INDIAN INSTITUTE OF TECHNOLOGY KANPUR

Post Graduate Project(Mth-698A/699A)

Final Report : Sign Language Recognition

Submitted By : Manjeet Yadav | 17816378

Supervisor : Dr. B.V. Rathish Kumar

Contents

ACKNOWLEDGEMENT

I would like to express my sincere thanks to MTH department IITK for offering me this opportunity. I am deeply indebted to Prof. B.V. Rathish Kumar for their constant guidance and supervision as well as for providing the necessary information regarding the project and also for their support in completing the project.

Project Demo Video Link : Click here

ABSTRACT

A practical sign language translator is an essential way for communication between the deaf community and the other public. So here I present the implementation and development of an American Sign Language (ASL) finger spelling translator based on a convolutional neural network (CNN). I produced a robust CNN model which classifies letters A-Z and three other classes “del”, “space” and “nothing” i.e. total 29 classes correctly in real time video captured using webcam. The limitations of the dataset and the encouraging results achieved, I am confident that with further research and more data, I can produce a generalized translator for all ASL words as well. Objective of the project can be summarized in following points.

1. To Apply data augmentation technique for generating more data on available ASL dataset from Kaggle.

2. To apply appropriate image pre-processing techniques in order to remove the noise and obtain the region of Interest (ROI).

3. To design the model and architecture for CNN to train the pre-processed images and achieve the maximum possible accuracy.

4. To develop a Tensorflow model to predict and display letter signed in real time video captured using webcam.

BACKGROUND

Sign languages are developed primarily to aid deaf and dumb people. They use a concurrent and specific combination of hand movements, hand shapes and orientation in order to convey particular information. One such set of language is the American Sign Language (ASL) system which is predominantly used in western countries. Certain aspect that distinguishes ASL from other sign languages is that ASL devoid of any temporal inflections in its finger spelling chart and also the usage of single hand only.

With the advent of artificially intelligent algorithms coupled with the availability of big data and large computational resources has led to a huge growth in the field of healthcare, robotics, autonomous self-driving vehicles, Human Computer Interaction (HCI) etc.

HCI finds its applications in augmented reality systems, facial recognition systems and also hand-gesture recognition system. This project falls under the domain of HCI and aims towards recognizing various alphabets (A-Z) of the ASL family. Hand-gesture recognition is a challenging task, particularly the ASL recognition is complicated due to its usage of both the hands. In the past many works have been performed in this respect using sensors (like glove sensor) and other image processing techniques (like edge detection technique, Hough Transform etc.) but were unable to achieve satisfactory results. However, with the new deep learning techniques like CNN, the performance in this field has grown significantly leading to many new future possibilities.

Many people are speech and/or hearing impaired, and they thus use hand gestures to communicate with other people. However, apart from a handful number of people, not everyone is aware of this sign language and they may require an interpreter which can be inconvenient and expensive. This project aims to narrow this communication gap by developing software which can predict the ASL alphanumeric hand gestures in real time.

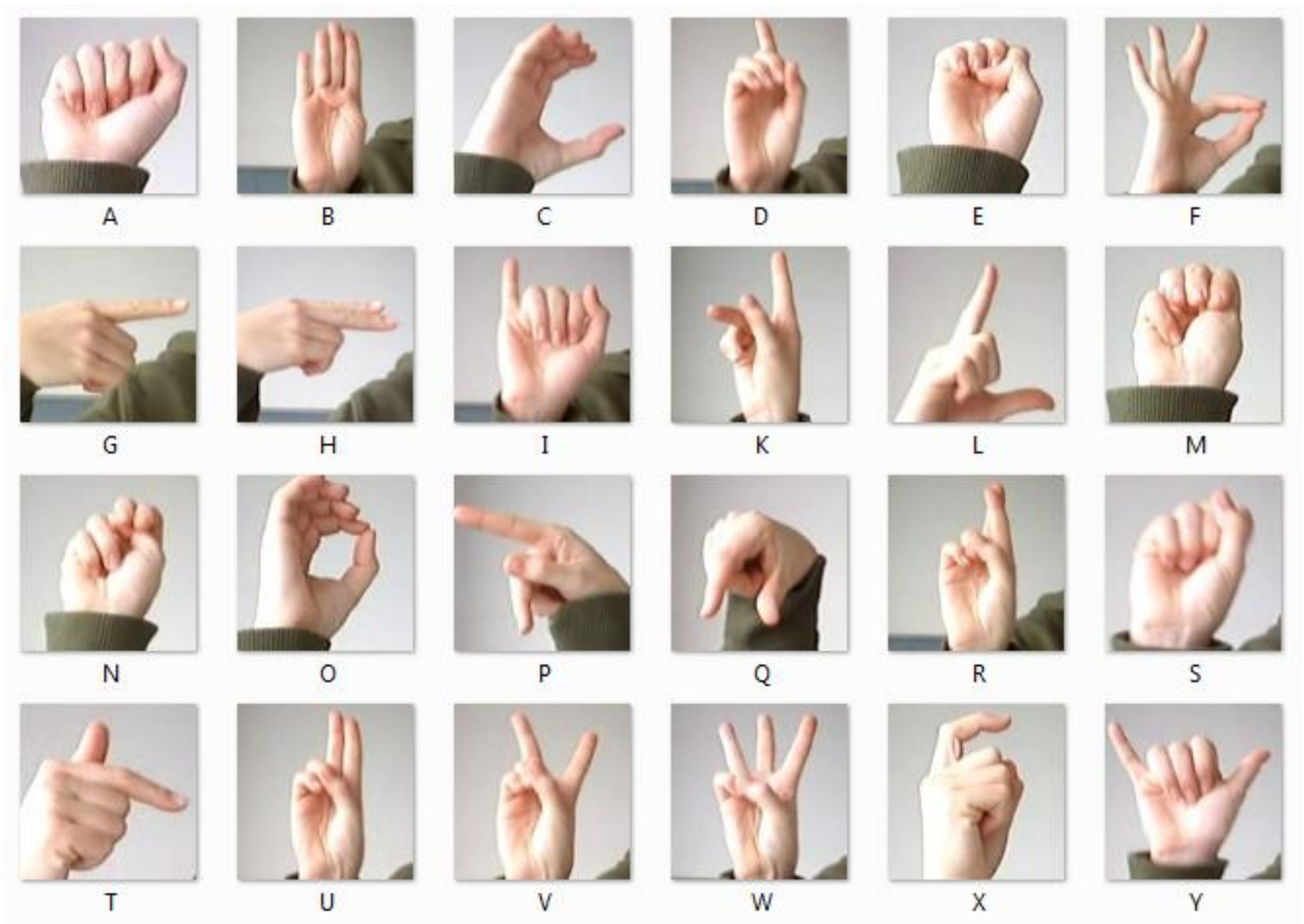
INTRODUCTION

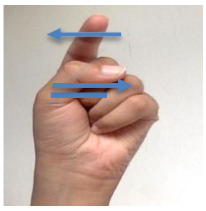
There have been several advancements in technology and a lot of research has been done to help the people who are deaf and dumb. Aiding the cause, Deep learning, and computer vision can also be used to make an impact on this cause.
Dumb people are generally deprived of normal communication with other people in the society. It been observed that they find it really difficult at times to interact with normal people with their gestures, as only a very few of those are recognized by most people. Since people with hearing impairment or deaf people cannot talk like normal people so they have to depend on some sort of visual communication in most of the time.
The goal of this project was to build a neural network able to classify which letter of the American Sign Language(ASL) alphabet is being signed, given an image of a signing hand. This project is a first step towards building a possible sign language translator, which can take communications in sign language and translate them into written and oral language. Such a translator would greatly lower the barrier for many deaf and mute individuals to be able to better communicate with others in day to day interactions.
This goal is further motivated by the isolation that is felt within the deaf community. Loneliness and depression exists in higher rates among the deaf population, especially when they are immersed in a hearing world. Large barriers that profoundly affect life quality stem from the communication disconnect between the deaf and the hearing. Some examples are information deprivation, limitation of social connections, and difficulty integrating in society .
Most research implementations for this task have used depth maps generated by depth camera and high resolution images. The objective of this project was to see if neural networks are able to classify signed ASL letters using simple images of hands taken with a personal device such as a laptop webcam. This is in alignment with the motivation as this would make a future implementation of a real time ASL-to-oral/written language translator practical in an everyday situation.

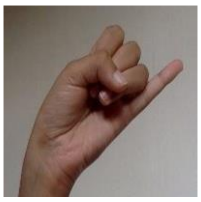
ASL Recognition System
A

AMERICAN SIGN LANGUAGE (ASL):

American Sign Language (ASL) is the non-verbal way of communication based on English language. Which can be expressed by movements of the hands and face. E3It is the primary language of many North Americans who are deaf and find difficulties in hearing . It is not a universal sign language. Different sign languages are used in different countries or regions. For example, British Sign Language (BSL) is a different language compared to ASL so the person who knows ASL may not understand BSL. ASL is forth most commonly used Language in US. ASL is a language completely segregated and different from English. ASL contains all the significant features of language, with its own rules for pronunciation, word formation, and word order. While every language has ways of indicating different functions, such as asking question instead of making a statement.




Z


J

RELATED WORK

Characterization of sign language is between two parameter one being manual and other non-manual. The manual parameter consists of motion, location, hand shape, and hand orientation. The non-manual parameter includes facial expression, mouth movements, and motion of the head . Sign language does not include the environment which kinetics does. Few terms are use in the sign language like signing space, which refers to signing taking place in 3D space and close to truck and head. Signs are either one-handed or two-handed. When only the dominant hand is in use to perform the signs they are denoted as one-hand signs else whenthe non-dominant hand also comes in the phase it is termed as two- handed signs.

