# Eye State Recognizer Using Light-Weight Architecture for Drowsiness Warning

Duy-Linh Nguyen(✉), Muhamad Dwisnanto Putro, and Kang-Hyun Jo

School of Electrical Engineering, University of Ulsan, Ulsan, Korea
{ndlinh301,dwisnantoputro}@mail.ulsan.ac.kr, 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:** Convolutional neural network (CNN) · Deep learning · Drowsiness warning · Eye detection · Eye classification · Eye state recognizer

## 1 Introduction

The traffic accident is a great threat to human beings all over the world. More than one 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 architecture 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.

N. T. Nguyen et al. (Eds.): ACIIDS 2021, LNAI 12672, pp. 518–530, 2021.
https://doi.org/10.1007/978-3-030-73280-6_41

Deploying the application is very simple on PC or on Jetson Nano device and a common camera.

The paper has two main contributions as follows:

- Proposed two light-weight Convolutional Neural Network architectures, includes eye detection and classification.
- Develop the eye state recognizer can run on small processor devices supporting for drowsiness warning system without ignoring the accuracy.

The remaining of this paper is organized: Sect. 2 present the previous related methodologies to eye state detection, drowsiness warning system, their strengths, and weaknesses. Section 3 shows detail about the proposed technique. Section 4 describes and analyzes the results. The paper finalizes by Sect. 5 with the conclusion and future work.

## 2 Related Work

In the related work section, the paper will show several methodologies applied to eye state detection and drowsiness warning system. These methodologies can be grouped into the untraining 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 [4–6,9] 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 [13], and methods based on a fuzzy-system that using eye segmentation [11]. Scale Invariant Feature Transformation (SIFT) in [15] consider image information in continuous video, method based on the movement in facial and eyelid [19], method computes the variance in the values of black pixels in these areas [17]. 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) [10], Active Appearance Models (AAM) [27], Principal Component Analysis (PCA) subspace analysis [29]. 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 explosive growth of machine learning, the widespread application of convolutional neural networks in image classification, object detection and recognition is increasing. Many typical CNNs can be used to classify eye state such as Lenet [16], Alexnet [12], VGG [24], Resnet [8] 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 [28], Haar-like feature [18], Adaboost [14] 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

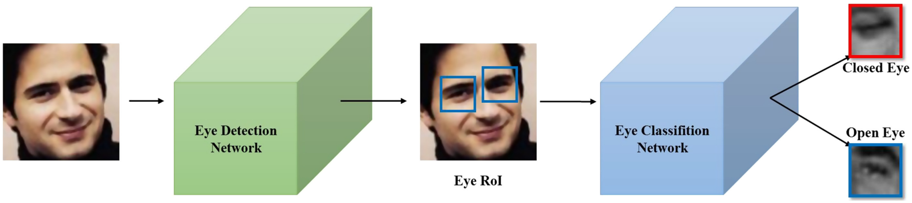

**Fig. 1.** The proposed pipeline of the eye state recognizer. It consists of two main networks: eye detection and eye classification network. The input image goes through the eye detection network and the RoI eye regions are generated in this network, then these regions will be classified by the eye 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 study proposes a convolutional neural network architecture that locates the eye areas in the images. This network extracts the feature maps by using the basic layers and components in CNN such as convolution and max-pooling layers, C.ReLU, and Inception modules. After that, two sibling convolution layers will be applied for classification and regression. The detail of the proposed network is described in Fig. 2.

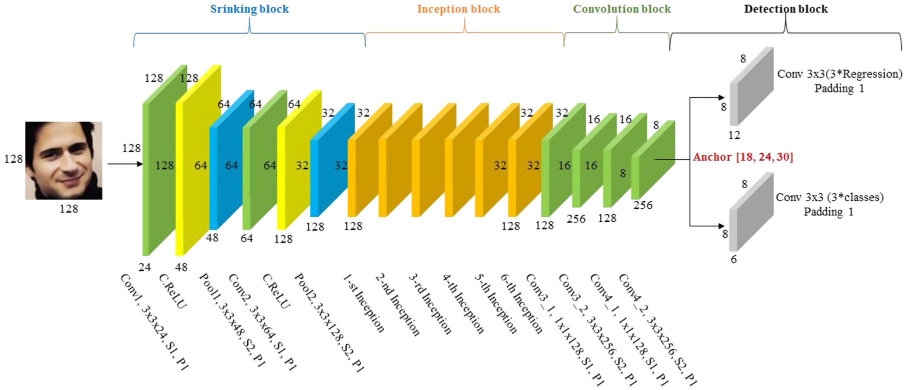

**Fig. 2.** The proposed eye detection network consists of four sequential blocks: Shrinking, Inception, Convolution, and Detection block.

