Hand Gesture Recognition

Aashish Pandey *Anjan Shrestha* *Nitin Dunday Mohan* *Asher Pranay*

GitHub Link : https://github.com/anjanshrestha123/hand-gesture-recognition

INCREMENT 1:

Motivation:

With the evolution of Convolution Neural Networks, the task of image processing/detection has been more simple and effective. The deep learning techniques implicitly learns the image features and create an end to end model for image classification. However, in this project we are extracting the various image features explicitly using image processing tools and using those features to classify the images. We would like to compare the results of our model with CNN model to understand the difference between the two techniques.

Significance:

The task of gesture recognition has very high significance. It can be used to provide the input to smart devices . One of the most important applications of gesture recognition include sign language translation. Furthermore, gesture recognition has important applications on warning road hazards, creating music, robot remote control etc.

Objectives:

The Main objectives of the project is done by the milestones such as

1. Data collection (Collection the information from the Kaggle data set)
2. Preprocessing Data (Removing all the outliers and getting the features to process the model)
3. Data Model (Apply Machine Learning Model: Feature Engineering like Sharpening the image, using linear and horizontal gradients)
4. Evaluate the Model (Processing the Model information and evaluating using test and train

data.)
5. Final prediction (Final Output is predicted and displayed to the user).

Features:

We will use various techniques to generate the classification features from the images. We will apply ORG keypoints detection to generate the key points. Also we will apply various image enhancement techniques like, background removal to focus more on hand gesture, image sharpening to get clear borders for the fingers. And finally edge detection techniques to detect the edges in the image. Finally we use the relative position of edges to generate the identifying features for each gesture image. The detail of feature generation is explained in the Detail Design of Features section below.

****

Related work (Background):

As hand gesture recognition is very important for human and computer interaction, many researches have been performed in this area. [2] proposed a technique which has six static and eight dynamic hand gestures with 93.09% accuracy. They have used VGG16 pretrained model which is a type of CNN based classifier that consists of 13 convolution layers and 3 fully connected layers.

[3] has provided some glimpse of Deep Learning feature extraction techniques. In these techniques, traditional feature extraction has been replaced by convolution neural networks (CNN) as it has the ability to extract complex features present in the image. Multiple works have been performed on this techniques such as SuperPoint which is a Self-Supervised Interest Point Detection and Description, D2-Net which is a Trainable CNN for Joint Description and Detection of Local Features, LF-Net which is a Learning Local Features from Images, Image Feature Matching Based on Deep Learning and Deep Graphical Feature Learning for the Feature Matching Problem.

[4] performed research on the significance of hand gesture recognition in vehicular automation. They have described two approaches used in gesture detection systems. One of the approaches is accelerometer based approach which works on 18 gestures over 3700 traces from different subjects, and the system achieved almost perfect recognition for user dependent recognition, mixed user and user independent recognition etc. Another approach is glove based approach which consists of active data glove based detection method and passive data glove based method.

Similarly, [5] proposed a system that performs hand gesture recognition with skin detection using deep learning methods. In this system, skin color is detected first and then features are extracted using contour extraction and hand region segmentation and gesture is detected using the pyramidal pooling module and attention mechanism.

Dataset:

The dataset we are using in this project is “Hand Gesture Recognition Dataset” from Kaggle. The dataset is generated by Leap Motion Sensor. It consists of 10 different hand gestures performed by 10 different persons. Each of the gestures are considered as our labels. Hence, we have 10 labels (*Palm, Okay,C, Fist, thumb, index , etc).* We have encoded the labels into integer values 0-9. Finally, for each label/class we have exactly 2000 images. Therefore, our overall dataset has 20,000 images with 2000 belonging to each class.

Detail Design of Features:

Extracting distinct/unique features from the image is the most important step for classification of the images. The images in the same class must have similar features, and the images in the different class must have different features. In order to extract those specific features from our dataset points, we are planning to apply following steps for feature generation:

1. **Foreground Enhancement:** Our training and testing images consist of a hand gesture with some background. Our first step will be to enhance hand image (or background subtraction) such that the features will be generated for hand gestures rather than background noise.

