Two Stream Deep Convolutional Neural Network for Eye State Recognition and Blink Detection

Ritabrata Sanyal
Dept. of Computer Science,
Kalyani Government Engineering College
Kolkata, India
sritabrata@gmail.com

Kunal Chakrabarty
Dept. of Computer Science
Kalyani Government Engineering College
Kolkata, India
kunalchakrabarty@gmail.com

***Abstract*—Eye state recognition and blink detection has been an important research problem in various fields like driver fatigue and drowsiness measurement, dry eye detection, video spoofing detection, psychological status analysis and many others. Hence an automated eye state classification and blink detection algorithm which is robust to a variety of conditions is required for this purpose. To this end, we propose a novel approach towards detection of eye blinks from a video stream by classifying the eye state of every frame as open or closed. First the eyes are localized from a frame with robust state-of-the-art facial landmark detectors. Then binary masks of the eyes are computed to capture and focus on how much the eyes are open. We propose a novel two stream convolutional neural network model which is jointly trained with the extracted eye patches, their masks as inputs and the corresponding eye state as output. With the eye state predicted by our network for every frame, we model a Finite State Machine to check for blinks by comparing number of consecutive frames with eyes closed against average human blink duration. Extensive experimentation has been done on a various number of popular benchmark datasets both for eye state classification and blink detection. Our proposed eye state classifier achieves a 3.2% and 3.86% improvement over the state-of-the-art in terms of accuracy and equal error rate (EER). The blink detector achieves a 1-2 % improvement over the state-of-the-art in terms of precision and recall. Hence our algorithm outperforms the existing methods for eye state classification and blink detection to the best of our knowledge.**

***Index Terms*—eye blink detection, eye status recognition,two stream joint networks, convolutional neural networks, facial landmarks, deep learning, video analysis**

I. Introduction

Eye state recognition and blink detection is a very important problem in computer vision. They are useful in many fields such as drowsiness detection, facial expression recognition, fatigue measurement, human computer interaction technology and many others. The main purpose of this paper is to develop an automated eye state classifier and blink detector system that is robust to variety of facial poses, expressions, blur, occlusion and illumination conditions.

A. Related Work

In this section we will explore some of the previous works done in this field. A lot of the methods discussed below have implemented some variation of the Viola and Jones' algorithm [1]

Morris et al. [2] suggested using spatio-temporal filtering and variance maps to locate the face and eye feature points with a Lucas Kanade feature tracker for eye tracking. The blink detection, however is affected by head movements and a large drop in accuracy is seen in such cases.

Sirohey et al. [3] suggested using eye corners, eyelids and irises to track eye movements and eye blinks. Two types of tracking were implemented, frame to frame tracking and flow based tracking. The paper claims that the flow based tracking approach is the better of the two and gets around 90% accuracy in tracking eyelids. Chau et al. [4] used eye blink detection and tracking to build a human computer interaction system. The algorithm used by Grauman et al. [5] was used to build the BlinkLink blink detector system that could reliably classify blinks as being voluntary or involuntary based on duration with a claimed accuracy of 95.3 %. Change in lighting conditions was noticed to have an adverse effect on said accuracy.

Another gaze tracking method was suggested by Orozco et al. [6] where two appearance based trackers (ABT's) were combined for eyelid and iris tracking respectively. The ABT assigned for eyelid tracking was able to robustly identify blinks, making it suitable for real time applications. Both forced and spontaneous blinks were handled correctly. After blinking the iris tracker correctly recovers the adaptation that was lost during the blink. The authors did not explicitly use the model for detecting blinks and so no accuracy was mentioned.

More recently, Heishman et al. [7] relied on using image flow analysis for eye blink analysis including analyzing ambiguous blink states like partial blinks or asymetric eyelid movements. These ambiguous blink parameters are calculated and the flow analysis calculates the magnitude and direction of eye movement. A deterministic finite state machine is used for calculating eye blink characteristic data. With the advent of deep learning based CNN's eye blink detection witnessed a breakthrough.

Huang et al. [8] uses a P-FDCN based neural network for blink detection towards driver fatigue detection. A fatigue detection convolutional network is built and ptojection cores are then incorporated into it to create a P-FDCN. The paper claims 94.9% recognition rate on the common eyes in the wild

