effort in ML applications is

spent on these data study and feature planning stages.

2.1.2.2. Featureless Learning

In contrast to feature-based learning, featureless ML does not require explicit feature extraction during the data processing step. Although the ML or NN algorithms still require features, these features are extracted from the raw data by the algorithms themselves thereby removing the burden from the operator. This translates to a more generalizable, repeatable, and uniform ML analysis. This type of ML is more scalable since human intervention is less important when scaling to datasets of different sizes and properties. The main disadvantages of this type of ML are that it requires more computational time and power, and the data are more susceptible to noise.

2.2. Algorithms

Algorithms are the brain of ML. Most supervised ML algorithms work by assigning weights to the input data, passing the weighted sum through an activation and/or threshold functions to get a ML output, calculating the error between the actual label and the ML output, reassigning the weights to minimize this error and so on. The major difference between the various algorithms comes from the nature of the activation functions and the way the errors are minimized. In our research, we used three supervised algorithms: Logistic Regression, Support Vector Machines, and Decision Tree. These approaches are described in subsequent sections.

2.2.1. Logistic Regression

Logistic Regression is a widely used supervised linear classification algorithm that uses probabilities to assign weights and predict outcomes. It uses inverse logit

function[1] or "Sigmoid" as the activation function [21]. Equation 1 [16] shows the formula of this Sigmoid function. In this formula, 'z' is the net input defined as the linear combination of weights and the input features, as shown by Equation 2 [16]. Figure 2 shows this algorithm's overall architecture, which includes the input features 'x,' their corresponding weights 'w,' the net input 'z,' the Sigmoid activation function 'φ,' the threshold step function and the 'Error.' This error is defined as the difference between the output of the Sigmoid function and the actual label. The cost function of the Logistic Regression, shown in Equation 3 [16], is the function of these two parameters. The objective of the algorithm is to minimize this cost function 'J.'

$$\varphi(z) = \frac{1}{1 + e^{-z}} \tag{1}$$

$$z = w_0 x_0 + w_0 x_0 + w_0 x_0 + \cdots + + w_m x_m \tag{2}$$

$$J(\varphi(z), y; w) = \begin{cases} -\log\big(\varphi(z)\big)\, if\ y = 1 \\ -\log\big(1 - \varphi(z)\big)\, if\ y = 0 \end{cases} \tag{3}$$

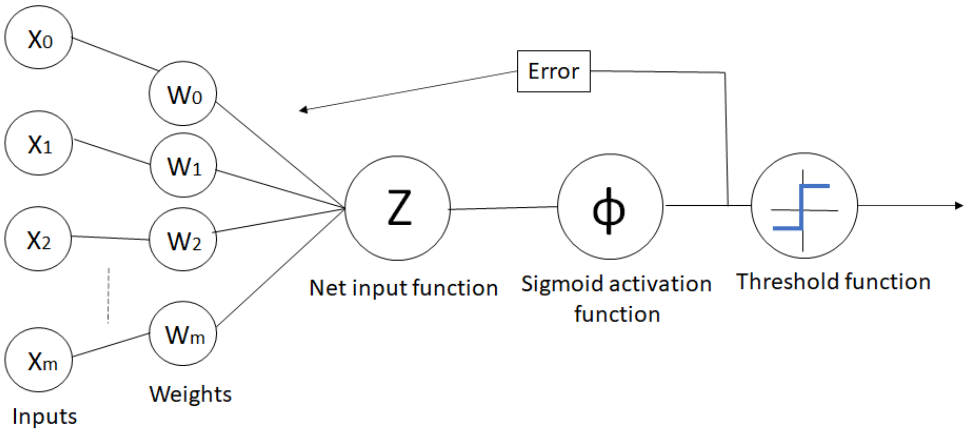


Figure 2. Architecture of Logistic Regression algorithm.

[1] The "logit" function is the quantile function associated with the standard logistic distribution.

2.2.2. Support Vector Machine

Support Vector Machine (SVM) is another popular supervised classifier algorithm. Unlike LR, SVM is capable of non-linear classification, in addition to the linear classification, which makes this algorithm more versatile. In the context of this algorithm, support vectors are defined as the training samples that are closest to the decision boundary. The imaginary curves connecting these support vectors with the same label are called hyperplanes. The distance between various hyperplanes is called margin. The objective of the SVM algorithm is to maximize this margin. The decision boundary with the best separation between various classes is chosen, thereby optimizing classification [22]. Figure 3 shows the various components and working of SVM.

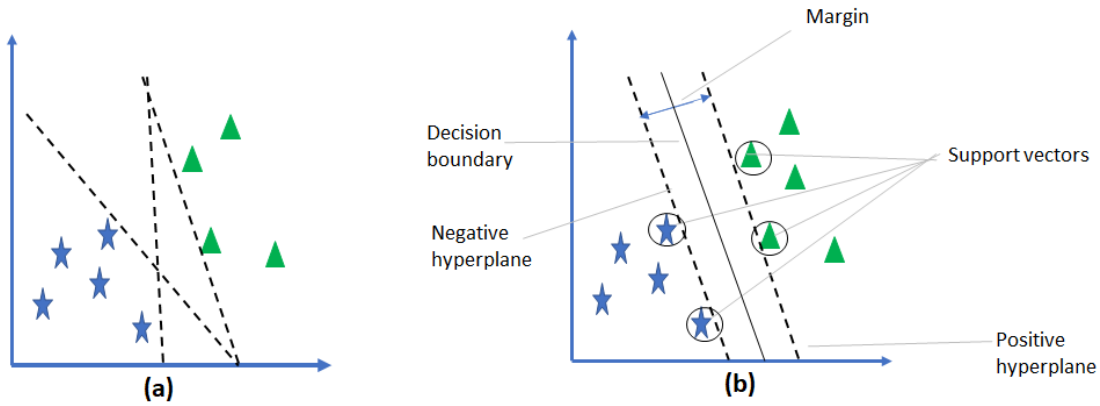


Figure 3. Components and working of SVM. (a) Plot showing the data with two distinctive labels and the dotted line representing the possible hyperplanes. (b) Same data as (a) but with maximized margin between the two hyperplanes.

2.2.3. Decision Tree

A Decision Tree is a tree diagram method of classification in which a decision causes each split or branching. These decisions are chosen at each step to maximize the Information Gain (IG) [23]. IG is defined as shown in Equation 4 [16] where 'f' is the feature column that performs the split, 'I' is the Impurity function, 'D_p' is the dataset of

the parent, 'D_j' is the dataset of the j^{th} child node, 'N_p' is the number of samples at the parent node and 'N_j' is the number of samples at the j^{th} child node. A child node can be generated with each iteration until the training examples at each node all belong to the same class. However, this can lead to overfitting, and thus, it is more practical to "prune" the tree by setting the maximum depth of the tree.

$$IG(D_p, f) = I(D_p) - \sum_{j=1}^{m} \frac{N_j}{N_p} I(D_j) \tag{4}$$

2.3. Neural Network

As the name suggests, Neural Networks (NN), also referred to as Artificial Neural Networks (ANN), Multi-layer Neural Networks, or Multi-Layer Perceptron (MLP), are fully connected networks of artificial neurons, such as the Logistic Regression neuron shown in Figure 2. A basic structure of NN is shown in Figure 4, which consists of the input layer, at least one hidden layer, and an output layer [24]. The NN is called Deep Neural Network (DNN) if there is more than one hidden layer [16].

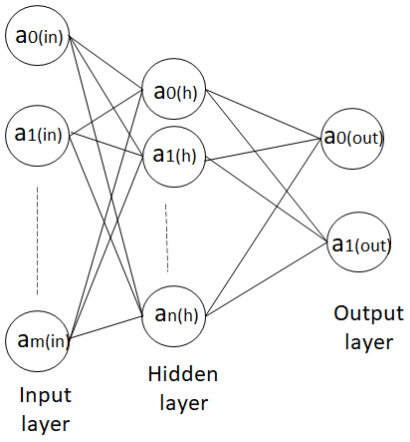


Figure 4. Basic structure of a Neural Network. This structure shows the input layer with 'm' number of inputs, hidden layer with 'n' number of nodes and output layer with two nodes.

The output of each neuron propagates forward through the network to generate an

output in the output layer starting at the input layer. Then, based on this cumulative network output, the error is calculated to minimize using a cost function. This error is then backpropagated, with partial derivative being calculated to each weight, and the model is updated [24]. This process is then repeated. Since derivatives are calculated during backpropagation, the activation function of the neurons needs to have derivatives. Some of the common activation functions used in NN are Sigmoid function (shown in Equation 1), hyperbolic tangent function (Equation 5) and Rectified Linear Unit (ReLU) function (Equation 6) [16]. The activation function for the output layer is usually set to the softmax function (Equation 7) [16].

$$\varphi_{tanh}(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \tag{5}$$

$$\varphi_{ReLU}(z) = \begin{cases} 0 \; if \; z < 0 \\ z \; if \; z > 0 \end{cases} \tag{6}$$

$$\varphi_{softmax}(z) = \frac{e^{z_i}}{\sum_{j=1}^{M} e^{z_j}} \tag{7}$$

2.4. Convolutional Neural Network

Convolutional Neural Network (CNN) are a special type of NNs in which the fully connected NN layers are preceded by at least one convolution layer [25]. CNNs are popular especially in the fields of image classification because of their ability to extract patterns from the images using convolution filters or kernels. Since we are using spectrograms in our research, which in their raw form are “unprocessed images,” CNN are given a consideration.

At the heart of the CNN is the convolution operation. Equation 8 [16] shows the mathematical definition of a convolution between input vector ‘x’ and filter ‘w.’ The ‘*’

symbol is a common notation for the convolution operator.

$$y[i] = x * w = \sum_{k=-\infty}^{\infty} x[i-k]w[k] \tag{8}$$

In practice, Equation 8 can be explained as the elements of convolution output 'y' being the dot product of input 'x' and rotated filter 'w^R' as 'w' slides over 'x.'

The output of convolution, in the case of CNN, produces feature maps. These feature maps are similar to the ML features, and thus, the convolution layer in a CNN is also known as feature extraction layer [16]. The difference here is that, unlike the traditional ML features, these features are generated automatically by convolution filters. Therefore, CNNs are ideal for featureless ML (i.e., without explicitly providing features).

3. RESEARCH METHODS

In this chapter, the methodology used to collect the raw data, the processes used to convert or transform the raw data to a ML ready format, and the setups of the ML and NN for the experimentation are described. Section 3.1 describes all aspects of the research related to the ULF-PLC data whereas Section 3.2 explains the methodology behind feature-based and featureless ML/NN.

3.1. Data

Figure 5 summarizes the steps taken to convert the raw ULF-PLC data to ML datasets. As shown in this figure, the raw ULF-PLC data (.wav format) were divided into -two sets: 'Raw Set 1' and 'Raw Set 2.' From these sets, feature-based (Dataset 1 and 2) and featureless (Dataset 3-8) ML/NN datasets were created. For feature-based datasets, three features: amplitude envelope, RMS energy and spectral centroid, were extracted from the raw data. For featureless datasets, raw time-series data and their spectrograms were used. The dimensionality of each of these ML/NN datasets is also shown in Figure 5. The feature-based and featureless time-series datasets were 1D datasets, magnitude spectrogram datasets were 2D datasets, and rectangular spectrogram datasets were 3D datasets. Each component of Figure 5 is described in detail in the subsequent sections and this figure will be referenced throughout these sections.

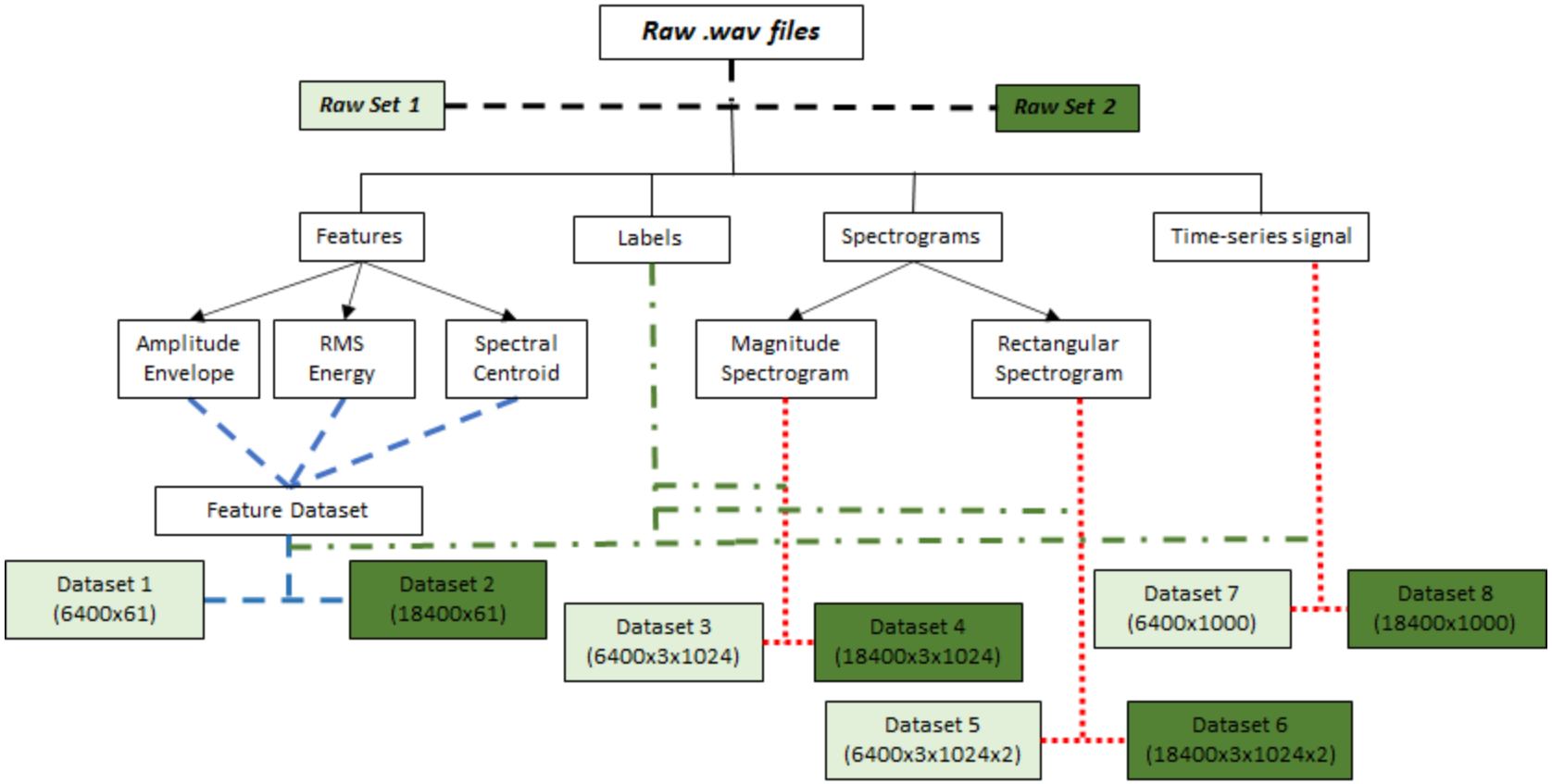


Figure 5. Chart showing the process of conversion of raw data files to feature-based and featureless ML/NN ready datasets. The blue dashed line shows the construction of feature-based datasets (Datasets 1&2) while the red dotted line is for the featureless datasets (Datasets 3 to 8). The datasets corresponding to 'Raw Set 1' (Datasets 1, 3 & 5) are lightly highlighted and the ones corresponding to 'Raw Set 2' are dark highlighted. The dimensionality of each dataset (axbx..) is shown as well. Please note that the first number in this dimensionality (a) corresponds to the number of independent samples in the ML/NN dataset and the numbers following (bx..) corresponds to the dimension of each sample. Hence, feature datasets (1&2) and time-series datasets (7&8) are 1D datasets, magnitude spectrograms (3&4) are 2D datasets and rectangular spectrograms (5&6) are 3D datasets.

3.1.1. Transmission and Collection

The experimental dataset(s) used in this research study was collected by an undergraduate group a few years ago as part of a Senior Design project [15]. Unfortunately, attempts to gather new data using evolved acquisition systems and a different architectural approach were foiled by lack of appropriate and timely support from institutional information technology and networking personnel.

The data transmission and collection architecture leveraged for the experimental dataset(s) is shown graphically in Figure 6. As shown in this figure, baseband modulated ULF-PLC data were transmitted from a research lab and collected using a remote data acquisition system at a nearby substation. To create the experimental dataset(s), an Ametek CS3000 programmable current source [26] was used to generate On-Off Keyed (OOK) current signals with frequencies varying from 690 Hz to 2010 Hz and amplitudes varying from 0mA (while 'Off' or 'Low') and 10mA to 1A (while 'On' or 'High'). These signals were then injected into the power line via a specially constructed narrowband filter. This filter, designed by the Senior Design group [15] based on specifications provided by Dr. McClellan, is a transformer-based bandpass filter or "resonant tank" which blocks the power signal from entering into the output port of the current source while allowing the signal generated by the current source to pass into the power line. The signal injected into the power line first enters a split-phase distribution link typical of conventional "wall outlets," then into power grid via three-phase distribution transformers which "step up" the voltage for longer distance transmission (e.g., 13kV). At each "level change," the transformer will jointly adjust the voltage (upward) and current (downward), according to the structure of the transformer's coils (turns ratio).

Some signature of the transmitted ULF-PLC signal is introduced into all three phases of the distribution and transmission links during this voltage transition as a result of magnetic cross-coupling within the transformer [3]. In the substation, the power signal and the injected ULF-PLC signal are then collected using a conventional high-quality Data Acquisition device (DAQ), including Current Transformers (CT) which sense the current disturbances introduced by the injected ULF-PLC signal [27]. The ULF-PLC communication signal originates at the LV region in the lab and travels towards the HV region in the substation of the distribution grid, thereby making the PLC path upstream, simplex, and inter-leveled.

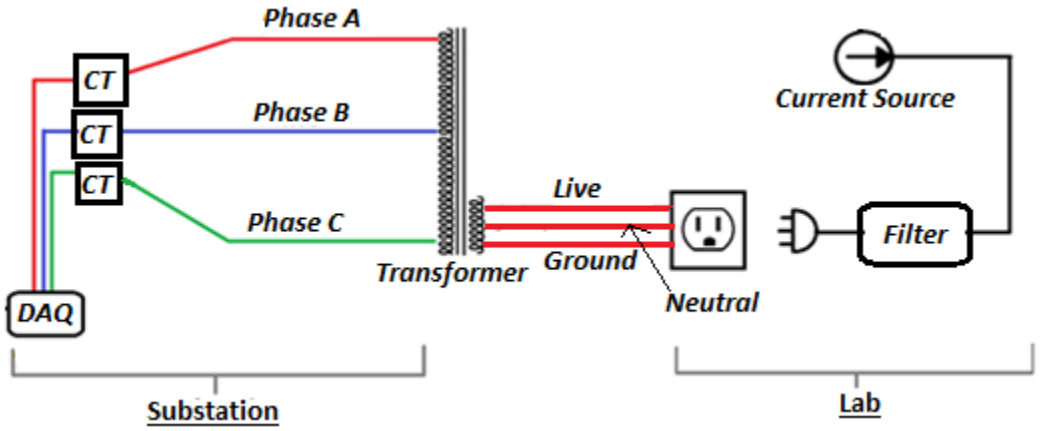


Figure 6. Data transmission and capture architecture for upstream, inter-level ULF-PLC. The current source located in the lab generates a modulated current signal which is injected into the power line and is collected at the substation.

3.1.2. Raw Time-Series Data

The raw data (shown as 'Raw .wav files' in Figure 5) captured using substation-resident, remote DAQ are a set of three-phase time-series data sequences consisting of a 60Hz power signal and its harmonics, communication signal (ranging from 690Hz-2010Hz) and its harmonics, and time-variant noise at all frequencies. The acquired signals were sampled at 8kHz using 16-bits of resolution per sample (see Appendix B for

more details). The fundamental of the power signal dominates these raw data because of its relatively high amplitude, as shown in Figure 7. The three-phase raw time-series data plotted in this figure contain the PLC signal at 1595Hz, but it is not apparent from this plot alone. Therefore, we hypothesized that analysis of these time-series data via an ML/NN approach would not yield a good result. To test this hypothesis, unprocessed time-series datasets, Dataset 7 and Dataset 8 shown in Figure 5, were used for ML/NN evaluation.

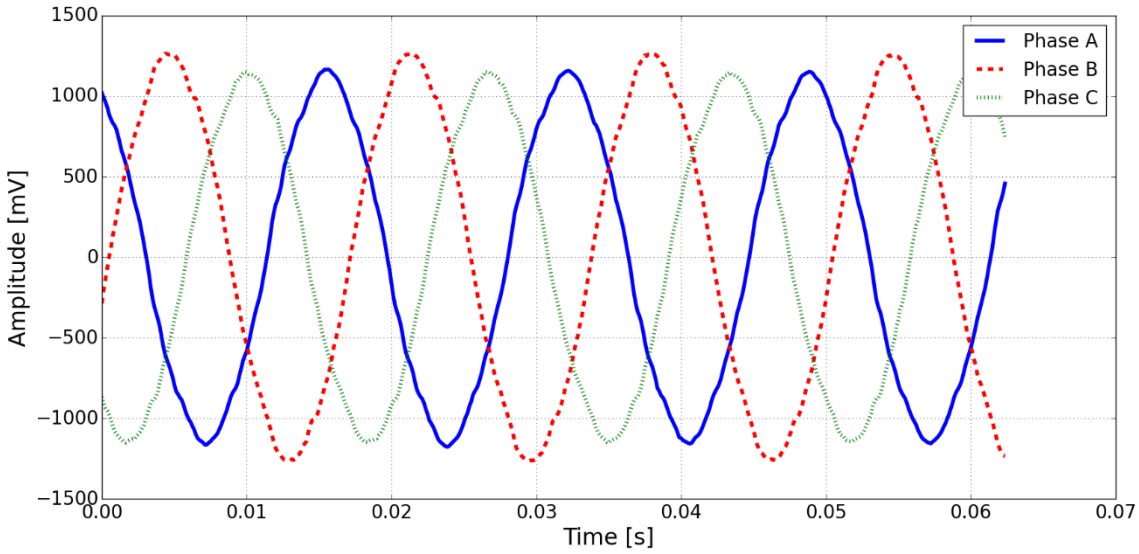


Figure 7. Time-domain plot of three-phase data received at the substation. The three phases correspond to the three phases of the power signal, but these signals also contain ULF-PLC signal at 1595Hz. However, the ULF-PLC signal is not visually identifiable from this plot since the power signal is very dominant.