Fig: Original Image

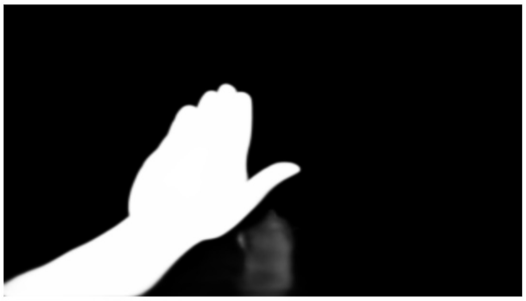


Fig: Gesture enhanced

2. **Keypoint Detection:** In this step. We will generate several key points in the gesture enhanced image using ORB detector and use the key points as one of the features of the image. In the

picture below, we can see the key points in the palm image are distributed differently from key points in the index image. Hence we can use this information to distinct one image from other

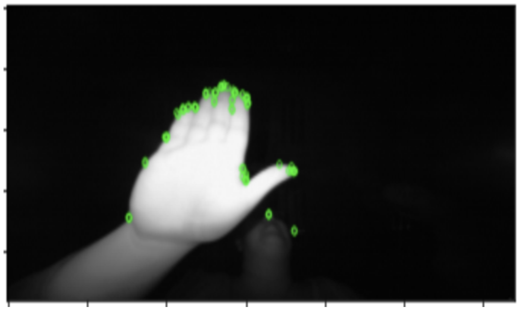


Fig: Distribution of Key Points in palm

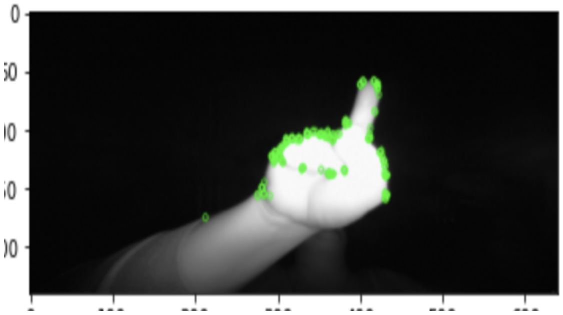


Fig: Distribution of Keypoints in Index

3. **Edge Detection**: We are planning to use the edge information of an image as a feature. The vertical/horizontal edges and their relative position in each gesture are different from each other. Hence, this information will also help us(our classification model) distinguish between the gestures. In order to generate distinct edges, we are planning to use image sharpening in gesture enhanced images.
4. Furthermore, we also implemented **Harris Corner Detection** techniques presented in [6] in an attempt to get a comprehensive feature. We will not be using any features related to color as our dataset consists of grayscale images.

In this way, we will generate distinct features from gesture images and feed these features to our classification model for the prediction.

Analysis:

As part of the data analysis, we looked at the class distribution of the dataset. It is very important to perform/consider the distribution before designing the model. As mentioned in Kaggle documentation, the dataset has perfectly balanced class distribution with each class having exactly 2000 images.

Along with the class distribution, we also looked for the sample data points(images) , their shape and how they look after applying each feature detection technique. Few of the images are presented in the “Detail Design of Features” section above. We are planning to perform some more analysis for our final submission.

