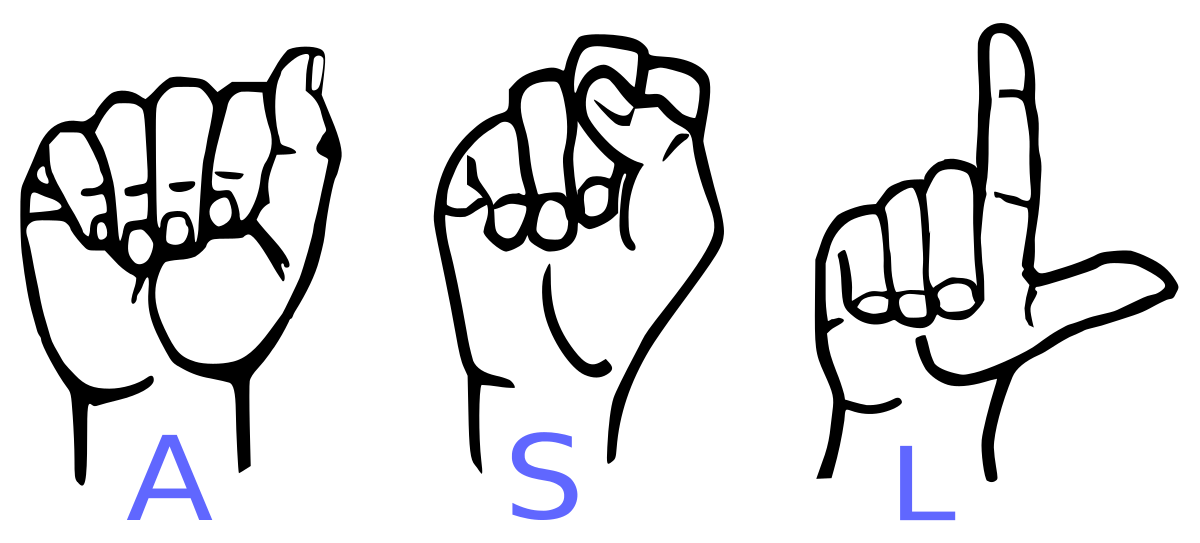
**ML Mini Project**

Sign Language Recognition



**Team Members:**

185001153 - Sivaguru R

185001179 - Surya Prakash G

185001180 - Surya S S

185001195 - Vikraman S

**Problem Statement**

**What is Sign Language?**

Here Sign Language refers to the standardised [American Sign Language.](https://www.nidcd.nih.gov/glossary/american-sign-language-asl) It is a complete, natural language that has the same linguistic properties as spoken [languages](https://www.nidcd.nih.gov/glossary/language), with grammar that differs from English. ASL is expressed by movements of the hands and face. It is the primary language of many North Americans who are deaf and hard of hearing, and is used by many hearing people as well.

**Motivation**

Hand gesture is one of the methods used in sign language for non-verbal communication. It is most commonly used by deaf & dumb people who have hearing or speech problems to communicate among themselves or with normal people. Various sign language models have been developed by many makers around the world but they are neither flexible nor cost-effective for the end users. Hence, we introduced a model that is able to automatically recognize sign language to help deaf and dumb people to communicate more effectively with each other or normal people.

**Objective**

In this project we are creating a CNN model which classifies the different signs in American Sign Language. This model takes 28x28 image as input and predicts the sign.

**Dataset**

American Sign Language (ASL) is a complete, natural language that has the same linguistic properties as spoken languages, with grammar that differs from English. ASL is expressed by movements of the hands and face. It is the primary language of many North Americans who are deaf and hard of hearing, and is used by many hearing people as well. The dataset format is patterned to match closely with the classic MNIST. Each training and test case represents a label (0-25) as a one-to-one map for each alphabetic letter A-Z (and no cases for 9=J or 25=Z because of gesture motions). The training data (27,455 cases) and test data (7172 cases) are approximately half the size of the standard MNIST but otherwise similar with a header row of label, pixel1, pixel2…. pixel784 which represent a single 28x28 pixel image with grayscale values between 0-255. The original hand gesture image data represented multiple users repeating the gesture against different backgrounds. The Sign Language MNIST data came from greatly extending the small number (1704) of the color images included as not cropped around the hand region of interest.