**Shrinking Block:** This block quickly decreases the input image spatial by choosing the appropriate kernel size. The stride used in Conv1, Pool1, Conv2, and Pool2 is 1, 2, 1, and 2, respectively. The 3 × 3 kernel size is mainly used and the input image size is resized to 128 × 128. On the other hand, C. ReLu (Concatenated Rectified Linear Unit) [23] is also used to increase efficiency while ensuring accuracy. The C.ReLU module is described in Fig. 3(a). This block shrunk down the input image size to $32 \times 32$. In another word, the size is reduced by four times while maintaining the important information of the input image.

**Inception Block:** To build the Inception block, a combination of six Inception modules [26] is used. Each Inception module consists of four branches, using consequential convolution operations with kernel size $1 \times 1$, $3 \times 3$ and the number of kernels is 24, 32. Following each convolution operation, the Batch Normalization and ReLU activation function are used. In some branches, the max-pooling operation is also used and final by concatenation operation to combine the results of branches. With a multi-scale approach according to the width of the network, these branches can enrich receptive fields. Figure 3(b) shown the Inception module in detail. The feature map with a size is 32 × 32 × 128 will be unchanged from $1-st$ to $6-th$ Inception and provided the various information of features when processed by this block.

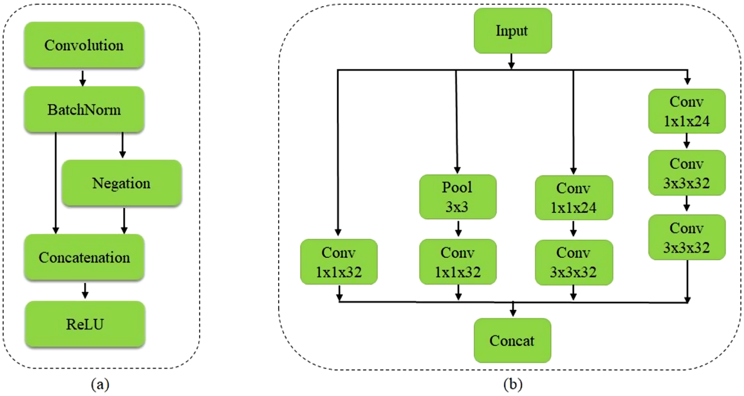

**Fig. 3.** (a) C.ReLU and (b) Inception module

**Convolution Block:** The Convolution block is the final stage of the feature map extraction process. This block mainly using common $1 \times 1$ and $3 \times 3$ convolution operations to decrease the size and increase the dimension of the feature maps. Conv3_1 and Conv4_1 use 128 of $1 \times 1$ kernels, Conv3_2 and Conv4_2 use 256 of $3 \times 3$ kernels. The Batch Normalization and ReLU activation function are also used after each convolution operation. It outputs the feature map size of $8 \times 8 \times 256$, which proves that the size of the feature map is further reduced by four times and the number of kernels is doubled from 128 to 256. The role of the Convolution block is the link between the feature map extraction and the detection stage.

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

The loss function of the eye detection network is similar to RPN (Region Proposal Network) in Faster R-CNN [22], the softmax-loss to compute loss for classification and the smooth $L1$ loss for regression task. For an image, the loss function is defined as below:

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

where $p_i$ is the predicted probability of anchor $i$ being an object, $p_i^*$ is ground-truth label. The anchor is positive when $p_i^* = 1$ and it is negative if $p_i^* = 0$. $t_i$ is the center coordinates and dimension (Height and Width) of the prediction and $t_i^*$ is the ground truth coordinates. $L_{cls}\left(p_i, p_i^*\right)$ 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 Eq. (1) is normalized by $N_{cls}$ and $N_{reg}$ and balancing by parameter $\lambda$. $N_{cls}$ is normalized by the mini-batch size, $N_{reg}$ is normalized by the number of anchor locations and $\lambda$ is set by 10.

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

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.

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

### 3.2 Eye Classification Network

Figure 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 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.

