Eye Contact Detection via Deep Neural Networks

Viral Parekh, Ramanathan Subramanian(✉), and C.V. Jawahar

International Institute of Information Technology, Hyderabad, India
parekh.viral@research.iiit.ac.in, {s.ramanathan,jawahar}@iiit.ac.in

Abstract. With the presence of ubiquitous devices in our daily lives, effectively capturing and managing ***user attention*** becomes a critical device requirement. While gaze-tracking is typically employed to determine the user's focus of attention, *gaze-lock* detection to sense eye-contact with a device is proposed in [16]. This work proposes eye contact detection using ***deep neural networks***, and makes the following contributions: (1) With a convolutional neural network (CNN) architecture, we achieve superior eye-contact detection performance as compared to [16] with *minimal data pre-processing*; our algorithm is furthermore validated on multiple datasets, (2) Gaze-lock detection is improved by combining head pose and eye-gaze information consistent with social attention literature, and (3) We demonstrate gaze-locking on an Android mobile platform via CNN model compression.

Keywords: Eye contact detection · Human-Computer Interaction · Convolutional neural networks

1 Introduction

The importance of ***eye-contact*** in non-verbal human communication cannot be understated. Right from infanthood, humans use eye-contact as a means for attracting and acknowledging attention, and can effortlessly sense others' eye-gaze direction [5]. In today's ubiquitous computing environment, it becomes critical for devices to effectively attract and manage users' attention for proactive communication and information rendering. Therefore, HCI would greatly benefit from devices that can sense user attention via eye-contact– a phenomenon termed ***gaze locking*** in [16].

Gaze locking is a sub-problem of *gaze-tracking*, where the objective is to determine where the user is looking. Gaze tracking has been extensively studied by the HCI [11,12], psychology [14,20], medical [7] and the multimedia/computer vision communities [10,19]. Gaze-tracking techniques (with the exception of few such as [8]) have inferred the point-of-gaze using eye-based cues even though social attention literature has identified that other cues such as head orientation contribute significantly to this end [9].

C. Stephanidis (Ed.): HCII Posters 2017, Part I, CCIS 713, pp. 366–374, 2017.
DOI: 10.1007/978-3-319-58750-9_51

This paper proposes gaze-locking using deep convolutional neural networks (CNNs), which have recently become popular for solving visual recognition problems as they obviate the need for hand-crafted features (*e.g.*, expressly modeling head pose). Specifically, our work makes the following research contributions:

(1) Even though the gaze-locking methodology outlined in [16] detects eye-contact from distant faces, it requires an elaborate processing pipeline which includes: eye region rectification for head pose compensation, eye mask extraction, compression of a high-dimensional eye appearance feature vector via dimensionality reduction and a classifier for gaze-lock detection. Differently, we leverage the learning power of CNNs for gaze-locking with minimal data pre-processing. We validate our model on three datasets, and obtain over 90% detection accuracy on the Columbia Gaze (CG) [16] test set. In comparison, [16] reports 92% accuracy on the CG *training set.*

(2) Different from [16] and most gaze-tracking methods, we use facial appearance, which implicitly conveys face pose, in addition to eye appearance. As seen in Fig. 1, face orientation crucially determines if the user is gaze-locked with a (reference) camera or not. The eyes in the left and right images have very similar appearance; however, eye-contact is clearly made only in the right instance when one infers gazing direction as the eye orientation *relative* to head pose. Combining face and eye cues achieves superior gaze locking than either of the two as demonstrated in prior works [17].

(3) CNNs are implemented on CPU/GPU clusters given their huge computation and memory requirements; their implementation on mobile platforms is precluded by the limited by the computation and energy resources in these environments. We demonstrate gaze-locking on an Android mobile platform via CNN compression using ideas from the *dark knowledge* concept [6].

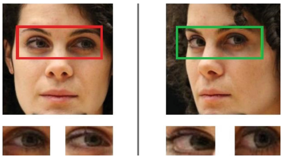

Fig. 1. Left image is ***non-gaze-locked***, while right image is ***gaze-locked***. Their eye crops however look very similar.

2 Methodology

Figure 2 presents our proposed system and the convolutional neural network (CNN) architecture. CNNs *automatically learn* problem-specific features, obviating the need for devising hand-crafted descriptors like HoG [3]. Furthermore, replacing the largely independent *feature extraction* and *feature learning* modules by an end-to-end framework allows for efficient handling of classification errors. System components are described below.

