Eye State Recognizer Using Light-weight Architecture for Drowsiness Warning

Duy-Linh Nguyen[1], Muhamad Dwisnanto Putro[2], and Kang-Hyun Jo[3]

[1] School of Electrical Engineering, University of Ulsan, Ulsan, Korea
ndlinh301@mail.ulsan.ac.kr
[2] School of Electrical Engineering, University of Ulsan, Ulsan, Korea
dwisnantoputro@mail.ulsan.ac.kr
[3] School of Electrical Engineering, University of Ulsan, Ulsan, Korea
acejo@ulsan.ac.kr

Abstract. The eye are a very important organ in the human body. The eye area and eyes contain lots of useful information about human interaction with the environment. Many studies have relied on eye region analyzes to build the medical care, surveillance, interaction, security, and warning systems. This paper focuses on extracting eye region features to detect eye state using the light-weight convolutional neural networks with two stages: eye detection and classification. This method can apply on simple drowsiness warning system and perform well on Intel Core I7-4770 CPU @ 3.40 GHz (Personal Computer - PC) and on quad-core ARM Cortex-A57 CPU (Jetson Nano device) with 19.04 FPS and 17.20 FPS (frames per second), respectively.

Keywords: Deep learning · Convolutional neural network (CNN) · Eye detection · Eye state recognizer · Eye classification · Drowsiness warning

1 Introduction

The traffic accident is a great threat to human beings all over the world. About 1.35 million people die each year from road traffic crashes and 90% of the main cause is from drivers [3]. One of the main causes is driver drowsiness. This situation usually occurs when a driver lacks sleep, uses alcohol, uses drugs, or goes on a long trip. To detect driver's drowsiness, many methods have been specifically conducted such as analyzing human behavior, vehicle behavior, and driver physiology [21]. Human behavior can be surveillance through the extraction of facial features, eye features, yawning, or head gestures. Vehicle behavior can be monitored via vehicle movement in the lanes and relative to other vehicles operating nearby. Driver physiology can be estimated by sensors that measure heart rate, blood pressure, or sudden changes in body temperature. However, deploying applications to monitor vehicle behavior and examine human physiology requires huge complexity and costly techniques. In addition, it can cause uncomfortable and unfocus for the driver during road operation. From the above analysis, this paper proposes the light-weight convolutional neural network design supports

the driver's drowsiness warning system. The system consists of two main stages based on extracting the eye area features: eye detection and classification. Deploying the application is very simple on PC or on Jetson Nano device and a common camera.
The main contributions of this paper are as follows:
1 - Proposed two light-weight Convolutional Neural Network architectures, includes eye detection and classification.
2 - Develop the eye state recognizer can run on small processor devices supporting for drowsiness warning system without ignoring the accuracy.

The organization of this paper as follows: Section 2 present the previous methodologies relative to eye state detection, drowsiness warning system, their strengths and weaknesses. Section 3 discuses more detail about proposed methodology. Section 4 describes and analyzes the results. Finally, section 5 concludes the paper and present the future work.

2 Related work

In this section, the paper will present several methodologies employed on eye state detection and drowsiness warning system. These methodologies can be divided into the untraining methodologies and training methodologies.

2.1 Untraining methodologies

In the untrained method, sensors are often used to measure the signal obtained from parts of the human body or objects. In addition, several image processing algorithms are also used to extract characteristics on the image from which to make predictions. The techniques used in [10, 4–6] rely on sensors arranged around the eyes to gather and analyze electrical signals. These techniques can collect signals very quickly but are uncomfortable for the user and may be subject to interference due to environmental influences. Therefore, it leads to low accuracy while expensive implementation. In the Computer Vision field, there are many methods to extract eye area and inside eye features without training. Specifically, the methods include iris detection based on calculating the variance of maximum iris positions [7], methods based on matching the template [14], and methods based on a fuzzy-system that using eye segmentation [12]. Scale Invariant Feature Transformation (SIFT) in [16] consider image information in continuous video, method based on the movement in facial and eyelid [20], method computes the variance in the values of black pixels in these areas [18]. These methods can provide powerful feature information but require complex computation and are very sensitive to illumination.

