SIGN LANGUAGE RECOGNITION - USING HAND GESTURES IN LIVE VIDEO

M.Sc DISSERTATION

BY

RAMAN KUMAR

DEPARTMENT OF COMPUTER SCIENCE

KUMAUN UNIVERSITY NAINITAL

NAINITAL – 263001 (INDIA)

SESSION -2020- 2021

ACKNOWLEDGMENT

I would like to thank Dr. Ashish Mehta , Professor and Head , Department of Computer Science , Kumaun University, Nainital , for giving us the opportunity to undertake the project. I thank them for their immense guidance, and appreciate their timely engagement. I would like to extend special gratitude to the Mr. Anubhav Mehra for his guiding and motivation, and other teaching and non-teaching staff of department.I heartily convey my sincere and special thanks to my fellow classmates for their support and help.

CANDIDATE'S DECLARATION

I declare that this written submission represents my ideas in my own words and where other's ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misinterpreted or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the university and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

____

(Signature)

RAMAN KUMAR

(Name of the student)

190120670004

(Enrollment no./Roll no.)

Date: ___________

DEPARTMENT OF COMPUTER SCIENCE
KUMAUN UNIVERSITY NAINITAL

CERTIFICATE

I hereby declare that the work which is being presented in the dissertation entitled **"SIGN LANGUAGE RECOGNITION - USING HAND GESTURES IN LIVE VIDEO"** in partial fulfillment of the requirements for the award of the Degree of Masters of Science in Computer Science and submitted in the Department of Computer Science of the Kumaun University Nainital is an authentic record of my own work under the supervision of Dr. Ashish Mehta , Professor and Head , Department of Computer Science , Kumaun University, Nainital .

The matter presented in this dissertation has not been submitted by me for the award of any other degree of this or any other University/Institution.

(RAMAN KUMAR)
Candidate

(_________) (Dr. Ashish Mehta)
Counter sign by Convener Supervisor

Date:

Abstract

Sign language is one of the oldest and most natural form of language for communication, but since most people do not know sign language , so we have come up with a real time method using neural networks for finger-spelling based American sign language(ASL). In our method, the hand is first passed through a filter and after the filter is applied the hand is passed through a classifier which predicts the class of the hand gestures. Our method provides 80% accuracy for the 24 letters of the alphabet(excluding J and Z,as they need motion) in the real world data.

Motivation

For interaction between normal people and D&M people a language barrier is created as sign language structure which is different from normal text. So they depend on vision based communication for interaction. If there is a common interface that converts the sign language to text, the gestures can be easily understood by the other people. So research has been made for a vision based interface system where D&M people can enjoy communication without really knowing each other's language. The aim is to develop a user friendly human computer interfaces (HCI) where the computer understands the human sign language. There are various sign languages all over the world, namely American Sign Language (ASL), French Sign Language, British Sign Language (BSL), Indian Sign language(ISL), Japanese Sign Language and work has been done on other languages all around the world.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ASL	American Sign Language
ANN	Artificial Neural Network
BSL	British Sign Language
BG	Background
HCI	Human Computer Interfaces
D&M	Deaf and Mute
ISL	Indian Sign language
FSL	French Sign Language
RGB	Red Green Blue
IDE	Integrated Development Environment
CSV	Comma Separated Value
ReLu	Rectified Linear Unit
CNN	Convolutional Neural Network

Chapter 1

Introduction

Communication is a vital a part of our lives. Deaf and Mute folks being unable to talk and listen, expertise tons of issues whereas human action with traditional folks. There square measure some ways by which individuals with these disabilities try and communicate. One in every of the foremost distinguished ways in which is that the use of signing, i.e. hand gestures. it's necessary to develop an application for recognizing gestures and actions of signing in order that deaf and mute folks will communicate simply with even those that don’t perceive signing. The target of this work is to require an elementary step in breaking the barrier in communication between the conventional folks and D&M folks with the assistance of signing. American signing (ASL) could be a complete, language that has identical linguistic properties as spoken languages, with synchronic linguistics that differs from English. Signing is expressed by movements of the hands and face. It's the first language of the many North Americans United Nations agency square measure deaf and arduous of hearing, and is employed by several hearing folks furthermore.

American sign language is a predominant sign language Since the only disability D&M people have is communication related and they cannot use spoken languages hence the only way for them to communicate is through sign language. Communication is the process of exchange of thoughts and messages in various ways such as speech, signals, behavior and visuals.

Deaf and Mute(D&M) people make use of their hands to express different gestures to express their ideas with other people. Gestures are the non-verbally exchanged messages and these gestures are understood with vision. This nonverbal communication of deaf and mute people is called sign language.

Sign language is a visual language and consists of 3 major components:

Finger-spelling	**Word level sign vocabulary**	**Nonmanual features**
Used to spell words letter by letter .	Used for the majority of communication.	Facial expressions and tongue, mouth and body position.

Sign notations in ASL are as follows:

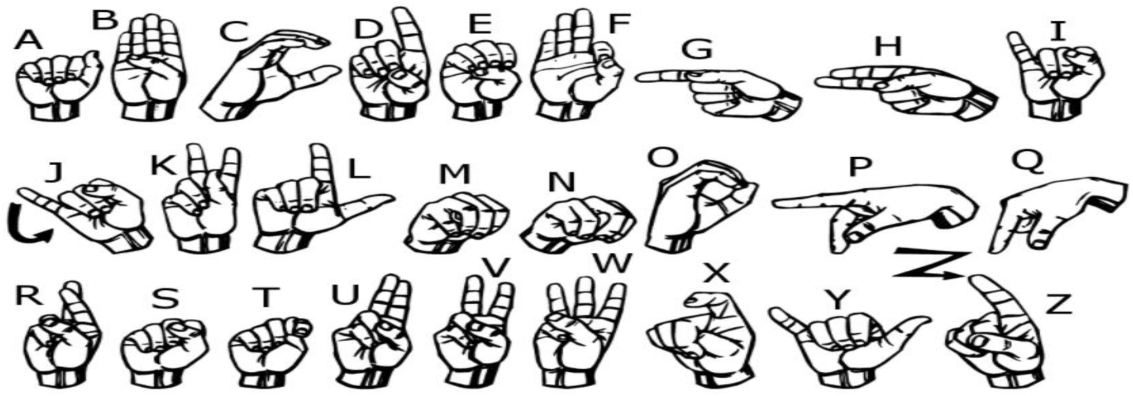


Fig 1.1 Sign Notation For ASL

Sign Language is that the most natural and communicative manner for the hearing impaired folks. Nowadays People do not feel the necessity of learning sign languages which results in isolation of the deaf folks. However if the computer can be programmed in such some way that it will translate sign language to text format, the distinction between the conventional people and also the deaf community may be decreased .

ASL has advantage over ISL, as Indian sign language (ISL) uses both hands to represent each alphabet and gesture. ISL alphabets are derived from British Sign Language (BSL) and French Sign Language (FSL)[16]. Most of the researchers in this area concentrate on the recognition of American Sign Language (ASL) since most of the signs in ASL are single handed and thus, complexity is less. Another attractive feature is that ASL already has a standard database that is available for use.

