Epileptic Seizure Classification

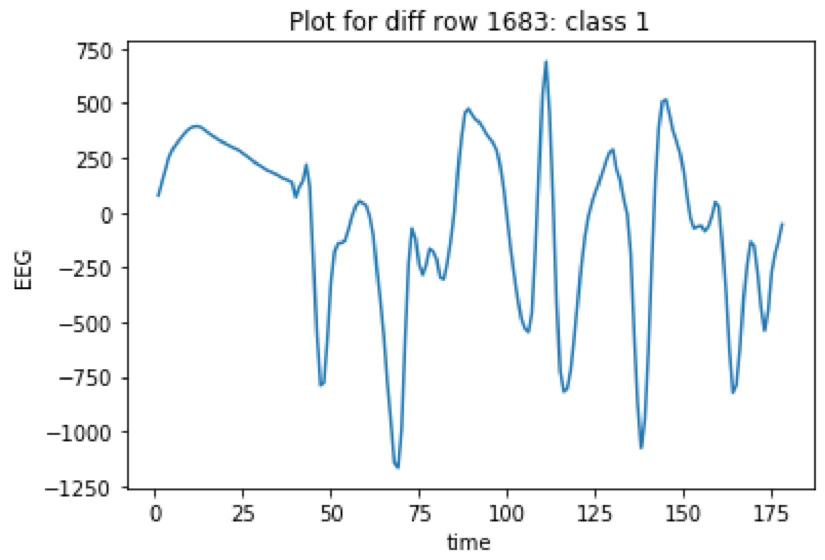
Group 15

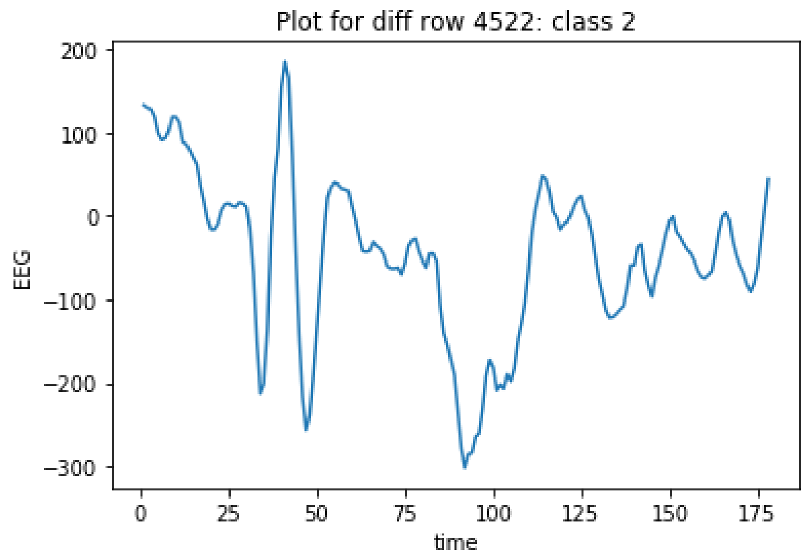
Ankur Parihar	16114015
Parvat Yadav	16114043
Rishikesh Chaudhary	16114054
Sagar Dhurwe	16114059

Data Overview

- 11500 single channel EEG waves in csv.
- 178 data column in each wave corresponding to 1 second snapshot.
- Last column describing one of five categories
  1. Recording of seizure activity
  2. Tumor identified and recording from infected area
  3. Tumor identified and recording from healthy area
  4. Patient had their eyes closed while recording EEG
  5. Patient had their eyes opened while recording EEG
- All subjects falling in classes 2, 3, 4, and 5 are subjects who did not have epileptic seizure. Only subjects in class 1 have epileptic seizure.