**Link:** <https://www.kaggle.com/datamunge/sign-language-mnist>

**Files**

* Train.csv
  + [ Label, Pixels ] – 785
  + Samples – 27455
* Test.csv
  + [ Label, Pixels ] – 785
  + Samples – 7172
* Sample.csv
  + [ Label, Pixels ] – 785
  + Samples – 50

**Evaluation Metrics**

**Matthews Correlation Coefficient**

The Matthews correlation coefficient is a more reliable statistical rate which produces a high score only if the prediction obtained good results in all of the four confusion matrix categories (true positives, false negatives, true negatives, and false positives), proportionally both to the size of positive elements and the size of negative elements in the dataset.

accuracy=(TP+TN)/(TP+TN+FP+FN)

**Classification Report**

1. **TN / True Negative:**when a case was negative and predicted negative.
2. **TP / True Positive:**when a case was positive and predicted positive.
3. **FN / False Negative:**when a case was positive but predicted negative.
4. **FP / False Positive:**when a case was negative but predicted positive.

**Precision**

Precision is the ability of a classifier not to label an instance positive that is actually negative. For each class it is defined as the ratio of true positives to the sum of true and false positives.

precision = TP/(TP + FP)

**Recall**

Recall is the ability of a classifier to find all positive instances. For each class it is defined as the ratio of true positives to the sum of true positives and false negatives.

Recall = TP/(TP+FN)

**F1-score**

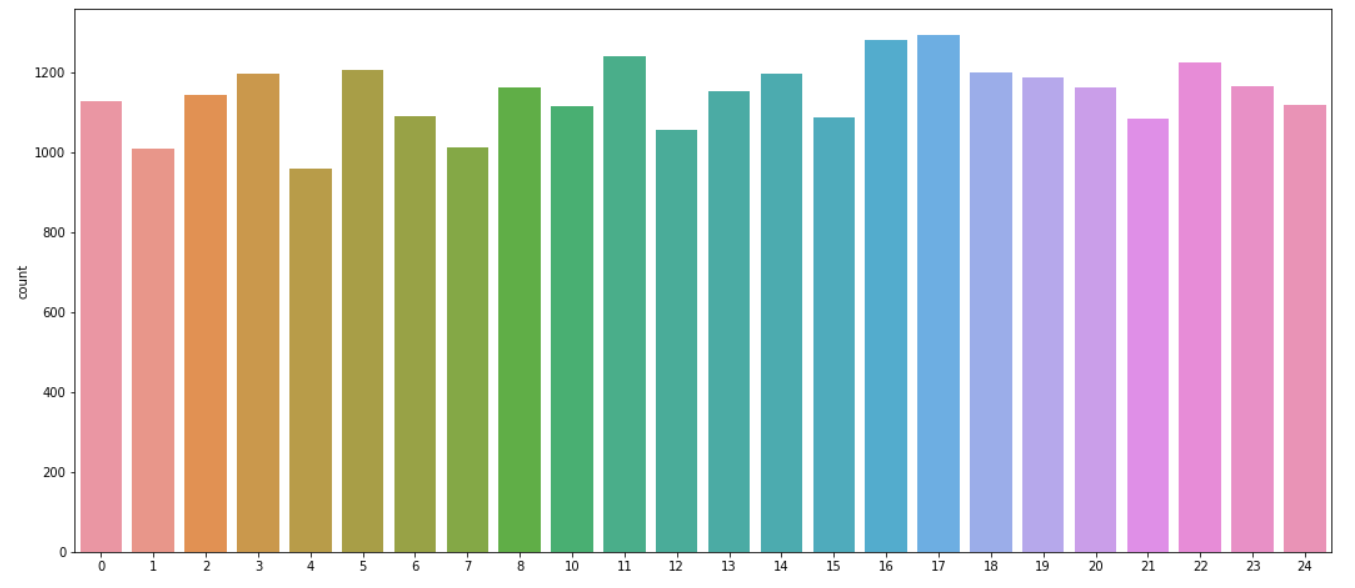
The F1 score is a weighted harmonic mean of precision and recall such that the best score is 1.0 and the worst is 0.0.

F1 Score = 2\*(Recall \* Precision) / (Recall + Precision)

**Proposed Solution**

**Data exploration**

To check whether the training dataset is equally distributed using countplot.

