Automatic Group Photograph Enhancement

Yuliang Zou, Yayan Zhao, Zhusheng Wang, Enpei Xi
University of Michigan, Ann Arbor
{ylzou, yayanzh, zhusheng, exi}@umich.edu

Abstract

We propose a framework to automatically enhance the quality of group photos[1]. Given a set of photos of the same group of people, our system assigns quality score to each detected face based on facial expressions (i.e. smiles and eye-closure). Utilizing these scores, the system selects the photo with the highest total quality score as the base photo. Finally, we replaces each face in the base with the highest score of the same person from other photos to synthesize a perfect group photo. Results show that our framework is feasible.

1 Introduction

In our daily life, we always want to take photos with our families and friends, to commemorate the wonderful moments we share together. However, such group photographs can often turn out to be unsatisfactory since it is so difficult to ensure that everyone has a nice facial expression at the same time [2]. When a group photograph has been taken, we will always disappointingly find that some people are looking away, some people are closing their eyes and some people are exactly wearing a sad expression in that photograph. And as long as kids are involved, this so-called simple task can be really more challenging. See Figure 1 for more details.

Figure 1: Examples of unsatisfying faces [5]

This project aims at synthesizing a perfect group photograph automatically from a given set of group photos. On the premise that these photos are taken at the same scene, it is likely to select a best face of each person among those photos, yielding a ”perfect” group photo.

2 Related Work

Albuquerque et al. [1][5] proposed a framework to select an attractive portrait of a person from a video sequence based on smile and eye-closure attributes. Their framework detects eye and mouth

[1]Please refer to our GitHub repository: https://github.com/Yuliang-Zou/Automatic_Group_Photography_Enhancement

regions in a face using the AdaBoost algorithm [17] and trains an SVM classifier [7] over PCA-based features for labeling these regions as good or bad. Our quality evaluation shares the same attributes with them. However, our work differs from theirs in many aspects. Firstly, we formulate face detection and quality evaluation as a multi-task learning problem, and solve it with Convolutional Neural Networks (CNN) [8][9][10][11]. Secondly, instead of using a dataset collected in a semi-controlled environment, we train our model with large scale datasets with real-world images [18][19]. Thus, our system is more robust and generalizes pretty well to new data. Shah et al. [2] proposed a framework to solve the exactly same problem we want to solve. However, they trained and tested their model on their own collected dataset, which is not persuasive enough.

The contributions of this project are as follows:

(i) We propose a framework to automatically enhance group photos, utilizing state-of-the-art CNN models [11] and large scale datasets [18][19] to have a good generalizing ability.

(ii) We annotate the FDDB dataset [18] to support smiles and eye-closure labels, which allows us to train face detector and facial expression simultaneously.

3 Technical Part

The whole system can be divided into several components as shown in Figure 2.

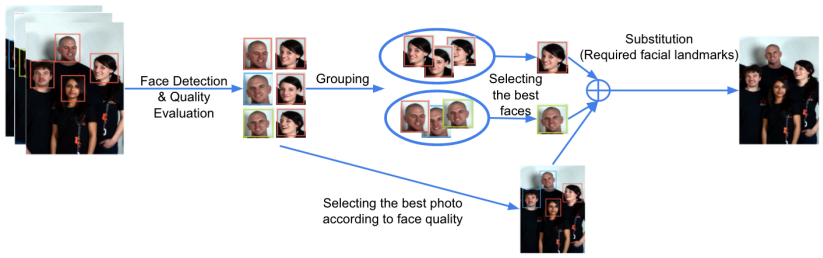

Figure 2: System pipeline [5]

3.1 Face detection

The first stage is to solve the problem of face detection. With the recent popularity of deep learning [8], it is shown that many computer vision tasks including general object detection and recognition can be tackled with CNN based algorithms [9][10][11]. For object detection task, region-based CNN is now the mainstream. Among these algorithms, Faster R-CNN[11] provides a general framework for real-time object detection. Although it is designed for general object detection, we here use it as a face detector.

Faster R-CNN. In this part, we briefly introduce the key idea of the Faster R-CNN framework. More details can be found in the original paper [11].

