A Robust Eyeball Detection based on Computer Vision Approaches

R. Priyanka [1], S. Sridhar [2], R.Surya [3], J. Vimal James [4], P. Kavitha [5]

[1] Assistant Professor, [2, 3, 4] B.E. Student, [5] Associate Professor
Department of Computer Science and Engineering, P.S.R. Engineering College,
Sivakasi, TamilNadu, India.

Published in IJIRMPS (E-ISSN: 2349-7300), Volume 12, Issue 1, (January-February 2024)

Abstract

Eye contact is among the most primary means of social communication used by humans. Quantification of eye contact is valuable as a part of the analysis of social roles and communication skills, and for clinical screening. Estimating a subject's looking direction is a challenging task, but eye contact can be effectively captured by a wearable point-of-view camera which provides a unique viewpoint. While moments of eye contact from this viewpoint can be hand-coded, such a process tends to be laborious and subjective. In this work, we develop a deep neural network model to automatically detect eye contact in egocentric video. It is the first to achieve accuracy equivalent to that of human experts. We train a deep convolutional network using a dataset of 4,339,879 annotated images, consisting of 103 subjects with diverse demographic backgrounds. The network achieves overall precision of 0.936 and recall of 0.943 on 18 validation subjects, and its performance is on par with 10 trained human coders with a mean precision 0.918 and recall 0.946. We used a media pipe package for iris movement such as left, right and center. And also, the eye close and open feature.

Key words: Residual Network, Rectified Linear Unit, Exponential Linear Units, Artificial Neural Network

1. Introduction

Eyes and their movements are important in expressing a person's desires, needs and emotional states. The significance of eye movements with regards to the perception of and attention to the visual world is certainly acknowledged since it is the means by which the information needed to identify the characteristics of the visual world is gathered for processing in the human brain. Hence, robust Iris Tracking and Gaze Tracking are considered to play a crucial role in the development of human-computer interaction, creating attentive user interfaces and analyzing human affective states. Iris tracking is widely investigated as alternative interface methods. They can be used as a base to develop an Iris tracking system which achieves the highest accuracy, best performance and lowest cost. There are many proposed approaches. Some approaches may be implemented using low computational hardware such as a micro-controller due to the simplicity of the used algorithm. 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.

2. Related Work

Face Recognition and Eye Detection using Python

It contains a technique for eye detection and face recognition using morphological image processing by Python OpenCV. Their will be facial land-marking for different object but specially in this paper for eye and face. It is observed that their is different number land marks points for each region. The low luminous, high density which are the characteristic of eye as compare to rest all parts of face. Proposed method uses Haar classifier technique with additional Python programming efficiency. This results in detection of face and eye in fraction of second and with greater accuracy. This technique used is really highly efficient and accurate for detecting face and eyes.

An Introduction to Eye Tracking in Human Factors Healthcare Research and Medical Device Testing

Eye tracking is a powerful and sophisticated tool that provides an objective glimpse into the cognition of health-care providers, patients, caregivers, and medical device users. Insights gleaned from eye tracking can be harnessed to better understand – and ultimately improve – the dynamics of healthcare, which quite literally has the potential to save lives. Nonetheless, the use of eye tracking within healthcare research and medical device testing remains in its infancy, which at least partly reflects the learning curve that it demands. As such, the central aim of this article is to provide an easily digestible primer for healthcare researchers and practitioners interested in first getting started with eye tracking. The discussion offers a general overview of how it works, device types and notable specifications, a taxonomy of common metrics, and various sensible best practices and recommendations tailored to the use of wearable eye trackers in a high-fidelity simulated use study context.

Development of an Eye Tracking-based Human-Computer Interface for Real-Time Applications

The development of an eye-tracking-based human–computer interface for real-time applications is presented. To identify the most appropriate pupil detection algorithm for the proposed interface, we analyzed the performance of eight algorithms, six of which we developed based on the most representative pupil center detection techniques. The accuracy of each algorithm was evaluated for different eye images from four representative databases and for video eye images using a new testing protocol for a scene image. For all video recordings, we determined the detection rate within a circular target 50 pixel area placed in different positions in the scene image, cursor controllability and stability on the user screen, and running time. The experimental results for a set of 30 subjects show a detection rate over 84% at 50 pixels for all proposed algorithms, and the best result (91.39%) was obtained with the circular Hough transform approach. Finally, this algorithm was implemented in the proposed interface to develop an eye typing application based on a virtual keyboard. The mean typing speed of the subjects who tested the system was higher than 20 characters per minute.

