Capstone Proposal
Udacity Machine Learning Nanodegree

Mohamed Gamal

August 2019

Definition

Project Overview

According to the World Health Organization, over 6% of the world's population (i.e. around 455 million people: 385 millions of deaf, 70 millions of mute have disabling hearing loss or speech problems, the majority of those people live in low and middle-income countries, and in Egypt alone there are around 7.5 million.

A person who is not able to hear as well as someone with normal hearing –thresholds of 25 dB or better in both ears – is said to have hearing loss. Hearing loss may be mild, moderate, severe or profound. It can affect one ear or both ears, and leads to difficulty in hearing conversational speech or loud sounds. 'Deaf' people mostly have profound hearing loss, which implies very little or no hearing. They often use sign language for Communication. So, this project is considered as a contribution to solve the problem of deaf-mute people, where deaf-mute is term continues to be used to refer to deaf people who cannot speak an oral language or have some degree of speaking ability, such people communicate using sign Language.

Problem Statement

The main objective of this project is to develop an automatic system used in recognizing the sign language of the hand that means much for the speechless and deaf persons.
The proposed system will make it easier to communicate with deaf persons by converting their sign language to text or sound. The project has the following specific aims:
1. Low cost.
2. High accuracy.
3. Easy to use.
4. No physical devices attached to deaf-mute people.
5. Ease of communications between deaf-mute and other people using the developed system.

Metrics

The evaluation metric for this problem is simply the Accuracy Score. the model's accuracy produced after the end of training is around 92%. And the accuracy produced after using the testing data set is around 78%, which presents a valid assessment about the process of training and that the model results are reliable.

Analysis

Data Exploration and Visualization

The data set consists of 44 different gestures which are:

- Letters e.g(A,B,C,...).
- Numbers e.g (0,1,2,...,9).
- Words e.g (love ,peace,like,...).

A set of 1200 images per gesture (50 x 50 pixel each) are collected using a computer camera.The process of collecting data for every gesture is based on capturing images by the camera during streaming video, in a span of 10-15 seconds(shown in **Fig**. 1).This dataset will be split into 80:20 for training(80%)
and testing(20%).

Figure 1 A sample gesture of each letter.

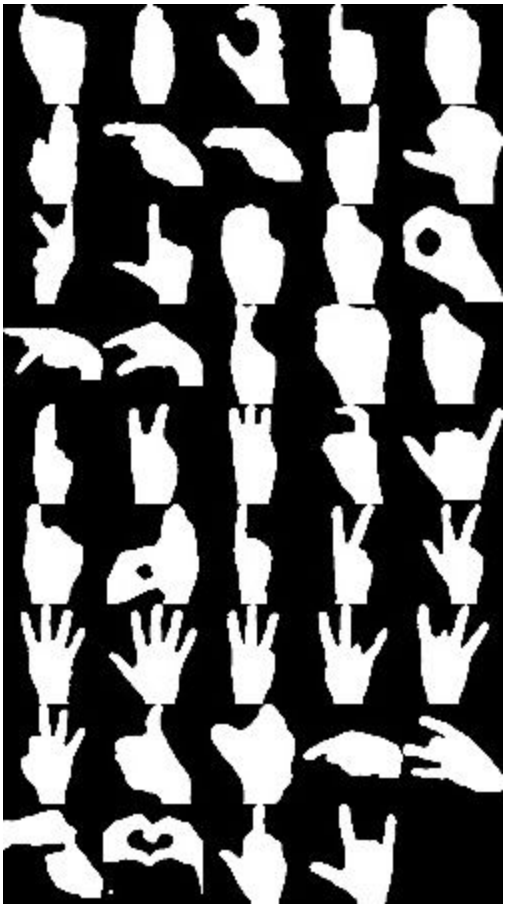


Fig 2 :A sample gesture of each letter in dataset

We can explore here Data conditioning This step deals with extracting important features that represent important characteristics of the gesture throughout the video and then storing these features (shown in **Fig.** 3). Our Objective in this step is to find features that represent shape, size, reflectivity and Other important properties. We want features that are generative and not discriminative as this will allow multiple gestures to be recognized with limited features.

