Powerline Fault Detection for Partial Discharge using Deep Learning

EE 272 Exploratory Project

Under the supervision of - Dr.Soumya R Mohanty, Associate Professor, Department of Electrical Engineering, IIT(BHU)

Students Team-

L.N. Saaswath- 19084011
Shivam Awasthi- 19085084
Ajitesh Pandey- 19084004
Nandini Jain- 19084025
Tanmay Sharma- 19085090

Abstract—Today, insulated overhead conductors are increasingly used in many places of the world due to the higher operational reliability, elimination of phase-to-phase contact, closer distances between phases and stronger protection for animals. However, the standard protection devices are often not able to detect the conductor's phase-to-ground fault and the more frequent tree/tree branch hitting conductor events as these events only lead to partial discharge (PD) activities instead of causing overcurrent seen on bare conductors.

Introduction

Medium voltage overhead power lines run for hundreds of miles to supply power to cities. These great distances make it expensive to manually inspect the lines for damage that doesn't immediately lead to a power outage, such as a tree branch hitting the line or a flaw in the insulator. These modes of damage lead to a phenomenon known as *partial discharge* — *an electrical discharge which does not bridge the electrodes between an insulation system completely.* Partial discharges slowly damage the power line, so left unrepaired, they will eventually lead to a power outage or start a fire.

This project aims to detect the faults so as to stop partial discharge in a power network sooner before causing severe damage to the power line.

What is Partial Discharge?

Partial Discharge(PD) is a localized electrical discharge that only partially bridges the insulation between conductors and which may or may not occur adjacent to a conductor.

PD occurs whenever there is a stressed region due to some impurity/cavity inside the insulation or when there is a protrusion outside it. The stressed region is formed if there are sharp edges or protrusions around the conductor.

Types of Partial Discharges in Power Cables

Internal Discharges: occurring in voids or cavities within solid or liquid dielectrics.

Surface Discharges: appearing at the boundary of different insulation materials.

Corona Discharges: occurring in gaseous dielectrics in the presence of inhomogeneous fields.

Treeing: continuous impact of discharges in solid dielectrics forming discharge channels.

Explanation of PD phenomenon

In a typical situation of PD, imagine there is an internal cavity/void or impurity in insulation. When High Voltage is applied on HV conductor, a field is also induced the cavity. Further, when the field increases the breakdown capacity of this defect, it breaks down and discharges different forms of energy which result in partial discharge.The detection and measurement of discharges is based on the exchange of energy taking place during the discharge. These exchanges are manifested as-

- Electrical pulse currents
- Dielectric losses
- M.radiation(light)
- sound(noise)
- Increased gas pressure
- Chemical Reactions

Classical Modes of Detection

- Partial Discharges can be detected by measuring the emissions they give off: Ultrasonic Sound, Transient Earth Voltages (TEV and UHF energy).

- It is possible to enhance the modes of detection by deep learning after doing some preprocessing.

Motivation:

We thought of doing this project so that we can Apply our knowledge of Deep Learning in Electrical domain. There are also various advantages of detecting partial discharge-

- By Detecting partial discharges repairs, we can reduce maintenance costs and prevent power outages.
- Repairs can be made before any lasting harm occurs.

We can do Feature Extraction on Signals and then feed them into the Neural Networks to do binary classification for whether the partial discharge pattern is present or not.

Data Acquisition

The data which we have used to train the models have been taken from the Smart meter project undertaken by the ENET Centre at VŠB.

It consists of signals which contain 800,000 measurements of a power line's voltage, taken over 20 milliseconds. As the underlying electric grid operates at 50 Hz, this means each signal covers a single complete grid cycle. The grid itself operates on a 3-phase power scheme, and all three phases are measured simultaneously.