To detect objects with different sizes, the traditional methods either use sliding window to iterate through the image and get detection directly [20], or use region proposal generation method to search potential regions first, then pass the selected regions in object classifier [21]. Both methods are time-consuming, and hence can not be used in real time. In order to speed up the detection, Faster R-CNN uses RPN (Region Proposal Network) to generate region proposals.

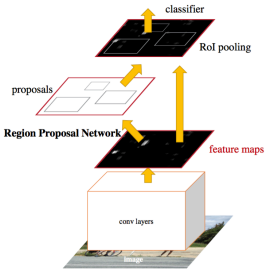

Figure 3: Faster R-CNN [11]

The pipeline is as follows: Firstly, hierarchical features are extracted from input image via a pile of Convolution layers. Then, RPN takes such features as input to generate region proposal. To deal with different scales and aspect ratios of objects, anchors are introduced in RPN. We can simply treat it as some pre-defined windows here. But instead of iterating through the input image, they iterate on the feature map given by a pile of Convolution layers to find potential regions that might contain foreground object. To conclude, we can regard RPN as a filter to select potential regions from all the regions of the input image. Finally the object classifier extract region feature from feature map, with the guidance of RPN, to get final detection result.

3.2 Quality evaluation

This is the essential part of the system. A good evaluation function would help to find the most satisfying faces. To simplify the problem, we design the evaluation function as a combination of ”smile score” and ”open eyes score”. The two kinds of scores can be acquired by binary classifiers.

The success of Faster R-CNN structure on object detection proves that the features extracted by CNN can be used to discriminate different objects and background. We believe that the learned feature map can also be used to determine if the face is smiling or not, opening eyes or not. Based on this assumption, we add two additional logistic regression function as output to train two binary classifiers for the new task. The output range is [0,1], the higher score, the more likely that the face is smiling (opening eyes). Then we simply add the two scores together to get the final quality score of the face.

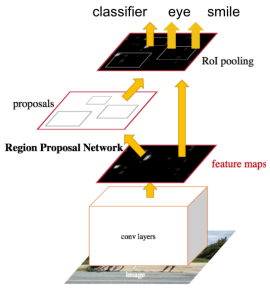

Figure 4: Detect face and evaluate face quality simultaneously. Credit to [11]

3.3 Identity grouping

Afterwards, we try to build the person-identities step by step based on the corresponding face features [2]. In order to achieve this aim, we try to perform face grouping with SIFT features [12]. And in our case SIFT features are features extracted from faces to help in reliable matching between different views of the same person [13]. Therefore, we hope to obtain the similarity score between two faces using SIFT features.

The following procedure is used to calculate the similarity score between two faces A_i, A_j. First, SIFT keypoints, will be extracted from the faces A_i, A_j respectively. Meanwhile, we acquire the number of keypoints in two faces. Then, the number of matching keypoints will be calculated based on the Euclidean distance of their feature vectors. A keypoint from A_i against A_j is accepted as matching keypoint only if the closest distance between this keypoints' feature vectors from A_i and all keypoints' feature vectors from A_j is less than Ratio(Here, we set the ratio to 0.8 times the distance to the second closest match [14]. Finally, we are able to calculate the similarity score using the following formula.

$$D = \frac{1}{2}\left(\frac{K_i}{Num(A_i)} + \frac{K_j}{Num(A_j)}\right) \times 100\% \tag{1}$$

where K_i is the number of matching keypoints from A_i against A_j, K_j is the number of matching keypoints from A_j against A_i, and $Num(A_i)$ is the number of keypoints found in A_i, $Num(A_j)$ is the number of keypoints found in A_j. $D \in [0\%, 100\%]$ is the similarity score and high D values indicates large similarity between two face images.

Since we are able to judge whether these two faces belong to same person, we go on to solve how to group the given faces efficiently. At first, we try to form a base group through a picture with max number of faces. And usually, this base group will contain all kinds of faces from every person in the dataset. Then if we want to add a new face, this face will be assigned to a certain group according to the Similarity Score. The process is shown briefly in Figure 5. And the threshold about matching is approximately 50%. One face by one face, we can achieve identity grouping finally.