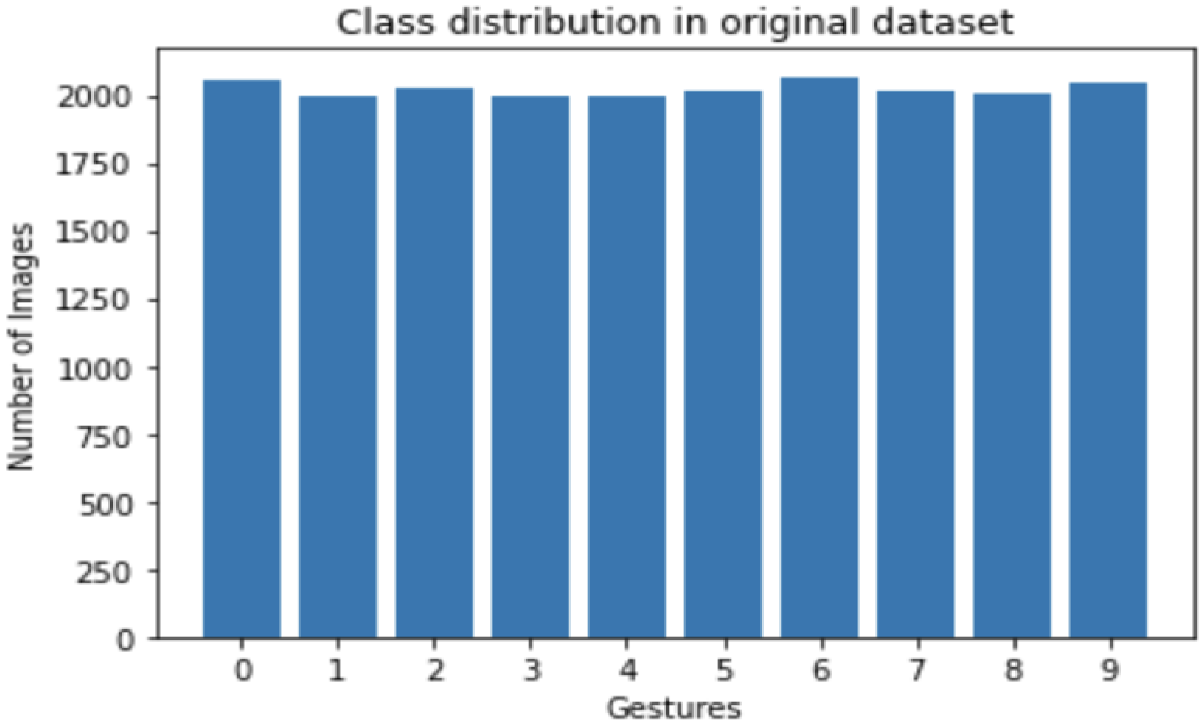


Fig: Class Distribution (Original Dataset)

Implementation:

The diagram below clearly demonstrates the overview of our project. For each image in the dataset, we first try to enhance the gesture by applying background substitution (**This part is not implemented as a part of Increment 1**). We plan to apply the background substitution so that the background noise in the images will be filtered and only the gesture part will be presented for the feature generation. In order to generate the features, we first sharpen the image by subtracting the smooth image from the original image. Next, the sharp image is subjected to various feature generating phases like Vertical/Horizontal Edge detection, ORG keypoint detection, Harris Corner detection etc.

The features generated from those phases are finally integrated and mapped with the true labels associated with the image. The dataset then created is splitted into training and testing dataset with 70% for training and 30% for testing. We finally created a Machine Learning Classifier(Decision Tree for increment 1, we will use a few other classifiers for final increment). The classifier trained with the training dataset will be used for evaluation using the testing dataset.