3.1.3. Processing

Since the raw-time series data are predictably not very useful in our case, the next logical step is to transform the signal into other domains and/or extract useful information (or features) from it. These transformation and feature-extraction approaches are described in subsequent sections.

3.1.3.1. Transformation

Transformation, for our purposes, is defined as the conversion of time-series data to frequency domain or time-frequency domain data. This transformation is reversible, and, in the absence of additional processing steps, maintains the integrity of the original data. Hence, there is very little loss of information in these kinds of transformations. The frequency and time-frequency transformations of our original time-series data are described in subsequent sections.

3.1.3.1.1. Frequency Domain

The time-domain raw data were transformed to frequency domain using the Fast Fourier transform (FFT) [28]–[30] . Figure 8 shows a representative example of the frequency domain plot of a single-phase sequence containing a ULF-PLC signal transmitted at 1595Hz. As seen in this figure, the power signal and its harmonics dominate the spectral plot. Nonetheless, as indicated by the red arrow in Figure 8, a smaller but prominent peak is present at 1595Hz which shows the presence of our transmitted PLC signal. However, the spectrum plot cannot show the time-varying nature of the signal, and thus did not provide us information about the OOK encoded data that were transmitted. Hence, the frequency-domain transformation of the entire acquired signal can be used to detect the presence or absence of the PLC signal but cannot be used to decode the time-varying information contained within it.

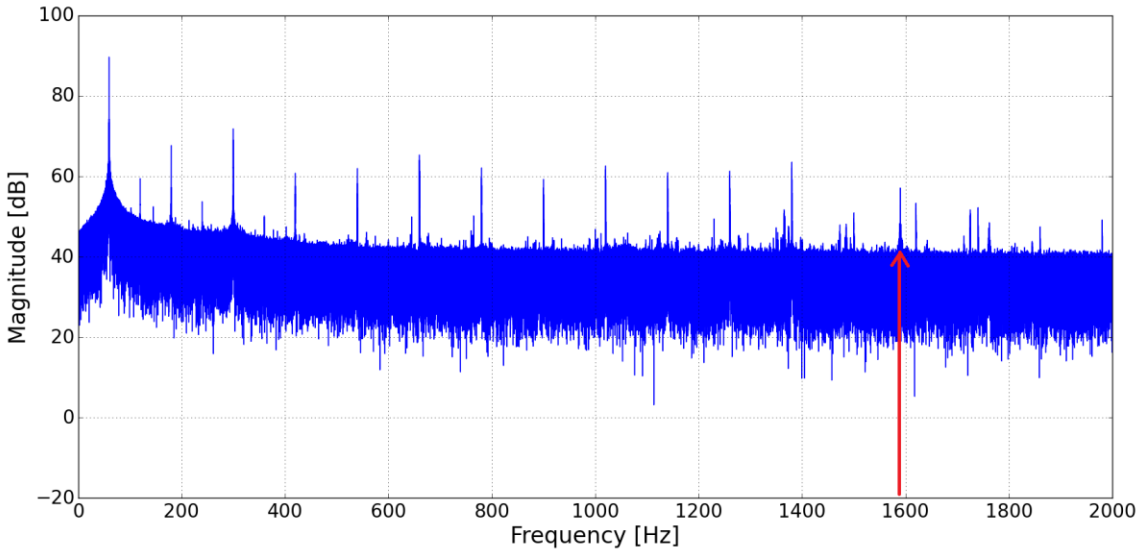


Figure 8. Frequency domain plot of the raw data showing presence of the ULF-PLC signal at 1595Hz. The highest peak in this plot shows the dominant power signal at around 60Hz. The odd harmonics of this power signal can be seen as the secondary peaks. Our test ULF-PLC signal, transmitted at 1595Hz, has a prominent peak as well, shown by the red arrow.

3.1.3.1.2. Time-frequency Domain

The deficiencies of the whole-signal frequency domain transformation, i.e., the inability to decode OOK information in our case, can be mitigated by appropriate time-frequency analysis such as a spectrogram. The transformation method(s) used to produce a spectrogram are very similar to the frequency-domain conversion, except the FFT is applied to frames or sub-sequences within the sequence instead of the entire time-series sequence, a process known as Short-Time Fourier Transform (STFT). This approach provides some indicative information regarding the presence or absence of the PLC signal in those time frames. Therefore, concatenating the spectral analysis of these time frames across the length of the data can be used to represent the transmitted information in its entirety.

In practice, there are a few measures that can be taken during this time-frequency transformation to enhance information contained in the raw data. Windowing the time

frames with window functions can minimize the spectral leakage effect [31]–[33]. Similarly, overlapping the time frames prevents information loss caused by the tapered ends of the windowing functions. Parameters related to these techniques, such as window type, frame length, overlap length, as well as other parameters such as frequency resolution affects the quality of the spectrograms, and thus, their values need to be carefully chosen (see Appendix C for values).

Since spectrograms can be produced using Fourier (or similar) transforms, the resulting dataset(s) may have complex-valued representations. Complex-valued data have both real and imaginary components. Equation 9 [34] shows the mathematical representation of a complex number 'z,' where 'a' is the real component, 'b' is the imaginary component and 'i' is the indeterminate satisfying $i^2 = -1$.

$$z = a + ib \tag{9}$$

In ML/NN implementation, these complex-valued data pose a critical challenge since the traditional ML/NN algorithms and architecture are typically not structured to operate on complex-valued datasets. One way to overcome this challenge is to devise new ML/NN frameworks which accommodate complex-valued inputs and have complex-valued weights or coefficients. This is an active area of research [35], [36] but out of scope for our study. An easier, more accessible approach is to use an appropriate, real-valued representation of the complex-valued data as input to conventional ML/NN structures. In this research, we utilize two different real-valued formats for complex-valued transform data: the magnitude format, and the rectangular format, as presented in subsequent sections.

3.1.3.1.2.1. Magnitude Format

The most common method of converting complex-valued data to real-valued data is by combining the rectangular or Cartesian real and imaginary components via a polar transformation. In polar form, the magnitude of a complex number is defined as the square root of the sum of squares of the real and imaginary components. Equation 10 [34] shows the formula for this magnitude where '|z|' represents the magnitude of a complex number 'z,' 'a' is the real component and 'b' is the imaginary component. The magnitude of a complex number represents its distance from the origin in the complex plane.

$$|z| = \sqrt{a^2 + b^2} \tag{10}$$

For time-series data, taking the magnitude of each complex-valued number of a 2D complex-valued spectrogram gives a magnitude spectrogram. This magnitude spectrogram has the same dimensions as its parent complex-valued spectrogram. Figure 9 shows a magnitude spectrogram of one of our captured ULF-PLC datasets where the PLC frequency is at 1595Hz. This figure shows a dotted band at around 1595Hz which correspond to the transmitted OOK PLC signals. The zoomed in version of this band is shown in the inset black box of Figure 9, which clearly shows the 'On' and 'Off' states of the ULF-PLC signal. The solid bands at various frequencies correspond to the power signal and its odd harmonics (shown by the black arrows in Figure 9). The harmonics are spaced 120Hz apart. The PLC signal also produces its own harmonics, though not as strong as power signal harmonics. Interestingly, the "harmonics" of the ULF-PLC signal are harmonically offset from the ULF-PLC signal based on the fundamental frequency of the power signal, not the ULF-PLC signal. As a result, they are "echoes" or "images" of the ULF-PLC signal with harmonic structure described by the power signal's frequency.

If the ULF-PLC signal is injected in the frequency bands between the harmonics of the power signal, then a magnitude spectrogram can provide a good estimation of the transmitted information, especially OOK or amplitude-shifted ULF-PLC signal as shown in Figure 9. Therefore, we used magnitude spectrogram datasets (Dataset 3 and Dataset 4 as shown in Figure 5) for ML/NN to evaluate the accuracy of decoding the PLC signals.

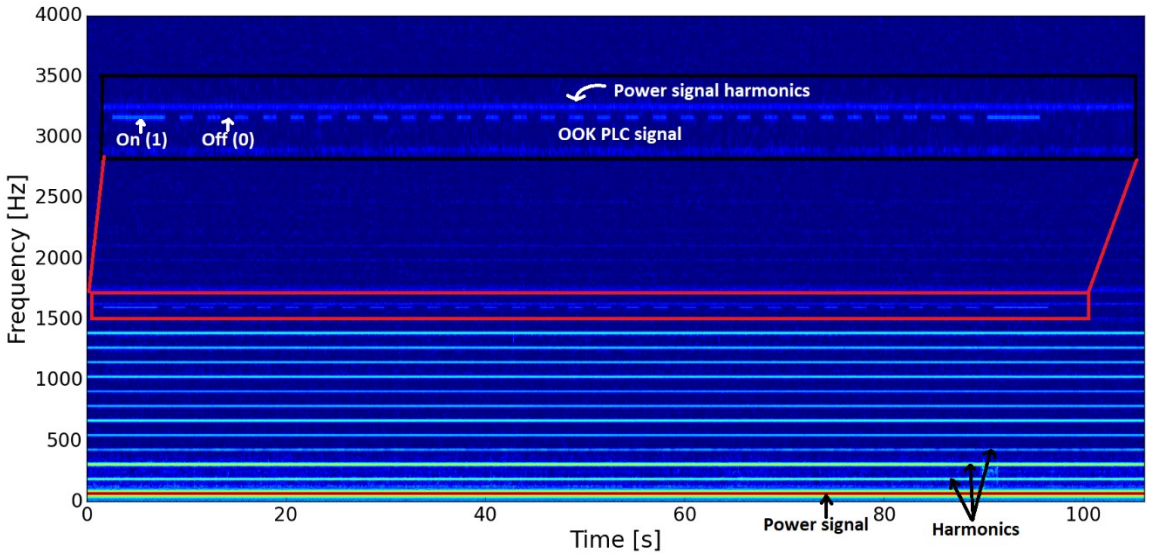
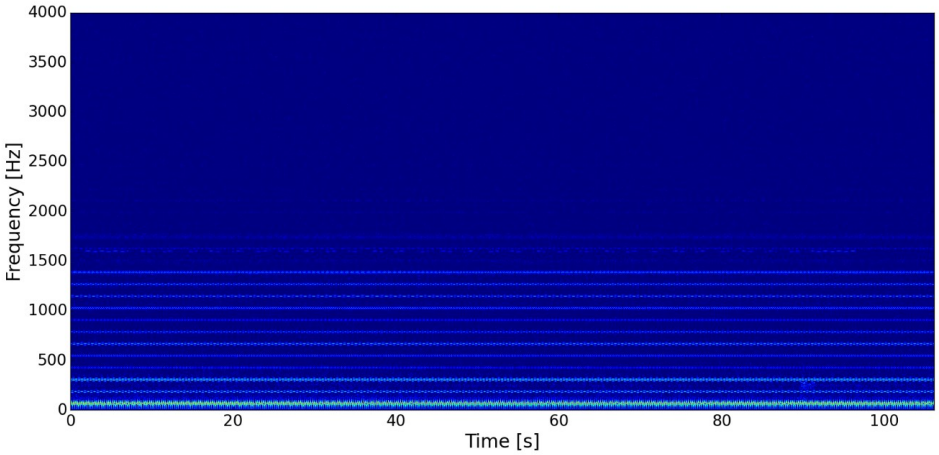


Figure 9. Magnitude spectrogram of the captured data showing the transmitted OOK ULF-PLC signal with the dotted horizontal band at around 1595Hz. The solid bands at lower frequencies correspond to the power signal and its harmonics. These bands are brighter than the ULF-PLC band because of their dominance in amplitude over the ULF-PLC signal in the power lines.

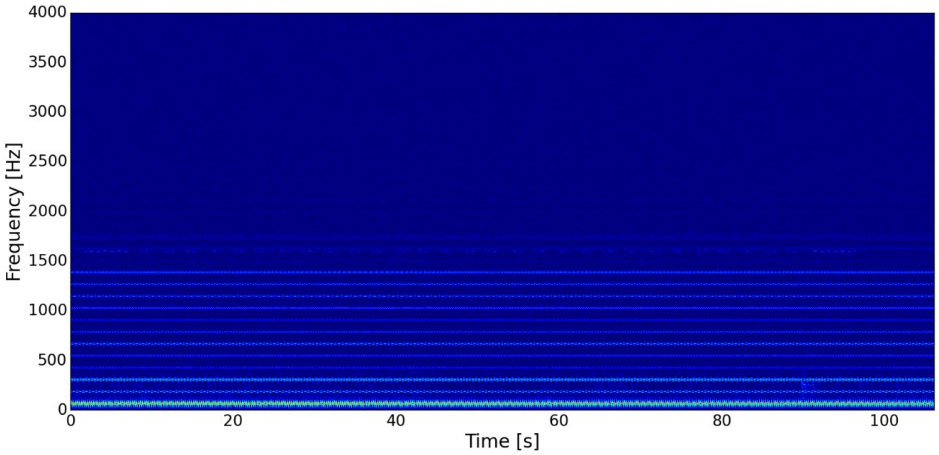
3.1.3.1.2.2. Rectangular Format

In addition to magnitude spectrograms, other types of spectrograms can be generated as well by manipulating the components of the complex data. By taking only the real components, a real spectrogram can be produced. Similarly, an imaginary spectrogram can be produced by only taking the imaginary components. Another example is a phase spectrogram, which can be generated by taking only the phase angle of the polar representation as opposed to the magnitude value. Plots in Figure 10 show the real spectrogram, imaginary spectrogram and phase spectrogram of the same dataset

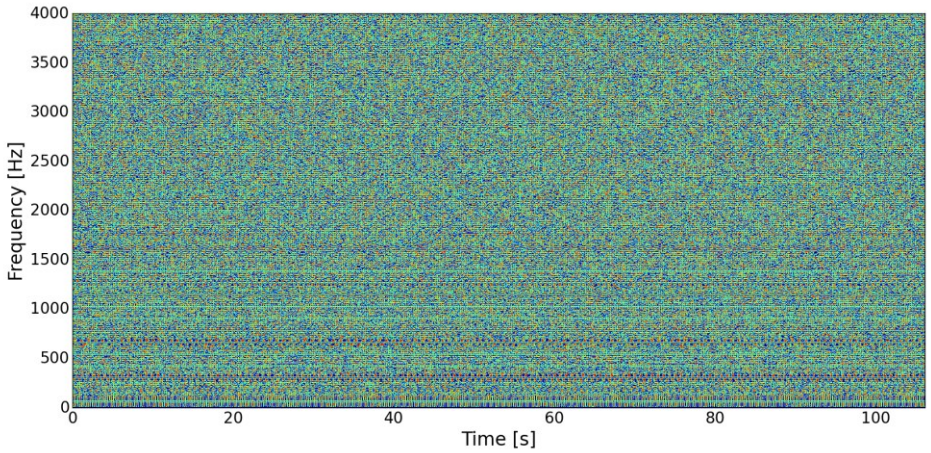
used for magnitude spectrogram in Figure 9.



(a) Real spectrogram



(b) Imaginary spectrogram



(c) Phase Spectrogram

Figure 10. Real spectrogram (a), imaginary spectrogram (b) and phase spectrogram (c) of captured data with ULF-PLC signal at 1595Hz.

As seen from plot (c) in Figure 10, the phase spectrogram is very noisy, and the

presence of the ULF-PLC signal is not visually apparent. Therefore, phase spectrogram(s) were not used in this research study. On the other hand, the real and imaginary spectrograms show a trace of the PLC signal at 1595Hz; however, this band is not as distinct as the magnitude spectrogram plot in Figure 9. This is expected since the real and imaginary spectrograms completely ignore the other component, resulting in huge loss of information. However, a combination of these two 2D spectrograms in a 3D space could potentially be better than magnitude spectrogram since this would be a direct representation of complex-valued data as shown in Equation 9. This 3D spectrogram is hereby referred to as 'Rectangular spectrogram' in this paper because of its correspondence to the rectangular form of the complex number. Figure 11 shows in detail how the rectangular spectrogram (and other spectrograms) is generated from the raw time-series data (as suggested in [37]). Our previous research study [38] showed that rectangular spectrogram indeed performs better than magnitude spectrogram, especially when the signal of interest is dominated by out-of-band interferers such as the power signal fundamental which is present in PLC transmissions. Therefore, we used 3D rectangular spectrograms datasets (Dataset 5 and Dataset 6 as shown in Figure 5) for ML/NN analysis and evaluation in this research.

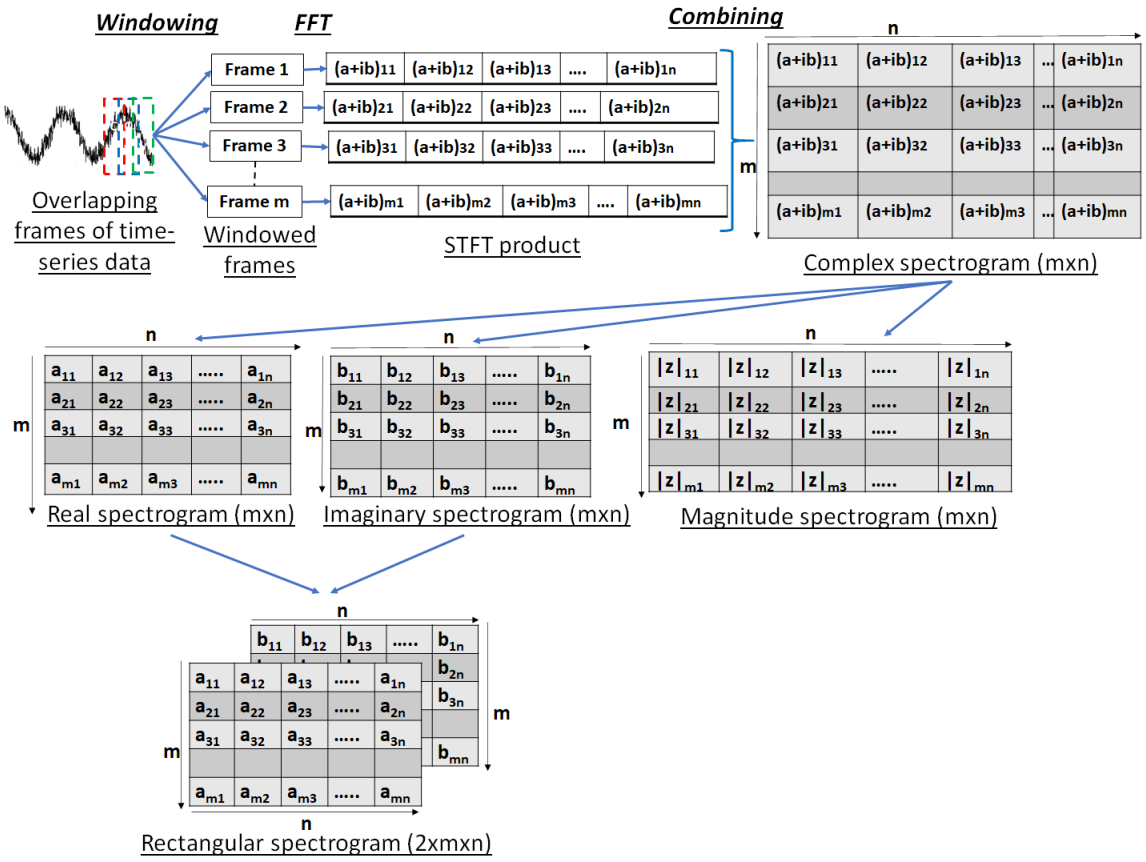


Figure 11. Process of generating rectangular spectrogram from raw time-series data. As shown here, the 1D raw-time series data are first divided into multiple overlapping frames. Then, each of these frames are windowed by a window function. Next, these time-series frames are transformed into frequency space using Fast Fourier Transform (FFT). The 1D complex-valued FFT products for all frames are then combined to get a 2D complex spectrogram. From this complex spectrogram, real (denoted by a_{xy}), imaginary (b_{xy}) and magnitude spectrogram ($|z|_{xy}$) are produced. Finally, the real and imaginary spectrograms are stacked to form a 3D rectangular spectrogram.

3.1.3.2. Feature Extraction

As mentioned in Chapter II, one of our project goals was to compare the performance of feature-based and featureless ML/NN techniques. For featureless NN, the raw time-series data as well as the various types of spectrograms can be used. However, for a traditional, feature-based ML/NN, a set of signal features needs to be extracted from the raw data. These features can then be compiled into a ML/NN ready dataset (Dataset 1 and Dataset 2 as shown in Figure 5). For our research, we chose the: amplitude envelope, root-mean-square energy, and spectral centroid as elements of our feature dataset. These

elements are described in subsequent sections.

3.1.3.2.1. Amplitude Envelope

This feature gives the change in the amplitude of the signal over time [39]. It effectively traces the outline of the signal in the time-domain and is loosely representative of the instantaneous energy in the signal. In our case, the raw signal's amplitude envelope would not provide any meaningful information, since as described in Section 3.1.2 and Figure 7, the 60Hz power signal dominates all other superimposed sinusoidal signals. Therefore, we filtered the raw frames with bandpass filters of 100Hz bandwidth starting from 1Hz and up to 3000Hz with no overlap (1Hz-100Hz, 101Hz-200Hz,…,2901-3000Hz). Hence, we divided each frame into thirty frequency-separated signals and calculated amplitude envelope for each. Our expectation was that the amplitude envelope of some of these signals, which contains our communication signal, would provide information about the bit that was transmitted in that frame.