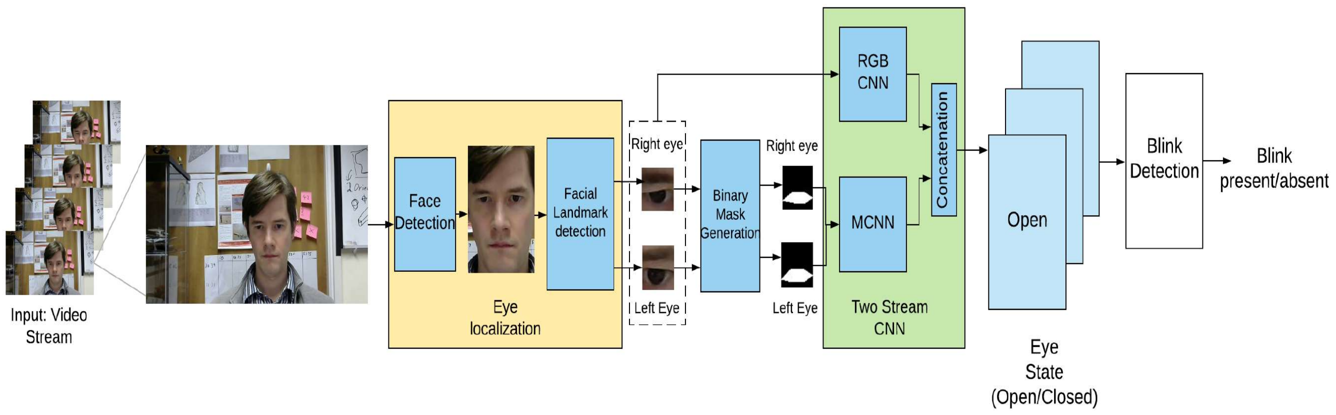


Fig. 1: Workflow of the entire process

dataset. Al-gawwam et al. [9] suggests using facial landmarks for eye extraction coupled with a Savitzky–Golay (SG) filter for smooth eye contours and blink detection. The model shows great results in a variety of lighting conditions and head poses.

B. Contribution

- Our proposed system takes in a video stream of a person's face as input, and for each frame classifies the state of their eyes. Finally, using the eye status information from every frame, our framework can determine the number of times the person has blinked.
- In this work we propose a novel two stream convolution neural network architecture for eye status recognition and eye blink detection that is robust to a variety of conditions. The model is based on joint training of two convolution neural networks which take in an eye patch and a masked eye patch as inputs and predicts whether the eye is open or closed.
- We also propose a novel eye localization technique based on facial landmark detection.
- The competency of our proposed system is evaluated on the popular benchmark datasets like ZJU Eye Blink [10], Closed Eyes in the Wild (CEW) [11], Talking Face [12], EyeBlink8 [13] and Blinks Dataset [14].
- The results show that our proposed two stage model outperforms other common existing methods for eye status classification and blink detection in a real time environment.

The paper is organized as follows. Section II describes the model architecture and the methodologies employed in this paper in detail. In Section III we discuss the experimental results and compare our model with the existing state-of-the-art. Finally the paper is concluded in Section IV

II. Methodologies

A. System Overview

Here, we give a brief description of the entire pipeline. We'll expound the intricacies of each component in the following subsections.

When a video stream is given as input, the system first detects the face region and the landmarks positions. With the information of the eye-brow and eye landmarks coordinates, the eyes are localized. Then a copy of both the extracted eye patches are binary masked by computing a convex hull on the set of their corresponding eye landmark coordinates. The extracted eye patches and their corresponding binary masks are then fed into two convolution neural network (CNN) streams which are jointly trained to predict the eye state i.e open or closed. The system records the eye state of each frame. A blink is said to be detected on a particular frame for which the eyes are open again for the first time after being closed for a number of frames. This number of frames threshold generally depends on the average blink duration of humans which is approximately range from 100 to 400 ms [15]. The overall workflow of the entire pipeline is depicted in Figure 1.

B. Eye Patch Extraction

When an image is given as input to the eye state recognition system, at first the facial region is detected. In our work, we have used the detector based on Multi Task Cascaded Convolutional Networks (MTCNN) [16] since it is one of the most accurate face detectors available in literature, especially suited for a real time purpose. Once the face is detected, the landmarks are detected by a robust state-of-the-art architecture based on Convolutional Experts Constrained Local Model (CE-CLM) [17] which uses a Point Distribution Model (PDM) for capturing landmark shape variations and Patch Experts which model local appearance variations of each landmark. It predicts 68 landmark positions for the eyes, eyebrows, lips, jawline and nose. Out of all the 68 landmarks, those corresponding to both the eyes and eyebrows are extracted. Using these information, we localize each eye patch within a bounding box as follows: The top and bottom bounds of an eye patch is set to the uppermost landmark coordinate of the corresponding eyebrow and the lowermost landmark coordinate of the eye respectively. The left and right bounds are set to the leftmost and rightmost landmark coordinates of the eye respectively. We use the eyebrow landmark positions to determine the upper bound of an eye patch, since it helps us to capture the full view of even a closed eye. In a closed eye, the landmarks of the upper and lower eyelids lie in a very close vicinity of each other, quite contrary to those of an open eye. If only the eye landmarks were used for determining the upper bound, the height of the extracted closed eye patch would have been very less compared to an open eye patch, which would in turn pose a major problem for the convolutional nets to learn useful features. The eye patch extraction technique is shown in Figure 2.

