16th International Conference on Machine Vision Applications (MVA)
National Olympics Memorial Youth Center, Tokyo, Japan, May 27-31, 2019.

# Spatio-temporal eye contact detection combining CNN and LSTM

Yuki Watanabe
Kyoto University
watanabe@ii.ist.i.kyoto-u.ac.jp

Atsushi Nakazawa
Kyoto University
nakazawa.atsushi@i.kyoto-u.ac.jp

Yu Mitsuzumi
Kyoto University
mitsuzumi@ii.ist.i.kyoto-u.ac.jp

Toyoaki Nishida
Kyoto University
nishida@i.kyoto-u.ac.jp

## Abstract

*Eye contact (mutual gaze) is fundamental for human communication and social interactions; therefore, it is studied in many fields. To support the study of eye contact, much effort has been made to develop automated eye-contact detection using image recognition techniques. In recent years, convolutional neural network (CNN) based eye-contact detection techniques are becoming popular due to their performance; however, they mainly use single frame for recognition. Eye contact is a human communication behavior, so temporal information, such as temporal eye images and facial poses, is important to increase the accuracy of eye-contact detection. We incorporate temporal information into eye-contact detection by using temporal neural network structures that combine CNNs and long short-term memory (LSTM). We tested several network combinations of CNNs and LSTM and found the best solution that uses the outputs of CNNs as well as the cell state vectors of LSTM in the fully connected layers. We prepared two types of eye contact video datasets. One dataset is based on online videos, and the other was taken by a first-person camera in assumed conversational scenarios. The results show that our method is better than the approaches that use single frames. Namely, our method performs 0.8781, while the existing method (DeepEC) performed 0.8319, in $F_1$-score.*

## 1 Introduction

Eye contact (mutual gaze) is a fundamental part of human communication and social interaction. In psychology, the 'eye-contact effect' is the phenomenon in which perceived eye contact with another human face affects certain aspects of the concurrent and/or immediately following cognitive processing [1]. Thus, eye contact greatly affects human behaviour in areas such as affective perceptions [2], social interactions [3] and development [4]. Eye contact is also used in medicine, such as in the diagnosis of autism spectrum disorders (ASDs) [5]. In dementia nursing, making appropriate eye contact is an important skill for communicating with patients [6, 7]. Our research mainly focuses on the nursing scenario, particularly on evaluating humane-care nursing skills for dementia by examining facial communication behaviours between caregivers and patients, such as the number of eye contact events and the relative facial positions and distances between caregiver and care receiver [8]. To enable such evaluation, we aim to develop a wearable care-skill evaluation system that gives care-skill scores and advice to users as feedback (Figure 2). To this end, we develop a system of automated eye-contact detection for caregiving using first-person videos (FPVs) taken using head-mounted cameras worn by caregivers(Figure 1).


Figure 1. Several scenes and first person views in an experiment of the skill analysis of the dementia nursing (Humanitude) using a simulated patient. In dementia nursing, skilled caregivers approach their faces close to the patients while making eye contacts.

Several efforts have been made to develop automated eye-contact detection using image-recognition techniques. Smith et al. [9] proposed an algorithm to detect *gaze-locking* (looking at a camera) faces using eye appearances and PCA+MDA. Ye et al. developed a pioneering algorithm that detects mutual eye gaze using wearable glasses [10, 11]. Petric et al. developed an eye-contact-detection algorithm that uses facial images taken with a camera embedded in a robot's eyes [12] to develop robot-assisted ASD-diagnosis systems.

In recent years, deep-learning-based approaches are being implemented for eye-contact detection. Mitsuzumi et al. developed the DNN-based eye contact deteciton algorithm (DeepEC)[13] that uses only cropped eye regions for eye-contact detection and performed better than existing methods. Eunji et al. develop the DNN-based PiCNN detector that accepts the facial region and output both facial postures and eye contact

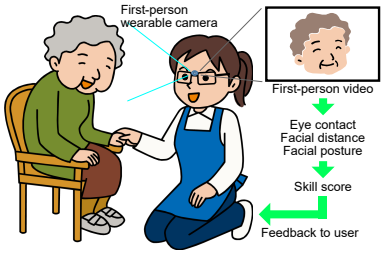



Figure 2. Conceptual illustration of our care-skill evaluation system. A caregiver wears a camera system and obtain FPV while care is given. From the video, the number of eye contact events, facial distances and facial postures between the caregiver and care receiver are obtained and used to evaluate the caregiver's care skills. The evaluation is given to the caregiver as feedback.