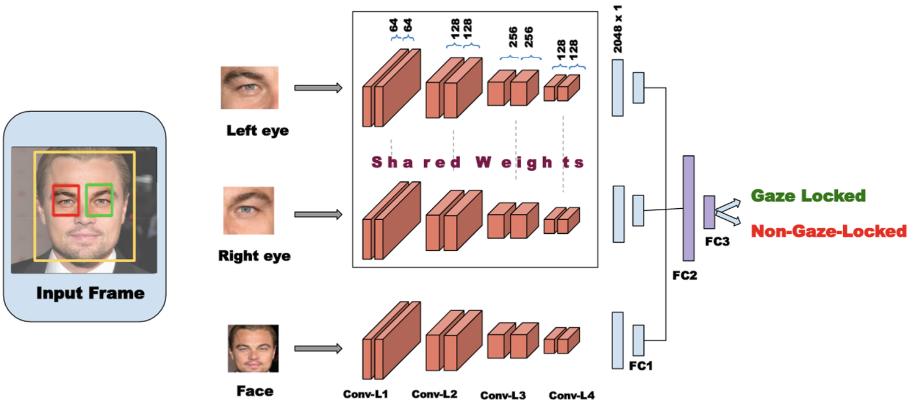

Fig. 2. Overview of our ***gaze-lock detector***. Inputs include 64 × 64 *left eye*, *right eye* and *face images*, and the detector outputs a binary label assigned as either *gaze-locked* or *non-gaze-locked*. CNN architecture has ***three*** parallel networks each comprising ***four*** convolutional layer blocks (denoted as filter size/number of filters): CONV-L1: 3 × 3/64, CONV-L2: 3 × 3/128, CONV-L3: 3 × 3/256, and CONV-L4: 3 × 3/128, and ***three*** fully-connected layers denoted as FC1 (of size 2048 inputs × 128 outputs), FC2: 384 × 128 and FC3: 128 × 2. (Color figure online)

2.1 Image Pre-processing

We essentially use the face and eye appearance to detect eye-contact, and pre-processing is limited to extraction of these regions. A state-of-the art facial landmark detector [1] is used to obtain 64 × 64 left and right eye patches. Since face pose serves as an additional cue, a 64 × 64 face patch obtained using the Viola-Jones detector [18] is also fed to the CNN. The red, green and blue channels for each patch are z-normalized prior to input.

2.2 CNN Architecture

Our system comprises three parallel networks (one each for face, left eye and right eye) with a VGGnet [15]-like configuration. CNNs are stacked with *convolutional* (Conv) layers composed of groups of *neurons* (or filters), which automatically compute locally salient features (or activations) from input data. Conv layers are interleaved with *max pooling* layers, which isolate the main activations on small data blocks and allow later layers to work on a 'zoomed out' version of previous outputs facilitating parameter reduction. Convolutions are also usually followed by a non-linear operation (called *rectified linear unit* or ReLU [13]) to make the CNN more expressive and powerful. Finally, in a *fully-connected* (FC) layer, neurons have access to *all* activations from a previous layer as against a Conv layer whose neurons only access local activations.

Each of our three networks have four blocks, with each block including two Conv layers, a ReLU and a max-pooling layer (only Conv layers are shown in Fig. 2). Similar activations are enforced for the left and right eye networks by constraining their neurons to learn identical/shared weights. The *filter size* or spatial extent of activations input to a Conv layer neuron is 3 × 3 for all blocks, and there are 64, 128, 256 and 128 neurons respectively in the four blocks. A stride length of 1 is used while convolving (computing dot product of) the filter with the input patches. The Conv-L4 outputs are vectorized to a 2048 dimensional vector, which is input to the FC1 layer with 128 outputs. FC1 outputs from the three networks are combined and fed to FC2 followed by FC3, which assigns the input label as either *gaze-locked* or *non-gaze-locked*. The CNN model was implemented on *Torch* [2], and trained over 250 epochs with a batch size of 100. An initial learning rate of 0.001 was reduced by 5.0% after every epoch. To avoid overfitting, a dropout technique was used to randomly remove 40% of the FC layer neurons during training. Interested readers may refer to [15] for further details.

