Power Quality Classification

—

TAG - G

Agenda

1) Problem statement
2) EDA on both the datasets.
3) Discussion about the pre-processing techniques employed.
4) Using 1-D CNN and MLP technique on the normalized data.
5) Comparison based on various metrics like no. of parameters used, RAM usage, model performance in time and frequency domain etc.
6) Applying ML techniques to the data which was normalized after converting it into frequency domain.

Problem Statement

This problem statement consists of two datasets. The dataset consists of power signals. Our aim is to classify them according to their power quality condition.

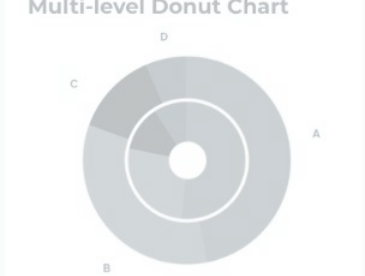
Dataset 1:

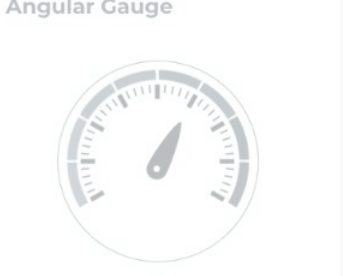
1) Sampling rate of 128 hertz and there are 5 classes.
2) This dataset does not have noise.

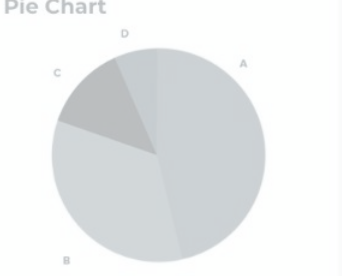
Dataset 2:

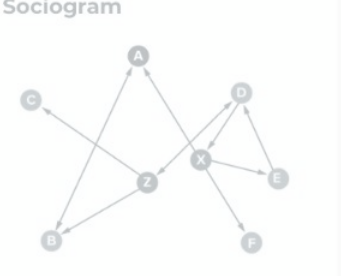
1) Sampling rate 256 hertz and there are 6 classes.
2) This dataset has noise.

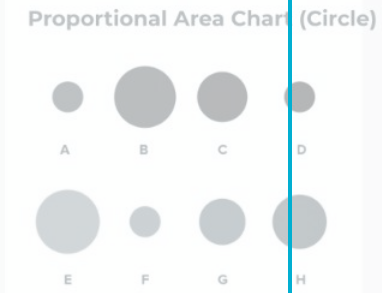
Exploratory Data Analaysis

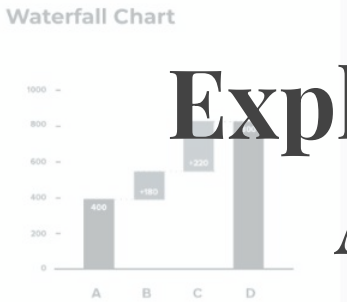


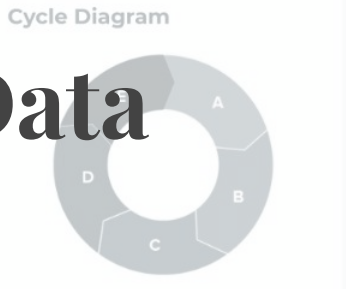


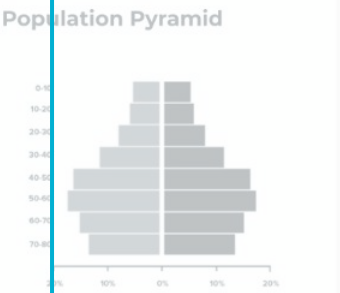


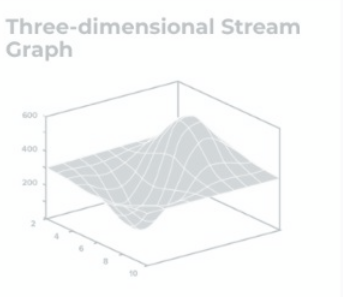


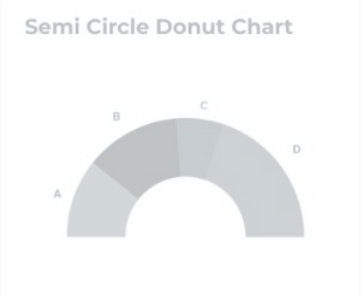








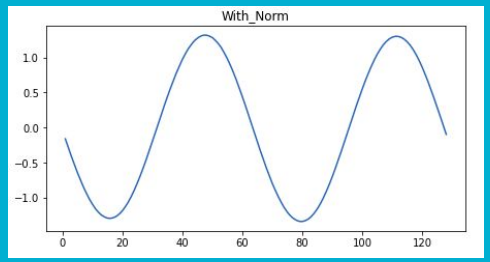




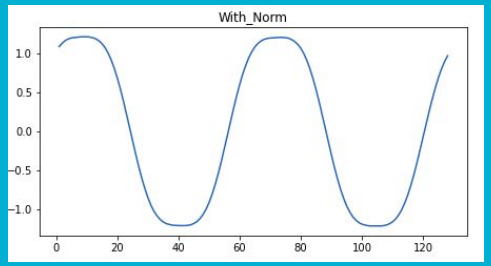
EDA on Dataset - 1 (Classes - 5, Sampling Rate - 128, W/o Noise)

With Normalization:

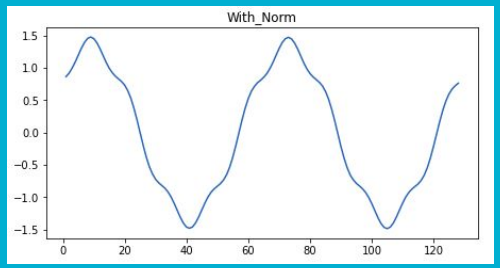
Type - 1 :



Type - 2 :

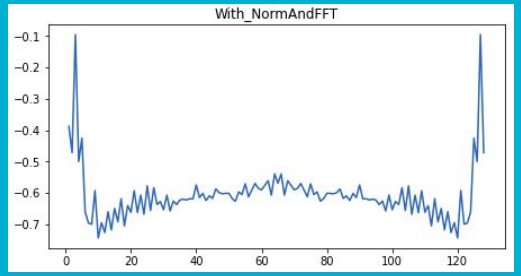


Type - 3 :

