Labeling Faces within Photo Albums

Jane Shelby Thompson
Marshall University
thompson516@live.marshall.edu

Abstract

Facebook has developed an algorithm to automatically suggest who a person in a photo is based on a number of features. This method is not as accurate as it should be, and often suggests a number of people who look completely unlike the person who is actually pictured. We aim to create an algorithm that more accurately and correctly labels the faces in an online photo album.

1. Introduction

Facebook has become a widely used and referenced website. Some of its most popular features are its photo albums, where people upload and store photographs. Users can "tag" the people in each photo and create a label for that person that links the photograph to their profile. This method can be very beneficial since it simplifies something and saves the user some time when creating a photo album online. We run into a problem when we find that Facebook automatically generates a list of people that are not even close to the real person in the photo. Throughout this summer, we showed the beginning steps to creating an algorithm that will automatically and correctly label all of the faces in a given album.

2. Dataset

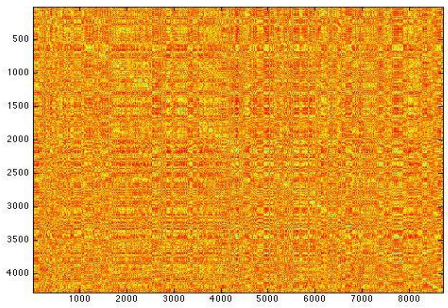
We used the PubFig83 dataset in our research. The PubFig83 dataset contains photos of 83 well known people, mostly celebrities, who have 100 or more photos documented. We use these images as both training and testing data to get results from our work.

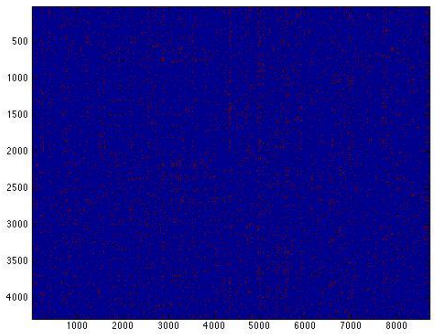
3. Methods

In order to begin, we decided to work with a few different methods. We worked with k-means, but the majority of our work was with pairwise distance, k-nearest neighbor, Euclidean distance, distance histograms, HOG, and finally SVM. These methods all gave different results, both good and bad.

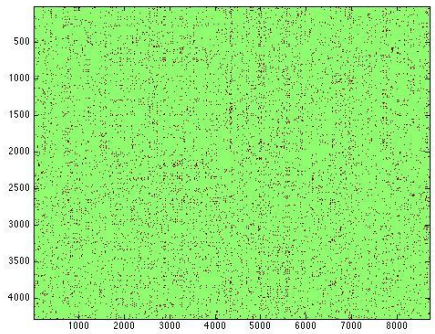
3.1 Pairwise Distance

When we began to examine the data in the PubFig83 Dataset, we decided it would be beneficial to look at the pairwise distance. We computed the cosine distance between the image features and plotted the resulting graph. This graph should have had a relatively straight diagonal line from the top left corner to the bottom right corner showing each image, and some similar color around these "clusters" to represent confusion. Instead, we got a graph (shown below) that was very noisy with little confusion. These were not good results.





kNN Graph Using 0



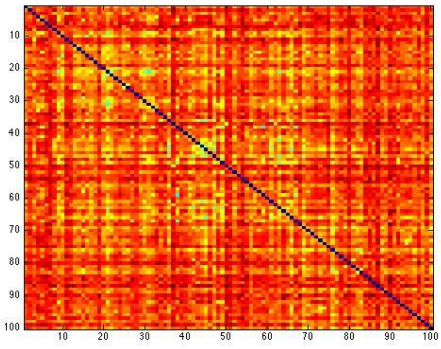
kNN Graph Using Infinity

3.2 K-Nearest Neighbors

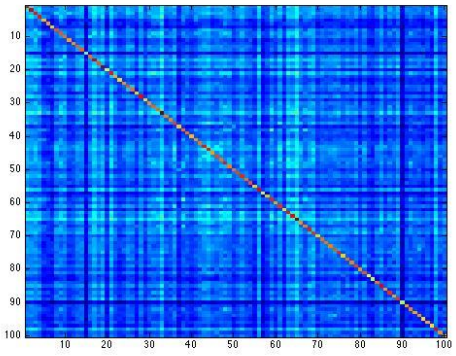
K-nearest neighbors is a method to determine what an object in a feature space is based on the training examples close to it. We use a few steps to graph kNN. We first iterate over the pairwise distance matrix and sort each feature in declining order. We then find top 5 k-neighbors and set all other rows in the matrix to 0. Next, we iterate over each row/threshold based on (i,i) where i is row number. Finally, we set all rows above threshold to 0 or Infinity (the results will appear different depending on what each space is set to). Our kNN graphs, like the pairwise distance graph, did not give good results; however, it is still beneficial to perform these tests.