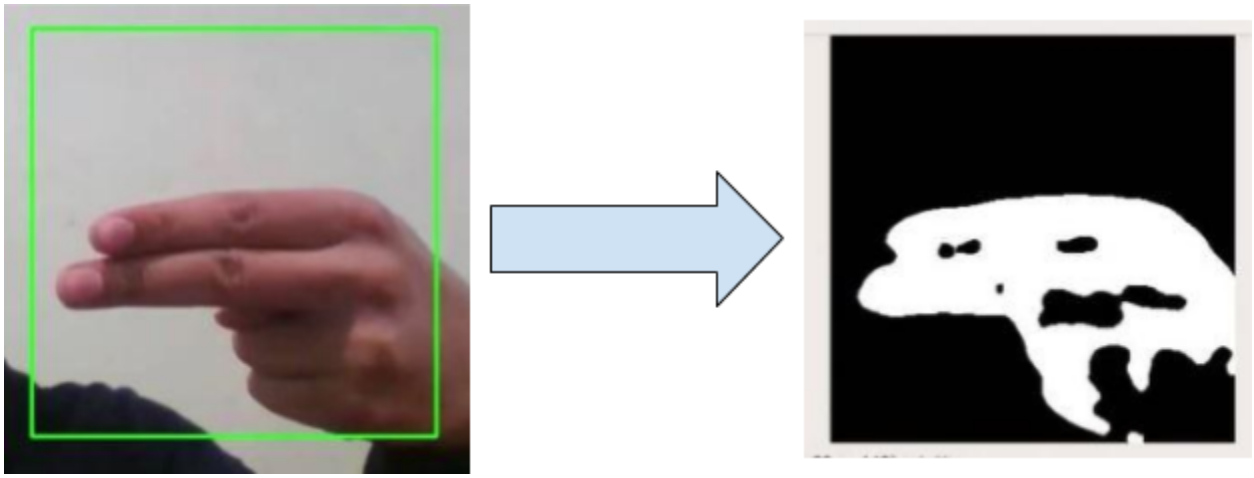


Figure 3 (left) original gesture image, (right) processed image based on the extracted feature

Data augmentation is a strategy that enables practitioners to significantly increase the diversity of data available for training models, without actually collecting new data. Data augmentation techniques such as cropping, padding, and horizontal flipping are commonly used to train large neural networks.

The strategy we went for is horizontal flipping, because it is more suitable for the type of data we have. This will double the number of images we have in the data set and will add the benefit of accounting for both left and right hand (shown in **Fig.** 4).

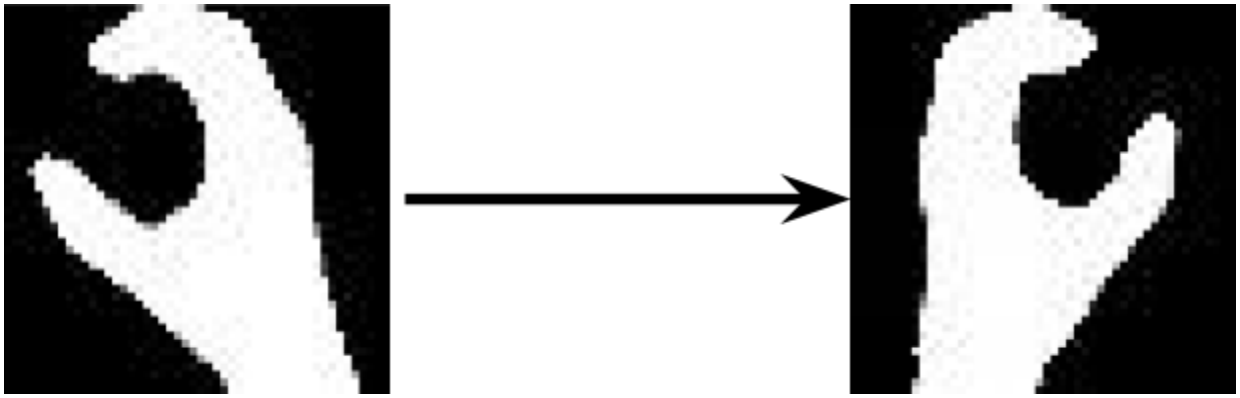


Figure 4 Each image in the data set is flipped.

Algorithms and Techniques

Our ASL letter classification is done using a convolutional neural network (CNN or ConvNet). CNNs are machine learning algorithms that have seen incredible success in handling a variety of tasks related to processing videos and images. Since 2012, the field has experienced an explosion of growth and applications in image classification, object localization, and object detection. A primary advantage of utilizing such techniques stems from CNNs abilities to learn features as well as the weights corresponding to each feature. Like other machine learning algorithms, CNNs seek to optimize some objective function, specifically the loss function. We utilized a softmax-based loss function:

$$Loss = \frac{1}{N}\sum_{i=1}^{N} -\log\left(\frac{e^{f_{i,y_i}}}{\sum_{j=1}^{C} e^{f_{i,j}}}\right) \quad \textbf{(1)}$$

$$f_j(z) = \frac{e^{z_j}}{\sum_{k=1}^{C} e^{z_k}} \quad \textbf{(2)}$$

$$\mathrm{N} = \text{total number of training examples}$$
$$\mathrm{C} = \text{total number of classes}$$

Equation (2) is the softmax function. It takes a feature vector z for a given training example, and squashes its values to a vector of [0,1]-valued real numbers summing to 1. Equation (1) takes the mean loss for each training

example, xi, to produce the full softmax loss. Using a softmax-based classification head allows us to output values akin to probabilities for each ASL letter. This differs from another popular choice: the SVM loss.
Using an SVM classification head would result in scores for each ASL letter that would not directly map to probabilities. These probabilities afforded to us by the softmax loss allow us to more intuitively interpret our results and prove useful when running our classifications through a language model.