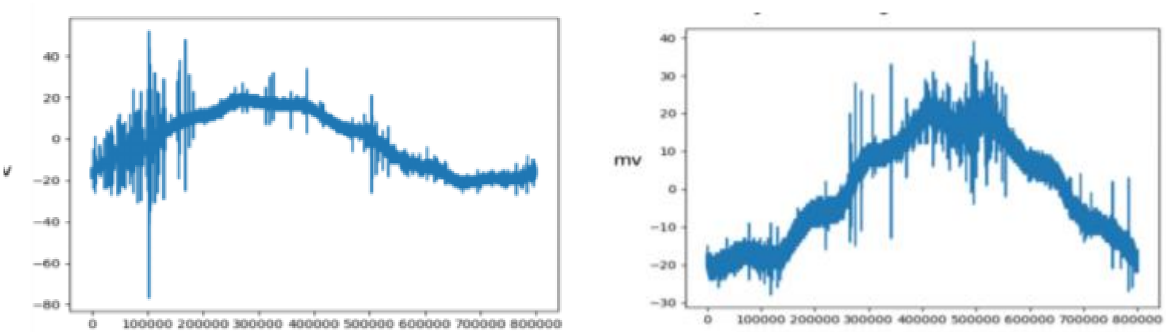
Data Preprocessing and Feature Extraction

The initial data consists of signals which contain :

- 800,000 measurement points for 8712 signals.
- The signals are **three-phased,** so there are 2904 distinct signalling instances.

If we use the data as is and pass it into a neural network for binary classification, it will be challenging for our model to be accurate because of the noise present on the signals.

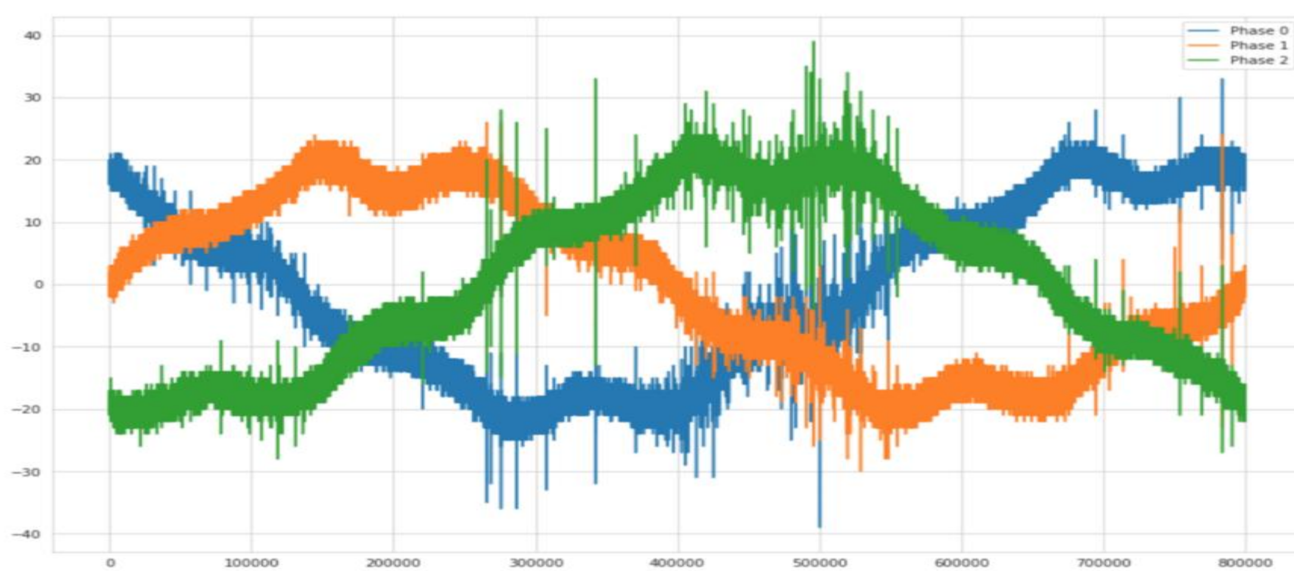
To reduce the noise and increase the data features, we can decouple the signals into their high and low-frequency components using FFT. So the number of signal channels that are fed into the model may very well be diversified using different decouplings in time and frequency domains. The mode of decoupling so far was filtering what we have to get new signals.