3.1.3.2.2. Root-Mean-Square Energy

The energy of a signal is the measure of the "strength" of the signal. A signal's energy is defined as the sum of the square of its magnitude [40]. Thus, Root-Mean-Square Energy (RMSE) is the square root of the mean energy of a signal. This is particularly useful for signals which are "balanced" or have zero mean in the time domain. Equation 11 [41] shows the formula for RMSE where x_i is the i^{th} sample of signal x and N is the total number of samples.

$$RMSE = \sqrt{\frac{(x_1^2 + x_2^2 + \cdots + x_N^2)}{N}} \tag{11}$$

In our case, the raw signal's energy (or instantaneous energy in a frame) would

again be dominated by the power signal. Hence, we frequency separated the frames as described in Section 3.1.3.2.1 and calculated RMSE for each of the thirty bandpass filtered signals of each frame. Like the amplitude envelope, we were expecting variations in the RMSE in frequency range containing ULF-PLC signals (for example 1501-1600Hz for the 1595Hz PLC signal) corresponding to the bit these frames were carrying.

3.1.3.2.3. Spectral Centroid

Amplitude envelope and RMSE are time-domain signal features, and thus, they were extracted or estimated directly from the time-series data. In contrast, the spectral centroid probes the frequency-domain representation of the raw data for important signal characteristics. The spectral centroid measures the center of mass of the signal's spectrum [42]. Our raw signal's spectrum had a primary peak at near 60Hz and large secondary harmonic peaks at odd multiples of 60Hz because of the dominant power signal. Whenever the communication signal is present in the raw signal, a peak is present at that frequency and time location, such as the peak shown by the red arrow in Figure 8. The presence and absence of the communication signal (corresponding to 1 and 0, or "energy" and "no energy" respectively) would noticeably shift the center of mass of the spectral representation, thereby providing a classification measure of the transmitted bit. Therefore, we included the spectral centroid of each frame as one of the features.

3.1.4. Dataset

In this section, the series of steps taken to convert the raw data into ML/NN ready format dataset, as shown in Figure 5, are described in detail. As discussed in Section 3.1.3.1.2, the use of conventional ML/NN structures require real-valued input data, so various pre-processing steps may be valuable in constructing, evaluating, and leveraging

various training structures and algorithmic constraints.

First, the bulk of the experimental dataset was divided into two sets. The first set (shown as 'Raw Set 1' in Figure 5) contained eight data files, all with the same frequency but with varying amplitude (see Appendix D). The second set ('Raw Set 2' in Figure 5) contained twenty-three files, all with the same amplitude but with varying frequencies (Appendix E). The amplitudes and frequencies for the two sets are shown in Table 1. 'Raw Set 1' was used to analyze the effect of PLC signal amplitude variations on the performance of ML/NN while 'Raw Set 2' was used to analyze the effect of PLC signal frequency variations on ML/NN performance. For every file in both sets, the capture sampling rate was 8000 samples per second and the signal length was 105 seconds (more in Appendix B).

Table 1. The two sets of raw data and their corresponding amplitudes and frequencies.

Raw Data Sets	**Amplitude (mA)**	**Frequency (Hz)**
Raw Set 1	*10, 20, 50, 100, 250, 500, 750, 1000*	*1170*
Raw Set 2	*1000*	*690, 750, 810, 870, 930, 990, 1050, 1110, 1170, 1230, 1290, 1350, 1410, 1470, 1530, 1590, 1650, 1710, 1770, 1830, 1890, 1950, 2010*

The bit duration, i.e., the length of the individual on or off signal representing a '0' or '1' bit, in the experimental dataset is about 2 seconds (8000*2=16000 samples) on average. However, to get more ML/NN dataset samples (corresponding to the rows of the dataset) per file, the bit size was fragmented and set to 1000 samples. Further, each file from 'Raw Set 1' and 'Raw Set 2' were trimmed from 105 seconds to 100 seconds to achieve 800,000 samples per file (per phase). Hence, each of these files provided 800 samples or rows (800,000 raw time-series samples ÷ 1000 samples per bit) in the feature-based ML/NN dataset.

Next, for each of these files, amplitude envelope, RMS energy and spectral centroid were calculated. Then, these features were combined for files in 'Raw Set 1' and 'Raw Set 2' to get 'Dataset 1' and 'Dataset 2' respectively. These two datasets were later used for feature-based ML/NN. These two datasets comprised a 2D dataset with 61 feature columns - 30 each for amplitude envelope and RMS energy, and 1 for spectral centroid - and 6400 rows (800 per file*8 files) for 'Dataset 1,' and 18400 rows (800 per file*23 files) for 'Dataset 2.'

Similarly, for each file in 'Raw Set 1' and 'Raw Set 2,' 2D magnitude spectrogram and 3D rectangular spectrograms were constructed (see Appendix C for spectrogram parameters). The combined magnitude spectrograms for files in 'Raw Set 1' gave rise to 'Dataset 3' and for files in 'Raw Set 2' to 'Dataset 4.' Similarly, with rectangular spectrograms, 'Dataset 5' and 'Dataset 6' were constructed from two sets of raw data. Finally, the raw time-series made up 'Dataset 7' and 'Dataset 8.' The labels were also extracted and added to each of the datasets. The label extraction was done by filtering the raw data using a narrow band-pass filter centered at the PLC carrier frequency and using a threshold function to determine the 'On/Off' state. Figure 12 shows the plot of the filtered signal overlaid with the extracted 'On/Off' (or digital) state which corresponds to the binary labels of the datasets.

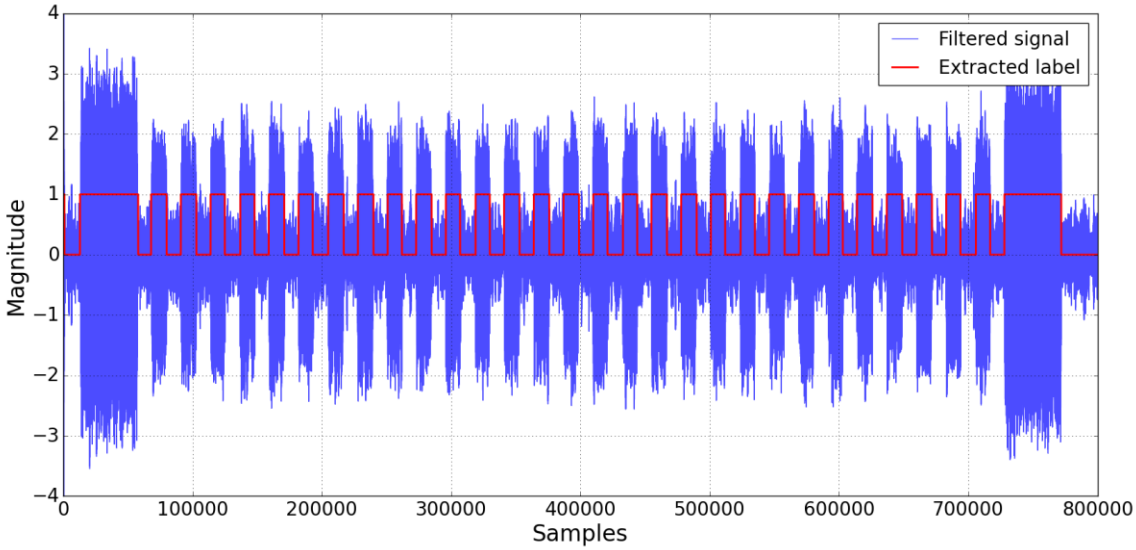


Figure 12. Plot showing the validity of feature extraction process. The raw signal after passing through band-pass filter centered at ULF-PLC frequency is shown by the blue line and the label extracted from this filtered signal is shown by the red line.

3.2. Machine Learning and Neural Network

3.2.1. Feature-based ML/NN

For feature-based ML/NN, 'Dataset 1' and 'Dataset 2' were used as discussed in Section 3.1.3.2 and Section 3.1.4. After these datasets were formed by compiling the features from the raw data and the corresponding labels were recorded (as shown in Figure 5), they were used in forming ML models with a 70:30 training-testing split. To form the models with various supervised algorithms, Python Sci-kit learn [43] was used for Logistic Regression (LR) [44], Support Vector Machines (SVM) [45], and Decision Tree (TREE) [46]. The hyperparameters for these algorithms were optimized using the grid search [47] method. Accuracy, precision [48], recall [49], and f1 scores [50] were computed to evaluate and compare these various models' training and testing performance. Learning curves [51] were plotted and evaluated to ensure the models were not overfitting or underfitting. Confusion matrices [52] were also plotted to visualize the

accurate label versus the predicted label.

In general, LR does well on linear classification. Our OOK PLC signals, by themselves (or isolated), are linearly separable. However, the background power line channel, with the power signal and its coherent harmonics, are highly non-linear. Hence, if featureless or unprocessed dataset were used on LR, we expect this algorithm to struggle. The feature extraction process we have utilized filters out these background non-linearities to some extent. Therefore, LR should still be able to do reasonably well with both feature datasets. SVM, on the other hand, has both linear and non-linear kernel and TREE is a non-linear classifier. Therefore, both these algorithms should be able to provide high performance measure on our feature datasets.

Besides these basic "one neuron" ML models, multi-neuron, multilayer ANN/DNN model was also tested using python's Tensor Flow [53] and Keras [54]. The various hyperparameters of these ANN/DNN models were optimized by manual trial and error method. Accuracy scores, loss and validation curves, and confusion matrix were generated to evaluate this ANN/DNN model's performance and this performance was compared with the other ML models. Since ANN/DNN is also a non-linear classifier, we expect this method to provide high classification accuracy. For low dimensional, low complexity feature dataset such as ours, this algorithm is probably an overkill and could potentially lead to overfitting. However, ANNs are more versatile than LR, SVM and TREE because of its multi-neuronal, multi-layer structure, and thus, could also be used in ULF-PLC data with more complex signal characteristics in the future.

3.2.2. Featureless ML/NN

ANN/DNN was also used for our featureless datasets because of its simple architecture, ease of implementation, scalability, and most importantly, its ability to process multi-dimensional datasets. The multi-neuron nodes and fully connected layers of ANN can process complex information, such as the 2D and 3D spectrogram datasets, where the one-neuron ML algorithms fail. Therefore, they are an ideal candidate to test our featureless ULF-PLC datasets.

The hyperparameters of the ANN including the number of hidden layers and the number of neurons in each layer were optimized for each dataset using a similar manual trial and error method as for the ANN for feature-based datasets. Accuracy scores, loss and validation curves, and confusion matrix were generated for the optimized models for each dataset to evaluate the performance of the models and compare them across datasets and with feature-based methods.

Within our featureless datasets, we expect the time-series dataset to fail as described in Section 3.1.2. As for the magnitude and rectangular spectrogram datasets, there is a tradeoff between the quality and the quantity of information between these two. Since our PLC data have OOK signals, the magnitude spectrogram is going to capture this On/Off state's amplitude information more clearly and compactly. However, as described in Section 3.1.3.1.2.2., the rectangular spectrogram is a more complete representation of the raw signal, and thus, contains more information about the transmitted PLC signal. Therefore, the accuracy and other performance measure of these two datasets in a NN model depends on how well the NN fits the respective data. With enough optimization, we expect both these datasets to provide similar performance.

4. RESULTS AND DISCUSSIONS

The results of our research are divided into three main sections based on the type of dataset used in ML/NN. Section 4.1 covers the results generated from feature-based dataset, Section 4.2 covers the results from the use of featureless dataset in ML/NN, and Section 4.3 compares the feature-based and featureless datasets as seen from our results. At the end of this chapter, the main results from these sections are summarized in Section 4.4.

4.1. Feature-based Dataset

As described in Section 3.1.4 and Figure 5, we constructed two feature-based datasets- 'Dataset 1' for amplitude analysis using 'Raw Set 1' and 'Dataset 2' for frequency analysis using 'Raw Set 2.' Both datasets had 61 feature columns, 30 each for the 'Amplitude Envelope' and 'RMS Energy,' and 1 for the 'Spectral Centroid.'

4.1.1. Amplitude Analysis

'Dataset 1' was used for PLC signal amplitude analysis using feature-based ML/NN. The primary goal of analyzing this dataset was to evaluate the effect of PLC signal amplitudes on the quality of the transmitted signal, using ML/NN performance as a metric. The primary objective was, therefore, to identify a discrete amplitude or range of amplitudes that is suitable for PLC. The secondary objective was to identify a ML/NN algorithm that is best for evaluating the transmitted PLC signal.

'Dataset 1' was a 2D feature-dataset with 6400 rows or samples, corresponding to the 8 'Raw Set 1' files (8 files*800 samples/file), and 61 columns. This dataset, after standardization, was fed into four different supervised ML algorithms: Logistic

Regression (LR), Support Vector Machines (SVM), Decision Tree (TREE) and Neural Network (NN). The hyperparameters of the LR, SVM, and TREE algorithms were optimized using grid search method [47] and through manual tuning for each of the amplitude subsets within this dataset. These optimized hyperparameters are shown in Table 2.

Table 2. Optimized hyperparameters of the LR, SVM and TREE for the various signal amplitudes of Dataset 1.

Algorithms	**LR**		**SVM**		**TREE**	
Hyperparameters	**C**	**solver**	**C**	**gamma**	**Max depth**	**Min samples split**
10mA	*0.0001*	*lbfgs*	*1*	*0.001*	*3*	*2*
20mA	*0.0001*	*liblinear*	*10*	*0.0001*	*1*	*1*
50mA	*0.0001*	*lbfgs*	*10*	*0.0001*	*1*	*1*
100mA	*1*	*lbfgs*	*10*	*0.001*	*3*	*2*
250mA	*0.1*	*lbfgs*	*10*	*0.001*	*1*	*1*
500mA	*0.01*	*lbfgs*	*1*	*0.01*	*1*	*1*
750mA	*1*	*lbfgs*	*1000*	*0.0001*	*5*	*7*
1A	*0.0001*	*liblinear*	*1*	*0.001*	*1*	*1*

Hyperparameters for the NN were optimized manually and are shown in Table 3. These static parameters as well as the NN architecture were consistent for all signal amplitudes.

Table 3. Optimized hyperparameters of the NN for Dataset 1.

No. of hidden layers		*2*
Hidden layer 1	**No. of nodes**	*64*
	Activation function	*Relu*
Hidden layer 2	**No. of nodes**	*32*
	Activation function	*Relu*
Output layer	**No. of nodes**	*2*
	Activation function	*Softmax*
Learning rate		*0.001*
Optimizer		*Adam*
Loss		*Sparse categorical crossentropy*
Epochs		*50*
Batch size		*16*

After optimization, all four of these algorithms were fitted with the training data for each amplitude subset (of Dataset 1). A 70:30 ratio of training to testing samples was used for all models, resulting in 560 samples for training and 240 samples for testing. After fitting, each amplitude subset was tested with the testing samples. The process was repeated ten times (with randomized training test split for each iteration) to explore the variance of the performance of these models.

The performance of the models, across all experimentation in this research, were evaluated primarily using the accuracy metric. This is because accuracy in our experimental context characterizes the Bit Error Ratio (BER), which is an important metric in digital communication. BER is the ratio of wrongly classified bits (or error bits) to the total number of transmitted or evaluated bits. Thus, BER is the "unit complement" of accuracy, i.e., BER + accuracy =100%. Therefore, higher accuracy translates to lower BER, which in turn means that communication is more efficient.

Figure 13 illustrates the test accuracies of the four models with various amplitudes contained within Dataset 1. As seen from this figure, all the models show a similar profile across the experimental amplitude range with lower test accuracies in the lower amplitudes and vice versa. The accuracies jump significantly when going from 100mA to 250mA and stay relatively stable beyond that. This jump represents the threshold amplitude necessary for the ULF-PLC signal to be transmitted in our inter-level ULF-PLC architecture. In other words, the effective "signal to noise" concept inherent in communications systems can be inferred via the ML training process.

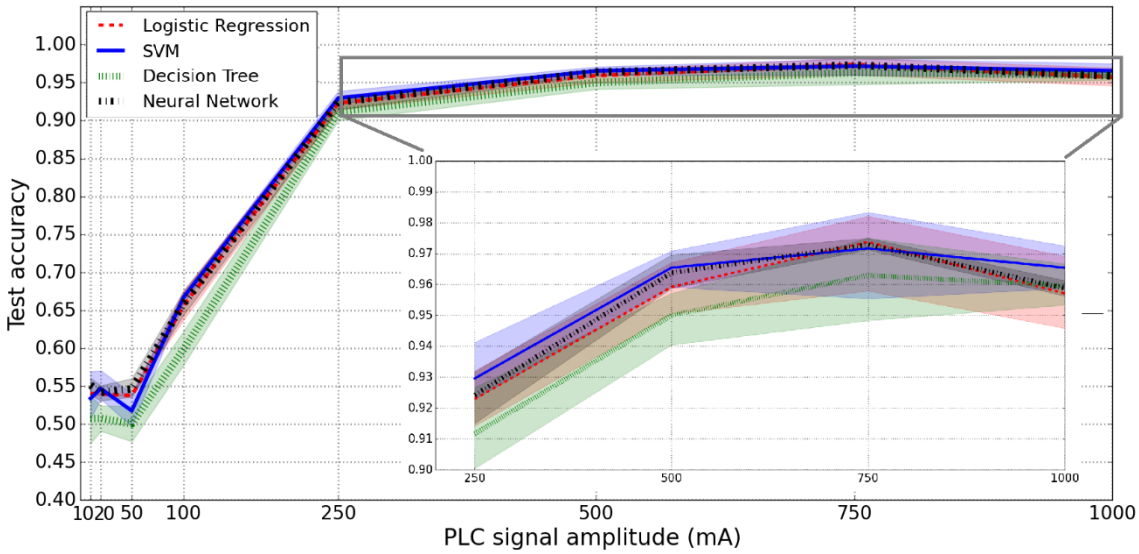


Figure 13. Graph showing test accuracy curves of Logistic Regression, SVM, Decision Tree and Neural Network models fitted with various ULF-PLC signal amplitude dataset (subsets of Dataset 1). The shadowed region of the curve represents the 95% confidence interval of the accuracies. The inset graph shows a zoomed in version of the curves for amplitudes of 250mA and higher.

The inset graph within Figure 13 shows the zoomed in version of the test accuracy curves with emphasis on transmitted signal amplitudes from 250mA to 1A. As can be seen from the experimental outcomes, SVM has slightly higher accuracy in most cases, which may be due to its non-linear classification capability. This conclusion is supported by the confusion matrices in Figure 14, which are taken from the best performing model of each algorithm fitted with 750mA amplitude dataset. The shaded region in each confusion matrix plot in Figure 14 shows the accurately classified samples whereas the non-shaded boxes show the misclassified ones from among 240 test samples. SVM has least number of misclassified samples (2) whereas Neural Network has most misclassified samples (6). LR and TREE had 3 and 5 misclassified samples, respectively. Accuracy of each model can be calculated from this confusion matrix with the formula shown in Equation 12. The difference in accuracies between the models, about 1.67% between the highest and lowest performing models shown in Figure 14, is not significant

enough to declare a clear winner among the algorithms.

$$Accuracy = 1 - \frac{Misclassified\ Samples}{Total\ Number\ of\ Samples} \tag{12}$$

This inset graph of Figure 13 also shows a slight increase in accuracies of all models as signal amplitude increases which conforms to intuition related to signal strength and SNR, but a slight decrease in accuracy above 750mA, which may be counter-intuitive. The increase in accuracy as the PLC signal amplitude increases is intuitive since we are using On/Off keying, a version of Amplitude Shift Keying (ASK), and thus, higher signal amplitude results in more distinct energy difference between the On state and the Off state, and higher SNR in a consistent channel. However, the decrease in accuracy above 750mA may be an effect of the highly nonlinear PLC channel which introduces coherent noise as a byproduct of modulation. In short, the introduction of high energy PLC signals induces higher energy replicas of introduced signal which are offset at 120Hz spacings (harmonically related to the fundamental). As a result, the PLC signal encounters additional, higher-energy coherent interference which can confound the training process. However, to examine this phenomenon, more tests in a wider range of amplitudes and frequency options is necessary.

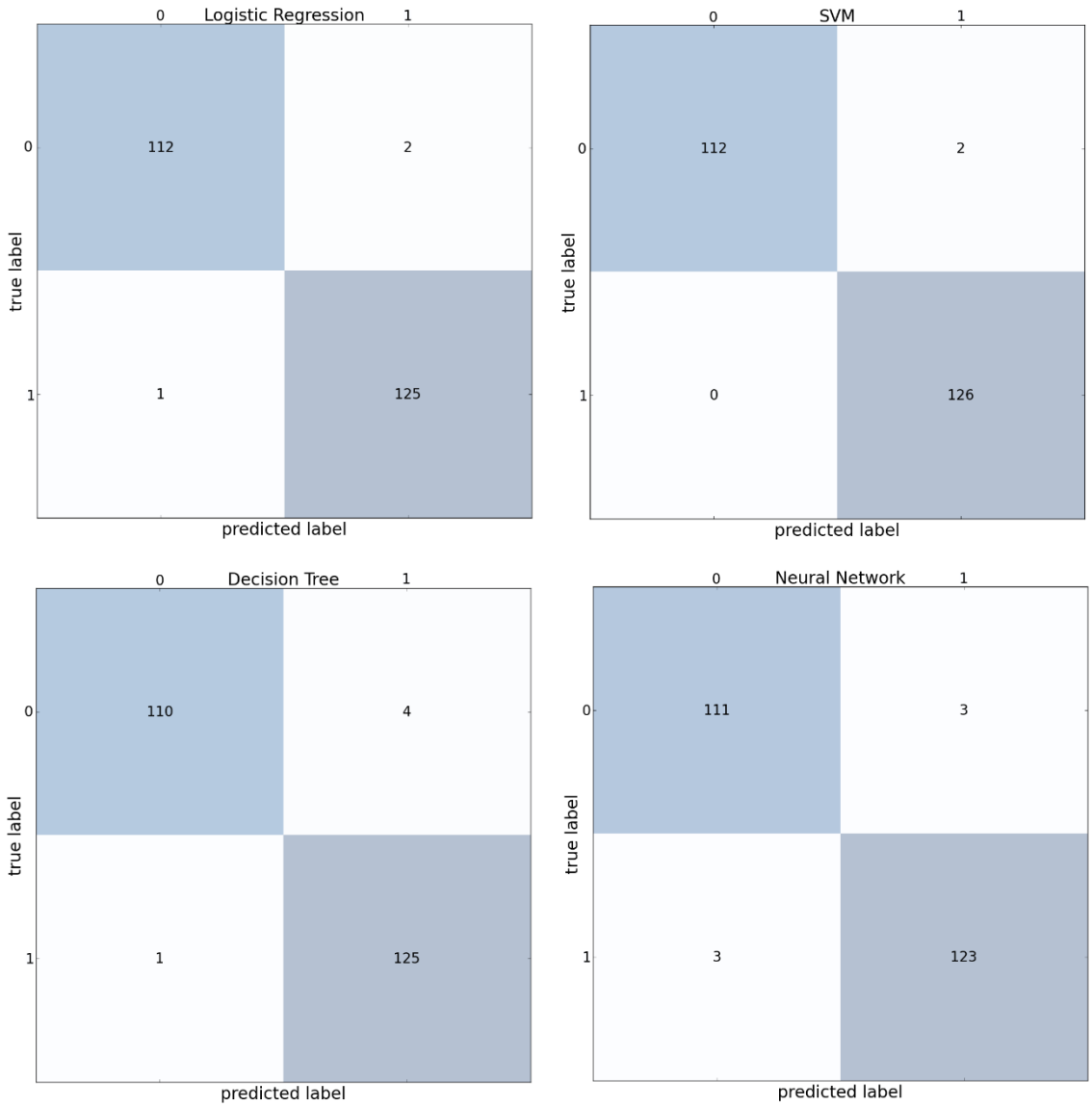


Figure 14. Figure showing confusion matrices of Logistic Regression, SVM, Decision Tree and Neural Network models fitted with 750mA subset of Dataset 1.