Transfer Learning is a machine learning technique where models are trained on (usually) larger data sets and refactored to fit more specific or niche data. This is done by recycling a portion of the weights from the pre-trained model and reinitializing or otherwise altering weights at shallower layers. The most basic example of this would be a fully trained network whose final classification layer weights have been reinitialized to be able to classify some new set of data. The primary benefits of such a technique are its less demanding time and data requirements. However, the challenge in transfer learning stems from the differences between the original data used to train and the new data being classified. Larger differences in these data sets often require re-initializing or increasing learning rates for deeper layers in the net.

TensorFlow We employed tensorflow, a deep learning framework, in order to develop, test, and run our CNNs.TensorFlow 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 ML and developers easily build and deploy ML powered applications.

Keras we have another option it is keras. We provide two library in code and can choose one of both.Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation.

Being able to go from idea to result with the least possible delay is key to doing good research.

Benchmark

Choosing a suitable network model After collecting the needed dataset and finishing its preprocessing, we are now ready to construct our model. Since we are dealing with images, the neural network we are going to build will be a convolutional neural network.
The proposed convolutional neural network has to be relatively small in order to perform calculations in a real-time fashion, hence we will build a network that is similar to one of the MNIST classifying models, and consists of only 9 layers.

Methodology

Data Preprocessing

The objective of the Data preprocessing step is to separate the hand from the background. We are going to use multiple image processing and filtering techniques in order to get a binary image where the hand gesture is shown in white, and all its background is shown in black.

The techniques and methods used during preprocessing are shown in the flowchart in **Fig. 5**. We will discuss each step separately and explain their importance for our goal.

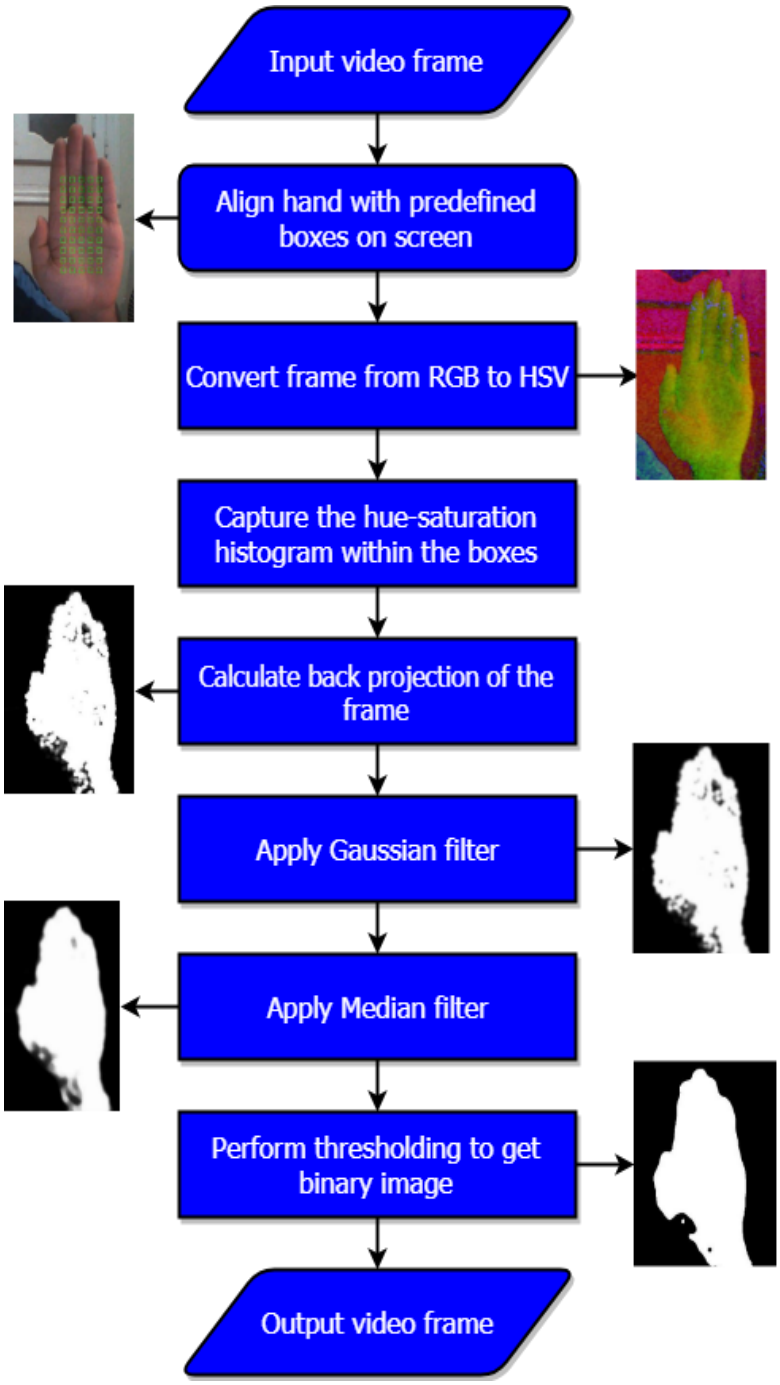


Figure 5 Preprocessing flowchart

Align hand with boxes

In order to get the correct coloring of the hand, it must be aligned with a specific boxes on screens. These boxes are determined beforehand and are shown on screen as green boxes (shown in **Fig. 6**) that must cover most of the hand to get all the range of its coloring.

