Face Detection and Recognition With Deep Convolutional Neural Network

Rui Zhang (u5963436)
Research School of Computer Science
The Australia National University
Canberra, ACT 2601
Email: u5963436@anu.edu.au

Abstract— **Face detection and recognition techniques are usually used to analyze people in the images and videos and widely applied on the cameras and mobile phones. To design a method which can count and identify students in the class, this project combines multi-task cascades convolutional neural networks (MTCNN)[1] to detect faces and the VGG face recognizer[2] to recognize faces. In the experiments, we use anuclass01.JPG as the calibration data to adjust the minimal detection window, the scale factor and thresholds used in the MTCNN and anuclass02.JPG and anuclass03.JPG are used as the test dataset. Our system achieves high detection and recognition accuracies: among 59 faces, 56 and 58 faces are detected in both test images without wrong detection; among detected faces, 82.14% and 72.41% are recognized correctly respectively with faces taken from anuclass01.JPG as the ground truth. Besides, we also try weak face alignment which aligns positions of eyes in the images, but it does not improve the recognition accuracy.**

I. INTRODUCTION

Detecting and recognizing students in the class can help the lecturers know the number of present students and evaluate the quality of their lectures. If most of students are present, the content of lectures may be interesting. So, we are going to design a method to count and recognize the present students. However, designing such a method can be very hard because various factors like backgrounds, occlusion, side faces and lighting can increase the difficulty of face detection and recognition and meanwhile the method needs running in real time if it is implemented in the cameras and mobile phones. From the early Viola-Jones algorithm[3], which uses a cascade of classifiers to detect faces in real time but hardly deals with side faces, to the recent Deep Convolutional Neural Network (CNN) [4][1] designed for precise face detection but suffering the low speed, researchers never stop exploiting different methods and neither find perfect approaches. Similarly in the field of face recognition, the early approach Eigenface[5] effectively recognizes aligned faces by extracting features with Principal Component Analysis but it fails to deal with faces without being aligned and images with extreme lighting. Current deep CNN architectures [6], [7], [8], [9], [2] are able to recognize weakly aligned faces but fails to be applied on the cameras or mobile phones because of their high complexity.

In this project, we combine two state-of-the-art methods, MTCNN [1] for face detection and VGG face recognizer [2] for face recognition. To deal with the problem of a known face matching multiple detected faces, we iteratively select the matching pair with highest similarity. Although both advanced techniques obtain state-of-the-art performances on various and challenging benchmarks and overcome shortcomings of early methods, their effects on our images are unknown. Thus, we use a class photo to calibrate hyperparameters of MTCNN and conduct experiments on two remaining class images and the results show that our system can address facial occlusion and side faces in face detection as well as recognition. Besides, we also try weak face alignment in the experiments but it does not improve the performance of our system.

II. Related Work

This project focuses on the tasks of face detection and recognition and there have been lots of famous solutions, which can be separated as non-neural-network and neural-network methods. Among non-neural-network methods for face detection, Viola-Jones algorithm [3] selects Haar features from images and exploits a cascade of classifiers to reject non-face regions. This algorithm provides a competitive real-time detector and can be used to detect other objects, such as bikes and cars. Apart from Viola-Jones algorithm, HOG object detector [10], DPM face detector[11] and other work extending both methods[12] are important non-neural-network approaches. For face recognition, eigenface[5] uses Principal Component Analysis to extract features of faces for face recognition but it can hardly be used to recognize unaligned faces. These methods are famous for their simplicity and efficiency so the most of the current cameras and mobile phones apply them to analyzing the images.

Challenges, like side faces and facial occlusion, faced by these non-neural-network methods motivate lots of work using deep neural networks to detect and recognize faces. CNN is firstly used for object detection in [13]. The proposed method, R-CNN, exploits selective search [14] to generate candidate regions and CNN to extract regional features. Besides, a SVM classifier is trained to categorize a region into face or non-face and after classification, a bounding box regression is used to refine the position of the bounding box. R-CNN lays a solid foundation for detecting objects with CNN but its low detection speed (47s/image on CPU and 13s/image on GPU)[13] motivates R-CNN variants: fast RCNN[15] and faster RCNN[16], which have higher detection accuracies and speed. R-CNN variants detain core ideas like extracting regional features with CNN and bounding box regression but simplify the multi-stage pipeline of R-CNN by compressing multiple stages into the CNN architecture to speed up detection. R-CNN and its variants are extended to be face detectors, such as multi-view face detector [4] and MTCNN[1]. MTCNN uses cascaded CNNs trained by the multi-task learning to detect faces. It comprises three stages to process an image and in each stage, the input candidate regions are scored and the bounding box regression vectors and predicted facial landmarks are returned. Besides, thresholding and non-maximum suppression are used to discard candidate regions. This method achieved state-of-the-art performance in both of face detection and alignment and is used in our system.