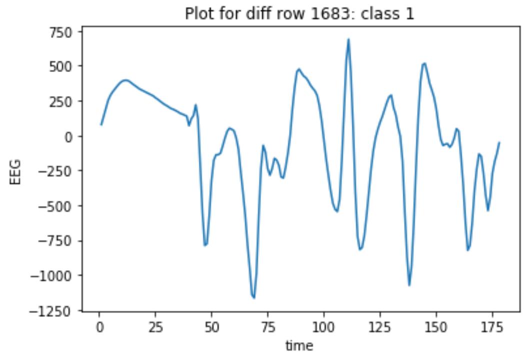
Sample Data Plots

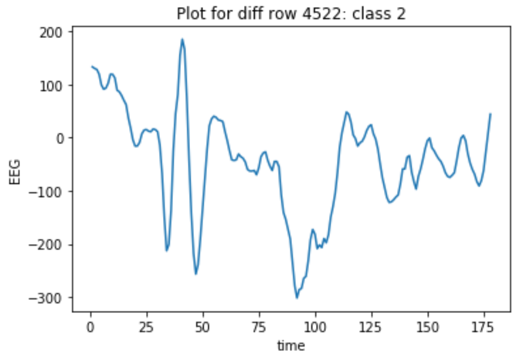


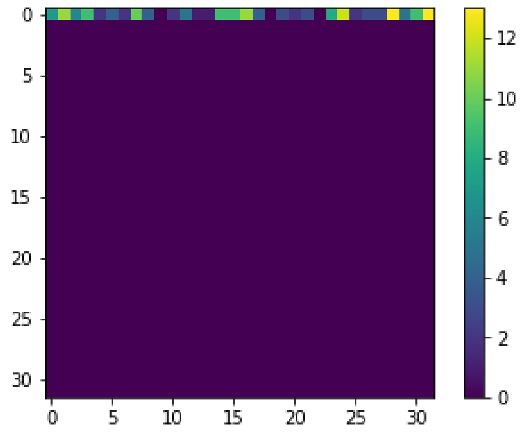


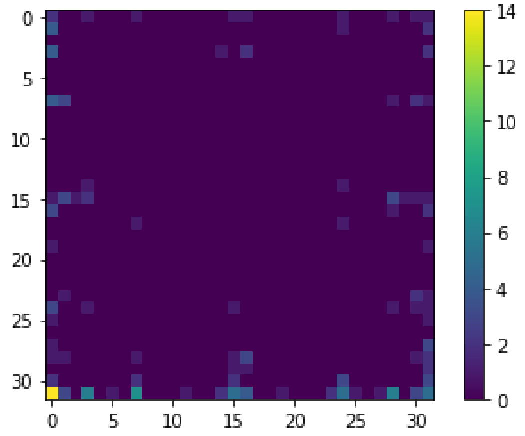
Stage 1: pulseNet

- Designed for detecting pulses in wave
- Its an extension and can be used to transform wave into categorical structure









Iteration 1

- Define an upper limit and lower limit
- Average of two data points define *under* or *over* limit.

Example

Upper_lim = 2, Lower_lim = -2

$$4, 3, 2, 1, 0, 1, 2, 3, 4 \\ \downarrow \\ 3, 3, 1, 1, 0, 0, 2, 2$$

$$if\ (avg > limit_{upper}\ or\ avg < limit_{lower})\ limit = under\ else\ limit = over$$

- Based on limit specification and type (*Increment* or *Decrement*) one of four categories is added to a list.