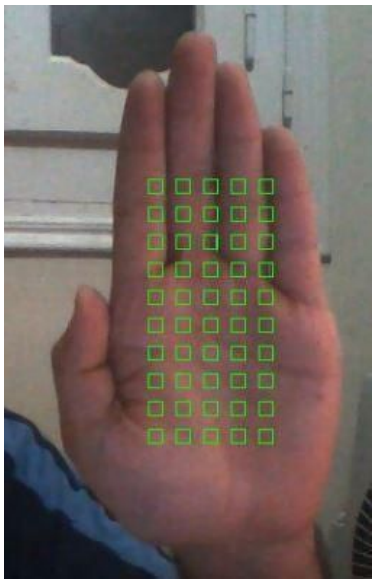


Figure 6 50 green boxes shown on screen And are covering most of the hand.

Convert frame colors from RGB to HSV

Color vision can be processed using RGB color space or HSV color space(**Fig** 7). RGB color space describes colors in terms of the amount of red, green, and blue present. HSV color space describes colors in terms of Hue, Saturation, and Value. In situations where color description plays an integral role, the HSV color model is often preferred over the RGB model.

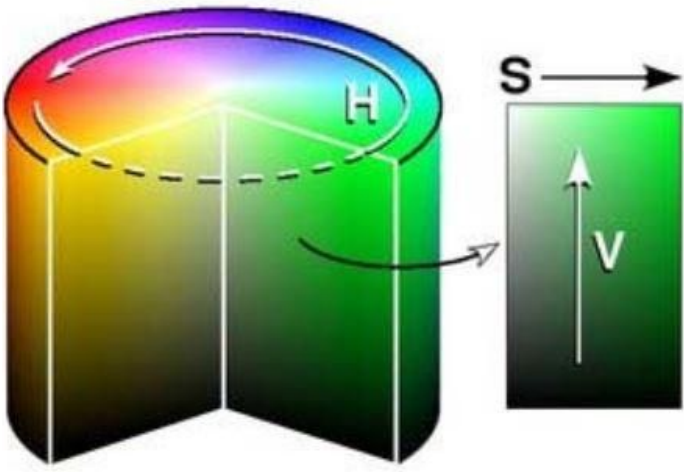


Figure 7 HSV color wheel.

The HSV model describes colors similarly to how the human eye tends to perceive color. RGB defines color in terms of a combination of primary colors, where as, HSV describes color using more familiar comparisons such as color, vibrancy and brightness.

The main reason of converting the frame from RGB to HSV is to obtain a better representation of the colors of the hand(shown in **Fig** 8), which facilitates the process of separating it from the background and surroundings.

Figure 8 Result of converting the image from RGB to HSV.

Hue-saturation histogram

Hue and saturation channel can provide a very good representation of any color. We collect both channels from all pixels that reside inside the small boxes we showed before in **Fig** 6, and store them in a histogram that should look similar to **Fig** 9.

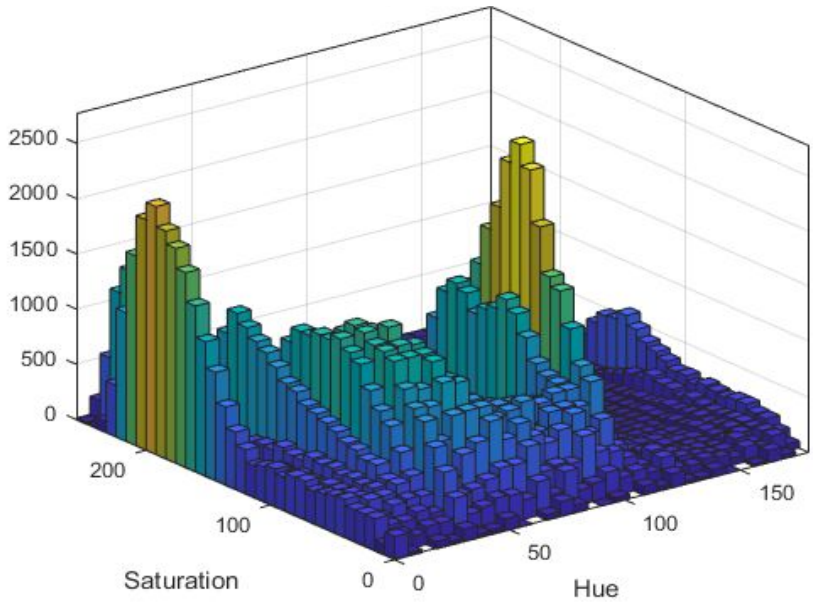


Figure 9 Hue-saturation histogram.

