Machine Learning Engineer Nanodegree

American Sign Language Alphabet

Hassan M. Abogabal

September 24th, 2018

I. Definition

1.1 Project Overview

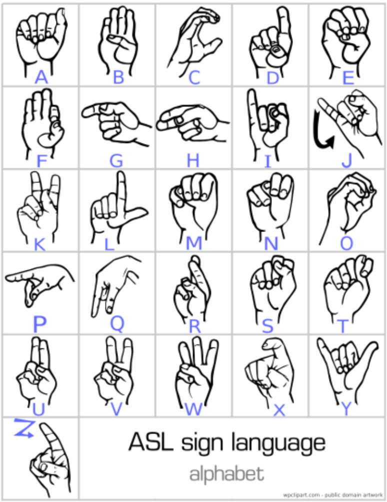
With the recent huge advances in technology, the importance of the visual recognition and image processing has significantly increased. Deep learning and computer images help us to understand images and extract information from them.

Hearing loss is a big problem that affect all people across the globe, so that the sing language was created to help the deaf communicate with other people and it’s difficult to other people to understand their sings’ .in this project, I created a machine learning project that take image of the hand movement and predict the equal alphabet represented in the image of the hand by the sing language using CNN and SVM.

1.2 Problem Statement

The goal is to create an algorithms that can use in app to help people better communicate with the deaf through predicting and interpreter the hand movement image to its equivalent alphabet.

The next image show the hand movement and their equivalent alphabet.

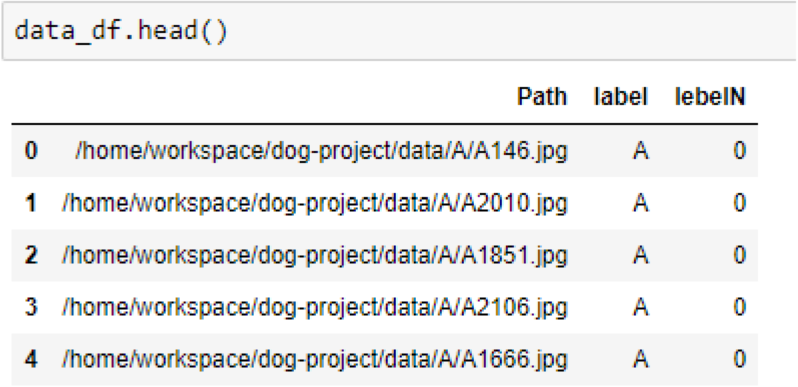


1.2.1 Datasets and Inputs

I used dataset which download from kaggle and it available here. The data set is a collection of images of alphabets from the American Sign Language, separated in 29 folders which represent the various classes each folder contains 3000 images .in this project I used only 1000 images from each class.

There are 29 classes, of which 26 are for the letters A-Z and 3 classes for *SPACE*, *DELETE* and *NOTHING*. These 3 classes are very helpful in real time applications, and classification

Because reading images as images paths form the folders sequentially, I got image paths which labels are sequentially next image describe dataset status



```
data_df.head()
```

	Path	label	lebelN
0	/home/workspace/dog-project/data/A/A146.jpg	A	0
1	/home/workspace/dog-project/data/A/A2010.jpg	A	0
2	/home/workspace/dog-project/data/A/A1851.jpg	A	0
3	/home/workspace/dog-project/data/A/A2106.jpg	A	0
4	/home/workspace/dog-project/data/A/A1666.jpg	A	0

So that I shuffled the dataset before splitting it

Splitting the dataset to 3 category:-

- Training data contains 20000 image paths and their labels
- Testing data contains 5000 image paths and their labels
- Validation data contains 4000 image paths and their labels

1.2.2 Solution Statement

It's clear from dataset description and the problem statement we have labeled data with 29 classes (English alphabet letters and Del, space, nothing).so we intend to build sing alphabet interpreter system using Classification algorithms to recognize alphabets from image.

1.3 Metrics

According to dataset description test set is randomized and shuffled so I'll use accuracy score on test set to evaluate the classifier.