4.1.2. Frequency Analysis

In real-life implementation of PLC, the amplitude parameter of the PLC signal can be set to a certain value (after analysis like shown in Section 4.1.1); however, the frequency parameter is more complicated to maneuver because of the highly reactive channel. The fundamental power signal in the PLC channel oscillates close to 60Hz and produces strong odd harmonics at 120Hz spacings. These dominant signals, therefore,

occupy a wide range of frequencies in the low-frequency PLC spectrum. This problem is exacerbated by the variable noise of the channel which can present time-variant interference to the PLC signals. Therefore, transmitting a PLC signal in these low frequency bands can be challenging. To address these issues, 'Dataset 2' was used in three different ways, which are described in in Sections 4.1.2.1 through 4.1.2.3.

4.1.2.1. Case 1- Known and Static Frequency

Case 1 simulates the scenario in which the frequency band to send the PLC signal is known and always the same. This would require a great deal of knowledge of the channel, so that the PLC frequency band does not overlap with the pre-existing harmonics (or the noise). Therefore, this is an unlikely scenario; however, if implemented correctly, is the easiest for the receiver to process.

To recreate this scenario, we divided 'Dataset 2' into individual frequency subsets (like amplitudes for 'Dataset 1' in Section 4.1.1) and used them as an expanded training set for the ML/NN. Similar to 'Dataset 1,' these subsets of 'Dataset 2' were standardized and fitted with LR, SVM, TREE and NN algorithms after hyperparameter optimization. These optimized hyperparameters are shown in Table 4. The model architecture for the NN was the same as for 'Dataset 1,' and thus, the hyperparameters are the same as listed in Table 3.

Table 4. Optimized hyperparameters of LR, SVM, and TREE for frequency subsets of Dataset 2 (Case 1).

Algorithms	LR		SVM		TREE	
Hyperparameters	**C**	**solver**	**C**	**gamma**	**Max depth**	**Min samples split**
690Hz	*0.0001*	*lbfgs*	*0.0001*	*0.0001*	*1*	*1.0*
750Hz	*10*	*lbfgs*	*10*	*0.0001*	*1*	*1.0*
810Hz	*10*	*lbfgs*	*10*	*0.0001*	*2*	*2*
870Hz	*10*	*lbfgs*	*1000*	*0.0001*	*4*	*2*
930Hz	*0.1*	*lbfgs*	*100*	*0.01*	*3*	*2*
990Hz	*0.1*	*lbfgs*	*100*	*0.001*	*1*	*1.0*
1050Hz	*0.01*	*lbfgs*	*10*	*0.001*	*1*	*1.0*
1110Hz	*0.001*	*liblinear*	*1*	*0.01*	*3*	*2*
1170Hz	*0.0001*	*liblinear*	*10*	*0.01*	*1*	*1.0*
1230Hz	*0.001*	*lbfgs*	*1*	*0.01*	*1*	*1.0*
1290Hz	*1000*	*liblinear*	*0.1*	*0.01*	*1*	*1.0*
1350Hz	*0.1*	*lbfgs*	*1000*	*0.01*	*1*	*1.0*
1410Hz	*0.1*	*lbfgs*	*1*	*0.01*	*1*	*1.0*
1470Hz	*0.01*	*liblinear*	*1*	*0.01*	*3*	*2*
1530Hz	*0.001*	*lbfgs*	*10*	*0.0001*	*1*	*1.0*
1590Hz	*0.01*	*lbfgs*	*100*	*0.001*	*1*	*1.0*
1650Hz	*0.1*	*liblinear*	*10*	*0.001*	*1*	*1.0*
1710Hz	*1*	*lbfgs*	*10*	*0.001*	*3*	*2*
1770Hz	*1*	*lbfgs*	*1*	*0.01*	*1*	*1.0*
1830Hz	*1*	*lbfgs*	*10*	*0.0001*	*1*	*1.0*
1890Hz	*1*	*liblinear*	*1*	*0.001*	*5*	*5*
1950Hz	*0.1*	*lbfgs*	*100*	*0.0001*	*2*	*2*
2010Hz	*0.1*	*lbfgs*	*10*	*0.0001*	*4*	*2*

The training to test ratio for each experiment was set to 70:30. Since, each frequency subset was trained and tested separately, there were 560 training samples and 240 test samples for every Case 1 experiments. After fitting the models with these training data, the models were tested with test data. Figure 15 shows the accuracies (mean accuracy curves and the 95% confidence interval from ten iterations) of the four models from these test data at various frequencies. The inset graph within Figure 15 shows the frequencies, from 930Hz to 1650Hz, where the model test accuracies were consistently

over 90%.

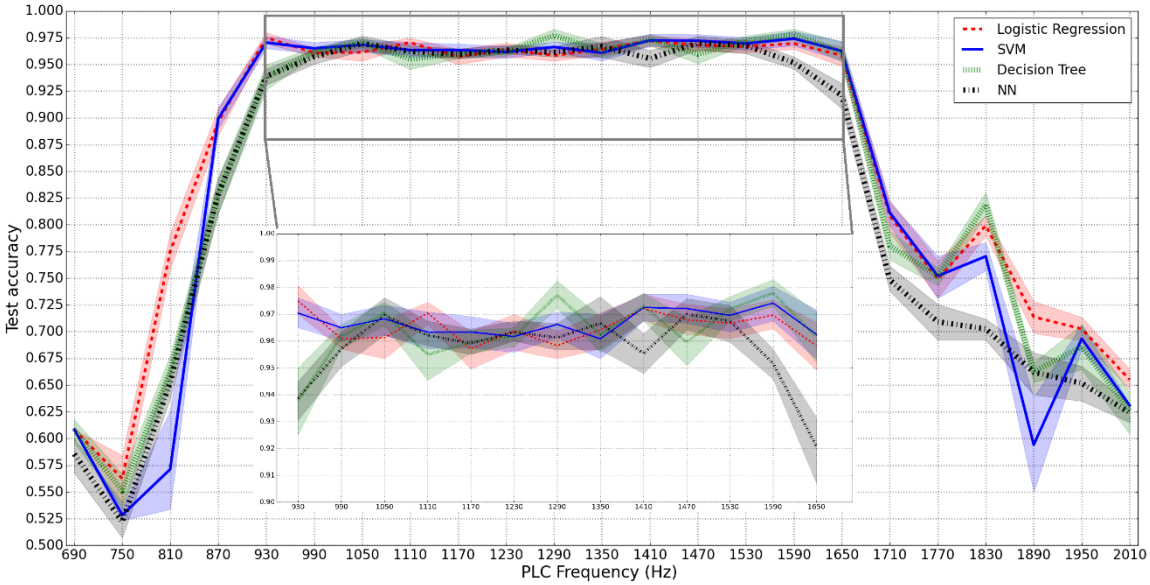


Figure 15. Graph showing test accuracy curves of Logistic Regression, SVM, Decision Tree and Neural Network models fitted with various ULF-PLC signal frequency dataset (subsets of Dataset 2). The shadowed region of the curve represents the 95% confidence interval of the accuracies. The inset graph shows a zoomed in version of the curves for frequencies from 930Hz to 1650Hz.

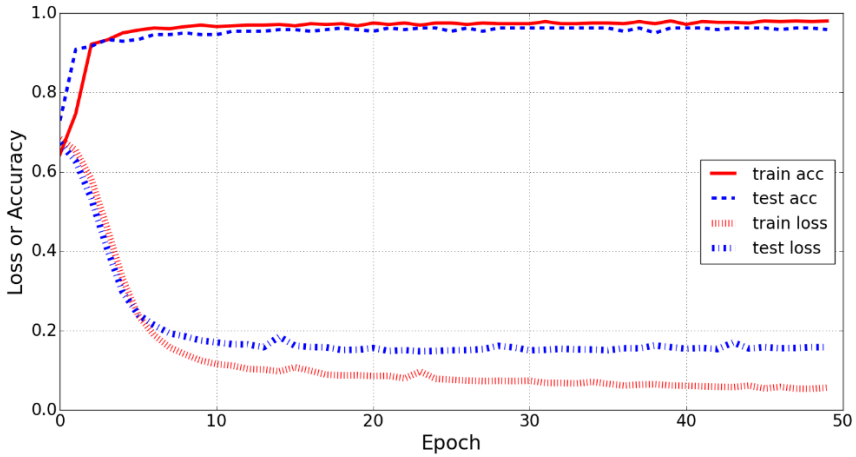
As seen in Figure 15, the lower and higher end of the experimental spectrum both have low test accuracies across all models. This is due to the bandpass filtering nature of the channel for the PLC signals as well as the limits of the coupling filter. At lower range of frequencies (below 870Hz in the graph), the harmonics of the fundamental power signal is very strong, and thus, the PLC signal is severely distorted. As frequency increases, these harmonics die off, opening dynamic subchannels into which the PLC signals can be introduced. However, beyond a certain frequency (after 1710Hz in the graph), the PLC signals are heavily attenuated by the transformers and other grid components, including the coupling filter, causing poor PLC output at the receiver. Therefore, there is a frequency window where the PLC signal transmission is optimal. Figure 15 shows that, in our case, this window is between 930Hz to 1650Hz. Within this window, the test accuracies are fairly high (mostly above 95%) and stable for all models,

as shown by the inset graph in Figure 15.

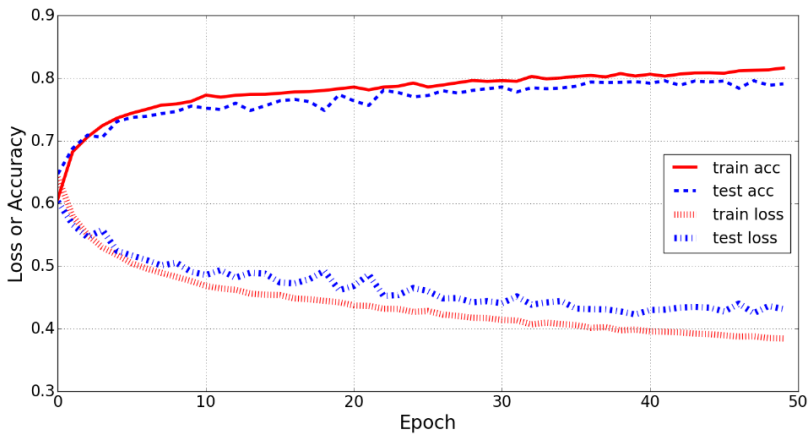
In summary, this Case 1 experimentation shows that with sufficient knowledge of the channel and with a set known PLC frequency, upwards of 95% accuracies can be obtained using feature-based ML/NN. As stated in Section 4.1.1., high accuracy of the models means low BER which translates to high efficiency in communication.

4.1.2.2. Case 2- Known but Dynamic Frequency

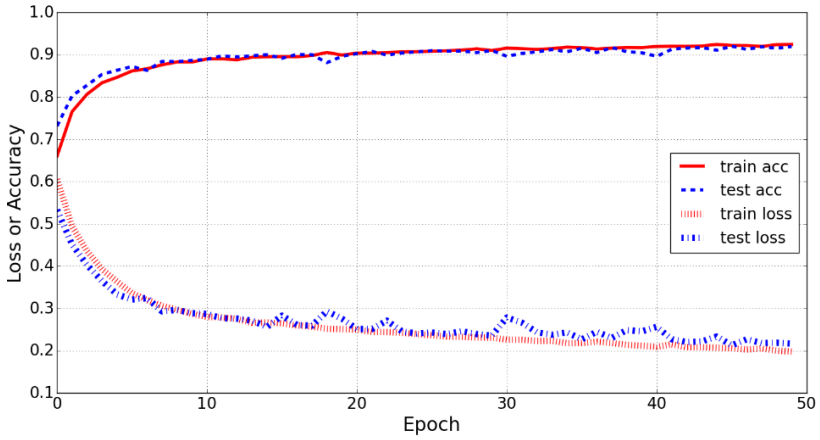
Case 2 is similar to Case 1 in the sense that it simulates a scenario where the PLC frequency is known. However, in this case this frequency is not constant, but changes frequently to some known set of values (e.g., frequency hopping). This case is more realistic than Case 1 because it accounts for the variable nature of the channel and the ability to dynamically select (or pre-select) open subsections of the low-frequency spectrum. However, this approach still relies on the assumption that the at least some of the frequency bands or sub-channels in the pre-defined set is always optimal for transmission. As a result, effective implementation of a frequency-hopped low-frequency PLC system would depend in large part on dynamic spectral analysis of the channel.
To simulate this case, we took 'Dataset 2' as a whole for training and testing, instead of separating it with respect to frequency as described for Case 1 in Section 4.1.2.1. Hence, 'Dataset 2' was randomly split into training and test set, and the ML/NN models were fitted with the training set and evaluated with the test set. The dataset is more complex in this case for ML/NN to generalize because the PLC frequency is not the same throughout, and thus, the weight for the feature column keeps changing. This results in slower convergence than Case 1, as shown in Figure 16.



(a) Case 1 at frequency of 1290Hz



(b) Case 2 with full 'Dataset 2'



(c) Case 2 with trimmed 'Dataset 2'

Figure 16. Graphs showing training and test (or validation) accuracy and loss curves of identical NN for (a) Case 1 with frequency of 1290Hz, (b) Case 2 with full dataset, and (c) Case 2 with trimmed dataset. The trimmed dataset for (c) is from 870Hz to 1710Hz.

As seen in Figure 16 (a), the accuracy and loss curves of both training and test set converge within the first few epochs for Case 1. However, for Case 2, the convergence takes more epochs. Figure 16 (b) is for the Case 2 dataset with full range of frequencies, i.e., the complete 'Dataset 2.' This plot shows that not only do the loss and accuracy curves not converge as fast as in Case 1, but these curves actually do not stabilize within the 50 epochs. In addition, the accuracies are much lower at the end of the training than in Case 1 (approximately 80% compared to around 95% for Case 1). This is because of the presence of 690Hz-810Hz and 1770Hz-2010Hz frequency data within 'Dataset 2' which contains distorted PLC signals as shown in Figure 15. The data corresponding to these frequencies dilute the dataset, thereby causing low accuracies in Case 2. Therefore, for a fairer comparison, we took out these frequency data from the dataset for further Case 2 analysis, resulting in a "trimmed" dataset for Case 2. Figure 16 (c) shows the accuracy and loss curves of this trimmed 'Dataset 2.' In addition to the increased accuracies, this plot shows that the accuracy and loss curves do stabilize.

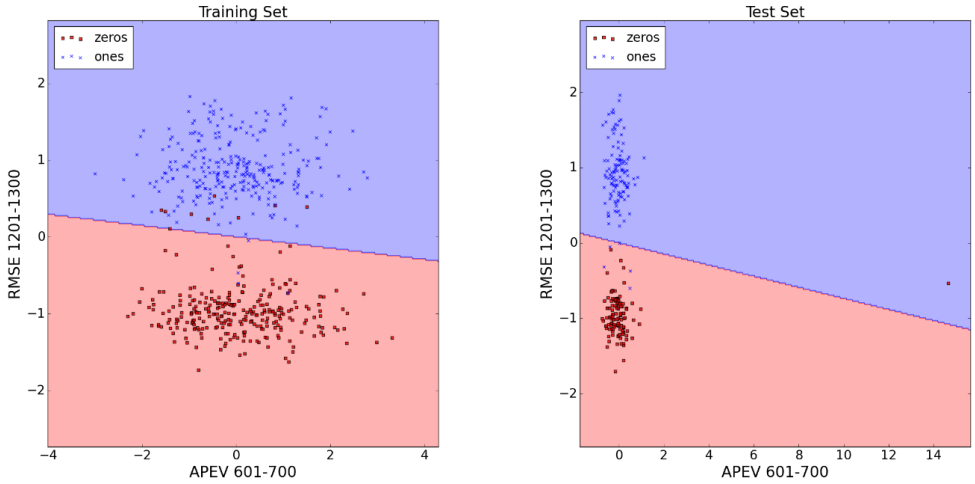
After trimming 'Dataset 2' to include only the frequency data from 870Hz-1710Hz (i.e., the passband of our PLC channel), next we fitted the four algorithms with this updated dataset. The training and test accuracies, precision, recall and F1 scores of these models are listed in Table 5.

Table 5. Performance of LR, SVM, TREE and NN with trimmed 'Dataset 2.' The values shown in the table are the mean +/- standard deviation from ten iterations.

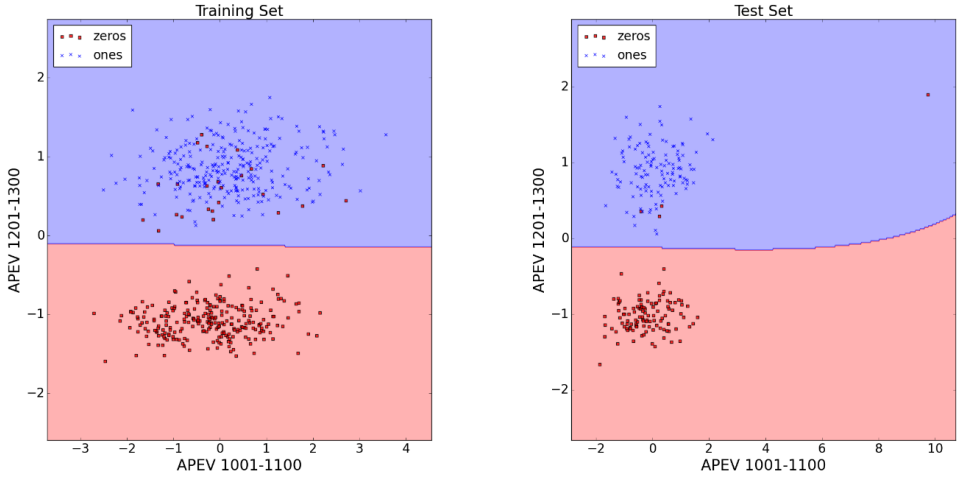
Algorithms	Training accuracy	Testing accuracy	Precision	Recall	F1 score
LR	*0.81790 +/- 0.00328*	*0.81870 +/- 0.00398*	*0.84334 +/- 0.00522*	*0.80534 +/- 0.00808*	*0.82387 +/- 0.00429*
SVM	*0.92355 +/- 0.00293*	*0.90826 +/- 0.00388*	*0.93420 +/- 0.00595*	*0.88838 +/- 0.00473*	*0.91070 +/- 0.00370*
TREE	*0.90768 +/- 0.00478*	*0.86604 +/- 0.00980*	*0.91220 +/- 0.02532*	*0.82626 +/- 0.02008*	*0.86661 +/- 0.00889*
NN	*0.90571 +/- 0.00313*	*0.90078 +/- 0.00409*	*0.91201 +/- 0.01740*	*0.89916 +/- 0.02417*	*0.90507 +/- 0.00517*

As seen in Table 5, SVM had the highest scores in all performance metrics for this trimmed 'Dataset 2': Case 2. TREE and NN displayed similar performance, while LR outcomes were substantially worse with approximately 10% difference in every metric compared to SVM.

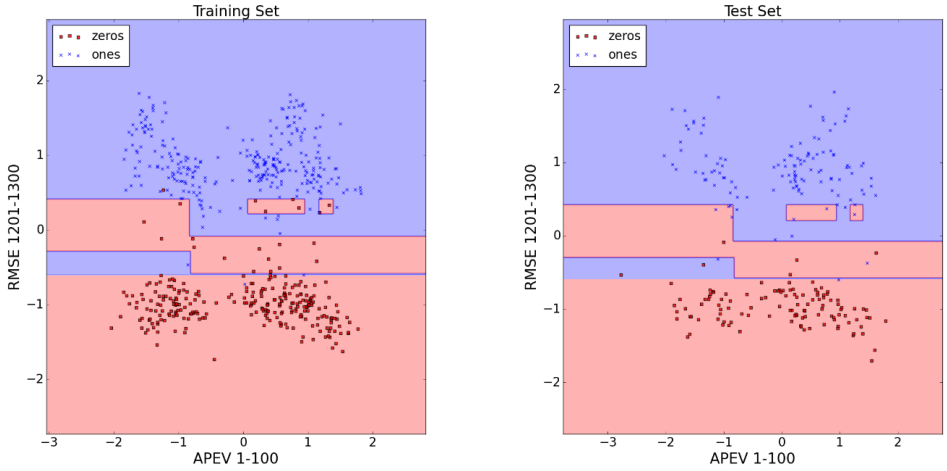
To analyze the possible causes of this discrepancy between the various algorithms, we constructed a 2D plot with decision regions for each of these models. The two feature columns for these 2D plots were selected using Sequential Backward Analysis (SBS) to give two most influential feature columns for each model (described in more detail in Section 4.1.3). Figure 17 shows these plots for LR, SVM and TREE models. As seen in Figure 17, LR displays a linear and continuous boundary, while the decision boundary of SVM is continuous and curved, and the decision boundaries of TREE are straight but discontinuous. These decision boundaries separate the two classes (i.e., On and Off or 1 and 0); therefore, the more flexible these boundaries are, the better the accuracy of the models' classification is going to be. The linearity of LR can sometimes prevent it from fitting some training data, whereas the SVM and TREE are more flexible, and thus, can have better performance, i.e., higher classification accuracy. However, SVM and TREE are more prone to overfitting for the same reason. In our current case, i.e., Case 2, there is no overfitting (or underfitting) on any of our models as shown by the small difference between training and testing accuracies.



