

Research Article

Real Time Eye Detector with Cascaded Convolutional Neural Networks

Bin Li^{1,2,3} and Hong Fu³ 

¹*Xian Institute of Optics and Precision Mechanics of CAS, Xian 710119, China*

²*University of Chinese Academy of Sciences, Beijing 100049, China*

³*Chu Hai College of Higher Education, Tuen Mun, Hong Kong*

Correspondence should be addressed to Hong Fu; hongfu@chuhai.edu.hk

Received 12 January 2018; Revised 12 March 2018; Accepted 14 March 2018; Published 22 April 2018

Academic Editor: Erich Peter Klement

Copyright © 2018 Bin Li and Hong Fu. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

An accurate and efficient eye detector is essential for many computer vision applications. In this paper, we present an efficient method to evaluate the eye location from facial images. First, a group of candidate regions with regional extreme points is quickly proposed; then, a set of convolution neural networks (CNNs) is adopted to determine the most likely eye region and classify the region as left or right eye; finally, the center of the eye is located with other CNNs. In the experiments using GI4E, BioID, and our datasets, our method attained a detection accuracy which is comparable to existing state-of-the-art methods; meanwhile, our method was faster and adaptable to variations of the images, including external light changes, facial occlusion, and changes in image modality.

1. Introduction

In recent years, eye detection has become an important research topic in computer vision and pattern recognition [1, 2], because the human eyes locations are essential information for many applications, including psychological analysis, facial expression recognition, auxiliary driving, and medical diagnosis [3]. However, eye detection is quite challenging in many practical applications. The cameras are sensitive to light variations and the shooting distance, which makes the human eyes very eccentric in a facial image. Sometimes the face is partially occluded and we are not able to obtain a complete facial image. For example, half of the face was covered in a cover test for detecting squint eyes [4]. In this case, some existing eye detection methods do not work, because they rely on a facial model detection to locate the eyes. An eye detector is also expected to work well in various image modalities, that is, infrared and visible images. Moreover, the eye detection algorithm should be fast because it is supposed to be online in many practical cases. Although many methods have been proposed to detect the eyes from facial images, it is difficult

to find one method that performs well in terms of accuracy, robustness, and efficiency. Therefore, we are attempting to develop an efficient and robust eye detection algorithm to fulfill the requirements of the applications as much as possible.

The rest of this paper is organized as follows. In the Related Work, a review of related works is presented. In the Method, we proposed an efficient method for eye detection, which consists of candidate region generation, eye region determination and classification, and eye center positioning. Then, the training scheme, evaluation results, and discussions were presented in the Training Scheme, Evaluation, and Further Discussion, respectively. Finally, conclusion remarks were drawn in the last section.

2. Related Work

Many algorithms have been proposed for image-based eye detection. These can be summarized into two categories, namely, traditional eye detectors and machine learning eye detectors.

Traditional eye detectors are typically designed according to the geometric characteristics of the eye. These eye detectors can be divided into two subclasses. The first subclass is the geometric model. Valenti and Gevers [5] used the curvature of isophotes to design a voting system for eye and pupil localization. Markuš et al. [6] proposed a method for eye pupil localization based on an ensemble of randomized regression trees. Timm and Barth [7] proposed the use of image gradients and squared dot products to detect the pupils. The second subclass is the template matching. The RANSAC [8] method was used to create an elliptic equation to fit the pupil center. Araujo et al. [9] described an Inner Product Detector for eye localization based on correlation filters. The traditional eye detectors sometimes can achieve good results, but they easily fail when there is a change in the external light or face occlusion.

Machine learning eye detectors can also be further divided into two subclasses. The first subclass is the traditional feature extraction followed by a cascaded classifier. Chen and Liu [10] applied the Haar wavelet and support vector machine (SVM) for fast classification. Sharma and Savakis [11] proposed learning histogram of oriented gradients (HOG) features in combination with SVM classifiers to obtain an efficient eye detector. Leo et al. [12, 13] used self-similarity information combined with shape analysis for eye center detection. A constraint based on the geometrical characteristics and neural classifier to detect the eye regions were proposed in [14]. Gou et al. [15] built a cascade regression framework for simultaneous eye localization and eye state estimation. Kim et al. [16] proposed generating eye candidate regions by using multiscale iris shape features and then verifying those candidate regions using the HOG and cell mean intensity features. With the popularity of deep learning algorithms [17], some researchers used the convolution neural network to train eye detectors, which forms the second subclass. Chinsatit and Saitoh [18] present a CNN-based pupil center detection method. In Fuhl's [19] research, coarse to fine pupil position identification was carried out using two similar convolutional neural networks and the authors proposed subregions from a downsampled input image to decrease computational costs. Amos et al. [20] trained a facial landmark detector using 68 feature points to describe the face model in which 12 feature points described the eye contour. The deep learning based methods have shown high robustness and detection accuracy compared with traditional methods. However, the efficiency is still an issue. The facial images are usually larger than 640×480 . This requires a large amount of computer resources if the CNNs have to perform a global search of the image. A quick and effective method is necessary to propose candidate regions such that only the selected candidate regions are fed into the CNNs.