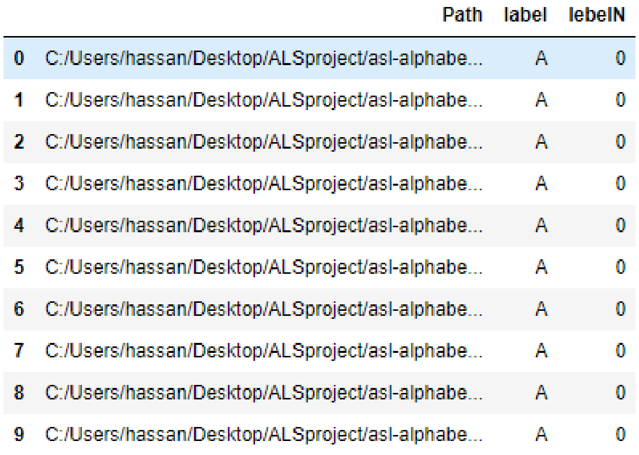
Accuracy = $(true\ positives + true\ negatives)\ /dataset\ size$

For MLP and CNN we will use the same function provided in Keras model API evaluate which Returns the loss value & metrics (accuracy) values for the model in test mode

II. Analysis

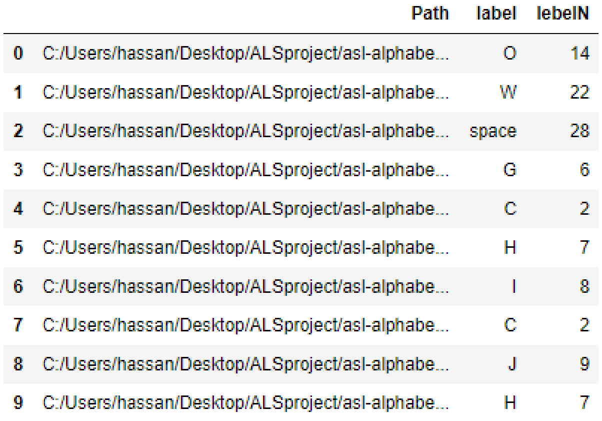
2.1 Data Exploration

In this project I loaded subset of the dataset which mentioned before in data frame using pandas lib and it contain images paths and their labels (A to Z and other 3 labels)
Then doing shuffled to data by using sklearn lib and finally splitting it to 3 category



	Path	label	lebelN
0	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	A	0
1	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	A	0
2	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	A	0
3	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	A	0
4	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	A	0
5	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	A	0
6	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	A	0
7	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	A	0
8	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	A	0
9	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	A	0

Figure 1(unshuffled dataset)

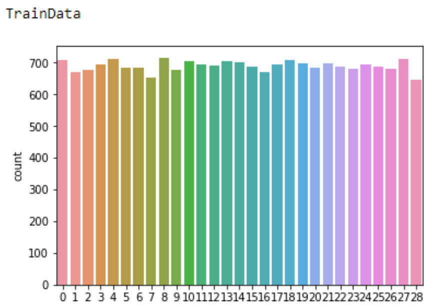


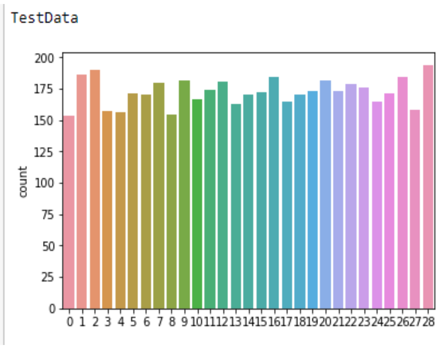
	Path	label	lebelN
0	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	O	14
1	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	W	22
2	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	space	28
3	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	G	6
4	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	C	2
5	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	H	7
6	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	I	8
7	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	C	2
8	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	J	9
9	C:/Users/hassan/Desktop/ALSproject/asl-alphabe...	H	7

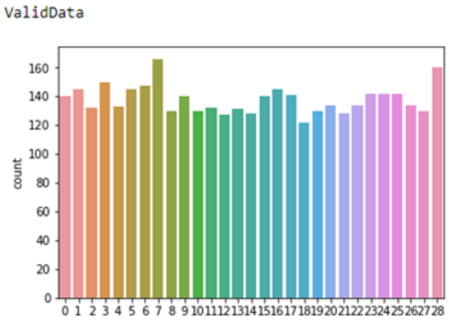
Figure 2(shuffled dataset)