(a) LR



(b) SVM



(c) TREE

Figure 17. Plots showing training and test data, plotted over two feature columns, along with the decision boundary and shaded target regions of (a) LR (b) SVM and (c) TREE models.

4.1.2.3. Case 3- Unknown and Dynamic Frequency

Case 3 simulates a scenario in which the PLC frequency band is unknown as well as dynamic. In this case, the ML/NN is trained with a range of frequencies, and the test is done with some frequency within or close to this range, but not present in the training set. Therefore, the only prior information needed in this scenario for real-life implementation is the frequency bound within which the PLC signal is likely to be in and a training set within this bound. The expectation is that the ML/NN will train within this range and be able to generalize well enough to classify signals in frequencies it never trained on. To simulate this scenario from our dataset, we separated 'Dataset 2' into individual frequency subsets, used all except one subset for training, and tested on the unused one. Our objective with this case was to test how well ML/NN models can generalize on data with untrained PLC frequency and to observe what effect the frequency has on this test. As with Case 2, including the complete 'Dataset 2' for this case would dilute the dataset and cause low accuracies. In practical use cases, the frequency bounds can be tested and made sure they fall within the bandpass range of the channel. Therefore, for Case 3, we used trimmed 'Dataset 2' as we did in Case 2 in Section 4.1.2.2.

Figure 18 shows the test accuracies of this experiment for the four algorithms. The frequencies on the X-axis are the frequencies of the test data. For example, for X=1170Hz, the ML/NN models were trained on trimmed 'Dataset 2' apart from the subset where PLC frequency was 1170Hz. After training, the test was done with this subset and the test accuracy values are plotted in Figure 18.

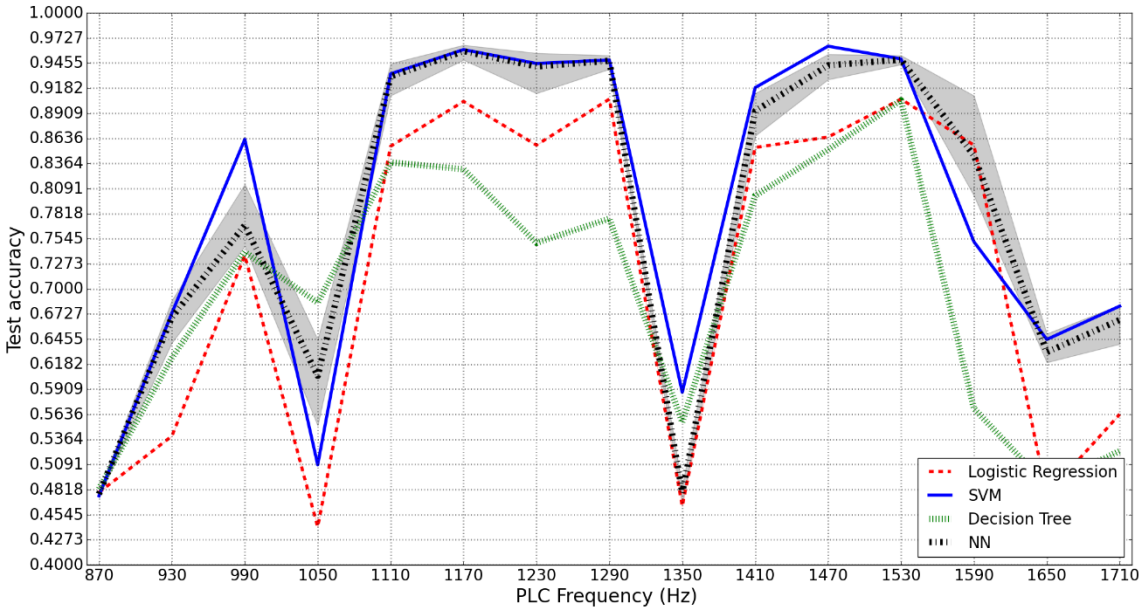


Figure 18. Test accuracies for Logistic Regression, SVM, Decision Tree and Neural Network for Dataset 2: Case 3. The PLC frequency on the x-axis represents the test frequency.

As seen in Figure 18, all of the algorithms had some success in generalizing to the test data with untrained PLC frequency characteristics. However, the performance is not consistent for any algorithms across the whole spectrum. The low accuracies at the extremes, i.e., 870Hz and 1710Hz can be explained by their training data. Their respective models are trained only on the frequencies higher or lower from the testing frequency. Therefore, the model only has the upper or lower bound subset to generalize the test set. This causes low accuracy. Extending this similarity measure argument to the whole dataset would mean that the frequency subsets in the middle of range would have the highest accuracy and the accuracy decreases as we move to either side from the middle, producing a bell-shaped accuracy profile. The curves in Figure 18 promise a similar trend; however, this bell-shaped profile is broken by two big dips at 1050Hz and 1350Hz. Upon closer examination of these models at these two frequencies, we found out that these dips are caused largely due to the robustness of the models themselves. The

architecture and hyperparameters of each of these models for all frequencies are kept the same throughout this case experiment, i.e., the models are not optimized for specific training set. This was done to simulate the real-life implementation of this case, where the models cannot be optimized for the unknown test frequency. Therefore, our models were not optimized for certain test sets, and they failed at those frequencies. Despite this setback, the models showed high accuracy in majority of the test frequencies, and thus, showed promising generalizing behavior. Further optimization of these models or using more complex models can rectify the vulnerabilities, and the practical implementation of Case 3 may be possible.

4.1.3. Feature Analysis

In addition to analyzing the amplitude and frequency of the PLC signal, we also wanted to investigate which features were the best for our PLC data. To do this, we used the Sequential Backwards Selection (SBS) technique to reduce the dimensionality of our datasets from N=61 to N=2, where 'N' is the number of feature columns in our dataset. We chose N=2 for our new dataset so that we could produce 2D plots of the dataset as shown in Figure 17. We performed SBS for 'Dataset 1' and for Case 1 of 'Dataset 2,' the results of which are shown in Table 6 and 7, respectively. We did not conduct SBS for Cases 2 and 3 of 'Dataset 2' because in these cases, multiple frequency subsets were used for training, and subsequently, the most impactful features would change multiple times during training.

Table 6. The two most impactful features for various amplitude subsets of 'Dataset 1' along with test accuracies of LR, SVM and TREE models on the full N=61 dataset and reduced N=2 dataset. APEV and RMSE denotes amplitude envelope and RMS energy features, respectively.

PLC Amp [mA]	Feature 1	Feature 2	LR Test Acc		SVM Test Acc		TREE Test Acc	
			N=61	N=2	N=61	N=2	N=61	N=2
10	*APEV 1-100*	*APEV 101-200*	*0.5417*	*0.5375*	*0.5375*	*0.5375*	*0.4833*	*0.5541*
20	*RMSE 601-700*	*RMSE 801-900*	*0.5208*	*0.5416*	*0.5417*	*0.5375*	*0.5167*	*0.5416*
50	*APEV 1-100*	*APEV 101-200*	*0.5375*	*0.5375*	*0.5417*	*0.5375*	*0.4667*	*0.4625*
100	*APEV 1101-1200*	*RMSE 1101-1200*	*0.6625*	*0.5083*	*0.6750*	*0.6291*	*0.5750*	*0.5166*
250	*APEV 1101-1200*	*APEV 1201-1300*	*0.9083*	*0.925*	*0.9208*	*0.9125*	*0.9042*	*0.7208*
500	*APEV 601-700*	*APEV 1101-1200*	*0.9667*	*0.9583*	*0.9667*	*0.9583*	*0.9542*	*0.9541*
750	*APEV 1001-1100*	*APEV 1101-1200*	*0.9708*	*0.9708*	*0.9750*	*0.9708*	*0.9542*	*0.9583*
1000	*APEV 1-100*	*APEV 1101-1200*	*0.9500*	*0.9625*	*0.9708*	*0.75*	*0.9500*	*0.9625*

In Table 6, columns 'Feature 1' and 'Feature 2' list the two most impactful features (for the corresponding amplitude subset in the row) gotten from SBS analysis. These two features were common for all three algorithms shown for a particular amplitude data subset, which indicates that even though the three algorithms work in different ways to optimize weights for each column (to generate ML models), eventually these weights were spread out in a similar fashion for all three models. Therefore, the impactful features are, in some respect, independent of the algorithms.

In Table 6, the two optimal features are named either APEV for 'Amplitude Envelope' or RMSE for 'Root Mean Square Energy,' while the range after these feature names denotes the frequency of the filtered signals. Evaluating these two features, we can see that, for PLC amplitudes above 100mA, APEV 1101-1200 (as shown by the green shaded cells in Table 6), i.e., amplitude envelope of the signal after filtering with 1101Hz to 1200Hz bandpass filter, is one of the two SBS features. This is significant because the

PLC frequency of all these signals is 1170Hz. Therefore, the ML models were able to identify the PLC frequency from the data and relate the presence and absence of the signal in this frequency to the On/Off state. This also means that SBS analysis could be used to identify certain signal characteristics, like frequency in our case, from the feature dataset. For amplitudes below 100mA, the PLC signals were distorted as shown in Figure 13. Therefore, the ML models for these amplitudes did not get to the same conclusion in identifying the optimal features.

Table 6 also lists the test accuracies of the three models with the full feature dataset (N=61) and SBS reduced dataset (N=2). Comparing these accuracies shows that in most cases the SBS reduced models perform as well as the full feature models. This can be attributed to the major impact these features have on the ML models. In some cases (such as LR for 250mA), the SBS reduced model's accuracy is even higher than the full feature models. This may be due to the weights becoming spread out over numerous non-impactful features in the full dataset which causes the impactful features to have lower weights compared to the SBS reduced dataset, thereby leading to lower accuracy.

Table 7 shows optimal features and accuracies for 'Dataset 2.' A similar conclusion can be drawn from this table as Table 6. One key difference here is that there is no common feature for all frequency subsets. This is because the PLC frequency changes when switching subsets, and thus, the impactful features keep changing. However, in most cases the PLC frequency falls within the bandpass frequency range of the RMS energy feature, as shown by the green shaded cells. Therefore, for this dataset, the RMS energy feature provides information about the signal frequency.

Table 7. Two most impactful features for various frequency subsets of 'Dataset 2' along with test accuracies of LR, SVM and TREE models on the full N=61 dataset and reduced N=2 dataset.

PLC Freq [Hz]	Feature 1	Feature 2	LR Test Acc		SVM Test Acc		TREE Test Acc	
			N=61	N=2	N=61	N=2	N=61	N=2
690	*APEV 1-100*	*APEV 101-200*	*0.6083*	*0.5917*	*0.6083*	*0.5917*	*0.6083*	*0.5917*
750	*APEV 701-800*	*APEV 1201-1300*	*0.5833*	*0.5250*	*0.5500*	*0.5250*	*0.5625*	*0.5250*
810	*APEV 101-200*	*RMSE 801-900*	*0.8167*	*0.6750*	*0.6875*	*0.6750*	*0.6583*	*0.6750*
870	*RMSE 701-800*	*RMSE 801-900*	*0.9000*	*0.8333*	*0.9042*	*0.8333*	*0.8292*	*0.8333*
930	*RMSE 901-1000*	*RMSE 1001-1100*	*0.9750*	*0.9750*	*0.9625*	*0.9750*	*0.9333*	*0.9750*
990	*APEV 1-100*	*RMSE 901-1000*	*0.9708*	*0.9750*	*0.9792*	*0.9750*	*0.9750*	*0.9750*
1050	*APEV 1-100*	*RMSE 1001-1100*	*0.9333*	*0.9500*	*0.9500*	*0.9500*	*0.9583*	*0.9500*
1110	*APEV 1-100*	*RMSE 1101-1200*	*0.9667*	*0.9625*	*0.9583*	*0.9625*	*0.9542*	*0.9625*
1170	*APEV 1-100*	*APEV 1101-1200*	*0.9500*	*0.9625*	*0.9667*	*0.9625*	*0.9500*	*0.9625*
1230	*APEV 1201-1300*	*RMSE 1201-1300*	*0.9625*	*0.9792*	*0.9750*	*0.9792*	*0.9708*	*0.9792*
1290	*APEV 601-700*	*RMSE 1201-1300*	*0.9583*	*0.9750*	*0.9750*	*0.9750*	*0.9833*	*0.9750*
1350	*APEV 1201-1300*	*APEV 1301-1400*	*0.9458*	*0.9458*	*0.9500*	*0.9458*	*0.9417*	*0.9458*
1410	*APEV 1-100*	*APEV 1401-1500*	*0.9667*	*0.9500*	*0.9625*	*0.9500*	*0.9583*	*0.9500*
1470	*RMSE 1301-1400*	*RMSE 1401-1500*	*0.9667*	*0.9667*	*0.9625*	*0.9667*	*0.9667*	*0.9667*
1530	*APEV 1401-1500*	*RMSE 1501-1600*	*0.9542*	*0.9542*	*0.9708*	*0.9542*	*0.9667*	*0.9542*
1590	*APEV 1-100*	*RMSE 1501-1600*	*0.9625*	*0.9833*	*0.9583*	*0.9833*	*0.9875*	*0.9833*
1650	*APEV 1-100*	*RMSE 1601-1700*	*0.9333*	*0.9458*	*0.9417*	*0.9458*	*0.9458*	*0.9458*
1710	*RMSE 1601-1700*	*RMSE 1701-1800*	*0.8333*	*0.8292*	*0.8208*	*0.8292*	*0.7708*	*0.8292*
1770	*RMSE 1701-1800*	*RMSE 1801-1900*	*0.7125*	*0.7292*	*0.6708*	*0.7292*	*0.7208*	*0.7292*
1830	*RMSE 1801-1900*	*RMSE 1901-2000*	*0.7958*	*0.8083*	*0.7667*	*0.8083*	*0.8000*	*0.8083*
1890	*RMSE 1201-1300*	*RMSE 1801-1900*	*0.7542*	*0.7042*	*0.7083*	*0.7042*	*0.6708*	*0.7042*
1950	*RMSE 1201-1300*	*RMSE 1901-2000*	*0.6917*	*0.7042*	*0.6958*	*0.7042*	*0.7000*	*0.7042*
2010	*APEV 1501-1600*	*RMSE 2001-2100*	*0.6375*	*0.6708*	*0.6292*	*0.6708*	*0.5833*	*0.6708*

4.2. Featureless Dataset

In addition to using the traditional feature dataset in ML/NN, we also investigated the use of featureless datasets. We did so by constructing 1D time-series, 2D magnitude spectrogram and 3D rectangular spectrogram datasets, as discussed in Chapter III. NNs

were used on each dataset because of their ability to take in multi-dimensional data. The multi-neuronal architecture of NNs is also more complex than one-neuron structures of LR, SVM or TREE, which could aid in parsing through our noisy featureless datasets. The structure and hyperparameters of the NN for the time-series and magnitude spectrogram dataset were identical to the NN used for feature dataset (see Table 3). For the rectangular spectrogram though, an additional convolution layer and a max pool layer were added before the fully connected NN layers. The complete architecture of this convolutional neural network (CNN) is shown in Figure 19.

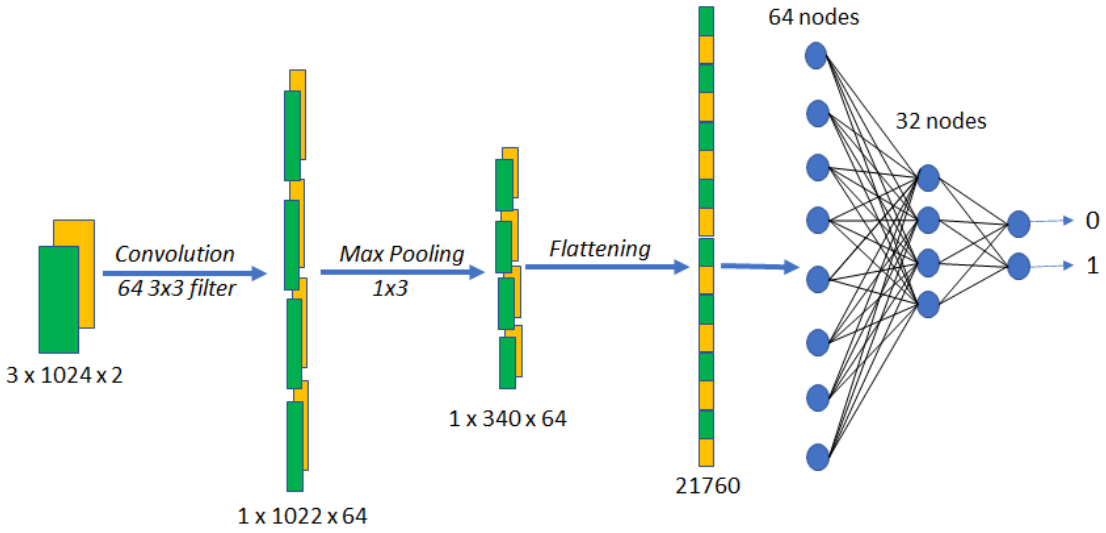


Figure 19. CNN architecture showing the various layers of operation on our rectangular spectrogram dataset.

As seen in Figure 19, each sample of the rectangular spectrogram is 3x1024x2, where 2 is the number of channels (i.e., stacked real spectrogram and imaginary spectrograms). The first operation is convolution with 64 3x3 filters producing 1x1022x64 feature maps. Then comes the max pool layer with pool size 1x3 which produces output of 1x340x64. This output is flattened to 1D array of size of 21760. The flattened feature maps are then fed into the fully connected NN. This NN is identical to

the ones used in feature dataset (see Table 3). The results of amplitude and frequency analysis of these three datasets are described in Section 4.2.1. and Section 4.2.2. respectively.

4.2.1. Amplitude Analysis

From 'Raw Set 1,' magnitude spectrogram, rectangular spectrogram and time-series datasets were generated to create 'Dataset 3,' 'Dataset 5' and 'Dataset 7' respectively. Figure 20 shows the test accuracy curves of NN/CNN models fitted with these datasets.

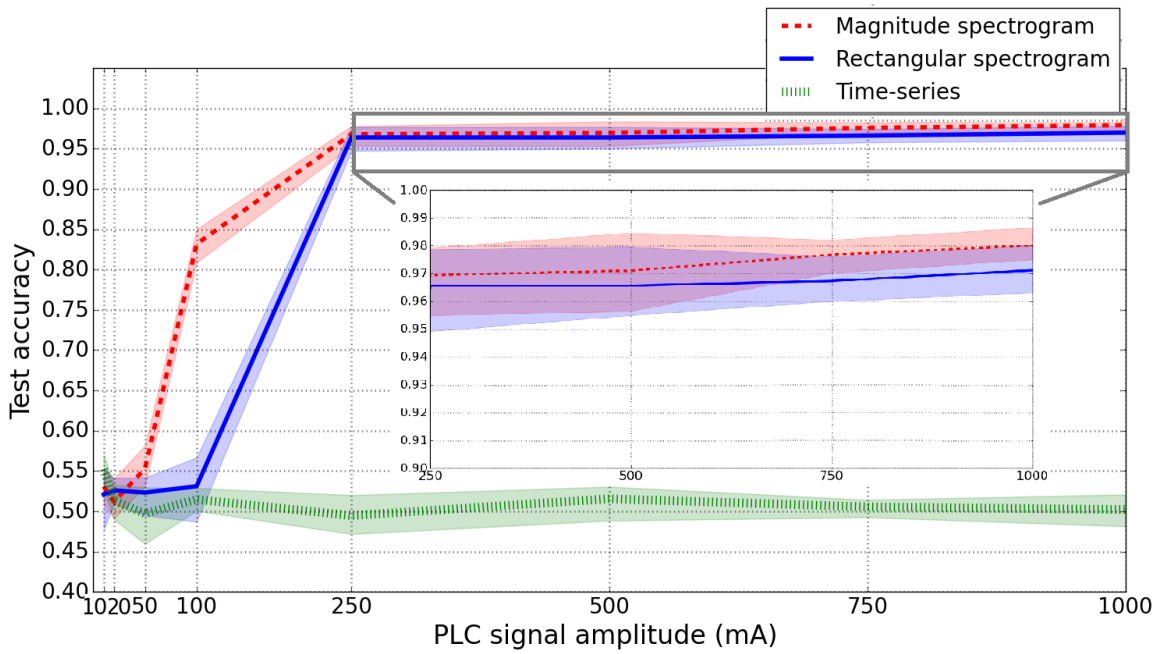


Figure 20. Graph showing test accuracy curves (with 95% confidence interval) of NN models fitted with magnitude spectrogram (Dataset 3), rectangular spectrogram (Dataset 5) and time-series datasets (Dataset 7) containing ULF-PLC signals of various amplitudes. The inset graph shows a zoomed in version of the curves for amplitudes of 250mA and higher.

As seen from the low accuracy curve of the time series in this figure, the time series dataset could not classify our On/Off keyed PLC signals effectively. The accuracy hovering around 50% for this dataset suggests the NN model could not find any relevant feature maps from this dataset. This result was expected since the unprocessed time series

data, as shown in Figure 7, are dominated by the power signal.

The accuracy curve profiles of NN/CNN models of magnitude and rectangular spectrograms look similar to the feature-based dataset models in Figure 13, as these datasets were generated from the same 'Raw Set 1.' As concluded from Figure 13, Figure 20 also shows that increasing the amplitude above 250mA is optimal for PLC, with the threshold being somewhere between 100mA and 250mA.