0 – Increment, under limit

1 – Decrement, under limit

2 – Increment, over limit

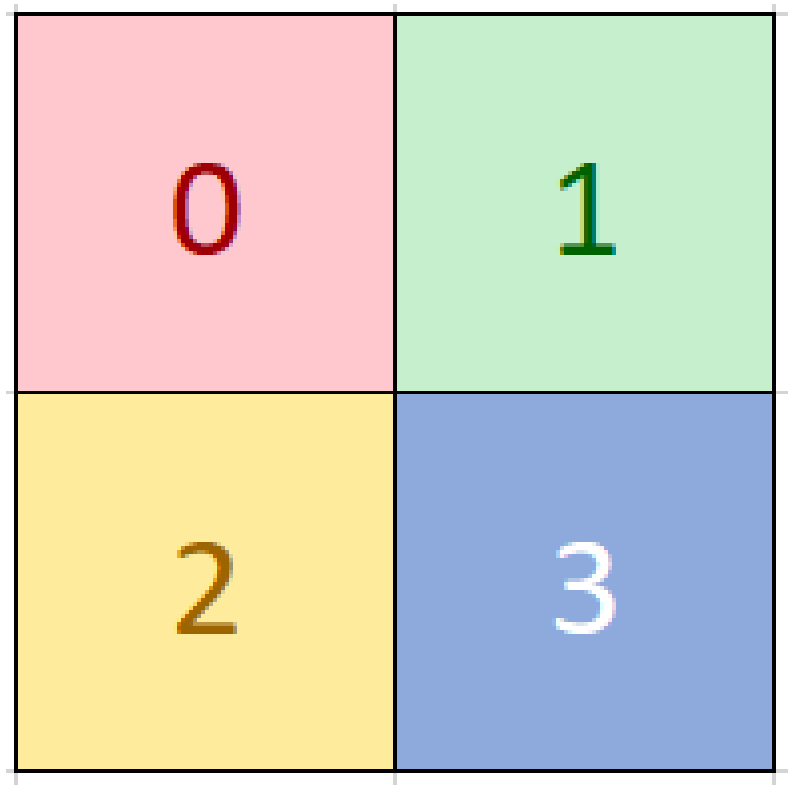
3 – Decrement, over limit

- Final list contains only categorical data

Iteration 2

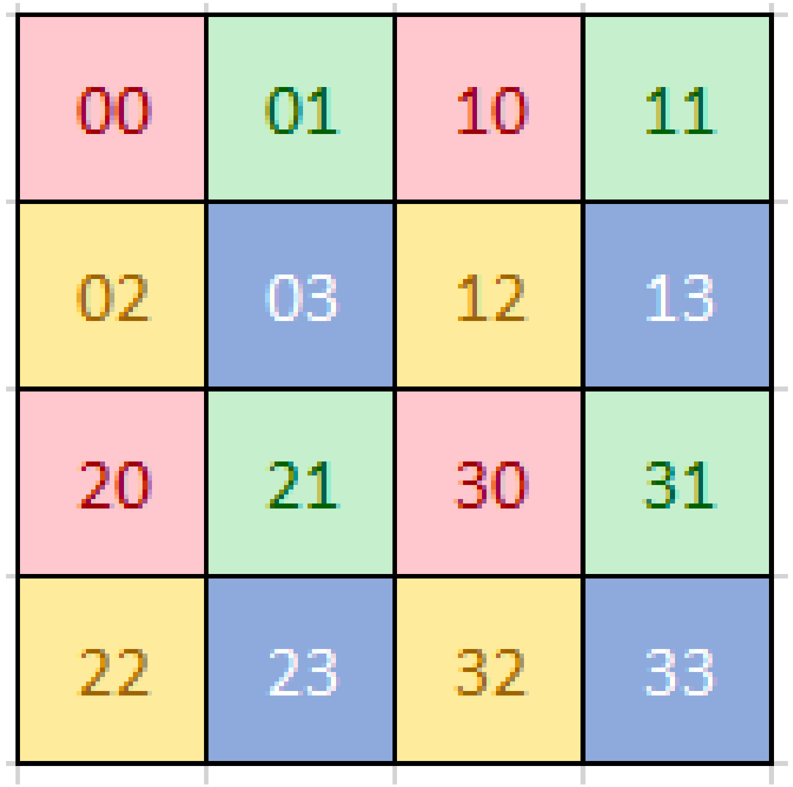
- Define a *pulselen* denoting grouping of wave data to analyze in continuous manner.
- Initialize an empty matrix with 0s. Matrix side is calculated as $side = (n_{cat})^{pulselen/2}$
- Select wave data groups with $stride = 1$. The data is called *pulse signature* and corresponds to an *address* of matrix cell. A*ddress* is used to determine location on matrix grid.
- Matrix is defined in a block and level scheme.
- Largest or upper level blocks are the most significant digit of the address and lowest or smallest level blocks denote the least significant digit of the address.