(b) PD signal (a) non-PD signal

Sample signal records of the two classes

(a) Partial Discharge, (b) non-partial discharge signals from the dataset

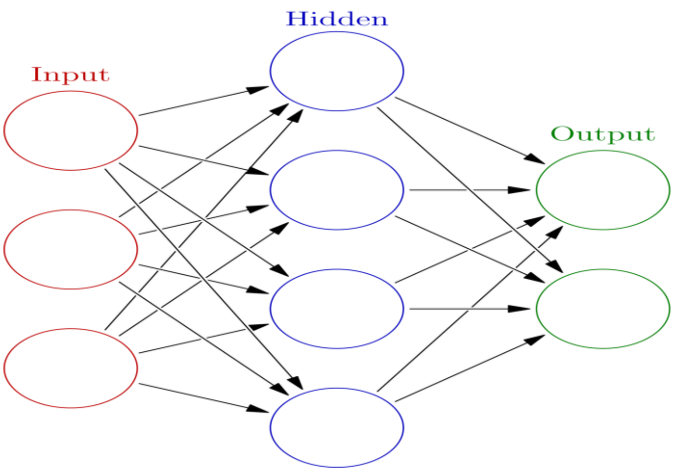


Visualisation of sample signals of the 3-phases captured in the same time frame from the dataset

Keywords and Definitions:

Artificial Neural Networks:

An Artificial Neural Network is a connection of neurons, replicating the structure of the human brain. Each connection of neuron transfers information to another neuron. Inputs are fed into the first layer of neurons, which processes it and transfers to another layer of neurons called hidden layers. After processing information through multiple layers of hidden layers, information is passed to the final output layer.

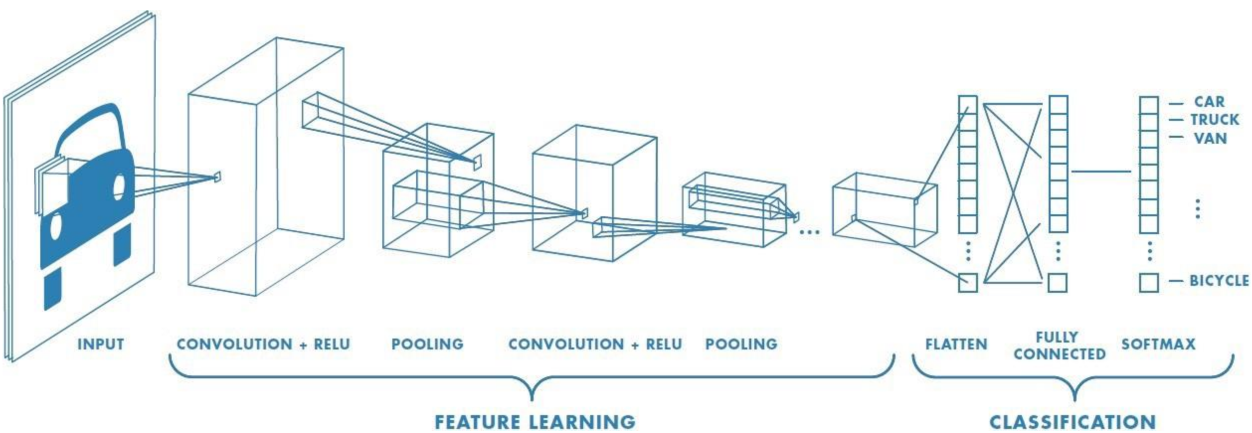


They are capable of learning, and they have to be trained. There are different learning strategies :

1. Unsupervised Learning
2. Supervised Learning
3. Reinforcement Learning

Convolutional Neural Networks:

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm that can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image, and differentiate one from the other. Unlike regular Neural Networks, in the layers of CNN, the neurons are arranged in 3 dimensions: width, height, depth. The neurons in a layer will only be connected to a small region of the layer (window size) before it, instead of all neurons in a fully connected manner. Moreover, the final output layer would have dimensions (number of classes) because by the end of the CNN architecture, we will reduce the full image into a single vector of class scores.