Calculate the Back Projection of frame

Back Projection is a way of recording how well the pixels of a given image fit the distribution of pixels in a histogram model. To make it simpler: For Back Projection, you calculate the histogram model of a feature and then use it to find this feature in an image.

This gives pixels values between 0 and 1 based on their density in the histogram (i.e. pixels that appear the most are set to 1 or are completely white, and pixels that appear the least are set to 0 or are completely black). Since the histogram contain color information from the hand only, it will be mostly white on a black background as shown in **Fig 10**.

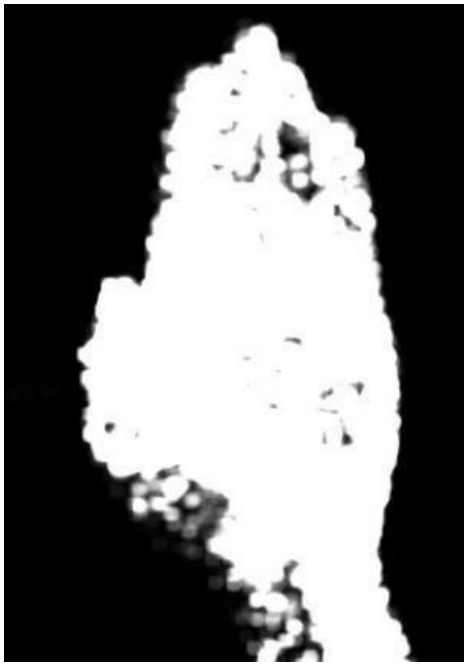


Figure 10 Back projection of the hand gesture.

Apply Gaussian filter

Gaussian filter is a linear smoothing filter. The effect of Gaussian filter is to blur an image and remove details and noise, in a similar fashion to the mean filter, but different in that it performs a weighted average of pixels and gives the largest weight to the center pixel.

The Gaussian filter is preferred over the mean filter because it performs the same smoothing effect, but preserves edges better than a similarly sized mean filter. The main parameter that controls the smoothing

effect in Gaussian filters is its standard deviation which can either be selected manually, or calculated based on the kernel size we choose.

The reason behind applying Gaussian filter during preprocessing is to smoothen the image and reduce black spots that are inside the bounds of the hand. The chosen kernel size is (11×11) and the result is shown in **Fig 11**.

Figure 11 Example of a Gaussian filtered image.

Apply Median filter

The Median Filter is a non-linear digital filtering technique and is a typical preprocessing step to improve the results of later processing. Median filtering is very widely used in digital image processing because, under certain conditions, it preserves edges while removing noise.

The main idea of the median filter is to run through the image pixel by pixel, replacing each pixel with the median of neighboring pixels. This further removes the black spots that are inside the hand boundaries and prepares the image for thresholding. We use a kernel with size 15 and the result is shown in **Fig** 12.

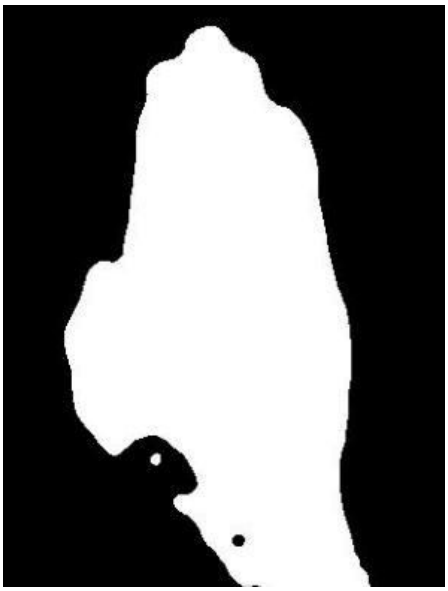
Figure 12 The result of applying a Median filter

Thresholding the frame

The final step in preprocessing is thresholding. Thresholding converts the image from grayscale to binary (shown in **Fig below**) based on a threshold that can be set manually or automatically.

We calculate the threshold automatically using a method called **Otsu Thresholding**. Otsu's thresholding method involves iterating through all the possible threshold values and calculating the variance for the pixel levels each side of the threshold, i.e. the pixels that either fall in foreground or background.

The aim is to find the threshold value where the sum of foreground and background variance is at its minimum.



Data Augmentation

As we mention before The strategy we went for is horizontal flipping, because it is more suitable for the type of data we have. This will double the number of images we have in the data set and will add the benefit of accounting for both left and right hand .

Implementation

In this section, the implementation of the system will be explored. We will discuss the proposed framework,the structuring element of the system, and show the output. At the end, we will start analysing the system to measure how good/robust the process of recognition is.

Proposed framework We will start by defining the main structuring blocks of our system. This is shown in **Fig. 13** below.