Sign language when evolved is different from spoken language so the grammar of the sign language is primarily different from spoken language. In spoken language, the structure of the sentence is one-dimensional; one word followed by another, while in sign language, a simultaneous structure exists with a parallel temporal and spatial configuration. As based on these characteristics, the syntax of sign language sentence is not as strict as in spoken language. Formation of a sign language sentence includes or refers to time, location, person, base. In spoken languages, a letter represents a sound. For deaf nothing comparable exists. Hence the people, who are deaf by birth or became deaf early in their lives, have very limited vocabulary of spoken language and faces great difficulties in reading and writing.

LITERATURE

The hand gesture recognition is a well contributed research area with a lot of different approaches inimplementing it, in this chapter, I review a variety of such approaches for hand gesture recognition. My entire literature review on hand gesture recognition can be categorized into three groups, image processing/statistical modelling based recognition ,classic machine learning based recognition and deep learning based recognition.

Triesch and Malsburg in 1996 developed an ASL hand gesture recognition system ,aiming at performing accurate gesture recognition even for images captured with complex background. In this system, they have eliminated hand segmentation assuming hand images input. For hand representation or feature extraction, the system uses Gabor filters as the Gabor filters are known to resemble the receptive fields of visual cortex. Upon obtaining the Gabor features, they performed a similarity matching technique for gesture recognition called Elastic Bunch- Graph Matching (EBMS) technique. For experimentation they have considered a subset of 10 gestures from American sign language and the dataset consists of 657 images captured from 24 people in three different backgrounds (Complex, uniform light and uniform dark). The system has achieved an accuracy of 86.2% under complex background condition and overall accuracy of 91% under all background conditions. Many researches have taken the path of HMM based modeling for hand gesture recognition pursuing it as action recognition problem.

I tried to do things differently from whatever has been done. I experimented with Number of layers and neurons, Activation function and other hyper parameters of CNN model and found a optimum Tesorflow model.

8. Description of Overall Software Structure

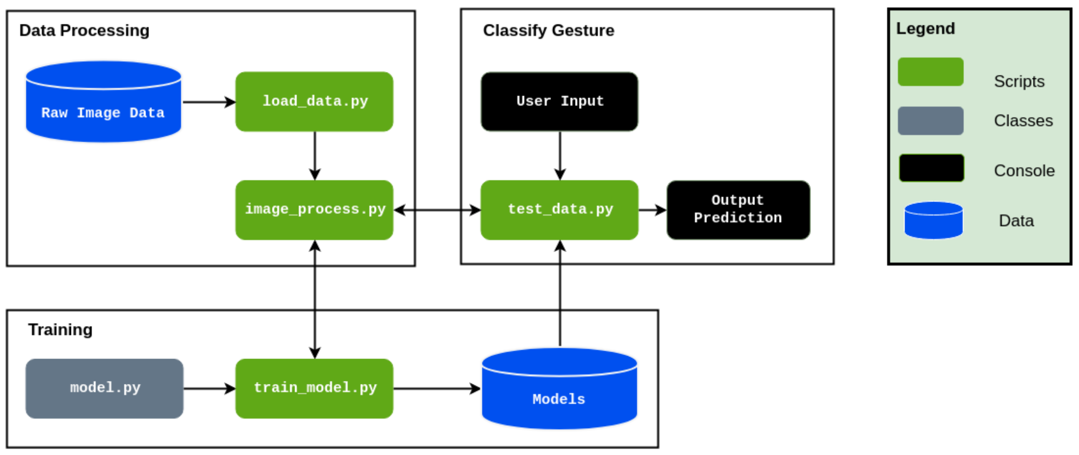


Figure : Block Diagram of Software

As shown in Figure 1, the project will be structured into 3 distinct functional blocks, **Data Processing**, **Training**,**Classify Gesture**.

1. **Data Processing** : The load data.py script contains functions to load the Raw Image Data and save the image data as numpy arrays into file storage. The process data.py script will load the image data from data.py and preprocess the image by resizing/rescaling the image, and applying filters and ZCA whitening to enhance features. During training the processed image data was split into training, validation, and testing data and written to storage. Training also involves a load dataset.py script that loads the relevant datasplit into a Dataset class. For use of the trained model in classifying gestures, an individual image is loaded and processed from the filesystem
2. **Training** : The training loop for the model is contained in train model.py. The model is trained with hyperparameters obtained from a config file that lists the learning rate, batch size, image filtering, and number of epochs. The configuration used to train the model is saved along with the model architecture for future evaluation and tweaking for improved results. Within the training loop, the training and validation datasets are loaded as Data loaders and the model is trained using Adam Optimizer with Cross Entropy Loss.The model is evaluated every epoch on the validation set and the model with best validation accuracy is saved to storage for further evaluation and use. Upon finishing training, the training and validation error and loss is saved to the disk, along with a plot of error and loss over training
3. **Classify Gesture** : After training, it can be used to classify a new ASL gesture that isavailable as a file on the filesystem. The user inputs the file path of the gesture image and the test data.py script will pass the file path to process data.py to load and preprocess the file the same way as the model has been trained.

9. DATA PROCESSING

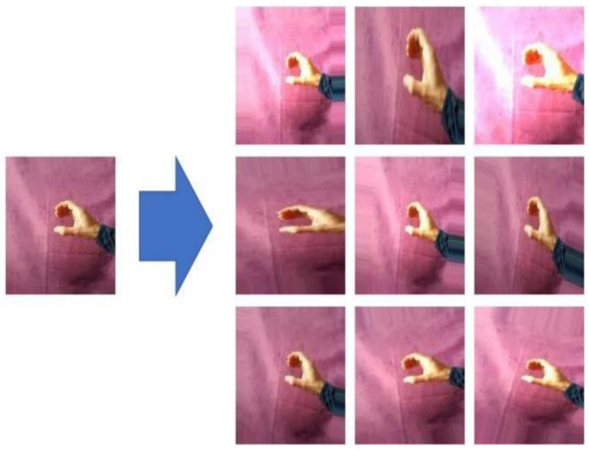
Data Processing is a task of converting **data** from a given form to a much more usable and desired form i.e. making it more meaningful and informative.
Using **Machine Learning** algorithms, mathematical modelling and statistical knowledge, this entire **process** can be automated. It involved several steps for this project which has been explained below.

9.1 Data Collection :

ASL alphabets dataset (Link) has been taken from Kaggle to train CNN model for this project. Dataset has 26 alphabets (A-Z) and 3 other (“nothing”, “del”, “space”) classes i.e. total 29 type of classes. The Dataset has 87 K of images in which each class has 3000 images. Each image is in RGB format of a particular size. Splitted dataset into traning and test dataset using **validation_split=0.2**

9.2 Data Augmentation :

data augmentation is a technique that can be used to artificially expand the size of a training dataset by creating modified versions of images in the dataset. Generated 6 images by each image so dataset size enlarged by 6 times.Training deep learning neural network models on more data can result in more skillful models, and the augmentation techniques can create variations of the images that can improve the ability of the fit models to generalize what they have learned to new images.The Keras **deep learning** neural network library provides the capability to fit models using image **data augmentation** via the **ImageDataGenerator** class.



9.3 Image Processing and Segmentation

1. Minimizing Background effect

As image in dataset might have different background which might confuse model like shown below.

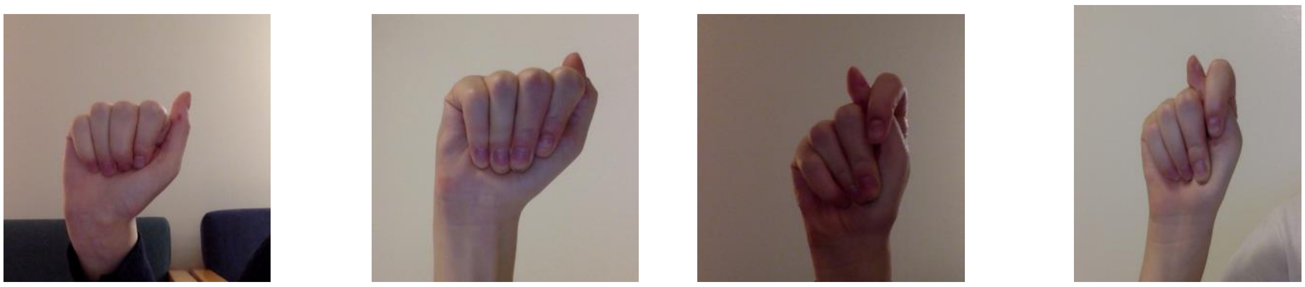


Figure : Examples of the signed letter A T from two test sets with differing lighting and background

So we need to remove the effect of background from image before feeding it to CNN model. In order to minimize the background effect , I applied the following steps :

Image Enhancement : A combination of brightness, contrast, sharpness, and color enhancement was used on the images. For example, the contrast and brightness were changed such that fingers could be distinguished when the image was very dark.

Edge Enhancement : Edge enhancement is an image filtering techniques that makes edges more defined. This is achieved by the increase of contrast in a local region of the image that is detected as an edge. This has the effect of making the border of the hand and fingers, versus the background, much more clear and distinct. This can potentially help the neural network identify the hand and its boundaries.

Image Whitening: ZCA, or image whitening, is a technique that uses the singular value decomposition of a matrix. This algorithm decorrelates the data, and removes the redundant, or obvious, information out of the data. This allows for the neural network to look for more complex and sophisticated relationships, and to uncover the underlying structure of the patterns it is being trained on. The covariance matrix of the image is set to identity, and the mean to zero.