Improvement of Face and Eye Detection Performance by using Multi-task Cascaded Convolutional Networks

Detection of face and eyes in unrestricted conditions has been a problem for years due to various expressions, illumination, and color fringing. Recent studies show that deep learning methods can attain impressive performance in the identification of different objects and patterns. As various systems may

use the human face as input material, the increase in facial and eye detection performance has some significance. This paper introduces an enhanced face and eye detection technique through the use of cascaded multi-task convolutional networks for our dataset. We propose in this paper a deep cascaded multi-task system that exploits their inherent correlation to improve their performance. We collected 100 videos containing about 18,265 images captured from our device and applied this dataset to the process and other systems proposed. The educated model was checked on our dataset and contrasted with the Haar cascade model as well. Our proposed method achieves 98% accuracy rate considering our dataset which is superior to the other techniques used to detect the face and eye from an image. Besides, this paper also reflects a study of different methods of detecting the eye and face in tabular format from videos. The experimental results however indicate that the proposed approach demonstrates enhanced eye and face detection output from videos.

3. Data Preparation

PoV video frames were decoded at 30 frames per second and saved to disk. In the training set, each frame is labeled by a single rater, with 1 for eye contact and 0 otherwise, which was abstracted from the onset–offset coding. In the validation set, each frame has annotations from multiple raters and the majority vote is used as the ground truth label. The datasets consist of 4,339,879 frames (281,152 with eye contact) for training and 353,924 frames (25,112 with eye contact) for validation.

4. Data Preprocessing

Preprocessing is used for reshaping the input data so that it can be fed into the CNN model. Noise from the image is removed or reduced to acceptable levels. Certain features are enhanced. Compared to a colorful image, gray-scaled images have less dimensions and contain less data. For quicker and efficient processing of images, a gray-scaled image is therefore more fruitful for the model. Hence, the gray scaling method is applied. The reason for using gray scaling is that it retains the important features of the image and reduces the computational load on the CPU because compared to colorful images, gray-scale images take up less space.

5. Training Algorithm

The deep Convolutional Neural Network (CNN) with a ResNet-50 backbone architecture 58 is used as our classifier model. 50 in ResNet-50 refers to the number of layers it has. The inputs consist of cropped face regions, resized to 224 × 224 pixels. We used a two-stage training process to support task transfer learning. In the first stage, training on datasets enables the model to learn the relationship between head pose and eye gaze direction. The model is trained to regress the 3D gaze direction based on the EYEDIAP dataset. Convergence is defined as reaching $< 6°$ mean absolute error on gaze angle and head pose. The model is then fine-tuned using our training dataset, in order to learn the condition of eye contact.

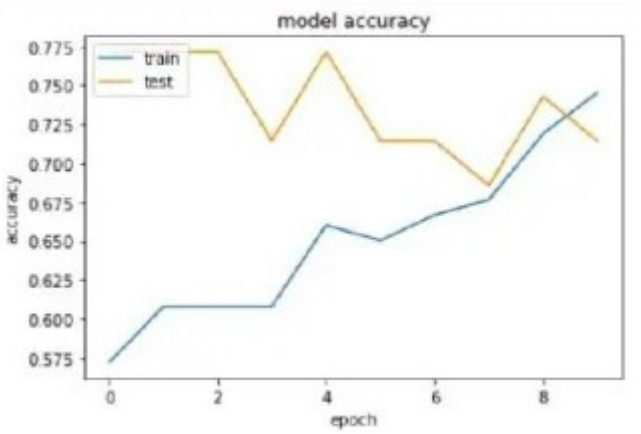
6. Eye Region Determination and Center Positioning

After generating the candidate eye regions, the CNN to make effective use of the datasets. Since ours generated candidate regions with different scales, the three different labeled candidate regions were inputted to the CNN model separately. To locate the actual eye center, we built the 2nd set of CNN that locates the pupil region in the eye region. It is composed of a convolution layer, an average pooling layer, a fully connected layer, and a logistic perceptron. The size of the other layers was adapted and the system choose the center of pupil region as the eye center.