Similar to the development in the field of face detection, current face recognition techniques mainly are based on deep CNN architectures and their idea are similar: extracting facial features by the deep CNN, such as DeepFace[17], DeepID's [6], [7], [8], FaceNet[9] and VGG face recognizer [2]. These approaches do not have as many steps as those for face detection and instead, they have deep architectures to extract more representative features. Besides, frontal faces are usually given for face recognition. Therefore, face recognition is kind of easier than face detection. Our method use the VGG face recognizer[2] which trains a deep CNN architecture to learn the face embeddings.

III. Methods

In this section, we are going to introduce two key methods: MTCNN and the VGG face recognizer. Furthermore, we explain how we align faces with facial landmarks.

A. Joint Face Detection and Alignment Using Multi-task Cascaded Convolutional Networks

Face detection and alignment are two necessary steps before face recognition and combining both steps makes it more efficient to recognize faces. MTCNN uses cascaded CNNs trained by multi-task learning to detect and align faces, which achieved the best performance on FDDB[18] and WIDER FACE [19] benchmarks for face detection and AFLW [20] benchmark for face alignment when it was proposed. The whole pipeline of MTCNN mainly has three stages, shown in Fig. 1.

1) Overall Framework: The three stages can be summarized as: (building pyramid image and) generating candidate regions, rejecting candidate regions, generate facial landmarks.

Firstly, the input image is scaled to a pyramid image to generate images with different size. Then, a 12x12 window is exploited to scan each image

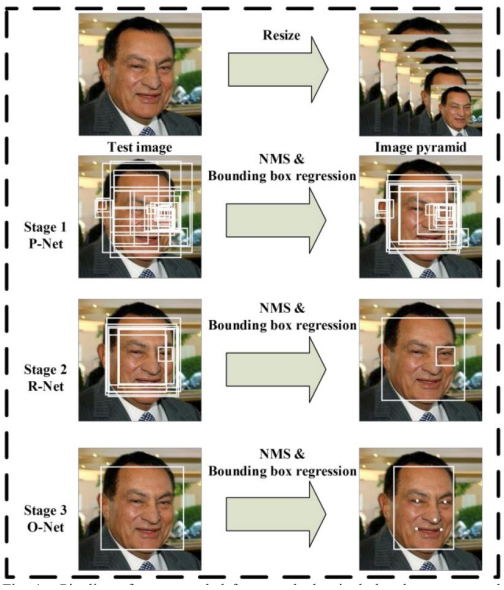

Fig. 1. Three-stage pipeline of MTCNN[1]

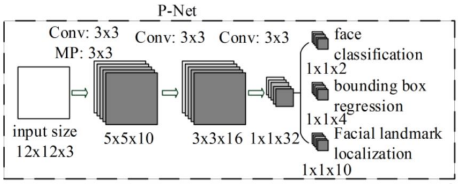

Fig. 2. P-Net, where "MP" means max pooling and "Conv" means convolution [1].

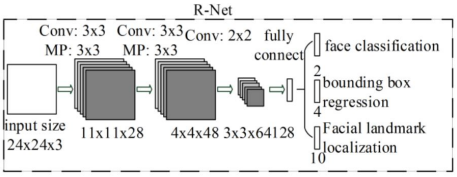

Fig. 3. R-Net, where "MP" means max pooling and "Conv" means convolution [1].

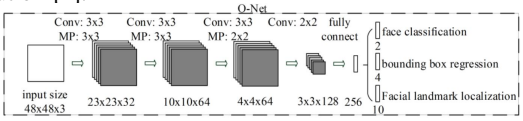

Fig. 4. O-Net, where "MP" means max pooling and "Conv" means convolution [1].

to generate regions and a CNN architecture, called Proposal Network (P-Net), to score each regions and return its bounding box regression vectors (see P-Net in Fig. 2). Each candidate region is given a score to represent its probability of being a face image and a threshold is set here to reject low-scoring regions. Besides, non-maximum suppression is applied to discard more candidate regions. More specifically, the intersection over unions (IoU's) of the highest-scoring region with other lower-scoring regions are calculated and lower-scoring regions with IoU's over a threshold are rejected. Then, we remove the highest-scoring region and select a new highest-scoring region to continue with previous steps. The returned bounding box regression vectors are used to refine the positions of bounding boxes and it contains the suggested positions.