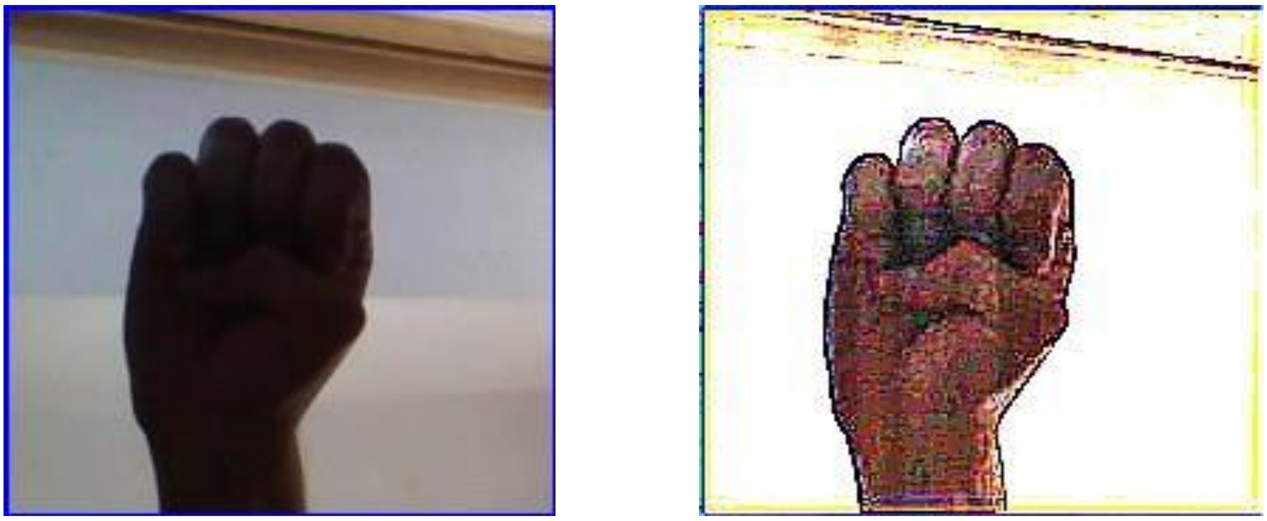


figure : Example of image processing for background effect minimization

2. RGB Color Recognition

Basically, any color image is a combination of red, green, blue colors. An important trade-off when implementing a computer vision system is to select whether to differentiate objects using colour or black and white and, if colour, to decide what colour space to use (red, green, blue or hue, saturation, luminosity). For the purposes of this project, the detection of skin and marker pixels is required, so the colour space chosen should best facilitate this. The camera available permitted the detection of colour information. Although using intensity alone (black and white) reduces the amount of data to analyze and therefore decreases processor load it also makes differentiating skin and markers from the background much harder (since black and white data exhibits less variation than colour data). Therefore it was decided to use colour differentiation.
Further maximum and minimum HSL pixel colour values of a small test area of skin were manually calculated. These HSL ranges were then used to detect skin pixels in subsequent frame (detection was indicated by a change of pixel colour to white). But Hue, when compared with saturation and luminosity, is surprisingly bad at skin differentiation (with the chosen background) and thus HSL shows no significant advantage over RGB.
Moreover, since conversion of the colour data from RGB to HSL took considerable processor time it was decided to use RGB. We will take the color image. Then make required portion of image as white by using Thresholding technique (as explained below) and garbage part that is background as black. Then we get black and white image.

3. Color image to Binary image Conversion

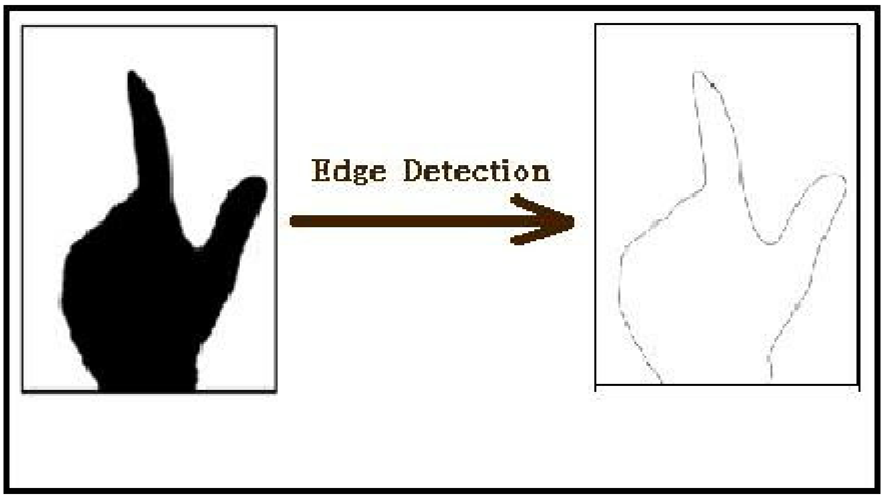
To convert any color to a grayscale representation of its luminance, first one must obtain the values of its red, green, and blue (RGB) primaries. Grayscale or grayscale digital image is an image in which the value of each pixel is a single sample, that is, it carries only intensity information. Images of this sort, also known as black and white, are composed exclusively of shades of gray, varying from black at the weakest intensity to white at the strongest. A binary image is a digital image that has only two possible values for each pixel. Typically the two colors used for a binary image are black and white though any two colors can be used. The color used for the object in the image is the foreground color while the rest of the image is the background colour. Until now a simple RGB bounding box has been used in the classification of the skin and marker pixels .

(a)

(b)

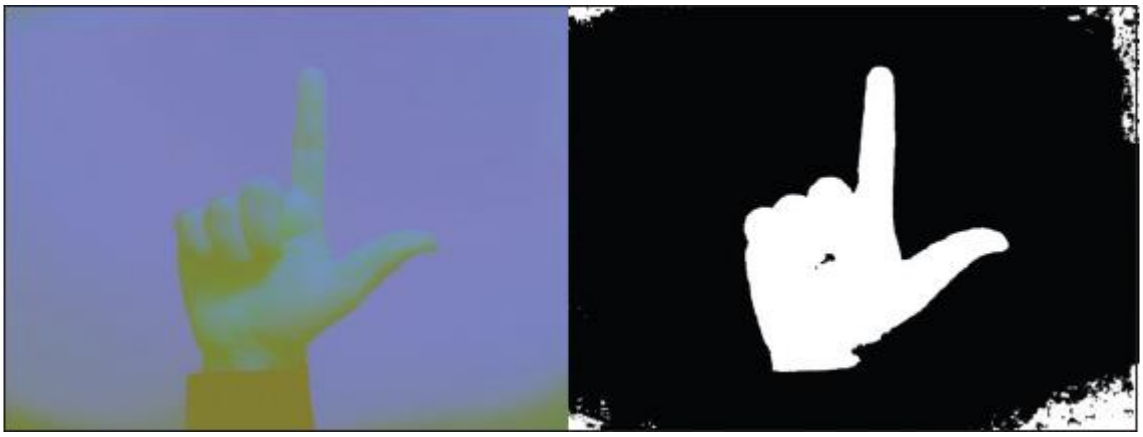
4. Edge Detection

One of the methods we tried was using a Canny edge detector to find relevant ”objects” in the field of view of the camera. The edges were then di- lated, and then all remaining holes in the mask were filled to create a solid, continuous mask. Once this was done, only the largest areas were taken in order to remove all the background clut- ter objects. This approach makes the simplify- ing assumption that the biggest objects seen in segmentation are typically of the most interest as well.



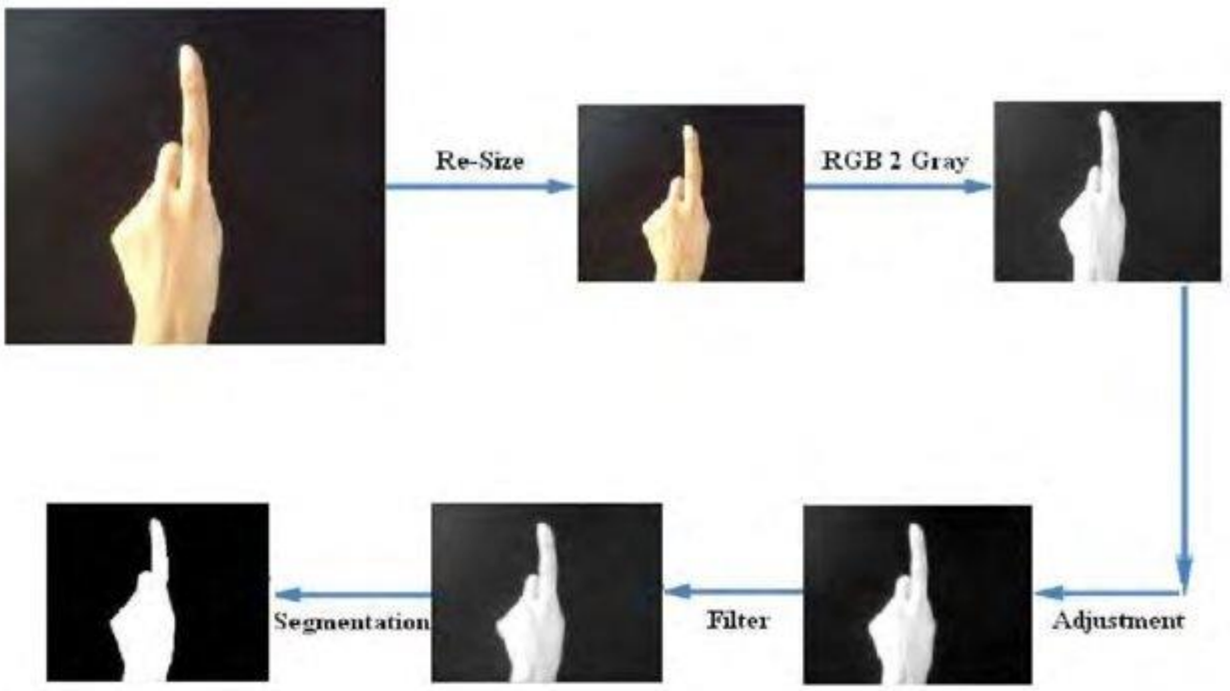
5. Image Segmentation & Thresholding

Thresholding is the simple method of image segmentation. In this method we convert the RGB image to Binary image. Figure shows the details of image processing Binary image is digital image and has only two values (0 or 1). For each pixel typically two colors are used black and white though any two colors can be used. Here, the background pixels are converted into black color pixels and pixels containing our area of interest are converted into white color pixels. It is nothing but the preprocessing. Non-black pixels in the transformed image are binarised while the others remain unchanged, therefore black.The hand gesture is segmented firstly by taking out all the joined components in the image and secondly by letting only the part which is immensely connected, in our case is the hand gesture