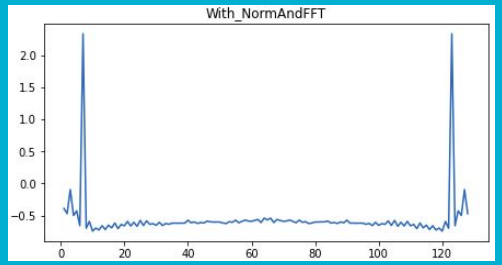


With Normalization and FFT:

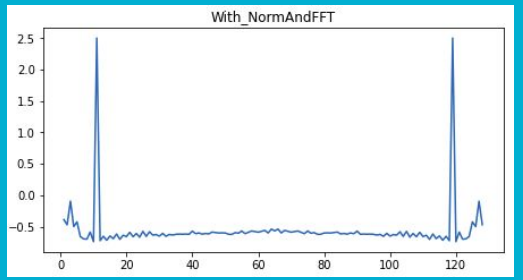
Type - 1 :



Type - 2 :



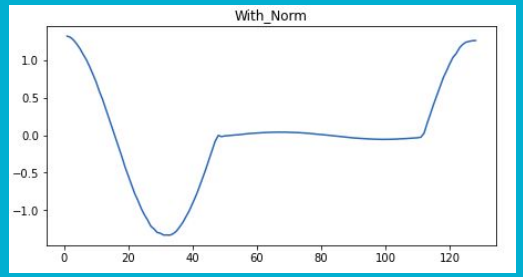
Type - 3 :



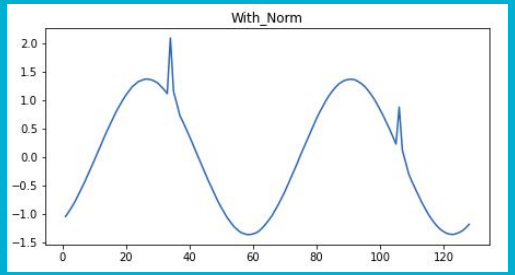
EDA on Dataset - 1 (Continued.....)

With Normalization:

Type - 4 :

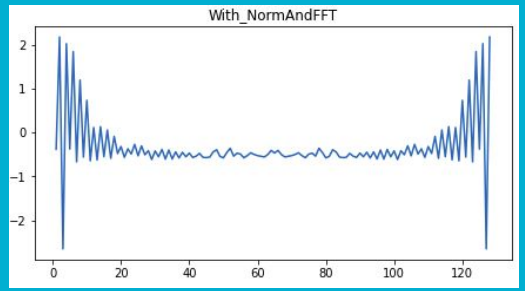


Type - 5 :

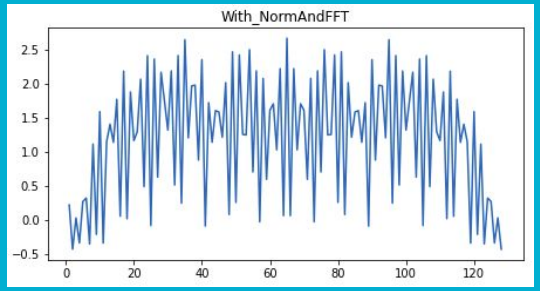


With Normalization and FFT:

Type - 4 :



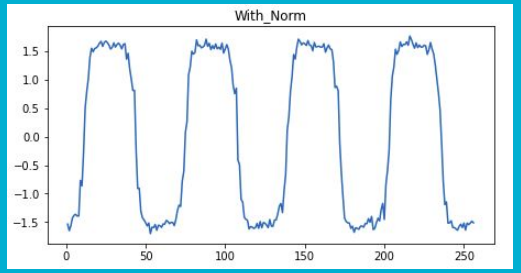
Type - 5 :



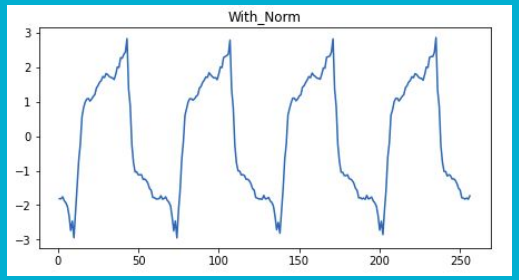
EDA on Dataset - 2 (Classes - 6, Sampling Rate - 256, With Noise)

With Normalization:

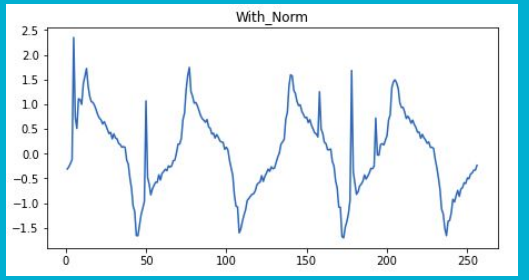
Type - 1 :



Type - 2 :

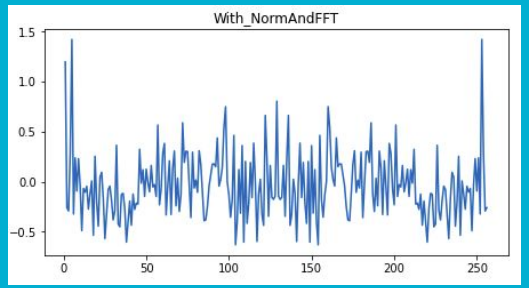


Type - 3 :

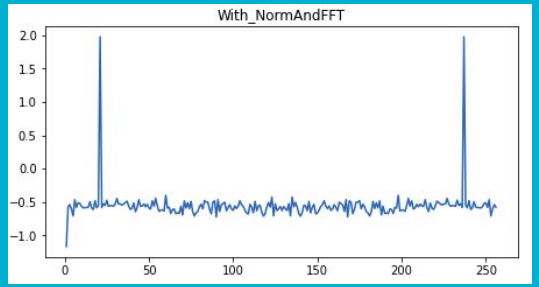


With Normalization and FFT:

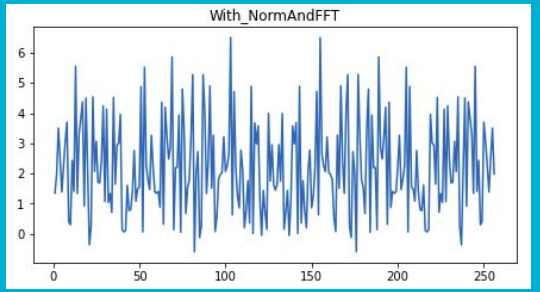
Type - 1 :



Type - 2 :