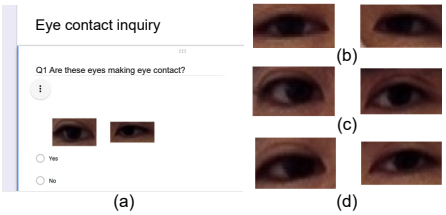



Figure 3. Preliminary experiment to examine eye-contact-detection performance of humans and DNNs. (a) 13 subjects answered a web form that asked whether cropped eye images taken from facial images showed eye contact or not. The rate of correct answers for human respondents was only 74% on average and 80% maximum, though the DNN (DeepEC) was correct 86% of the time. (b) - (d) Examples of positive (eye contact) images. The average correct answer rates for human respondents were (b) 46%, (c) 15% and (d) 7%.

states [14]. Zhang et al. presented an eye-contact detection algorithm based on their deep neural network (DNN) based gaze estimations [15]. This method detects and collects the facial region and gaze direction of other subjects from a FPV and finds eye contact by clustering the gaze directions. The eye contact detecter is construbted by using the clustered eye images.

However, most of the previous methods did not sufficiently consider temporal features of eye contact. Eye contact is a human communication behaviour, so temporal inference is important for increasing recognition accuracy. The pose-dependent eye contact (PEEC) algorithm [10, 11] used conditional random fields (CRF) for temporal inference; specifically, it applied the results of single-frame eye-contact detection, which consist of random forests, to CRF and obtained the results. However, the results of single-frame eye-contact detection are binary outputs (eye contact or aversion); thus, this algorithm could not consider the temporal correlation of eye images or facial positions for eye contact detection.

The limitations of single-frame recognition are also suggested by our preliminary experiment, in which we presented 50 cropped eye images to 13 subjects and asked whether the images showed eye contact or aversion (Figure 3). Surprisingly, the rate of correct answers for human respondents was only 74% on average and 80% maximum, while that of the DNN (DeepEC) was 86%. This finding supports the idea that we recognize eye contact not only by single eye-image frames, but also by temporal information, such as temporal eye and facial images and body posture.

To prove this idea, we developed several temporal DNN architectures that use long short-term memory (LSTM) and two eye-contact video datasets. The first dataset was obtained from YouTube videos in which a subject is talking to a camera, and the second dataset was the FPVs we obtained with assumed conversational scenarios. For each video, we manually annotated the eye-contact state frame by frame. The results indicate that out algorithm, which combines a single-frame eye contact detection algorithm (DeepEC) with temporal inference (LSTM), performs better than existing methods. Our main contributions are as follows.

**1.** We developed an eye-contact-detection algorithm that considers temporal information. We implemented several combinations of network architectures as well as temporal durations and found the best solution, which combines the CNN (DeepEC) and the cell state of LSTM with fully connected layers at the end of the network.

**2.** We prepared two eye-contact facial-image datasets. One is based on publicly available videos, and the other is a set of FPVs that assume conversational scenarios. We annotated the eye-contact states frame by frame and published the annotation results.

**3.** The results show that temporal inference improves detection performance, particularity for the recall performance. Also, we found proposed algorithm that combines CNN+LSTM is far better than CNN+CRF.

We describe the current and proposed algorithms in Section 2. We introduce the two datasets in Section 3, followed by the experimental results, discussion and conclusion in Sections 4 - 5.

## 2 Algorithms

We implemented our two algorithms that use a LSTM as well as DeepEC and compared their performance, as illustrated in Figure 4.

### 2.1 Single-frame eye-contact-detection algorithm: DeepEC

The DeepEC algorithm [13] (Figure 4(a)) first detects both eye regions in the target image frame us-

ing a facial-parts detector. Then, the resulting pair of eye images are separately input to two streams of seven-layer CNNs followed by two fully connected layers. The DeepEC has two variations: Naïve DeepEC, which uses only eye images, and DeepEC-HP, which uses the 3D facial position as well as eye images. In their original publication[13], Naïve DeepEC produced better results than DeepEC-HP, with Naïve DeepEC performing about 0.76 and 0.80 in precision and $F_1$ score, respectively, using publicly available facial-image datasets.