2.2 Exploratory Visualization

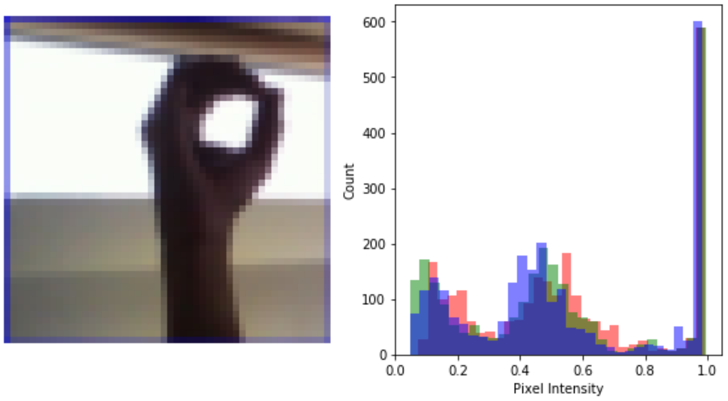
Show number of images in each class for train, test and valid data







Show the image intensity Histogram



2.3 Algorithms and Techniques

In this part we will discuss Deep Learning techniques which are very promised in computer vision field:

- Deep Neural networks consist of input layer and multiple nonlinear hidden layers and output layer, so the number of connections and trainable parameters are very large.
- The deep neural network needs very large set of examples to prevent over fitting
- One class type of Deep Neural Network with comparatively smaller set of parameters and easier to train is Convolution Neural Network (CNN).

- CNN is a multi-layer feed-forward neural network that extract features and properties from the input data (images or sounds). CNN trained with neural network back-propagation algorithm
- CNN have the ability to learn from high-dimensional complex inputs, nonlinear mappings from very large number of data (images or sounds).
- The advantage of CNN is that it automatically extracts the salient features which are invariant and a certain degree to shift and shape distortions of the input characters. Another major advantage of CNN is the use of shared weight in convolution layers, which means that the same filter is used for each input in the layer. The share weight reduces parameter number and improves performance.
- Pooling Layer It is common to periodically insert a Pooling layer in-between successive Convolution layers in a CNN architecture. Its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network, and hence to also control overfitting. The Pooling Layer operates independently on every depth slice of the input and resizes it spatially, using the MAX operation. The most common form is a pooling layer with filters of size 2x2 applied with a stride of 2 down samples every depth slice in the input by 2 along both width and height, discarding 75% of the activations. Every MAX operation would in this case be taking a max over 4 numbers (little 2x2 region in some depth slice). The depth dimension remains unchanged.

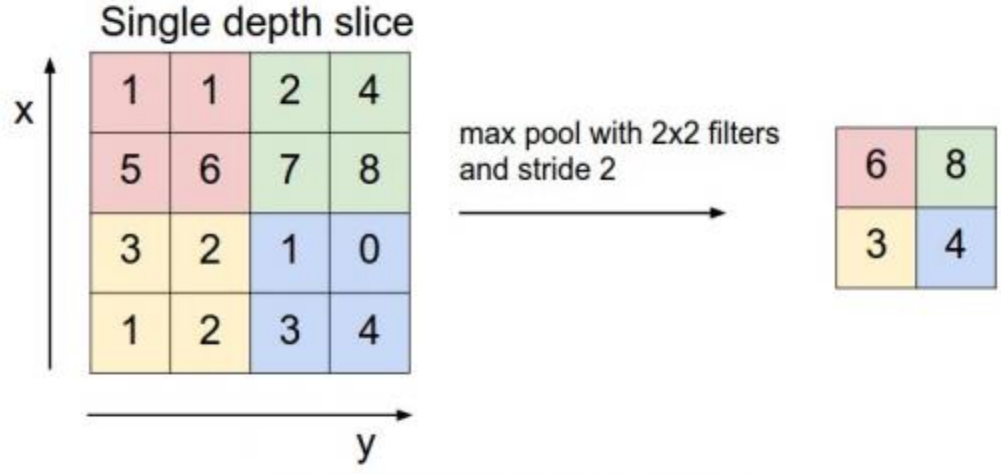


Image show Max Pooling operation

- Deep Neural Networks will be implemented using Keras The Python Deep Learning library.