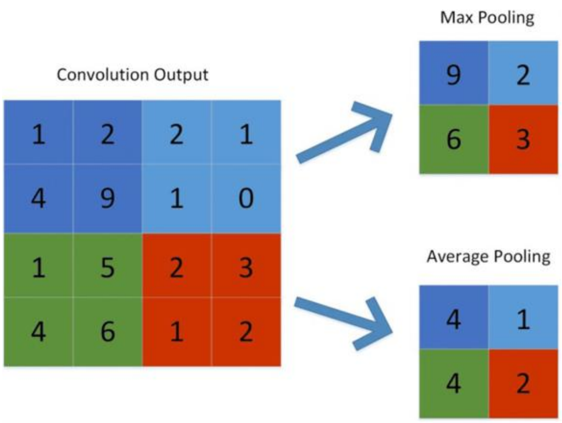


Example of a Convolutional Neural Network

1. ***Convolution Layer*** : In convolution layer we take a small window size (typically of length 3*3 or 5*5) that extends to the depth of the input matrix. The layer consists of learnable filters of window size. During every iteration we slide the window by stride size (typically 1 or 2), and compute the dot product of filter entries and input values at a given position. As we continue this process we will create a 2-Dimensional activation matrix that gives the response of that matrix at every spatial position. That is, the network will learn filters that activate when they see some type of visual feature such as an edge of some orientation or a blotch of some color.

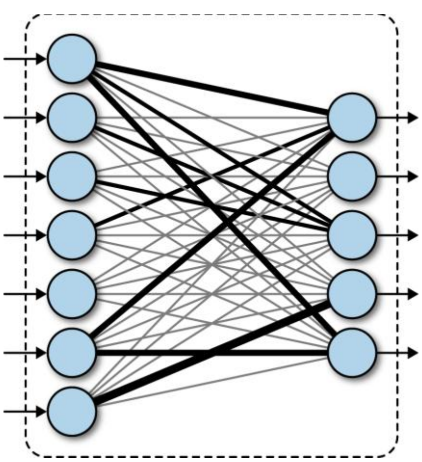
2. ***Pooling Layer*** : We use a pooling layer to decrease the size of the activation matrix and ultimately reduce the learnable parameters. There are two type of pooling :

a) *Max Pooling* : In max pooling we take a window size (for example window of size 2*2), and only take the maximum of 4 values. We will slide this window and continue this process, so we will finally get an activation matrix half of its original Size.
b) *Average Pooling* : In average pooling we take the average of all values in a window.



Working of Pooling

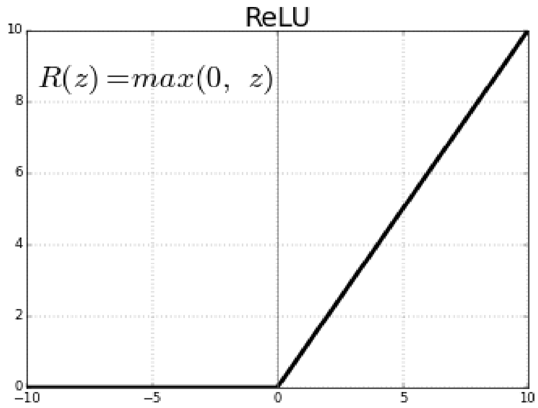
3. ***Fully Connected Layer*** : In convolution layer neurons are connected only to a local region, while in a fully connected region, we will connect all the inputs to neurons. They are also called densely connected layers.