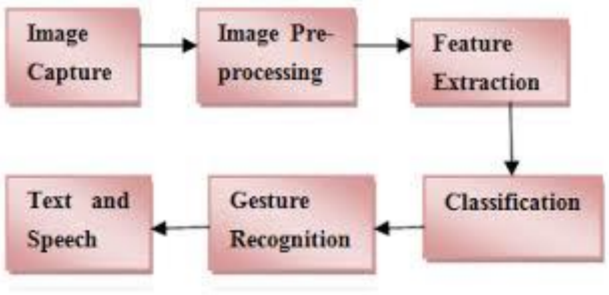


After Thresholding to input image with a proper threshold which depends on background as well, we have landmark points of hand gesture . We feed this resized binarized image to CNN model to identify hand gesture and displaying the prediction.

Flow chart shown below depicts all the process done in image processing and segmentation part for a particular ASL alphabet. Same process is done for all data image before feeding it to neural network as input.



Data has been pre-processed and ready for feeding it into CNN Model for Feature extraction. Following schematic diagram shows all processes in sequence.



10. **CNN Model Architecture**

10.1 **Neural Network Design**

A neural network is basically modelled as the structure shown in figure, in which can be observed a group of elements that interact to generate an output vector from an input vector which is described by the variable x. The training information is stored in the set of synaptic weight valuesof the neural network, and the output neuron is limited to a specific range of values of the activation function.

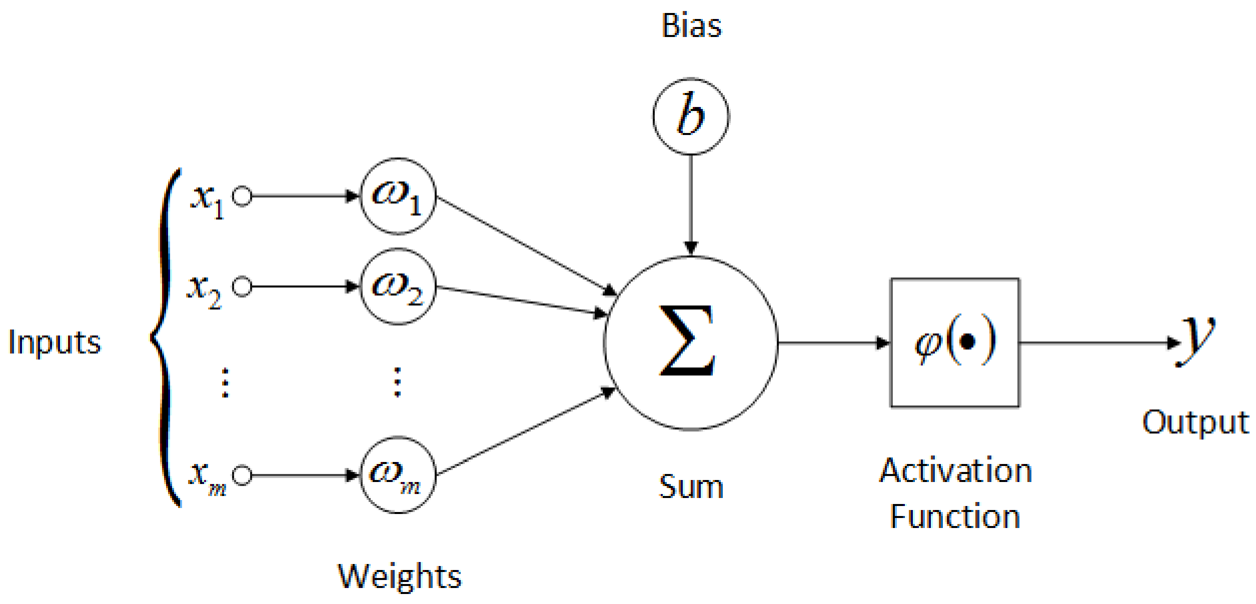
Output neuron can be shown

$$y_k=\phi(\sum w_{ki}.x_i+w_o), \qquad (1)$$

or

$$y_k= \phi(v_k) \qquad (2)$$

where the subscript i indexes units in the input layer, k in the hidden; w ki denotes the input to hiddenlayer weights at the hidden unit k. An adder ∑ which produces the weighted sum of inputs accordingto the respective weights of the connections. A activation function defines the output amplitude of that node given an input or set of inputs $\phi(v_k)$, and w_o is a threshold value.



A multilayer neural network was used in the design with a backpropagation algorithm. The structure of the network is formed by three layers, called the input layer, hidden layer and output layer.

the basic components can be seen in figure in which was used a simplified graphic notation. For the input and hidden layer neurons were employed a hyperbolic tangent activation function with 5 neurons each one and one neurons at the output layer with a linear activation function.

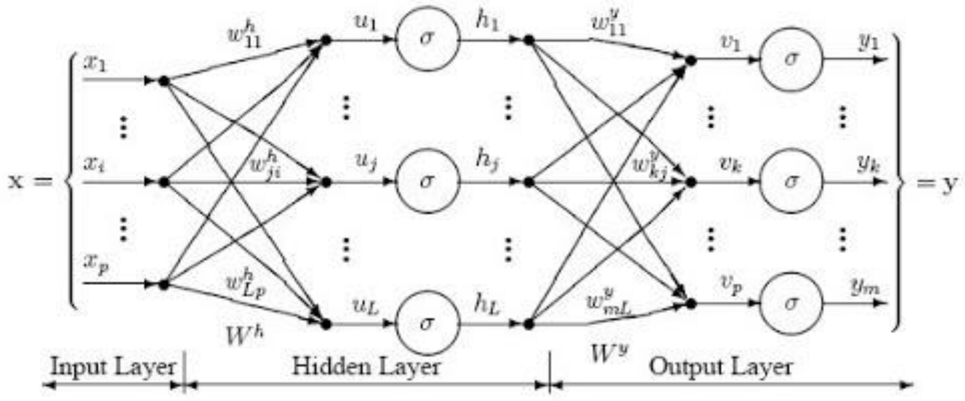
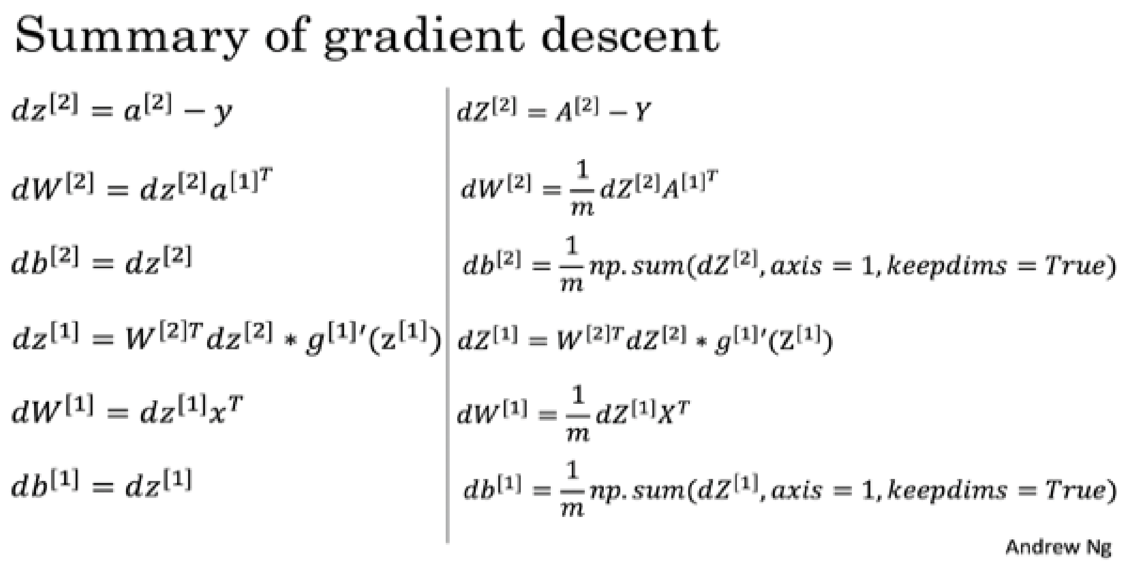


Figure : Employed Multilayer Neural Network

We need to propagate forward and backward for minimizig the loss function to calculating the weightages and other parameters. The following Gradient formulae are used to update weightage in each step.