Matrix: Structure



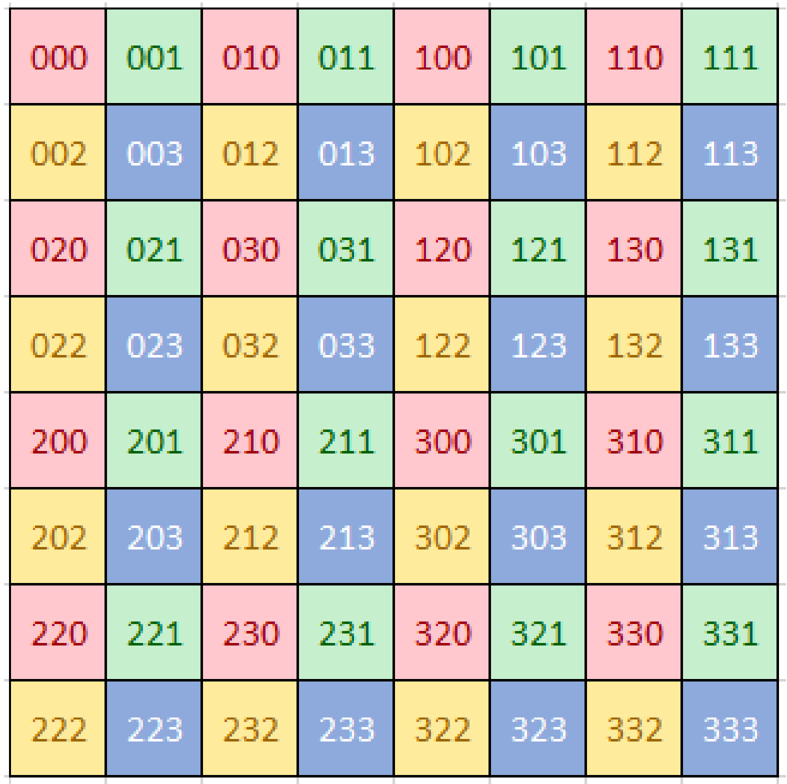
Level = 1

Matrix: Structure



Level = 2

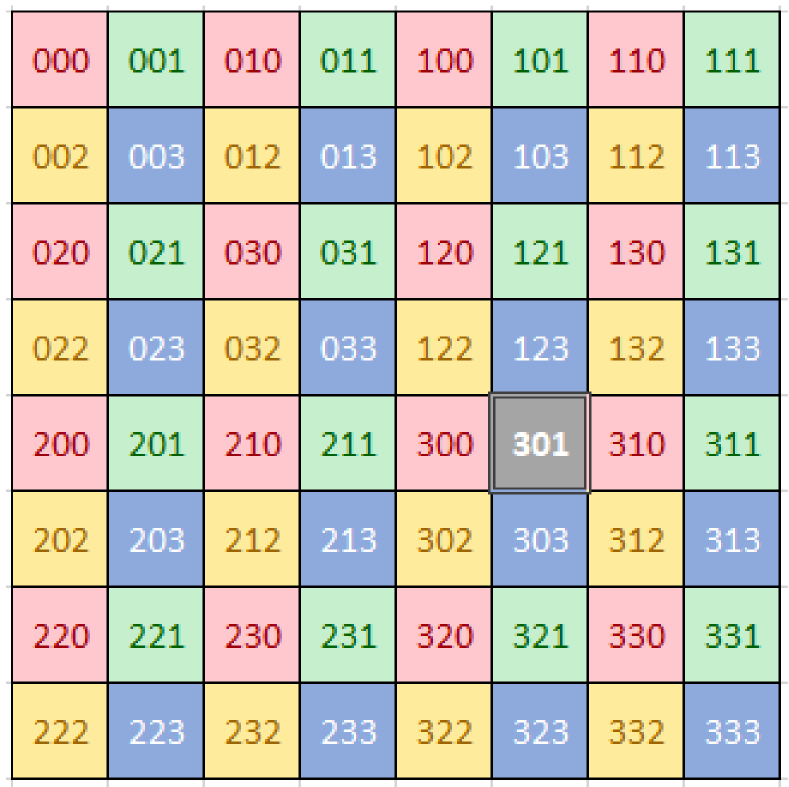
Matrix: Structure



000	001	010	011	100	101	110	111
002	003	012	013	102	103	112	113
020	021	030	031	120	121	130	131
022	023	032	033	122	123	132	133
200	201	210	211	300	301	310	311
202	203	212	213	302	303	312	313
220	221	230	231	320	321	330	331
222	223	232	233	322	323	332	333

Level = 3

Matrix: Structure



000	001	010	011	100	101	110	111
002	003	012	013	102	103	112	113
020	021	030	031	120	121	130	131
022	023	032	033	122	123	132	133
200	201	210	211	300	**301**	310	311
202	203	212	213	302	303	312	313
220	221	230	231	320	321	330	331
222	223	232	233	322	323	332	333

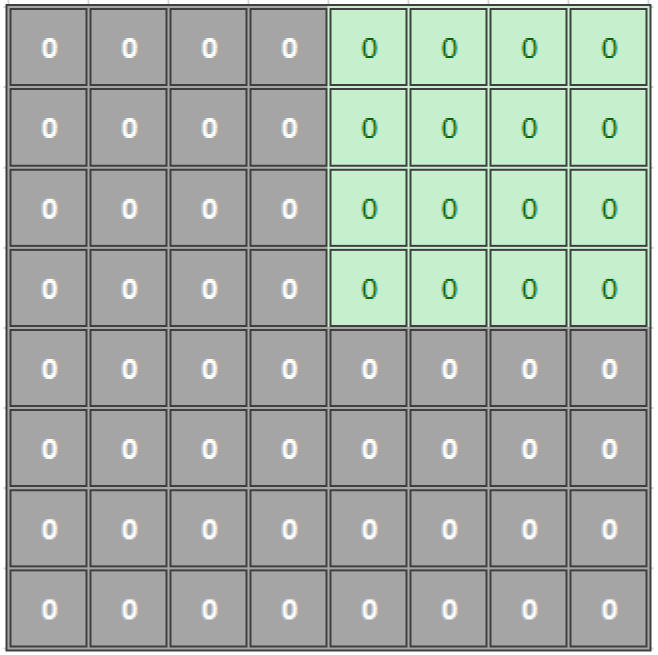
Level = 3, Address = 301

Matrix: How to insert pulse ?

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Level = 3, Address = 123

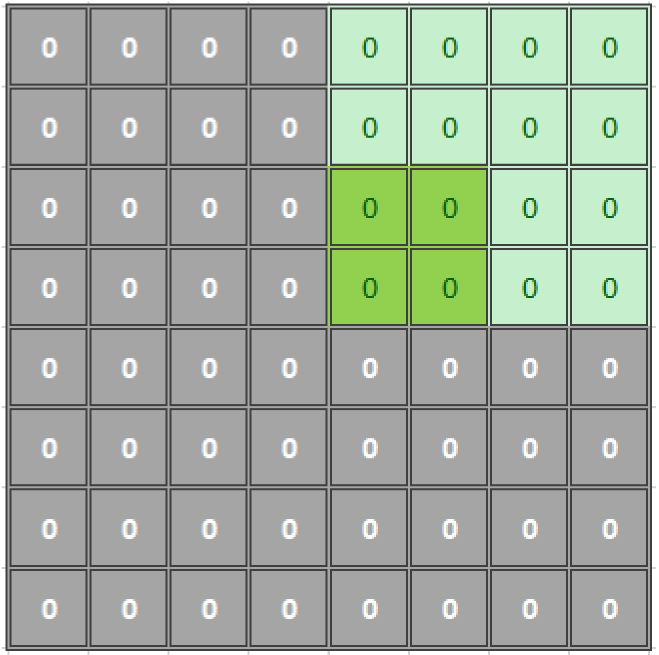
Matrix: How to insert pulse ?



0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Level = 3, Address = 123

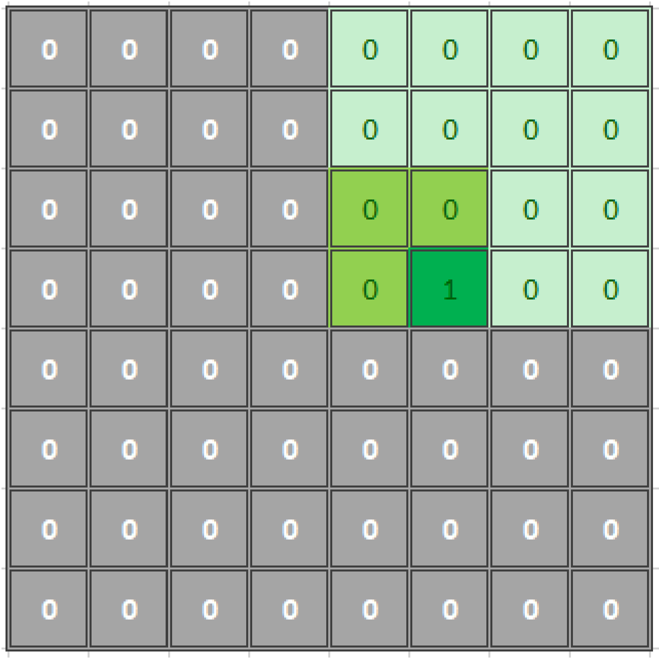
Matrix: How to insert pulse ?