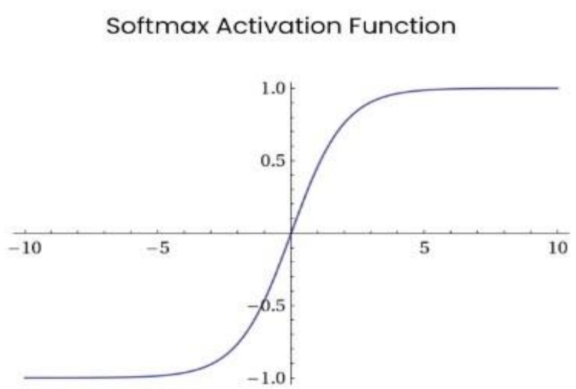
Representation of the fully connected layers

4. ***Activation function*** : the activation function of a node defines the output of that node given an input or set of inputs. Some examples of activation function generally used in Convolution neural networks are –

a.***Rectified Linear Unit (ReLU)*** **:** The rectified linear activation function or ReLU for short is a piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero.



b. **Softmax :** Softmax is a mathematical function that converts a vector of numbers into a vector of probabilities, where the probabilities of each value are proportional to the relative scale of each value in the vector.



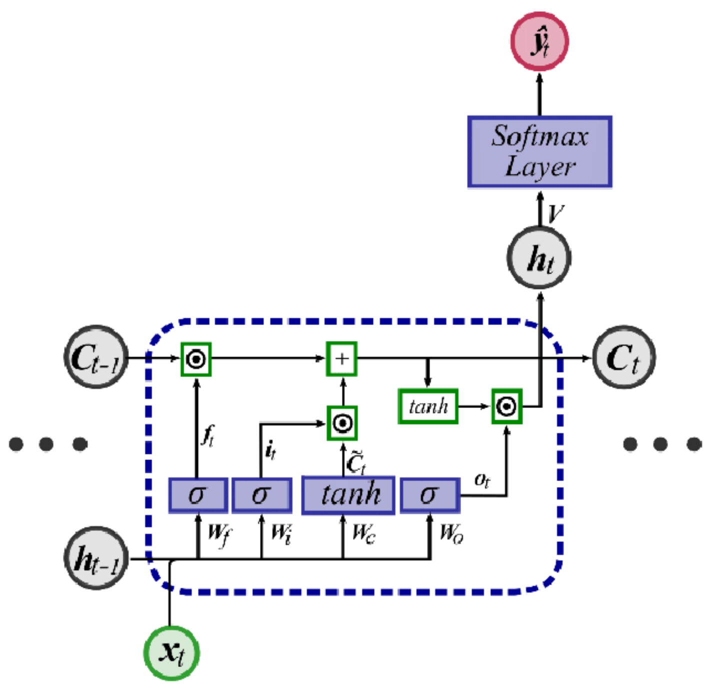
Representation of Softmax activation function

5. Final Output Layer : After getting values from the fully connected layer, we will connect them to the final layer of neurons (having count equal to total number of classes), that will predict the probability of each image to be in different classes.

Long Short-Term Memory Network:

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. It can not only process single data points (such as images), but also entire sequences of data (such as speech or video). For example, LSTM is applicable to tasks such as unsegmented, connected handwriting recognition, speech recognition and anomaly detection in network traffic or IDSs (intrusion detection systems).

A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell.



Block diagram of the LSTM recurrent neural network cell unit

Fast Fourier Transform:

The Fourier Transform of a 1D signal *x* of length *n* is following :

$$f_j = \sum_{k=0}^{n-1} x_k e^{2\pi i jk/n}, \forall\, j = 0, \ldots\ldots, n-1$$

The idea is to represent the signal in the complex space. It is roughly a sum of sinusoidal functions. And there is one coefficient per frequency present in the signal.

The frequency takes the following values :

- f = (1/dn) [0,1,.....(n/2)-1, -n/2,....,-1] if *n* is even
- f = (1/dn) [0,1,.....(n-1)/2, -(n-1)/2,...,-1] if *n* is odd

The structure of an FFT algorithm will be:

First, reordering the data by bit reversal. Then an outer loop is executed *logN* times and calculates transforms of length *2, 4, 8, ..., N*. For each stage of this process, two nested inner loops are used to implement the *Danielson-Lanczos formula*. The input array contains *N* complex time samples in a real array of length *2N*, with real and imaginary parts alternating. The output array contains the complex Fourier spectrum at *N* values of frequency. Real and imaginary parts again alternate.