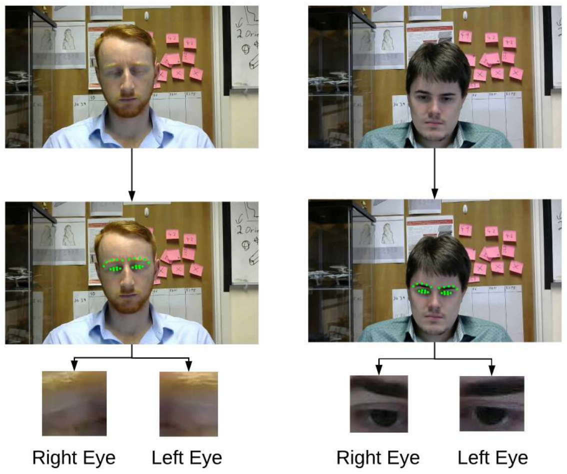


Fig. 2: Eye patch extraction

C. Binary Mask Generation

After extracting the eye patches, we make a copy of the patches and mask them. The binary mask is generated by computing a convex hull on the set of their corresponding eye landmark coordinates. The pixel values of the region inside the hull is set to a value of 1 (white) and those outside of it is set to 0 (black). Mask generation is done since it gives us an easy to compute feature map which can accurately distinguish a closed eye from an open one, by focusing just on the degree of openness of the eye. A mask corresponding to an open eye patch will have a greater percentage of it's pixel values equal to 1, contrary to that of a closed eye patch which will have most of its pixel values equal to 0. The high contrast between the white foreground (eye region only) and the black background (rest of the eye patch) in the mask will facilitate the convolutional neural network to focus just on the external shape and contour of the eye and thus, learn necessary mappings to the ground truth class (open/closed). Mask generation technique is shown in Figure 3.

D. Model Architecture

Upon extracting the eye patches and their corresponding binary masks as delineated in the previous sections, they are

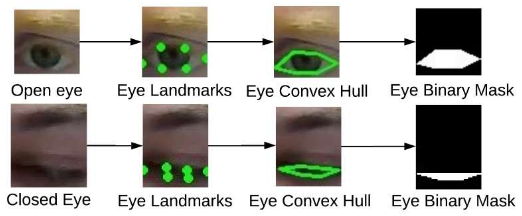


Fig. 3: Binary mask generation

fed as input to a two stream convolutional network(TSCNN). The CNN stream which receives an eye patch (RGB image) as input is termed the RGB-CNN, and the other stream which receives the binary mask of the same eye patch as input is termed the Mask-CNN or MCNN. The RGB-CNN network aims to learn the global features of the enitre eye patch, whereas the MCNN network focuses on learning the local features like the shape, contour of the eye and spatial distribution of white (eye-region) pixels and black (non-eye region) pixels in the eye patch. Given their different structures, the RGB-CNN and MCNN can extract different discriminative features of an eye patch. Hence both the networks are combined and jointly trained for eye state classification.

1) RGB-CNN: A simple convolutional neural network is proposed as a baseline model. It takes in a resized 32 x 32 RGB eye patch as input. We use a modified version of the classic LeNet-5 [18] architecture to suit our purpose. We use three convolutional layers, three max pooling layers and two fully connected layers. The contrasting differences between our network and LeNet-5 is that we use a max pooling layer after every convolutional layer to decrease the spatial dimensions of the feature maps [19] to bring down the total number of parameters in the network which in turn helps us to prevent overfitting [20]. The structure of the proposed RGB-CNN is shown in Figure 4a.

2) MCNN: This our second baseline model. This is a convolutional neural network which takes in a resized 32 x 32 masked eye patch as input . The model architecture is almost the same as RGB-CNN except the number of neurons were reduced by half in both the fully connected layers.This is done to reduce overfitting, which arises because the input binary mask image has only one color channel and much less variations as compared to the corresponding RGB eye patch. Thus the model complexity of the MCNN network needs to be less than RGB-CNN to learn the same quality of discerning features when trained on the same amount of data. The structure of the proposed RGB-CNN is shown in Figure 4b.

3) TSCNN: This is our proposed two stream convolutional network architecture. At first the two baseline models viz. RGB-CNN and MCNN are individually trained. Then both of them are jointly fine-tuned by concatenating their corresponding top fully connected layers. A fully connected layer is added on top of the concatenated layer along with a softmax function for classifying closed and open eyes. Inspired by [21], [22] , at training time, we freeze the weight values of the convolutional layers of the pre-trained RGB-CNN and MCNN models and retrain their top layers along with the concatenated and fully connected layer of the joint model. We also use a similar kind of an integrated loss function to train the two stream model, as delineated in [21]. The loss function of the entire network is mathematically represented as:

$$L_{TSCNN} = \sum_{i=1}^{3} \psi_i L_i \tag{1}$$

where L_i and ψ_i is the loss function and tuning hyperparameter weight of the i^{th} model respectively. ($i = 1$ for RGB-CNN; $i = 2$ for MCNN; $i = 3$ for the fused model). We empirically set the hyperparameters ψ_1, ψ_2, ψ_3 to 1, 1 and 0.5 respectively. Each loss function L_i is a cross entropy loss which is defined as:

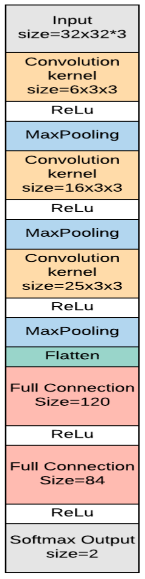
$$L_i = -\sum_{j=1}^{k} y_j log(\hat{y}_{i,j}) \tag{2}$$

where k is the number of classes which in our case is equal to 2 (open/closed), y_j is the j^{th} value of the ground truth label, $\hat{y}_{i,j}$ is the j^{th} value of softmax output of network i.

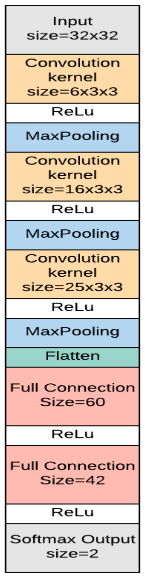
$$\hat{y}_{i,j} = g_s(l_{i,j}) \tag{3}$$

where $l_{i,j}$ is the j^{th} logit value of network i and $g_s(.)$ is the softmax activation function. Thus the loss function of the two stream joint network can be succinctly defined by:

$$L_{TSCNN} = -\sum_{i=1}^{3}\sum_{j=1}^{k} \psi_i y_j log(g_s(l_{i,j})) \tag{4}$$



(a) RGB-CNN



(b) Mask-CNN

Fig. 4: Model architecture

Finally the decision $\hat{D}$ of the TSCNN network is obtained using output of softmax layer of network 3 as:

$$\hat{D} = \arg\max_j \hat{y}_{3,j} \tag{5}$$

From Figure 5, we can see how the three softmax outputs (Output1, Output2 and Output) of TSCNN network is used to calculate the loss function L_{TSCNN} at training time, while the softmax output of only the integrated model is used at prediction time. To prevent overfitting, we have used the dropout regularization technique [23]. We use the Adam optimizer [24] with a learning rate of 0.001 to train the TSCNN network for 120 epochs.

E. Face Level Prediction

We do not consider the left and right eyes separately to train the TSCNN network. This is because doing so will increase the complexity of the network as it will have to discriminate between four classes (left-open, left-closed, right-open, right-closed) instead of two (open, closed) under the constraint of same amount of training data. Instead we exploit the bilateral symmetry property of human faces. After extracting the left and right eyes from a face, we flip the right eye images about a vertical axis to create an equivalent left eye image. Hence the network is trained with all the actual and generated left eye images.

At prediction time, the eyes are extracted from a face and the right eye to left eye flipping step is similarly followed. The TSCNN softmax outputs of the actual left eye and the generated left eye are averaged to give the final face level prediction.

F. Blink Detection

From a video stream as input, we detect whether the person's eyes are open or closed for every frame. A blink is said to be detected when the eyes are open again after being closed for a certain duration. The average blink duration in humans

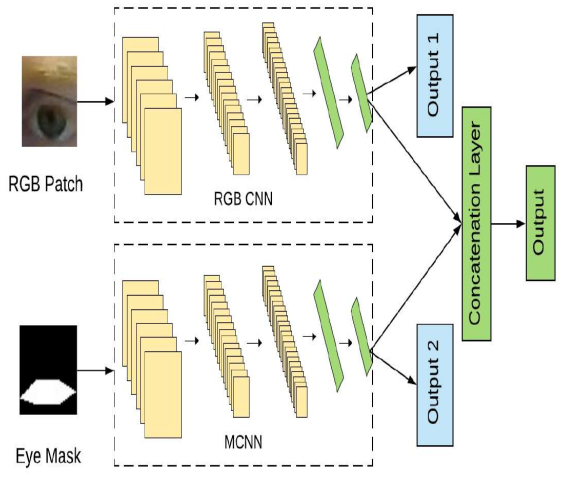


Fig. 5: Two stream model (TSCNN) architecture

ranges from 100 to 400 ms [15]. For a 30 fps camera, that amounts to 3 - 12 frames. Thus to detect a blink, our objective is to find for how many frames the eyes remain closed before opening again. If the eyes are closed for any number of frames lying outside this range, the blink is considered to be invalid. A Finite State Machine (FSM) is used to model this problem of checking true and false blinks. Here we define two parameters beforehand viz. $lower\_blink\_frame\_count := fps \times 0.1$ and $upper\_blink\_frame\_count := fps \times 0.4$ which are the minimum and maximum number of frames the eyes need to be closed for the FSM to detect a true blink (fps is the streaming rate of the video camera in frames/sec). We also maintain a counter called $closed\_frame\_counter$ to count number of frames for which the eyes are closed. It is initially set to zero. The initial state of the FSM is State 0 and the final state is State 3. When closed eyes are detected, the following states occur as shown in Figure 6 :