Besides the determination of eye regions, the classification of the left/right eye and the positioning of the eye center are also important for some applications, such as eye tracking system and eye disease detection. However, most of the existing eye detectors cannot efficiently determine the eye regions, distinguish between left or right eyes, and detect the eye center in one round. Therefore, we aim to propose a new method that overcomes the disadvantages of the existing methods.

3. Method

The overall workflow for the proposed method is shown in Figure 1. In the first step, we calculated the local extreme points and the gradient values in the full facial image; then a number of candidate eye regions were quickly generated, taking these feature points as the centers. In the second step, these candidate eye regions were evaluated by the 1st set of CNNs to determine the eye regions and the eye class (left or right). In the third step, the 2nd CNNs were employed to locate the eye center. In the following sections, we will introduce each step and the CNNs structures in detail.

3.1. Candidate Region Generation. Directly using CNNs on the high resolution face images (e.g., 1280×720) would require a large amount of computing resources. In order to reduce the time consumed, some existing methods [15, 16] used Viola and Jones' face detector [21] to detect face region. However, due to light change, occlusion, and other factors, sometimes the face detector cannot detect the face region accurately. This will directly affect the accuracy of the eye detection algorithm. The face detection rates on BioID, GI4E, and our datasets are 97.5%, 99.4% [15], and 38.6% (only the top half of the face is photographed in the images of our dataset, as shown in Figure 6(c); the face detection fails in these images), respectively. That is why we avoid using face detection, although it helps reduce the number of candidate regions. Thus, we need to quickly propose the valid eye candidate regions that can significantly reduce the search space of the accurate eye location. In our observation, we found that the pupil and iris were darker than other parts of the eye. The locations of the local extreme points in the image are more likely to be the rough center positions of the eyes. To find these extreme points, we convolved the facial image $I(x, y)$ with three Gaussian kernels of different variances to obtain Gaussian images $G_n(x, y, \sigma_n)$ ($n = 1, 2, 3$). Each pixel in $G_2(x, y, \sigma_2)$ was compared with its $5 \times 5 \times 3$ neighborhood pixels in $G_1(x, y, \sigma_1)$, $G_2(x, y, \sigma_2)$, and $G_3(x, y, \sigma_3)$. If the pixel (x, y) is the maximum or minimum in its neighborhood, its local gradient value $Gr(x, y)$ was calculated as follows:

$$\begin{aligned} Gr(x, y) &= \sqrt{(G_2(x+1, y, \sigma_2) - G_2(x-1, y, \sigma_2))^2 + (G_2(x, y+1, \sigma_2) - G_2(x, y-1, \sigma_2))^2}, \\ G_n(x, y, \sigma_n) &= \frac{1}{2\pi\sigma_n^2} \exp\left(-\frac{x^2 + y^2}{2\sigma_n^2}\right) * I(x, y), \end{aligned} \quad (1)$$

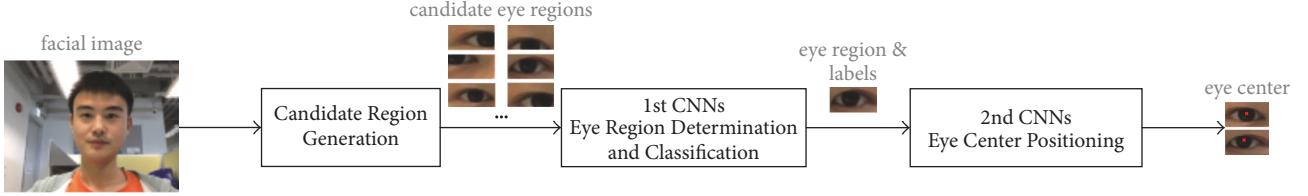
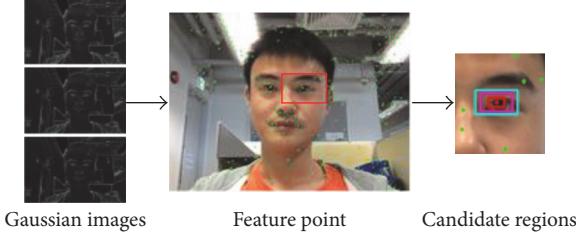


FIGURE 1: Workflow of the proposed method.

FIGURE 2: Top N candidate feature points and the candidate eye regions. Red, magenta, and cyan boxes stand for candidate regions of different scales; the length of each candidate box is twice the width.

where $G_n(x, y, \sigma_n)$ is the convolution of the Gaussian kernel and the facial image.