In this part we discuss the strengths and weaknesses for SVM

- SVM constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for classification, regression, or other tasks like outliers detection. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier.
- The intuition of SVM:
  tries to maximize the margin between the hyperplane and the samples. Thanks to that, the decisions it makes, are more accurate. Large distance between the decision boundary and the samples, makes the probability of miss classifying lower. This kind of classifiers are called large margin classifiers.
- Strengths of SVM: can model non-linear decision boundaries with wide margin, and there are many kernels to choose from. They are also fairly robust against overfitting, especially in high-dimensional space.
- Weaknesses: memory intensive, trickier to tune due to the importance of picking the right kernel, and don't scale well to larger datasets. Take more time to train and predict in large dataset unlike CNN
- The algorithm provided in sklearn.svm .SVC

2.4 Benchmark

- As mentioned in Metrics I used accuracy score on test set to evaluate the algorithms.
- The dataset is based on kernels in kaggle dashboard which achieved approx. 92% ,94 accuracy
- So we well choose the lowest accuracy mentioned in kaggle 92% as a benchmark

III. Methodology

3.1 Data Preprocessing

As mentioned in data Exploration the data is consists of images and each image is 200x200 pixels.

- Then need to convert this image to 4D array by basing it to path_imgto_tensors which take astring value (image path) as input and return 4D tensor suitable for supplying to a keras CNN.the function load the image and resizes it to 50*50 pixels. Then converted the image to an array
- Rescaling the array by diving every list element by 255 to change the range between 0-1
- Encode categorical integer labels using a One-Hot Encoding to work with output layer of Neural Networks.
- Doing reshape 4D array to 2D array to using SVM .

3.2 Implementation

I build tree deep neural networks MLP ,CNN and augmented image CNN

- MLP Architecture:

1- Input layer followed with flatten layer to set image pixels as a row

2- Three hidden layers the first one with 256 node, the second with 128 node and the third with 64 node.

3- Every layer of the hidden layers with Relu activation function and followed by drop out layer to prevent overfitting

4- Output layer with softmax activation function with 29 output class.

- CNN Architecture:

1- i used 3 CNN followed by the 1 fully connected hidden layer and the output layer.

2- the first 3 convolution layers are working as main layers 1st layer to detect simple features like edges, 2nd layer to detect more complex features like arcs, corners, squares, … etc. and 3rd layer to get more complex features.

3- using the pooling layer to reduce teh input for the next layer

4- After the last convolution layer and it's pooling there is a dropout to prevent overfitting which may cause by the complexity of the previous layer.

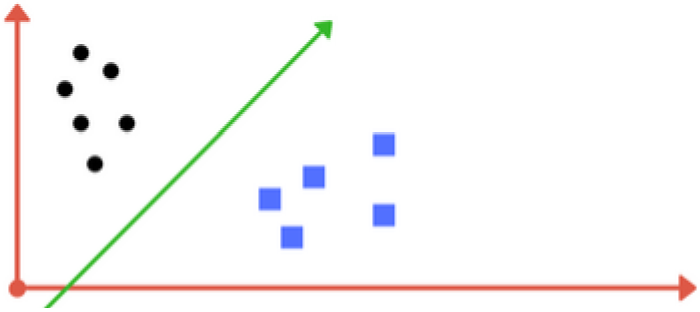
5- There is a flatten layer as preprocessing to fully connected layers

6- After that there is FC layer to enhance classification with Relu activation and followed by dropout to prevent overfitting.

7- The last layer is The output layer with softmax activation

- Augmented CNN
  Same as CNN architecture and changing the number of filters
- Support Vector Machine(SVM)

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. In two dimensional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side.' The data is separated by a hyperplane with dimensions = number of dimensions – 1



So, separating the data by a hyperplane is by trying to maximize the margin. I choose the kernel 'RBF' as it's a general purpose kernel.

3.3 Refinement

1- for Deep learning I used many trains of architectures to get a good result and CNN model got accuracy more than the benchmark