1) The FSM transitions from State 0 to State 1. In this state, the $closed\_frame\_counter$ gets incremented by one for every frame. If $closed\_frame\_counter$ exceeds $upper\_blink\_frame\_count$, an invalid blink is detected and the state is reset to State 0. The $closed\_frame\_counter$ is also reset to zero.
2) When open eyes are detected, the FSM transitions from State 1 to State 2. If $closed\_frame\_counter$ is less than $lower\_blink\_frame\_count$, an invalid blink is detected and the state is reset to State 0. The $closed\_frame\_counter$ is also reset to zero.
3) If $closed\_frame\_counter$ is more than $lower\_blink\_frame\_count$, the FSM transitions from State 2 to State 3. A true blink is detected and the state is reset to State 0. The $closed\_frame\_counter$ is also reset to zero.

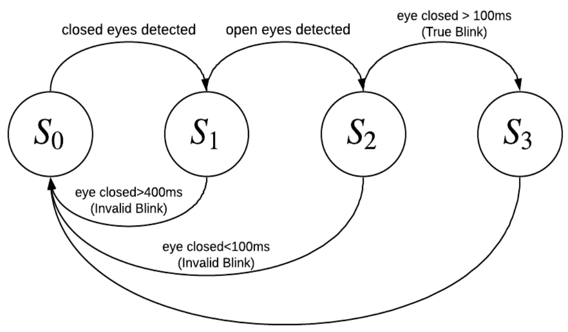


Fig. 6: Finite state machine to detect blinks

III. Experimental Results

A. Dataset Overview

There are many publicly available benchmark datasets pertaining to eye state recognition and blink detection. The datasets which are used in this study are briefly described as follows :

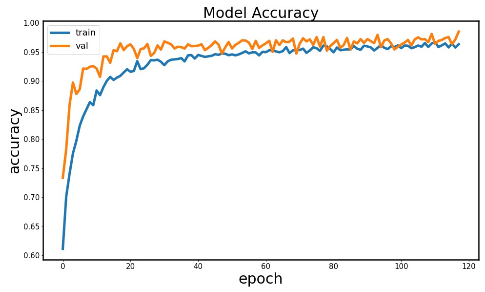


Fig. 7: Training and validation accuracy vs epochs for TSCNN model

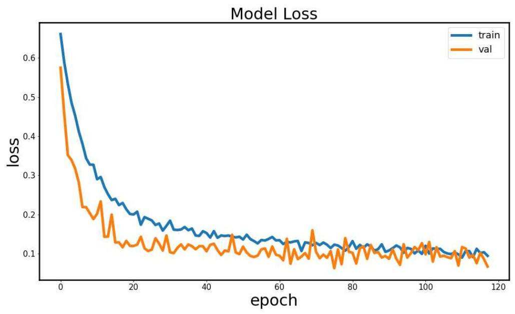


Fig. 8: Training and validation accuracy vs epochs for TSCNN model

- Closed Eyes in the Wild (CEW) : This dataset [11] contains 2423 subjects, among which 1192 subjects with both eyes closed are collected directly from Internet, and 1231 subjects with eyes open are selected from the Labeled Face in the Wild database. It consists of real world images in unconstrained settings subject to challenging variations caused by lighting, blur and occlusion.
- Blinks Dataset : It [14] consists of HD webcam footage of the faces of 6 volunteers watching a nature documentary. There is 5 minutes of footage for each volunteer, and the videos demonstrate a considerable spread of blinking rates, blink durations, and other challenges.
- ZJU : This dataset [10] consists of 80 videos of 20 individuals. Each individual has 4 clips: frontal view, upward view, with glasses, and without glasses. The few seconds clips acquisition were done with a 30 fps video camera with a resolution of 320 × 240. There is no facial expression and almost no head movements. This dataset has different numbers of ground truth eye blinks reported, as shown in Table VI
- Eyeblink8 : This dataset contains videos subject to variations in facial expressions, head movements, and looking down on a keyboard. The videos were captured at 30

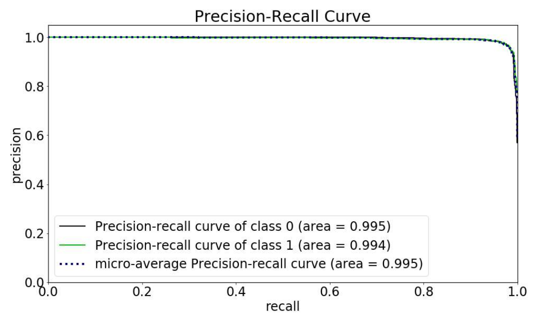


Fig. 9: Precision-Recall curve at eye level

fps with a resolution of 640 x 480 and average length from 5000 to 11,000 frames. It consists of 408 blinks on 70,992 video frames, as annotated by [25].

- Talking face : This dataset [12] consists of a single video of a subject talking in front of the camera making different facial expressions. The video is acquired at 25 fps with a resolution of 720 x 576 and has 61 annotated blinks [25].