Summary of gradient descent

$dz^{[2]} = a^{[2]} - y$	$dZ^{[2]} = A^{[2]} - Y$
$dW^{[2]} = dz^{[2]} a^{[1]^T}$	$dW^{[2]} = \frac{1}{m} dZ^{[2]} A^{[1]^T}$
$db^{[2]} = dz^{[2]}$	$db^{[2]} = \frac{1}{m} np.sum(dZ^{[2]}, axis = 1, keepdims = True)$
$dz^{[1]} = W^{[2]T} dz^{[2]} * g^{[1]\prime}(z^{[1]})$	$dZ^{[1]} = W^{[2]T} dZ^{[2]} * g^{[1]\prime}(Z^{[1]})$
$dW^{[1]} = dz^{[1]} x^T$	$dW^{[1]} = \frac{1}{m} dZ^{[1]} X^T$
$db^{[1]} = dz^{[1]}$	$db^{[1]} = \frac{1}{m} np.sum(dZ^{[1]}, axis = 1, keepdims = True)$

Andrew Ng

10.2 **CONVOLUTIONAL NUERAL NETWORK:**

Convolutional Neural Network is a deep learning technique which is developed from the inspiration of visual cortex which are the fundamental blocks of human vision. It is observed from the research that, thehuman brain performs aarge-scale convolutions to process the visual signals received by eyes, based on this observation CNNs are constructed and observed to be out performing all the prominent classification techniques. Two major operations performed in CNN are convolution ($wT * X$) and pooling(max()) and these blocks are wired in a highly complex fashion to mimic the human brain.
The neural network is constructed in layers, where the increase in the number of layers increases the network complexity and is observed to improve the system accuracy .The CNN architecture consists of three operational blocks which are connected as a complex architecture.

The functional blocks of Convolutional Neural Network
1.Convolutional Layer
2.Max Pooling layer
3.Fully-Connected layer

10.3 **Feature Extraction through CNN :**

The main building block of CNN is the convolutional layer. Convolution is a mathematical operation to merge two sets of information. In our case the convolution is applied on the input data using a convolution filter to produce a feature map.
We perform the convolution operation by sliding this filter over the input.

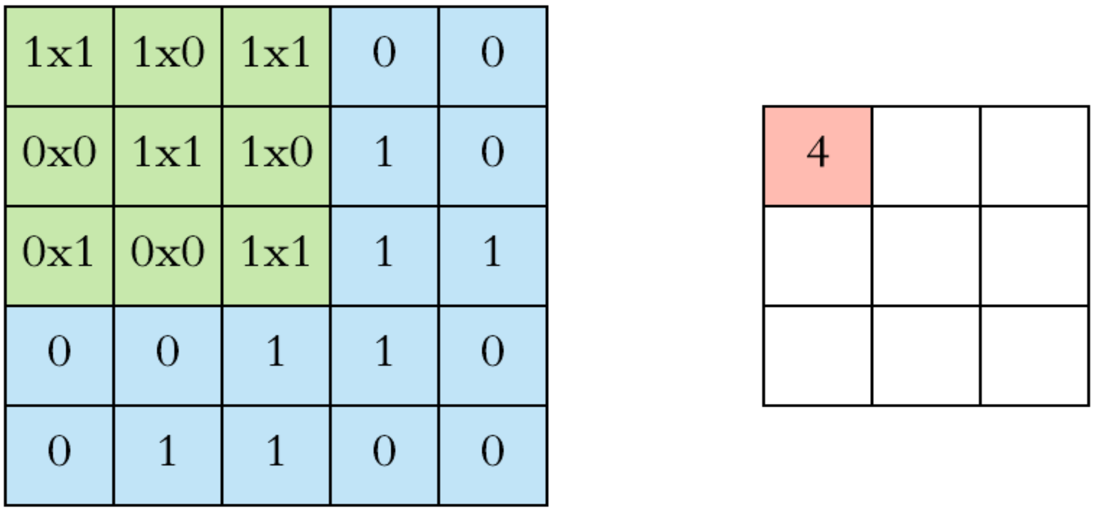
1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Input

1	0	1
0	1	0
1	0	1

Filter / Kernel

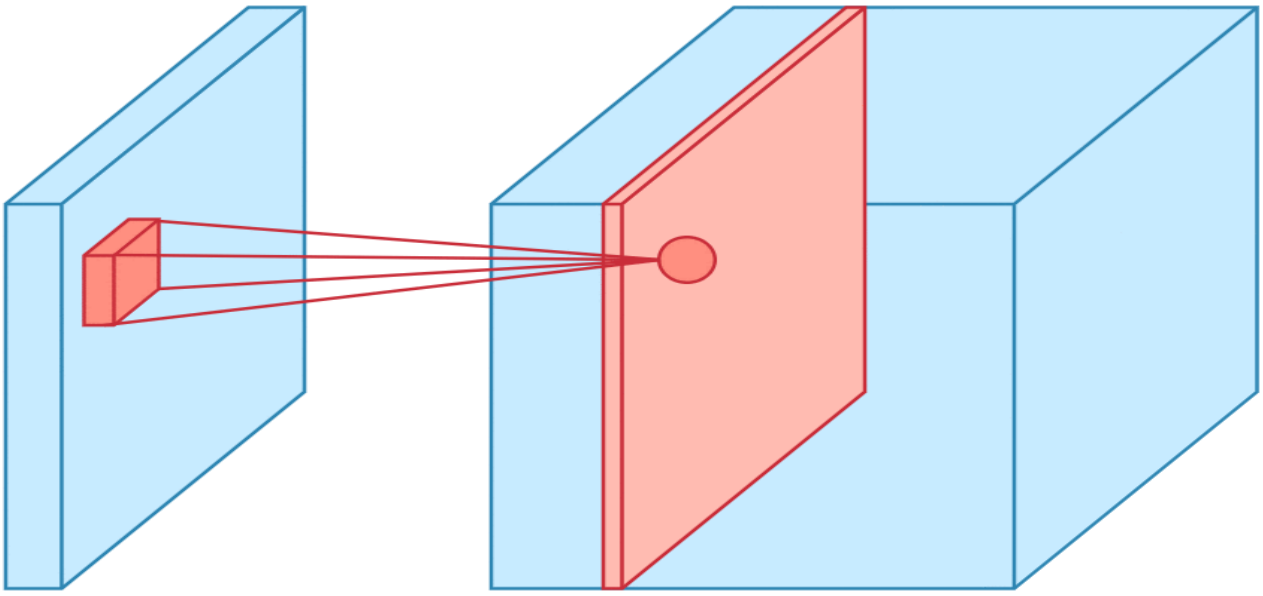
At every location, we do element-wise matrix multiplication and sum the result. This sum goes into the feature map. The green area where the convolution operation takes place is called the receptive field. We then slide the filter to the right and perform the same operation, adding that result to the feature map as well.



1x1	1x0	1x1	0	0
0x0	1x1	1x0	1	0
0x1	0x0	1x1	1	1
0	0	1	1	0
0	1	1	0	0

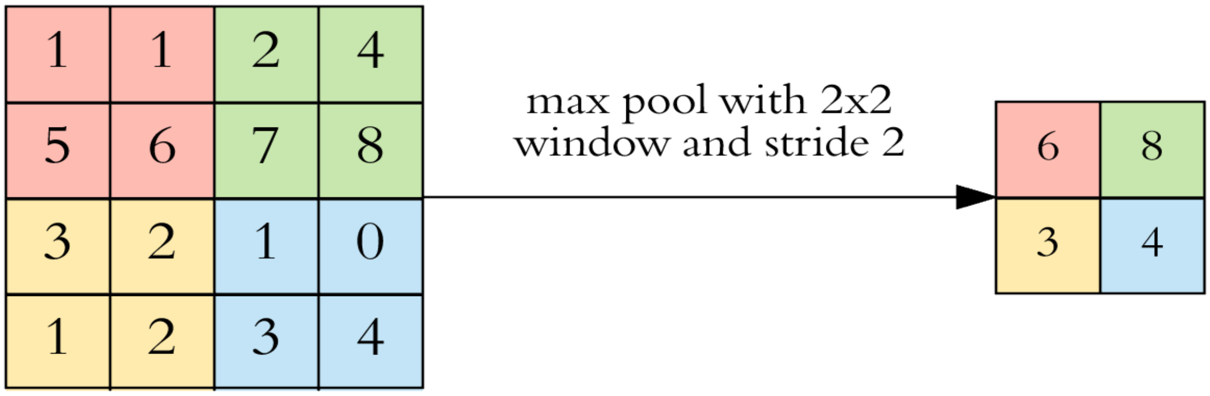
4		

We perform multiple convolutions on an input, each using a different filter and resulting in a distinct feature map. Let’s say we have a 32x32x3 image and we use a filter of size 5x5x3.Then Convolution happens like below .

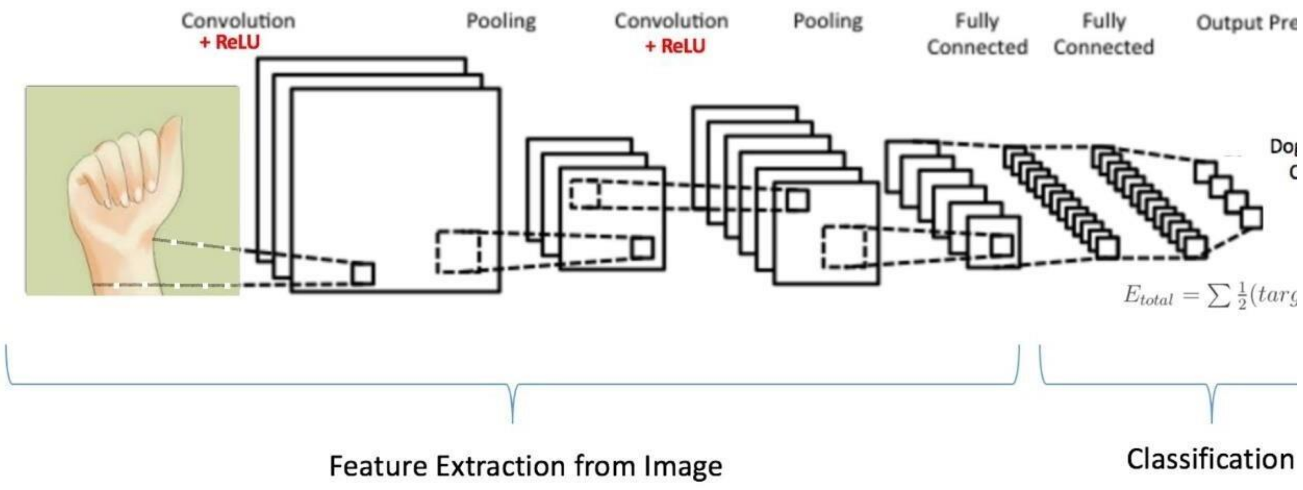


Max Pooling :

After a convolution operation we usually perform pooling to reduce the dimensionality. This enables us to reduce the number of parameters, which both shortens the training time and combats overfitting. Pooling layers down sample each feature map independently, reducing the height and width, keeping the depth intact. The most common type of pooling is max pooling which just takes the max value in the pooling window. Contrary to the convolution operation, pooling has no parameters. It slides a window over its input, and simply takes the max value in the window. Similar to a convolution, we specify the window size and stride.