In addition, we implemented an extension of DeepEC named DEEPEC+CRF which can conduct tempral eye-contact detection by using Conditional Random Field(CRF) in order to compare with our proposal algorithm in point of temporal learning. The CRF inputs sequence of binary outputs of DeepEC and learns its temporal dependency.

## 2.2 Spatio-temporal eye-contact-detection algorithms: TempEC and TempEC-HP

We implemented eye-contact-detection algorithms that use spatio-temporal images of eyes by combining CNNs and LSTM. Our algorithms use a series of eye images obtained from continuous image frames of video datasets as input, in contrast to DeepEC, which uses only single eye images.

We developed and evaluated two architectures: TempEC, which uses only eye images, and TempEC-HP, which uses the 3D facial pose as well as eye images, as illustrated in Figure 4(b). These algorithms consist of the following components.

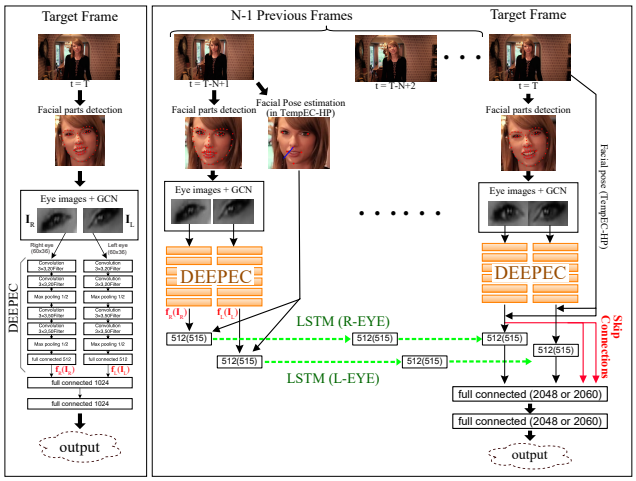
### 2.2.1 Eye region detection

To obtain eye images, we first obtain facial landmarks with face detection and shape prediction. In our implementation, we use the dlib face detector [16] and obtain 68 facial-landmark points.

Using these landmarks, we obtain the right and left eye regions in the target frame, from which we obtain each eye image used as input for the CNN, after grey-scaling and normalizing with global contrast normalization (GCN). Based on the landmarks, we obtain the coordinates of four corner points which determine eye region, At this time, we apply 10% margin to height and width of the region in order to accept the error of facial-landmark detection.

### 2.2.2 Head pose estimation (TempEC-HP)

TempEC-HP uses the 3D head position, which is computed from facial landmark points. Our position estimation is based on the EPnP algorithm [17] that transfers a set of 2D points to the 3D points. We choose six points (the tip of the nose, the chin, the left corner of the left eye, the right corner of the right eye and the left and right corners of the mouth) and obtain 3D rotation parameters of the head. In TempEC-HP, these three parameters are added to the network input with the facial image features.




(a) Singleframe Eye Contact detection(DEEPEC) (b) Proposed algorithm (TempEC, TempEC-HP)

Figure 4. Eye contact detection algorithms. (a) The single-frame eye-contact detection (DeepEC) first detects the eye regions of the target image frame and obtains a pair of right and left eye images. The eye-contact state is obtained by only the CNN that inputs the eye images. (b) Proposed temporal eye-contact-detection algorithms that uses multiple (i.e. $N$) image frames. First, it detects facial landmarks with the dlib face detector, with which it then obtains eye regions in each of the $N$ frames. The resulting $N$ pairs of eye images are inputted to CNNs that have a similar structure of DeepEC. These CNNs are followed by a LSTM network, which learns the temporal state of the eyes. Finally, the target eye-contact state is obtained by the following fully connected networks, which use not only the LSTM's outputs but also the CNN's outputs of the target frame ($t = T$) with skip connections.

### 2.2.3 Deep temporal eye-contact detector

Given the images of both eye regions and the 3D facial pose, we implemented our two deep temporal eye-contact-detection algorithms, as shown in Figure 4(b). The algorithms use ten continuous video frames – the target frame and nine preceding frames – for predictions. The number of input frames (10 frames) was experimentally determined according to the results of a preliminary experiment.