Comparing between the magnitude spectrogram and rectangular spectrogram, we expected close performance between these two datasets, as discussed in Section 3.2.2. The inset graph of Figure 20, which shows the NN model accuracies for these two datasets between 250mA and 1A, supports our intuition. The difference in accuracies between the two spectrogram models within this amplitude range is less than 1%. This means that both datasets are equally efficient for PLC in this amplitude range. However, a considerable difference in accuracies can be seen at 100mA PLC signal amplitude in Figure 20. The magnitude spectrogram model has classification accuracy of approximately 85% while the rectangular spectrogram model gets only up to about 55% at this amplitude. This could be because magnitude spectrogram is more efficient in retaining the amplitude information than rectangular spectrogram, as explained in Section 3.2.2. Therefore, the more efficient magnitude spectrogram model has lower amplitude threshold (for high classification accuracy) than the rectangular spectrogram. More data in the intermediate amplitudes between 100mA and 250mA are necessary to validate this conclusion.

4.2.2. Frequency Analysis

For frequency analysis, ‘Raw Set 2’ was transformed to magnitude spectrogram, rectangular spectrogram and time-series datasets creating ‘Dataset 4,’ ‘Dataset 6’ and ‘Dataset 8’ respectively. As in the cases using feature-based datasets, these three datasets were evaluated with three case scenarios in sections 4.2.2.1 through 4.2.2.3

4.2.2.1. Case 1- Known and Static Frequency

In Case 1, the NN/CNN is trained and tested within a particular frequency subset. The test accuracy results of the NN/CNN models are presented in Figure 21. As with amplitude datasets, the time-series dataset had the worst performance, as expected due to the strong out of band interference. The magnitude spectrogram models exhibited better overall performance, and particularly better accuracy than the rectangular spectrogram models, especially in the lower and the higher end of the spectrum. The magnitude spectrogram models also had a much narrower and more consistent confidence interval than the rectangular spectrogram. The confidence interval result is especially interesting since a larger confidence interval implies larger variance, and hence, the rectangular spectrogram dataset seems to have higher variance in its accuracy results. Higher variance suggests more unpredictability and less repeatability in the performance which is undesirable.

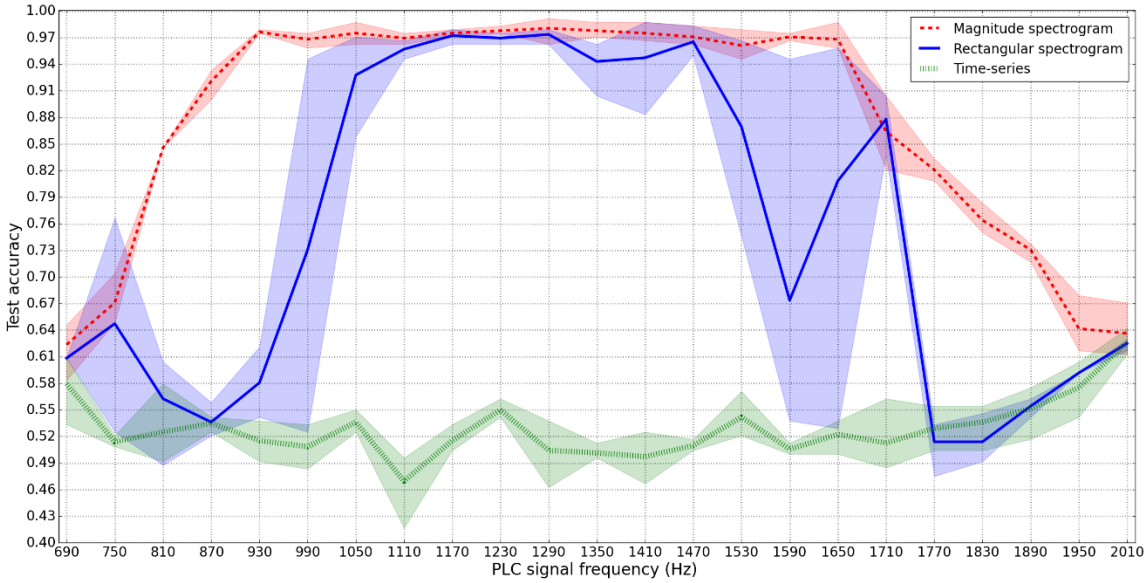


Figure 21. Graph showing test accuracy curves (with 95% confidence interval) of NN models fitted with magnitude spectrogram (Dataset 4), rectangular spectrogram (Dataset 6) and time-series (Dataset 8) datasets containing ULF-PLC signals of various frequencies.

This discrepancy in performance between magnitude spectrogram models and rectangular spectrogram models can be explained by the type and quantity of information each dataset contains. As described in Section 3.2.2., magnitude spectrogram, by definition, contains only the magnitude or energy information of the signal, which is directly related to the signal amplitude. So, for ASK signals, the magnitude spectrogram more clearly represents modulation transitions, thus simplifying the task of the NN. In contrast, the rectangular spectrogram holds more information about the signal, including phase data, which could be advantageous in some use cases. However, in the approaches discussed here, the larger size of the rectangular dataset causes data dilution resulting in lower performance compared to magnitude spectrogram models.

4.2.2.2. Case 2- Known but Dynamic Frequency

As in Case 2 of the feature-based ML/NN analysis (Section 4.1.2.2.), NN/CNN is trained with the trimmed featureless datasets (without separating frequency subsets). The

objective here is to explore which featureless datasets perform the best when trained with samples containing PLC signals of varying frequencies. Table 8 summarizes the result of this experimentation (with ten iterations of each model).

Table 8. Performance of magnitude spectrogram (Dataset 4), rectangular spectrogram (Dataset 6) and time-series (Dataset 8) NN/CNN models. The values shown in the table are the mean +/- standard deviation from ten iterations.

Dataset type	Training accuracy	Testing accuracy	Precision	Recall	F1 score
Time-series	*0.52595 +/- 0.00000*	*0.52583 +/- 0.00000*	*0.52583 +/- 0.00000*	*1.00000 +/- 0.00000*	*0.68924 +/- 0.00000*
Magnitude spectrogram	*1.00000 +/- 0.00000*	*0.95503 +/- 0.00326*	*0.96010 +/- 0.00413*	*0.95415 +/- 0.00512*	*0.95710 +/- 0.00314*
Rectangular spectrogram	*0.76138 +/- 0.15679*	*0.73655 +/- 0.14313*	*0.74463 +/- 0.14841*	*0.88257 +/- 0.08874*	*0.79102 +/- 0.07624*

As shown in Table 8, the magnitude spectrogram models outperform other models. The time-series models fail again due to the nature of the unprocessed time-series PLC data. Interestingly, the recall score of the time-series dataset is very close to 1. This does not mean that the time-series model performed well. Figure 22 shows the confusion matrix of one of these time-series models, which explains why the recall score was high even though the accuracy was low. As shown in this figure, the time-series model predicted all samples as '1' (On) which explains the low accuracy. The recall is calculated as shown in Equation 9. The true positives are the accurately classified positives (top left quadrant in Figure 22) while the false negatives are misclassified negatives (bottom left quadrant). In our case, both values are zero, giving 0÷0, which causes error, but is overwritten as 1 during processing. In this way, the recall was very high.

$$Recall = \frac{Truepositives}{Truepositives + Falsenegatives} \tag{9}$$

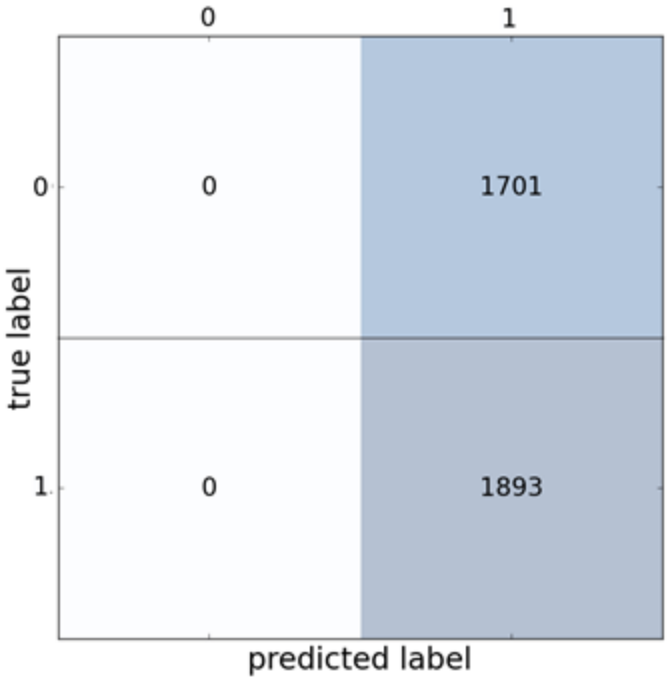


Figure 22. Confusion matrix of a featureless time-series model. This model was fitted with trimmed 'Dataset 8' for Case 2 analysis.

4.2.2.3. Case 3- Unknown and Dynamic Frequency

In Case 3, the NN/CNN models are trained using every frequency subset within the dataset (Datasets 4,6 and 8) except one and tested with the one which was left out of the training process. The objective here is to observe if the NN/CNN featureless models can generalize during the training process to be able to perform well on test data with untrained PLC frequency. The result of this experimentation is summarized in Figure 23.

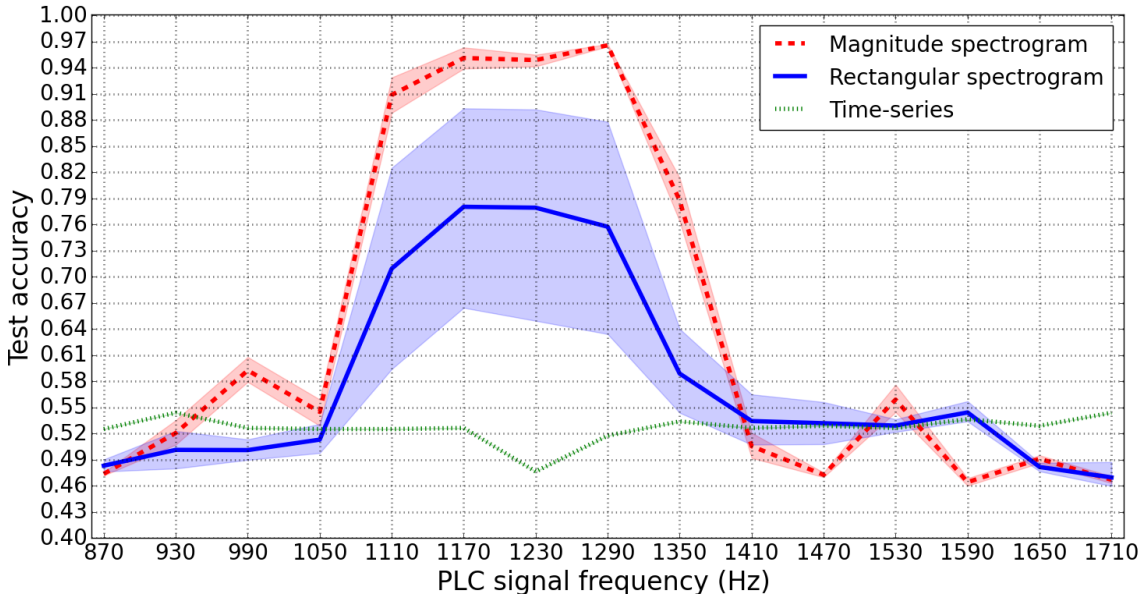


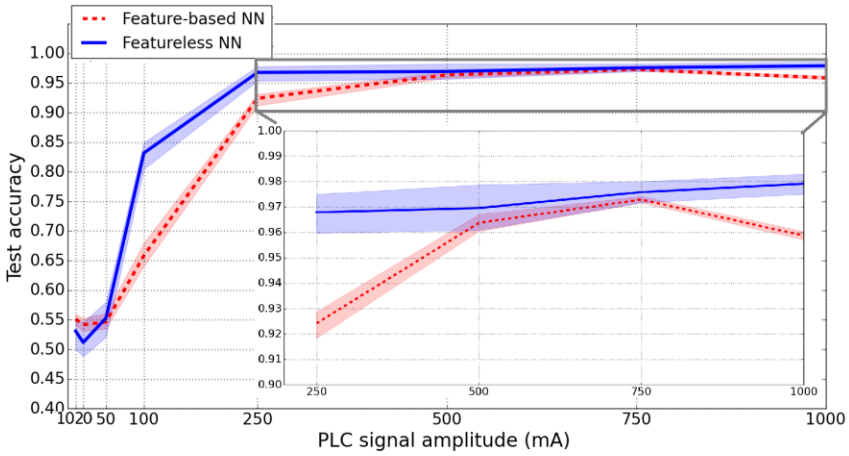
Figure 23. Test accuracy curves of NN/CNN models fitted with magnitude spectrogram, rectangular spectrogram, and time-series dataset for Case 3 analysis.

As seen in Figure 23, the time series data again failed, as expected and as described in Section 3.1.2. The magnitude and rectangular spectrogram models, however, had some success in generalizing to a new frequency data subset. The bell-shaped accuracy curves of both rectangular and magnitude spectrograms support the "similarity measure" hypothesis formulated previously in Case 3 of feature-based datasets in Section 4.1.2.3. This hypothesis was that the ML/NN models would be able to generalize the best (and perform well) to a test frequency when it trained on the greatest number of similar frequencies. The edge cases have the fewest "similar" training frequencies whereas the middle ones have the most, and hence, the accuracies for edge cases should be the lowest with increasing accuracy moving towards the middle. The curves for both spectrogram datasets shown in Figure 23 indeed show that the accuracies are low at the edge frequencies and greatest at mid frequencies, supporting our hypothesis. For practical implementation, this means that the training frequency range, for this type of unknown and dynamic PLC frequency scenario, should be wide enough so that the most probable

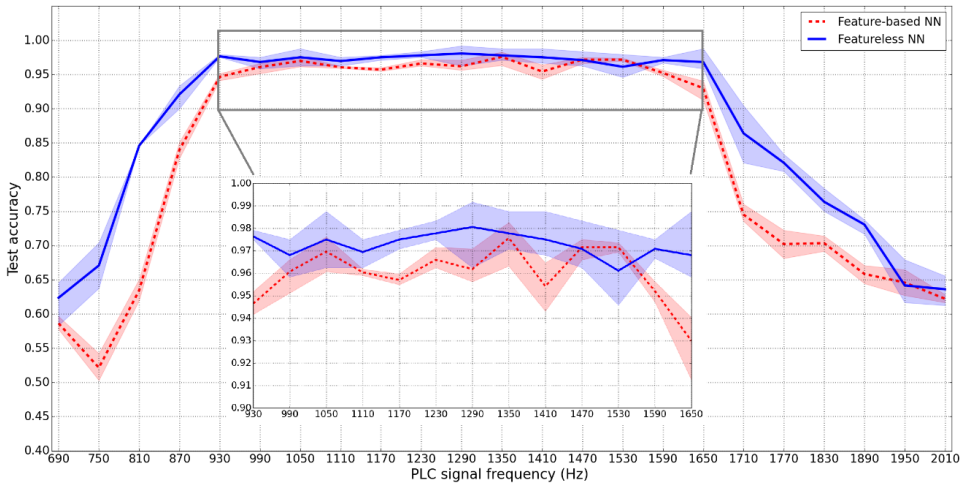
PLC test frequencies lie somewhere in the middle. This would ensure good classification accuracy (or low BER), and hence, efficient communication.

4.3. Feature-based Vs Featureless Learning

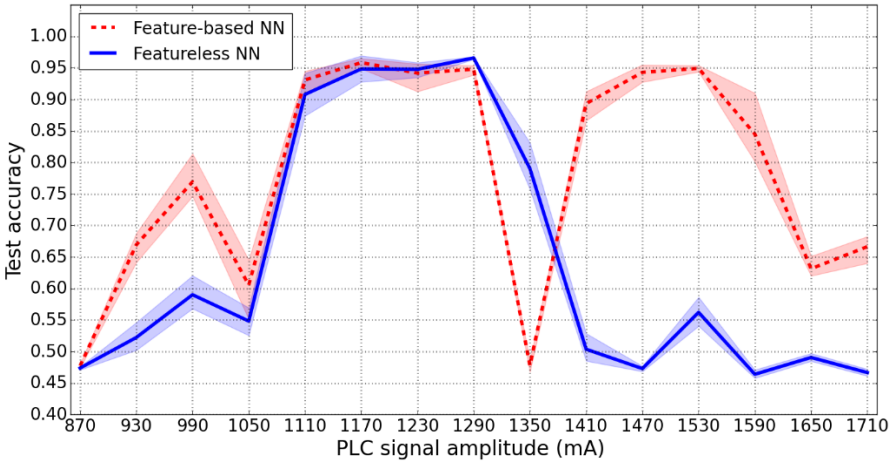
In Sections 4.1 and 4.2, the results of feature-based and featureless ML/NN/CNN were discussed separately. In the present section, these two methods are compared directly based on the previous results of NN models from each category, as shown in Figure 24. NN is used to compare the two methods because an identical NN (with same hyperparameters) was used with feature-based as well as featureless datasets. For featureless method, magnitude spectrogram is used as the dataset of choice (i.e., Datasets 3 and 4) because of its overall best performance as shown by results in Section 4.2.



(a) Amplitude analysis



(b) Frequency analysis: Case 1- known and static frequency



(c)) Frequency analysis: Case 3- unknown and dynamic frequency

Figure 24. Graphs showing test accuracy curves (with 95% confidence interval) of Neural Network models fitted with feature dataset and magnitude spectrogram-based featureless datasets. (a) Amplitude analysis using feature-based 'Dataset 1' and featureless 'Dataset 3,' (b) Case 1 frequency analysis using feature-based 'Dataset 2' and featureless 'Dataset 4' and (c) Case 3 frequency analysis using 'Dataset 2' and 'Dataset 4.'

Analyzing the graphs (a) and (b) in Figure 24 reveals that the featureless NN model had slightly higher accuracies than feature-based NN model when testing these models with a trained parameter. In both these cases, NN models were able to find the relevant features, using extracted features for the feature dataset and feature maps for featureless dataset. However, the information contained within the relevant feature maps is truer to the original raw signal than the lossy extracted features because of less data processing steps. Therefore, the featureless NN models had marginally better performance than feature-based NN.

On the other hand, when the models were asked to generalize to an untrained parameter, as in Case 3 shown in Figure 24 (c), the feature-based NN models perform better overall than featureless NN. This result may be due to the fact that the NN model needed to be more complex for the featureless dataset on account of its greater size (3x1024 per sample for magnitude spectrogram dataset whereas 1x61 for the feature dataset), higher dimensionality (2D for magnitude spectrogram while 1D for feature dataset) or the noise within the dataset (higher noise level in magnitude spectrogram dataset than the feature dataset). Another probable cause could be that the NN model overfitted the featureless dataset. Overfitting might not have been a problem with the feature-based dataset because of the limited number of features. However, for the featureless dataset, there are a lot more features (in the form of feature maps), and thus, the weights are spread out during training, causing overfitting of the training data. This overfitting would then result in poor generalization for untrained samples.

4.4. Summary of Results

The main results from Sections 4.1-4.3 are listed below:

1. Amplitude analysis of the feature-based (Section 4.1.1, Figure 13) and featureless datasets (Section 4.2.1, Figure 20) showed that the threshold amplitude for efficient PLC is about 250mA.
2. Frequency analysis of the feature-based (Section 4.1.2.1, Figure 15) and featureless datasets (Section 4.2.2.1., Figure 21) showed that frequency range from 930Hz - 1650Hz is optimal for PLC.
3. Section 4.1.3. showed that RMS energy and amplitude envelope were the most impactful features in the feature dataset. The analysis of these features could help in identifying the frequency of the PLC signal.
4. When the frequency of the test PLC signal is new to the ML/NN (i.e., ML/NN did not train on signals with the test frequency), the test accuracy depended on the location of the test frequency within the training set spectrum. When the test frequency was taken in the middle of the training spectrum, the accuracy was higher than when the test frequency was taken near the edge of the training set spectrum, as shown by Figure 18 and Figure 23.
5. Section 4.3 showed that feature-based dataset models had higher test accuracies when generalizing to untrained signal characteristics while featureless datasets had higher accuracies when the test samples and training samples had the same PLC frequency.

5. CONCLUSIONS AND FUTURE WORKS

5.1. Summary and Conclusions

This research study has been defined and executed based on a pre-existing collection of ultra-low-frequency PLC (ULF-PLC) data (see Section 3.1.4, Table 1 and Appendix B) which used amplitude-shifting and frequency-shifting to effect a communications channel from LV to HV region. Using this dataset, three broad questions related to the use of ML-driven algorithms in manipulation of such communication signals were explored. First, PLC signal characteristics were evaluated for use in nonstandard, ML-driven demodulation and decoding activities to recover transmitted information. Typically, signal processing algorithms are used for this purpose, but the potential for the use of ML techniques in "blind recognition" of ULF-PLC signals is compelling. Second, methods for pre-processing and formatting complex-valued communications signals were explored for use in conventional ML algorithms. ML algorithms and common toolkits typically require some effort in preparation of the dataset. Interestingly, conventional approaches are constructed to use only real-valued data, whereas communication signals depend directly on manipulation of signal characteristics in the complex domain. As a result, "best practices" do not exist for such mismatched data and algorithm configurations. Finally, the efficacy of a small collection of supervised ML techniques in decoding and processing the ULF-PLC communication data has been evaluated, with performance characteristics typical of ML-driven approaches and certain outcomes typical in the evaluation of communication signals. Since the domain of interest (i.e., ULF-PLC) is unique and poses challenges which are foreign to conventional communication systems as well as ML approaches, all outcomes

and learnings are couched in the context of this particular application. A brief overview and resulting conclusions of- PLC signal characteristics are described in Section 5.1.1, dataset formatting and related issues are described in Section 5.1.2, and finally, outcomes of supervised ML algorithms using ULF-PLC data are presented in Section 5.1.3.