In this work, we use the CEW and the Blinks dataset to train, validate and test the eye state classification models. Since there is an imbalance between open and closed classes in the CEW dataset, we take 1408 frames with closed eyes and 1369 frames with open eyes from the Blinks dataset to balance the classes (there were a total of 1408 frames annotated with closed eyes in the Blinks Dataset). Thus in our modified dataset, we have a total of 2600 faces per class. The eyes patches are extracted and flipped using the techniques delineated in Section II-B and Section II-E, giving us a total of 5200 eye patches per class. Some data augmentation techniques like random rotation, height shift and width shift are also used to increase size of dataset to prevent overfitting [19]. Following the standard practice, we use 70% of the data to train, 20% to validate and 10% to test the models.

To evaluate the blink detection algorithm, we use ZJU, Eyeblinks8 and Talking Face datasets.

TABLE I: Comparison of Equal Error Rate (EER) of TSCNN network with other models on the open database of NIR eye images. The red and blue colors represent the best and second best model in terms of least error rate respectively. (EER unit is in %)

Models	EER (%)
HOG-SVM [26] [27]	80.74
Fuzzy Segmentation [28]	32.53
Deep Residual CNN [26]	1.89
RGB-CNN	1.64
MCNN	2.57
TSCNN	1.18

TABLE II: Comparison between different baseline models and our proposed TSCNN model on the test set. The red and blue colors represent the first and second most accurate models respectively. Model names in bold letters are the networks used in our study.

Models	Accuracy(%)	EER(%)
Fuzzy-Segmentation [28]	80.23	16.28
HOG-SVM [27]	84.52	11.41
Deep Residual CNN [26]	95.33	5.87
RGB-CNN	94.75	6.12
MCNN	92.56	7.04
TSCNN	98.51	2.01

TABLE III: Eye level prediction metrics of the TSCNN network. (unit %)

Metric (%)	**Classes**		**Avg**
	Closed	**Open**	
Accuracy	98.51	98.51	98.51
Precision	98.21	98.51	98.36
Recall	98.42	98.02	98.22
F1 Score	98.31	98.26	98.23
Type I Error	2.61	2.32	2.47
Type II Error	2.32	2.61	2.47
EER	2.47	2.47	2.47

TABLE IV: Face level prediction metrics of the TSCNN network. (unit %)

Metric (%)	**Classes**		**Avg**
	Closed	**Open**	
Accuracy	92.14	92.14	92.14
Precision	94.23	91.38	92.81
Recall	91.37	95.11	93.24
F1 Score	92.78	93.21	93.03
Type I Error	6.27	8.43	7.35
Type II Error	8.43	6.27	7.35
EER	7.35	7.35	7.35

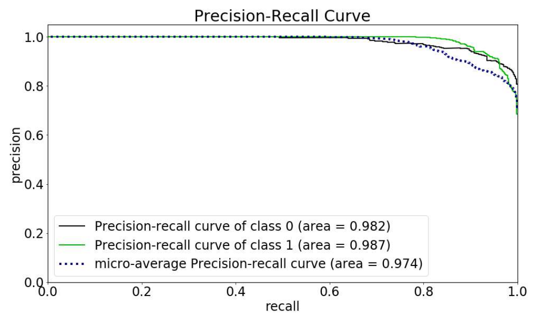


Fig. 10: Precision-Recall curve at face level

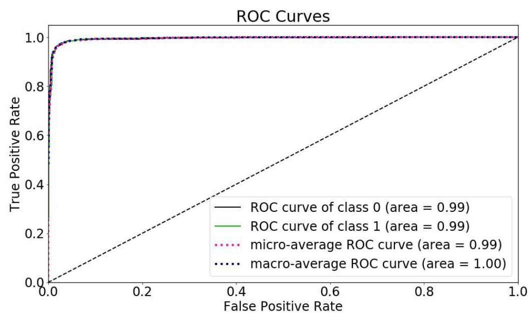


Fig. 11: ROC curve at eye level

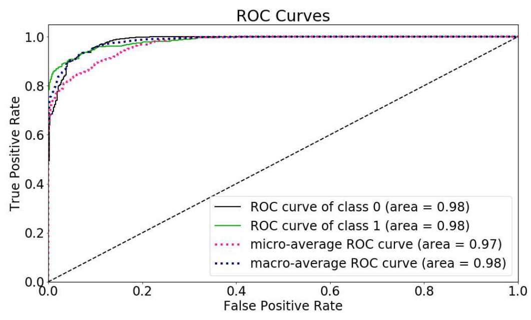


Fig. 12: ROC curve at face level

B. Eye State Classification Results

We use standard metrics like accuracy, precision, recall, F1 score, Type I Error Rate, Type II Error Rate and Equal Error Rate (EER) to evaluate our proposed TSCNN model. The training, validation accuracy and loss curves of the TSCNN model are shown in Figure 7 and Figure 8. The Precision-Recall and ROC curves as shown in Figure 10, Figure 9, Figure 12, Figure 11 are computed on the test set of our modified dataset. Table III lists all metric values of the TSCNN network with an eye as input as evaluated on the test dataset, whereas Table IV lists the same with an entire face image as input (as delineated in Section II-E).