As shown in Figure 4(b), each of these pairs of eye images $\mathbf{I}^t_{\mathrm{R}}$, $\mathbf{I}^t_{\mathrm{L}}$ are respectively input to the CNN. This CNN has the same structure as DeepEC with the exception of the last two fully connected layers; namely, it has two streams and six layers consisting of a pair of two convolution layers followed by max pooling layers. The CNN outputs a pair of 512-dimensional feature vectors of each eye image $\mathbf{f}_{\mathrm{R}}(\mathbf{I}^t_{\mathrm{R}})$ and $\mathbf{f}_{\mathrm{L}}(\mathbf{I}^t_{\mathrm{L}})$.

These feature vectors are input to two separate LSTM networks for the left and right eye images. In the TempEC algorithm, each LSTM accepts 10 vectors corresponding to a series of eye images and outputs one 512-dimensional feature vector. In the TempEC-HP algorithm, a series of 3D vectors that represent 3D head positions are additionally input to LSTM.

However, we found Naïve LSTM could not perform satisfactorily. To solve this problem, we prepared the fully connected layers, which have 2048 (512×4) units at the last frame that accept the outputs of the left and right DeepEC's and LSTM's cell state vectors. Because the result of the DeepEC of the current frames is directly used for eye-contact detection, and the temporal inference is also merged to the fully connected layers, we can ultimately obtain better results than the Naïve implementations of DeepEC and LSTM.

## 3 Datasets

To train and evaluate the proposed algorithms, we prepared eye-contact video datasets based on publicly available videos from YouTube and on our original FPV videos that assume conversational scenarios.

### 3.1 Publicly available videos from YouTube

We used 13 videos in which a person talks to a camera. For each frame in the videos, two people annotated the eye-contact state. We took a consensus of the annotations and made ground-truth data. A list of the videos and their properties is shown in Table 1 and Figure 5.

### 3.2 First person eye contact video dataset

Assuming *in-the-wild* applications, we additionally prepared first-person-view videos containing conversational scenarios. Conversational scenarios were taken in a lab environment in which two participants were talking. One participant wore a Pivothead Kudu first-person camera [18], which consists of a frontal-view camera in the middle of a pair of eye glasses and takes full HD (1920 × 1080 pixels) video at 30 fps. A list of the videos and their properties is shown in Table 4 and Figure 5. We took three video clips from six participants and two test-video clips from two participants.


Figure 5. Eye-contact dataset using publicly available videos from YouTube or first-person camera (names with *).

## 4 Experiments

We conducted an experiment to compare the performances of the proposed and an existing algorithm using the datasets. One video was chosen for testing and the others were used for learning. We iterated this step for the 16 videos and obtained the average performance.

The learning of the networks with DeepEC, TempEC and TempEC-HP was conducted as follows. We first computed the bounding rectangles of eyes using the facial-landmark points obtained by dlib. The obtained eye images were then rescaled such that the image was (60 × 36) pixels. We used static CNN hyper-parameters for all of the experiments. Specifically, the drop-out rate was 0.5, and Leaky ReLU activation function's $\alpha$ was set to 0.01. We used Adam optimizer [19] with the learning rate set to 0.001, the decay as 0.004 per epoch and $\beta_1$ and $\beta_2$ as 0.9 and 0.999, respectively.

The results are given in Table 1 and illustrated in Figure 6. The results show that our algorithms performed the best. Namely, the TempEC algorithm thoroughly outperformed DeepEC in precision = 0.8561, recall = 0.8544 and $F_1$ score = 0.8706. TempEC-HP outperforms to TempEC, DeepEC and DeepEC+CRF in recall and $F_1$-score, with a recall of 0.9248 and $F_1$ score of 0.8781 on average. Figure 6 shows the area under the curve (AUC) of our algorithm is larger than that of DeepEC. TempEC-HP had the best AUC (0.870) followed by TempEC (AUC = 0.85). Regarding the accuracy, TempEC-HP achieved a 25% improvement in miss-detection rate, with 0.1751 in comparison to DeepEC's 0.2330.

The more detailed results are in Table 4. Each row corresponds to one video of datasets. Three from the bottom which marked with * are first person view videos and the others are obtained from YouTube.

Table 1. Test results with all videos of dataset.

| | Precision | Recall | $F_1$ score |
|---|---|---|---|
| DEEPEC | 0.8512 | 0.7808 | 0.8319 |
| DEEPEC+CRF | 0.7693 | 0.8339 | 0.7876 |
| TempEC | **0.8561** | 0.8544 | 0.8706 |
| TempEC-HP | 0.8364 | **0.9248** | **0.8781** |

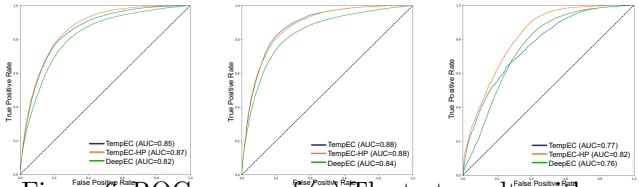



Figure 6. ROC curves of (a) The test results with all dataset, (b) The test results with videos from YouTube, (c) The test results with first person view videos.