3 Experiments and Results

3.1 Datasets

To expressly address eye-contact detection, authors of [16] compiled the **Columbia Gaze** (CG) dataset which comprises 5880 images of 56 persons viewing over 21 different gaze directions and 5 different head poses. Of these, 280 are *gaze-locked*, while 5600 are *non-gaze-locked*– sample CG images are shown in Fig. 3 (left). The CG dataset is compiled in a controlled environment, and contains little variation in terms of illumination and background. The limited size of the CG dataset makes it unsuitable for training CNNs, and we therefore used two large datasets to train our CNN, namely, (1) **MPIIGaze** [21] comprising 213,659 images compiled from 15 subjects during everyday laptop use. As shown in Fig. 3 (center-top), MPIIGaze images vary with respect to illumination, face size and background. However, only cropped eye images (center-bottom) are publicly available for MPIIGaze; (2) The **Eyediap** dataset [4] (Fig. 3 (right)) contains 19 HD videos with more than 3000 images each captured from 16 participants. We ignore the depth information available for this dataset, and only use the raw video frames for our purpose.

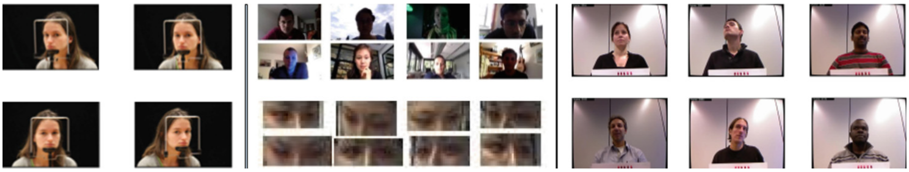

Fig. 3. (left) Sample images from the **CG** dataset. (center-top) Original exemplars and (center-bottom) publicly available eye-only images from **MPIIGaze**. (right) Sample images from **Eyediap**.

3.2 Data Synthesis and Labeling

As only 280 *gaze-locked* images exist in the CG dataset, we generated 2280 *gaze-locked* and 5900 *non-gaze-locked* samples by scaling and randomly perturbing original images as described in [16]. On the contrary, we downsampled the number of images for the MPIIGaze and Eyediap datasets. MPIIGaze comprises images with continuous gaze direction from 0° to $-20°$ *pitch* (vertical head rotation) and $-20°$ to 20° *yaw* (horizontal rotation). The 3D gaze direction (x, y, z) is converted to 2D angles (θ,ϕ) as $\theta = \arcsin(-y), \phi = \arctan(-x, -z)$. Then, *gaze-locking* implies $(\theta, \phi) = (0, 0)$. This way, we obtained 6892 *gaze-locked* and 12000 *non gaze-locked* images from MPIIGaze. Likewise, Eyediap images show users making eye-contact with various screen regions on a 24″ PC monitor. We labeled images with the target looking straight ahead (around screen center) as *gaze-locked*, and others as *non gaze-locked*. Table 1 presents the training and test sets statistics for the three datasets. We now discuss gaze-locking results with different train and test sets.

Table 1. Training and test set details for the various datasets.

Attribute	CG	MPIIGaze	EYEDIAP
Total images	5880	214076	125000
Synthesized	8180	18892	24575
Training set	7000	15000	19660
Test set	1180	3892	4915

Experiment 1 (Ex1). To begin with, we used only the CG dataset for model training[1]. Specifically, we trained our detector with (a) images of only *one eye*; (b) images from *both eyes*; (c) only *face* images, and (d) *face-plus-eye* images as in Fig. 2.

Experiment 2 (Ex2). Here, we repeated Ex1(a) and (b)[2], but first pre-trained the CNN with MPIIGaze and *fine-tuned* the same using CG. Fine-tuning involved modifying only the FC layer weights by re-training with CG images, assuming that the learned Conv-L4 activations were relevant for both MPIIGaze and CG.

Experiment 3 (Ex3). We repeated Ex1(a–d), but pre-trained the CNN with Eyediap followed by fine-tuning on CG.

[1] The CNN was trained and validated with a 80:20 split of the training set in all experiments.

[2] Since MPIIGaze does not contain face images, we could not repeat Ex1(c) and (d).

Table 2. Detection performance for Ex1(a)–3(d) and comparison with [16]. Model tested on **CG** in all cases. [16] reports results only on the training set.