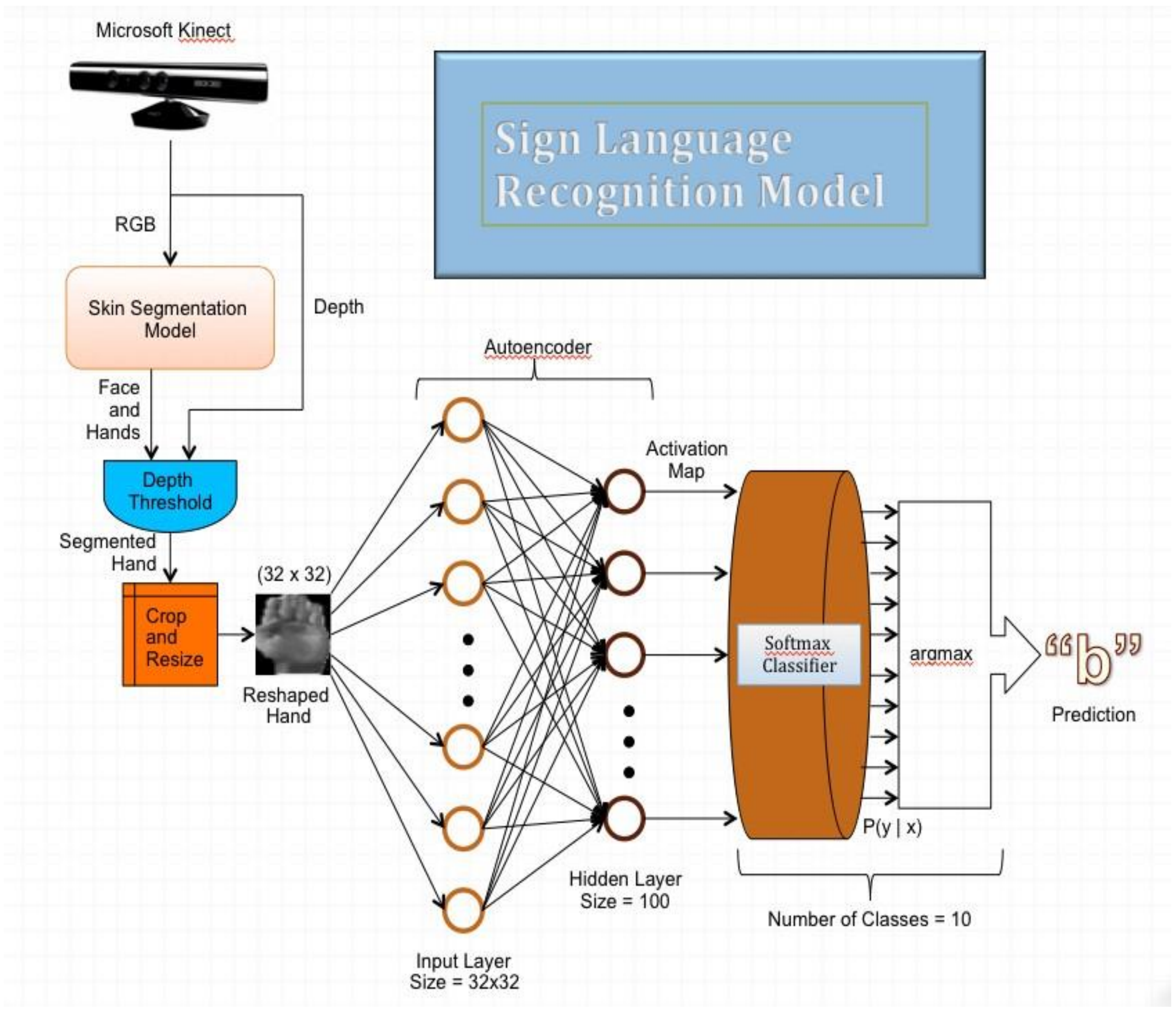


To extract the features from ASL pre-processed images , we feed those image in CNN model ,and Feature extraction takes place like below. class prediction is done by softmax layer at end.



11. **System Architecture :**

A CNN model is used to extract features from the frames and to predict hand gestures. It is a multilayered feedforward neural network mostly used in image recognition. The architecture of CNN consists of some convolution layers, each comprising of a pooling layer, activation function, and batch normalization which is optional. It also has a set of fully connected layers. As one of the images moves across the network, it gets reduced in size. This happens as a result of max pooling. The last layer gives us the prediction of the class probabilities.



Classification :

In our proposed system, we apply a 2D CNN model with a tensor flow library. The convolution layers scan the images with a filter of size 3 by 3. The dot product between the frame pixel and the weights of the filter are calculated. This particular step extracts important features from the input image to pass on further. The pooling layers are then applied after each convolution layer. One pooling layer decrements the activation map of the previous layer. It merges all the features that were learned in the previous layers' activation maps. This helps to reduce overfitting of the training data and generalizes the features represented by the network. In our case, the input layer of the convolutional neural network has 32 feature maps of size 3 by 3, and the activation function is a Rectified Linear Unit. The max pool layer has a size of 2×2. The dropout is set to 20 percent and the layer is flattened. The last layer of the network is a fully connected output layer with 29 units, and the activation function is Softmax. Then we compile the model by using "category cross-entropy" as the loss function and "Adam" as optimizer.

12. MODEL TRAINING PROCESS :

During the training process of the first stage, it is used a backpropagation algorithm. The supervised backpropagation learning scheme modifies the weight in the opposite direction of thegradient of the error function to minimize a mean squared error of whole patterns, which are used totrain the neural network. These algorithms build models that predict the desired values. Its an gradient based algorithm, which start with the initial weight vector, estimates the error function andits gradient for training, and it is obtained a new modified weight vector. This is repeated till the errorfinds the set limit [8]. Therefore, by definition, the weights are updated through the expression:

$$w^{m+1}=w^{m}+\alpha(-\nabla^{m}), \qquad (3)$$

where is α the learning rate of the network, and ∇ gradient of error function about to wm .Inthe backpropagation algorithm is used the mean squared error that is calculated from a desired output m d as:

$$(e^{m})^{2}=(d^{m}-w^{m}.x^{m})^{2} \qquad (4)$$

therefore, the gradient is obtained from the error

$$\nabla^{m}=-2.e^{m}\,\phi(v^{m}).x^{m} \qquad (5)$$

Replacing in (3), it is obtained the following expression:

$$w^{m+1}=w^{m}+2.\alpha.\phi(v^{m}).x^{m} \, (6)$$

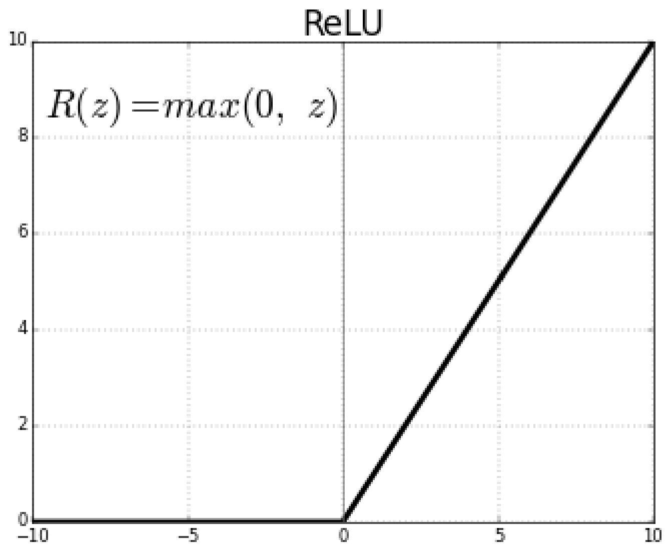
This process is made for all the neurons of each layer in the network.

12.1 ACTIVATION FUNCTIONS :

In a neural network, the activation function is responsible for transforming the summed weighted input from the node into the activation of the node or output for that input
A neural network without an activation function is essentially just a linear regression model
There are different type of activation functions which are used in different purpose.I trained my model with both sigmoid and Relu activation functions separately and found out Relu activation was working the best for this project. Let's see what is advantage of using Relu over sigmoid and why it is giving better result in this case.

(i) ReLU Activation Function :

In order to use stochastic gradient descent with backpropagation of errors to train deep neural networks, an activation function is needed that looks and acts like a linear function, but is, in fact, a nonlinear function allowing complex relationships in the data to be learned.
The rectified linear activation function overcomes the vanishing gradient problem & allowing models to learn faster and perform better. The rectified linear activation is the default activation when developing multilayer Perceptron and convolutional neural networks. ReLU function is its derivative both are monotonic.