For comparison of our proposed model with the existing literature, we use the Segmentation based Fuzzy Logic System [28], Deep Residual CNN System [26], HOG-SVM System [27] as baselines. We also use the RGB-CNN and MCNN models as baselines for comparing the competency of the integrated TSCNN network to them. All these models are evaluated on the test set of our modified dataset. From Table II, we can see that our proposed TSCNN model outperforms other baseline models including RGB-CNN and MCNN in terms of accuracy and Equal Error Rate (EER).

To realize the competency of our TSCNN model to generalize on different datasets, we test it on open NIR eye images dataset [29] and compare with the baseline models. The evaluated EER values of the models other than those used in our network, are already provided in [26]. From Table I, we can see our TSCNN network achieves an Equal Error Rate of 1.18%, which outperforms the other models on the NIR dataset.

C. Blink Detection Results

The proposed eye blink detection algorithm is evaluated by comparing detected blinks with ground-truth blinks on the popular benchmark datasets like ZJU, Eyeblink8 and Talking face as described in Section III-A. Precision and recall metrics are used as evaluation metrics. To calculate these metrics, we need ground truth negative classes. In order to do that, we divide the number of frames with open eyes in the datasets by average blink duration (7 frames)(See Section II-F) and this is used as negative (number of non eye blinks). We followed the approach delineated by [13]. From Table V, we can see that, our proposed blink detection method outperforms other existing methods in terms of precision and recall on the benchmark datasets.

D. Programming Frameworks used

We use OpenCV [31] library for all image processing tasks, Openface 2.0 [32] toolkit for facial region and facial landmark detection. For building the neural networks, we use Keras [33] library with Tensorflow backend.

IV. Conclusion

In this work, we have proposed a pipeline for eye status recognition and blink detection using facial landmarks and a two stream convolutional neural network architecture. Experimental results show that our proposed TSCNN model yields more robust and accurate eye state recognition than its components RGB-CNN and MCNN can individually yield. Hence integrating the features of both an eye patch and its mask yield results better than they individually do. Also upon comparing our proposed method with the existing methods for eye state classification we can easily see our model outperforms theirs'

TABLE V: Result of the proposed method and of some popular techniques

Reference	Dataset	Precision (%)	Recall (%)
Fogelton et al. [25]	ZJU	100	98.08
Lee et al. [30]	ZJU	94.4	91.7
Druratovsky et al. [13]	ZJU	91	73.1
Al-gawwam et al. [9]	ZJU	100	98.01
Proposed Method	ZJU	100	99.3
Fogelton et al. [25]	Talking face	95	93.44
Lee et al. [30]	Talking face	83.3	91.2
Druratovsky et al. [13]	Talking face	92.2	96.7
Al-gawwam et al. [9]	Talking face	98.38	98.38
Proposed Method	Talking face	99.42	99.29
Fogelton et al. [25]	Eyeblink8	94.69	91.91
Al-gawwam et al. [9]	Eyeblink8	96.65	98.78
Proposed Method	Eyeblink8	98.85	99.12

TABLE VI: Different numbers of reported ground truth eye blinks in the ZJU dataset

Reference	Blinks
Druratovsky et al. [13]	264
Fogelton et al. [25]	261
Radlak et al. [34]	258
Lee et al [30]	255

TABLE VII: Processing time and rate of a face level prediction of our proposed TSCNN network

Time(ms)	37.2
FPS (frames/s)	27

by a large margin. We also used our TSCNN eye state classifier to detect blinks by modelling the problem with a Finite State Machine. Experimental results reveal our eye blink detection system outperforms other state-of-the-art blink detectors. Yet in terms of time, our system achieves an average processing rate of 27 frames/sec, which is not the best we could have for real time scenarios. Therefore to improve the average processing time, we will continue to work on this problem and explore other end-to-end convolutional network architectures.

References

[1] P. Viola, M. Jones *et al.*, "Rapid object detection using a boosted cascade of simple features," *CVPR (1)*, vol. 1, no. 511-518, p. 3, 2001.

[2] T. Morris, P. Blenkhorn, and F. Zaidi, "Blink detection for real-time eye tracking," *Journal of Network and Computer Applications*, vol. 25, no. 2, pp. 129–143, 2002.

[3] S. Sirohey, A. Rosenfeld, and Z. Duric, "A method of detecting and tracking irises and eyelids in video," *Pattern Recognition*, vol. 35, no. 6, pp. 1389–1401, 2002.

[4] M. Chau and M. Betke, "Real time eye tracking and blink detection with usb cameras," Boston University Computer Science Department, Tech. Rep., 2005.