2-changing dropout ratio from 0.1 to (0.2 ,0.3) improved the accuracy

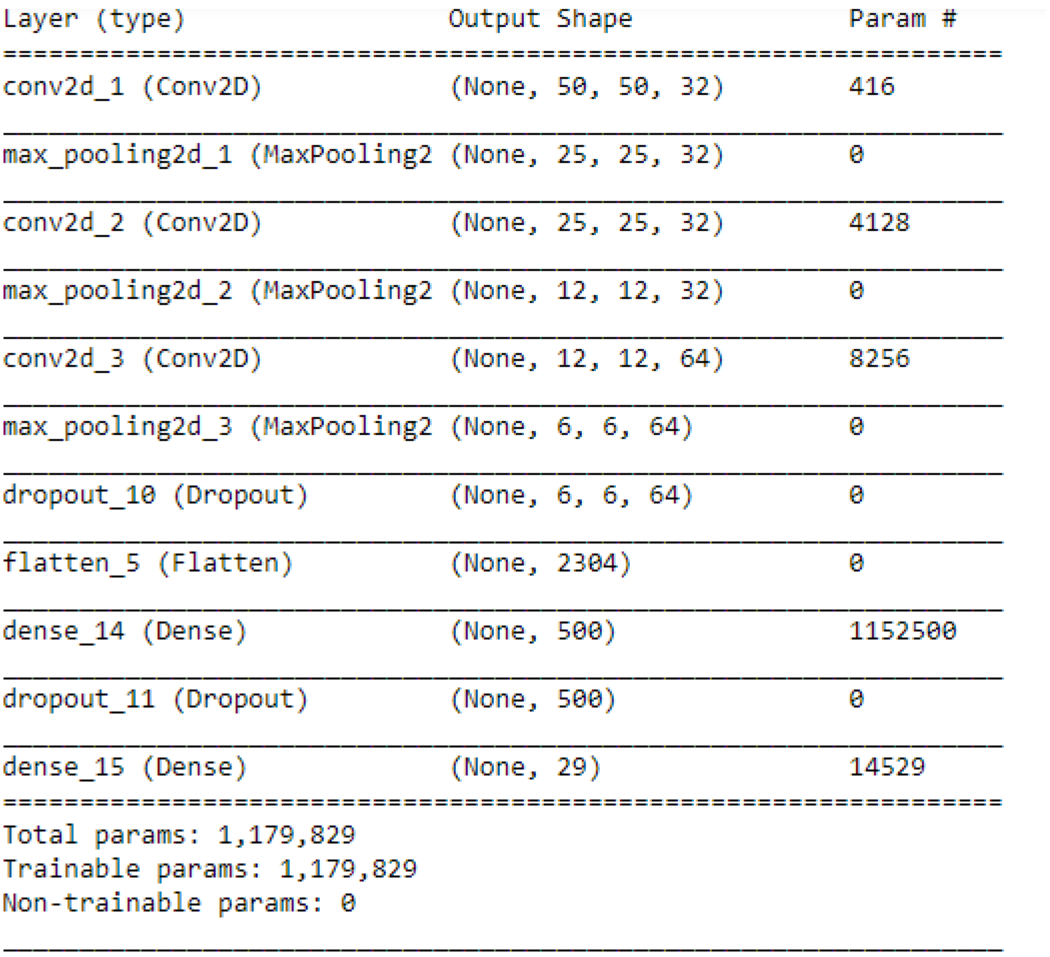
IV. Results

Model Evaluation and Validation

During development, a validation set was used to evaluate the models in tree deep learning algorithms ,and I used CNN architecture and aug_CNN architecture as final models

CNN model architecture:-

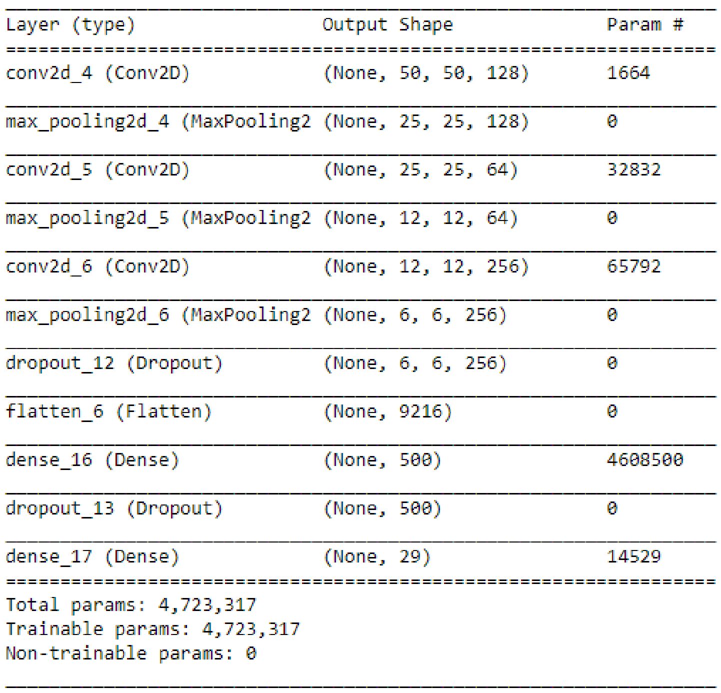
Test accuracy is 96.40%



```
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_1 (Conv2D)            (None, 50, 50, 32)        416       
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 25, 25, 32)        0         
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 25, 25, 32)        4128      
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 12, 12, 32)        0         
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 12, 12, 64)        8256      
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 6, 6, 64)          0         
_________________________________________________________________
dropout_10 (Dropout)         (None, 6, 6, 64)          0         
_________________________________________________________________
flatten_5 (Flatten)          (None, 2304)              0         
_________________________________________________________________
dense_14 (Dense)             (None, 500)               1152500   
_________________________________________________________________
dropout_11 (Dropout)         (None, 500)               0         
_________________________________________________________________
dense_15 (Dense)             (None, 29)                14529     
=================================================================
Total params: 1,179,829
Trainable params: 1,179,829
Non-trainable params: 0
_________________________________________________________________
```