2.2 Training methodologies

The training method is based on extracting the features and learning them. There are some traditional methods such as Support Vector Machine (SVM) [11], Active Appearance Models (AAM) [26], Principal Component Analysis (PCA) subspace analysis [28]. These methods may achieve better eye classification accuracy than the untrained methods, but they need to be improved or combined with other techniques to adapt to variability in real-time.

With the strong development of deep learning, the widespread application of convolutional neural networks in object detection, image classification, and recognition is growing. Many typical CNNs can be used to classify eye state such as Lenet [17], Alexnet [13], VGG [23], Resnet [9] and so on. In these methods, the feature extracted automatically from the dataset through the training process and then classifies the images based on these features. Their performance is reasonable on accuracy and loss function. However, these models have heavy training time depend on the depth of models and size of input images. In addition, the complicated construction of the eye and eye area requires to improve to the CNN models to accommodate accuracy and loss function.

Recently years, several studies have used traditional face detection methods such as Viola-Jones [27], Haar-like feature [19], Adaboost [15] in combination with CNNs to detect eye status. However, these techniques often face illumination conditions, not frontal face, occluded or overlap, and oblique face position.

3 Methodology

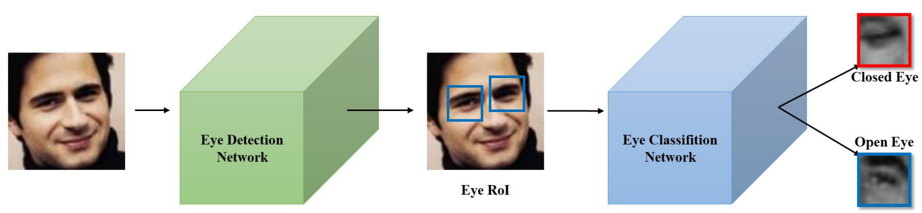

Fig. 1. The proposed pipeline of the eye state recognizer. It consists of two main networks: eye detection and eye classification network. The input image goes through the eye detection network and the RoI eye regions are generated in this network, then these regions will be classified by the eye classification network to predict eye state (open and closed for each eye).

The proposed pipeline of the detailed eye state recognizer is shown in Fig. 1. The pipeline consists of two networks which are eye detection and eye classification network. In the eye detection network, we proposed light-weight and efficient CNN to extract the Region of Interest (RoI) of the eye region and then crop these areas. The output of this network goes through the eye classification network, which is a simple CNN for classifying eyes. The output are eye states: closed eye and open eye in each eye region.

3.1 Eye detection network

This paper proposes a convolutional neural network that allows locating eye areas in the image based on predefined bounding boxes. This network exploits the functions of the basic layers and modules in CNN such as convolution and max-pooling layers, C.ReLU, and Inception modules to extract the feature map. Then, two sibling convolution layers will be applied to classify and regress eye bounding box coordinates. The proposed network is described in detail, as shown in Fig. 2. To extract the feature map quickly and efficiently, the blocks in this architecture use the sequence of layers as follows:

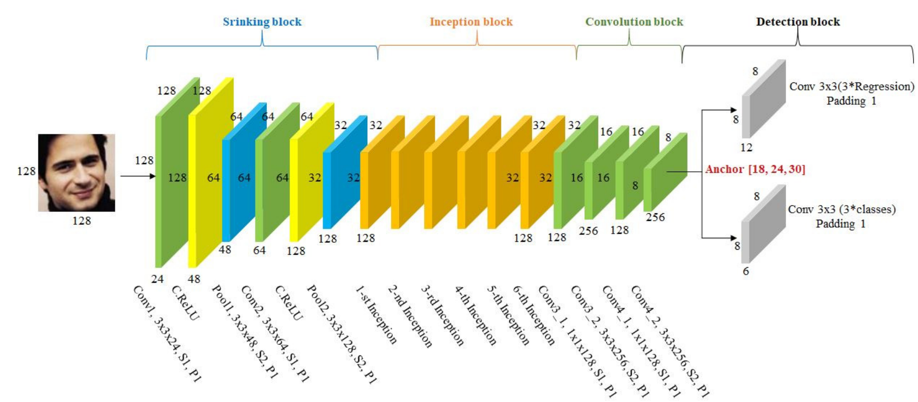

Fig. 2. The proposed convolutional neural network for eye detection. It is built based on four blocks which are Shrinking, Inception, Convolution, and Detection block.