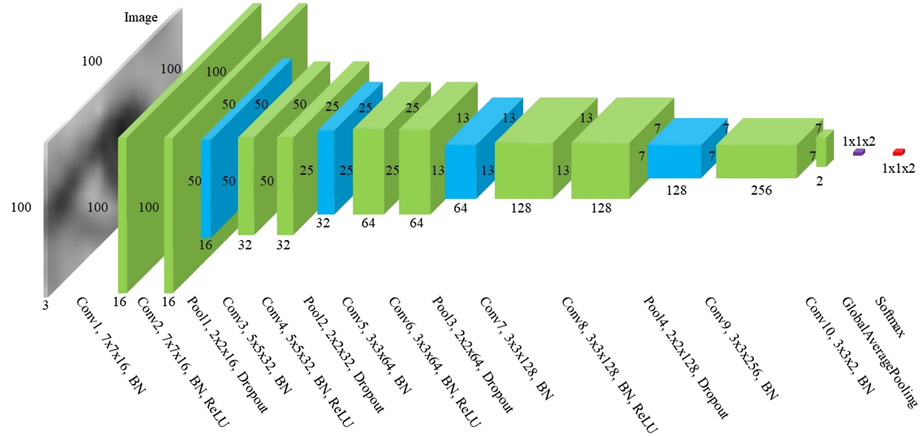

**Fig. 4.** The proposed eyes classification network. The network is based on nine sequential layers of convolution with filters of sizes 7 × 7, 5 × 5 and 3 × 3. Following the convolution layer groups are the ReLU activation functions and the average pooling layers. The global average pooling layer is 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.

## 4 Experimental Result

### 4.1 Dataset Preparation

The eye detection network is trained on the CEW (Closed Eyes In The Wild) [25], 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, and 1,231 subjects with eyes open. The image size is 100 × 100 (pixels) and it is extracted the eye patches of 24 × 24 from the central location of 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 the face of 23 different persons. The eye position label is set manually and generated coordinate of ground-truth bounding box based on this position with 36 × 36 size. GI4E is a dataset 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 image resolution is 800 × 600 pixels in PNG format. 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 split into two subsets with 80% for training and 20% for the testing phase.

The eyes classification network is trained and evaluated on Closed Eyes In The Wild (CEW) dataset [22]. 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 training phase is implemented on GeForce GTX 1080Ti GPU, tested on Intel Core I7-4770 CPU @ 3.40 GHz, 8 GB of RAM (PC), and quad-core ARM Cortex-A57 CPU, 4 GB of RAM (Jetson Nano device). Many configurations have been used in the training phase. The network used the Stochastic Gradient Descent optimization method, the batch size of 16, the weight decay is $5.10^{-4}$, the momentum is 0.9, the learning rates from $10^{-6}$ to $10^{-3}$. In order to generate the best bounding box, the threshold of IoU (Intersection over Union) is set by 0.5. 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

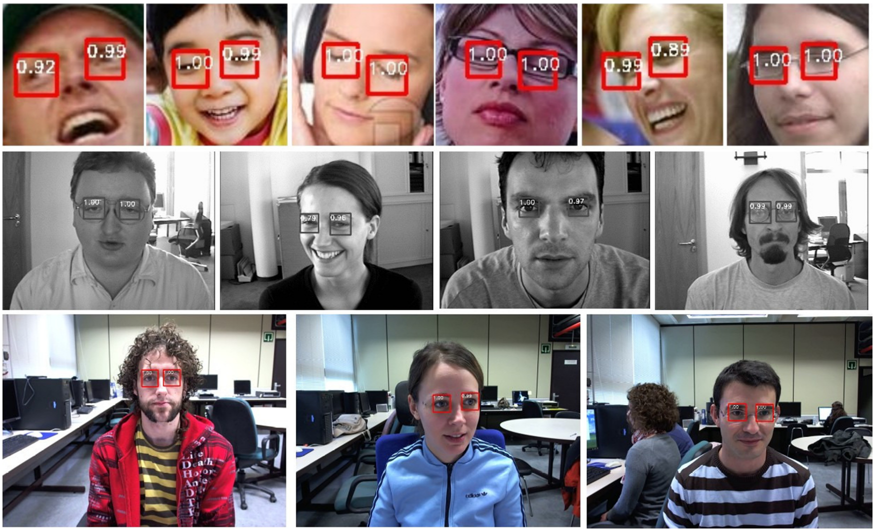

**Fig. 5.** The qualitative result of the eye detection network on CEW dataset (first row), on BioID Face dataset (second row), and on GI4E dataset (third row). This network can detect eye location in different situations: head poses, the glasses-wearing, laboratory environment. The number in each bounding box shows a confidence score of prediction.