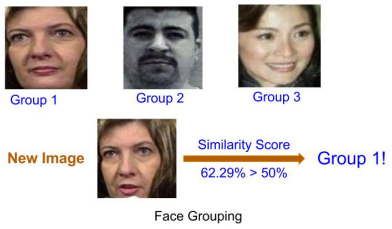

Figure 5: An illustration of face grouping

3.4 Facial landmark extraction

In order to identify a person in an image, we first need to find where in the image a face is located. Therefore, face detection – locating a face in an image and returning a bounding point that contains the face – was a hot research area. Applications such as Head pose estimation, Face Morphing, and video effect used in snapchat are based on this theory. A regular method is extracting from 5 major landmarks/facial points, which are eyes, tip of nose and corners of mouth. However, this method is not precise enough to do further facial implementation. There is a popular method can extend the old one to handle 68 landmarks by deep learning without involving significant increase in run time cost [15].

The Facial Landmarks Detection can be separated as Landmarks Detection of Facial Components and Facial Contour[16]. Base on the objective of our project, we will focus on detection of facial components. The landmarks of facial components can be detected as a linear regression problem, where a loss function is the total loss from all landmarks in a batch of N images. Comparing to the typical convolutional neural network for classification, the CNN model used in this problem includes use of loss function, but it still features in similar manner to the classical one. The learning process of CNN constrained on facial components include two processes, forward propagation and backward propagation with filter updates. As typical CNN, the forward propagation includes convolution and subsampling. A set of feature maps will be used to convolute with the original image. The results of convolution computes the output of neurons that are connected to local regions in the input. After that, another process will be implemented is called pooling, or subsampling, which is

a form of non-linear down-sampling. There are several nonlinear functions to implement pooling among which max pooling is the most common. It partitions the input image into a set of non-overlapping rectangles, which are called hidden layers, and, for each such sub-region, outputs the maximum. And repeat the above steps on the hidden layers. Finally, after several convolutional and max pooling layers, the high-level reasoning in the neural network is done via fully connected layers. Neurons in a fully connected layer have full connections to all activations in the previous layer. At this point, the loss function can be applied to fully connected layers to classify the facial components and after training, landmarks can be identified by this process.

3.5 Face replacement

The last stage is to composite an ideal group photo through automatic face replacement.Based on our previous work, the system first select a photo with the highest total score as the target photo, then each face in the target photo will be automatically replaced by the face with the highest score of the same person [2]. The general process of face replacement is divided into two parts: face alignment and seamless blending.

Face Alignment As shown in Figure 6, when our system is supplied with the facial landmarks extracted in the previous step, we could calculate a convex hull of the landmarks which is a boundary those points without concavities. Then the system will perform a Delaunay triangulation of the points on the hull, as shown in Figure 16 (see Appendix). Delaunay triangulation allows us to divide a face into small parts and calculate the affine transform for each corresponding triangle between the source and target face. Through this affine transform, we could align the three corners of the triangles and transform all pixels inside the source triangle to the target face. However, the alignment alone cannot get an ideal outcome, since the illumination, skin tone or skin texture may vary significantly from face to face, which results in visible seams, and can only be partly hidden.

Figure 6: Face Alignment

Seamless Blending Here we adopt the Poisson Image Editing from Perez, et al. [3],which has been widely used for seamless image composition. It permits the seamless importation of both opaque and transparent source image regions into a destination region [3].The reason why Poisson blending works better than other methods is that the human system is more sensitive to contrast than intensity values,and its basic idea is to change the gradient field v of the interesting region(Ω) in the target image to that of the source image g, and keep the pixels on the boundary unchanged,as shown in Figure 17 (see Appendix).