Secondly, the selected candidate regions from the first stage are feed into another CNN architecture, Refine Network (R-Net, see details in Fig. 3), to further reject regions which are not face regions. R-Net is similar to P-Net, which also returns the scores and the bounding box regression vectors of input regions. As what is done in the fist stage, if a region has a score lower than a threshold, it will be rejected and non-maximum suppression and bounding boxes regression are applied to rejecting more regions and refining positions the left regions.

Finally, selected regions from the second stage are feed into a third CNN architecture, Output Network (O-Net, see details in Fig. 4). Its inputs and outputs are identical to those of R-Net. But this stage returns all selected regions as well as their facial landmarks. Notably, all of three CNN architectures return facial landmarks of each input but only facial landmarks from O-Net is used to do face recognition.

2) Training: To train three CNN architectures, we leverage three tasks: face/non-face classification, bounding box regression and facial landmark localization [1]. Face/non-face classification is a binary classification problem so the cross-entropy loss is used and for each sample x_i, the loss is expressed as:

$$L_i^{det} = -(y_i^{det} log(p_i) + (1 - y_i^{det}) log(1 - p_i)) \quad (1)$$

where p_i is the probability of a region being a face and $y_i^{det} \in 0, 1$ is the ground truth label. For bounding box regression, we use Euclidean loss for each sample x_i:

$$L_i^{box} = \|\hat{y}_i^{box} - y_i^{box}\|_2^2 \quad (2)$$

where $\hat{y}_i^{box} \in R^4$ is the predicted bounding box from the network, which has position of left top

layer	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
type	input	conv	relu	conv	relu	mpool	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	relu	mpool	conv
name	–	conv1_1	relu1_1	conv1_2	relu1_2	pool1	conv2_1	relu2_1	conv2_2	relu2_2	pool2	conv3_1	relu3_1	conv3_2	relu3_2	conv3_3	relu3_3	pool3	conv4_1
support	–	3	1	3	1	2	3	1	3	1	2	3	1	3	1	3	1	2	3
filt dim	–	3	–	64	–	–	64	–	128	–	–	128	–	256	–	256	–	–	256
num filts	–	64	–	64	–	–	128	–	128	–	–	256	–	256	–	256	–	–	512
stride	–	1	1	1	1	2	1	1	1	1	2	1	1	1	1	1	1	2	1
pad	–	1	0	1	0	0	1	0	1	0	0	1	0	1	0	1	0	0	1

layer	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37
type	relu	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	softmx
name	relu4_1	conv4_2	relu4_2	conv4_3	relu4_3	pool4	conv5_1	relu5_1	conv5_2	relu5_2	conv5_3	relu5_3	pool5	fc6	relu6	fc7	relu7	fc8	prob
support	1	3	1	3	1	2	3	1	3	1	3	1	2	7	1	1	1	1	1
filt dim	–	512	–	512	–	–	512	–	512	–	512	–	–	512	–	4096	–	4096	–
num filts	–	512	–	512	–	–	512	–	512	–	512	–	–	4096	–	4096	–	2622	–
stride	1	1	1	1	1	2	1	1	1	1	1	1	2	1	1	1	1	1	1
pad	0	1	0	1	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0

Fig. 5. Deep CNN architecture from [2]

corner, width and height of the bounding box and $y_i^{box} \in R^4$ is the ground truth bounding box. Facial landmark localization is also a regression problem so for each sample x_i, its Euclidean loss is

$$L_i^{landmark} = \|\hat{y}_i^{landmark} - y_i^{landmark}\|_2^2 \qquad (3)$$

where $\hat{y}_i^{landmark} \in R^{10}$ is the predicted facial landmark from the network, consists of ten coordinates of five facial key points, specifically, eyes, the nose as well as mouth corners and $y_i^{landmark}$ is the ground truth facial landmark.