Each network in the pipeline is individually trained and tested on the dataset of image and a comprehensive network was tested on a real-time system using a common camera connecting the PC based on CPU and Jetson Nano device with the quad-core ARM Cortex-A57 CPU. As a result, 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 eye detection network on CEW, BioID Face, and GI4E test set 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.

**Table 1.** The testing result of the eye detection network on CEW, BioID Face and GI4E test set.

| Dataset | Average precision (%) |
|---|---|
| CEW | 96.48 |
| BioID Face | 99.58 |
| Gi4E | 75.52 |

**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 |

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 [20]. 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 $< 0.5$ m), the images obtained via the camera are mainly images in the frontal face. This condition improves eye detection and open and closed eye classifier. Table 3 shown the speed testing result of the eye state recognizer on the camera.

**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 |

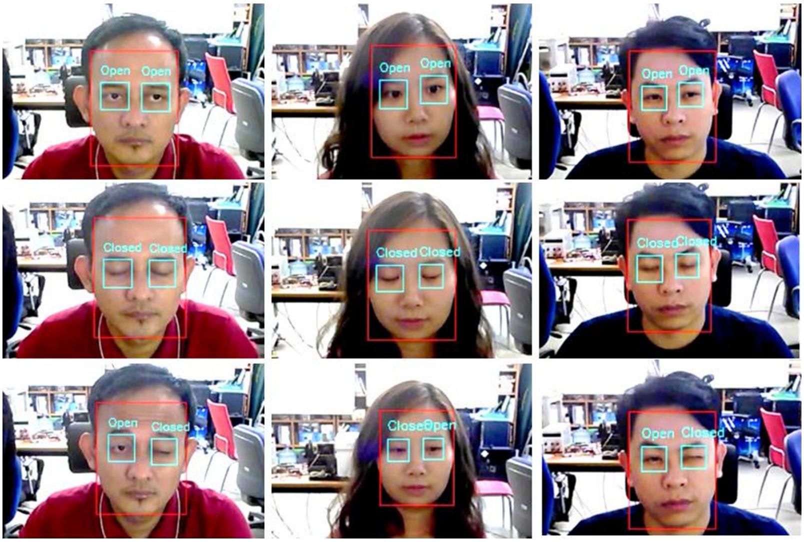

**Fig. 6.** The qualitative result when testing on camera connects with Jetson Nano device with three participants (two males and one female). The result shows in two open eyes (first row), two closed eyes (second row), one closed eye and one ope eye (third row).

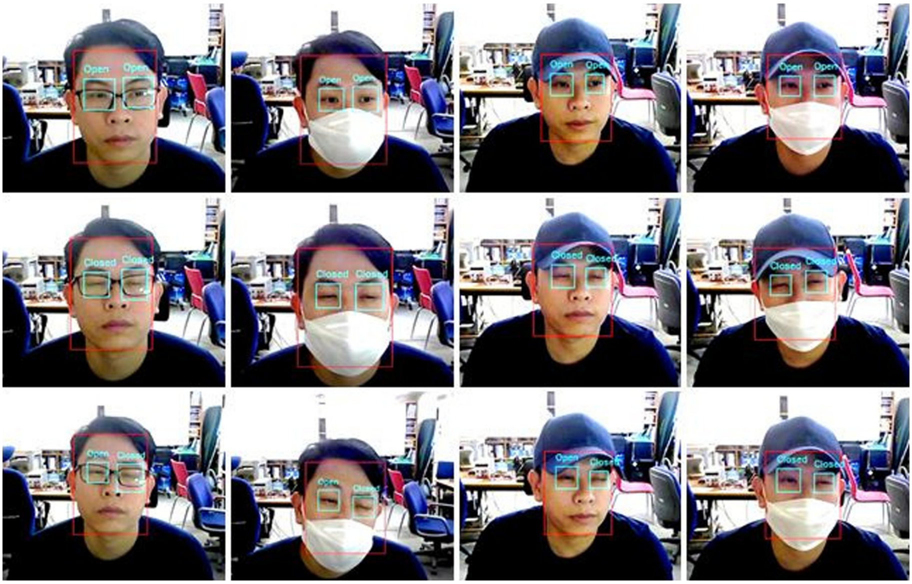

**Fig. 7.** The qualitative result when testing on camera connects with Jetson Nano device with four situations: glasses (first column), face mask (second column), hat (third column), face mask and hat-wearing (fourth column).

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. Figure 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°, vertical rotation head at an angle greater than 90°, 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.