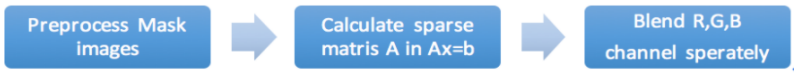

Figure 7: Seamless Blending

The pipeline of this algorithm is shown in Figure 7. First we need to generate a mask which indicates the overlapping region between the source and target face. Then a linear system of equations $Ax = b$ is required to compute the resulting image from the source and target gradients, where A is the sparse coefficients matrix, x is the output image, and b is the desired gradient matrix. For pixels outside the masked region, the output image pixel is simply the same as the target image. and A is simply an identity matrix. For pixels inside the masked region, the output image pixel x at (i, j) depends on its neighbors according to the following equation: $4 * x(i, j) - x(i+1, j) - x(i-1, j) - x(i, j+1) - x(i, j-1)$ = desired pixel gradient. Finally, each image's R,G,B channels are blended separately. While the same sparse coefficients matrix A may be used for each color channel, the gradient matrix b is different for each color channel. We could even retain some of the gradient of the target image in the mask area for a better result which is called Mixing Gradients. For each points in Ω, we pick

the gradient to each neighbor from either the source image or the target image, whichever gradient has a higher absolute value [3],which could be represented by the equation below.

$$v(x) = \begin{cases} \nabla f^*(x), & \text{if } |\nabla f^*(x)| > |\nabla g(x)| \\ \nabla g(x), & \text{otherwise} \end{cases} \tag{2}$$

for all $x \in \Omega$.

4 Experiments

4.1 Face Detection

We train a Faster R-CNN[2] face detection model on the recently leased WIDER face dataset [27]. There are 12,880 images and 159,424 faces in the training set. In Figure 15 (see Appendix), we demonstrate some randomly sampled images of the WIDER dataset. We can see that there exist great variations in scale, pose, and the number of faces in each image, making this dataset challenging. We train the face detection model based on a pre-trained ImageNet model, VGG16 [22]. We randomly sample one image per batch for training, using momentum SGD solver 50k iterations with a base learning rate of 1e-3. We run the Momentum another 20K iterations reducing the base learning rate to 1e-4.

With the same settings, we did two experiments here. The first one is to use data from WIDER dataset only, and the AP (Average Precision) of face we got is only 0.06 in WIDER test set, which is too low to accept. We think using face category only might not fully use the power of RPN. So we did another experiment. With the additional training data from VOC2007 dataset [23], the AP of face we got this time is 0.328, which improves a lot. Then we tested this model on a benchmark dataset FDDB [18] to evaluate the performance. And we found the AP of face for the whole dataset is 0.902, which is so-called state-of-the-art. We think the auxiliary category information[3] help the RPN to better learn how to discriminate foreground object and background, and therefore improve the performance of face detection.

We further demonstrate qualitative face detection results in Figure 8. It can be observed that the Faster R-CNN model can deal with challenging cases with multiple overlapping faces and faces with extreme poses and scales.

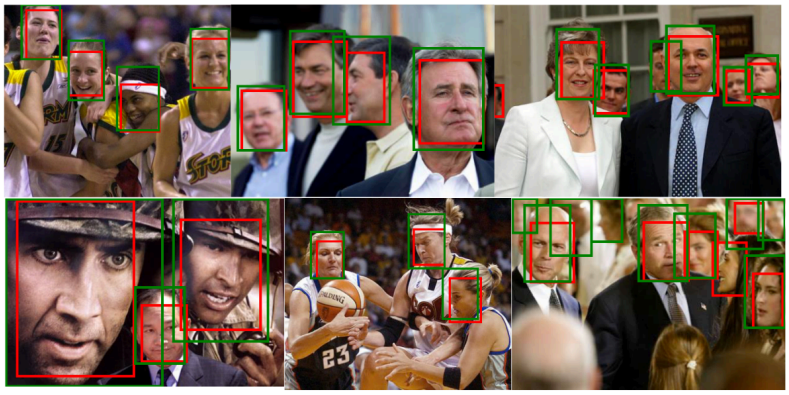

Figure 8: Examples of face detection

As we can see, the model might fail to detect some faces with extremely challenging cases (i.e. the right down one in Figure 8). However, this will not be an issue since group photos with such faces

[2]We use a TensorFlow version implementation: `https://github.com/smallcorgi/Faster-RCNN_TF`. Some implementation details might be different from the author released version.

[3]There are 20 classes in VOC2007 dataset, one of which is "person". Since this category contains face, but they don't have bounding box for faces. In order to eliminate the potential negative effect, we simply did not use any image containing this category.

will have a low overall quality score and will be selected as the base. We can ignore the information for such faces.

