Cairo University
Faculty of Engineering

Department of Computer Engineering

Neural Network

Project

Submitted to

Dr. AbdElMoniem Bayoumi

Eng.Mohammed Shawky

Submitted by

Name	Sec	BN
Aya Ahmed Musad Husein	1	15
Salma Ragab Hassan	1	31
Shredan Abdullah kamal Moghazy	1	33
Nada Osman Abdalaziz Osman	2	30

(a)-Our project pipeline for hand gesture recognition involves several steps:

1. Data acquisition:

- Collecting a dataset of hand gesture images. includes examples of different hand gestures.

- Ensuring that the dataset covers a variety of hand shapes, sizes, orientations, and lighting conditions to make the model robust.

2. Data preprocessing:

- Preprocessing the acquired data to enhance the quality and standardize the input.

- Common preprocessing techniques include extraction of cr channel then thresholding upon it, then applying Guassian filter, then applying morphological operation.

3. Feature extraction:

- Extract meaningful features from the detected hand region that capture the discriminative information for gesture recognition.

- Popular techniques include extracting hand shape descriptors, such as LBP features, Hu moments, hog.

4. Gesture classification:

- Training a machine learning or deep learning model on the extracted features to classify different hand gestures.

- Choosing an appropriate classification algorithm such as Support Vector Machines (SVM), Random Forests, or Convolutional Neural Networks (CNN) depending on the complexity of the problem.

- Splitting the dataset into training and testing sets for model evaluation.

5. Model training and evaluation:

- Training the chosen model using the labeled hand gesture dataset.

- Evaluating the trained model using various metrics such as accuracy, precision, time.

6. Hand gesture recognition test:

- Deploying the trained model to perform hand gesture recognition.

- Mapping the predicted gesture labels to specific commands.

(b)-Preprocessing Module.

- Preprocessing the acquired data to enhance the quality and standardize the input.

- Common preprocessing techniques include extraction of cr channel then thresholding upon it, then applying Guassian filter, then applying morphological operation.

In details:

I've made more than 3 versions of the preprocessing module, at first I use skin segmentation and then applying the common preprocessing techniques as histogram equalization to enhance the image, adaptive thresholding, morphological operations multiple times then binarization

The one we finally settle to based on the idea of extracting the Cr channel to avoid the lighting and shadows effect as much as can be then Apply thresholding to the Cr channel using Otsu's thresholding then applying Guassian blur to filter the image 5 times, finally applying morphological opening 10 times.

(c)- Feature Extraction/Selection Module.

at the first we use hog , hu , LBP feature extraction techniques and then noticed that LBP extract huge number of features (16,000) so it is huge number , so we searched for technique reduce number of features found PCA so we used it on the level of all images so when we use it in the test images we cant apply PCA on the level of one image so we tried to remove the LBP to reduce the features and it doesn't affect the accuracy just one less percentage , so we use hog,hu moments only for nw and we searched a lot for anther techniques(contour based , region based , sift) but they extract different lengths of features on the level of the image so it makes error in the splitting and training so we don't use them , finally we use hog , hu moments only

(d)- Model Selection/Training Module,(e) Performance Analysis Module.

I divided the data into train and test (80% training , 20% test) , then tried classification algorithms P Logistic Regression , Decision Tree , K-Nearest Neighbors , Gaussian Naive Bayes , AdaBoost , Random Forest , Gradient Boosting , SVC . I also tried many possible parameters for them , then used RandomizedSearchCV to pick the best algorithm and the best parameters as well . Then we measure the accuracy of each model and we then select best 5 models (odd number) from these models to build our own model which uses voting scheme , each model vote for the input with a specific output , we then take the output that took majority of these votes to be the final output .

Here is the accurancies we got for all the classifiers :

Classifier	Accurancy
Logistic Regression	77
Decision Tree	60
K-Nearest Neighbors	82
Gaussian Naive Bayes	63
AdaBoost	73
Random Forest	80
Gradient Boosting	82
SVC	87

Note : We ran classifiers in different pcs to save time , so output of these (all) classifiers can not be found in the code files , but you can run the file to make sure of them .

Note: We generated another data set for testing only , but we got that best 5 classifiers are :

Logistic Regression , K-Nearest Neighbors , AdaBoost , Random Forest , SVC . So we used them in our final model .

we made model file where we load the model then read the test images one by one and make the pre-processing and extract the features then make the model we trained to predict the label of the test image so I store the all prediction in file and in also in array to compare with the true labels to give an accuracy for test data (ours) .

(g) Enhancements and Future work.

Robustness to variations: Improve the robustness of the system to handle variations in lighting conditions, hand orientations, occlusions, or noisy backgrounds. This can be achieved through data augmentation, advanced preprocessing techniques, or adversarial training to make the system more resilient to such variations.