Shrinking block: The Shrinking block quickly shrinks the input image space by selecting the appropriate kernel size. As shown in Fig. 2, Conv1, Pool1, Conv2, and Pool2 layer use stride of 1, 2, 1, and 2, respectively. This block mainly uses a 3×3 kernel size and the input image size is 128×128. In addition, C. ReLu (Concatenated Rectified Linear Unit) [15] is also used to increase the efficiency while ensuring accuracy. The C.ReLU module is described in Fig. 3(a). After going through this block, the image size is shrunk down from the original image size to 32×32. That means the input image size is reduced by four times while preserving the important information of the original image.

Inception block: The Inception block is a combination of six Inception modules [25]. Each Inception module consists of multiple convolution branches with many different kernel sizes. Specifically, this module is designed with four branches, using convolution operations with kernel size 1×1 and 3×3 and the number of

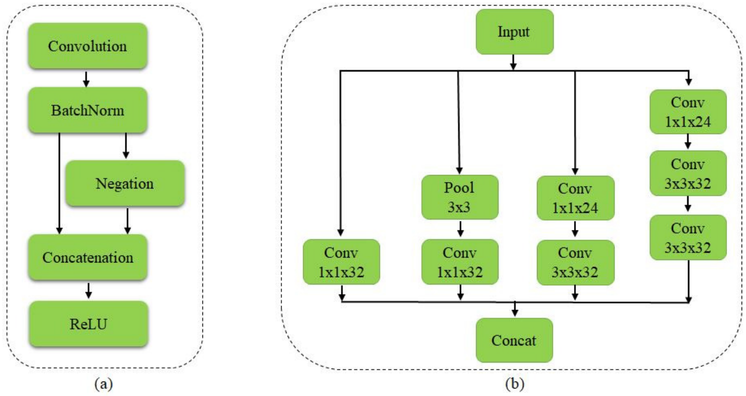

Fig. 3. (a) C.ReLU and (b) Inception module

kernels is 24 and 32. After each convolution operation, the Batch Normalization and ReLU activation function are used. In addition, it uses a max-pooling operation and final by concatenation operation to combine the results of branches. As a multi-scale block according to the width of the network, these branches can enrich receptive fields. Fig. 3(b) shown detail about the Inception module. The feature map with size is $32 \times 32 \times 128$ will be maintained from $1-st$ to $6-th$ Inception and provided the various information of features when processed by this block.

Convolution block: The Convolution block is the final stage of the feature map extraction process. In this block mainly using common convolution operations with kernel sizes of 1×1 and 3×3 to shrink the size and increase the dimension of the feature map. As described in Fig. 2, Conv3_1 and Conv4_1 use 1×1 kernel size with 128 kernels, Conv3_2 and Conv4_2 use 3×3 kernel size with 256 kernels. The Batch Normalization and ReLU activation function are also used after each convolution operation. The output of this block is the feature map size of $8 \times 8 \times 256$, which means the size of the feature map is further reduced by four times and the number of kernels is doubled from 128 to 256. This block is the bridge between the feature map extraction process and the detection process.

Detection block: The end of the eye detection network is the detection block. This block uses one two-siblings convolution operation with kernel size is 3×3 for classification and bounding box regression. These layers apply on a feature map with size 8×8 which is an output feature map from the previous block. The detector uses square anchors of various sizes to predict the position of the corresponding eye in the original image. In this case, it uses three square anchors with sizes 18, 24, 30 for small eye sizes, medium eye sizes, and large eye sizes, respectively. Finally, the detector generates a four-dimensional vector (x, y, w, h) as location offset and a two-dimensional vector (eye or not eye) as label classification.