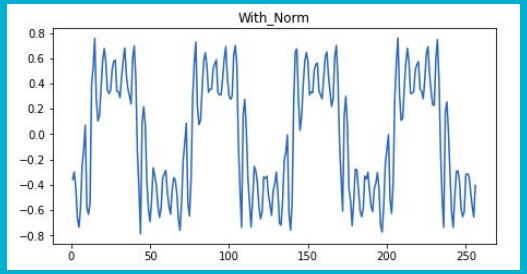
Type - 3 :



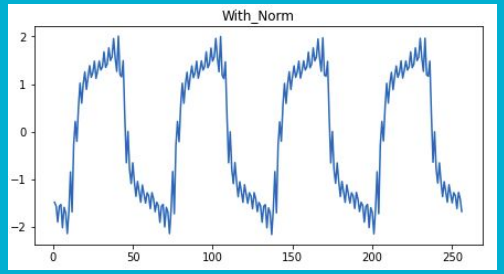
EDA on Dataset - 2 (Continued.....)

With Normalization:

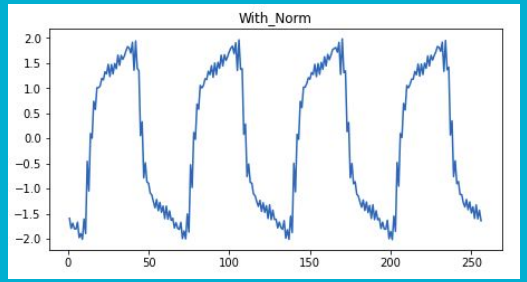
Type - 4 :



Type - 5 :

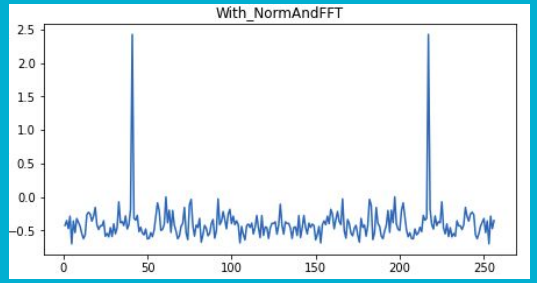


Type - 6 :

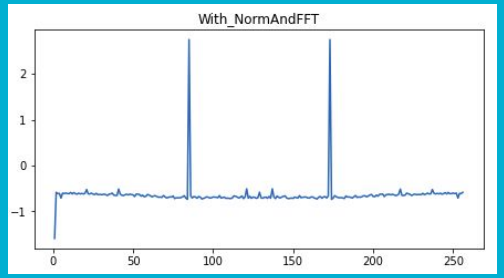


With Normalization and FFT:

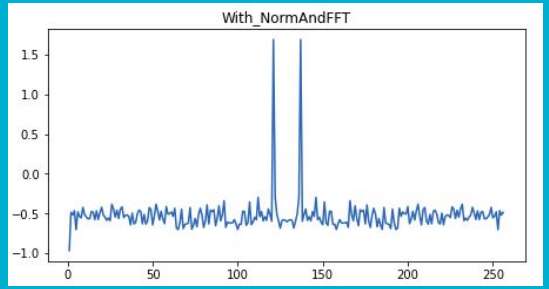
Type - 4 :

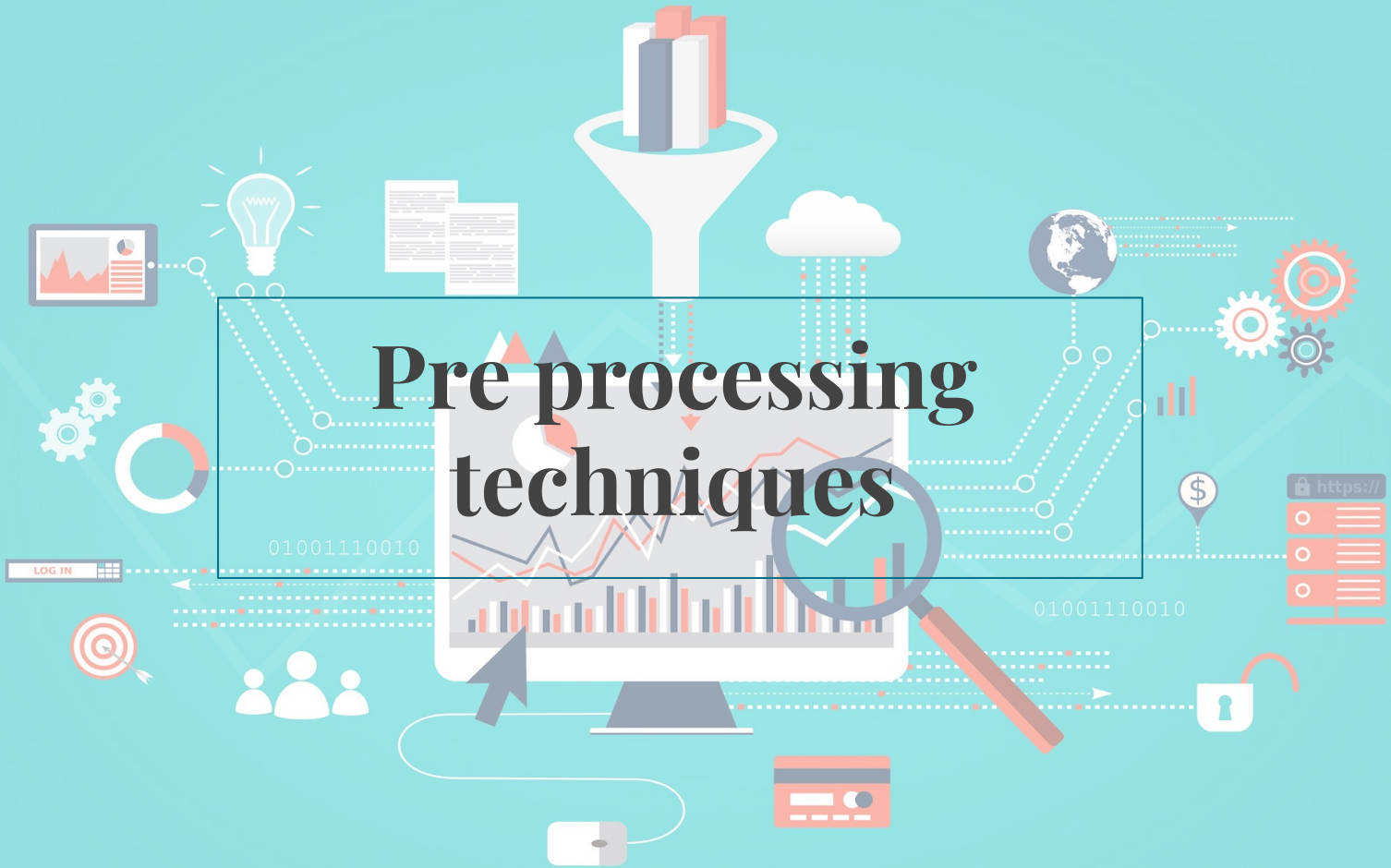


Type - 5 :



Type - 6 :




Pre processing techniques
LOG IN
https://
01001110010
01001110010

Pre processing techniques

1) Fast Fourier Transform

We have used FFT here because the features in the frequency domain which are frequency and phase have been proven to be suitable features for classifying the data.