3.3 Euclidean Distance Graph and Distance Histogram Graph

We felt that calculating the Euclidean distance and distance histogram might give us the results we were looking for. Using our own programs to calculate and graph the Euclidean distance and distance histogram, we discovered the results still were not satisfactory. We zoomed in on the top-leftmost 100x100 pixel section and discovered that we were still not achieving a good amount of confusion, and the confusion we did encounter was positioned in random spots of the graph instead of surrounding the individual



Zoomed-in 100x100 pixel Euclidean Distance Graph (cell size 10)



Zoomed-in 100x100 pixel Distance Histogram Graph (cell size 10)



Original image of Adam Sandler

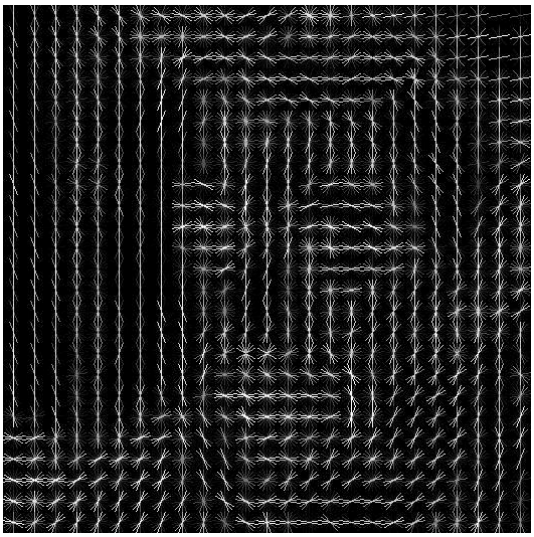


Image after HOG is applied (cell size 10 pixels)

images.

3.5 HOG

One of the other main methods we worked with was Histogram of Oriented Gradients, or HOG for short. HOG counts gradient orientation in parts of an image, and is computed on a number of nxn pixel (where n is a number) cells in an image. This method uses overlapping to improve accuracy. First we apply horizontal and vertical derivatives to each feature vector, and bin angles and add weight (magnitude) to each angle bin. These bins and their weights are represented in the form of a histogram. Our HOG program worked very well and we were able to get a number of excellent results using it on the PubFig83 dataset.

3.6 SVM

The last method we use is SVM. We have not finished our experimentation with this, and should we continue our research, we will be working mostly with SVM. For SVM, we continue to use PubFig83 dataset. We feed in the testing and training labels and HOG data to get the predicted classes. Using the training labels and data we get an SVM model, and using the testing labels and data we get the accuracy.

4. Results

Our graph results are pictured throughout this paper. Though most of the results are poor, we have managed to get some good results from the SVM. Using nearest neighbor, accuracy of the labels and data is 58.36%, while using SVM the accuracy of the labels and data is 87.58%. This supports the thought that SVM works better in the practice of face labeling.

5. Conclusion

Overall, these methods are useful in face detection but we need to do more research. This particular project is still in its beginning stages and we would like to continue to work on it. Given research can be continued, it is very likely that we can find a much better algorithm to use in photo album labeling than that which Facebook currently uses.

6. References

[1] Stone, Z.; Zickler, T.; Darrell, T., "Autotagging Facebook: Social network context improves photo annotation," *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW '08. IEEE Computer Society Conference on* , vol., no., pp.1,8, 23-28 June 2008

[2] Becker, B.C.; Ortiz, E.G., "Evaluation of face recognition techniques for application to facebook," *Automatic Face & Gesture Recognition, 2008. FG '08. 8th IEEE International Conference on* , vol., no., pp.1,6, 17-19 Sept. 2008

[3] Rowley, H.A.; Baluja, S.; Kanade, T., "Neural network-based face detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on* , vol.20, no.1, pp.23,38, Jan 1998

[4] Osuna, E.; Freund, R.; Girosi, F., "Training support vector machines: an application to face detection," *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on* , vol., no., pp.130,136, 17-19 Jun 1997