We selected the top N extreme points with the largest gradient value as the candidate feature points. The selection of parameter N will be discussed in detail in the Training Scheme. We aim to ensure that the candidate regions can completely cover the eye region and make the number of candidate feature points as small as possible. Then, we generated three different sizes of candidate eye regions R_{iC_m} ($i = 1, 2, 3$) centered on each candidate feature point C_m ($m = 1, 2, 3, \dots$), which ensures that the generated candidate eye regions can completely cover the eye region as shown in Figure 2.

3.2. Eye Region Determination and Classification. After generating the candidate eye regions, we aimed to develop a method that can quickly and robustly predict eye location and classify the eye as left eye or right eye. We developed a set of convolutional neural networks (CNNs) to make effective use of the datasets. The core architecture of our 1st set of CNNs is summarized in Figure 3. Since ours generated candidate regions with different scales, the three different labeled candidate regions were inputted to the CNN model separately.

In our 1st set of CNNs, three sub-CNNs were built and each carries the same structure. In each sub-CNNs, the first layer was a convolutional layer with a kernel size of 5×5 pixels, two pixel strides, and one padding, and the convolution layer was followed by a maximum pooling layer with a window size of 3×3 and two pixel strides. The second layer was a convolutional layer with a kernel size of 3×3 pixels, one pixel stride, one padding, and no pooling layer. The third layer was similar to the first layer, except that the

convolutional kernel size was 3×3 pixels. Through three stages of convolution and pooling, in which the convolutional layer learned the edges, eye structure, and other basic features, the pooling layer helped the networks to be robust to details in the changes. Next, we used fully connected (FC) layers to combine a deeper knowledge and produce the final region label and confidence index of each candidate region. Finally, we choose the candidate region with the maximum index as the eye region and according the region label to classify the left or right eyes. We then used the coordinates of this region and restored it to the original facial image. All CNNs' weights were initialized based on ImageNet's [22] weight values for fine tuning, which will help us to train the network with faster convergence and obtain good experimental results.

3.3. Eye Center Positioning. Although the 1st set of CNNs outputs the eye region and the eye class (left or right), it still lacks precise positioning of the eye center. The existing eye detectors usually treat the center of the eye region as the eye center, which is inaccurate if the subject is not looking straightwards. Some cases are shown in Figure 4.

To locate the actual eye center, we built the 2nd set of CNNs that locates the pupil region in the eye region. The 2nd CNNs' architecture is shown in Figure 5. Compared to the 1st set of CNNs, the structure of the 2nd network is relatively simple. It is composed of a convolution layer, an average pooling layer, a fully connected layer, and a logistic perceptron. The input eye region for this set of CNNs depends on the first CNNs' output. The size of the other layers was adapted accordingly, and we chose the center of pupil region as the eye center.

4. Training Scheme

4.1. Datasets. In our experiments, we used the GI4E [23] public dataset, which contains 103 test subjects with 1,236 visible facial images and the BioID [24] dataset, which consists of 1,521 frontal face images with significant variation in illumination and head pose. In addition, we built our own infrared/visible facial image datasets from 42 subjects when they were going through a strabismus examination. In this examination, we randomly covered their left or right eyes to check if they have strabismus. For each subject, we collected 20 facial images with different pupil positions. This variation allowed us to train robust models that generalized well with novel faces. Some image samples are shown in Figure 6. The

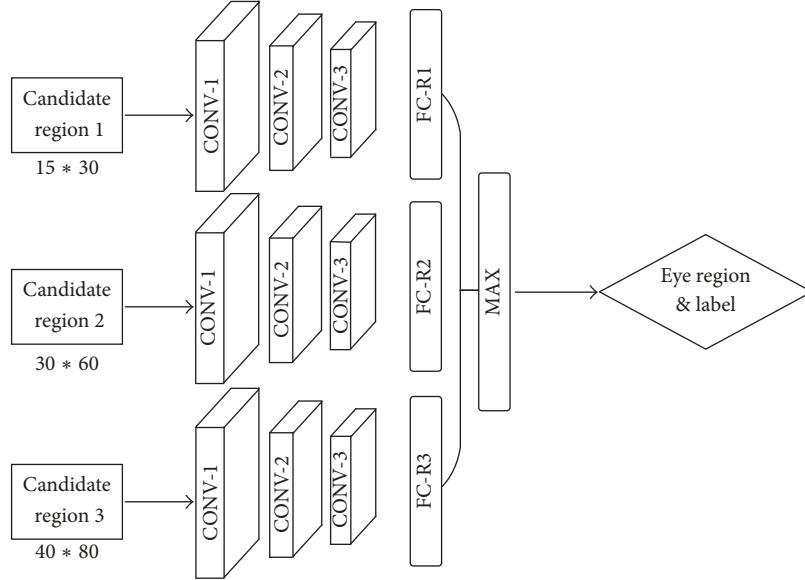


FIGURE 3: Structure of the 1st set of CNNs.