FFT is a technique which computes the DFT of a sequence in an efficient manner. It actually converts the wave from time domain to frequency domain. DFT is obtained by decomposing a sequence of values into components of different frequencies.

Let $x_0, \ldots, x_{N-1}$ be complex numbers. The DFT is defined by the formula

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i 2\pi k n / N} \qquad k = 0, \ldots, N-1,$$

where $e^{i2\pi/N}$ a primitive Nth root of 1.

Pre processing techniques (cont..)

2) Normalization

After doing the FFT we can see the output wave has spikes out at two positions which are the frequencies of that wave. The other points of the wave are way smaller than these two and that is why there is chance that their contribution might be neglected and the resulting model might end up creating a bias. So to avoid this we are doing normalization.

Standardization:

$$z = \frac{x - \mu}{\sigma}$$

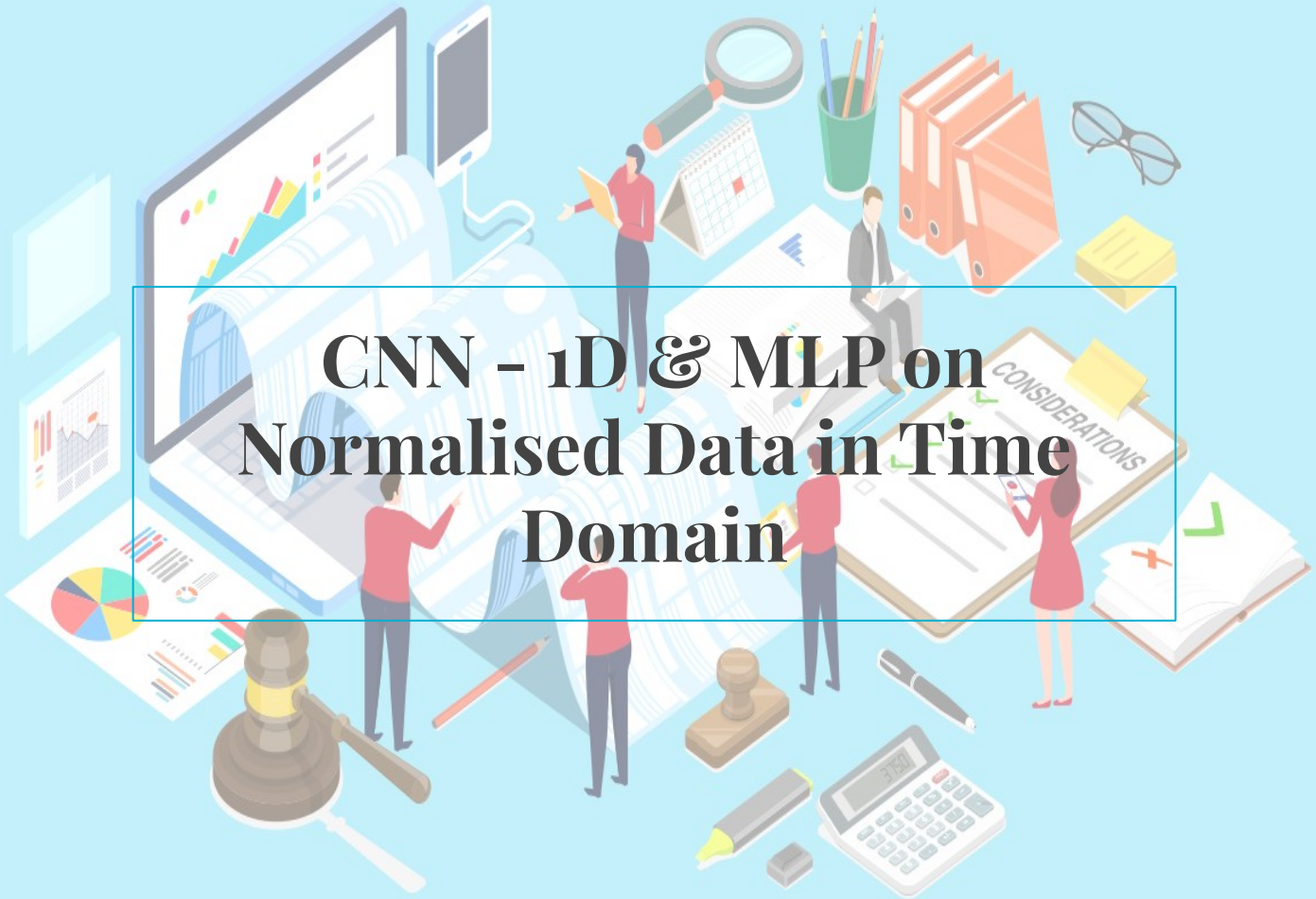
with mean:

$$\mu = \frac{1}{N}\sum_{i=1}^{N}(x_i)$$

and standard deviation

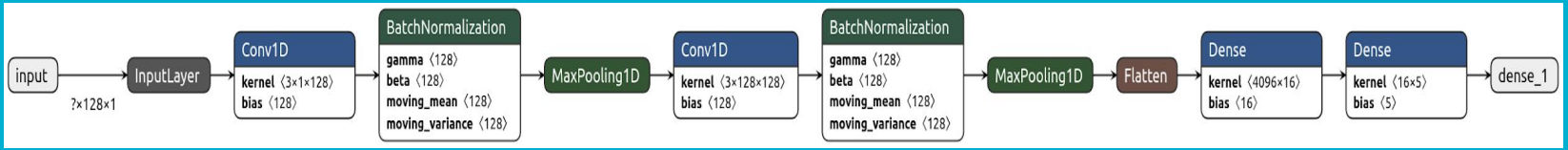
$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