When compared with ASL, Indian Sign Language relies on both hands and thus, an ISL recognition system is more complex and ASL is simple.

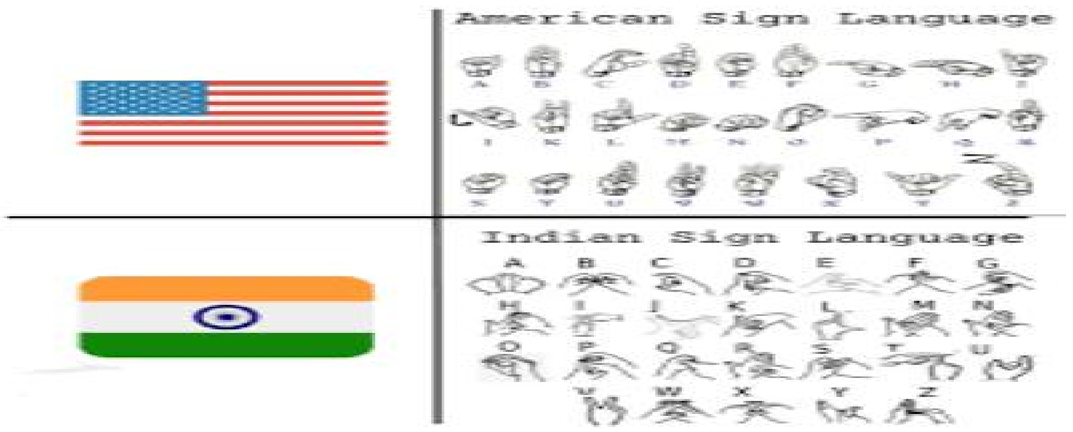


Fig 1.2 Difference between ASL and ISL

Deaf and Dumb people rely on sign language interpreters for communications. The signs are captured by using web cam. This signs are processed for feature extraction using some colour model. The extracted features are compared by using pattern matching algorithm. In order to calculate the sign recognition, the features are compared with testing database. Finally, recognized gesture is converted into text. This system provides an opportunity for a deaf-dumb people to communicate with non-signing people without the need of an interpreter.

The goal of this project is to build a neural network able to classify which letter of the American Sign Language (ASL) alphabet is being signed, given feed 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.

.

Chapter 2

Review of Literature

2.1 APPROACHES

A various hand gestures were recognized with different methods by different researchers in which were implemented in different fields. The recognition of various hand gestures were done by

1. **Vision based approaches**
2. **Data glove based approaches**
3. **Soft computing approaches like Artificial Neural Network, Fuzzy logic, Genetic Algorithm and others like PCA, Canonical Analysis, etc.**

The recognition techniques are divided into three broad categories such as

1. **Hand segmentation approaches**
2. **Feature extraction approaches**
3. **Gesture recognition approaches.**

"Application research on face detection technology uses OpenCV technology in mobile augmented reality" introduces the typical technology[16]. Open source computer vision library, OpenCV[3] for short is a cross-platform library computer vision based on open source distribution.

Data gloves and Vision based method are commonly used to interpret gestures for human computer interaction. The sensors attached to a glove that finger flexion into electrical signals for determining the hand posture in the data gloves method.

The camera is used to capture the image gestures in the vision based method. The vision based method reduces the difficulties as in the glove based method.

“Hand talk-a sign language recognition based on accelerometer and semi data” this paper introduces American Sign Language conventions. It is part of the “deaf culture” and includes its own system of puns, inside jokes, etc.

“Hand gesture recognition and voice conversion system for dumb people” proposed lower the communication gap between the mute community and additionally the standard world. The projected methodology interprets language into speech. The system overcomes the necessary time difficulties of dumb people and improves their manner.

Conversion of RGB to gray scale and gray scale to binary conversion introduced in the intelligent sign language recognition using image processing. Basically any colour image is a combination of red, green, blue colour. A computer vision system is implemented 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).

2.2 RELATED WORK

In the recent years there has been tremendous research done on the hand gesture recognition. With the help of literature survey done we realized the basic steps in hand gesture recognition are :-

- Data acquisition
- Data preprocessing
- Feature extraction
- Gesture classification

2.2.1Data acquisition:

The different approaches to acquire data about the hand gesture can be done in the following ways:

Use of sensory devices: It uses electromechanical devices to provide exact hand configuration, and position. Different glove based approaches can be used to extract information .But it is expensive and not user friendly.

Vision based approach: In this methods computer camera is the input device for observing the information of hands or fingers. The Vision Based methods require only a camera, thus realizing a natural interaction between humans and computers without the use of any extra devices. These systems tend to complement biological vision by describing 7 artificial vision systems that are implemented in software and/or hardware. The main challenge of vision-based hand detection is to cope with the large variability of human hand's appearance due to a huge number of hand movements, to different skin-colour possibilities as well as to the variations in view points, scales, and speed of the camera capturing the scene.

2.2.2 Data preprocessing and Feature extraction for vision based approach:

● In [1] the approach for hand detection combines threshold-based color detection with background subtraction.We can use Adaboost face detector to differentiate between faces and hands as both involve similar skin-color.

● We can also extract necessary image which is to be trained by applying a filter called Gaussian blur. The filter can be easily applied using open computer vision also known as OpenCV and is described in [3].

● For extracting necessary image which is to be trained we can use instrumented gloves as mentioned in [4]. This helps reduce computation time for preprocessing and can give us more concise and accurate data compared to applying filters on data received from video extraction.

● We tried doing the hand segmentation of an image using color segmentation techniques but as mentioned in the research paper skin color and tone is highly dependent on the lighting conditions due to which output we got for the segmentation we tried to do were no so great. Moreover we have a huge number of symbols to be trained for our project many of which look similar to each other like the gesture for symbol 'V' and digit '2', hence we decided that in order to produce better accuracy for our large number of symbols, rather than 8 segmenting the hand out of a random background we keep background of hand a stable single color so that we don't need to segment it on the basis of skin color . This would help us to get better results.

2.2.3 Gesture classification :

● In [1] Hidden Markov Models (HMM) is used for the classification of the gestures .This model deals with dynamic aspects of gestures.Gestures are extracted from a sequence of video images by tracking the skin-colour blobs corresponding to the hand into a body– face space centered on the face of the user. The goal is to recognize two classes of gestures: deictic and symbolic.The image is filtered using a fast look–up indexing table. After filtering, skin colour pixels are gathered into blobs. Blobs are statistical objects based on the location (x,y) and the colourimetry (Y,U,V) of the skin colour pixels in order to determine homogeneous areas.

● In [2] Naïve Bayes Classifier is used which is an effective and fast method for static hand gesture recognition. It is based on classifying the different gestures according to geometric based invariants which are obtained from image data after segmentation.Thus,unlike many other recognition methods, this method is not dependent on skin colour. The gestures are extracted from each frame of the video,with a static background. The first step is to segment and label the objects of interest and to extract geometric invariants from them. Next step is the classification of gestures by using a K nearest neighbor algorithm aided with distance weighting algorithm (KNNDW) to provide suitable data for a locally weighted Naïve Bayes‟ classifier.