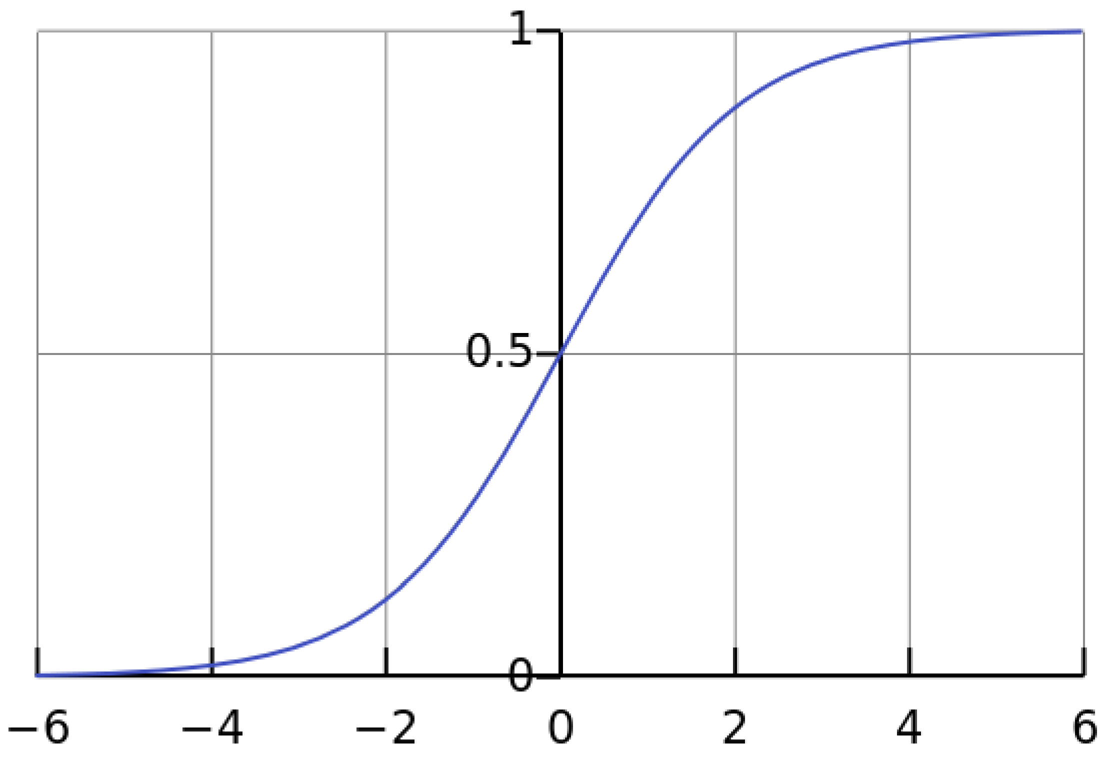


(ii) Sigmoid Activation Function

A sigmoid function is a bounded, differentiable, real function that is defined for all real input values and has a non-negative derivative at each point and exactly one inflection point. A sigmoid "function" and a sigmoid "curve" refer to the same object. In general, a sigmoid function is monotonic, and has a first derivative which is bell shaped.

$$A = \frac{1}{1+e^{-x}}$$

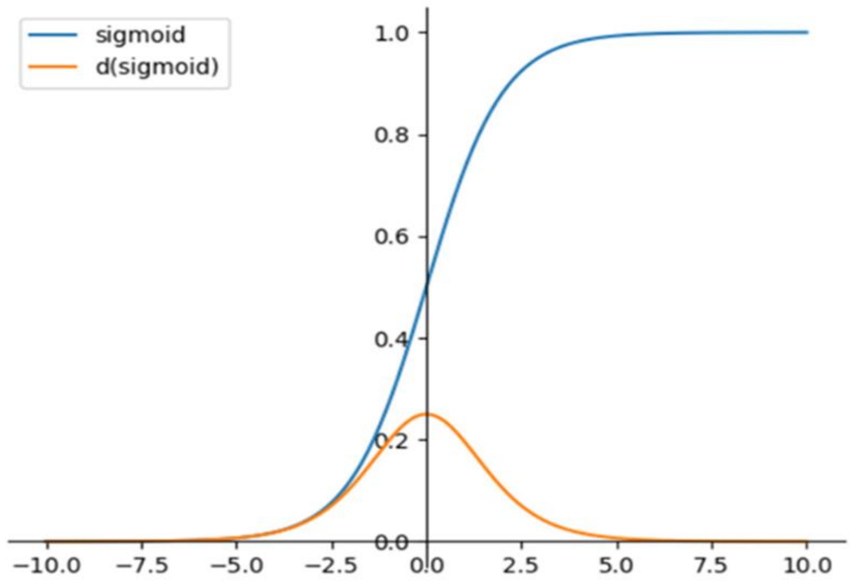
The main reason why we use **sigmoid function** is because it exists between (0 to 1). Therefore, it is especially used for models where we have to predict the probability.



Advantage of ReLU over Sigmoid :

Sigmoid Activation function has issue of vanishing gradient. As derivative of sigmoid always lies between 0 and 0.25 , so after few iteration updation rate becomes too small which makes convergence as well training very slow.

$$\frac{d\sigma}{dx} = \sigma(x)(1 - \sigma(x))$$



Sigmoid tend to vanish gradient (cause there is a mechanism to reduce the gradient as "a" increases, where "a" is the input of a sigmoid function. Gradient of Sigmoid: S′(a)=S(a)(1−S(a)).

When "a" grows to infinite large,

S′(a)=S(a)(1−S(a))=1×(1−1)=0.

Alongwith with this , sigmoid requires complicated computation. while ReLU has very easy computation and it does not has Vanishing gradient problem. So ReLU is preferable whenever all such factors are making contribution.

After Training Model with both activation functions separately, I observed ReLU function was having better accuracy and convergence rate. So I used ReLU as activation function for my model.

13. Model Overview :

After Training Model with a lot of variations of Activation functions , Number of layers , Optimizer and hyper parameters. I found optimum model for this project which has following