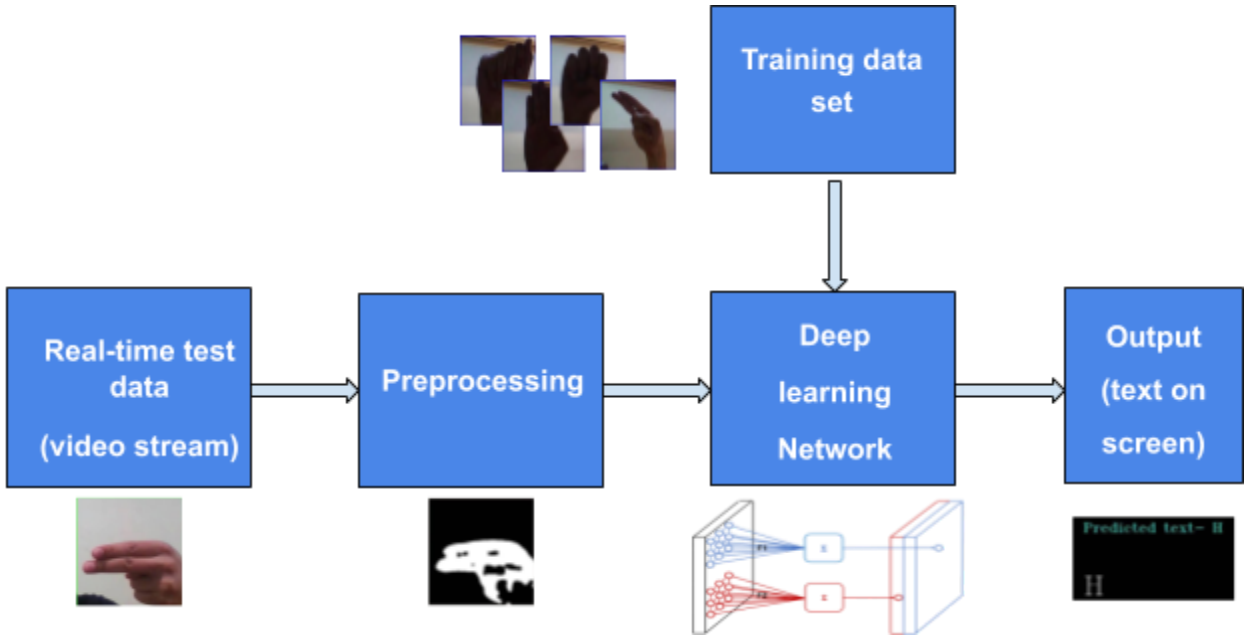


Figure 13 A block diagram of the proposed framework.

Real-time test data The testing data enters the system through a video feed taken by a camera. The video is processed frame-by-frame by our system to identify the gesture shown on the video.
To avoid any unnecessary objects in the background. We defined a certain box shown on screen (**Fig. 3.2**). The processing operation are done only inside that box.

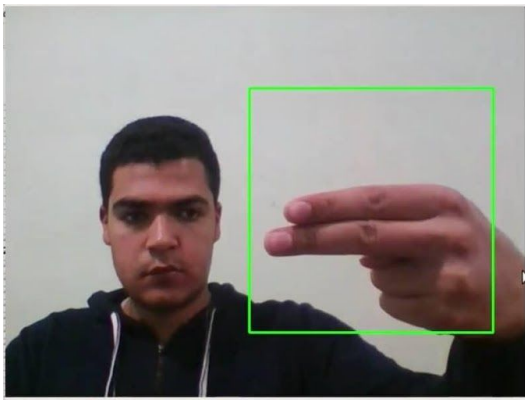


Figure 14: A box contains the portion of frame
That will be processed.

Preprocessing as we mention it above The objective of the preprocessing step is to separate the hand from the background. We are going to use multiple image processing and filtering techniques in order to get a binary image where the hand gesture is shown in white, and all its background is shown in black.

Training data set A set of 2400 image per gesture (50 x 50 pixel each) are collected using a computer camera.

Output text After the input data is processed and classified by the neural network, the gesture will be translated and written as a text on screen.

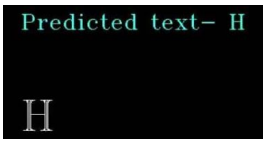


Figure 3.16: the gesture is recognized and translated to text on screen.

Refinement

I train a CNN model with more layers as shown below in (**fig 15**).The loss function that works well with classification problems is the Cross entropy loss function, and the chosen optimization algorithm is stochastic gradient descent.