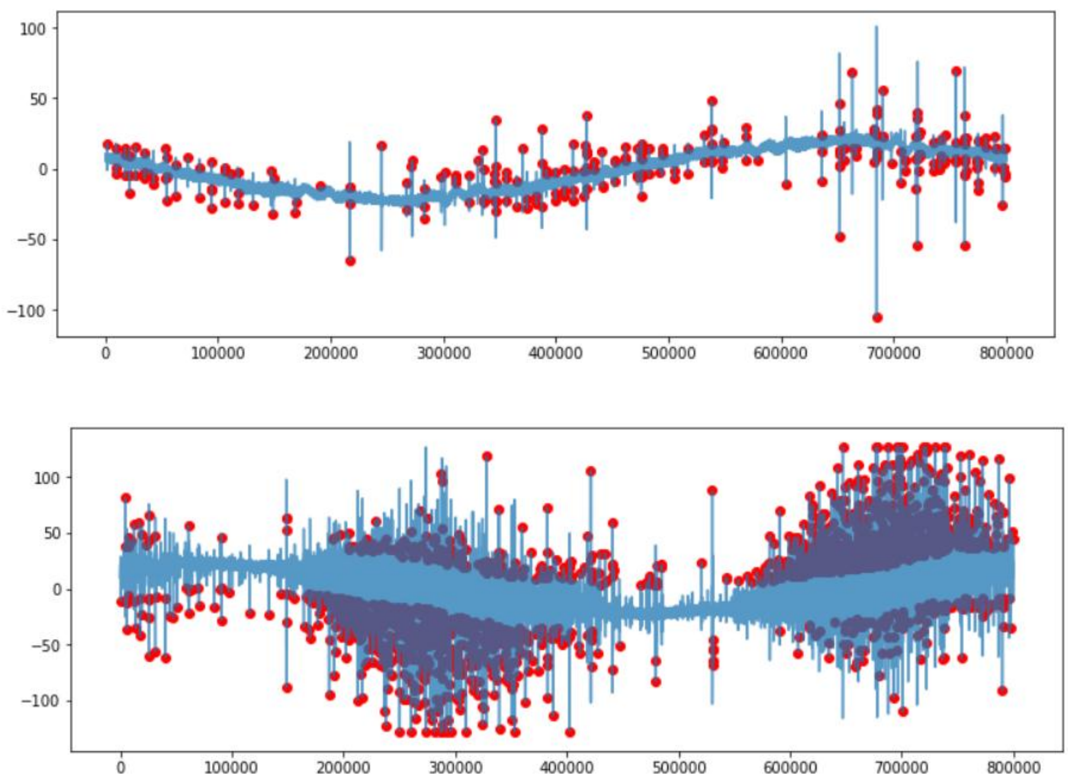
Denoising Data using FFT

The FFT is one of the most important algorithms that have changed the world fundamentally. It offers a computationally fast and efficient way for DFT calculation. In data science, the FFT can be used to denoise data. The algorithm is nontrivial but well-known, and it’s also well packed in many programming languages.