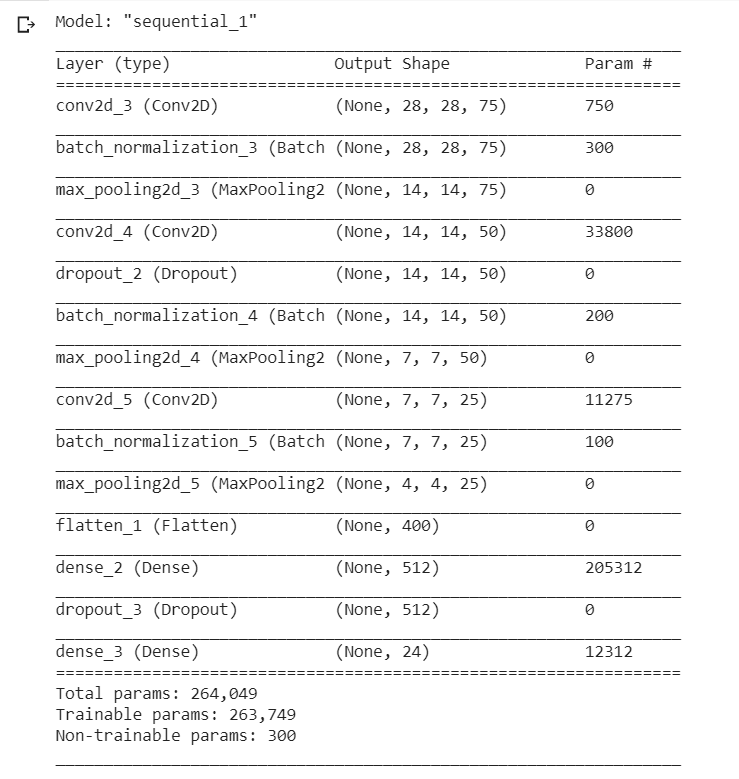
****

**Data preprocessing**

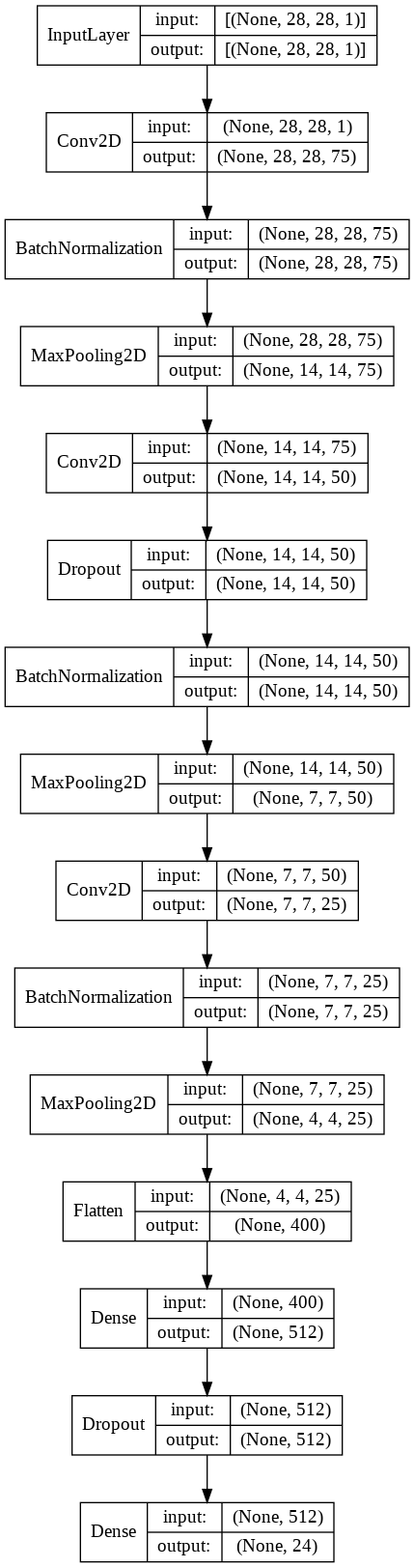
To Reshape the images and convert the labels to one-hot encoded values using label binarizer. For categorical variables where no such ordinal relationship exists, the integer encoding is not enough. One-hot encoding can be applied to the integer representation. This is where the integer encoded variable is removed and a new binary variable is added for each unique integer value.