[5] K. Grauman, M. Betke, J. Gips, and G. R. Bradski, "Communication via eye blinks-detection and duration analysis in real time," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1. IEEE, 2001, pp. I–I.

[6] J. Orozco, F. X. Roca, and J. Gonzàlez, "Real-time gaze tracking with appearance-based models," *Machine Vision and Applications*, vol. 20, no. 6, pp. 353–364, 2009.

[7] R. Heishman and Z. Duric, "Using image flow to detect eye blinks in color videos," in *2007 IEEE Workshop on Applications of Computer Vision (WACV'07)*. IEEE, 2007, pp. 52–52.

[8] R. Huang, Y. Wang, and L. Guo, "P-fdcn based eye state analysis for fatigue detection," in *2018 IEEE 18th International Conference on Communication Technology (ICCT)*. IEEE, 2018, pp. 1174–1178.

[9] S. Al-gawwam and M. Benaissa, "Robust eye blink detection based on eye landmarks and savitzky–golay filtering," *Information*, vol. 9, no. 4, p. 93, 2018.

[10] G. Pan, L. Sun, Z. Wu, and S. Lao, "Eyeblink-based anti-spoofing in face recognition from a generic webcamera," in *2007 IEEE 11th International Conference on Computer Vision*. IEEE, 2007, pp. 1–8.

[11] F. Song, X. Tan, X. Liu, and S. Chen, "Eyes closeness detection from still images with multi-scale histograms of principal oriented gradients," *Pattern Recognition*, vol. 47, no. 9, pp. 2825–2838, 2014.

[12] "Talking face dataset." [Online]. Available: https://www-prima.inrialpes.fr/FGnet/data/01-TalkingFace/talking_face.html

[13] T. Drutarovsky and A. Fogelton, "Eye blink detection using variance of motion vectors," in *Computer Vision - ECCV 2014 Workshops*, L. Agapito, M. M. Bronstein, and C. Rother, Eds., vol. 8927. Springer, 2015, pp. 436–448. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-16199-0_31

[14] W. J. Faithfull, "Blink detection dataset," 2015. [Online]. Available: https://will.faithfull.me/blinks-dataset/

[15] "Average duration of a single eye blink – human homo sapiens." [Online]. Available: https://bionumbers.hms.harvard.edu/bionumber.aspx?s=y&id=100706&ver=0

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

[17] A. Zadeh, Y. Chong Lim, T. Baltrusaitis, and L.-P. Morency, "Convolutional experts constrained local model for 3d facial landmark detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2519–2528.

[18] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[20] R. Caruana, S. Lawrence, and C. L. Giles, "Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping," in *Advances in neural information processing systems*, 2001, pp. 402–408.

[21] H. Jung, S. Lee, J. Yim, S. Park, and J. Kim, "Joint fine-tuning in deep neural networks for facial expression recognition," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2983–2991.

[22] L. Zhao, Z. Wang, G. Zhang, Y. Qi, and X. Wang, "Eye state recognition based on deep integrated neural network and transfer learning," *Multimedia Tools and Applications*, vol. 77, no. 15, pp. 19 415–19 438, 2018.

[23] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[25] A. Fogelton and W. Benesova, "Eye blink detection based on motion vectors analysis," *Computer Vision and Image Understanding*, vol. 148, pp. 23–33, 2016.

[26] K. Kim, H. Hong, G. Nam, and K. Park, "A study of deep cnn-based classification of open and closed eyes using a visible light camera sensor," *Sensors*, vol. 17, no. 7, p. 1534, 2017.

[27] L. Pauly and D. Sankar, "Detection of drowsiness based on hog features and svm classifiers," in *2015 IEEE International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*. IEEE, 2015, pp. 181–186.

[28] K. W. Kim, W. O. Lee, Y. G. Kim, H. G. Hong, E. C. Lee, and K. R. Park, "Segmentation method of eye region based on fuzzy logic system for classifying open and closed eyes," *Optical Engineering*, vol. 54, no. 3, p. 033103, 2015.

[29] W. Fuhl, T. Santini, D. Geisler, T. Kübler, W. Rosenstiel, and E. Kasneci, "Eyes wide open? eyelid location and eye aperture estimation for pervasive eye tracking in real-world scenarios," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*. ACM, 2016, pp. 1656–1665.

[30] W. O. Lee, E. C. Lee, and K. R. Park, "Blink detection robust to various facial poses," *Journal of neuroscience methods*, vol. 193, no. 2, pp. 356–372, 2010.

[31] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[32] T. Baltrusaitis, A. Zadeh, Y. C. Lim, and L.-P. Morency, "Openface 2.0: Facial behavior analysis toolkit," in *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*. IEEE, 2018, pp. 59–66.

[33] F. Chollet *et al.*, "Keras," https://github.com/fchollet/keras, 2015.

[34] K. Radlak and B. Smolka, "Blink detection based on the weighted gradient descriptor," in *Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013*. Springer, 2013, pp. 691–700.