## 5 Discussion and Conclusion

We developed an eye-contact-detection algorithm that uses temporal features as well as static image features. Our algorithm shows better performance for various types of dataset. It combines CNNs and LSTM and successfully learned both stational features and temporal dependence. In the experiments, the proposed TempEC and TempEC-HP outperformed DeepEC, especially TempEC-HP, which achieved a 25% improvement relative to existing algorithms in the miss-detection rate.

In our preliminary experiment, a simple concatenation of CNNs and LSTM was not effective. We concluded that such a primitive combination was not suitable to learn both static and temporal features at the same time. Thus, we introduced a skip connection in the final step of estimation that jumps over the LSTM networks and directly links the CNN outputs to the final fully connected layers. Adopting this structure, our algorithm's performance improved, as shown in Section 4. These results show that the skip connection enables the algorithm to successfully learn both static and temporal features at the same time.

Surprisingly, in the comparison of TempEC and TempEC-HP, some tests showed that TempEC performed better than TempEC-HP, despite our expectation that TempEC-HP would completely outperform TempEC because the facial pose information would help in detecting eye contact with various face directions. However, these results do not indicate that facial pose information is useless. In our algorithm, 3D facial-pose estimation is based on facial-landmarks, of which detection is mostly accurate but has a certain degree of error. This error is not significant, which is why it is not a problem when used to obtain eye region, but in facial-pose estimation, such a small error sometimes causes a large incorrect gap between two contiguous frames, as shown in Figure 7. The facial pose of two adjacent frames should be close because a human's face cannot move a large amount in a short time (namely, 0.03 sec because this video was recorded in 30 fps). Due to this problem, facial-pose estimation is occasionally not sufficiently reliable, which causes TempEC-HP to perform poorly. Hence, the performance of TempEC-HP can be improved by using a more accurate facial-detection or facial-pose estimation algorithm.

Figure 7. Example of failed head-position estimation. The above two faces are clipped from two adjacent movie frames. The right one is just 0.03 seconds after the left one. Despite the two head positions appearing to be almost the same, the algorithm's estimation shows extremely different vectors.

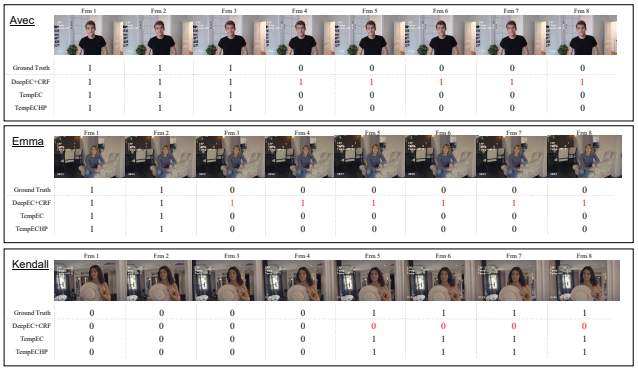



Figure 8. Several examples showing the differences of DeepEC+CRF, TempEC and TempEC-HP. CRF tends to 'smoothen' the temporal inference while other two correctly estimate the state change.

Another notable finding was that introducing temporal inference increased the performance in recall, which means the temporal information contributed to 'overlooked' effects of eye contact. This phenomenon is also appears to be indicated in our preliminary experiments (Figure 3), which show that we cannot distinguish the eye-contact state from eye images alone but can detect it from videos. In our experiment, CRF could not improve the result of DeepEC. As seen in the several examples (Figure 8), CRF tends to smoothen the result of DeepEC, which may contribute to avoid