5.1.1. PLC Signal Characteristics

The dataset available for this study (Appendix B) contained transmitted ULF-PLC signals which leveraged conventional baseband amplitude and frequency shifting. Phase shifting was not present in the signals contained in the dataset. As a result, effective discrimination of amplitude and frequency shifting in the ULF-PLC signals are key parameters that need to be optimized for accurate decoding of the information contained in each signal. As presented the energy or signal-to-noise analysis of Section 4.1.1 (Figure 13) and Section 4.2.1 (Figure 20), 250mA appears to be a critical threshold amplitude for effective transmission of the binary, amplitude shifted ULF-PLC signals contained in the experimental dataset. As presented in the frequency analysis of Section 4.1.2.1 (Figure 15) and Section 4.2.2.1 (Figure 21), the optimal frequency range for low-frequency, inter-leveled PLC appears to be confined in a distinct passband around 930Hz-1650Hz. Below this band, the strong odd harmonics of the fundamental power signal interfere directly with the PLC signal resulting in very low SNR, while above this band, other grid components such as large, series transformers and other elements attenuate the signals substantially, causing poor signal propagation from LV to HV regions of the distribution grid.

The threshold amplitude and usable frequency range of ULF-PLC signals are dependent on various factors such as the distribution grid architecture (between the PLC

signal transmitter and receiver), the power load and temporal noise/artifacts in the distribution grid, etc. Therefore, the values of amplitude and frequency range as presented in this research may be relatively common but are not universally applicable. However, the bandpass nature of the power line channel is indeed characteristic of ULF-PLC.

5.1.2. Dataset Characteristics

One of the primary objectives of this study was to explore how the raw PLC data could be used in ML, what classes of algorithms might be more effective, and what types of data pre-processing might be most desirable. To that end, we investigated two different methods- feature-based and featureless.

For feature-based methods, three sets of features were extracted from the raw ULF-PLC data. The amplitude envelope and RMS energy were extracted from raw time-domain data, and the spectral centroid was extracted from the transformed frequency-domain data. As presented in Section 4.1.3, SBS analysis indicated that amplitude envelope and RMS energy were the most impactful features. This is logical since these features relate directly to the amplitude information of the ULF-PLC signals contained in the dataset (Appendix B). In cases where other modulation techniques may be in use (e.g., phase modulation), additional features need to be explored to create effective feature-based dataset for ML.

To explore featureless methods, 1D raw time-series dataset, 2D magnitude spectrogram dataset and 3D rectangular spectrogram dataset were created from the ULF-PLC dataset. These three pre-processed dataset formats were then used to train a NN/CNN with the aim of classifying the On/Off state information of the transmitted

ULF-PLC signals. In this evaluation, the time-series format was the least effective because of the dominance of power signal in this domain, as shown in Figure 7. The strong "out of band" interference posed by the fundamental power signal essentially dominated and confused the training process. By preprocessing the time-domain data using a spectral transform, however, additional information can be presented to the training process which improves ML performance. Interestingly, the increase in dimensionality of the preprocessed data also changes the outcomes of the training process. In particular, as shown in Table 8, Figure 20, Figure 21, and Figure 23, the dataset preprocessed into a magnitude spectrogram alone (ignoring phase information) performed better than the dataset preprocessed into rectangular (real, complex) format. This outcome is logical given the constraints of the ULF-PLC dataset, which does not contain information which has been phase modulated. Rather, the magnitude information present in the magnitude spectrogram relate directly to the amplitudes of the PLC signal and contain all of the relevant transmitted data, thereby retaining the On/Off state information more efficiently than the rectangular spectrogram. If more sophisticated frequency and/or phase shift methods are used for modulating data into the ULF-PLC channel, then the use of rectangular spectrogram formats is likely to outperform magnitude spectrogram formats, as concluded in our previous research [38]. Therefore, there are merits to all of these featureless datasets, in PLC application and beyond.

The outcomes from feature-based and featureless datasets were compared directly using an identical NN. The results of this comparison are discussed in Section 4.3 and shown in Figure 24. Interestingly, the featureless magnitude spectrogram models had higher test accuracy than the feature-based models when the test data had the same PLC

signal parameter (amplitude or frequency) as the training data. However, when the test data had completely different signal parameters (as described in Section 4.1.2.3 and Section 4.2.2.3), the feature-based models were more consistent in providing high classification accuracies. As a result, we conclude that NN models overfit with the larger featureless dataset, causing it to generalize poorly on the test data with untrained parameters. Therefore, feature-based methods may be more appropriate when generalization is desired.

In addition to testing accuracy, comparison of feature-based and featureless methods using metrics such as the complexity of the ML models, time of processing, scalability, etc. may be valuable outcomes. In our tests using the ULF-PLC dataset, feature-based models had similar performance to featureless models despite employing simpler ML. The use of simpler ML models along with smaller dataset size of the feature-based method resulted in a requirement for substantially less computer processing power and training time, as noted anecdotally. However, the feature-dataset also requires data preprocessing, which requires time and knowledge of the domain, and which was not included in our anecdotal comparison of the methods. In contrast, although featureless methods require more complex ML/NN models, they are also easier to scale to varying architectures and applications and require less domain-specific knowledge to optimize. Therefore, feature-based methods are desirable when training time and computer processing power are a consideration, while featureless methods are attractive when scaling and ease of deployment are the main concerns.

5.1.3. ML/NN Algorithm Characteristics

To evaluate the ULF-PLC dataset, we utilized three supervised ML algorithms: Logistic Regression (LR), Support Vector Machine (SVM) and Decision Tree (TREE). We also used Neural Network (NN) and Convolutional Neural Network (CNN) models. The comparison between LR, SVM, TREE and NN for training using a feature-dataset is presented in in Section 4.1. These outcomes indicate that the classification accuracies of these models were very similar, with SVM slightly better in most cases due to its non-linear capability. However, this also meant that SVM models needed much more stringent regularization to avoid overfitting, which increases implementation complexity. In contrast, NN and CNN may be more suitable for featureless datasets because of their relatively more complex architectures and native ability to process multi-dimensional datasets. For the same reason, NN and CNN approaches are also more flexible and scalable.

5.2. Future Works

For this study, two sets of ULF-PLC data were available as described in Section 3.1.4 and Table 1. To validate and strengthen the conclusions of this thesis, additional ULF-PLC data are necessary for training and testing, using multiple variations of modulation and transmitted power. New data should have the same amplitude and frequency characteristic as the existing data as well as wider ranges of these parameters, and including other conventional modulation techniques (e.g., phase modulation). Further, the labels for the data used in this study were extracted using conventional DSP techniques. In the future, these data labels can be recorded while modulating the information and transmitting the signals, which would make the labels more accurate.

The complexity of the transmitted PLC signal can be increased too. This can be done by using various conventional modulation schemes like Amplitude Shift Keying (ASK), Frequency Shift Keying (FSK), Phase Shift Keying (PSK) and their more sophisticated variants (e.g., M-ary PSK, M-ary ASK, Quadrature Amplitude Modulation (QAM), and so on). Additionally, multicarrier transmission techniques, i.e., passing PLC signals in multiple frequency bands via transform-based methods such as Orthogonal Frequency-Division Multiplexing (OFDM) and variants, can increase the bit rate of the ULF-PLC transmission, which could have major effect in the scope of its application. The ML/NN-based bit classification techniques which are presented in this thesis can also be applied to these more complex ULF-PLC modulation techniques but will require additional evaluation and classification of efficiency and data pre-processing approaches.

At the receiver, the demodulation/decoding and processing steps can be further streamlined and optimized. Rectangular spectrograms need very little pre-processing and can be used with conventional modulation schemes such as ASK, FSK and PSK. Therefore, the use of rectangular spectrogram data may serve as a good candidate for streamlined and universal application of ULF-PLC. However, such an approach would also need more complex ML techniques.

At the transmitter, the approach to modulating ULF-PLC signals depends on test equipment including a lab-grade current source (Ametek CS3000) [26]. Miniaturization of this bulky device is a major challenge that needs to be resolved to be able to employ the proposed inter-level ULF-PLC in real-world applications. Further, an automated approach to probing the power line channel spectrum is critical to find the optimal frequency bands for subsequent transmission (e.g., via dynamic spectral allocation [55],

or DySpan [56] techniques). This would ensure that the transmitted ULF-PLC signals are exposed to the least amount of interference from the power signal harmonics, the latent grid noise, and the coherent interference due to harmonically related images of the ULF-PLC signals themselves.

APPENDIX SECTION

Appendix A

GitHub page

The data files and python scripts used for in this thesis can be found at:

https://github.com/kushal-thapa/ML_for_PLC_thesis

Appendix B

Features of the raw ULF-PLC data

Number of raw PLC files: 30

File type: .wav

Length of each file: 105 seconds (trimmed to 100 seconds during processing)

PLC signal amplitude: 10mA, 20mA, 50mA, 100mA, 250mA, 500mA and 1A

PLC frequency: 690Hz-2010Hz with 60Hz spacings

Number of channels: 3

Sampling rate: 8000 samples per second

Appendix C

Spectrogram parameters

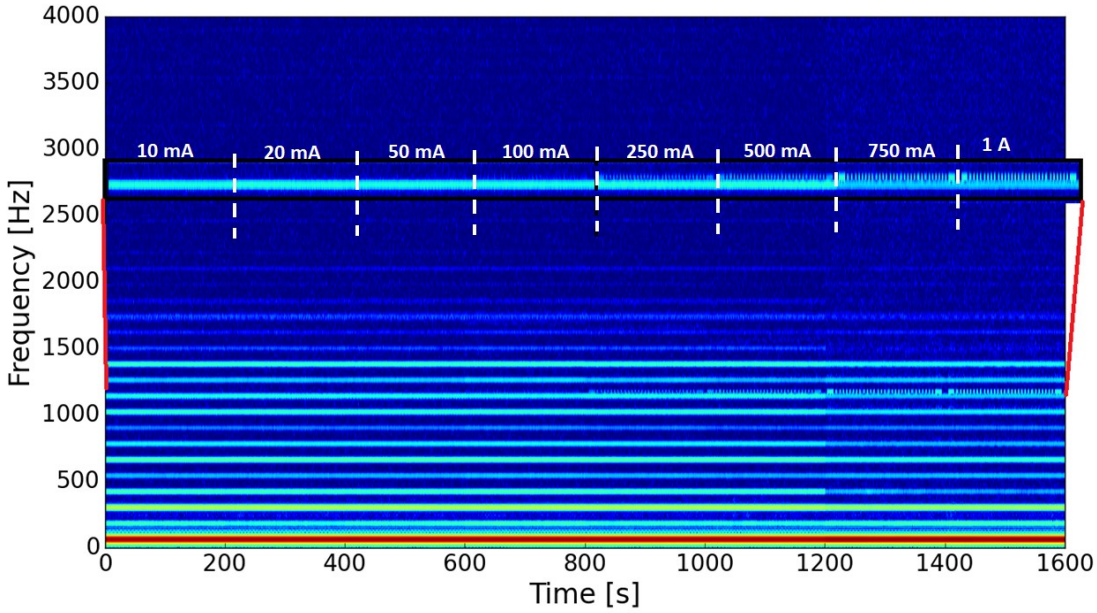
Frame size = 500 samples

Frame overlap = 250 samples

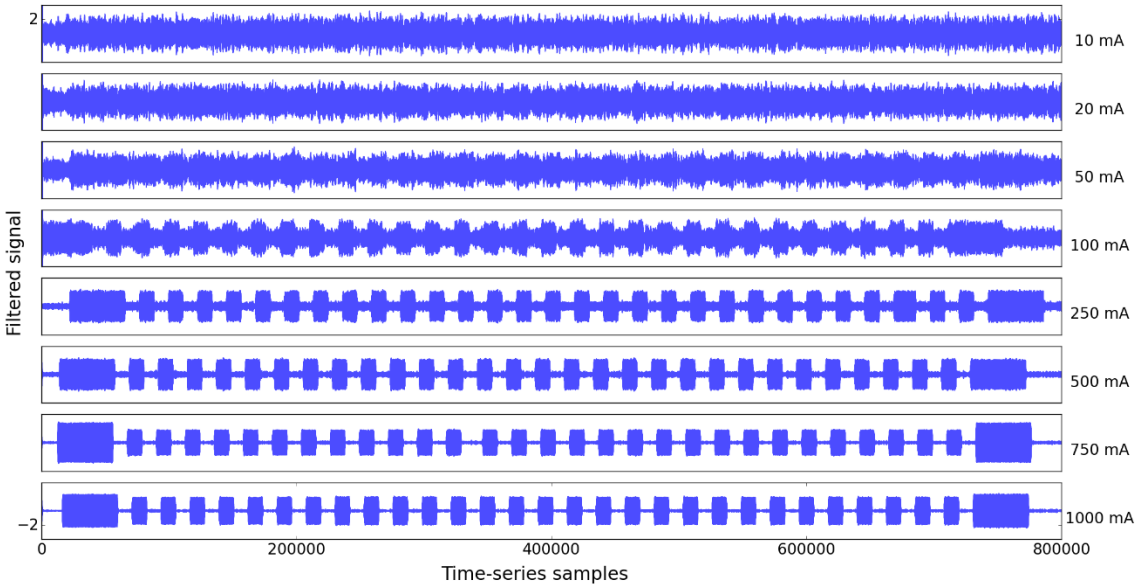
FFT size =1024

Window = Hanning

Appendix D

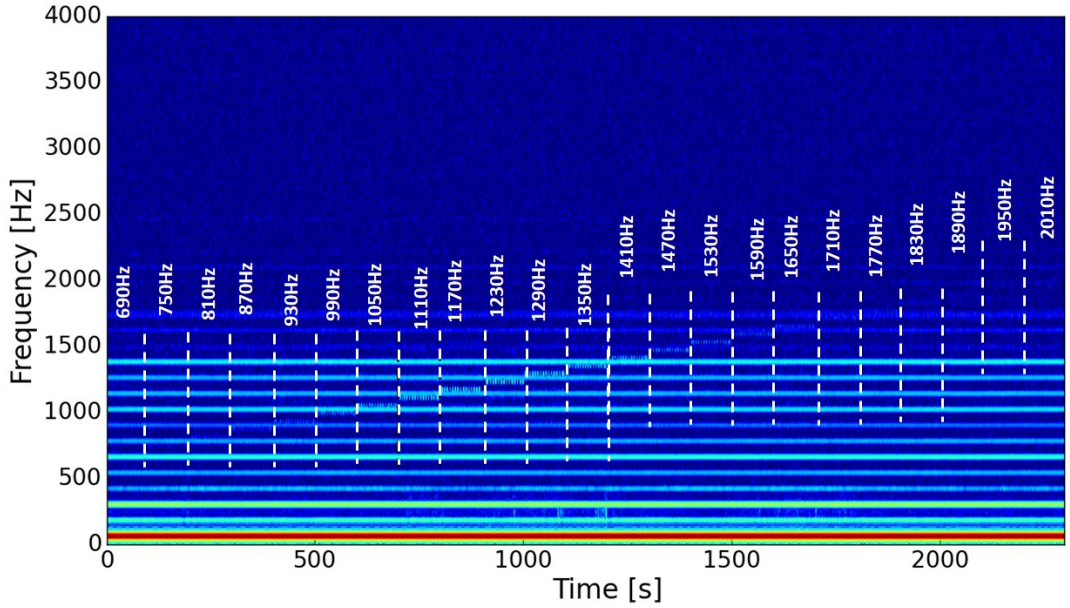


Appendix Figure 1. Magnitude spectrogram of the combined files of 'Raw Set 1.' The red line points to 1170Hz, the ULF-PLC band for this set of files. The zoomed version of this band is shown in the black box. The vertical dotted white lines separate the eight amplitude sections from within this combined spectrogram.

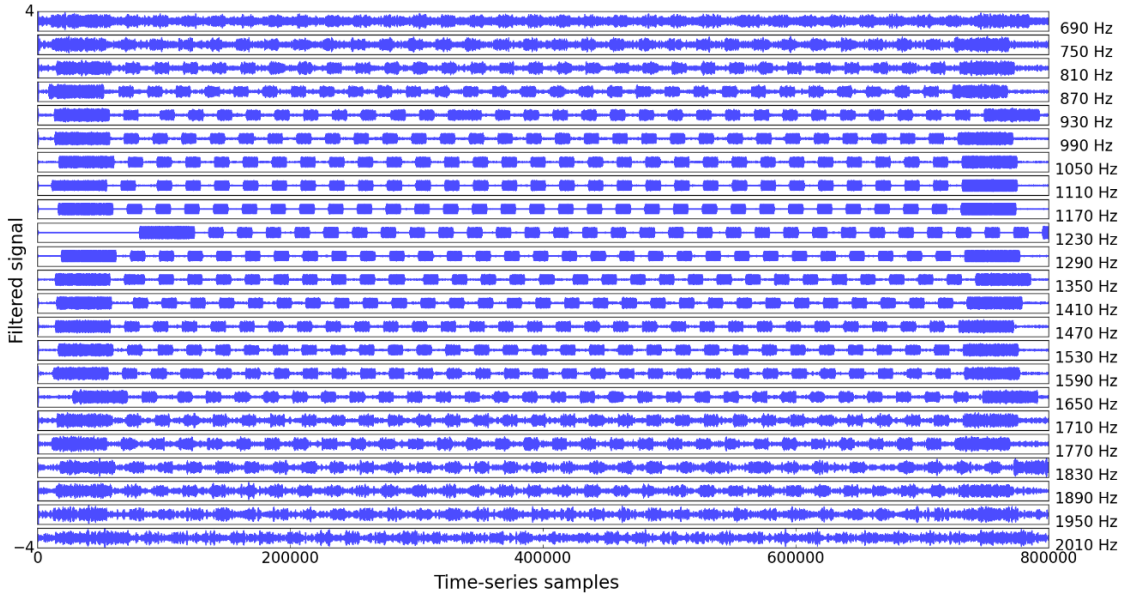


Appendix Figure 2. Subplots of the filtered time-series signals of the eight files of 'Raw Set 1.'

Appendix E



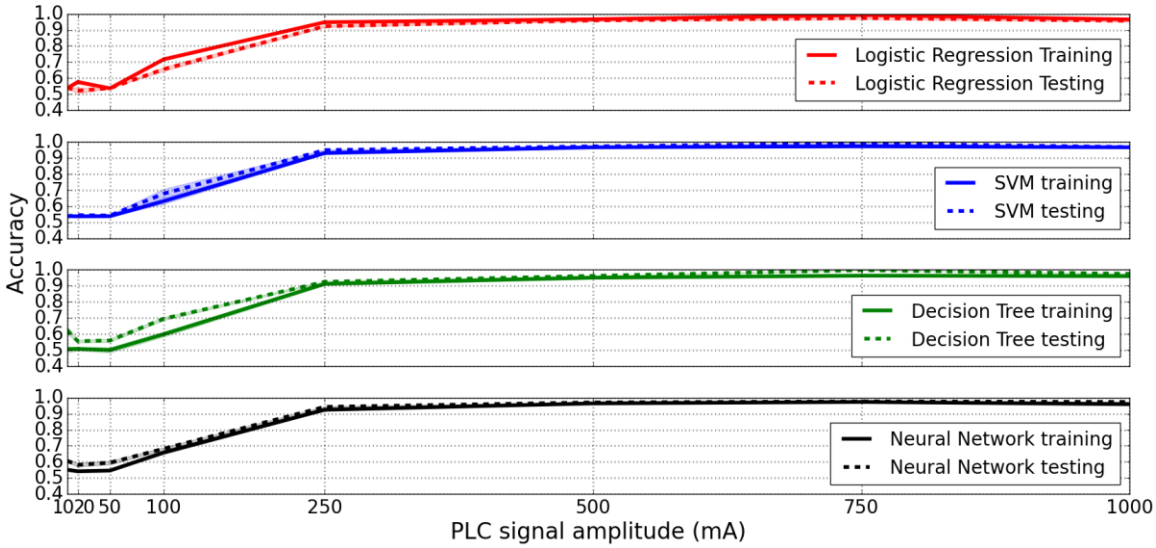
Appendix Figure 3. Magnitude spectrogram of the combined files of 'Raw Set 2.' The vertical dotted white lines separate the twenty-three different frequency sections from within this combined spectrogram.



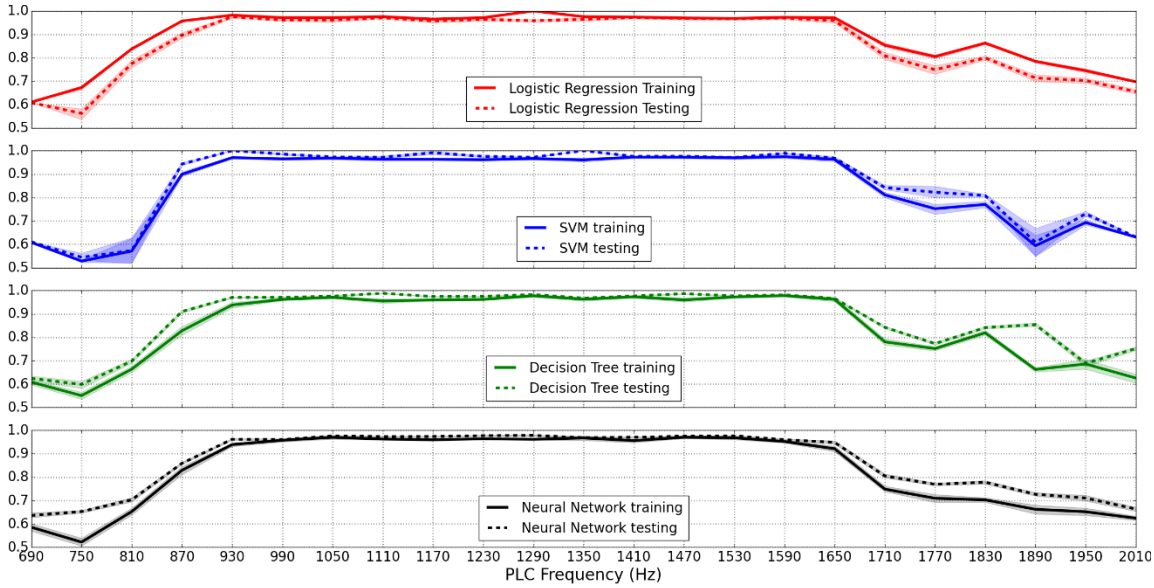
Appendix Figure 4. Subplots of the filtered time-series signals of the twenty-three files of 'Raw Set 2.'