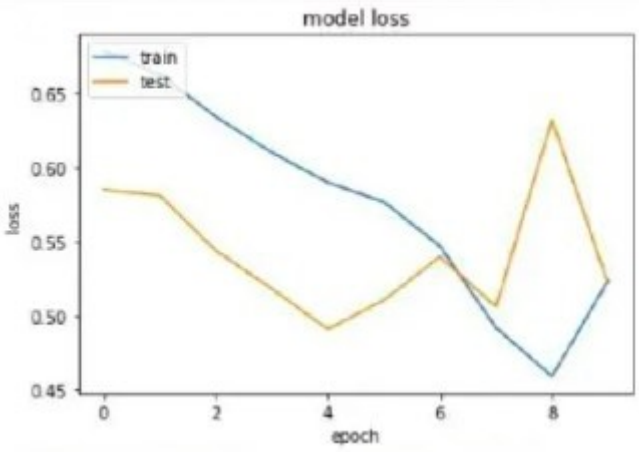
7. Detecting Center, Right, Left, Eye Close and Eye Open

From the multiface landmarks the landmarks for the eye region are detected and multiplied with the help of NumPy. The ratio function is called and based on the difference in the ratio the areas are determined. The ratio for the right is ≤ 0.42, center is > 0.42 and ≤ 0.57 is determined as left. The ratio for the eye open is > 4.5, and else eyes are open. This is done by the trial and error method.

8. Model Accuracy



9. Model Loss

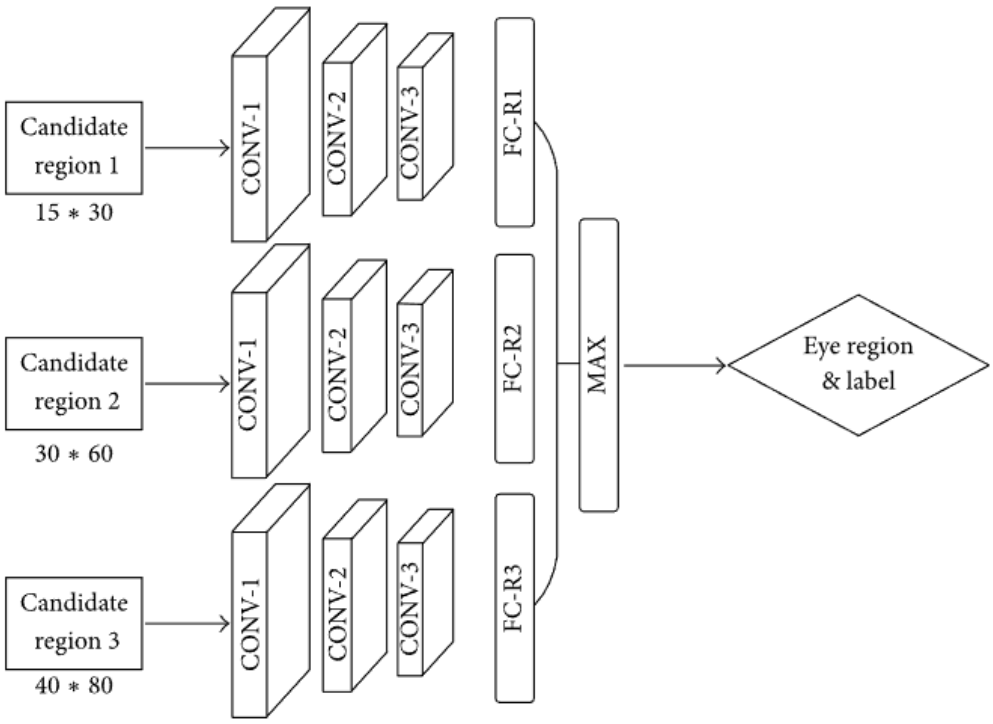


Initially, 4-5 alternate layers of convolution and pooling was used. The accuracy was close to 0.52 with loss up to 7.35. The ResNet-152 model with 34 layers of convolution and pooling with pre-trained weights were then employed. The accuracy came around 0.7 and loss in the order of e-4 on validation data. By varying epochs in between 4 and 30, we found the accuracy of the model peaked at 7 epochs.