	1(a)	1(b)	1(c)	1(d)	2(a)	2(b)	3(a)	3(b)	3(c)	3(d)	Smith *et al.* [16]
Acc (%)	70.8	70.6	68.4	64.4	86.1	90.8	85.5	90.2	88.4	**92.7**	92.00
MCC	0.69	0.72	0.67	0.36	0.74	0.81	0.74	0.80	0.78	**0.83**	0.83

Experiment 4 (Ex4). To examine the effect of our framework on datasets other than CG, we repeated Ex1(a–d) with a CNN trained on CG and fine-tuned with Eyediap.

3.3 Results and Discussion

Gaze-locking results are tabulated in Tables 2 and 3. Detection performance is evaluated in terms of accuracy, and the Mathews correlation coefficent (MCC). MCC is useful while evaluating binary classifier performance on unbalanced datsets, as with our case where the number of *gaze-locked* instances are far less than *non-gaze-locked* ones. In Ex1, accuracy and MCC decrease as more information is input to the CNN (*e.g.*, face = plus-eyes vs eyes/face only), contrary to our expectation. This reduction is attributable to *overfitting* due to the small CG dataset size in comparison to the number of CNN parameters.

However, the benefit of using additional information for gaze-lock detection is evident from Ex2, Ex3 and Ex4 (Ex2 and Ex3 involve pre-training of the CNN model with larger and visually richer datasets). Using *two-eye* information as against *one-eye* in Ex2 improves accuracy and MCC by 4.7 and 7% respectively. Ex3 and Ex4 results are consistent with social attention literature. They confirm that while gaze direction is more critical than head pose for inferring eye contact, combining head and eye orientation cues is optimal for gaze-locking. Our system achieves a best accuracy of 93% and MCC of 0.83 on the CG dataset. Table 2 also compares our results with the state-of-the-art [16]. [16] reports detection results on the *training set*, while our results are achieved on an independent test set. With minimal data pre-processing, our model performs similar to [16] using only eye appearance, and outperforms [16] with face-plus-eye information. Finally, while the results for Ex4 again confirm the insufficiency of the CG dataset for training the CNN, the gaze-locking performance significantly improves on incorporating facial and binocular information.

Table 3. Detection results for Ex4. Model trained on **CG** and fine-tuned/tested on **Eyediap**.

Input	One eye	Both eyes	Face only	Face & eyes
Acc	62.9	65.6	64.5	**66.9**
MCC	0.57	0.58	0.57	**0.61**

3.4 Visualizing CNN Activations

Figure 4 illustrates four neuronal activations learned in the Conv-L1 layer of our CNN model for the input eye and face images. Conv-L1 activations are informative as ReLU network activations are dense in the early layers, and progressively become sparse and localized. As eye gaze direction is given by the pupil orientation, the eye activations capture edges and textures relating to the pupil. Similarly, the face network activations encode face shape and structural details for pose inference.

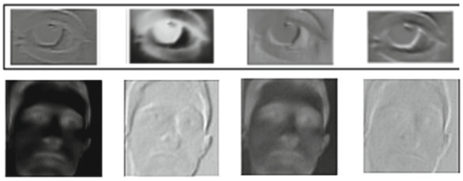

Fig. 4. Exemplar Conv-L1 neuron outputs for input eye (top) and face (bottom) images.

4 CNN Implementation on Android

While our CNN based gaze-lock detector requires minimal pre-processing, the end-to-end framework obviates need for heuristics as with the eye mask extraction phase in [16]. Our system achieves 15 fps throughput on an Intel Core I7 2.6 GHz, 16 GB RAM PC with GeForce GTX 960M GPU. However, CNNs require large computational and memory resources which precludes their implementation on mobile devices with limited computation and energy capacity.

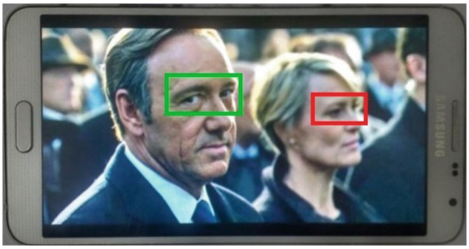

Fig. 5. Compressed version of our model working on an Android (Quad-core, 2.3 GHz, 3GB RAM) phone. Green rectangle denotes gaze-locking, while red denotes non-gaze-locking. (Color figure online)

This problem can be circumvented by compressing knowledge in a large, complex model to train a simpler model with minimal accuracy loss using the *dark-knowledge* concept [6]. Figure 5 shows our gaze-lock detector on an Android platform, which has a throughput of 1 fps. A more efficient implementation described in [8] can achieve upto 15 fps throughput.