The loss function used in eye detection network is same as RPN in Faster R-CNN [22], a 2-classes softmax-loss for classification task and the smooth $L1$ loss for regression. The loss function for an image is defined as follows:

$$L\left(\{p_i\}, \{t_i\}\right) = \frac{1}{N_{cls}} \sum_i L_{cls}\left(p_i, p_i^*\right) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}\left(t_i, t_i^*\right), \quad (1)$$

where i is the index of an anchor in a mini-batch and p_i is the predicted probability of anchor i being an object, p_i^* is ground-truth label and $p_i^* = 1$ if the anchor is positive and is 0 if the anchor is negative. t_i is the center coordinates and dimension of the prediction and t_i^* is the ground truth coordinates of bounding box. $L_{cls}\left(p_i, p_i^*\right)$ is the classification loss using the softmax-loss shown as in Eq. (2), $L_{reg}\left(t_i, t_i^*\right) = R(t_i - t_i^*)$ with R is the Smooth loss $L1$ defined as in Eq. (3). The two terms are normalized by N_{cls} and N_{reg} and weighted by a balancing parameter λ. N_{cls} is normalized by the mini-batch size, N_{reg} is normalized by the number of anchor locations and λ is assigned by 10.

$$L_{cls}\left(p_i, p_i^*\right) = -\sum_{i \in Pos} x_i^p log\left(p_i\right) - \sum_{i \in Neg} log\left(p_i^0\right), \quad (2)$$

where $x_i^p = \{0, 1\}$ is indicator for matching the $i-th$ default box to ground-truth of category p, p_i^0 is the probability for non-object classification.

$$R(x) = \begin{cases} 0.5x^2 & if\,|x| < 1 \\ |x| - 0.5 & otherwise \end{cases} \quad (3)$$

3.2 Eye classification network

Fig. 4 shows a detailed description of the classification network architecture. Similar to the CNN of classification, this network is built based on sequential layers as convolution layers, average pooling layers, and uses Softmax function to classify the data.

This network architecture uses one group of two Convolution layers with 7×7 filter size, one group of two convolution layers with 5×5 filter size, two groups of two convolution layers with 3×3 filter size followed by each group of one average pool layer and one ReLU activation function. The feature extractor ends by one Convolution layer with a 3×3 filter size. The spatial dimensions of the feature map are reduced from 100×100 to 7×7. The global average pooling layer to further reduce the dimension of the feature map to 1×1. Finally, the network uses the Softmax activation function to generate the predicted probability of each class (open and closed eyes). Usage of Global average pooling can minimize

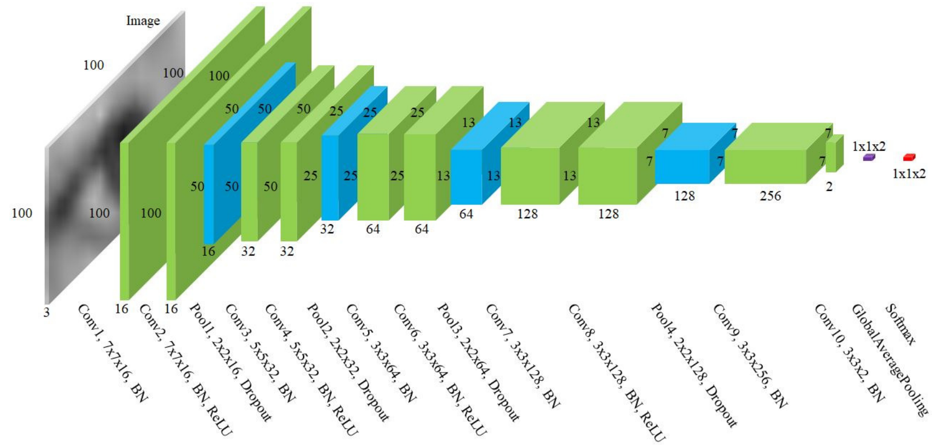

Fig. 4. The proposed eyes classification network. The network is based on nine sequential layers of convolution with filters of sizes 7×7, 5×5 and 3×3. Following the convolution layer groups are the ReLU activation functions and the average pooling layers. The global average pooling layer is used to quickly reduce the size of the feature map. Finally, it applies a Softmax activation function to compute the probability of open or closed eyes.

the possibility of overfitting by reducing the total number of parameters in the network. On the other hand, to increase the ability to avoid network overfitting, the Batch Normalization method is also used after convolution operations. The classifier uses the Cross-Entropy loss function to calculate the loss during training.