4.2 Quality Evaluation

Since we want to train the model to detect faces and output smile and eye-closure scores, we need a dataset with both bounding box and smile/eye-closure labels. So far as we know, there is no such a dataset. So we decide to annotate the benchmark dataset FDDB to add smile/eye-closure labels[4]. They have already split the whole dataset set into 10 folds randomly, we just take the first 7 folds as training set, the remaining as test set for this task.

On top of the Faster R-CNN face detection framework, we modify the functions to support new utilities. We train the face detection model based on the pre-trained face detection model in previous section. We randomly sample one image per batch for training. We run the Momentum SGD solver 50k iterations with a base learning rate of 1e-3(1e-4)[5] and run another 20K iterations reducing the base learning rate to 1e-4(1e-5).

We test the two models on the test set[6], and got the following results:[7]

Model	eye				smile			
	Accuracy	Precision	Recall	F1-score	Accuracy	Precision	Recall	F1-score
1e-3	83.81%	0.80	0.46	0.58	83.31%	0.87	0.61	0.72
1e-4	85.75%	0.77	0.60	0.68	84.23%	0.84	0.67	0.75

Figure 9: Quality evaluation performances

According to the evaluation above, the model with base learning rate 1e-4 is better. And we decide to use this model in our system.

Figure 10: Examples of quality evaluation

[4]There are about 930 faces in the dataset with closed eyes, 3133 faces with open eyes, 971 faces undetected. And as for the smiles, there are 1459 faces with smile, 2645 faces without faces, 930 faces undetected.

[5]Here we did two experiments with different base learning rates: 1e-3 and 1e-4.

[6]And we also test the face detection utility of this model on the FDDB test set. The AP is 0.907.

[7]Data is unbalanced. We here set the category with less number as positive to evaluate precision, recall, and F1-score.

4.3 Identity Grouping

In order to test the performance of our face grouping algorithm based on SIFT features individually, we try to use viola-jones face detection method [20] to extract the faces from given images, which are obtained from the LFW dataset [24]. Afterwards, we will evaluate our face grouping algorithm through similarity score between two chosen faces.

As you can see in the following figure, we try to extract group1 to group10 from LFW dataset, and each group contains 5 images corresponding to the same person. Among each group, we can carry out 10 experiments between any two images. So it is obvious that we can totally perform 100 experiments about judging whether two faces belonging to the same person will be considered as one person. Then we extract 5 images from 5 different groups randomly and then mix these images together. Similarly, we can also perform 100 experiments about judging whether two faces belonging to two different persons will be considered as two persons.

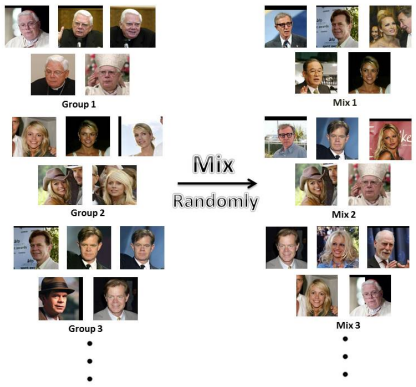

Figure 11: Evaluate identity grouping

Finally, we set the threshold to 44%, the accuracy that two faces belonging to the same person are judged as one person is 73%. Meanwhile, the accuracy that two faces belonging to two different persons are judged as two persons is 68%. Since the faces extracted through the viola-jones face detection method [20] are not ideal, i.e. the faces may include hat, hair or other interfering background, the performance of our face grouping algorithm will be influenced considerably. Therefore, if we use the better extracted faces using our own method, the results will be better correspondingly. And it will be demonstrated clearly as well as directly in Part 4.6.

4.4 Facial Landmark Extraction

As discussed before, the regular method to solve this problem is using Convolutional Neural Network, which requires large data sets for training. It takes time to finish the whole process, but fortunately, there are open source codes with using a python package called dlib can do it easily. And it is a well-developed way to find 68 facial landmarks precisely using Convolutional Neural Network.