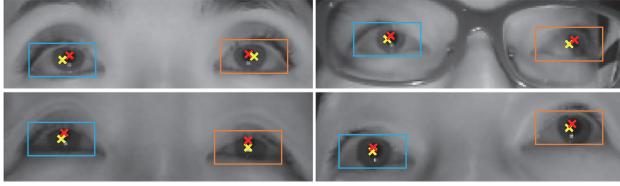


FIGURE 4: Detected eye regions, region centers, and actual eye centers. The yellow cross represents the center of the eye region and the red cross is the actual center of the eye.

TABLE 1: The percentage of images with at least one effective feature point versus the parameter N .

N	100	150	200	250	300
Percentage of images	60%	85%	96%	97%	97%

image resolution was 800×600 , 384×286 , and 1280×720 pixels.

4.2. Feature Points Selection. In our work, we set $\sigma_1 = 0.6$, $\sigma_2 = 1.0$ and $\sigma_3 = 1.4$. In order to increase the search efficiency, we only select the top N candidate feature points. We set N to 100, 150, 200, 250, and 300 and counted the percentage of images with at least one effective feature point, as shown in Table 1 (here “an effective feature point” means that the region centered at this point is a real eye region). We can find that when $N = 200$, 96% of the images in the datasets have at least one effective feature point. Moreover, the number of valid images does not increase significantly when N increases. Therefore, we selected the top 200 points with the largest gradient values as the eye candidate feature points.

4.3. Training. For each selected candidate point, we generated three candidate boxes as shown in Figure 7. In our

algorithm, the sizes of the proposed candidate boxes were 15×30 , 30×60 , and 40×80 , respectively, to ensure that the eye was fully covered by at least one candidate region. We manually labeled the left eye or the right eye region in each image as R_H . These regions were used as positive training samples. To generate more training samples, we evaluated the overlapped area I between the candidate regions R_{iC_m} and R_H . If I is greater than a threshold T and this eye is the left eye, we marked this region as label 1. If I is greater than a threshold T and this eye is the right eye, we marked this region as label 2. If I is smaller than a threshold T , we marked this region as label 0.

$$\frac{R_{iC_m} \cap R_H}{R_{iC_m} \cup R_H} = I \begin{cases} > T, & \begin{cases} \text{left,} & \text{label} = 1, \\ \text{right,} & \text{label} = 2, \end{cases} \\ < T, & \text{label} = 0. \end{cases} \quad (2)$$

Next, we put the marked regions into the 1st set of CNNs, resulting in three indexes between 0 and 1 as shown in Figure 7. These indexes represent the confidence of the CNNs that the eye is within the candidate box. Thus, the candidate box of the highest index was selected as the eye region. Then this eye region was fed into the second set of CNNs which outputs the accurate eye center. We used global search within the eye region to improve the accuracy of eye center detection.

5. Evaluation

The evaluation of our method was carried out on an Intel(R) Core(TM) i5-6600 desktop computer with 16 GB RAM and NVIDIA GeForce GTX 745 GPU. This algorithm was implemented using MATLAB (R2016a). During the evaluation, the two stages of cascaded CNNs’ output were discussed separately. We evaluated our method on the GI4E, the BioID,

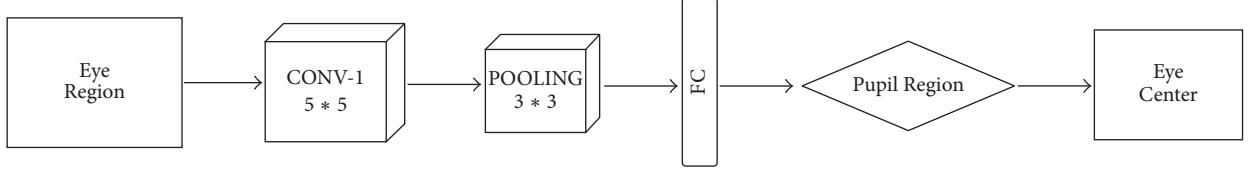


FIGURE 5: Structure of the 2nd CNNs.