4 Experimental result

4.1 Dataset preparation

The eye detection network is trained on the CEW (Closed Eyes In The Wild) [24], BioID Face [1] and GI4E (Gaze Interaction for Everybody) dataset [2]. CEW dataset contains 2,423 subjects, among which 1,192 subjects with both eyes closed are collected directly from Internet, and 1,231 subjects with eyes open are selected from the Labeled Face in the Wild (LFW) database. The image size is cropped and resized from coarse faces images to the size 100×100 (pixels) and then extract eye patches of 24×24 centered at the localized eye position. BioID Face dataset consists of 1,521 gray level images with a resolution of 384×286 pixel. Each one shows the frontal view of a face of one out of 23 different test persons. The eye position label is assigned manually and generated coordinate of ground-truth bounding box based on this position with size is 36×36. GI4E is dataset of images for iris center and eye corner detection. The database consists of a set of 1,339 images acquired with a standard webcam, corresponding to 103 different subjects and 12 images each. The images in this dataset have a resolution of 800×600 pixels in PNG format. The images are associated to a ground-truth text file. It contains manually annotated 2D iris and corner points (in pixels). The coordinate of ground-truth of the bounding box also generated from the position of iris by size is 46×46. Each original data set is divided into two basic data sets with 80% for training and 20% for testing phase.

The eyes classification network is trained and evaluated on Closed Eyes In The Wild (CEW) dataset [24]. This dataset contains 2,384 eye images with closed eyes and 2,462 face images with open eye images. To improve the classification capacity, this dataset has been augmented by flipping vertically, changing the contrast and brightness. The dataset was divided into 80% images for the training set and 20% images for the evaluation set.

4.2 Experimental setup

The experiments in this paper are trained on GeForce GTX 1080Ti GPU, tested on Intel Core I7-4770 CPU @ 3.40 GHz, 8GB of RAM (PC) and quad-core ARM Cortex-A57 CPU, 4GB of RAM (Jetson Nano device). In order to train the eye detection network, many configurations have been used to improve face detection quality. The Stochastic Gradient Descent optimization method used, the batch size of 16, the weight decay is 5.10^{-4}, the momentum is 0.9, the learning rates from 10^{-6} to 10^{-3}. The threshold of IoU (Intersection over Union) is 0.5 to produce the best bounding box. For the eye classification network which uses some basic configuration for image classification such as the Adam optimization method, batch size of 16, the learning rate is 10^{-4}.

4.3 Experimental result

Each network in the pipeline is individually trained and tested on the image dataset and a comprehensive network was tested on a real-time system using a camera connecting the PC using the CPU and Jetson Nano device with the quad-core ARM Cortex-A57 CPU. For the image datasets, the eye detection network achieved results on CEW, BioID Face, and GI4E dataset with 96.48%, 99.58%, and 75.52% of AP, respectively. The testing result of the network on CEW, BioID Face, and GI4E dataset shown in Table 1 and Fig. 5. The classification results of the eye classification network on the CEW dataset are shown in Table 2. The proposed classification network outperforms compared to popular classification networks with a very small number of parameters.

Finally, the entire system was tested on a camera connected to a CPU-based PC and Jetson Nano device. In order to increase efficiency for eye state recognition, the pipeline adds the trained face detection network in previous work in [8]. The distance from the camera to the human face is equal to the distance in the car. Because the distance is set quite close (distance < 0.5 meters), the images obtained via the camera are mainly images in the frontal face. This condition

Fig. 5. The qualitative result on CEW dataset (first row), on BioID Face dataset (second row), and on GI4E dataset (third row). The proposed network can detect eye location at many different pose shown as in CEW dataset and when wearing the glasses shown as in BioID Face and Gi4E dataset. The number in each bounding box shows a confidence score of prediction.

Table 1. The testing result of the proposed network on CEW, BioID Face and GI4E test dataset.

Dataset	**Average Precision (%)**
CEW	96.48
BioID Face	99.58
Gi4E	75.52

improves eye detection and open and closed eye classifier. Table 3 shown the speed testing result of the eye state recognizer on the camera.