| Name | Total frames | Face detected | Eye contacted | DeepEC | | | DeepEC+CRF | | | TempEC | | | TempEC-HP | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Precision | Recall | $F_1$ | Precision | Recall | $F_1$ | Precision | Recall | $F_1$ | Precision | Recall | $F_1$ |
| Avec | 10,498 | 8,940 | 4,671 | **0.8878** | 0.8262 | 0.8559 | 0.7881 | 0.8584 | 0.8217 | 0.8764 | 0.888 | 0.8822 | 0.8833 | **0.9241** | **0.9032** |
| Aziz | 11,508 | 6,711 | 5,058 | 0.9386 | 0.8695 | 0.9027 | 0.8838 | 0.9077 | 0.8956 | 0.935 | 0.8944 | 0.9142 | **0.9471** | **0.9636** | **0.9553** |
| Derek | 11,765 | 4,550 | 3,710 | 0.8865 | 0.8758 | 0.8811 | 0.8495 | 0.9292 | 0.8876 | **0.9244** | 0.8924 | 0.9081 | 0.8761 | **0.9478** | **0.9106** |
| Elle | 11,739 | 6,458 | 4,005 | **0.897** | 0.7371 | 0.8092 | 0.8522 | 0.7773 | 0.813 | 0.8479 | 0.9079 | 0.8769 | 0.8556 | **0.9486** | **0.8997** |
| Emma | 13,325 | 6,302 | 3,878 | 0.8196 | 0.8278 | 0.8237 | 0.752 | **0.8821** | 0.8119 | 0.8334 | 0.8701 | **0.8513** | **0.8441** | 0.7273 | 0.7814 |
| Wataru | 1,675 | 1,306 | 1,049 | **0.8785** | 0.9162 | 0.897 | 0.836 | 0.9615 | 0.8944 | 0.8492 | 0.9762 | 0.9083 | 0.8307 | **0.9964** | **0.906** |
| James | 9,125 | 3,270 | 2,110 | 0.8957 | 0.423 | 0.5746 | 0.8306 | 0.4803 | 0.6086 | 0.8839 | **0.9102** | **0.8968** | **0.9175** | 0.8197 | 0.8658 |
| Kendall | 10,735 | 4,331 | 3,094 | 0.8772 | 0.8423 | 0.8594 | 0.8102 | 0.894 | 0.85 | **0.8836** | **0.917** | **0.9** | 0.8654 | 0.9139 | 0.889 |
| Liza | 10,739 | 7,440 | 6,097 | 0.9313 | 0.8562 | 0.8922 | 0.8838 | 0.9156 | 0.8944 | **0.9539** | 0.894 | 0.923 | 0.9359 | **0.9144** | **0.925** |
| Neil | 16,487 | 8,904 | 5,677 | 0.8404 | 0.8894 | 0.8642 | 0.4223 | 0.9031 | 0.5755 | **0.8496** | 0.9729 | **0.9071** | 0.7952 | **0.9753** | 0.8761 |
| Selena | 11,043 | 6,052 | 3,634 | **0.8322** | 0.7179 | 0.7709 | 0.7767 | 0.7887 | 0.7826 | 0.8018 | 0.7517 | 0.776 | 0.7736 | **0.9282** | **0.8439** |
| Mai | 9,840 | 2,565 | 1,330 | **0.7349** | 0.7762 | 0.755 | 0.698 | 0.8231 | 0.7554 | 0.7022 | 0.9459 | **0.806** | 0.6315 | **0.988** | 0.7705 |
| Taylor | 13,945 | 9,444 | 5,489 | 0.8085 | 0.6674 | 0.7312 | 0.7473 | 0.7308 | 0.7389 | **0.8889** | 0.8028 | 0.8437 | 0.8141 | **0.9571** | **0.8798** |
| Imaizumi* | 6,594 | 2,343 | 1,664 | 0.565 | 0.8873 | 0.6904 | 0.5165 | **0.9543** | 0.6702 | 0.6188 | 0.9148 | 0.7382 | **0.6333** | 0.9312 | **0.7539** |
| Kitazumi* | 21,336 | 20,364 | 18,560 | 0.9024 | 0.6593 | 0.762 | 0.7993 | 0.7561 | 0.7771 | **0.9473** | 0.3612 | 0.523 | 0.9077 | **0.8818** | **0.8945** |
| Ogawa* | 37,917 | 29,220 | 28,032 | **0.9242** | 0.7207 | 0.8098 | 0.8622 | 0.7801 | 0.8191 | 0.9018 | 0.771 | 0.8313 | 0.8714 | **0.979** | **0.9221** |
| Total | 208,271 | 128,200 | 98,058 | 0.8512 | 0.7808 | 0.8319 | 0.7693 | 0.8339 | 0.7876 | **0.8561** | 0.8544 | 0.8706 | 0.8364 | **0.9248** | **0.8781** |

Table 2. Experimental results for each video of dataset.

the 'jittering' effects of single frame estimations but does not solve the temporal inference problem fundamentally. Thus, we think our current algorithm that combines the internal states of single frame recognition and LSTM is better solution.

