Interpreting iTracker

David Canagasabey[*1], **Halim Lee**[*2], **Sidharth Gupta**[*1]

Department of Computer Science[1], Department of Electrical & Computer Engineering[2]
{david.canagasabey, halim.lee, sid.gupta}@mail.utoronto.ca

Abstract

Eye-tracking technology provides accessibility to many kinds of people; such as individuals with motor control disabilities. The task is difficult, however, because a subtle mixture of head, pupil, and face movements can correspond to a large difference in screen coordinates, as shown in physiology research. Current state-of-the-art neural networks, such as iTracker, intake both zoomed-in pupil images and zoomed-out face images, and show exceptional empirical results with eyetracking. However, a concrete interpretibility study has yet to be done on iTracker, and it's unknown if the model actually accumulates small differences in the head, pupil, and face movements when making predictions. In this paper, we perform such a study by applying Deep Dream and SmoothGrad on the iTracker model. We isolate the zoomed-in pupil branches from the network, and compare their Deep Dream and SmoothGrad results with the whole iTracker model. Our results show that the zoomed-in branches alone require large differences to change screen positioning, whereas the whole iTracker model only requires small differences. Thus, we conclude that the whole iTracker model indeed converges with physiology research by collecting small differences in the head, face, and pupil movements, and that's plausibly why it's so accurate.

1 Introduction

Eye-tracking has profound applications; from computer vision, to accessibility, to healthcare. Classical eye-tracking models have not performed very well at the task, and it's only the recent advancement of neural networks that have made eye-tracking models a bigger research area [4]. A current state-of-the-art eye-tracking model is iTracker, which is a convolutional neural network (CNN) model that vastly outperforms classical models [5]. But what makes eye-tracking a hard computational task? Many studies have been done on the physiology of eye-tracking, and one such reason is that the task depends on a mixture of subtle movements from the head, face, and pupils [9]. With iTracker's excellent empirical performance, it's plausible that it converges with research in physiology, however no such interpretibility study has been done to verify this hypothesis.

2 Related work

The CNN-based approaches to eye-tracking that we are comparing fall under the broader set of appearance-based methods [3], where the algorithm is given the raw image data as input. Rather than use a single network, CNN-based eye-tracking models typically make use of several networks, each responsible for a smaller region of the input (for example, one network per eye). A common enhancement for the purpose of efficiency is weight-sharing between these networks. Some works exist which compare standard models on the task [2] [7] [1]. There are many existing papers comparing the accuracy of different models and how various transformations on the input impact it [7] [1], yet to the best of our knowledge, there is little work attempting to understand and interpret the features learned by these models [2] .

3 Methods

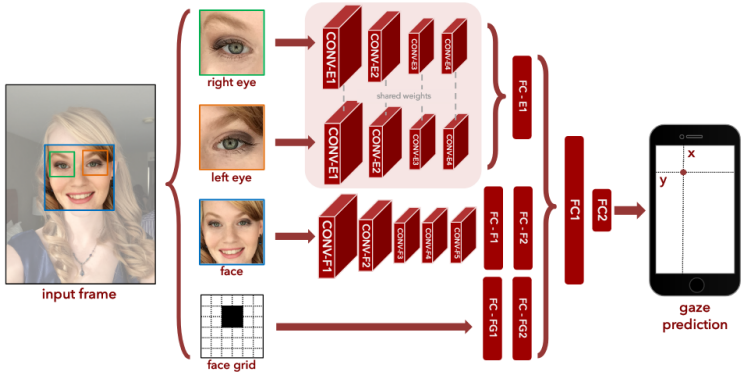

Figure 1: Algorithm Box of iTracker [5]

We'll start by explaining the iTracker CNN model [5], shown in Figure 1. It is composed of three AlexNet [6] branches, which process the right eye, left eye, and face, and one dense MLP branch that processes a face grid (boolean grid identifying the location of the face) as input.

In our study we will analyze 4 different models: the left-eye, right-eye, and face branches, and the full iTracker model. Our goal is to show that the individual branches require large changes in the input image to move screen coordinates, and the whole model will only require small changes.

The three branch models don't directly connect to an MLP that outputs screen coordinates, so we isolated those branches and reconnected them with their own FC layers. Our choice of FC layers mimicks what's done in the whole iTracker, as we compress from size 4192, to 128, to 2.

3.1 Transfer Learning

For each branch, we pre-populate the convolutional layers with weights from the full iTracker model, and then initialize the new FC layers randomly. Then, we retrain all layers with a 10GB partition of the GazeCapture dataset [5] for finetuning. We have two reasons for choosing a subset of GazeCapture: the first is for tractability given our computational resources, and the second is to emphasize our final results. If we can show that an individual branch that is pre-trained from the whole model still requires large differences in the input image to change screen positions, then we would only see that result more prominently if the branch was trained from scratch on a larger dataset.