Within the speed achieved 19.04 FPS on Intel Core I7-4770 CPU @ 3.40 GHz and 17.20 FPS fps when tested on the quad-core ARM Cortex-A57 CPU, the recognizer can work well in normal conditions without delay. Fig. 6 and Fig.7 shown the qualitative result of the pipeline when testing on camera connect with Jetson Nano device. The result proves that the system also can recognize eye state with several cases such as glasses, face mask, hat, face mask and hat-wearing. However, under noise conditions such as the illumination, head tilted horizontally at an angle greater than 45 degrees, vertical rotation head at an angle greater than 90 degrees, or the head bows down, the efficiency of the recognizer may be significantly reduced because it can not detect the eyes to classify these areas. In fact, these cases can be referred to as unusual cases that can be alerted when developing a drowsiness warning system.

Fig. 6. The qualitative result when testing on camera connects with Jetson Nano device with three participants (two males and one female). The result shows in two open eyes (first row), two closed eyes (second row), one closed eye and one ope eye (third row).

Fig. 7. The qualitative result when testing on camera connects with Jetson Nano device with four situations: glasses (first column), face mask (second column), hat (third column), face mask and hat-wearing (fourth column).

Table 2. The comparison of classification results of the eyes classification network with popular classification networks on the CEW dataset.

Network	Accuracy (%)	Number of paramenters
Proposed	97.53	632,978
VGG13	96.29	7,052,738
ResNet50	94.85	23,591,810
Alexnet	96.71	67,735,938
LeNet	96.70	15,653,072

Table 3. The speed testing results of eyes status recognizer on the camera.

Device	Face Detection (fps)	Eye Detection (fps)	Eye Classification (fps)	Total (fps)
Jetson Nano	100.02	30.89	63.08	17.20
PC	198.22	42.52	41.73	19.04

5 Conclusion and future work

This paper has proposed an eye state recognizer with two light-weight modules using convolutional neural networks. The eye detection network uses several basic layers in CNN, C.ReLu, Inception module. The eye classification network is a simple convolutional neural network that consists of convolution layers alternating with the average pooling layers, then ending by the global average pooling layer and Softmax function. The optimal number of parameters and computation makes it can be deployed on portable devices and CPU-based computers. In the future, the eye state recognizer will be integrated with several modern techniques, advanced optimized, and increase applicability in real-time systems. On the other hand, the dataset needs to collect and annotate under variety of conditions to ensure this recognizer works properly such as glasses, hat, face mask wearing.

Acknowledgement

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government. (MSIT)(2020R1A2C2008972)

References

1. The bioid face database. https://www.bioid.com/facedb, accessed: 2020-10-23
2. Gi4e - gaze interaction for everybody. http://www.unavarra.es/gi4e/databases?languageId=1, accessed: 2020-10-23
3. Road traffic injuries. https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries, accessed: 2020-10-22