**CNN Model**



Plot Model



**Compiling the Model**

Epoch 1/25

215/215 [==============================] - 100s 459ms/step - loss: 1.2562 - accuracy: 0.6439 - val\_loss: 3.1461 - val\_accuracy: 0.1325

Epoch 00001: val\_loss improved from inf to 3.14608, saving model to sign\_lang\_recog.h5

Epoch 2/25

215/215 [==============================] - 99s 459ms/step - loss: 0.0259 - accuracy: 0.9949 - val\_loss: 1.1506 - val\_accuracy: 0.6286

Epoch 00002: val\_loss improved from 3.14608 to 1.15065, saving model to sign\_lang\_recog.h5

Epoch 3/25

215/215 [==============================] - 98s 456ms/step - loss: 0.0082 - accuracy: 0.9989 - val\_loss: 0.3368 - val\_accuracy: 0.8873

Epoch 00003: val\_loss improved from 1.15065 to 0.33677, saving model to sign\_lang\_recog.h5

Epoch 4/25

215/215 [==============================] - 98s 456ms/step - loss: 0.0038 - accuracy: 0.9994 - val\_loss: 0.1432 - val\_accuracy: 0.9479

Epoch 00004: val\_loss improved from 0.33677 to 0.14322, saving model to sign\_lang\_recog.h5

Epoch 5/25

215/215 [==============================] - 98s 458ms/step - loss: 0.0052 - accuracy: 0.9989 - val\_loss: 0.3175 - val\_accuracy: 0.8974

Epoch 00005: val\_loss did not improve from 0.14322

Epoch 6/25

215/215 [==============================] - 97s 451ms/step - loss: 0.0096 - accuracy: 0.9973 - val\_loss: 0.5928 - val\_accuracy: 0.8404

Epoch 00006: val\_loss did not improve from 0.14322

Epoch 7/25

215/215 [==============================] - 99s 458ms/step - loss: 0.0048 - accuracy: 0.9994 - val\_loss: 0.1231 - val\_accuracy: 0.9618

Epoch 00007: val\_loss improved from 0.14322 to 0.12314, saving model to sign\_lang\_recog.h5

Epoch 8/25

215/215 [==============================] - 100s 464ms/step - loss: 7.9829e-04 - accuracy: 1.0000 - val\_loss: 0.1820 - val\_accuracy: 0.9497

Epoch 00008: val\_loss did not improve from 0.12314

Epoch 9/25

215/215 [==============================] - 100s 463ms/step - loss: 0.0015 - accuracy: 0.9997 - val\_loss: 0.3582 - val\_accuracy: 0.8993

Epoch 00009: val\_loss did not improve from 0.12314

Epoch 10/25

215/215 [==============================] - 100s 464ms/step - loss: 0.0022 - accuracy: 0.9997 - val\_loss: 0.4224 - val\_accuracy: 0.9010

Epoch 00010: val\_loss did not improve from 0.12314

Epoch 11/25

215/215 [==============================] - 99s 462ms/step - loss: 0.0239 - accuracy: 0.9921 - val\_loss: 0.3006 - val\_accuracy: 0.9176

Epoch 00011: val\_loss did not improve from 0.12314

Epoch 12/25

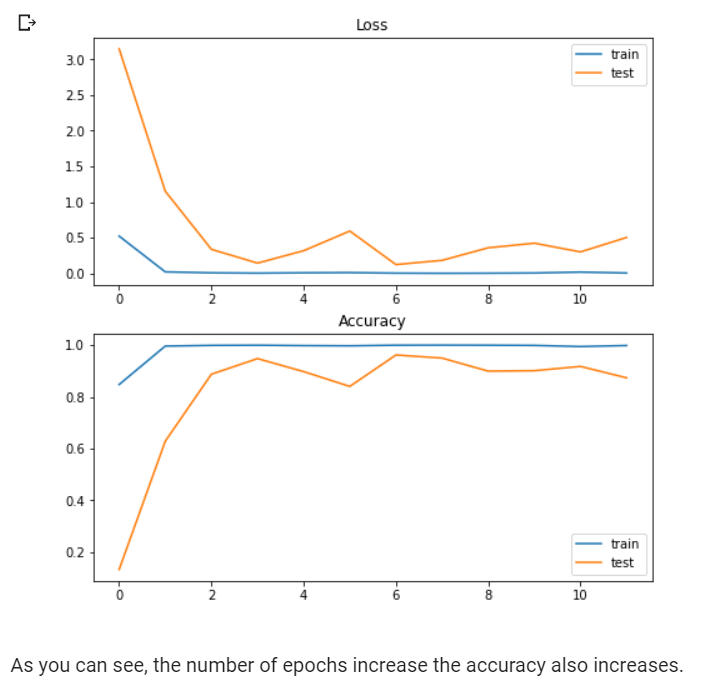
215/215 [==============================] - 99s 460ms/step - loss: 0.0034 - accuracy: 0.9988 - val\_loss: 0.5026 - val\_accuracy: 0.8734

Restoring model weights from the end of the best epoch.