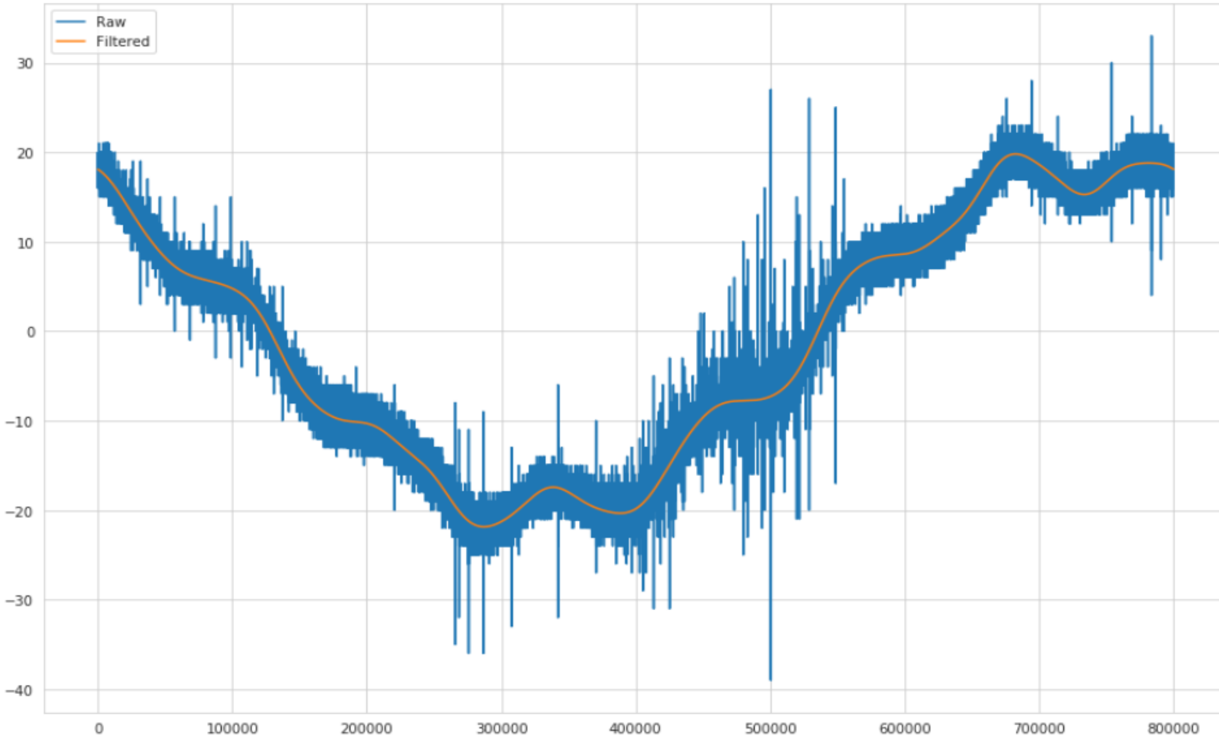
Denoising Algorithm:

The denoising steps are the following :

- Apply the *fft* to the signal
- Compute the frequencies associated with each coefficient
- Keep only the coefficients that have a low enough frequency (in absolute)
- Compute the inverse *fft*



Visualising the signal with the peaks that need to be filtered out



Raw signal and Filtered phase signal with desired threshold using FFT

The Approach and Our Model

In order to classify the powerlines as faulty or not, a binary classification problem, with time dependent data, the signals, we implemented a CNN+LSTM model to approach this problem.
The CNN is used for extraction of features from the signal and the LSTM for the extraction of temporal dependencies.

Tackling the big dataset in front, the signal data was first processed and minimised since 800,000 measurements for a single signal would be computationally too high for a LSTM. Hence each part is shortened and then fed into the model. This becomes a feature in itself. Later we filtered out low and high frequencies by denoising with fast fourier transform and considered it to be additional features for the model.