0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Level = 3, Address = 123

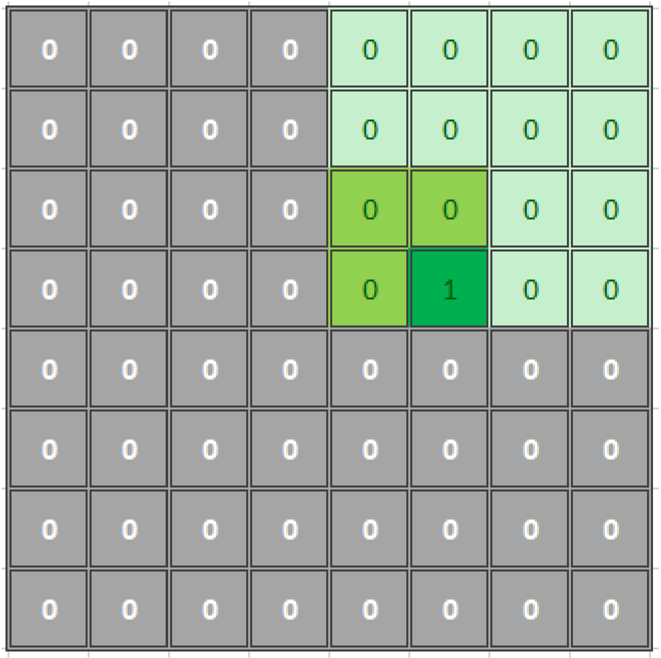
Matrix: How to insert pulse ?



0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Level = 3, Address = 123

Matrix: How to insert pulse ?



0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Level = 3, Address = 123

Matrix: features: Effective and Efficient

- Convert large wave into smaller data (image or matrix) without significant reduction in important data.

  No matter how long the pulse is we are only looking for slopes or similar properties in it and store in definite matrix which can be feed to deep learning networks.

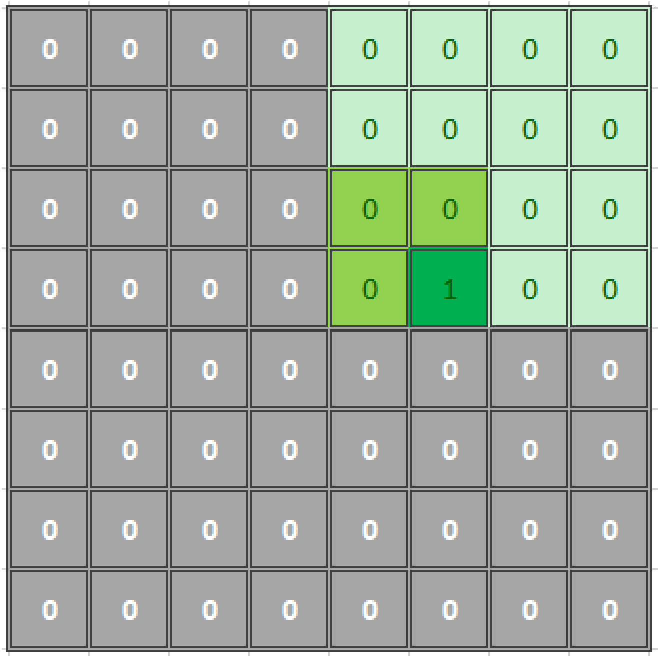
- Stores every sub-wave of length pulselen effectively and efficiently.

  Each sub wave of pulse have some unique signature or address it they differ and similar pulses have same signature. To store each pulse of a type we need only one cell space. A specific cell of matrix contains the number of pulses with that signature.

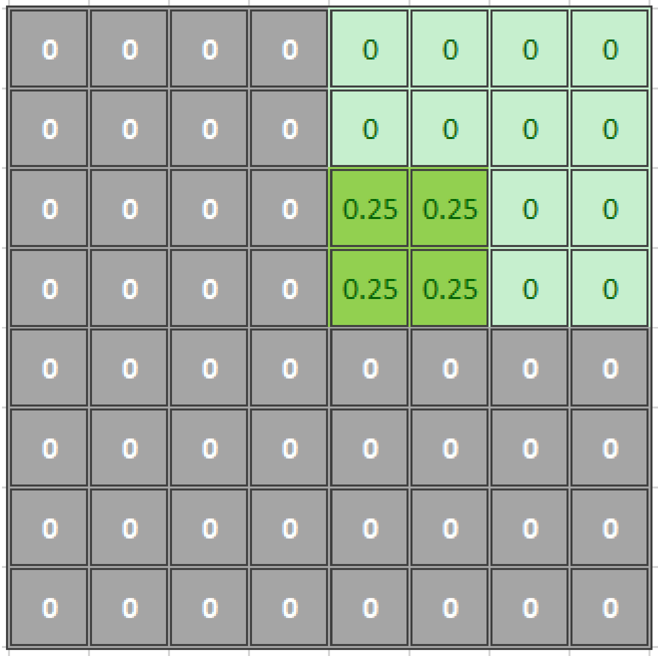
Matrix: features: Focus

- We can choose to increment one pixel or a set of pixel depending on level of detail.