B. Deep Face Recognition

VGG-face-recognition network [2] is one of the state-of-the-art deep CNN architectures for face recognition. It has 11 blocks in the network with totally 37 layers (see details in Fig. 5). The last block is used for face identification and the rest layers are used to extract features of faces. More concretely, when training the CNN architecture excluding the last layer, we try to minimize the Euclidean distance between features of images of one person and meanwhile maximize the Euclidean distance of images of different persons, which has a similar idea to word embeddings [21]. The last block is trained for face identification (multi-class classification) after face embeddings are gotten. Thus, the whole CNN architecture is designed for face identification but face embeddings (in the 34_{th} layer) can be used for face verification (binary classification). In our system, this CNN architecture will be applied to extracting features of detected faces and known faces.

C. Face Alignment

In order to improve the performance of face recognition, we align faces using facial landmarks returned by MTCNN. More specifically, the alignment method is similarity geometry transformation and we select positions of eyes to compute the transformation matrix. Known faces are used as references and detected faces are aligned to them. However, the weak face alignment method does not help to improve the performance and even reduce the recognition accuracy.

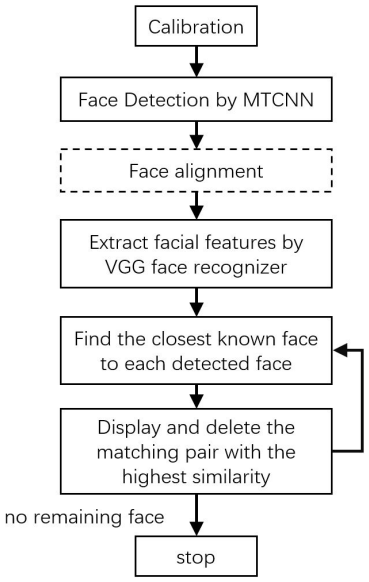

Fig. 6. The flowchart of our method

IV. IMPLEMENTATION AND RESULTS

In this section, we briefly introduce the implementation of our method and interpret results of experiments. Also, we analyze the problems in our method and provide solutions. All resulting images are shown in the appendix.

A. Implementation

The overall implementation is shown in Fig. 6. Firstly, we calibrate hyperparameters of MTCNN: minimal size of faces and thresholds for three CNNs of MTCNN. We use the frontal class image, i.e., anuclass01.JPG, to adjust hyperparameters. We leverage the detection accuracy and speed and finally get the minimal size of face images, 50×50, and thresholds, 0.8, 0.8 and 0.7 respectively. With these parameters, we run MTCNN on the remaining images to detect faces. Note that we directly borrow pre-trained models[1][2] of [1], [2].

According to the given face images, we can hardly determine whether two similar faces belong

[1]https://github.com/kpzhang93/MTCNN_face_detection_alignment
[2]http://www.vlfeat.org/matconvnet/pretrained/

to the same person because we do not have the ground truth and even if two faces are very similar, we cannot 100% assure that they belong to the same person. Therefore, instead of using images given by the lecturer, we use faces detected in anuclass01.JPG as known faces. If detected faces in other images have the same positions as those of known faces, we regard them as faces of the same persons. For example, a known face is located at the first seat on the lift side of the first row and a detected face is also located there so we regard them as faces of the same person.

Secondly, we try using face alignment to improve the face recognition accuracy. We use known faces as references and align eyes of detected faces with those of known faces by the similarity geometry transformation. Concretely, the transformation matrix can be gotten based on the positions of eyes in two images and this matrix can transform detected faces into new faces whose eyes are aligned with eyes of known faces.

After face alignment, we extract features of detected and known faces and use the cosine similarity to measure the similarity between different faces. The face embeddings are in the $34th$ layer of the VGG face recognizer and we need to normalize each face embedding to the unit length. For each detected face, there is a most similar known face but some known face may be the most similar face to multiple detected faces. Thus, to address this problem, we record and delete the matching pair with the highest similarity and calculate the similarity between the remaining known and detected faces until there is no remaining image.

B. Results of Face Detection

To evaluate the face detection, we use two images anuclass02.JPG and anuclass03.JPG as test images. We run our method on both images and evaluation of performances are shown in Table I. Among 59 faces, 56(94.92%) faces in the anuclass02.JPG are found and 58(98.31%) in the anuclass03.JPG. Our method achieves the detection precision of 100% on both images and the detection recall of 94.92% and 98.31% respectively, which means all detected regions are faces and almost all faces are detected.