● According to paper on “Human Hand Gesture Recognition Using a Convolution Neural Network” by Hsien-I Lin , Ming-Hsiang Hsu, and Wei-Kai Chen graduates of Institute of Automation Technology National Taipei University of Technology Taipei, Taiwan, they construct a skin model to extract the hand out of an image and then 9 apply binary threshold to the whole image. After obtaining the threshold image they calibrate it about the principal axis in order to center the image about it. They input this image to a convolutional neural network model in order to train and predict the outputs. They have trained their model over 7 hand gestures and using their model they produce an accuracy of around 95% for those 7 gestures.

Chapter 3

Methodologies

The system is a vision based approach. All the signs are represented with bare hands and so it eliminates the problem of using any artificial devices for interaction like electromagnetic gloves etc.

3.1 Tools Used:

- Python 3.6. or above
- TensorFlow 2.0 framework, Keras API
- Real-time computer vision using OpenCV
- IDE (Jupyter)
- Scikit-learn
- NumPy

3.2 Description of the Dataset Used

The original MNIST image dataset of alphabets using hand gestures is a popular benchmark for image-based machine learning methods but researchers have renewed efforts to update it and develop drop-in replacements that are more challenging for computer vision and original for real-world applications. As noted in one recent replacement called the Fashion-MNIST dataset, the Zalando researchers quoted the startling claim that "Most pairs of MNIST digits (785 total pixels per sample) can be distinguished pretty well by just one pixel". To stimulate the community to develop more drop-in replacements, the Sign Language MNIST is presented here and follows the same CSV format with labels and pixel values in single rows. The American Sign Language letter database of hand gestures represent a multi-class problem with 24 classes of letters (excluding J and Z which require motion).

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 and test data 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 50x50 pixel image with grayscale values between 0-255. The original hand gesture image data represented multiple users repeating the gesture against different backgrounds.

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

3.3 CNN Model Used (3 layer) :

1st Convolution Layer : The input picture has resolution of 50x50 pixels. It is first processed in the first Convolutional layer using 64 filter weights (3x3 pixels each). This will result in a 48X48 pixel image, one for each Filter-weights.

1st Pooling Layer : The pictures are down sampled using max pooling of 2x2 i.e we keep the highest value in the 2x2 square of array. Therefore, our picture is down sampled to 24x24 pixels

2nd Convolution Layer : Now, these 24 x 24 from the output of the first pooling layer is served as an input to the second convolutional layer.It is processed in the second convolutional layer using 64 filter weights (3x3 pixels each).This will result in a 22 x 22 pixel image.

2nd Pooling Layer : The resulting images are down sampled again using max pool of 2x2 and is reduced to 11x 11resolution of images.

3rd Convolution Layer : Now, these 11x 11 from the output of the Second pooling layer is served as an input to the Third convolutional layer .It is processed in the third convolutional layer using 64 filter weights (3x3 pixels each).This will result in a 9x9 pixel image.

3rd Pooling Layer : The resulting images are down sampled again using max pool of 2x2 and is reduced to 4x 4 resolution of images.

FLATTEN : We then Flatten our tensor object before input into our Dense Layer A flatten operation on a tensor reshapes the tensor to have the shape that is equal to the number of elements contained in tensor.

In our CNN it goes from 4 * 4 * 64 to 1024 * 1

1st Densely Connected Layer :

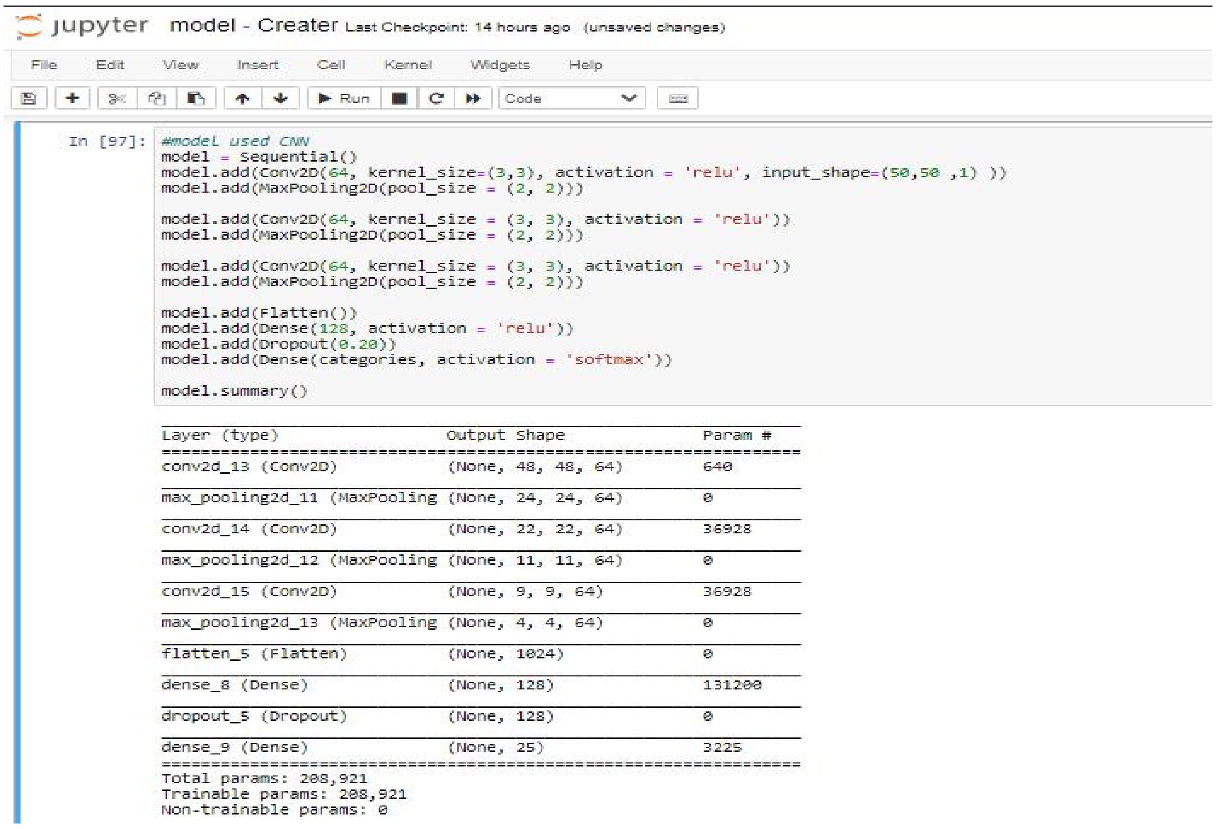
Now these images are used as an input to a fully connected layer with 128 neurons and the output from the third convolutional layer is reshaped to an array of 4x4x64 =1024 values. The input to this layer is an array of 1024 values. The output of these layer is fed to the 2nd Densely Connected Layer.

2nd Densely Connected Layer :

Now the output from the 1st Densely Connected Layer are used as an input to a fully connected layer with 25 neurons.