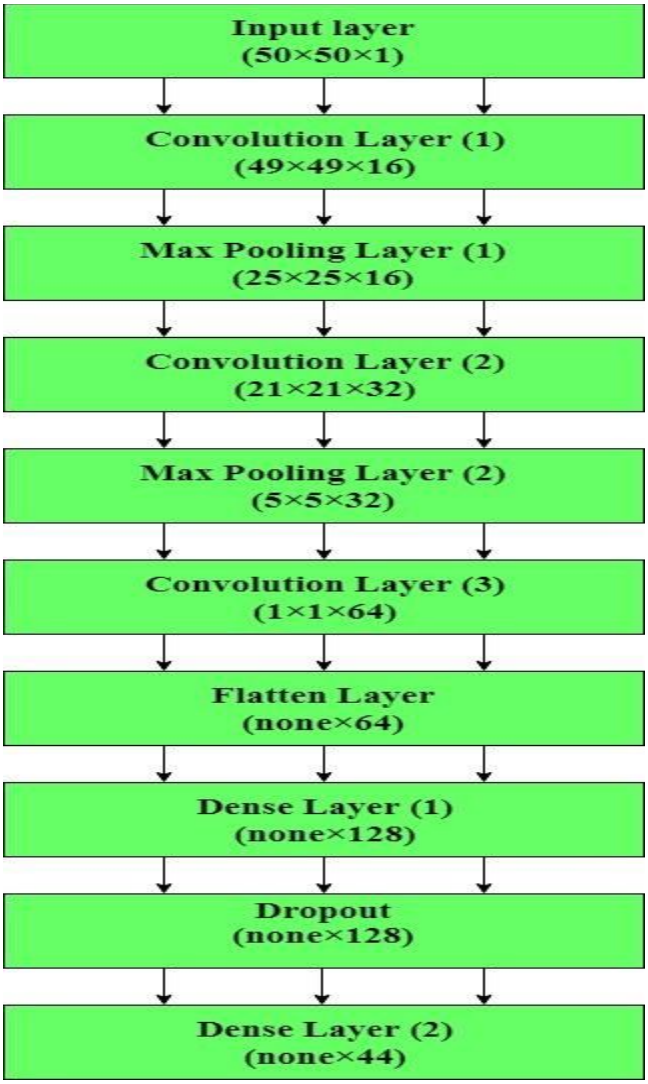


Figure 15: Representation of the proposed network layers.

Results

Model Evaluation and Validation

The data set consists of 44 gestures, each gesture consists of 2400 image, to make a total of 105.600 image. In the beginning, we are going to split the data set into 2 sets: a training set, which contains 88.000 image, and a testing set which contains 17.600 image.

The model is trained for 1 epoch. The loss function that is chosen to assess the model accuracy is Cross Entropy, while the optimization method that is chosen is Stochastic Gradient Descent.

In **Fig** 16, we can see the model accuracy during training at each iteration, while in **Fig** 17, we can see the model accuracy during testing at each iteration.

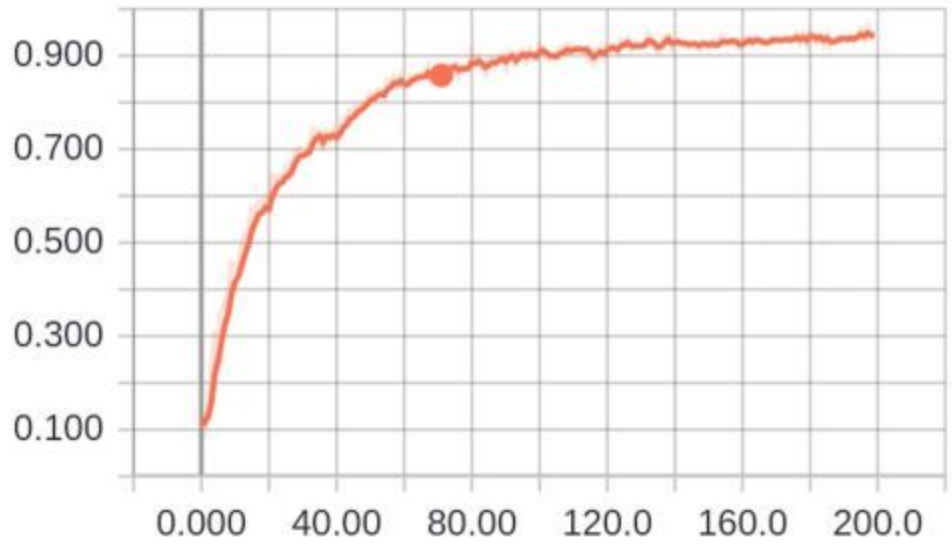


Figure 16 Training accuracy (vertical axis is accuracy, horizontal axis is no. of iterations).

