Margareta Kušan 11.2.2016.

Matconvnet

MatConvNet is a MATLAB toolbox implementing Convolutional Neural Networks (CNNs) for computer vision applications. Many pre-trained CNNs for image classification, segmentation, face recognition, and text detection are available. As a part of the project VGG, there is a pretrained CNN that gives face recognition results. As a face detector, they used DPM implementation of [3], using cascade detector from [4]. but it can be replaced by any other face detector. The detector in the moment of writing this does not provide mex files for Windows, only Linux and Mac.

There are 2622 classes (names) available in the pretrained CNN. As I was trying to evaluate this pretrained CNN, I got the results of softmax function which is used in the last layer of the CNN.

Results

What I did was trying to run image by image of famous people which are in the list of 2622 classes and image by image of people that weren't in these classes. I ran 3 images of every of 3 people (3*3) that were in the classes and 3 images of every of 3 people (3*3) that were not in the classes. I made a script that would calculate the distance between softmax responses. For the distance I chose Euclidian distance. I had to multiply the output scores for each class with 100 in order to get the proper distance values.

Three people from the pretrained CNN classes:

Figure 1. Katherine Heigl, results for every image respectively: a) Katherine_Heigl, score: 0.993, b) Katherine_Heigl, score: 0.999, c) Katherine_Heigl, score: 0.998

Figure 2. Alice Cooper, results for every image respectively: a) Alice_Cooper, score: 1.000, b) Alice_Cooper, score: 1.000, c) Alice_Cooper, score: 1.000

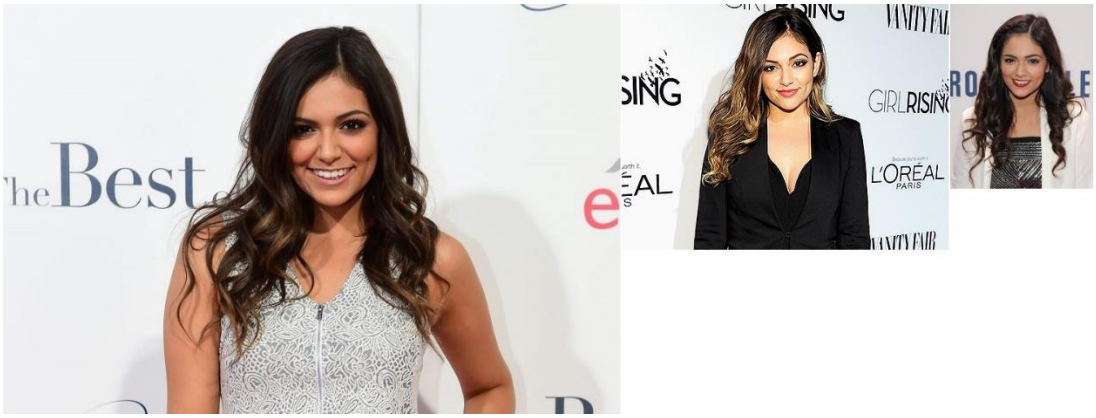

Figure 3. Bethany Mota, results for every image respectively: a) Bethany_Mota, score: 1.000, b) Bethany_Mota, score: 0.999, c) Bethany_Mota, score: 0.997

Three people that were not used to train the CNN:

Figure 4. Jelena Rozga, result for every image respectively: a) Sarah_Roemer, score: 0.500, b) Carrie_Underwood, score: 0.205, c) Kelli_Berglund, score: 0.122