  For example, while parsing pulses we decide to ignore the effect of last data then we can increase all 4 final blocks ($n\_cat = 4$) by 0.25 or more general way we can increase all n_cat final points by $1/n\_cat$



0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

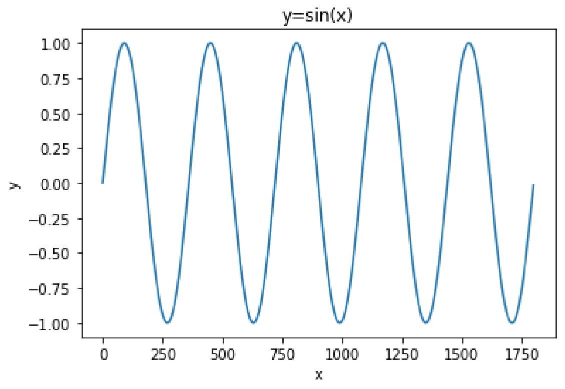


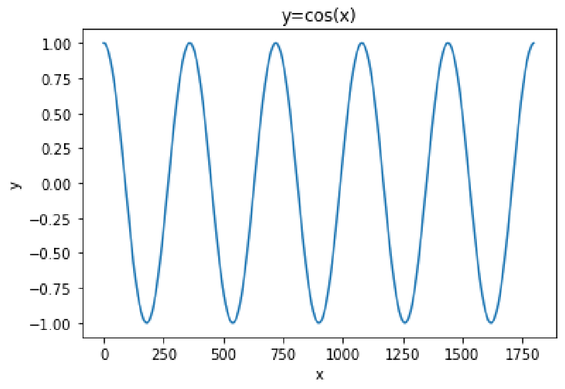
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0.25	0.25	0	0
0	0	0	0	0.25	0.25	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

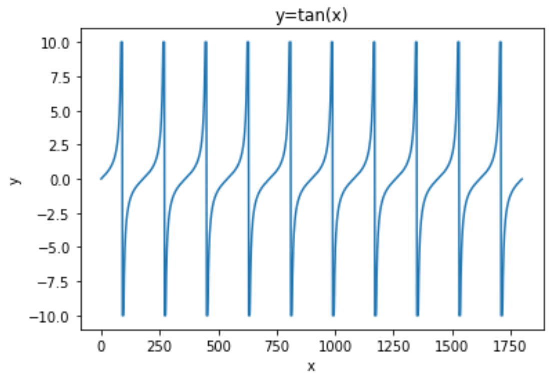
Matrix: features: Phase and Amplitude

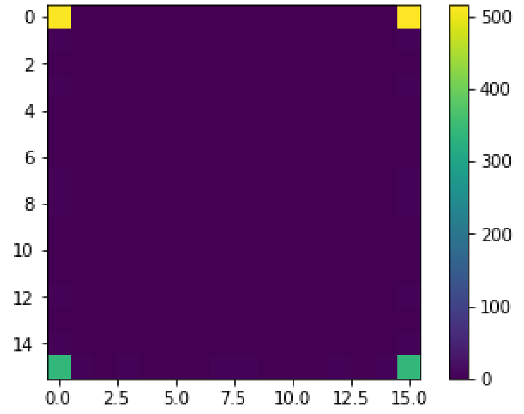
- Behaves same with shifted waves.

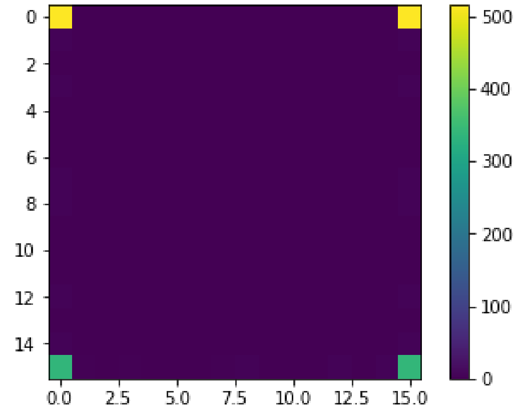
  For example, it does not differentiate between sin wave and cos wave but effectively differentiate between sin wave and tan wave.