TABLE I
PERFORMANCE OF OUR METHOD

	anuclass02.JPG	anuclass03.JPG
detection precision (%)	100	100
detection recall (%)	94.92	98.31
recognition accuracy without face alignment (%)	**82.14**	**72.41**
recognition accuracy with face alignment (%)	71.43	55.17

C. Results of Face Recognition

For face recognition without face alignment, 46(82.14%) faces are recognized correctly among 56 detected faces in the anuclass02.JPG and 42(72.41%) faces are correct among 58 detected faces in the anuclass03.JPG. Adding weak face alignment before face recognition does not improve the recognition accuracy (see results in the table I).

V. DISCUSSION

According to the results, both of MTCNN and the VGG face recognizer performs well but their complexity is so high that it takes around 4s for MTCNN to process an image and around 1s for the VGG face recognizer to extract the feature of a face image. Besides, our method lack effective face alignment, which reduces the recognition accuracy. We found the weak face alignment used in the experiment could improve the recognition accuracy when using the anufaces dataset as known faces (we will not show results here). However, it does not work here. One possible reason is that most of unaligned faces are recognized correctly and the weak face alignment uses only positions of eyes and a very simple geometry transformation, which adds noise into aligned images and reduce the recognition accuracy. Another problem is faces, which do not have counterparts in the known faces, cannot be recognized correctly. We attempted to give a "unknown" label to these faces by thresholding the largest similarity between them and known faces but failed. These faces can be very similar to some unknown face.

To address the problem of efficiency, the best way is to simplify methods because neural-network-based methods are well-known for their complexity. For example, we can try replacing multiple steps in MTCNN with a CNN and using

a shallow CNN architecture rather than the deep VGG face recognizer. Improving the performance of devices is also possible but costly. For face alignment, we can try a bit more complex methods, such as the projective transformation with more facial landmarks. Labeling faces without correspondences in the known faces can possibly be solved if we can align faces. Because no people in our class have similar appearances and different aligned faces tend to have a low similarity, thresholding can be used to determine whether a detected face corresponding to a known face.

VI. Conclusion

In this project, we use MTCNN and the VGG face recognizer to detect and recognize faces respectively and to address the problem that a known face may be the most similar faces of several detected faces, we iteratively select the matching pair with the highest similarity. Our method has high detection and recognition accuracies but it processes images slow due to its high complexity. It lacks an effective face alignment to improve the recognition accuracy and cannot deal with detected faces which do not have corresponding known faces.

VII. Learning Outcomes

1. The advantages of disadvantages of early and current methods. 2. How to use MatConvNet and MatCaffe. 3. It is necessary to combines different kinds of methods, such as image processing and machine learning, to detect and recognize faces.

References

[1] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, 2016.

[2] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition." in *BMVC*, vol. 1, no. 3, 2015, p. 6.

[3] P. Viola and M. J. Jones, "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[4] S. S. Farfade, M. J. Saberian, and L.-J. Li, "Multi-view face detection using deep convolutional neural networks," in *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*. ACM, 2015, pp. 643–650.

[5] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[6] Y. Sun, X. Wang, and X. Tang, "Deep learning face representation from predicting 10,000 classes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1891–1898.

[7] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Advances in neural information processing systems*, 2014, pp. 1988–1996.

[8] Y. Sun, D. Liang, X. Wang, and X. Tang, "Deepid3: Face recognition with very deep neural networks," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[9] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 815–823.

[10] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.

[11] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[12] M. Mathias, R. Benenson, M. Pedersoli, and L. Van Gool, "Face detection without bells and whistles," in *European Conference on Computer Vision*. Springer, 2014, pp. 720–735.

[13] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[14] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

[15] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1440–1448.

[16] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[17] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1701–1708.

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

[19] S. Yang, P. Luo, C.-C. Loy, and X. Tang, "Wider face: A face detection benchmark," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5525–5533.

[20] M. Köstinger, P. Wohlhart, P. M. Roth, and H. Bischof, "Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2144–2151.

[21] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

VIII. Appendix

A. Results of Face Detection and Recognition

Fig. 7 and 8 show face detection and recognition results on anuclass02.JPG. Fig. 9 and 10 show face detection and recognition results on anuclass03.JPG. It is clear that results with weak face alignment are worse.

Fig. 7. The face detection result on anuclass02.JPG(without face alignment)

Fig. 8. The face detection result on anuclass02.JPG(with face alignment)

Fig. 9. The face detection result on anuclass03.JPG (without face alignment)

Fig. 10. The face detection result on anuclass03.JPG(with face alignment)