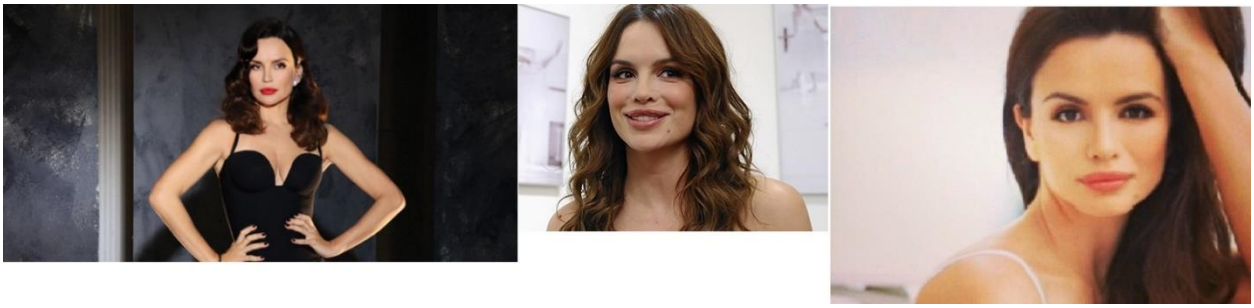

Figure 5. Severina, result for every image respectively: a) Roma_Downey, score: 0.359, b) Kasia_Smutniak, score: 0.359, c) Geena_Davis, score: 0.051

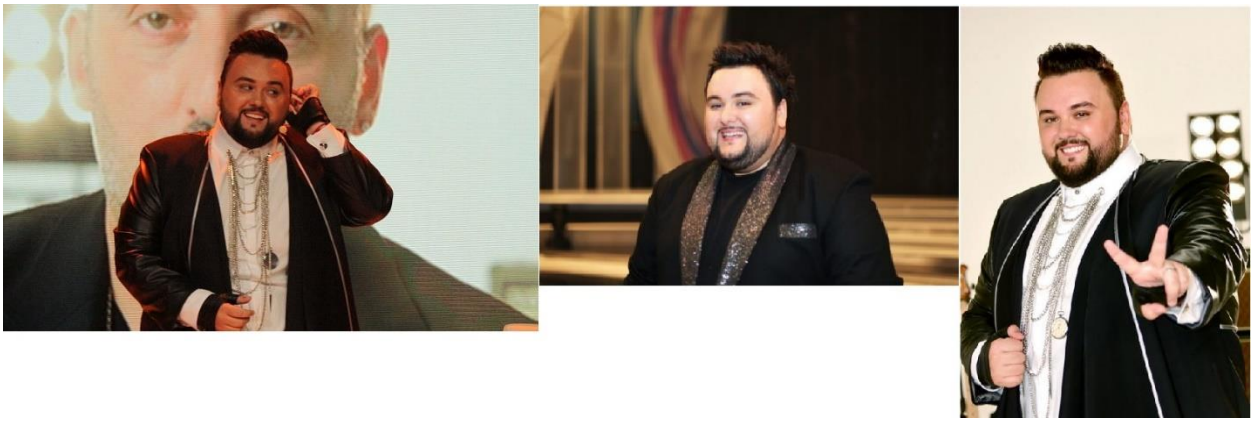

Figure 6. Jacques Houdek, result for every image respectively: a) Jack_Whitehall, score: 0.032, b) Max_Adler, score: 0.073, c) Jack_Whitehall, score: 0.111

Inserting the distance values between these images into a distance matrix and running k-means on the data resulted in providing these 6 clusters:

Cluster	Image
1	rozga1
2	zak1, zak2, zak3, rozga2, rozga3, seve2
3	coop1, coop2, coop3
4	heigl1, heigl2, heigl3
5	mota1, mota2, mota3
6	seve1, seve2

Conclusion

Looking at the table, it seems that this face recognition software is really good at classifying people that exist in the training classification of the CNN, however the results are not good for people that the CNN is unfamiliar with. Very high score with the group of people that exist in the classification might be due to using the same images as they were used for training the CNN, if the authors used images obtained from first google results like I did. Additional research is needed by running the software recognition in batch version on a larger dataset of new people and then looking at the results. The way I used here is to collect more images for one person and then using the softmax responses I measured the Euclidian distance (or some other distance). In the next step clustering is made in order to see if the clustering algorithm finds similarities between images of the same people.

References

[1] O. M. Parkhi, A. Vedaldi, A. Zisserman

Deep Face Recognition (url)

British Machine Vision Conference, 2015.

[2] A. Vedaldi, K. Lenc

MatConvNet - Convolutional Neural Networks for MATLAB (url)

arXiv:1412.4564, 2014.

[3] M. Mathias, R. Benenson, M. Pedersoli, L. Van Gool

Face detection without bells and whistles (url)

European Conference on Computer Vision, 2014.

[4] P. Felzenszwalb, R. Girshick, D. McAllester

Cascade Object Detection with Deformable Part Models (url)

Computer Vision and Pattern Recognition 2010.