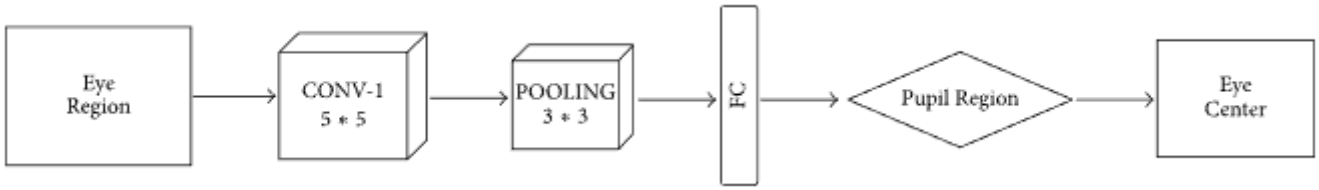
10. CNN based Eye Detection

The CNNs model combines local receptive fields, shared weights, sub- sampling to ensure some degree of shift, scale and distortion invariance. This model can learn lots of feature maps by convolving input image with a linear filter, adding a bias term and applying a non-linear function. Specially, in order to reduce features' dimension and avoid overfitting, max-pooling layer is introduced. The whole net optimizes parameters by minimizing cost function using Stochastic Gradient Descent (SGD). In this study, we designed five CNNs of different patch size to segment liver tumors. A detailed illustration of CNNs with input patch size of 17 × 17 and seven layers is shown in bellow Figure, which included three

convolutional layers, two max- pooling layers, a fully-connected layer and a softmax classifier. The max-pooling operation was adopted after the convolutional layers.



The first convolutional layer C1 is composed of 32 feature maps which are connected to all of image patched of 15 × 30 through filters of size 5 × 5 and stride of one pixel. The size of the feature maps generated in this layer was 13 × 13. The second layer S2 is max-pooling with kernel size of 30 × 60 and stride of 2. Receptive fields were not overlapping. Layer C3 took the output of S2 as input with size 40 × 80. We again used the same convolutional operation to obtain 64 feature maps with the size of 4 × 4. Then we did the same operation like S2, in Layer S4 we got the feature maps with size 2 × 2. The last convolutional layer C5 consisted of 128 feature maps of size 1 × 1. The sixth fully-connected layer F6 was applied in the top of CNNs in order to discover the relationships between high-level features obtained from previous layers, where there were 64 neurons in this layer. The final layer contained two units fully connected with the layer F6.