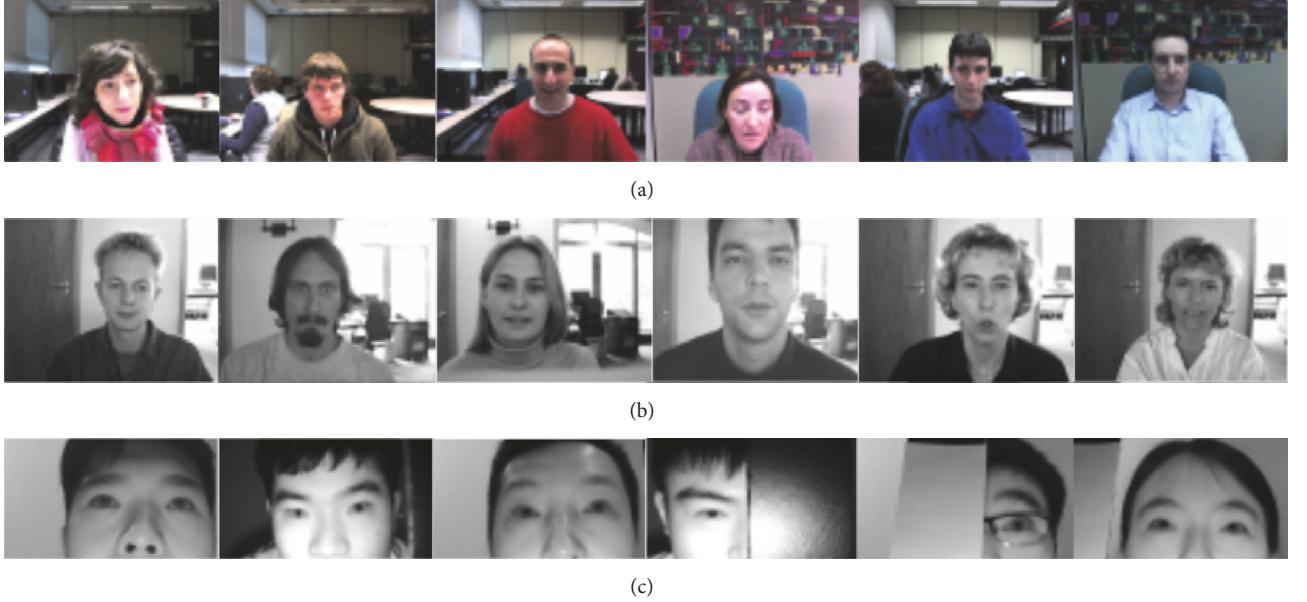


FIGURE 6: Samples of images in the datasets. (a) shows samples from GI4E datasets and (b) is from BioID datasets, while (c) is from our datasets. Note the significant variation in illumination, head position, infrared/visible image, and full/half covered face.

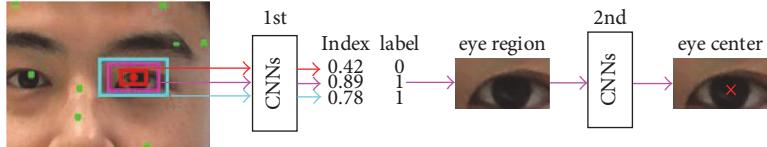


FIGURE 7: CNNs' output. The green points represent the selected eye candidate feature points. The indexes show the confidences and the regions predicted by the first set of CNNs. The magenta rectangle was selected as the final prediction result because it had the highest index, 0.89, and this eye was predicted to be the left eye. The eye region was fed into the second set of CNNs outputting the accurate eye center. The red cross represents the eye center.

and our facial datasets. We randomly divided the datasets into two parts for the CNNs' training and testing.

5.1. Eye Detection and Classification. Figure 8 shows some qualitative results of the eye regions detection and classification by our proposed 1st set of CNNs output and the Amos et al. [20] method. Amos et al. method relies on face detection to locate the eyes. It works well when the whole face is included in the image. However, it fails if the face is occluded, for example, the last image. On the contrary, our method works well even if the face is occluded. This proves that our approach can successfully detect the eye locations even when the face cannot be detected.

To measure the eye classification and detection capability, we defined the normalized error as

$$N_{\text{error}} = \frac{\max(d_l, d_r)}{d_{l-r}}, \quad (3)$$

where d_l and d_r are the Euclidean distances between the ground truth and the positions of the calculated left and right eyes, and d_{l-r} is the Euclidean distance between the ground truth of the left and right eyes. We compared our method with state-of-the-art methods that have been applied to the BioID dataset for a discretized $N_{\text{error}} \in \{0.05, 0.1, 0.25\}$ and measured the average time to process one image.

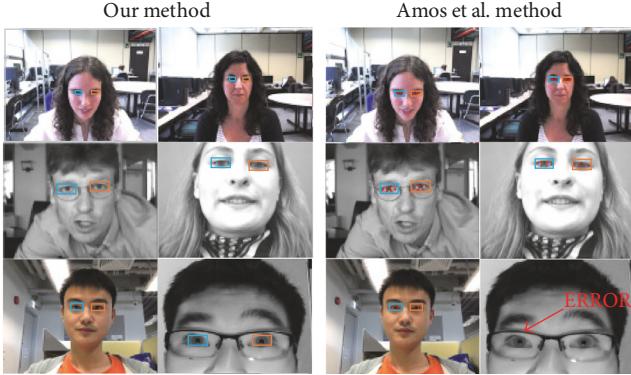


FIGURE 8: Some results of eye location and classification. The blue box represents the right eye while the orange box represents the left eye.