Another feature would also be the DC sum of the absolute values of the 3-phases for a time-frame. With first part of this model being CNN for the features and the second part of the same model is a LSTM as mentioned before, to deal with the time-series problem.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
time_distributed_6 (TimeDist (None, None, 8, 64)       832
_________________________________________________________________
time_distributed_7 (TimeDist (None, None, 6, 64)       12352
_________________________________________________________________
time_distributed_8 (TimeDist (None, None, 6, 64)       0
_________________________________________________________________
time_distributed_9 (TimeDist (None, None, 3, 64)       0
_________________________________________________________________
time_distributed_10 (TimeDis (None, None, 192)         0
_________________________________________________________________
lstm_2 (LSTM)                (None, 100)               117200
_________________________________________________________________
dropout_4 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_3 (Dense)              (None, 100)               10100
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 101
=================================================================
Total params: 140,585
Trainable params: 140,585
Non-trainable params: 0
_________________________________________________________________
```

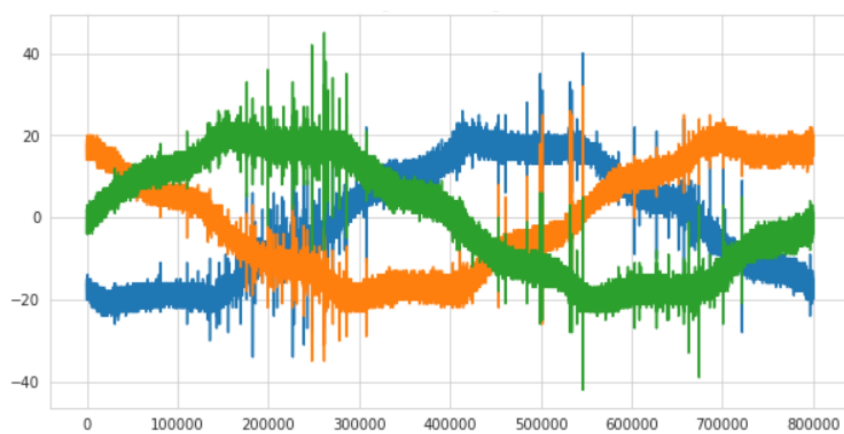
The Summary of the model

RESULTS

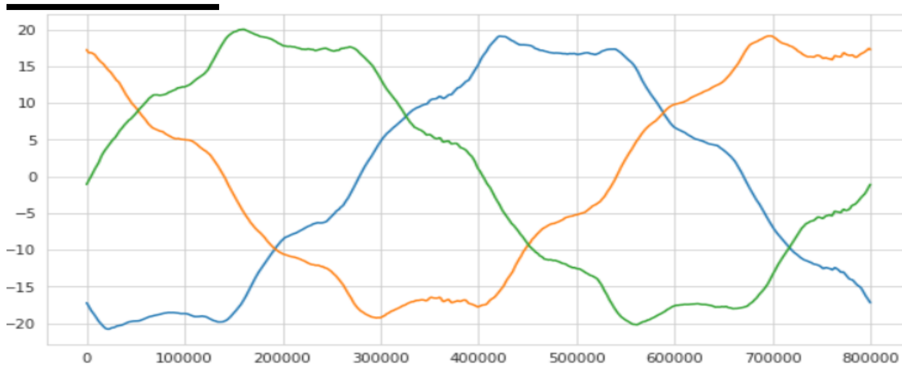
If we use the initial data without doing any pre-processing and feature engineering we would get an AUC(Area Under Curve-for binary classification) score of about 0.8-0.87 and 0.45LB score. This is because it becomes very difficult for our model to deal with the noise present in the signals.

However, after doing denoising with the help of FFT, we were able to increase AUC score to 0.92-0.95 and 0.51LB score. Here is a look at the difference in the quality of input signal before and after denoising.

BEFORE



AFTER



CONCLUSION

By retraining the model three times, we were able to get this accuracy. Further improvements may be made if we increase the number of epochs and play around with the threshold.

Perhaps if we use the data obtained from the spectral analysis as extra features, we can further increase the accuracy.

Other Models Possible

Logistic Regression- We can first use stratified k fold cross-validation to find the best hyperparameter and then train the model using the best hyperparameter.

Random Forest classifier is another approach which can be used as well.

There are various other Thresholding methods for denoising the signals as well but we chose fast fourier transform because it works well with our model.

Future Scope

We have used the Knowledge of Deep Learning (CNN+LSTM) in this project to detect partial discharge and overcome fault detection in signals.

We are currently using fft to split the signals into high and low frequency channels. We can further improve the results by using spectral analysis and adding them as additional features. Further, we can enhance the experiment with cross-validation, adaptive learning rate, early stopping, ensembling etc.

There are many organisations which are working on detecting these faults by using smart meters equipped with these types of models.

References

1. https://arxiv.org/abs/1905.01588
2. https://arxiv.org/abs/2011.02336
3. https://realpython.com/python-scipy-fft/
4. https://bigdata.seas.gwu.edu/data-set-5-vsb-power-line-fault-detection-data-set/