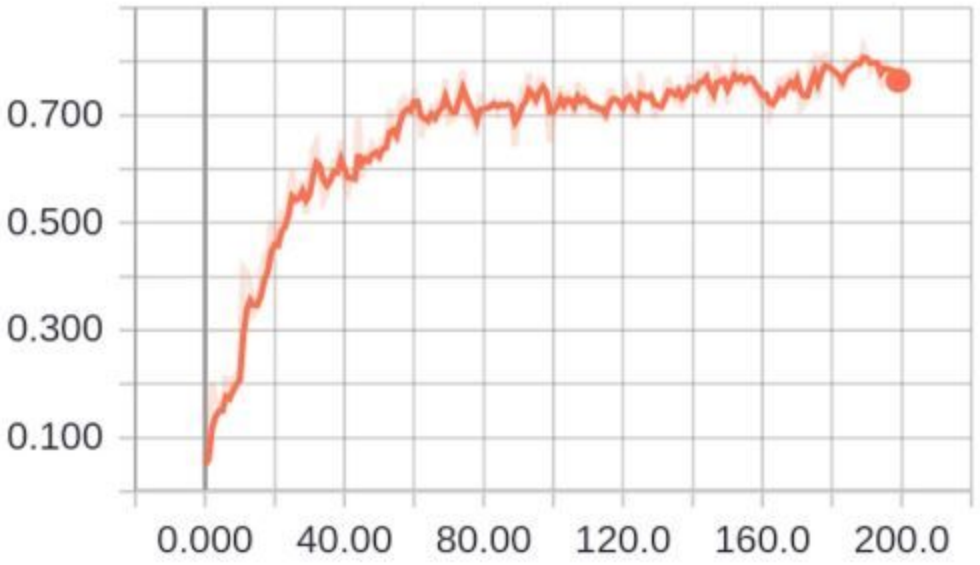


Figure 17 Testing accuracy (vertical axis is accuracy, horizontal axis is no. of iterations).

The model's accuracy produced after the end of training is around 92%. And the accuracy produced after using the testing data set is around 78%, which presents a valid assessment about the process of training and that the model results are reliable.

Justification

The Benchmark model(1st model) has a test accuracy of 85.60% , while the final model has test accuracy of 92%.

-The per-image classification accuracy is higher than 90% .

- The classification delay is about 3 seconds, which is about the same as that of the benchmark.

-The processing delay is around 5 seconds, which is better than that of the benchmark.

So, clearly final model is stronger model.

Conclusion

Free-Form Visualization

A snapshot of the system during recognition can be seen in **Fig** 18.

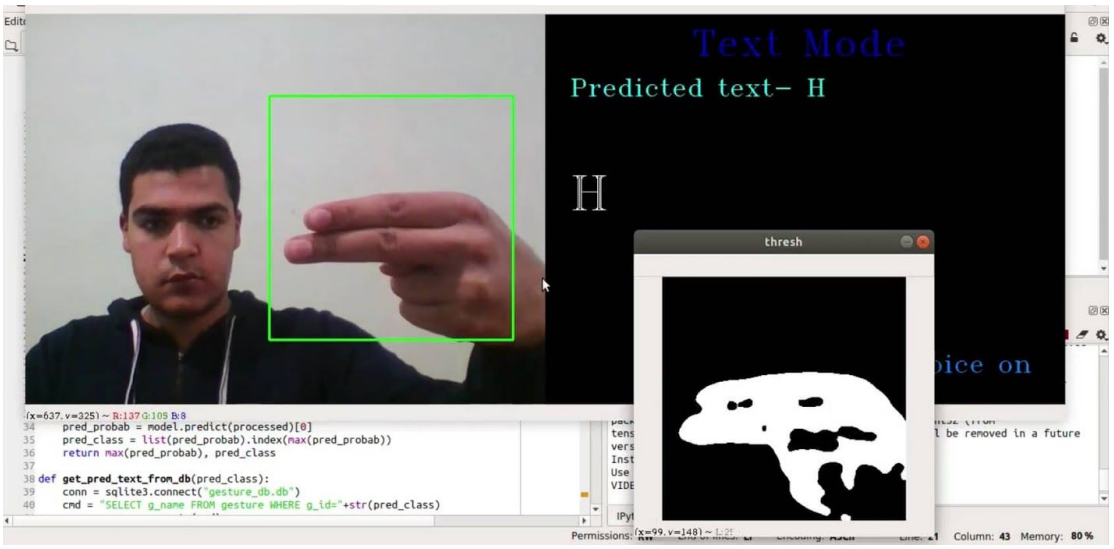


Figure 18 Performing gesture recognition using the trained model.

Reflection

At the start of working on the project, we had to study different methods of image processing, and explore different models in deep learning in order to figure out how to build a robust recognition system. Afterwards, we started building a convolutional network suitable for real-time application and prepared the data to train it. At the end, we

analyzed the system to find out its accuracy and limitations, and proposed some methods that might improve the total performance.

This project provided a great overview of the field of computer vision, deep learning, and image processing, particularly in the areas of hand detection and gesture recognition. Although this system proved successful, many limitations were observed, such as background noise and brightness effect on the performance of the detection.

Improvement

- Explore different techniques to reduce brightness effect on recognition (different models, different preprocessing, new camera, ...etc).
- Improve the system accuracy without affecting its detection speed by using both modern hardware and advanced software.
- Collect data for Arabic sign language and training the system to recognize it.
- Expand the data set to contain more gesture like words.
- Try working on data set that contain videos instead of images to be able to detect gestures that require movement.