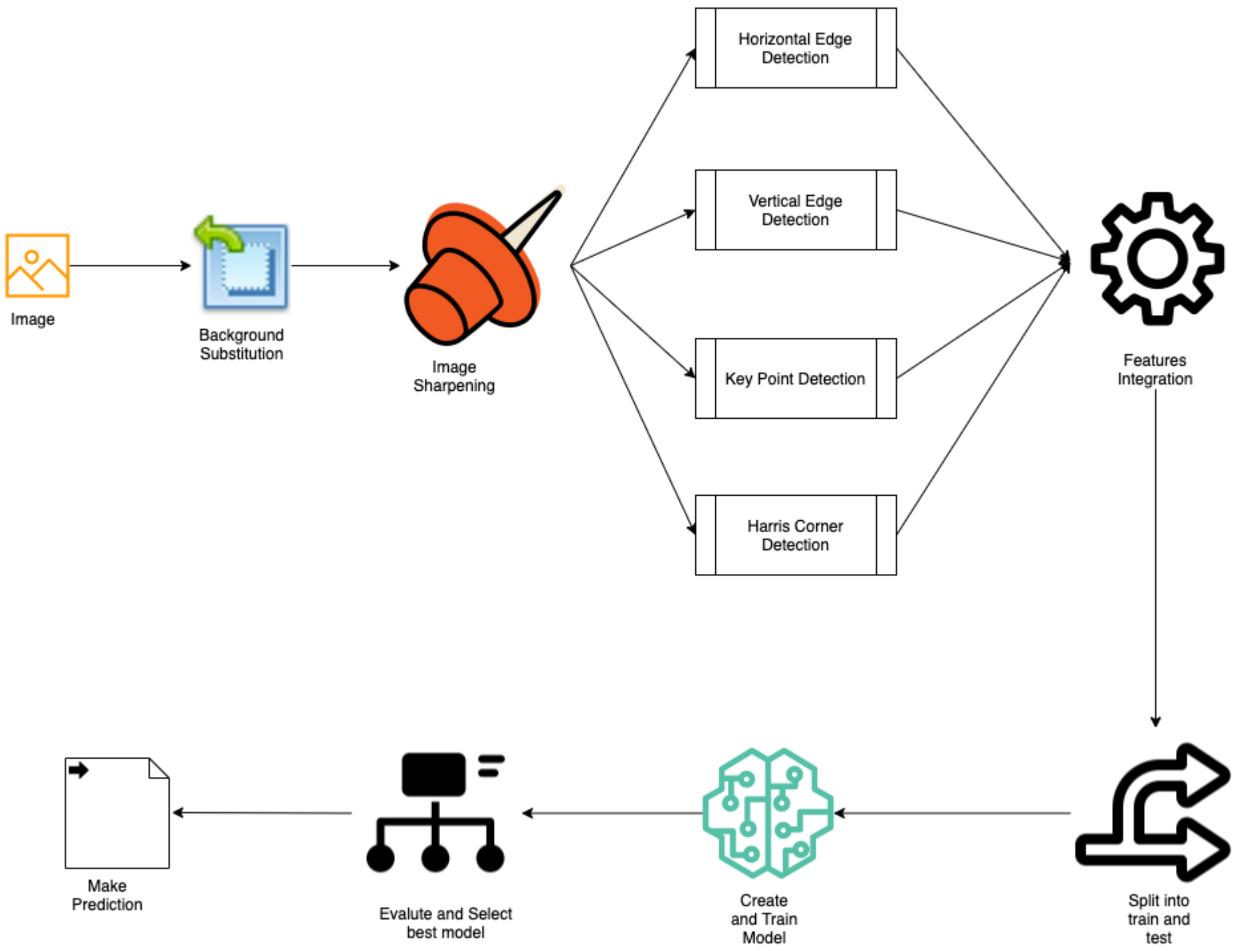
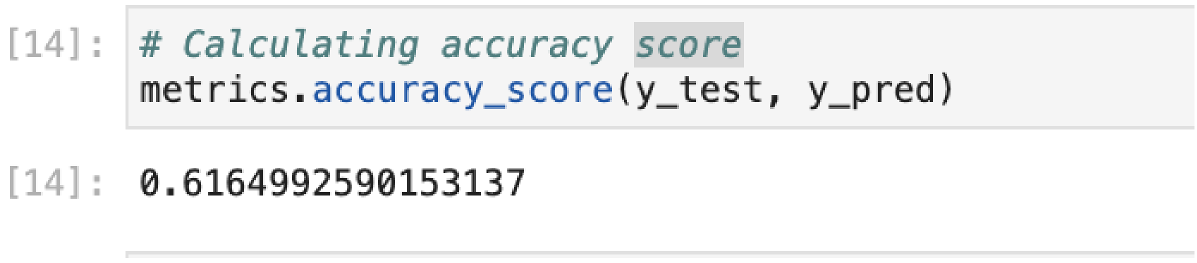


Fig: Implementation Overview

Preliminary Results:

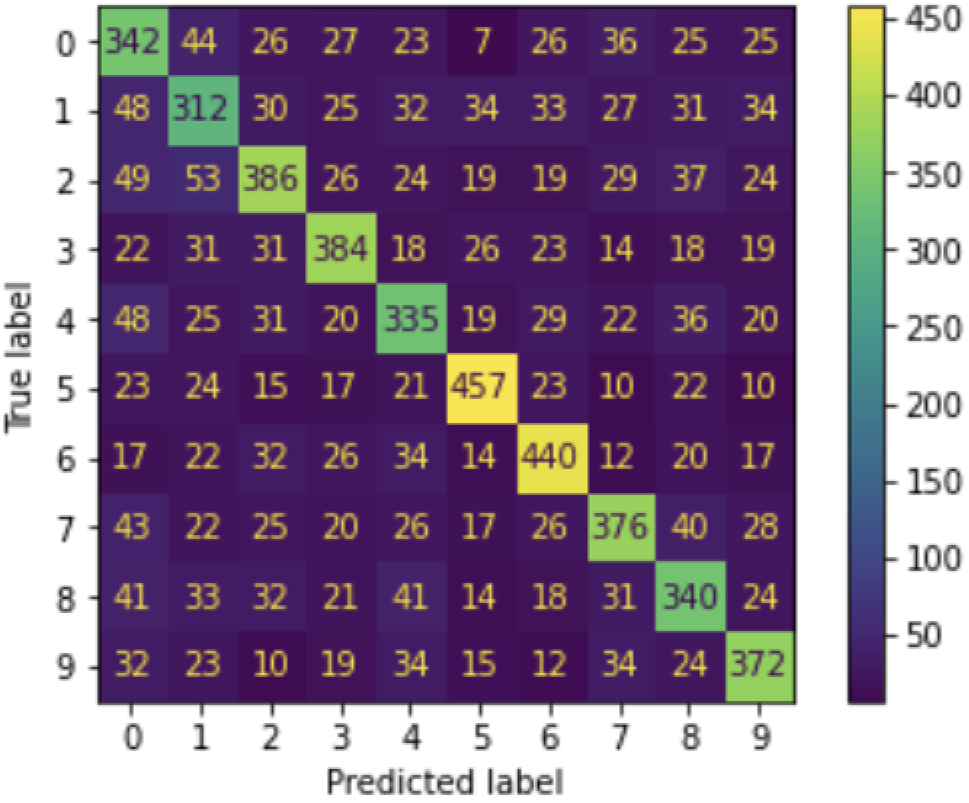
We have not yet implemented several important parts of our projects (like background subtraction and keypoint detection) due to time constraints. However, we were able to generate other important features. We have also created and trained a Machine Learning model(Decision Tree) for this increment. With the preliminary result, we were able to get an accuracy of **0.62.**

The accuracy and confusion matrix of our classification is shown in the images below.



```
[14]: # Calculating accuracy score
      metrics.accuracy_score(y_test, y_pred)

[14]: 0.6164992590153137
```



	0	1	2	3	4	5	6	7	8	9
0	342	44	26	27	23	7	26	36	25	25
1	48	312	30	25	32	34	33	27	31	34
2	49	53	386	26	24	19	19	29	37	24
3	22	31	31	384	18	26	23	14	18	19
4	48	25	31	20	335	19	29	22	36	20
5	23	24	15	17	21	457	23	10	22	10
6	17	22	32	26	34	14	440	12	20	17
7	43	22	25	20	26	17	26	376	40	28
8	41	33	32	21	41	14	18	31	340	24
9	32	23	10	19	34	15	12	34	24	372

Fig: Confusion Matrix

Project Management:

• **Project Management**.

We are having weekly group meetings for the project. We meet at the library every Saturday after the class time and discuss/implement the project. Also, we have created a whatsapp group for instant messages. Each team member posts the blockers as well as important tutorials in the group so each has a clear idea about the status of the project. We are also using GitHub to share our work. We have tried to distribute the workload among the team members. The table below shows the contribution of each team member

o **Implementation status report**

Group members' Contribution

Group member	**Contribution**	**Responsibility(completed)**	**Work to be done**

Aashish Pandey	30%	Project Design and Feature Engineering	Train and Evaluate the model
Anjan Shrestha	30%	Project Design and Related Works	Train and evaluate the model
Nitin Dunday Mohan	25%	Data Extraction and Analysis	Report Writing and Plots Generation
Asher Pranay	15%	Data extract and Analysis	Report Writing and Plots Generation

References/ Bibliography:

1. https://www.kaggle.com/benenharrington/hand-gesture-recognition-database-with-cnn/data
2. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8368821&tag=1
3. https://towardsdatascience.com/image-feature-extraction-traditional-and-deep-learning-techniques-ccc059195d04
4. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.678.7152&rep=rep1&type=pdf
5. https://iopscience.iop.org/article/10.1088/1742-6596/1213/2/022001/pdf
6. https://towardsdatascience.com/image-feature-extraction-traditional-and-deep-learning-techniques-ccc059195d04
7. https://towardsdatascience.com/tutorial-using-deep-learning-and-cnns-to-make-a-hand-gesture-recognition-model-371770b63a51