The comparison between the proposed method and the state-of-the-art methods is presented in Table 2. Note that when $N_{\text{error}} < 0.05$, our method (85.6%) does not perform as well as Valenti and Gevers (86.1%), Araujo et al. (88.3%), and Gou et al. (91.2%) methods. However, our proposed method does not require clustering and is robust even if the tester wears glasses. Our algorithm is stabler than Valenti and Gevers's and Araujo et al.'s method. In Gou et al.'s method, an image is not taken into account if the face detector does not locate the face. On the contrary, we run our method on all the images in the dataset. When $N_{\text{error}} < 0.25$, the detection accuracy of ours (99.5%) is comparable to Timm and Barth (99.7%), Markuš et al. (99.7%), and Gou et al. (99.8%) methods, but the Markuš approach also relies on face detection accuracy. Although Leo et al.'s method does not require face detection, in the case of eyeglasses reflection and special head poses, the detection accuracy and efficiency are lower. Furthermore, our method has significantly less computational complexity and achieves the best performance by far (13 ms to process one image).

5.2. Eye Center Positioning. Figure 9 shows some results of eye detection, classification, and eye center positioning by our proposed method on GI4E, BioID, and our datasets, where the blue rectangle represents the right eye region, the orange rectangle denotes the left eye region, and the red cross shows the estimated eye center. Even though the subject is with glasses, we can still estimate the eye location by the cascaded CNNs.

Figure 10 reports some facial images of each dataset in which the proposed method fails in detection, classification of the eyes, or eye center estimation. In most cases, this is because of eye shadows, strong light on the glasses, and eyelid occlusion where the proposed method cannot accurately locate the eye regions and pupil centers.

To verify the effectiveness of our proposed method, we also reported our results in terms of the average eye center detection rate as a function of the pixel distance correctly between the algorithmically established and hand-labeled eye location. Figure 11 shows a plot of the eye center detection performance of our algorithm on different dataset. Our method

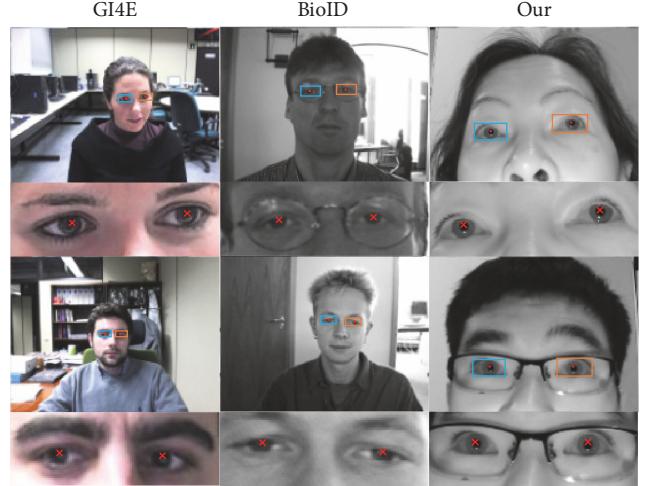


FIGURE 9: Samples of eye detection, classification, and eye center estimation results on BioID, GI4E, and our datasets. Red cross represents estimated eye center.

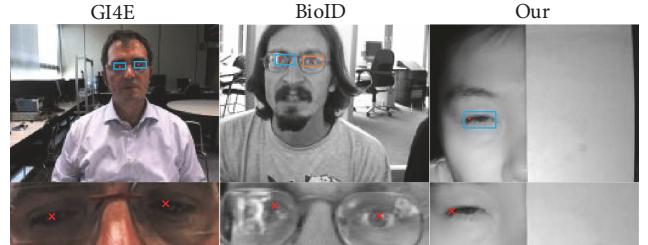


FIGURE 10: Some samples from the BioID, GI4E, and our datasets in which the proposed method fails in the eye classification and eye center detection.

has a preferable eye center detection rate, over 90%, on each dataset within an error threshold of ten pixels. Compared with result in testing on BioID and our datasets, results on GI4E are better since the images have a higher resolution and smaller changes in illumination.

5.3. Further Discussion. The experimental results demonstrate that our proposed method can achieve satisfactory result on both eye region and eye center detection on benchmark datasets. Based on the cascaded CNNs framework, we not only improved the eye detection speed, but also reduced training time. Actually, if we only used a single network to perform both eye region detection and eye center positioning, it may get a better result. But it must require a bigger dataset and is time-consuming. For example, it took 3 days to train 500 images (1280×720) in our dataset using Faster R-CNN [26]. However, our proposed method only needed 4 hours. In addition, most of the existing methods rely on the accuracy of the face detector and are hard to apply in practice.