## 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 optimization of the number of parameters and computation cost makes it can be applied on edge devices and CPU-based computers. The eye state recognizer will be integrated with several modern techniques and advanced optimization in the future. On the other hand, the dataset needs to collect and annotate under a variety of conditions to ensure this recognizer works properly such as glasses, hat, and face mask-wearing.

**Acknowledgement.** This work was supported by the National Research Foundation of Korea (NRF) grant funded by the government (MSIT).(No.2020R1A2C2008972).

## References

1. The bioid face database. https://www.bioid.com/facedb. Accessed 23 Oct 2020
2. Gi4e - gaze interaction for everybody. http://www.unavarra.es/gi4e/databases?languageId=1. Accessed 23 Oct 2020
3. Road traffic injuries. https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries. Accessed 22 Oct 2020
4. Bulling, A., Ward, J., Gellersen, H., Tröster, G.: Eye movement analysis for activity recognition using electrooculography. IEEE Trans. Pattern Anal. Mach. Intell. **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. Physiol. Meas. **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. Multimedia Comput. Commun. Appl. **3**, 2:1–2:20 (2007)
8. 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
9. Hsieh, C.S., Tai, C.C.: An improved and portable eye-blink duration detection system to warn of driver fatigue. Instrum. Sci. Technol. **41**(5), 429–444 (2013). https://doi.org/10.1080/10739149.2013.796560
10. Jo, J., Lee, S., Jung, H., Park, K., Kim, J.: Vision-based method for detecting driver drowsiness and distraction in driver monitoring system. Opt. Eng. **50**, 7202 (2011). https://doi.org/10.1117/1.3657506
11. 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. Opt. Eng. **54**(3), 1–19 (2015). https://doi.org/10.1117/1.OE.54.3.033103
12. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. Commun. ACM **60**(6), 84–90 (2017). https://doi.org/10.1145/3065386
13. Królak, A., Strumiłło, P.: Eye-blink detection system for human–computer interaction. Univers. Access Inf. Soc. **11**(4), 409–419 (2012). https://doi.org/10.1007/s10209-011-0256-6
14. Kégl, B.: The return of adaboost.mh: multi-class hamming trees (2013)
15. 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 2007), pp. 481–487 (2007). https://doi.org/10.1109/CRV.2007.54
16. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998). https://doi.org/10.1109/5.726791
17. Lee, W., Lee, E.C., Park, K.: Blink detection robust to various facial poses. J. Neurosci. Methods **193**, 356–72 (2010). https://doi.org/10.1016/j.jneumeth.2010.08.034
18. Mita, T., Kaneko, T., Hori, O.: Joint Haar-like features for face detection. In: Tenth IEEE International Conference on Computer Vision (ICCV 2005) Volume 1, vol. 2, pp. 1619–1626 (2005). https://doi.org/10.1109/ICCV.2005.129
19. 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
20. Nguyen, D.L., Putro, M.D., Jo, K.H.: Eyes status detector based on light-weight convolutional neural networks supporting for drowsiness detection system. In: IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society, pp. 477–482 (2020). https://doi.org/10.1109/IECON43393.2020.9254858
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. Shang, W., Sohn, K., Almeida, D., Lee, H.: Understanding and improving convolutional neural networks via concatenated rectified linear units. CoRR abs/1603.05201 (2016). http://arxiv.org/abs/1603.05201
24. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556 (2015)
25. Song, F., Tan, X., Liu, X., Chen, S.: Eyes closeness detection from still images with multi-scale histograms of principal oriented gradients. Pattern Recogn. **47**(9), 2825–2838 (2014). https://doi.org/10.1016/j.patcog.2014.03.024
26. Szegedy, C., et al.: Going deeper with convolutions. CoRR abs/1409.4842 (2014). http://arxiv.org/abs/1409.4842
27. Trutoiu, L.C., Carter, E.J., Matthews, I., Hodgins, J.K.: Modeling and animating eye blinks. ACM Trans. Appl. Percept. **8**(3) (2011). https://doi.org/10.1145/2010325.2010327
28. Viola, P., Jones, M.: Robust real-time face detection. Int. J. Comput. Vis. **57**, 137–154 (2004). https://doi.org/10.1023/B:VISI.0000013087.49260.fb
29. Wu, J., Trivedi, M.M.: An eye localization, tracking and blink pattern recognition system: algorithm and evaluation. TOMCCAP **6** (2010). https://doi.org/10.1145/1671962.1671964