3.2 DeepDream and SmoothGrad

DeepDream (originally called "Inceptionism") [8] is a method where we use gradient methods to alter an input image, rather than the model parameters. We will use a variant of DeepDream, which will focus on the final output layer rather than the intermediate layers. This method consists of using stochastic gradient descent on an input image for a fixed number of iterations to learn an input that gives a desired target output. We use gradient descent instead of ascent, because we want to minimize the distance to the new target, rather than maximizing the total activation on an intermediate layer. We will compare the initial input to the learned input to observe the difference needed to change the output of the model.

We will use SmoothGrad [10] to study the gradients of the model as a function. This method consists of computing the gradient with respect to a particular image, making several copies of this gradient, adding independent Gaussian noise to each copy, and finally taking the average of these copies. The purpose of this procedure is to eliminate noise from the gradient to more clearly observe the features that impact the behaviour of the model. We will produce plots of the denoised gradient to perform these observations.

4 Results

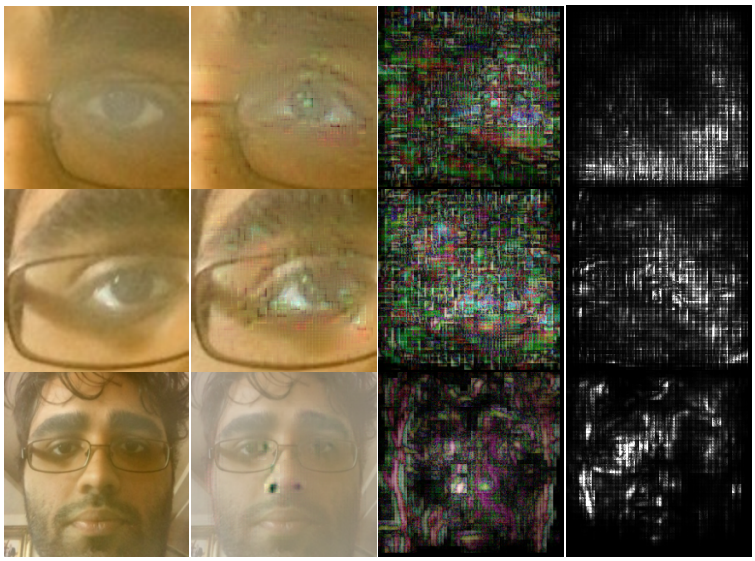

Figure 2: Left eye branch (first row), right eye branch (second row), face branch (third row). Deep dream is used to move the gaze from down to upwards.

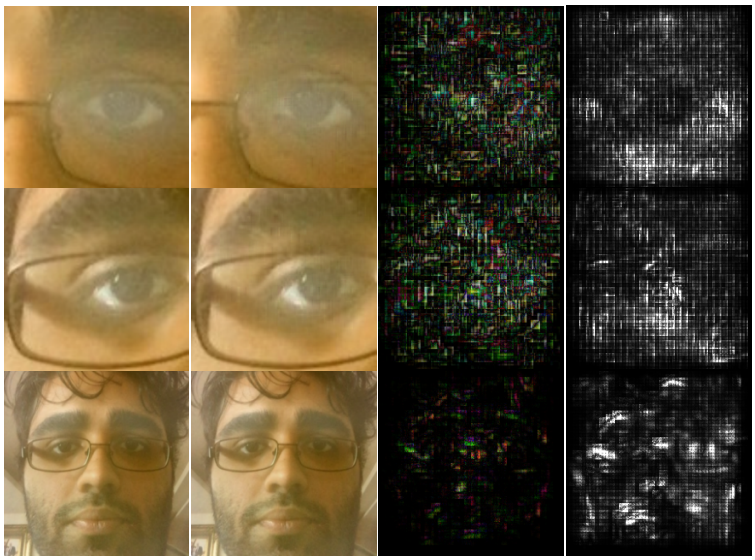

Figure 3: Results for the whole iTracker model. Left-eye image (first row), right-eye image (second row), and face images (third row). Deep dream is used to move the gaze from down to upwards.

Figures 2 and 3 shown here and Figures 4-7 in the Appendix show the outputs of our analysis methods on 3 hand-picked examples. To clarify, the third column shows the natural logarithm of the squared difference between the original image and the DeepDream learned image.

5 Discussion

5.1 Visual interpretation

First we will discuss the SmoothGrad denoised gradients. For gradients of the eye images in the individually trained branches, we see that the area where the gradient is strongest is mostly at the eye itself, whereas in the full model, the gradient in the area surrounding the eye is also noticeable. For the face images, we see the gradient of the full model more closely resembles the features of a face (eyes, nose, mouth, face contour) than the gradient of the individually trained branches.