Final layer:

The output of the 2nd Densely Connected Layer serves as an input for the final layer which will have the number of neurons as the number of classes we are classifying (24 alphabets).


jupyter
model - Creater Last Checkpoint: 14 hours ago (unsaved changes)
File Edit View Insert Cell Kernel Widgets Help
Run
Code
In [97]:

```
#model used CNN
model = Sequential()
model.add(Conv2D(64, kernel_size=(3,3), activation = 'relu', input_shape=(50,50 ,1) ))
model.add(MaxPooling2D(pool_size = (2, 2)))

model.add(Conv2D(64, kernel_size = (3, 3), activation = 'relu'))
model.add(MaxPooling2D(pool_size = (2, 2)))

model.add(Conv2D(64, kernel_size = (3, 3), activation = 'relu'))
model.add(MaxPooling2D(pool_size = (2, 2)))

model.add(Flatten())
model.add(Dense(128, activation = 'relu'))
model.add(Dropout(0.20))
model.add(Dense(categories, activation = 'softmax'))

model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_13 (Conv2D)           (None, 48, 48, 64)        640
_________________________________________________________________
max_pooling2d_11 (MaxPooling (None, 24, 24, 64)        0
_________________________________________________________________
conv2d_14 (Conv2D)           (None, 22, 22, 64)        36928
_________________________________________________________________
max_pooling2d_12 (MaxPooling (None, 11, 11, 64)        0
_________________________________________________________________
conv2d_15 (Conv2D)           (None, 9, 9, 64)          36928
_________________________________________________________________
max_pooling2d_13 (MaxPooling (None, 4, 4, 64)          0
_________________________________________________________________
flatten_5 (Flatten)          (None, 1024)              0
_________________________________________________________________
dense_8 (Dense)              (None, 128)               131200
_________________________________________________________________
dropout_5 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_9 (Dense)              (None, 25)                3225
=================================================================
Total params: 208,921
Trainable params: 208,921
Non-trainable params: 0
```

Fig 3.1 Model Summary

Activation Function :

We have used ReLu (Rectified Linear Unit) in each of the layers(convolutional as well as fully connected neurons). ReLu calculates max(x,0) for each input pixel. This adds non-linearity to the formula and helps to learn more complicated features.It helps in removing the vanishing gradient problem and speeding up the training by reducing the computation time.

Pooling Layer :

We apply Max pooling to the input image with a pool size of (2, 2) with relu activation function.This reduces the amount of parameters thus lessening the computation cost and reduces over-fitting.

Dropout Layers:

The problem of over-fitting, where after training, the weights of the network are so tuned to the training examples they are given that the network doesn't perform well when given new examples.This layer "drops out" a random set of activations in that layer by setting them to zero.The network should be able to provide the right classification or output for a specific example even if some of the activations are dropped out[5].

We are using a dropout layer of value **0.20** to avoid over-fitting.

Optimizer :

As we have found out the cross entropy function, we have optimized it using Gradient Descent in fact with the best gradient descent optimizer is called Adam Optimizer.

We have used Adam optimizer for updating the model in response to the output of the loss function. Adam combines the advantages of two extensions of two stochastic gradient descent algorithms namely adaptive gradient algorithm(ADA GRAD) and root mean square propagation(RMSProp)

3.4 Training and Testing :

We convert our input images(RGB) into grayscale and apply Gaussian blur to remove unnecessary noise.We apply threshold to extract our hand from the background and resize our images to 50x 50 pixels.

We feed the input images after preprocessing to our model for training and testing after applying all the operations mentioned above.

Now once trained we can save our model and used it later as pre-trained model,since it’s very time consuming process. Here model is saved with name ‘**new_model77.h5**’. We can use it to test on live feed.

Testing is done with the help of in-build webcam or using an IP camera by connecting it . Now live feed is given to the model .

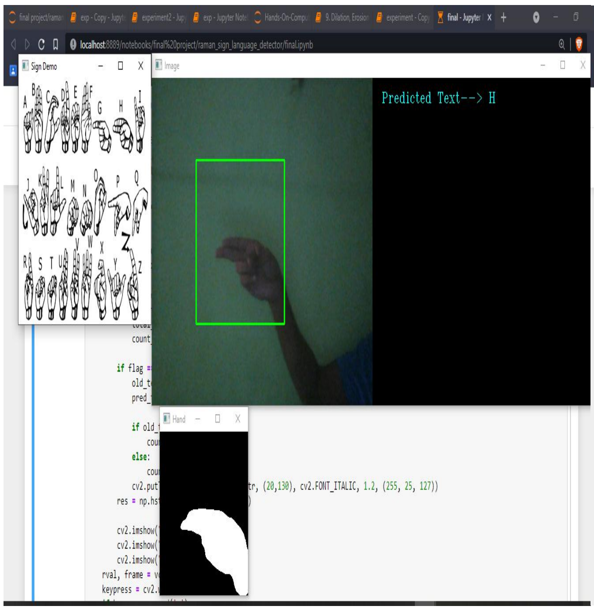


Fig 3.2 ASL sign for H

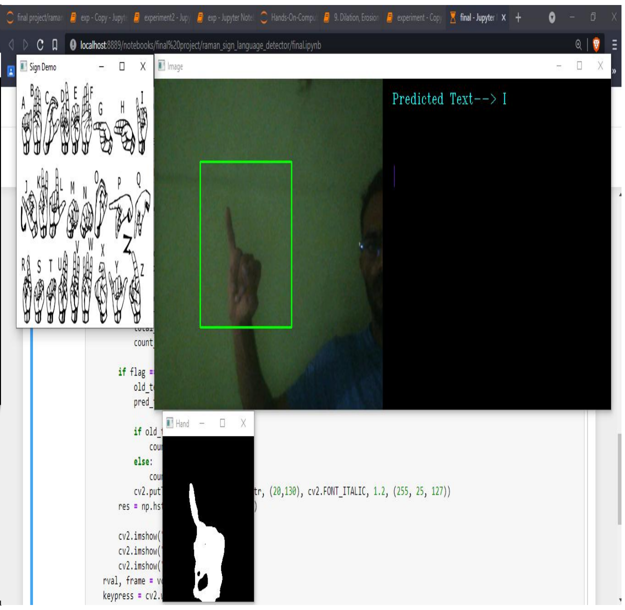


Fig 3.3 ASL sign for I

The prediction layer estimates how likely the image will fall under one of the classes. So the output is normalized between 0 and 1 and such that the sum of each values in each class sums to 1. We have achieved this using softmax function.