Aug_CNN model architecture:

Test accuracy 91.04%



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_4 (Conv2D)            (None, 50, 50, 128)       1664      
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 25, 25, 128)       0         
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 25, 25, 64)        32832     
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 12, 12, 64)        0         
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 12, 12, 256)       65792     
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 6, 6, 256)         0         
_________________________________________________________________
dropout_12 (Dropout)         (None, 6, 6, 256)         0         
_________________________________________________________________
flatten_6 (Flatten)          (None, 9216)              0         
_________________________________________________________________
dense_16 (Dense)             (None, 500)               4608500   
_________________________________________________________________
dropout_13 (Dropout)         (None, 500)               0         
_________________________________________________________________
dense_17 (Dense)             (None, 29)                14529     
=================================================================
Total params: 4,723,317
Trainable params: 4,723,317
Non-trainable params: 0
_________________________________________________________________
```

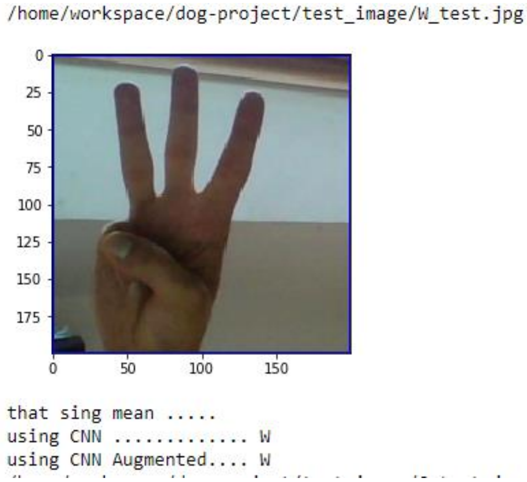
To verify the robustness of the final model, a test was conducted using images of deferent labels

The following observations are based on the result of the test:

1. The classifier can reliably detect one hand per image.
2. Images have to be same background.
3. The classifier can't detect more than one hand per image.
   - Justification
     My final test accuracy based on (CNN model) is 96.4 it's better of my benchmark model accuracy.
     It can be seen that that the application is useful for interpreter alphabet.
     In summary, the application is useful in a limited domain, but to the bigger problem, additional data with good resolution and different image background will have to be used.

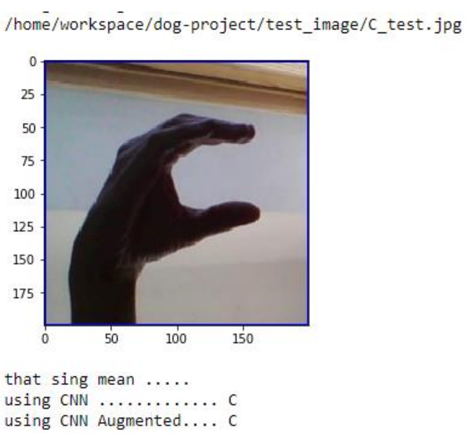
V. Conclusion

5.1 free-form Visualization



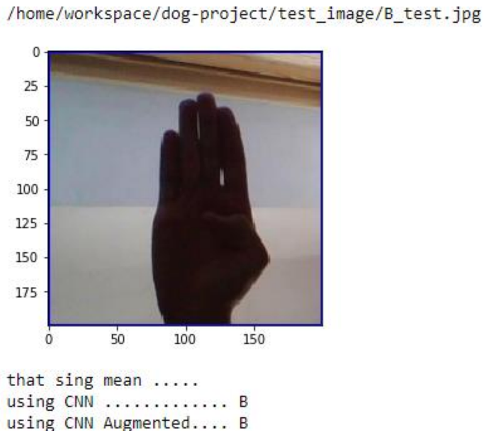
```
/home/workspace/dog-project/test_image/W_test.jpg
```