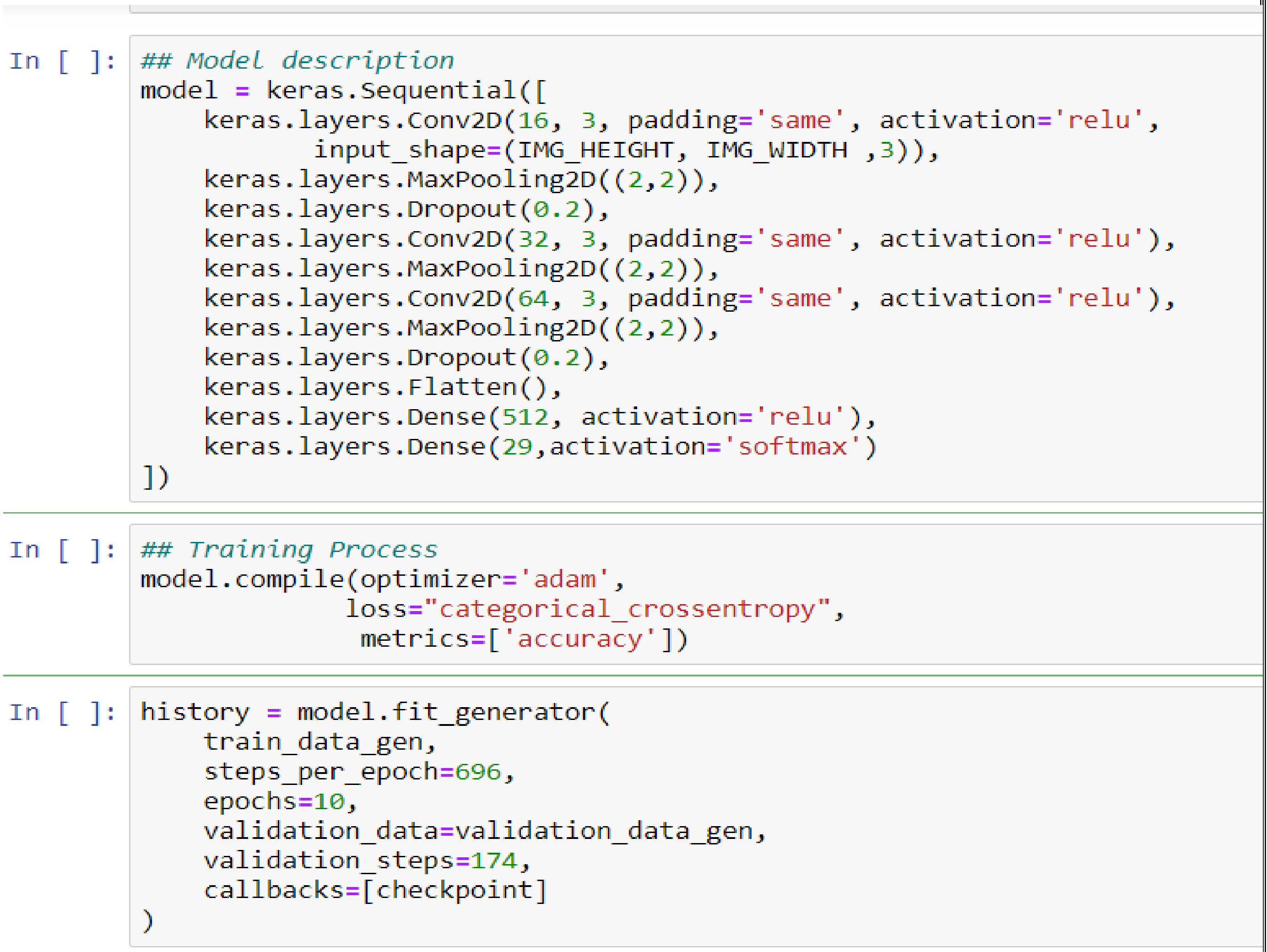
(1) 5 Layers

3 Conv2D each followed by Max pooling layer and dropout ,

2 Fully connected (FC) Layers with output layer having 29 classes

(2) Activation Function : **ReLU**

(3) Optimizer : **Adam**



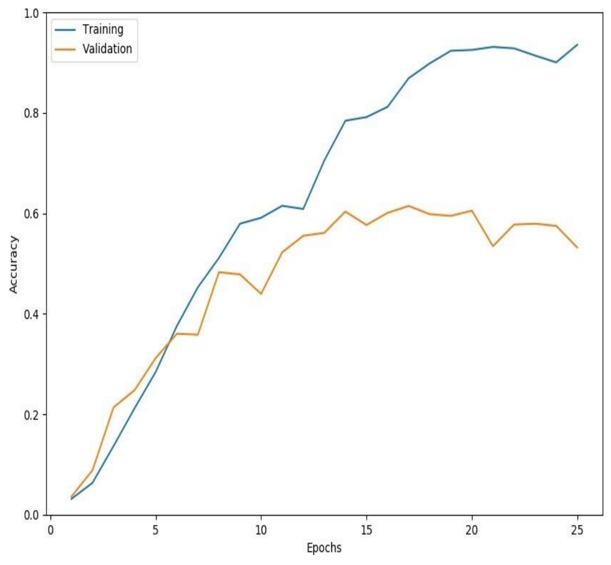
```
In [ ]: ## Model description
        model = keras.Sequential([
            keras.layers.Conv2D(16, 3, padding='same', activation='relu',
                   input_shape=(IMG_HEIGHT, IMG_WIDTH ,3)),
            keras.layers.MaxPooling2D((2,2)),
            keras.layers.Dropout(0.2),
            keras.layers.Conv2D(32, 3, padding='same', activation='relu'),
            keras.layers.MaxPooling2D((2,2)),
            keras.layers.Conv2D(64, 3, padding='same', activation='relu'),
            keras.layers.MaxPooling2D((2,2)),
            keras.layers.Dropout(0.2),
            keras.layers.Flatten(),
            keras.layers.Dense(512, activation='relu'),
            keras.layers.Dense(29,activation='softmax')
        ])
```

```
In [ ]: ## Training Process
        model.compile(optimizer='adam',
                      loss="categorical_crossentropy",
                       metrics=['accuracy'])
```

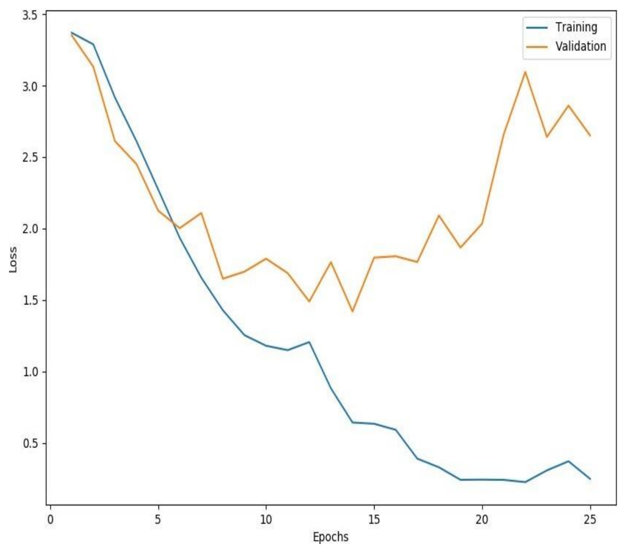
```
In [ ]: history = model.fit_generator(
            train_data_gen,
            steps_per_epoch=696,
            epochs=10,
            validation_data=validation_data_gen,
            validation_steps=174,
            callbacks=[checkpoint]
        )
```

14. **Training and Validation Accuracy of Model**

Model is performing well on test dataset. Model accuracy was 95.5% on new test dataset which has not been used in training.After training of Model , plotted few graphs which shows how does accuracy improves with increasing number of epochs for validation ad training dataset

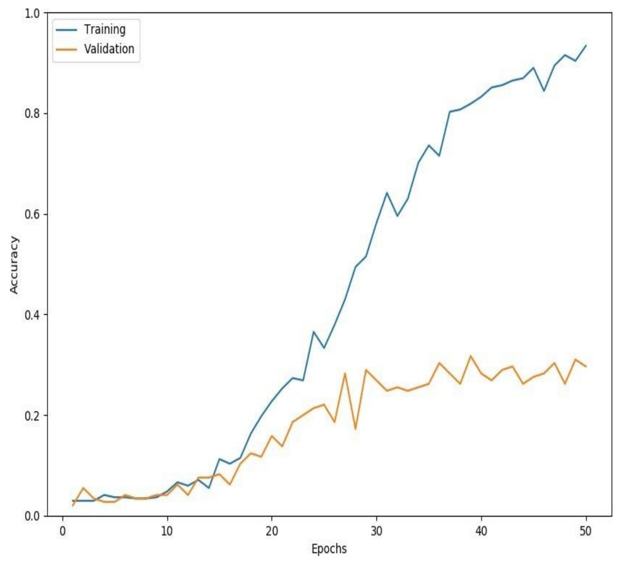


(a) Validation &Training Accuracy

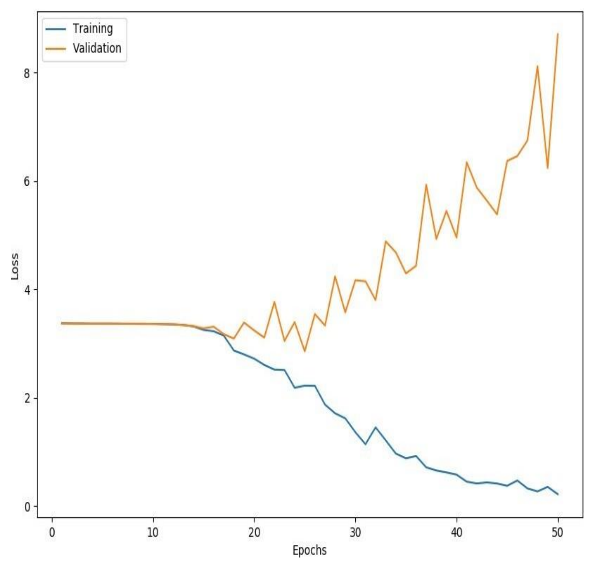


(b) Validation &Training Loss

Figure : Validation & Training Performance of Model Trained on Original Image



(a) Validation &Training Accuracy



(b) Validation &Training Loss

Figure: Validation & Training Performance of Initial model Trained on small datase

15. CONCLUSION

In this paper, a real-time ASL fingerspelling recognition with CCNs networks was built using real coloring images to help creating a writing system for use as the input to a PC using a normal webcam. However, the model that was built in this work is the first fingerspelling recognition system to classify a total of 26 alphabets (A-Z) and three other classes for "space" , "delete" and "nothing". New datasets were built to contain a wider variety of features for example different skin tones, different lightings, different backgrounds, and a wide variety of hand gestures. This work utilized 29 classes, comprised of 26 classes for American Sign Language alphabets from A to Z and three others "del", "space" and "nothing"

The system has been trained on 87 k data from Kaggle which includes 3000 images for each class. Model achieved a maximal accuracy of about 96.40 % for training and 96.75 % for the validation set. In addition, the system showed a high accuracy with the introduction of new test data that had not been used in the training, with the lowest accuracy achieved, which is 95.95 %. Attempts should be made to recognize the dynamic ASL gestures in future works. In addition, the dataset should be improved by involving more volunteers to cover all the different skin tones and increase the number of images in each class.

DIFFICULTIES

Sign languages are very broad and differ from country to country in terms of gestures, body language and face expressions. The grammars and structure of a sentence also varies a lot. There were two groups {V,K} and {A,S} of alphabets which were quite similar in appearance in ASL. Due to this similar appearance model was getting confused few times in right prediction.

FUTURE WORK

I look forward to use the numerical digits (0-9) in our datasets and improve the model so that it recognizes more alphabetical features while at the same time get a high accuracy. We would also like to enhance the system by adding speech recognition so that blind people can benefit as well.
I am working on research paper for this project and hoping to get it published soon.

REFERENCES

[1] . M. Panwar, "Hand gesture based interface for aiding visuallyimpaired," Proc. IEEE Int. Conf. Recent Adv. Comput. Softw.Syst. RACSS2012, pp. 80-85.

[2] . Aliaa A. A.Youssif, AmalElsayedAboutabl, HebaHamdyAli,"Arabic Sign Language (ArSL) Recognition System UsingHMM ", (lJACSA) International Journal of AdvancedComputer Science and Applications, Vo1.2, Issue. 11,2011

[3] .L. Gu, X. Yuan, and T. Ikenaga, "Hand gesture interface basedon improved adaptive hand area detection and contoursignature," IEEE Int. Symp. Intel!. Signal Process. Commun.Syst. (ISPACS 2012), no. Ispacs, pp. 463—468

[4] Summer Research Fellowship report by Muskan Dhiman

[5] . H. Y. Lai and H. J. Lai, "Real-Time Dynamic Hand GestureRecognition," IEEE Int. Symp. Comput. Consum. Control,2014no. 1,pp. 658-661

[6] .Maeda Y, Wakamura M 2005 Simultaneous perturbation learning rule for recurrent neural networks and its FPGA implementation IEEE Trans. Neural Network 16 6 1664 – 1672.

[7] .Gallaudet University Research Institute: A Brief Summary of Estimates for the Size of the Deaf Population

[8] Applied Deep Learning - Part 4: Convolutional Neural Networks, Arden Dertat