Hand in the Dark:

Standardized Hand Gesture Recognition

Course Project of Northwestern EECS 349, Spring 2015 (Instructor: Prof. Doug Downey)

Jerry Li, Ivy Zheng, Yiyao Fu

*Computer Science and Material Science, Northwestern University*

jiamingli2017@u.northwestern.edu

*Computer Science and Mathematical Methods in the Social Science, Northwestern University*

xunzheng2017@u.northwestern.edu

*Mathematics and Political Science, Northwestern University*

annieyf@u.northwestern.edu

1. Introduction of Task

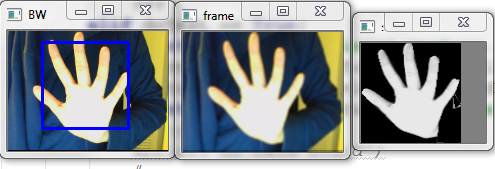
Gesture recognition is to interpret human motion after capturing the image. In this project, the input is a hand gesture picture, and the output is type of gesture as pre-defined in the dataset.

This task is meaningful and desirable for its wide range of applications, including various commands, UI’s and communication programs. For example, people can use gesture to send simple commands to the computer via a camera without touching the keyboard.

1. Information of Dataset

In this project, we focus on gesture recognition of standardized pictures, i.e. how to take standardized hand gesture pictures, and how to classify them.

To create the dataset, we make use of OpenCV’s cascade object detection for hand detection with a public hand HAAR cascade xml file, trained on about 20,000 positives and 20,000 negatives[[1]](#footnote-1). We also write a code that captures frame-by-frame images. This way, when given a video from the camera, we can detect the hand in the frame, crop the hand and save only the hand as a 100\*100 gray-scale picture, as shown in Fg. 1



F.g. 1: Picture Preprocessing

The complete dataset include 3130 processed pictures of 8 gestures from 7 people, with an average of 391 pictures of a gesture, and 56 pictures of a gesture from a specific person. For the purpose of future training, the images are taken by gesture, so that pictures in the same folder belong to the same category and the same person. We rename each folder so it shows the gesture category of the images and the person the gestures are from.

1. Training and Testing

The attribute we use is the grayed pictures, encoded as a list of pixels. Each picture is labeled with a gesture ID, which is also what the algorithm needs to predict. To train the dataset, our code first reads directories’ names as a list of categories, read each picture as a list of pixels, and then label each picture respectively by the folder it is in. Lastly, we first converted the pictures to binary, i.e. if a pixel is not black, then it is white. This will lessen the noise and improve our performance, as shown in the next section.

We use k-nearest neighbor as our classifier. We subtract two image matrices and square the difference as the measure of similarity, find the picture with the most similarity with the input, and return the gesture id of this picture as the predicted gesture id for the input.

To test our classifier, we use cross validation across different people. Specifically, the cross validation function shuffles our dataset, take images from one persona as testing set, and the rest as training set. Then, our function predicts the gesture id for testing set, validates it, and returns precision, recall and f1 in a csv file.

1. Experiment Results

We first experimented on the number of people from whom the gesture pictures were taken. We started with one person and used 10-fold cross validation. The precision reaches 0.998. However, when we added one person, the result dropped below 0.374. We think this is because the similarity between pictures from a single person, and thus started to increase the number of people and started to use validation across different people. However, as we increased the number of people, the precision increased from 0.358 and finally reached 0.625 with 7 people.

We then tested different k values for kNN method. We tested k from 1 to 10, and reached our best result at k=2, with a precision of 0.630.

We then implemented a function that converts gray-scale images to binary. As mentioned, this would lessen the noise and thus increased the precision to 0.844, which we think is a satisfactory result.

Finally, we test the data using different classifiers on Weka. As a benchmark, we first ran 2 Nearest Neighbor on Weka, with 66% training and the rest testing. We got 63.91% instances classified correctly. We then ran several other classifiers and got the following result:

TABLE 1

Performance of Various Classifiers on Weka

|  |  |
| --- | --- |
| **Classifier** | **Correctly Classified** |
| 2NN | 63.91% |
| Decision Tree | 61.09% |
| Attributed Selected Classfier | 58.83% |
| Neural Net | 53.67% |
| AdaBoost | 24.53% |

This shows how nearest neighbor is a better classifier.

1. Conclusion and Future Work

From the experiment, we can see that our proposed method work well in predicting the hand gesture, and give a better result than many other classifiers, and can potentially run better with a larger dataset.

There are also some future work we can do:

1. XXXXXX
2. YYYYYY
3. ZZZZZZ

1. Details of this xml file can be found in Envision, “HAAR xml file”, at http://nmarkou.blogspot.com/2012/02/haar-xml-file.html [↑](#footnote-ref-1)