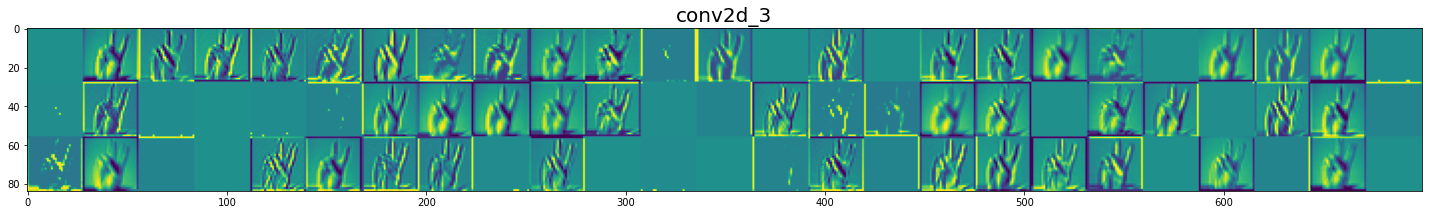
Epoch 00012: val\_loss did not improve from 0.12314

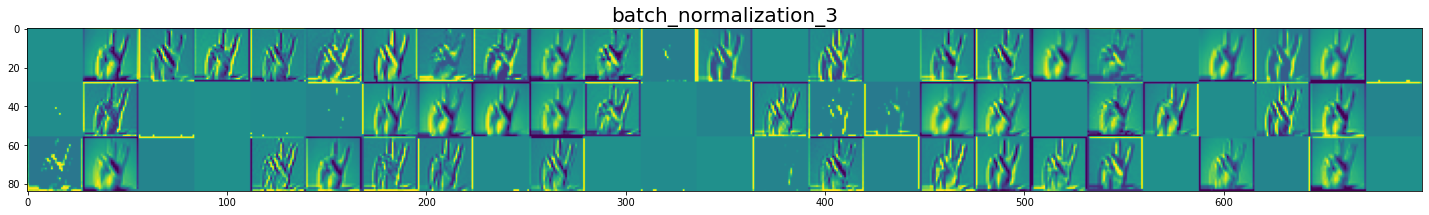
Epoch 00012: early stopping

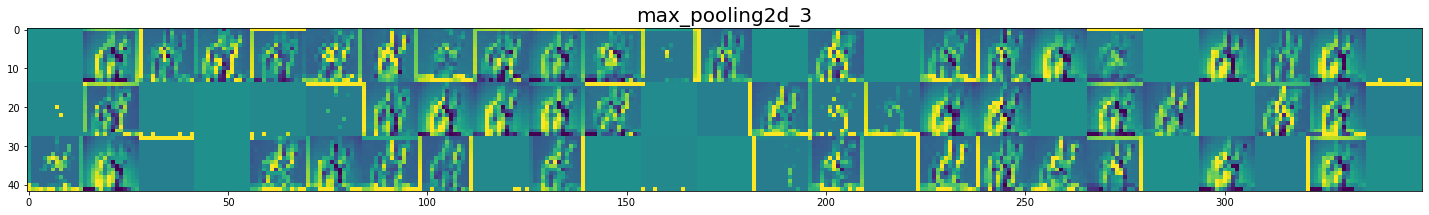
**Plotting Accuracy and Loss**

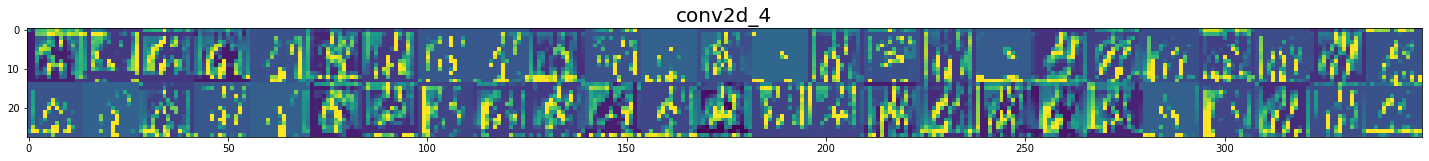


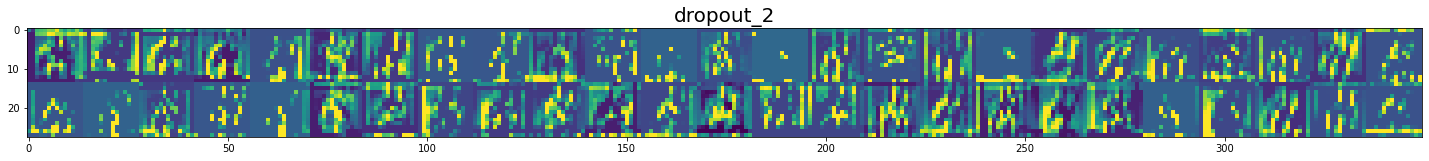
**Visualizing intermediate layer outputs**