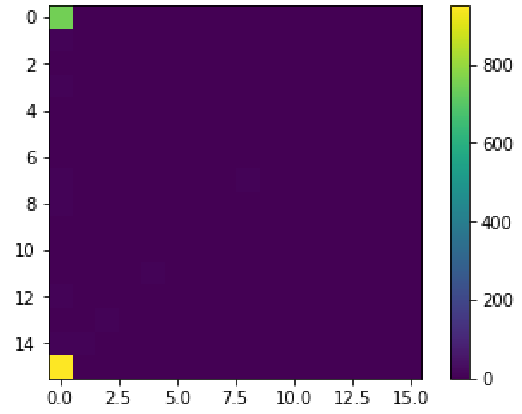






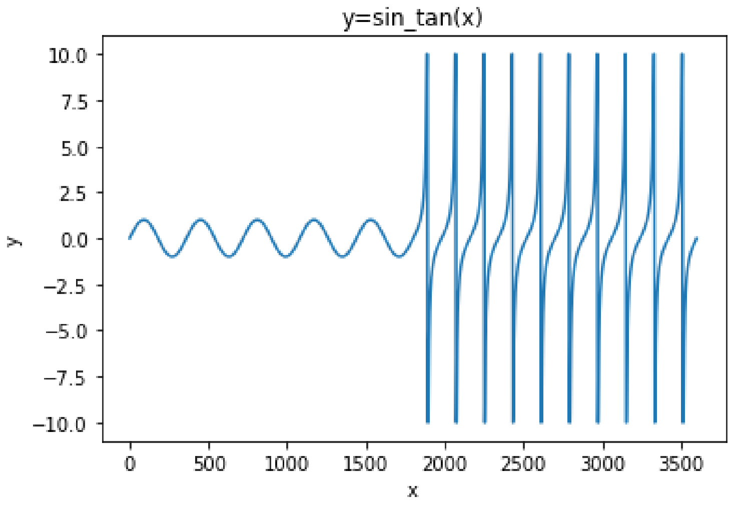


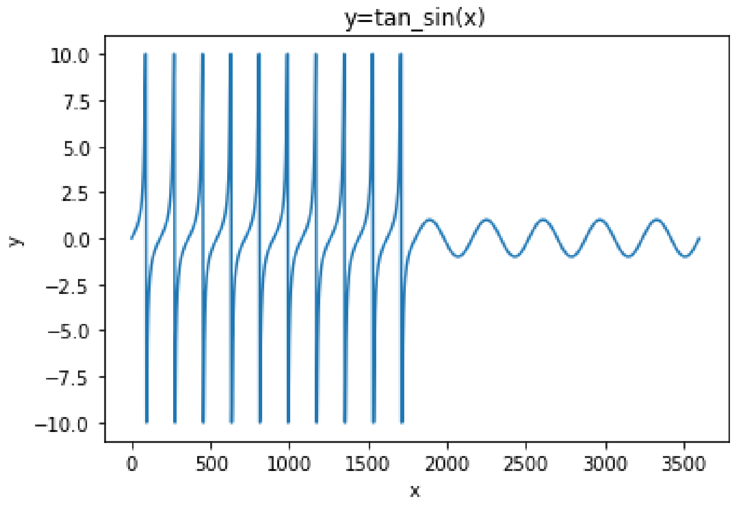


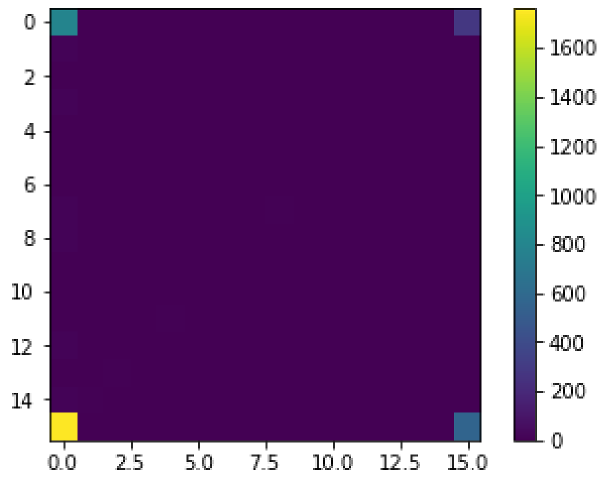


Matrix: features: Pulse Order

- Pulse order in wave does not matter. But pulse count does matter. Good for independent pulses.