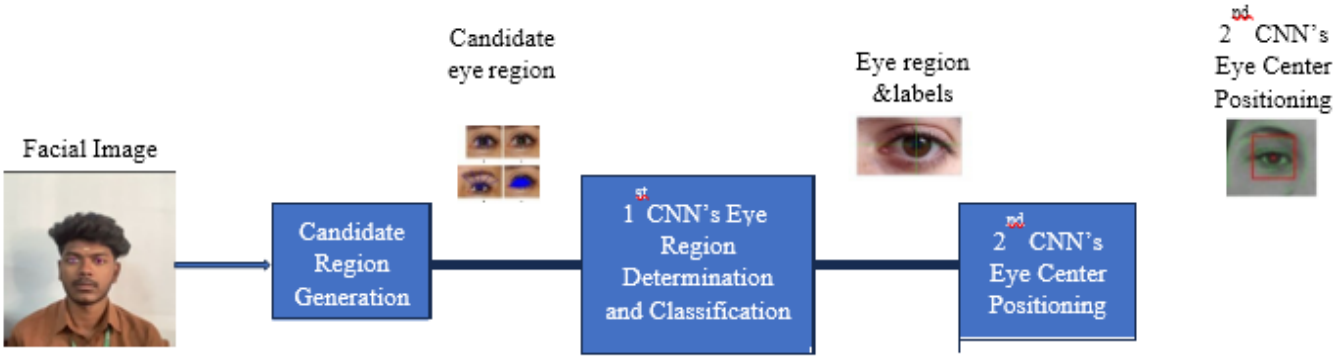
11. Experiments and Results

In the experiments, we designed CNNs architectures with different input patch sizes to study their impact on the accuracy of eye detection. The feature detector is a two-dimensional (2-D) array of weights, which represents part of the image. While they can vary in size, the filter size is typically a 3 × 3 matrix; this also determines the size of the receptive field. The filter is then applied to an area of the image, and a dot product is calculated between the input pixels and the filter. This dot product is then fed into an output array. Afterwards, the filter shifts by a stride, repeating the process until the kernel has

swept across the entire image. The final output from the series of dot products from the input and the filter is known as a feature map, activation map, or a convolved feature.

It only needs to connect to the receptive field, where the filter is being applied. Since the output array does not need to map directly to each input value, convolutional (and pooling) layers are commonly referred to as "partially connected" layers. However, this characteristic can also be described as local connectivity.

The Architecture of CNNs for Eye Detection



12. Results

In Eye Detection, existing method showed 84.34% of accuracy. Compared to traditional machine learning methods, the CNNs method performed better, it yielded 94.7% accuracy.

Detecting Right Eye's Position

Detecting Eyes

Values of Ratio Points

13. Conclusion

Working with a computer gives us a pleasant experience and delivers the expected results excellently. The project "A Robust Eyeball Detection based on Computer Vision Approaches" is to mark the user to have the basic knowledge of computers. This project is useful for online proctoring systems. Here we used an effective cascaded CNNs method to detect the eye location. Our method can simultaneously detect left and right eye locations and center even when the face is 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 feature points combined with the cascaded CNNs in order to achieve significantly high efficiency and satisfactory classification rate.

14. Future Enhancement

This system can be extended into an Android and web application where users can use it both in the web

and Android version. In the future, more datasets will be collected to train more powerful eye recognition models. Moreover, by using CNN, we can be able to detect all kinds of videos such as blurry and less quality videos. We plan to improve the work to handle all kinds of video quality.

References

[1] X. Deng, G. Du (2008). 3D Liver Tumor Segmentation Challenge 2008. MICCAI Workshop.
[2] Alessandro Giusti, Chiara Zocchi, Alberto Rovetta (2001). A Noninvasive System for Evaluating Driver Vigilance Level Examining both Physiological and Mechanical Data. IEEE Transactions on Intelligent Transportation Systems, Vol. 10, No. 1.
[3] Chin-Teng Lin, Yu-Chieh Chen, Ruei-Cheng Wu, Sheng-Fu Liang, Teng-Yi Huang (2005). Assessment of driver's driving performance and alertness using EEG-based fuzzy neural networks. IEEE International Symposium on Circuits and Systems.
[4] Guang-Yuan Zhang, Bo Cheng, Rui-Jia Feng, Jia-Wen Li (2008). Real-time driver eye detection method using Support Vector Machine with Hu invariant moments. International Conference on Machine Learning and Cybernetics, Vol. 34, pp. 856-876.
[5] H. Fu, Y. Wei, F. Camastra, P. Arico, H. Sheng (2006). Advances in Eye Tracking Technology: Theory, Algorithms, and Applications. Computational Intelligence and Neuroscience, vol. 2016, pp, 1765-1654.
[6] H. Mosa, M. Ali, K. Kyamakya (2013). A computerized method to diagnose strabismus based on a novel method for pupil segmentation. Proceedings of the International Symposium on Theoretical Electrical Engineering, vol. 27, pp. 823-876.
[7] Lifang Deng, Xingliang Xiong, Jin Zhou, Ping Gan, Shixiong Deng (2012). Fatigue Detection Based on Infrared Video Pupillography. 4th International Conference on Bioinformatics and Biomedical Engineering (iCBBE).
[8] L. Zhang, Y.Cao, F. Yang, Q. Zhao (2017). Machine Learning and Visual Computing. Applied Computational Intelligence and Soft Computing, vol. 2017, pp. 18-76.