In our experimental environment, the eye regions and centers were located in approximately 9 ms, wherein approximately 2 ms was spent in proposing the eye candidate feature points while 2 ms was spent in calculating the accurate region of the left eye or right eye (1st set of CNNs), and 30 ms was

TABLE 2: Eye localization and classification comparison on the BioID datasets.

Method	$N_{\text{error}} < 0.05$	$N_{\text{error}} < 0.1$	$N_{\text{error}} < 0.25$	Time (ms)
Valenti and Gevers [5]	86.1%	91.7%	97.9%	*
Timm and Barth [7]	82.5%	93.4%	99.7%	*
Amos et al. [20]	84.1%	90.2%	96.1%	500
Araujo et al. [9]	88.3%	92.7%	98.9%	83
Asadifard and Shanbezadeh [25]	47.0%	86.0%	96.0%	45
Markuš et al. [6]	85.7%	95.3%	99.7%	28
Leo et al. 2013 [12]	78.0%	86.0%	90.0%	330
Leo et al. 2014 [13]	80.6%	87.3%	94.0%	330
Gou et al. [15]	91.2%	99.4%	99.8%	67
Ours	85.6%	95.9%	99.5%	13

* indicates that we cannot measure the processing time.

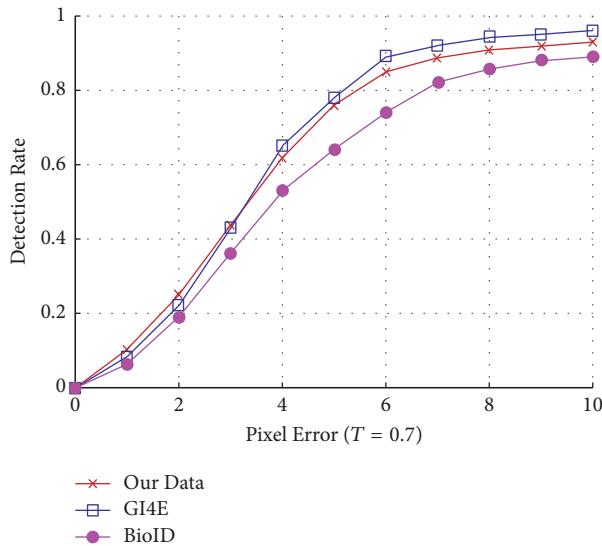


FIGURE 11: Eye center detection performance of our method on different datasets.

spent in locating the center of the eye (2nd CNNs). The frame rate is around 30–60 fps for most eye detection tasks. This shows that our proposed method can deal with real time problems that arise in realistic scenarios.

6. Conclusions

In this paper, we presented an effective cascaded CNNs method to detect the eye location in facial images. Our method can simultaneously detect left and right eye locations and center even when the face was blocked and is insensitive to visible or infrared light images. In addition, eye positioning does not rely on the face detector. For the evaluation, we tested our method using over 5,000 facial images and found that our proposed eye detector was efficient and effective. We used features points combined with the cascaded CNNs in order to achieve significantly high efficiency and satisfactory classification rate. In our future work, we plan to collect more facial images to train more powerful eye recognition models.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The work presented in this paper was fully supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region (Reference no. UGC/FDS13/E04/14).

References

- [1] H. Fu, Y. Wei, F. Camastra, P. Arico, and H. Sheng, “Advances in Eye Tracking Technology: Theory, Algorithms, and Applications,” *Computational Intelligence and Neuroscience*, vol. 2016, Article ID 7831469, 2016.
- [2] L. Zhang, Y. Cao, F. Yang, and Q. Zhao, “Machine Learning and Visual Computing,” *Applied Computational Intelligence and Soft Computing*, vol. 2017, Article ID 7571043, 2017.
- [3] H. Mosa, M. Ali, and K. Kyamakya, “A computerized method to diagnose strabismus based on a novel method for pupil segmentation,” in *Proceedings of the International Symposium on Theoretical Electrical Engineering*, 2013.
- [4] L. Birgit, “Pediatric Ophthalmology, Neuro-ophthalmology, Genetics: Strabismus-new Concepts in Pathophysiology, Diagnosis, and Treatment,” in *Pediatric Ophthalmology, Neuro-ophthalmology, Genetics: Strabismus-new Concepts in Pathophysiology, Diagnosis, and Treatment*, M. C. Brodsky, Ed., Springer Science Business Media, and Treatment, 2010.
- [5] R. Valenti and T. Gevers, “Accurate eye center location through invariant isocentric patterns,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 9, pp. 1785–1798, 2012.
- [6] N. Markuš, M. Frljak, I. S. Pandžić, J. Ahlberg, and R. Forchheimer, “Eye pupil localization with an ensemble of randomized trees,” *Pattern Recognition*, vol. 47, no. 2, pp. 578–587, 2014.
- [7] F. Timm and E. Barth, “Accurate eye centre localisation by means of gradients,” *VisappII*, pp. 125–130, 2011.
- [8] L. Świrski, A. Bulling, and N. Dodgson, “Robust real-time pupil tracking in highly off-axis images,” in *Proceedings of the 7th Eye Tracking Research and Applications Symposium, ETRA 2012*, pp. 173–176, USA, March 2012.