```
that sing mean .....
using CNN ............. W
using CNN Augmented.... W
```



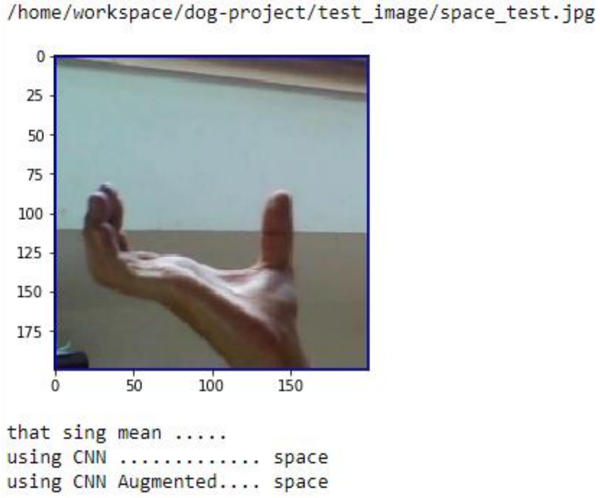
```
/home/workspace/dog-project/test_image/C_test.jpg
```

```
that sing mean .....
using CNN ............. C
using CNN Augmented.... C
```



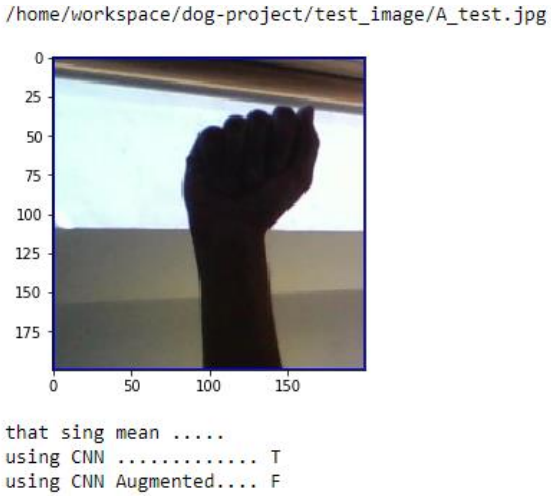
```
/home/workspace/dog-project/test_image/B_test.jpg
```

```
that sing mean .....
using CNN ............. B
using CNN Augmented.... B
```



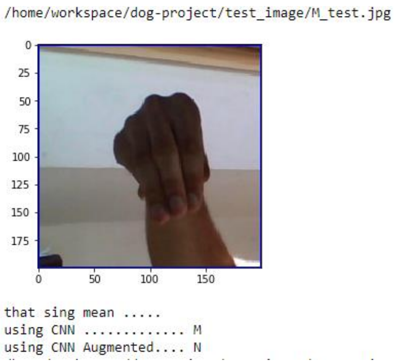
```
/home/workspace/dog-project/test_image/space_test.jpg
```

```
that sing mean .....
using CNN ............. space
using CNN Augmented.... space
```



```
/home/workspace/dog-project/test_image/A_test.jpg
```

```
that sing mean .....
using CNN ............. T
using CNN Augmented.... F
```



```
/home/workspace/dog-project/test_image/M_test.jpg
```

```
that sing mean .....
using CNN ............. M
using CNN Augmented.... N
```

In image above there are 2 failure cases can be clearly indentified :

Image is too low-contrrast , compared to it's backgroud and bluring in it

5.2 Reflection

Through working on this project we got some of important point to focus:-

1. Data Exploration and Data Visualiztion are very important steps to start with
2. Preproseeing data is an important step to imporve the model
3. The Deep learing are very promised in computer vision and image recognition
4. Know how to deals with image as matrix
5. SVM is good and cheap to applay in image classification but it take more time to train and predict in large data

5.3 Improvement

1. the CNN odel can be improved using the of GPUs large RAM can faclite to deal with large numer of image

2- using transfer learning and augmetation helping to improve the result

3-good dataset with high resolution will be improve accuracy of the predict and make small number of failure in the programe