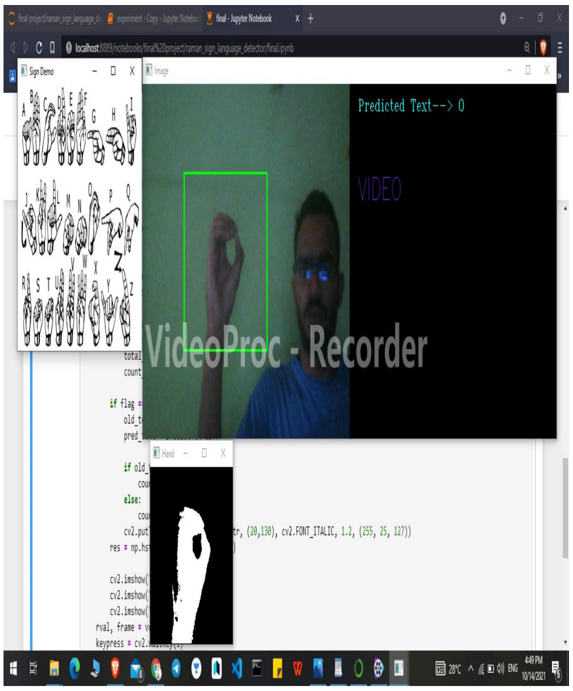
Chapter 4

Results and Discussions

We have achieved an accuracy of **99.83%** in our model using only 3 layer Convolutional Neural Network(CNN) for the train and test data, and an accuracy score of **79.854%** on real world unseen data, which is a better accuracy then most of the current research papers on American sign language. Most of the research papers focus on using devices like kinect for hand detection. In [7] they build a recognition system for Flemish sign language using convolutional neural networks and kinect and achieve an error rate of 2.5%. In [8] a recognition model is built using hidden Markov model classifier and a vocabulary of 30 words and they achieve an error rate of 10.90%. In [9] they achieve an average accuracy of 86% for 41 static gestures in Japanese sign language. Using depth sensors map [10] achieved an accuracy of 99.99% for observed signers and 83.58% and 85.49% for new signers. They also used CNN for their recognition system.

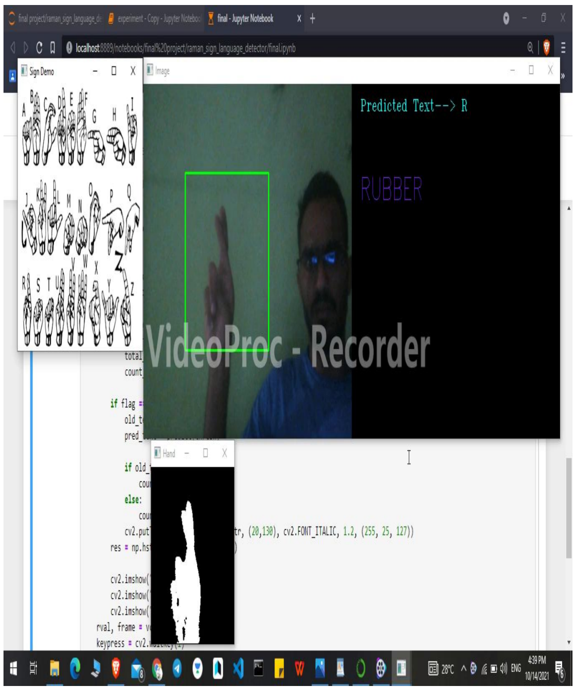
One thing should be noted that my model doesn't uses any background subtraction algorithm whiles some of the models present above do that. So once we try to implement background subtraction in our project the accuracy may vary. On the other hand most of the above 21 projects use kinect devices but our main aim was to create a project which can be used with readily available resources. A sensor like kinect not only isn't readily available but also is expensive for most of audience to buy and our model uses a normal webcam of the laptop or an IP camera of phone hence it is great plus point.

We have achieved success in building a neural network for recognizing sign language characters and forming words .Some of the results are as:



```
        total
        count

    if flag =
        old_t
        pred_

        if old_
            cou
        else:
            cou
        cv2.put                     tr, (20,130), cv2.FONT_ITALIC, 1.2, (255, 25, 127))
    res = np.hs                     )

    cv2.imshow(
    cv2.imshow(
    cv2.imshow(
rval, frame = v
keypress = cv2.
```

Fig 4.1 Recognizing the Word VIDEO

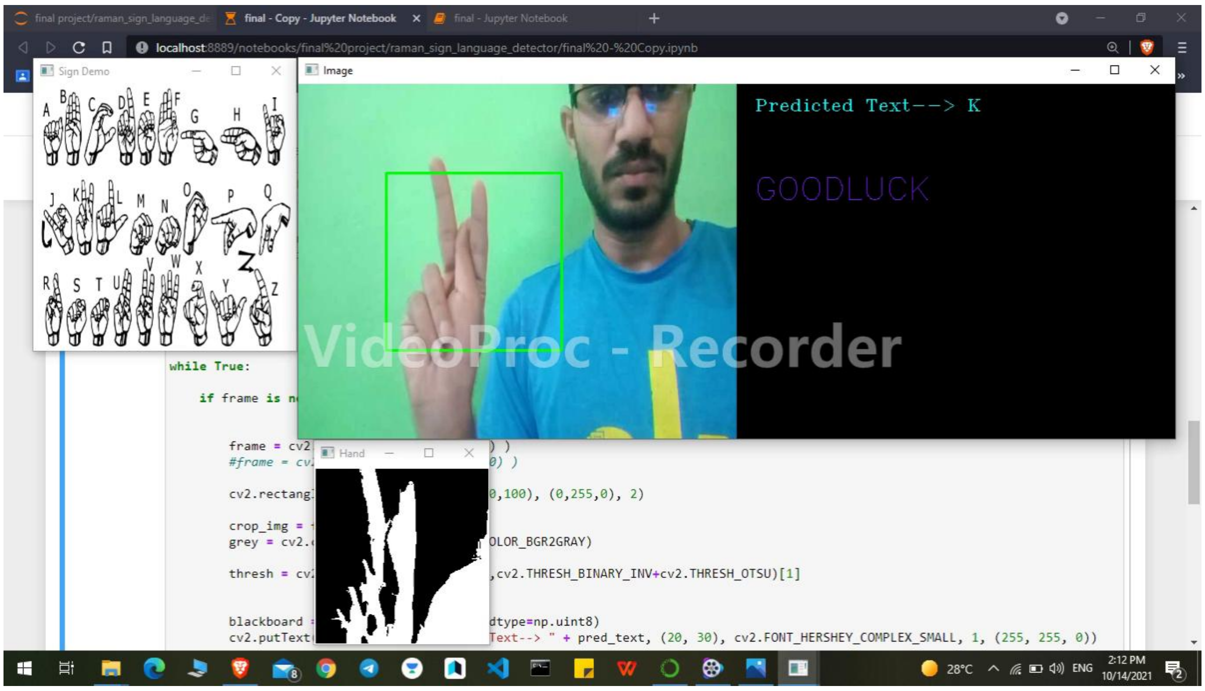


```
        total
        count

    if flag =
        old_t
        pred_

        if old_
            cou
        else:
            cou
        cv2.put                     r, (20,130), cv2.FONT_ITALIC, 1.2, (255, 25, 127))
    res = np.hs                     )

    cv2.imshow(
    cv2.imshow(
    cv2.imshow(
rval, frame = v
keypress = cv2.
```

Fig 4.2 Recognizing the Word RUBBER

We also have tried with various backgrounds :