Our results show great performance in eye-contact detection, and further, they show the potential of temporal learning of eye behaviour, with which we can evaluate the care skills of caregivers and find eye movements peculiar to ASDs. Outside of the medical field, analysis of temporal eye behaviour can enable the production of effective advertisements and arrangements of items.

## 6 Acknowledgements

This work is supported by Grants-in-Aid for Scientific Research 17H01779, 26249029, 15H02738 and JST, CREST, JPMJCR17A5.

## References

[1] Senju, A., Johnson, M.H.: The eye contact effect: mechanisms and development. Trends in Cognitive Sciences **13** (2009) 127–134

[2] Adams Jr, R.B., Kleck, R.E.: Effects of direct and averted gaze on the perception of facially communicated emotion. Emotion **5** (2005) 3

[3] Csibra, G., Gergely, G.: Social learning and social cognition: The case for pedagogy. Processes of change in brain and cognitive development. Attention and performance XXI **21** (2006) 249–274

[4] Farroni, T., Csibra, G., Simion, F., Johnson, M.H.: Eye contact detection in humans from birth. Proceedings of the National Academy of Sciences **99** (2002) 9602–9605

[5] Joseph, R.M., Ehrman, K., McNally, R., Keehn, B.: Affective response to eye contact and face recognition ability in children with asd. Journal of the International Neuropsychological Society **14** (2008) 947–955

[6] Society, A.: Factsheet: Communicating (2016) [Online; accessed 18-Nov-2016].

[7] Gineste, Y., Pellissier, J.: Humanitude: comprendre la vieillesse, prendre soin des hommes vieux. A. Colin (2007)

[8] group, A.: Title. (2016)

[9] Smith, B.A., Yin, Q., Feiner, S.K., Nayar, S.K.: Gaze locking: passive eye contact detection for human-object interaction. In: Proceedings of the 26th annual ACM symposium on User interface software and technology, ACM (2013) 271–280

[10] Ye, Z., Li, Y., Fathi, A., Han, Y., Rozga, A., Abowd, G.D., Rehg, J.M.: Detecting eye contact using wearable eye-tracking glasses. In: Proceedings of the 2012 ACM Conference on Ubiquitous Computing, ACM (2012) 699–704

[11] Ye, Z., Li, Y., Liu, Y., Bridges, C., Rozga, A., Rehg, J.M.: Detecting bids for eye contact using a wearable camera. In: Automatic Face and Gesture Recognition (FG), 2015 11th IEEE International Conference and Workshops on. Volume 1., IEEE (2015) 1–8

[12] Petric, F., Miklić, D., Kovačić, Z.: Probabilistic eye contact detection for the robot-assisted asd diagnostic protocol. In Lončarić, S., Cupec, R., eds.: Proceedings of the Croatian Compter Vision Workshop, Year 4, Osijek, Center of Excellence for Computer Vision, University of Zagreb (2016) 3–8

[13] Mitsuzumi, Y., Nakazawa, A., Nishida, T.: Deep eye contact detector: Robust eye contact bid detection using convolutional neural network. In: Proceedings of the British Machine Vision Conference (BMVC). (2017)

[14] Chong, E., Chanda, K., Ye, Z., Southerland, A., Ruiz, N., Jones, R.M., Rozga, A., Rehg, J.M.: Detecting gaze towards eyes in natural social interactions and its use in child assessment. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies **1** (2017) 43

[15] Zhang, X., Sugano, Y., Bulling, A.: Everyday eye contact detection using unsupervised gaze target discovery. In: Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology, ACM (2017) 193–203

[16] King, D.E.: Dlib-ml: A machine learning toolkit. Journal of Machine Learning Research **10** (2009) 1755–1758

[17] Lepetit, V., Moreno-Noguer, F., Fua, P.: Epnp: An accurate o (n) solution to the pnp problem. International journal of computer vision **81** (2009) 155
[18] Povithead: Pivothead KUDU (2016) [Online; accessed 29-Aug-2016].
[19] Dozat, T.: Incorporating nesterov momentum into adam. (2016)