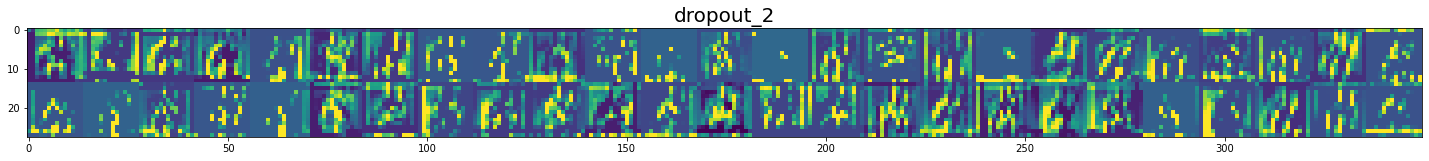


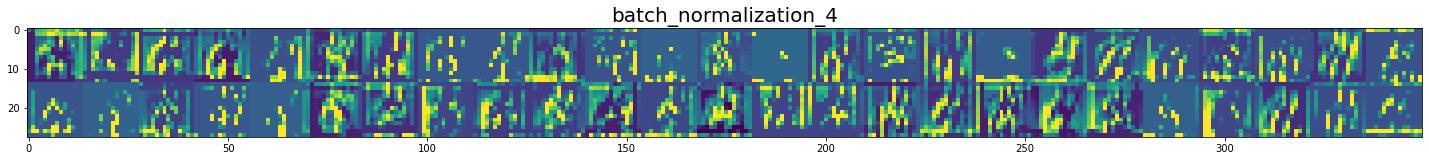


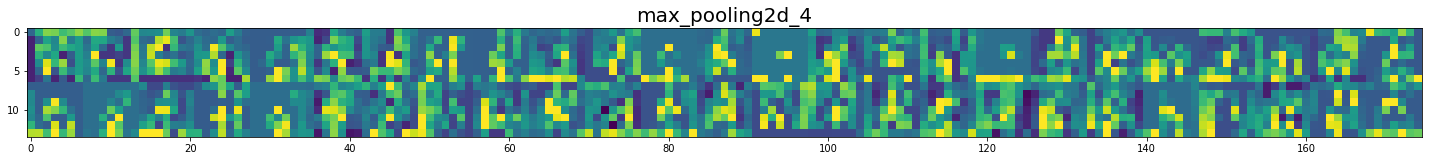


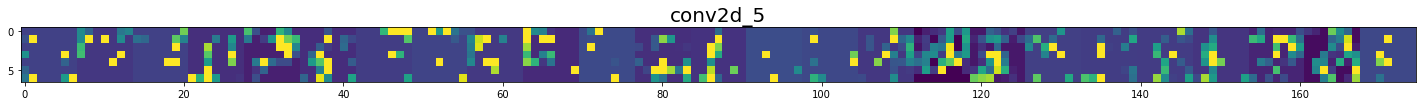


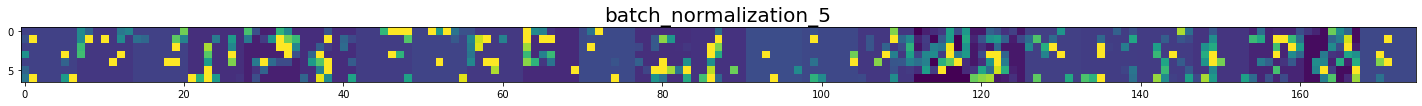