- [9] G. M. Araujo, F. M. L. Ribeiro, E. A. B. Silva, and S. K. Goldenstein, "Fast eye localization without a face model using inner product detectors," pp. 1366–1370.
- [10] S. Chen and C. Liu, "Eye detection using discriminatory Haar features and a new efficient SVM," *Image and Vision Computing*, vol. 33, pp. 68–77, 2015.
- [11] R. Sharma and A. Savakis, "Lean histogram of oriented gradients features for effective eye detection," *Journal of Electronic Imaging*, vol. 24, no. 6, Article ID 063007, 2015.
- [12] M. Leo, D. Cazzato, T. De Marco, and C. Distante, "Unsupervised approach for the accurate localization of the pupils in near-frontal facial images," *Journal of Electronic Imaging*, vol. 22, no. 3, Article ID 033033, 2013.
- [13] M. Leo, D. Cazzato, T. De Marco, and C. Distante, "Unsupervised eye pupil localization through differential geometry and local self-similarity matching," *PLoS ONE*, vol. 9, no. 8, Article ID e102829, 2014.
- [14] T. D'Orazio, M. Leo, and A. Distante, "Eye detection in face images for a driver vigilance system," in *Proceedings of the 2004 IEEE Intelligent Vehicles Symposium*, pp. 95–98, ita, June 2004.
- [15] C. Gou, Y. Wu, K. Wang, K. Wang, F.-Y. Wang, and Q. Ji, "A joint cascaded framework for simultaneous eye detection and eye state estimation," *Pattern Recognition*, vol. 67, pp. 23–31, 2017.
- [16] H. Kim, J. Jo, K.-A. Toh, and J. Kim, "Eye detection in a facial image under pose variation based on multi-scale iris shape feature," *Image and Vision Computing*, vol. 57, pp. 147–164, 2017.
- [17] D. Xie, L. Zhang, and L. Bai, "Deep Learning in Visual Computing and Signal Processing," *Applied Computational Intelligence and Soft Computing*, vol. 2017, pp. 1–13, 2017.
- [18] W. Chinsatit and T. Saitoh, "CNN-Based Pupil Center Detection for Wearable Gaze Estimation System," *Applied Computational Intelligence and Soft Computing*, vol. 2017, pp. 1–10, 2017.
- [19] W. Fuhl, *PupilNet: convolutional neural networks for robust pupil detection*, arXiv preprint arXiv:1601.04902, convolutional neural networks for robust pupil detection, PupilNet, 2016.
- [20] B. Amos, B. Ludwiczuk, and M. Satyanarayanan, *Openface: A general-purpose face recognition library with mobile applications*, CMU School of Computer Science, Openface, 2016.
- [21] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.
- [22] J. Deng, W. Dong, and R. Socher, "ImageNet: a large-scale hierarchical image database," in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, Miami, Fla, USA, June 2009.
- [23] A. Villanueva, V. Ponz, L. Sesma-Sánchez, M. Ariz, S. Porta, and R. Cabeza, "Hybrid method based on topography for robust detection of iris center and eye corners," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 9, no. 4, article 25, 2013.
- [24] O. Jesorsky, K. J. Kirchberg, and R. W. Frischholz, "Robust face detection using the hausdorff distance," in *Audio- and Video-Based Biometric Person Authentication: Third International Conference, AVBPA 2001 Halmstad, Sweden, June 6–8, 2001 Proceedings*, J. Bigun and F. Smeraldi, Eds., vol. 2091 of *Lecture Notes in Computer Science*, pp. 90–95, Springer, Berlin, Germany, 2001.
- [25] M. Asadifard and J. Shanbehzadeh, "Automatic adaptive center of pupil detection using face detection and CDF analysis," in *Proceedings of the International MultiConference of Engineers and Computer Scientists 2010, IMECS 2010*, pp. 130–133, hkg, March 2010.
- [26] R. Shaoqin, "Faster R-CNN: Towards real-time object detection with region proposal network," in *Advances in Neural Information Processing Systems*, pp. 91–99, Faster R-CNN, Towards real-time object detection with region proposal network, 2015.