Appendix F

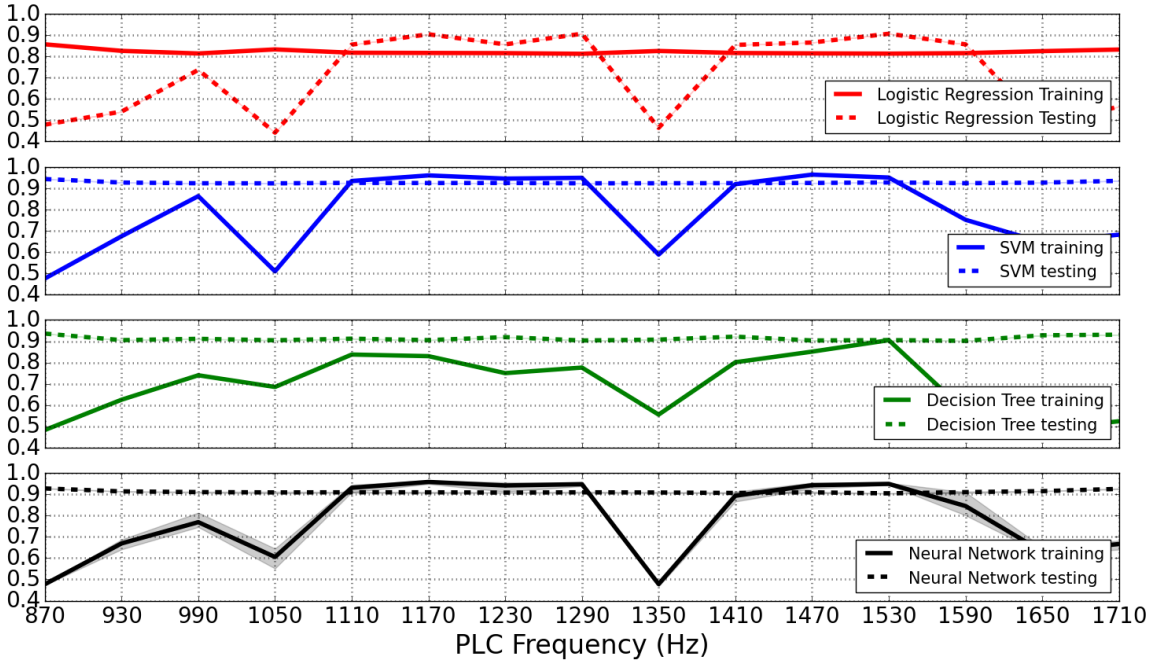
Train and test accuracies:



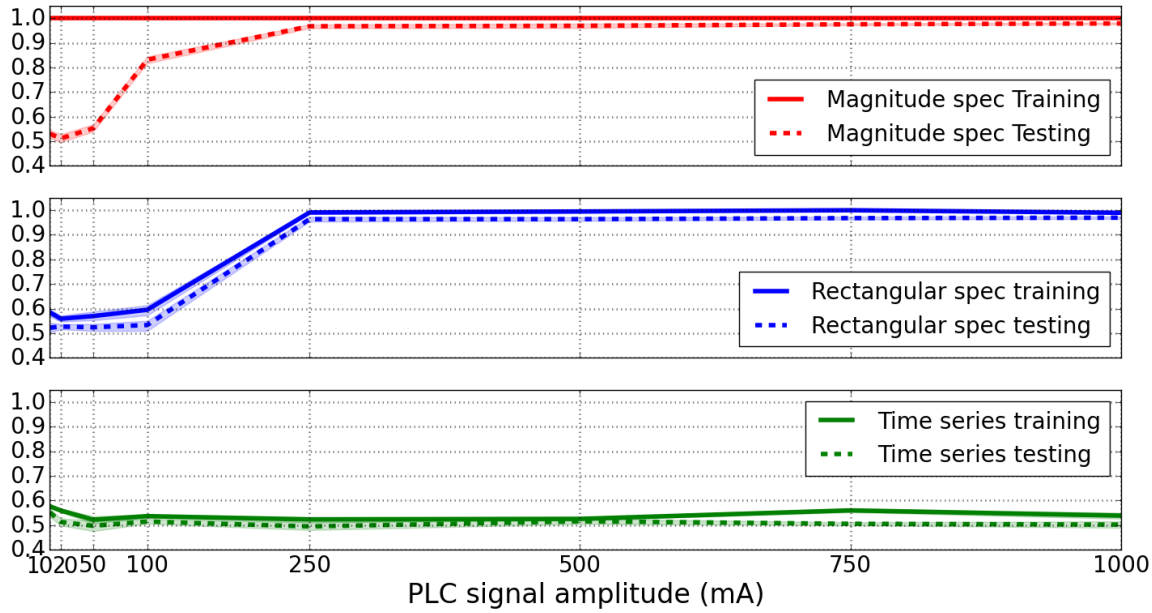
Appendix Figure 5. Subplots of train and test accuracies of LR, SVM, TREE and NN for Dataset 1.



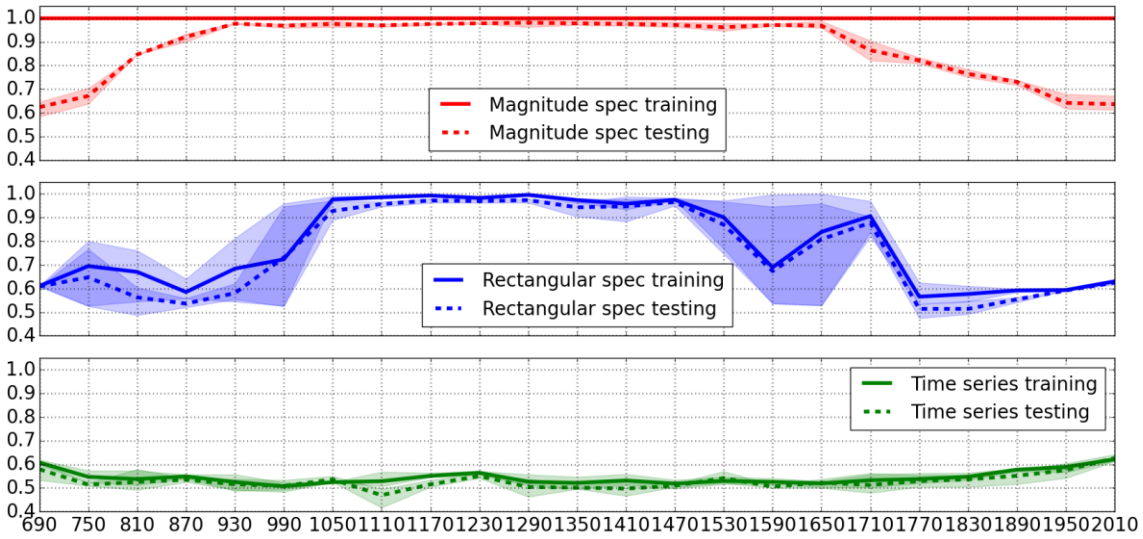
Appendix Figure 6. Subplots of train and test accuracies of LR, SVM, TREE and NN for Dataset 2: Case 1.



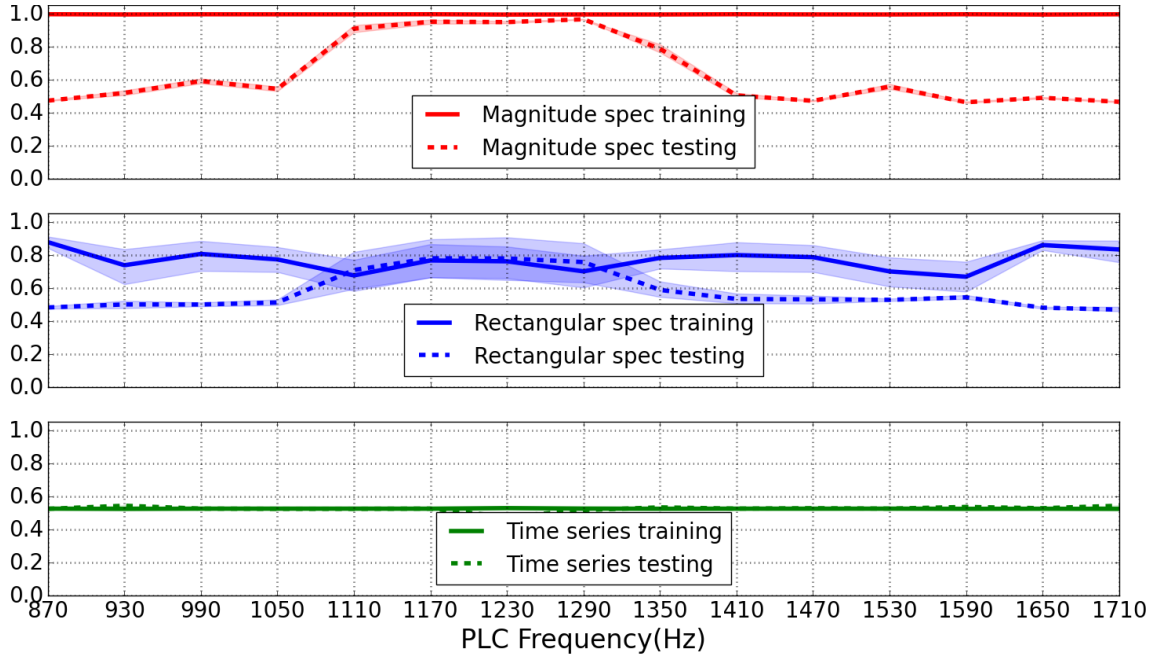
Appendix Figure 7. Subplots of train and test accuracies of LR, SVM, TREE and NN for Dataset 2: Case 3.



Appendix Figure 8. Subplots of train and test accuracies of NN models for magnitude spectrogram (Dataset 3), rectangular spectrogram (Dataset 5) and time-series (Dataset 7) dataset.



Appendix Figure 9. Subplots of train and test accuracies of NN models for magnitude spectrogram (Dataset 4), rectangular spectrogram (Dataset 6) and time-series (Dataset 6) dataset: Case 1.



Appendix Figure 10. Subplots of train and test accuracies of NN models for magnitude spectrogram (Dataset 4), rectangular spectrogram (Dataset 6) and time-series (Dataset 8) dataset: Case 3.

REFERENCES

[1] K. Dostert, "Telecommunications over the power distribution grid–possibilities and limitations," *IIR-Powerline*, vol. 6, no. 97, 1997.

[2] K. Rabie, A. M. Tonello, N. Al-Dhahir, J. Song, and A. Sendin, "IEEE Access Special Section Editorial: Advances in Power Line Communication and its Applications," *IEEE Access*, vol. 7, pp. 133371–133374, 2019, doi: 10.1109/ACCESS.2019.2942516.

[3] S. U. Ercan, O. Ozgonenel, Y. E. Haj, C. Christopoulos, and D. W. P. Thomas, "Power line communication design and implementation over distribution transformers," in *2017 10th International Conference on Electrical and Electronics Engineering (ELECO)*, Nov. 2017, pp. 190–194.

[4] A. K. (Orem Schurig UT), "Universal lan power line carrier repeater system and method," 1998 [Online]. Available: http://libproxy.txstate.edu/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=edspgr&AN=edspgr.05818821&site=eds-live&scope=site

[5] R. (Guangzhou) Du and Q. (Antioch Xiao CA), "Baby Monitor," 2009 [Online]. Available: http://libproxy.txstate.edu/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=edspap&AN=edspap.20090212926&site=eds-live&scope=site

[6] P. An, X. Chen, J. Wang, S. Zhou, and X. Shan, "Analysis on Application of Repeater Technology in Powerline Communications Networks," in *2007 IEEE International Symposium on Power Line Communications and Its Applications*, Mar. 2007, pp. 273–277. doi: 10.1109/ISPLC.2007.371136.

[7] D. Dzung, I. Berganza, and A. Sendin, "Evolution of powerline communications for smart distribution: From ripple control to OFDM," in *2011 IEEE International Symposium on Power Line Communications and Its Applications*, Apr. 2011, pp. 474–478. doi: 10.1109/ISPLC.2011.5764444.

[8] S. McClellan, D. Valles, and G. Koutitas, "Dynamic Voltage Optimization Based on In-Band Sensors and Machine Learning," *Appl. Sci.-BASEL*, vol. 9, no. 14, Jul. 2019, doi: 10.3390/app9142902.

[9] A. M. Tonello, N. A. Letizia, D. Righini, and F. Marcuzzi, "Machine Learning Tips and Tricks for Power Line Communications," *IEEE Access*, vol. 7, pp. 82434–82452, 2019, doi: 10.1109/ACCESS.2019.2923321.

[10] M. Korki, N. Hosseinzadeh, H. L. Vu, T. Moazzeni, and C. H. Foh, "A channel model for power line communication in the smart grid," in *2011 IEEE/PES Power Systems Conference and Exposition*, Mar. 2011, pp. 1–7. doi: 10.1109/PSCE.2011.5772561.

[11] B. Varadarajan, Il Han Kim, A. Dabak, D. Rieken, and G. Gregg, "Empirical measurements of the low-frequency power-line communications channel in rural North America," in *2011 IEEE International Symposium on Power Line Communications and Its Applications*, Apr. 2011, pp. 463–467. doi: 10.1109/ISPLC.2011.5764442.

[12] G. Prasad, Y. Huo, L. Lampe, A. Mengi, and V. C. M. Leung, "Fault Diagnostics with Legacy Power Line Modems," in *2019 IEEE International Symposium on Power Line Communications and its Applications (ISPLC)*, Apr. 2019, pp. 1–6. doi: 10.1109/ISPLC.2019.8693385.

[13] M. Nassar, J. Lin, Y. Mortazavi, A. Dabak, I. H. Kim, and B. L. Evans, "Local Utility Power Line Communications in the 3–500 kHz Band: Channel Impairments, Noise, and Standards," *IEEE Signal Process. Mag.*, vol. 29, no. 5, pp. 116–127, Sep. 2012, doi: 10.1109/MSP.2012.2187038.

[14] F. Aalamifar, H. S. Hassanein, and G. Takahara, "Viability of powerline communication for the smart grid," in *2012 26th Biennial Symposium on Communications (QBSC)*, May 2012, pp. 19–23. doi: 10.1109/QBSC.2012.6221343.

[15] J. Williams, E. Drawhorn, and J. Morrow, "Narrowband, Ultra-Low Frequency Power Line Communication (PLC) for Channel Characterization Purposes."

[16] S. Raschka, *Python machine learning : machine learning and deep learning with Python, scikit-learn, and TensorFlow.*, Second edition, Fully revised and Updated. Packt Publishing, 2017. [Online]. Available: http://libproxy.txstate.edu/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=cat00022a&AN=txi.b5161466&site=eds-live&scope=site

[17] R. Choudhary and H. K. Gianey, "Comprehensive review on supervised machine learning algorithms," 2017, pp. 37–43.

[18] P. Dayan, M. Sahani, and G. Deback, "Unsupervised learning," *MIT Encycl. Cogn. Sci.*, pp. 857–859, 1999.

[19] E. Alpaydin, *Introduction to machine learning.*, Second edition. MIT Press, 2010. [Online]. Available: http://libproxy.txstate.edu/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=cat00022a&AN=txi.b1923201&site=eds-live&scope=site

[20] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[21] U. Michelucci, *Applied deep learning : a case-based approach to understanding deep neural networks.* Apress, 2018. [Online]. Available: https://libproxy.txstate.edu/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=cat00022a&AN=txi.b4758883&site=eds-live&scope=site

[22] L. Wang, *Support vector machines : theory and applications.* Springer, 2005. [Online]. Available: https://libproxy.txstate.edu/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=cat00022a&AN=txi.b2634857&site=eds-live&scope=site

[23] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE Trans. Syst. Man Cybern. Syst. Man Cybern. IEEE Trans. IEEE Trans Syst Man Cybern*, vol. 21, no. 3, pp. 660–674, May 1991, doi: 10.1109/21.97458.

[24] B. Yegnanarayana, *Artificial neural networks*. PHI Learning Pvt. Ltd., 2009.

[25] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*, vol. 1, no. 2. MIT press Cambridge, 2016.

[26] "California Instruments CS Series - AC Current Source," *Ametek Programmable Power*. www.powerandtest.com/power/ac-power-sources/cs-series (accessed Apr. 16, 2021).

[27] "DI-1100 USB Data Acquisition Starter Kit," *DATAQ Instruments*. https://www.dataq.com/products/di-1100/ (accessed Apr. 08, 2021).

[28] S. Bochner, K. Chandrasekharan, and K. Chandrasekharan, *Fourier transforms*, no. 19. Princeton University Press, 1949.

[29] E. O. Brigham and R. Morrow, "The fast Fourier transform," *IEEE Spectr.*, vol. 4, no. 12, pp. 63–70, 1967.

[30] K. R. Rao, D. N. Kim, and J. J. Hwang, *Fast Fourier transform : algorithms and applications.* Springer, 2010. [Online]. Available: https://libproxy.txstate.edu/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=cat00022a&AN=txi.b2587772&site=eds-live&scope=site

[31] D. A. Lyon, "The discrete fourier transform, part 4: spectral leakage," *J. Object Technol.*, vol. 8, no. 7, 2009.

[32] A. Breitenbach, "Against spectral leakage," *Measurement*, vol. 25, no. 2, pp. 135–142, 1999.

[33] V. Mathews and D. Youn, “Spectral leakage suppression properties of linear and quadratic windowing,” *IEEE Trans. Acoust. Speech Signal Process.*, vol. 32, no. 5, pp. 1092–1095, 1984.

[34] T. M. Apostol, *Mathematical analysis.*, Second edition. Addison-Wesley Pub. Co., 1974. [Online]. Available: http://libproxy.txstate.edu/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=cat00022a&AN=txi.b1197127&site=eds-live&scope=site

[35] J. S. Dramsch, M. Lüthje, and A. N. Christensen, “Complex-valued neural networks for machine learning on non-stationary physical data,” *Comput. Geosci.*, vol. 146, p. 104643, Jan. 2021, doi: 10.1016/j.cageo.2020.104643.

[36] H. G. Zimmermann, A. Minin, and V. Kusherbaeva, “Comparison of the complex valued and real valued neural networks trained with gradient descent and random search algorithms,” 2010, pp. 213–218.

[37] J. Shima, “Weak signal processing systems and methods,” 10879946, Dec. 29, 2020 Accessed: Mar. 08, 2021. [Online]. Available: http://patft1.uspto.gov/netacgi/nph-Parser?Sect1=PTO1&Sect2=HITOFF&d=PALL&p=1&u=%2Fnetahtml%2FPTO%2Fsrchnum.htm&r=1&f=G&l=50&s1=10879,946.PN.&OS=PN/10879,946&RS=PN/10879,946

[38] K. Thapa, S. McClellan, and D. Valles, “An Evaluation of Neural Network Performance Using Complex-Valued Input Data,” in *ICDT 2021 : The Sixteenth International Conference on Digital Telecommunications*, Porto, Apr. 2021, pp. 27–31. [Online]. Available: https://www.thinkmind.org/index.php?view=article&articleid=icdt_2021_1_60_18006

[39] T. Smyth, “Amplitude Envelopes,” *Amplitude Envelopes*. http://musicweb.ucsd.edu/~trsmyth/sinusoids171/Amplitude_Envelopes.html (accessed Dec. 09, 2020).

[40] B. Boashash, Ed., “Chapter 4 - Advanced Time-Frequency Signal and System Analysis,” in *Time-Frequency Signal Analysis and Processing (Second Edition)*, Oxford: Academic Press, 2016, pp. 141–236. doi: 10.1016/B978-0-12-398499-9.00004-2.

[41] S. Tjoa, “Energy and RMSE,” *musicinforetrieval.com*. https://musicinformationretrieval.com/energy.html (accessed May 19, 2021).

[42] J. M. Grey and J. W. Gordon, “Perceptual effects of spectral modifications on musical timbres.,” *J. Acoust. Soc. Am.*, vol. 63, pp. 1493–1500, May 1978.

[43] F. Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python,” *J. Mach. Learn. Res.*, vol. 12, no. 85, pp. 2825–2830, 2011.

[44] “sklearn.linear_model.LogisticRegression,” *scikit learn*. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html (accessed Jun. 01, 2021).

[45] “sklearn.svm.SVC.” https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html (accessed Jun. 01, 2021).

[46] “sklearn.tree.DecisionTreeClassifier.” https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html (accessed Jun. 01, 2021).

[47] “sklearn.model_selection.GridSearchCV.” https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html (accessed Jun. 01, 2021).

[48] “sklearn.metrics.precision_score.” https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_score.html (accessed Jun. 01, 2021).

[49] “sklearn.metrics.recall_score.” https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html (accessed Jun. 01, 2021).

[50] “sklearn.metrics.f1_score.” https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html (accessed Jun. 01, 2021).

[51] “Plotting Learning Curves — scikit-learn 0.24.2 documentation.” https://scikit-learn.org/stable/auto_examples/model_selection/plot_learning_curve.html (accessed Jun. 01, 2021).

[52] “sklearn.metrics.confusion_matrix.” https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html (accessed Jun. 01, 2021).

[53] M. Abadi *et al.*, “Tensorflow: Large-scale machine learning on heterogeneous distributed systems,” *ArXiv Prepr. ArXiv160304467*, 2016.

[54] A. Gulli and S. Pal, *Deep learning with Keras*. Packt Publishing Ltd, 2017.

[55] P. Leaves *et al.*, “Dynamic spectrum allocation in composite reconfigurable wireless networks,” *IEEE Commun. Mag.*, vol. 42, no. 5, pp. 72–81, 2004.

[56] F. Wunsch *et al.*, “DySPAN spectrum challenge: Situational awareness and opportunistic spectrum access benchmarked,” *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 3, pp. 550–562, 2017.