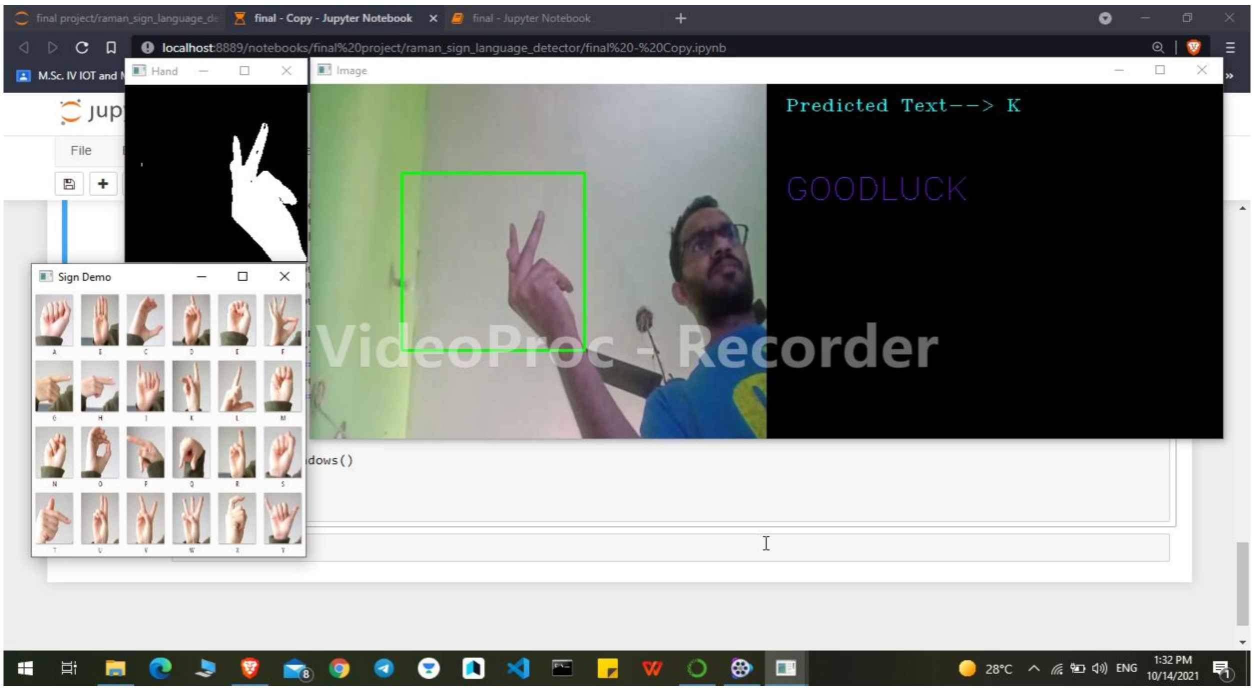
1. **With wall as Background using Web Cam**



```
while True:

    if frame is n

        frame = cv2                     ) )
        #frame = cv                     0) )

        cv2.rectang                     0,100), (0,255,0), 2)

        crop_img =
        grey = cv2.                     OLOR_BGR2GRAY)

        thresh = cv                     ,cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)[1]

        blackboard =                    dtype=np.uint8)
        cv2.putText                     Text--> " + pred_text, (20, 30), cv2.FONT_HERSHEY_COMPLEX_SMALL, 1, (255, 255, 0))
```

Fig 4.3 Recognizing Word GOODLUCK with wall as BG

2. **With roof as Background using IP cam**

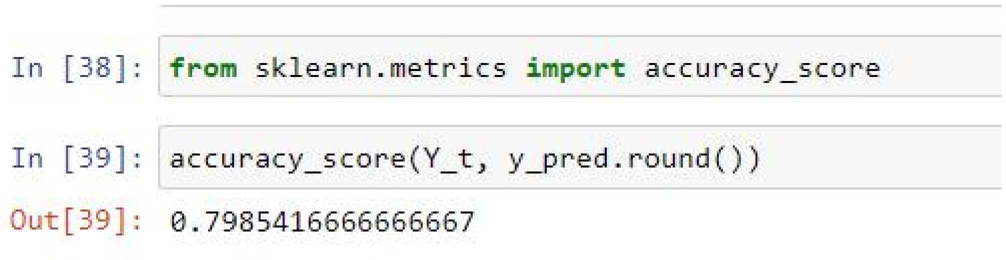


```
Predicted Text--> K

GOODLUCK
```

Hand
Image
Sign Demo
VideoProc - Recorder
localhost:8889/notebooks/final%20project/raman_sign_language_detector/final%20-%20Copy.ipynb
final - Copy - Jupyter Notebook
final - Jupyter Notebook

Fig 4.4 Recognizing Word GOODLUCK with roof as BG

Accuracy score can be maximize using more layers and more accurate data.

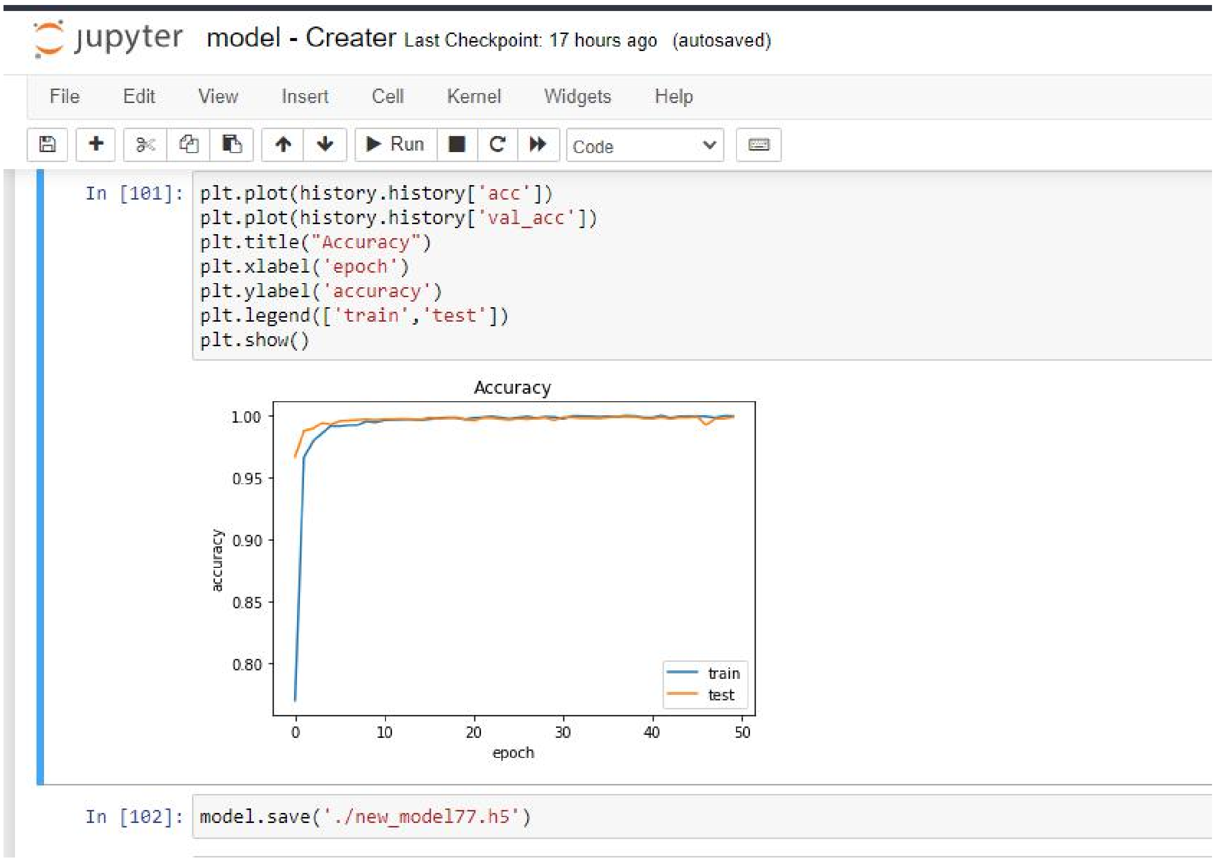


```
In [38]: from sklearn.metrics import accuracy_score

In [39]: accuracy_score(Y_t, y_pred.round())
Out[39]: 0.7985416666666667
```