Beside the regular method, we would try another method on Matlab to extract facial landmarks. First, we would extract facial components as patches using a build-in toolbox called Computer Vision System Toolbox, and it outputs coordinates of corners of patches. These patches include eyes, nose and mouth. After that, blob will be used to estimate the feature point of each patch, such as corner, lips, and eye brow, and these will be marked as our landmarks. Currently, we can extract about 20 facial landmarks using this method. As the image size increases, the accuracy will become lower. So the image need to be resized before the process. Although facial landmarks outputted by this method are not as much as CNN, it still good enough to detect the facial components and they can be used in our project.

Figure 12: Facial landmark extraction with Matlab

4.5 Face replacement

From the experiment results, we can say that our face replacement method works well in most occasions. But we also can see that there are still some limitations for this method. To be specific, first, the results may be affected mostly by the different angle from both faces, which makes the result image looks incompatible. Second, some factors such as skin tone, color temperature and brightness of the photo may have negative effects on results as well. And it has magnified the impact especially for photos from different genders.

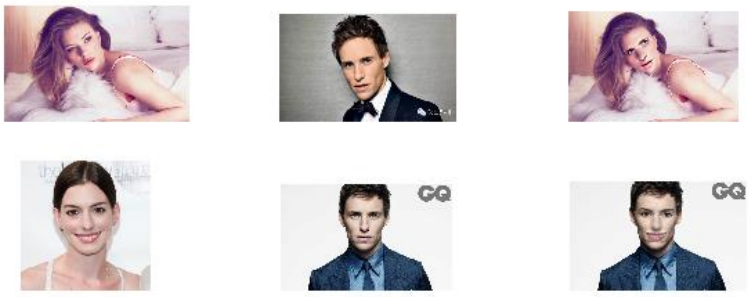

Figure 13: Examples of face replacement

4.6 Overall

Here we show the full procedure of the system on the example group photos:

Figure 14: Examples of face replacement

5 Conclusion

In this paper, an automatic photography enhancement method that replaces faces for a group of people was introduced. As a group project, we separated it into several sub-tasks, which include face detection, quality evaluation, identify grouping, facial landmark extraction, and face replacement. These designs are able to capture faces from a group of people and replace them with the best one they have step by step. Although the final result that combined all works does not seem perfect, we still did great in each sub-task. In future, this whole automatic process is expected to become more compatible and has a better result.

Acknowledgments

We would like to thank the NIPS community for providing this LaTeXtemplate.

References

[1] Albuquerque G, Stich T, Magnor M A. Qualitative Portrait Classification[C]. *VMV*. 2007: 243-525.

[2] Shah R, Kwatra V. All smiles: automatic photo enhancement by facial expression analysis[C]. *European Conference on Visual Media Production*. 2012:1-10.

[3] P. Perez, M. Gangnet, and A. Blake.Poisson image editing, *ACM Trans. on Graphics*, Proc. of Siggraph, **vol. 22**, no. 3, pp. 313?318, July 2003.

[4] V. Kwatra, A. Schdl, I. Essa, G. Turk, and A. Bobick. Graphcut textures: Image and video synthesis using graph cuts. *ACM Transactions on Graphics*, SIGGRAPH 2003, **22**(3), July 2003.

[5] Albuquerque G, Stich T, Sellent A, et al. The good, the bad and the ugly: Attractive portraits from video sequences[C]. *Visual Media Production (CVMP 2008)*, 5th European Conference on. IET, 2008: 1-4.

[6] SATYA MALLICK, Delaunay Triangulation and Voronoi Diagram using OpenCV, 2016

[7] Cortes C, Vapnik V. Support-vector networks[J]. *Machine learning*, 1995, **20**(3): 273-297.

[8] LeCun Y, Bengio Y, Hinton G. Deep learning[J]. *Nature*, 2015, **521**(7553): 436-444.

[9] LeCun Y, Boser B, Denker J S, et al. Backpropagation applied to handwritten zip code recognition[J]. *Neural computation*, 1989, **1**(4): 541-551.

[10] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]. *Advances in neural information processing systems*. 2012: 1097-1105.