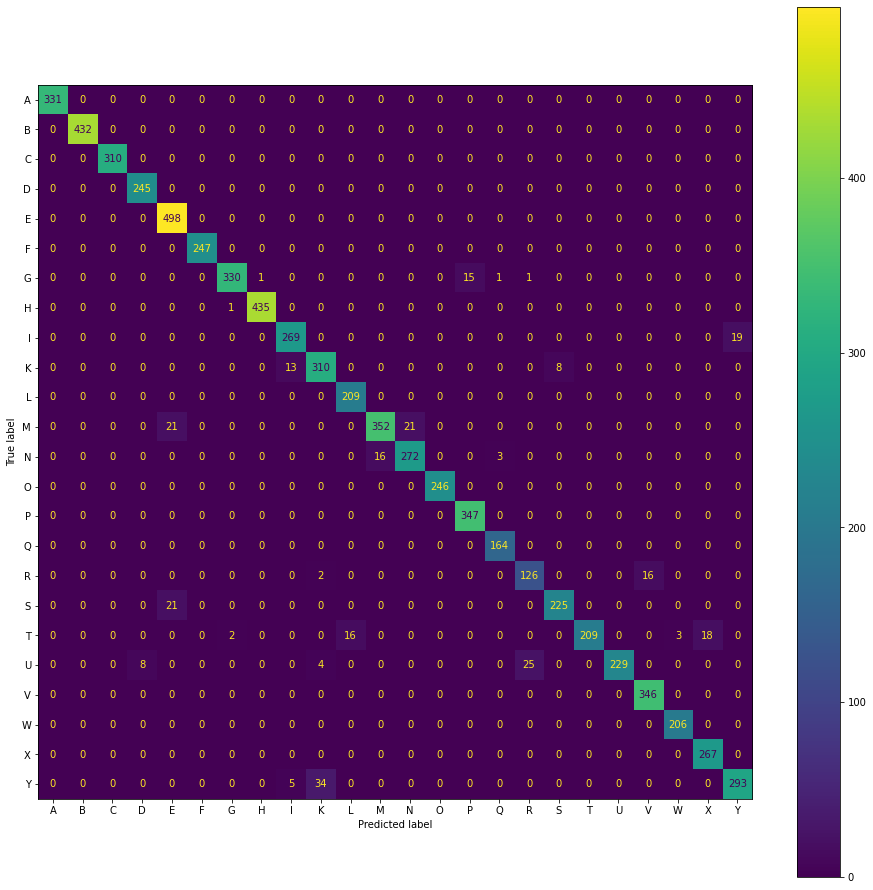




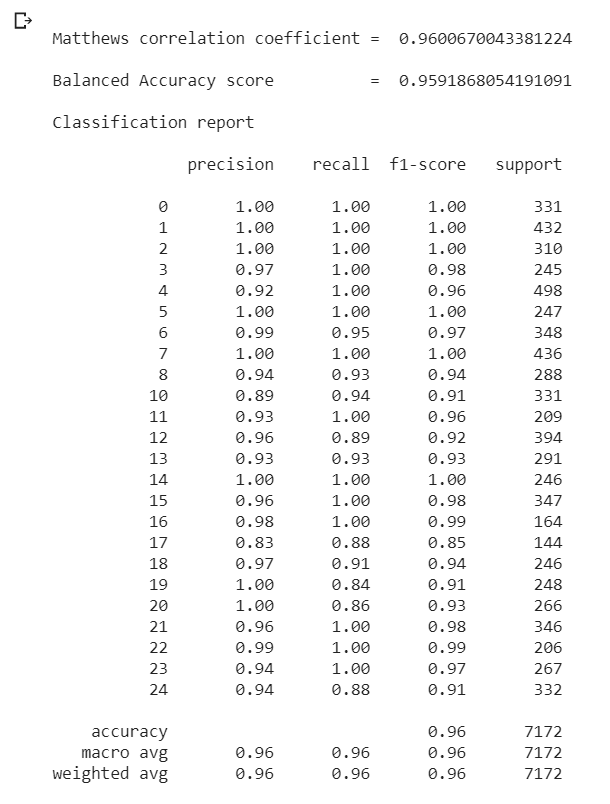


**Metrics to evaluate Model**

**Confusion matrix**



**Classification report**



**Prediction**