Fig 4.4 Accuracy score of model on real world data

Accuracy chart :

This chart shows the relation between accuracy and Epoch applied,as number of Epoch increases Accuracy also increases,and less data is lost .



```
In [101]: plt.plot(history.history['acc'])
          plt.plot(history.history['val_acc'])
          plt.title("Accuracy")
          plt.xlabel('epoch')
          plt.ylabel('accuracy')
          plt.legend(['train','test'])
          plt.show()
```

```
In [102]: model.save('./new_model77.h5')
```

Fig 4.5 Accuracy chart of model

Here we can see that we have saved the model for future uses. And we do not have the need to train it again and again.

Chapter 5

Challenges Faced

There were many challenges faced during the project. The very first issue we faced was of dataset. We wanted to deal with raw images and that too square images as CNN in Keras as it was a lot more convenient working with only square images. Second issue was to select a filter which we could apply on our images so that proper features of the images could be obtained and hence then we could provided that image as input for CNN model. We tried various filter including binary threshold,adaptive threshold, canny edge detection, Gaussian blur etc. but finally we settled with Gaussian blur filter. More issues were faced relating to the accuracy of the model we trained.We have to deal with the background problem,since we have used CNN so background plays an important role.On wall as a BG we can achieve good Accuracy compare to other BG.Other challenge we faced is related to motion of hand required for J and Z ,since dataset we used is in .CSV format we do not have data for these two alphabets.The most challenging thing is the hardware and system requirements.For training the model we require an advance computer system with more power,as it can damage the normal PC that I used. We have problem with some alphabets which has the similar confusing sign in ASL, as M,N,E and A,S,T .These two groups of alphabets have same confusing signs which create problem for our model to predict the output.

Chapter 6

Summary and Conclusions

In this report, a functional real time vision based American sign language(ASL) recognition model for D&M people have been developed for ASL alphabets. We have achieved an accuracy of **99.83%** for the train and test data,and achieved final accuracy of **79.854%** on real world unseen data. We can add more layers to improve our prediction in which we can verify and predict symbols which are more similar to each other. Doing so we can detect almost all the symbols provided that they are shown properly, there is no noise in the background and lighting is adequate.

Chapter 7

Future Scope

We are planning to achieve higher accuracy even in case of complex backgrounds by trying out various background subtraction algorithms. We are also thinking of improving the preprocessing to predict gestures in low light conditions with a higher accuracy. We are also planing to train model on our own dataset for better results ,as data provided by us will match more with real world data and hence accuracy can be improved.we can also made modifications so that we can make words with these alphabets and also predicts the similar words. We can also make a model that translate the given words into any given language,thus can act as a translator.Model can be improvised to convert text into speech that can be very helpful for mute community.

Chapter 8

Appendix

Key Words and Definitions

➢ Feature Extraction and Representation :

The representation of an image as a 3D matrix having dimension as of height and width of the image and the value of each pixel as depth (1 in case of Grayscale and 3 in case of RGB). Further, these pixel values are used for extracting useful features using CNN.

➢ Artificial Neural Networks(ANN) :

Artificial Neural Network is a connections of neurons, replicating the structure of human brain. Each connection of neuron transfers information to another neuron. Inputs are fed into first layer of neurons which processes it and transfers to another layer of neurons called as hidden layers. After processing of information through multiple layers of hidden layers, information is passed to final output layer.

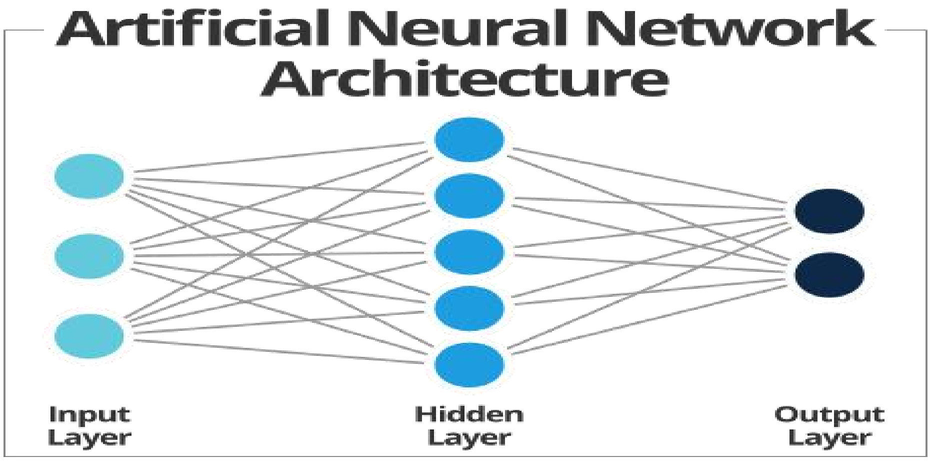


Fig 8.1 ANN Architecture

➢ Convolution Neural Network(CNN) :

Unlike regular Neural Networks, in the layers of CNN [15], the neurons are arranged in 3 dimensions: width, height, depth. The neurons in a layer will only be connected to a small region of the layer (window size) before it, instead of all of the neurons in a fully-connected manner. Moreover, the final output layer would have dimensions (number of classes), because by the end of the CNN architecture we will reduce the full image into a single vector of class scores.