[11] Ren S, He K, Girshick R, et al. Faster R-CNN: Towards real-time object detection with region proposal networks[C]. *Advances in neural information processing systems*. 2015: 91-99.

[12] D. G. Lowe. "Distinctive image features from scale-invariant keypoints", *International Journal of Computer Vision*, **vol. 60(2)**, pp. 91-110, June 2004.

[13] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, **60**, 2004.

[14] Antonopoulos P, Nikolaidis N, Pitas I. Hierarchical face clustering using sift image features[C]. *Computational Intelligence in Image and Signal Processing*, 2007. IEEE Symposium on. IEEE, 2007: 325-329.

[15] Zhanpeng Zhang, Ping Luo, Chen Change Loy, Xiaoou Tang. Facial Landmark Detection by Deep Multi-task Learning, *ECCV*, 2014

[16] Wissam J. Baddar, Dae Hoe Kim, Seong Tae Kim, Yong Man Ro, A deep facial landmarks detection with facial contour and facial components constraint, 2016 IEEE International Conference on Image Processing, 2016

[17] Viola P, Jones M. Rapid object detection using a boosted cascade of simple features[C]. *Computer Vision and Pattern Recognition*, 2001. Proceedings of the 2001 IEEE Computer Society Conference on. IEEE, 2001, 1: I-511-I-518 **vol. 1**.

[18] Jain V, Learned-Miller E G. Fddb: A benchmark for face detection in unconstrained settings[J]. *UMass Amherst Technical Report*, 2010.

[19] Yang S, Luo P, Loy C C, et al. WIDER FACE: A Face Detection Benchmark[J]. *arXiv preprint* arXiv:1511.06523, 2015.

[20] Viola P, Jones M J, Snow D. Detecting pedestrians using patterns of motion and appearance[J]. *International Journal of Computer Vision*, 2005, **63(2)**: 153-161.

[21] Girshick R. Fast r-cnn[C]. Proceedings of the IEEE International Conference on Computer Vision. 2015: 1440-1448.

[22] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. *arXiv preprint*, arXiv:1409.1556, 2014.

[23] Everingham M, Van Gool L, Williams C K I, et al. The pascal visual object classes challenge 2007 (voc 2007) results (2007)[J]. 2008.

[24] Huang G B, Ramesh M, Berg T, et al. Labeled faces in the wild: A database for studying face recognition in unconstrained environments[R]. *Technical Report 07-49, University of Massachusetts, Amherst*, 2007.

Appendix

Figure 15: Examples of WIDER dataset [19]

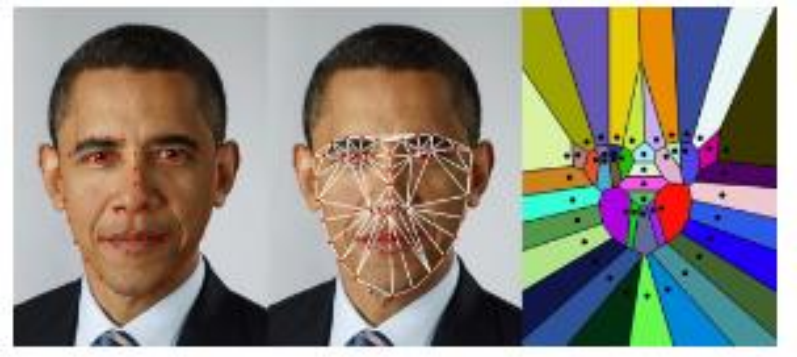

Figure 16: Examples of Delaunay triangulation [6]

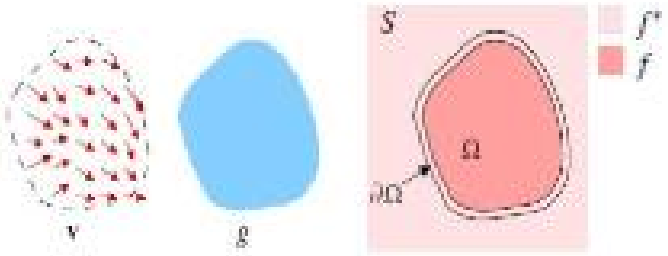

Figure 17: Examples of Poisson Image Editing [3]