5 Conclusion

This work exploits the power of deep CNNs to perform passive eye-contact detection with minimal data pre-processing. Combining facial appearance with eye information improves gaze-locking performance. Our system can also run on an Android mobile device with limited throughput. Our end-to-end system with minimal heuristics can be leveraged by today's smart devices for capturing and managing user attention (*e.g.*, a *smart selfie* application), as well as in image/video retrieval (detecting shots where a certain character is facing the camera). Future work involves implementation of a seamless, real-time vision-voice system for assistive applications such as photo-capturing for the blind.

References

1. Baltrusaitis, T., Robinson, P., Morency, L.P.: Constrained local neural fields for robust facial landmark detection in the wild. In: International Conference on Computer Vision Workshops, pp. 354–361 (2013)
2. Collobert, R., Kavukcuoglu, K., Farabet, C.: Torch7: a matlab-like environment for machine learning. In: BigLearn, NIPS Workshop (2011)
3. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR, pp. 886–893. IEEE Computer Society, Washington, DC (2005)
4. Funes Mora, K.A., Monay, F., Odobez, J.M.: EYEDIAP: a database for the development and evaluation of gaze estimation algorithms from RGB and RGB-D cameras. In: Eye Tracking Research and Applications, pp. 255–258. ACM, New York (2014)
5. Hains, S.M., Muir, D.W.: Infant sensitivity to adult eye direction. Child Dev. **67**, 1940–1951 (1996)
6. Hinton, G., Vinyals, O., Dean, J.: Distilling the Knowledge in a Neural Network. CoRR, March 2015
7. Holzman, P.S., Proctor, L.R., Levy, D.L., Yasillo, N.J., Meltzer, H.Y., Hurt, S.W.: Eye-tracking dysfunctions in schizophrenic patients and their relatives. Arch. Gen. Psychiatry **31**(2), 143–151 (1974)
8. Krafka, K., Khosla, A., Kellnhofer, P., Kannan, H., Bhandarkar, S., Matusik, W., Torralba, A.: Eye tracking for everyone. In: CVPR (2016)
9. Langton, S.R.: Do the eyes have it? Cues to the direction of social attention. Trends Cogn. Sci. **4**(2), 50–59 (2000)
10. Li, R., Shi, P., Haake, A.R.: Image understanding from experts' eyes by modeling perceptual skill of diagnostic reasoning processes. In: CVPR, pp. 2187–2194 (2013)
11. Majaranta, P., Bulling, A.: Eye tracking and eye-based human–computer interaction. In: Fairclough, S.H., Gilleade, K. (eds.) Advances in Physiological Computing. HIS, pp. 39–65. Springer, London (2014). doi:10.1007/978-1-4471-6392-3_3
12. Morimoto, C.H., Mimica, M.R.: Eye gaze tracking techniques for interactive applications. CVIU **98**(1), 4–24 (2005)

13. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: ICML, pp. 807–814 (2010)
14. Rayner, K.: Eye movements in reading and information processing: 20 years of research. Psychol. Bull. **124**, 372–422 (1998)
15. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556 (2014)
16. Smith, B.A., Yin, Q., Feiner, S.K., Nayar, S.K.: Gaze locking: passive eye contact detection for human-object interaction. In: User Interface Software and Technology, pp. 271–280. ACM (2013)
17. Subramanian, R., Staiano, J., Kalimeri, K., Sebe, N., Pianesi, F.: Putting the pieces together: multimodal analysis of social attention in meetings. In: ACM International Conference on Multimedia, pp. 659–662. ACM (2010)
18. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: CVPR, vol. 1, pp. 1–511. IEEE (2001)
19. Volokitin, A., Gygli, M., Boix, X.: Predicting when saliency maps are accurate and eye fixations consistent. In: CVPR, pp. 544–552 (2016)
20. Vrânceanu, R., Florea, C., Florea, L., Vertan, C.: NLP EAC recognition by component separation in the eye region. In: Wilson, R., Hancock, E., Bors, A., Smith, W. (eds.) CAIP 2013. LNCS, vol. 8048, pp. 225–232. Springer, Heidelberg (2013). doi:10.1007/978-3-642-40246-3_28
21. Zhang, X., Sugano, Y., Fritz, M., Bulling, A.: Appearance-based gaze estimation in the wild. In: CVPR, pp. 4511–4520. IEEE Computer Society (2015)