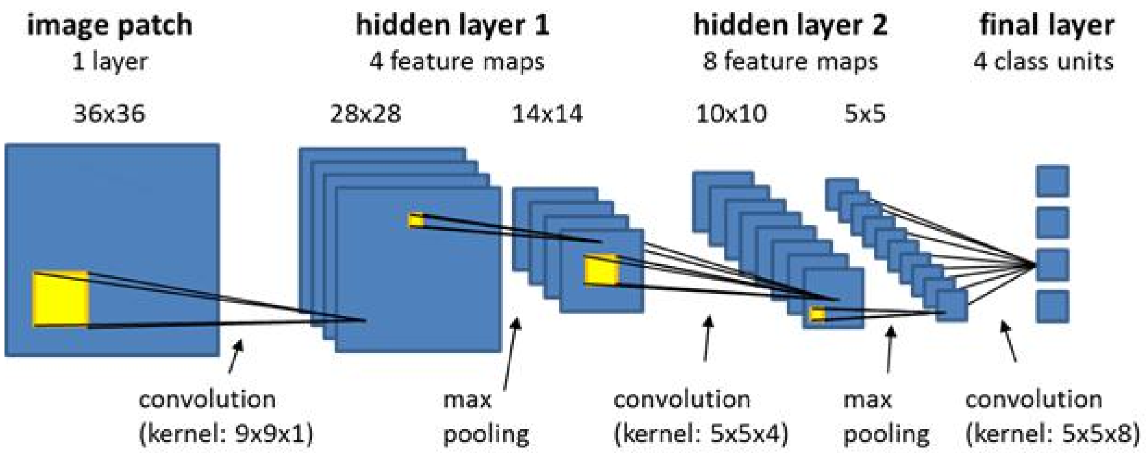


Fig 8.2 CNN Model Architecture

1. Convolution Layer :

In convolution layer we take a small window size [typically of length 5*5] that extends to the depth of the input matrix. The layer consist of learnable filters of window size. During every iteration we slid the window by stride size [typically 1], and compute the dot product of filter entries and input values at a given position. As we continue this process well create a 2-Dimensional activation matrix that gives the response of that matrix at every spatial position. That is, the network will learn filters that activate when they see some type of visual feature such as an edge of some orientation or a blotch of some color

2. **Pooling Layer :**

We use pooling layer to decrease the size of activation matrix and ultimately reduce the learnable parameters. There are two type of pooling :

i. **Max Pooling :** In max pooling we take a window size [for example window of size 2*2], and only take the maximum of 4 values. Well lid this window and continue this process, so well finally get a activation matrix half of its original Size.

ii. **Average Pooling :** In average pooling we take average of all values in a window.

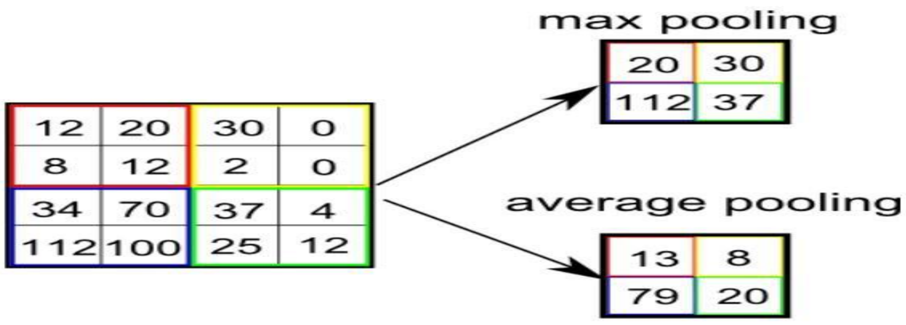


Fig 8.3 Pooling

3. **Fully Connected Layer :**

In convolution layer neurons are connected only to a local region, while in a fully connected region, well connect the all the inputs to neurons.

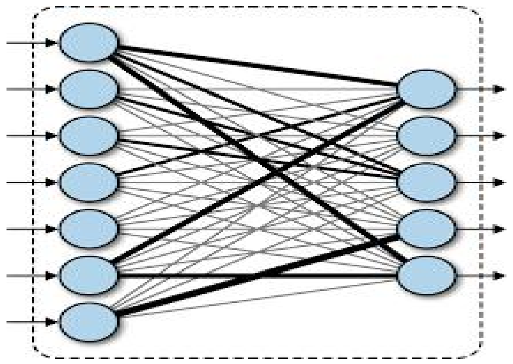


Fig 8.4 Fully Connected Layer

4. **Final Output Layer :**

After getting values from fully connected layer, well connect them to final layer of neurons[having count equal to total number of classes], that will predict the probability of each image to be in different classes.

➢ TensorFlow :

TensorFlow [14] is an end-to-end open-source platform for Machine Learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in Machine Learning and developers easily build and deploy Machine Learning powered applications.

TensorFlow offers multiple levels of abstraction so you can choose the right one for your needs. Build and train models by using the high-level Keras API, which makes getting started with TensorFlow and machine learning easy.

If you need more flexibility, eager execution allows for immediate iteration and intuitive debugging. For large ML training tasks, use the Distribution Strategy API for distributed training on different hardware configurations without changing the model definition.

➢ OpenCV :

OpenCV(Open Source Computer Vision) is an open source library of programming functions used for real-time computer-vision[11]. It is mainly used for image processing, video capture and analysis for features like face and object recognition.

Chapter 9

References

[1] T. Yang, Y. Xu, and “A. , Hidden Markov Model for Gesture Recognition”, CMU-RI-TR-94 10, Robotics Institute, Carnegie Mellon Univ.,Pittsburgh,PA, May 1994.

[2] Pujan Ziaie, Thomas M uller , Mary Ellen Foster , and Alois Knoll“A Na ıve Bayes Munich,Dept. of Informatics VI, Robotics and Embedded Systems,Boltzmannstr. 3, DE-85748 Garching, Germany.

[3] https://docs.opencv.org/2.4/doc/tutorials/imgproc/gausian_median_blur_bilateral _filter/gausian_median_blur_bilateral_filter.html

[4] Mohammed Waleed Kalous, Machine recognition of Auslan signs using PowerGloves: Towards large-lexicon recognition of sign language.

[5] aeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convol utional-Neural-Networks-Part-2/

[6] http://www-i6.informatik.rwth-aachen.de/~dreuw/database.php

[7] Pigou L., Dieleman S., Kindermans PJ., Schrauwen B. (2015) Sign Language Recognition Using Convolutional Neural Networks. In: Agapito L., Bronstein M., Rother C. (eds) Computer Vision - ECCV 2014 Workshops. ECCV 2014. Lecture Notes in Computer Science, vol 8925. Springer, Cham

[8] Zaki, M.M., Shaheen, S.I.: Sign language recognition using a combination of new vision based features. Pattern Recognition Letters 32(4), 572–577 (2011) 25

[9] N. Mukai, N. Harada and Y. Chang, "Japanese Fingerspelling Recognition Based on Classification Tree and Machine Learning," 2017 Nicograph International (NicoInt), Kyoto, Japan, 2017, pp. 19-24. doi:10.1109/NICOInt.2017.9

[10] Byeongkeun Kang , Subarna Tripathi , Truong Q. Nguyen ”Real-time sign language fingerspelling recognition using convolutional neural networks from depth map” 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)

[11] https://opencv.org/

[12] https://en.wikipedia.org/wiki/TensorFlow

[13] https://en.wikipedia.org/wiki/Convolutional_neural_network

[14] https://www.tensorflow.org/

[15] https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[16] https://www.ripublication.com/ijaer18/ijaerv13n9_90.pdf