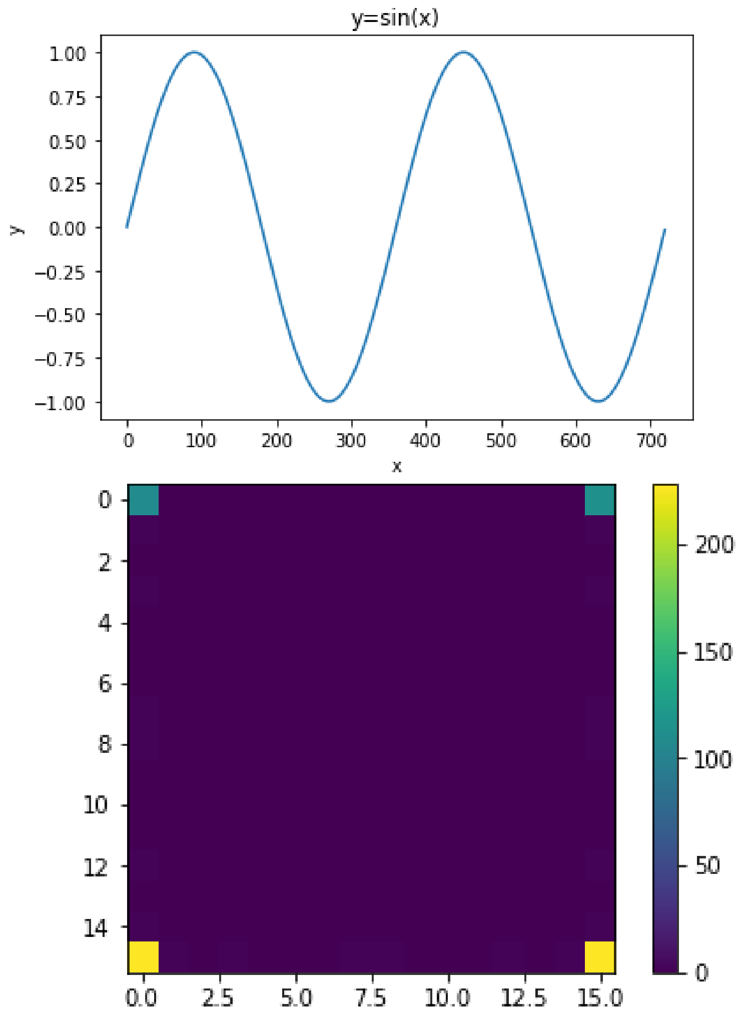


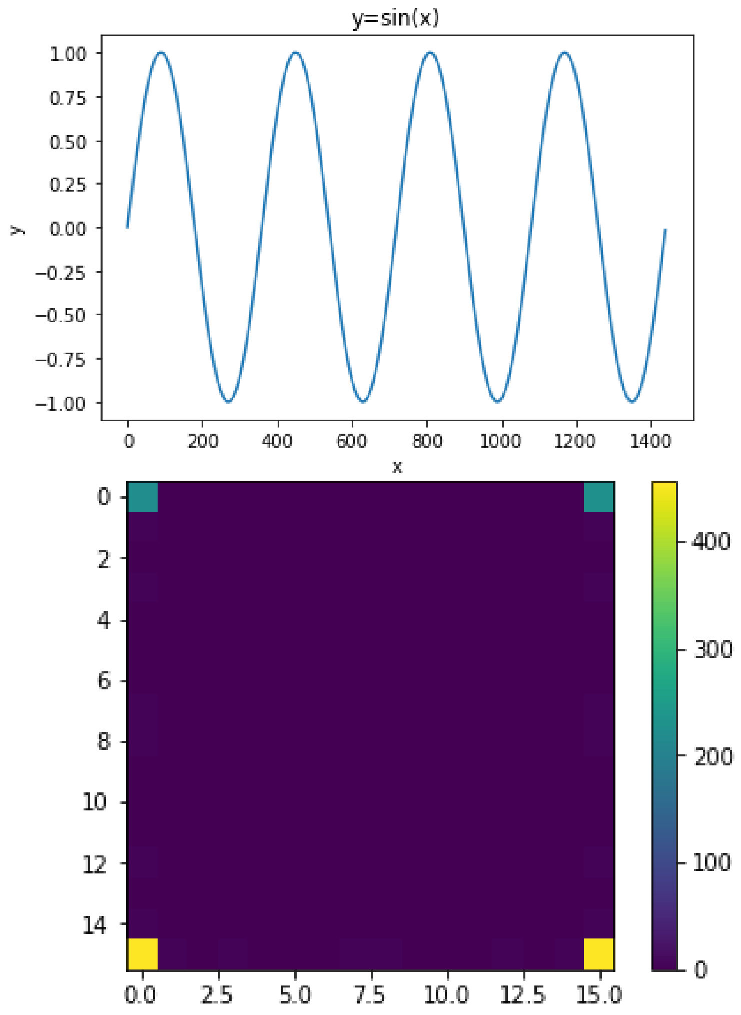




Matrix: features: Repeated wave as a Layer

- A repeated shape can be seen as a layer. When the shape repeats the layer simple doubles.





Stage 2: CNN Classification

- In this step we feed the exported images from previous step into a tensor-flow CNN image classifier and get the accuracy of model.

```
# Convolutional Layer 1.
filter_size1 = 3
num_filters1 = 32

# Convolutional Layer 2.
filter_size2 = 3
num_filters2 = 32

# Convolutional Layer 3.
filter_size3 = 3
num_filters3 = 64

# Fully-connected layer.
fc_size = 128             # Number of neurons in fully-connected layer.

# Number of color channels for the images: 1 channel for gray-scale.
num_channels = 3

# validation split
validation_size = .20
```

Stage 2: CNN Classification

- Image resolution was down-sampled using 2x2 max-pooling followed by *ReLU* operation.
- The predicted class function uses *softmax* to normalize and *argmax* for estimation.
- Using TensorFlow's in-built *cross-entropy* function.
- We use TensorFlow training library's *AdamOptimizer*
- The model was trained with $pulselen = 4,5,6$. We trained the model for multiple classification as well as binary classification (class 1 against all other).
- The testing accuracy we get for all cases after 10,000 iterations is listed below:

Class \ $pulselen$	4	5	6
Binary	95.0%	96.0%	95.8%
Multiclass	66.4%	65.0%	68.8%