CNN - 1D & MLP on Normalised Data in Time Domain



1D - Convolutional Neural Network (Dataset - 1)

Architecture:



Accuracy:

	MIN	AVG	MAX
Validation	58.08%	94.82%	100%
Training	57.69%	97.63%	100%
Test accuracy	100.00%		

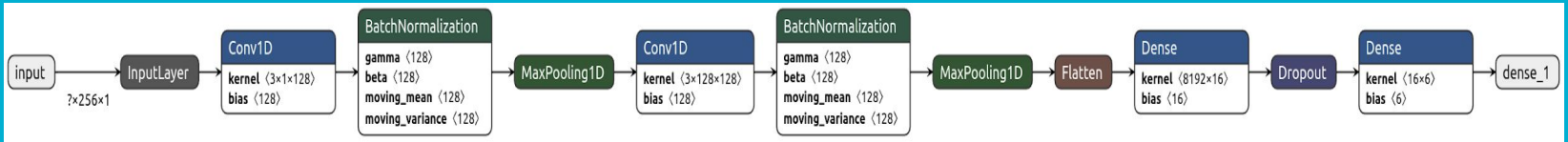
Summary:

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 128, 128)	512
batch_normalization (BatchNo	(None, 128, 128)	512
max_pooling1d (MaxPooling1D)	(None, 64, 128)	0
conv1d_1 (Conv1D)	(None, 64, 128)	49280
batch_normalization_1 (Batch	(None, 64, 128)	512
max_pooling1d_1 (MaxPooling1	(None, 32, 128)	0
flatten (Flatten)	(None, 4096)	0
dense (Dense)	(None, 16)	65552
dense_1 (Dense)	(None, 5)	85

Total params: 116,453
Trainable params: 115,941
Non-trainable params: 512

1D - Convolutional Neural Network (Dataset - 2)

Architecture:



Accuracy:

	MIN	AVG	MAX
Validation	23.42%	79.54%	100%
Training	75.88%	98.73%	99.53%
Test accuracy	99.44%		

Summary:

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 256, 128)	512
batch_normalization (BatchNo	(None, 256, 128)	512
max_pooling1d (MaxPooling1D)	(None, 128, 128)	0
conv1d_1 (Conv1D)	(None, 128, 128)	49280
batch_normalization_1 (Batch	(None, 128, 128)	512
max_pooling1d_1 (MaxPooling1	(None, 64, 128)	0
flatten (Flatten)	(None, 8192)	0
dense (Dense)	(None, 16)	131088
dense_1 (Dense)	(None, 6)	102

Total params: 182,006
Trainable params: 181,494
Non-trainable params: 512

Multi-Layer Perceptron (Dataset - 1)

Architecture:

Accuracy:

	MIN	AVG	MAX
Validation	43.42%	84.33%	99.50%
Training	29.40%	82.70%	98.78%
Test accuracy	98.37%		

Summary:

```
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 64)                8256
_________________________________________________________________
dense_1 (Dense)              (None, 32)                2080
_________________________________________________________________
dense_2 (Dense)              (None, 16)                528
_________________________________________________________________
dense_3 (Dense)              (None, 5)                 85
=================================================================
Total params: 10,949
Trainable params: 10,949
Non-trainable params: 0
_________________________________________________________________
```

Multi-Layer Perceptron (Dataset - 2)

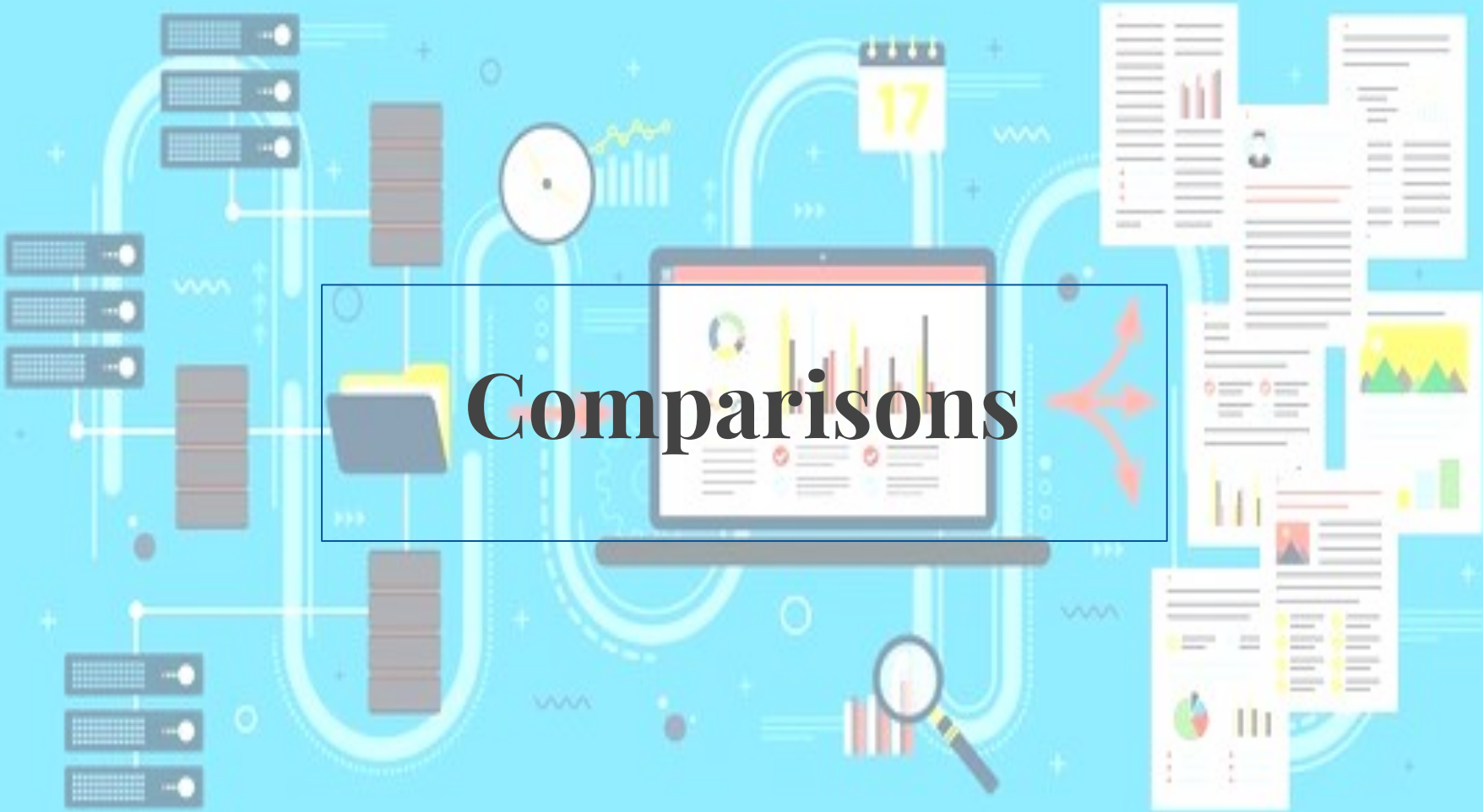
Architecture:

Accuracy:

	MIN	AVG	MAX
Validation	85.06%	90.11%	94.33%
Training	46.00%	91.12%	95.25%
Test accuracy	93.77%		

Summary:

```
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 64)                16448
_________________________________________________________________
dense_1 (Dense)              (None, 32)                2080
_________________________________________________________________
dense_2 (Dense)              (None, 16)                528
_________________________________________________________________
dense_3 (Dense)              (None, 6)                 102
=================================================================
Total params: 19,158
Trainable params: 19,158
Non-trainable params: 0
_________________________________________________________________
```


Comparisons

Comparison - MLP (Dataset - 2)

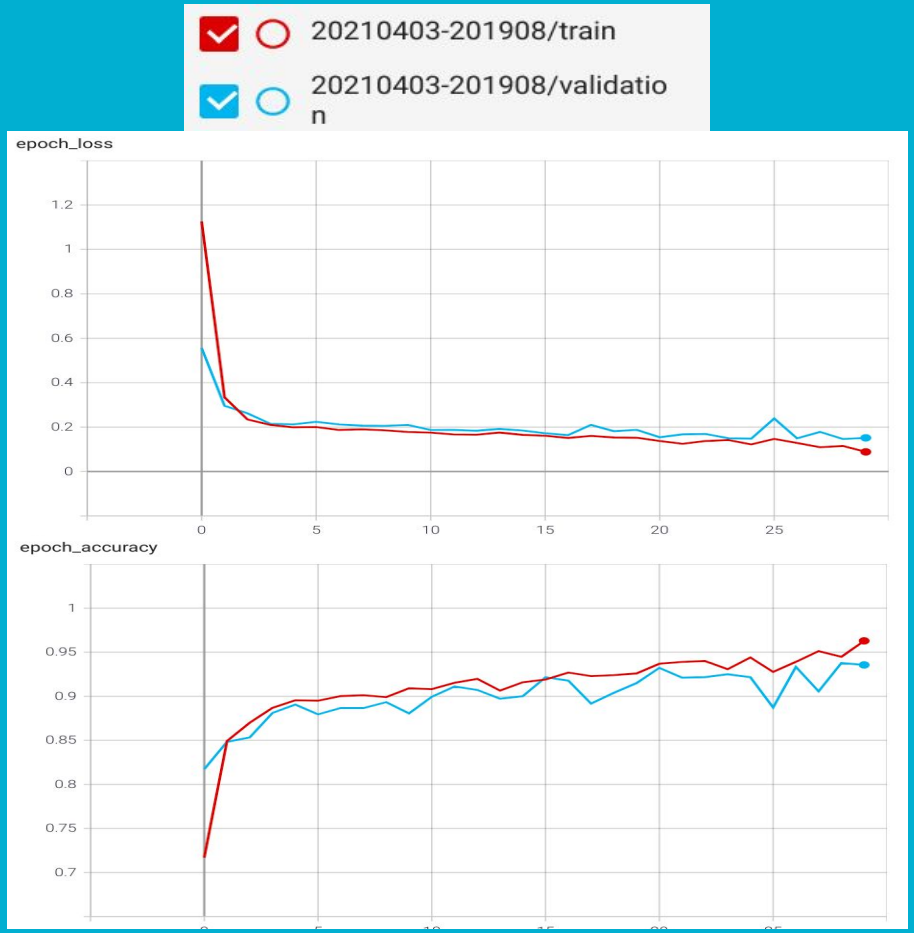
Accuracy - W/O FFT:

	MIN	AVG	MAX
Validation	85.06%	90.11%	94.33%
Training	46.00%	91.12%	95.25%
Test accuracy	93.77%		

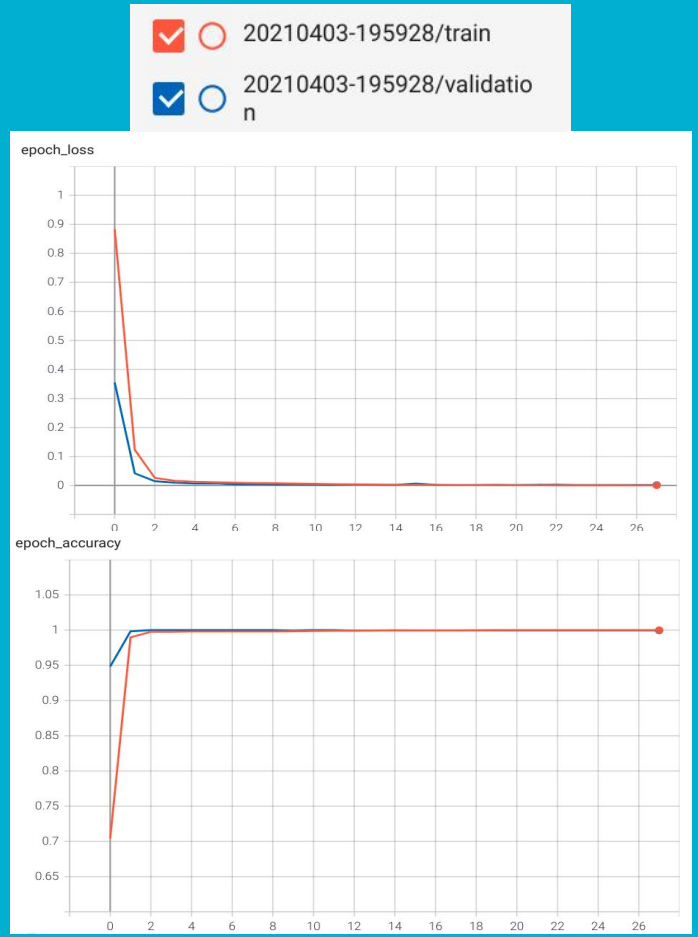
Accuracy - With FFT:

	MIN	AVG	MAX
Validation	94.83%	99.78%	100%
Training	70.39%	98.90%	100%
Test accuracy	99.38%		

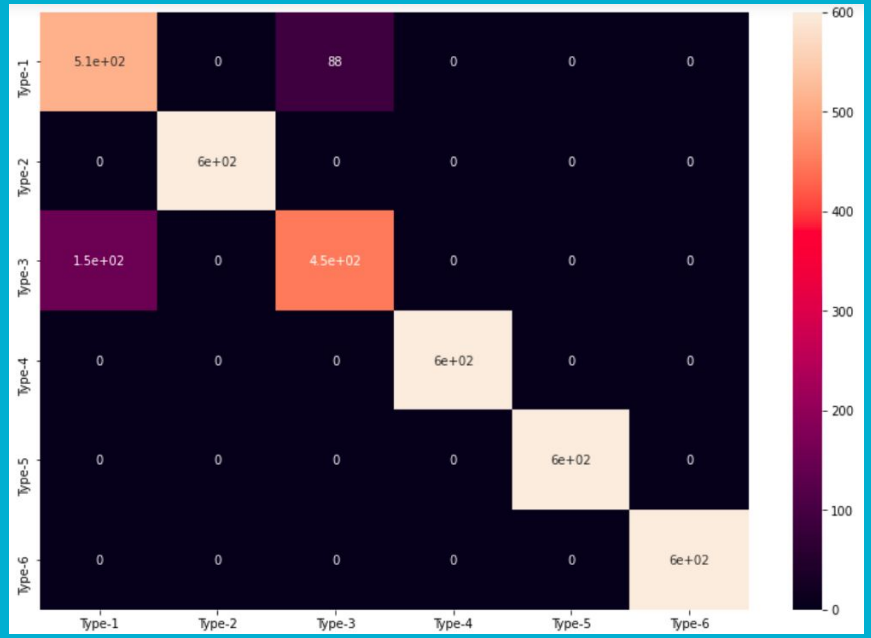
W/O FFT:



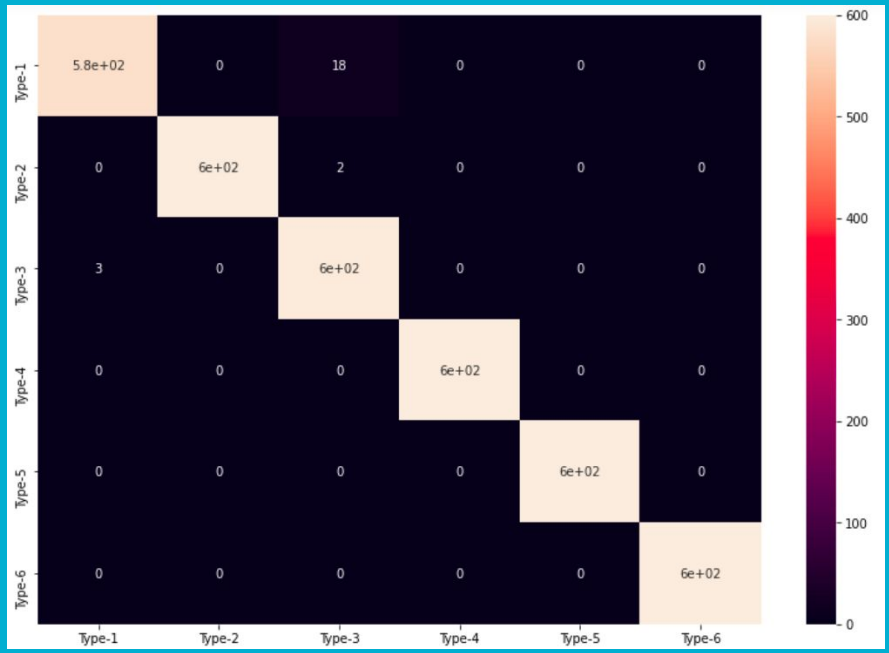
With FFT:



W/O FFT:



With FFT:



Comparison - CNN - 1D (Dataset - 2)

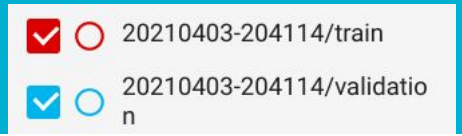
Accuracy - W/O FFT:

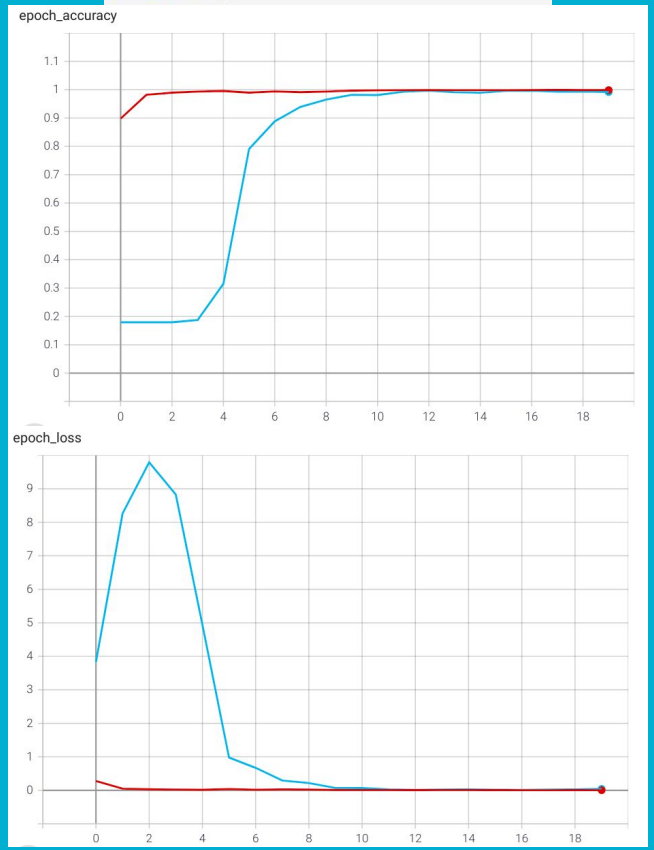
	MIN	AVG	MAX
Validation	23.42%	79.54 %	100%
Training	75.88%	98.73 %	99.53%
Test accuracy	99.44%		

Accuracy - With FFT:

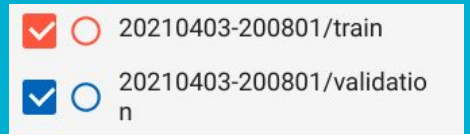
	MIN	AVG	MAX
Validation	51.08 %	90.69 %	100%
Training	97.49 %	99.84 %	100%
Test accuracy	99.80%		

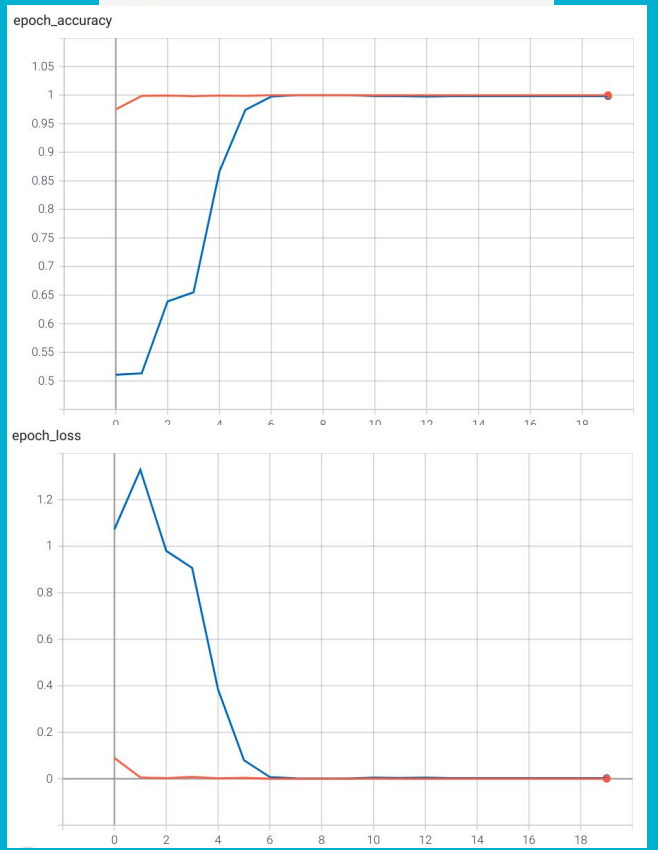
W/O FFT:



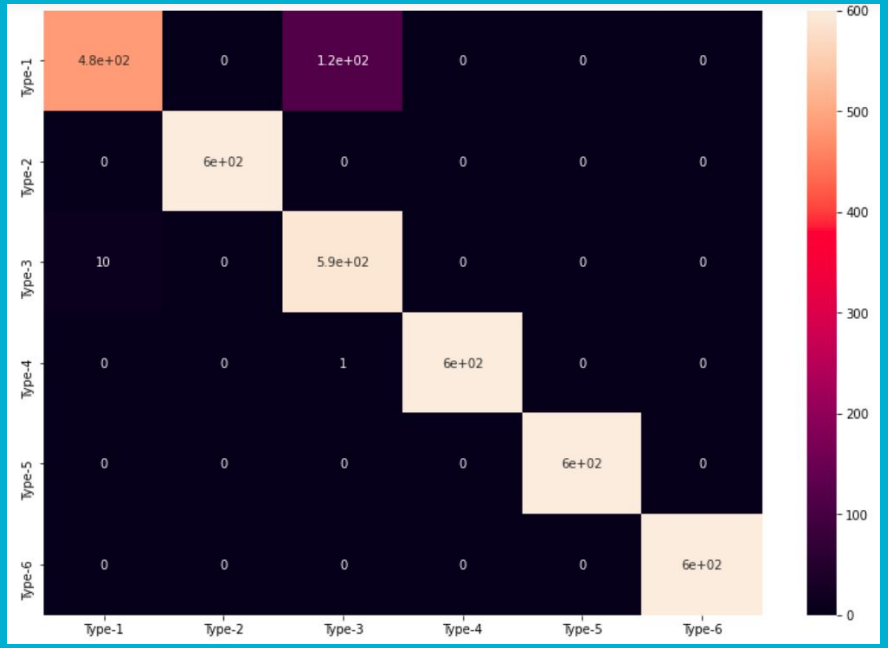


With FFT:

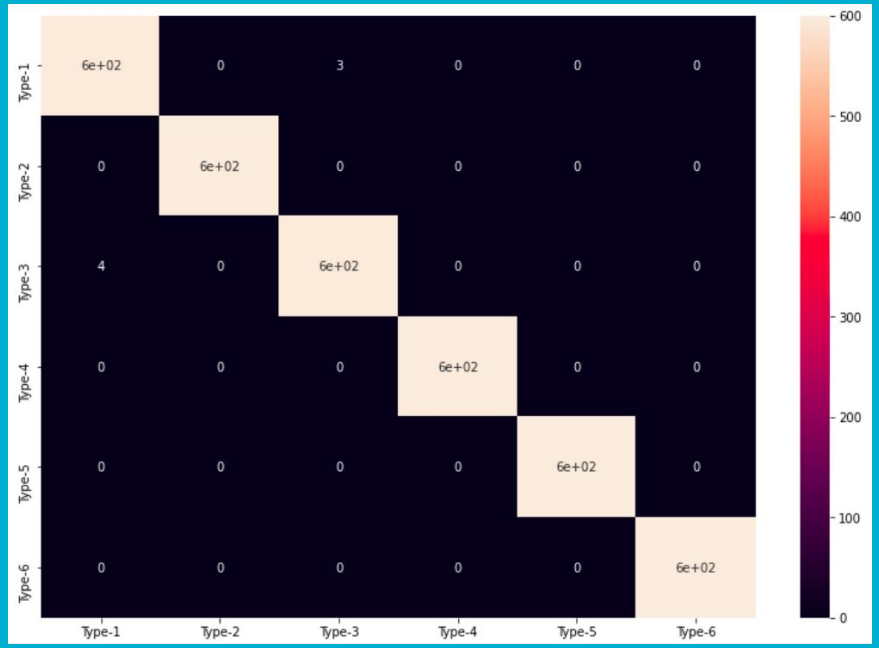




W/O FFT:



With FFT:



Memory & RAM Utilization (STM32CubeMX)

Multi-Layer Perceptron

1D – Convolutional Neural Network

Using ML Algorithms

	Test Accuracy	
	W/O FFT	With FFT
Radom Forest (n_estimators=24)	89.05%	99.33%
Gaussain Naive-Bayes	80.82%	99.69%
Support Vector Machine	94.33%	99.80%

Thank You