4. Bulling, A., Ward, J., Gellersen, H., Tröster, G.: Eye movement analysis for activity recognition using electrooculography. IEEE Transactions on Pattern Analysis and Machine Intelligence **33**, 741–753 (2011)
5. Champaty, B., Pal, K., Dash, A.: Functional electrical stimulation using voluntary eyeblink for foot drop correction. In: 2013 Annual International Conference on Emerging Research Areas and 2013 International Conference on Microelectronics, Communications and Renewable Energy. pp. 1–4 (2013). https://doi.org/10.1109/AICERA-ICMiCR.2013.6575966
6. Chang, W., Lim, J., Im, C.: An unsupervised eye blink artifact detection method for real-time electroencephalogram processing. Physiological measurement **37 3**, 401–17 (2016)
7. Colombo, C., Comanducci, D., Bimbo, A.D.: Robust tracking and remapping of eye appearance with passive computer vision. ACM Trans. Multim. Comput. Commun. Appl. **3**, 2:1–2:20 (2007)
8. Duy-Linh, N., Muhamad Dwisnanto, P., Kang-Hyun, J.: Eyes status detector based on light-weight convolutional neural networks supporting for drowsiness detection system. In: The 46th Annual Conference of the IEEE Industrial Electronics Society. Singapore (2020)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CoRR **abs/1512.03385** (2015), http://arxiv.org/abs/1512.03385
10. Hsieh, C.S., Tai, C.C.: An improved and portable eye-blink duration detection system to warn of driver fatigue. Instrumentation Science & Technology **41**(5), 429–444 (2013). https://doi.org/10.1080/10739149.2013.796560, https://doi.org/10.1080/10739149.2013.796560
11. Jo, J., Lee, S., Jung, H., Park, K., Kim, J.: Vision-based method for detecting driver drowsiness and distraction in driver monitoring system. Optical Engineering **50**, 7202 (11 2011). https://doi.org/10.1117/1.3657506
12. Kim, K.W., Lee, W.O., Kim, Y.G., Hong, H.G., Lee, E.C., Park, K.R.: Segmentation method of eye region based on fuzzy logic system for classifying open and closed eyes. Optical Engineering **54**(3), 1 – 19 (2015). https://doi.org/10.1117/1.OE.54.3.033103, https://doi.org/10.1117/1.OE.54.3.033103
13. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Commun. ACM **60**(6), 84–90 (May 2017). https://doi.org/10.1145/3065386, https://doi.org/10.1145/3065386
14. Królak, A., Strumiłło, P.: Eye-blink detection system for human—computer interaction. Univers. Access Inf. Soc. **11**(4), 409–419 (Nov 2012). https://doi.org/10.1007/s10209-011-0256-6, https://doi.org/10.1007/s10209-011-0256-6
15. Kégl, B.: The return of adaboost.mh: multi-class hamming trees (2013)
16. Lalonde, M., Byrns, D., Gagnon, L., Teasdale, N., Laurendeau, D.: Real-time eye blink detection with gpu-based sift tracking. In: Fourth Canadian Conference on Computer and Robot Vision (CRV '07). pp. 481–487 (2007). https://doi.org/10.1109/CRV.2007.54
17. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998). https://doi.org/10.1109/5.726791
18. Lee, W., Lee, E.C., Park, K.: Blink detection robust to various facial poses. Journal of neuroscience methods **193**, 356–72 (11 2010). https://doi.org/10.1016/j.jneumeth.2010.08.034

19. Mita, T., Kaneko, T., Hori, O.: Joint haar-like features for face detection. In: Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1. vol. 2, pp. 1619–1626 Vol. 2 (2005). https://doi.org/10.1109/ICCV.2005.129
20. Mohanakrishnan, J., Nakashima, S., Odagiri, J., Shanshan Yu: A novel blink detection system for user monitoring. In: 2013 1st IEEE Workshop on User-Centered Computer Vision (UCCV). pp. 37–42 (2013). https://doi.org/10.1109/UCCV.2013.6530806
21. Ramzan, M., Khan, H.U., Awan, S.M., Ismail, A., Ilyas, M., Mahmood, A.: A survey on state-of-the-art drowsiness detection techniques. IEEE Access **7**, 61904–61919 (2019). https://doi.org/10.1109/ACCESS.2019.2914373
22. Ren, S., He, K., Girshick, R.B., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. CoRR **abs/1506.01497** (2015), http://arxiv.org/abs/1506.01497
23. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR **abs/1409.1556** (2015)
24. Song, F., Tan, X., Liu, X., Chen, S.: Eyes closeness detection from still images with multi-scale histograms of principal oriented gradients. Pattern Recognition **47**(9), 2825 – 2838 (2014). https://doi.org/https://doi.org/10.1016/j.patcog.2014.03.024, http://www.sciencedirect.com/science/article/pii/S0031320314001228
25. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S.E., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. CoRR **abs/1409.4842** (2014), http://arxiv.org/abs/1409.4842
26. Trutoiu, L.C., Carter, E.J., Matthews, I., Hodgins, J.K.: Modeling and animating eye blinks. ACM Trans. Appl. Percept. **8**(3) (Aug 2011). https://doi.org/10.1145/2010325.2010327, https://doi.org/10.1145/2010325.2010327
27. Viola, P., Jones, M.: Robust real-time face detection. International Journal of Computer Vision **57**, 137–154 (05 2004). https://doi.org/10.1023/B:VISI.0000013087.49260.fb
28. Wu, J., Trivedi, M.: M.m.: An eye localization, tracking and blink pattern recognition system: Algorithm and evaluation. TOMCCAP **6** (03 2010). https://doi.org/10.1145/1671962.1671964