Now we will discuss the DeepDream learned images. The left and right eye images in the individual branches show changes in the shape of the eyelid and lighting (an added reflection to the left eye in Figure 2 and a darkened bottom eyelid in Figure 6). In the second example (Figure 4), we see a white patch added to right side of each eye, covering the right corner of the eye and the right edge of the iris. This white patch can be interpreted as extending the sclera (the white part of the eye).

The learned face images for the individually trained branches are not readily interpretable, however the difference images (shown in the third column) can provide insight into which features most strongly impact the output of the model. We see stronger changes made in the contour of the face, as well as the eyes, eyebrows, and nose. For the nose, we see that the strongest changes occur at the nostrils. It is worth noting that the shape and relative location of the nostrils can be used to identify the angle of the head.

The learned images for the full model (in Figures 3,5,7) are not noticeably different from the original images. However, the difference images show that subtle changes were made to the image, which were enough to strongly influence the output of the image. This supports our hypothesis that the full model can detect subtle changes in the eyes and other facial features.

5.2 Numerical difference comparison

We also compare the whole iTracker model to the individual branches numerically. For each image in a set of 20, we compute the DeepDream difference matrix and sum all elements inside. We do the same with the associated SmoothGrad matrix. The summed DeepDream values tell us how much "difference" is computed for the image, and the summed SmoothGrad values tells us how much "importance" is computed by the model. In Table 1 (last page) we show these summed values for each model, averaged across the 20 images.

In Table 1, we can see that the full iTracker model computes higher DeepDream sums and lower SmoothGrad sums than each of the individual branches on the left, right, and face inputs. Intuitively, this numerically shows that the full iTracker model computes large differences, but does not set high importance on any specific area. This is in contrast to an individual branch, which computes smaller overall differences, while placing high importance on a specific targeted area. This continues to provide evidence that the full iTracker model detects many small, subtle differences in the input images, as defined in physiology research.

6 Conclusion

In this work, we perform an interpretation study on the state-of-the-art eye-tracking model, iTracker. We compare the whole iTracker model with it's three individual branch components, visually and numerically. We use DeepDream to investigate the differences each model looks for, and we use SmoothGrad to investigate what each model deems important. In our numerical comparisons, we find that the whole iTracker model produces larger summed Deep Dream differences and smaller summed SmoothGrad results compared to separated individual branches. Our results conclusively show that the whole iTracker model collects many differences across the head, face, and pupil areas, but doesn't highlight any one region specifically. This converges with research in eye-tracking physiology.

Sidharth trained the iTracker models, while David and Halim trained Deep Dream and SmoothGrad respectively. For a specific description of each team member's contributions, please see the Attributions section (last page). Project code is included as interpreting_itracker.ipynb.

References

[1] A. A. Akinyelu and P. Blignaut. "Convolutional Neural Network-Based Methods for Eye Gaze Estimation: A Survey". In: *IEEE Access* 8 (2020), pp. 142581–142605. DOI: 10.1109/ACCESS.2020.3013540.

[2] Avoy Datta. "MobileGaze: An efficient framework for mobile gaze tracking". In: (Mar. 2019).

[3] Dan Hansen and Qiang Ji. "In the Eye of the Beholder: A Survey of Models for Eyes and Gaze". In: *IEEE transactions on pattern analysis and machine intelligence* 32 (Mar. 2010), pp. 478–500. DOI: 10.1109/TPAMI.2009.30.

[4] Ahmad F. Klaib et al. "Eye tracking algorithms, techniques, tools, and applications with an emphasis on machine learning and Internet of Things technologies". In: *Expert Systems with Applications* 166 (2021), p. 114037. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2020.114037. URL: https://www.sciencedirect.com/science/article/pii/S0957417420308071.

[5] Kyle Krafka et al. "Eye Tracking for Everyone". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[6] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Commun. ACM* 60.6 (May 2017), pp. 84–90. ISSN: 0001-0782. DOI: 10.1145/3065386. URL: https://doi.org/10.1145/3065386.

[7] Joseph Lemley et al. "Efficient CNN Implementation for Eye-Gaze Estimation on Low-Power/Low-Quality Consumer Imaging Systems". In: *CoRR* abs/1806.10890 (2018). arXiv: 1806.10890. URL: http://arxiv.org/abs/1806.10890.

[8] Alexander Mordvintsev, Christopher Olah, and Mike Tyka. *Inceptionism: Going deeper into neural networks*. June 2015. URL: https://ai.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html.

[9] Marcus Nyström, Richard Andersson, and Kenneth. Holmqvist. "The influence of calibration method and eye physiology on eyetracking data quality". In: *Behavior Research Methods* (2013). DOI: 10.3758/s13428-012-0247-4. URL: https://doi.org/10.3758/s13428-012-0247-4.

[10] Daniel Smilkov et al. "SmoothGrad: removing noise by adding noise". In: *CoRR* abs/1706.03825 (2017). arXiv: 1706.03825. URL: http://arxiv.org/abs/1706.03825.

7 Appendix

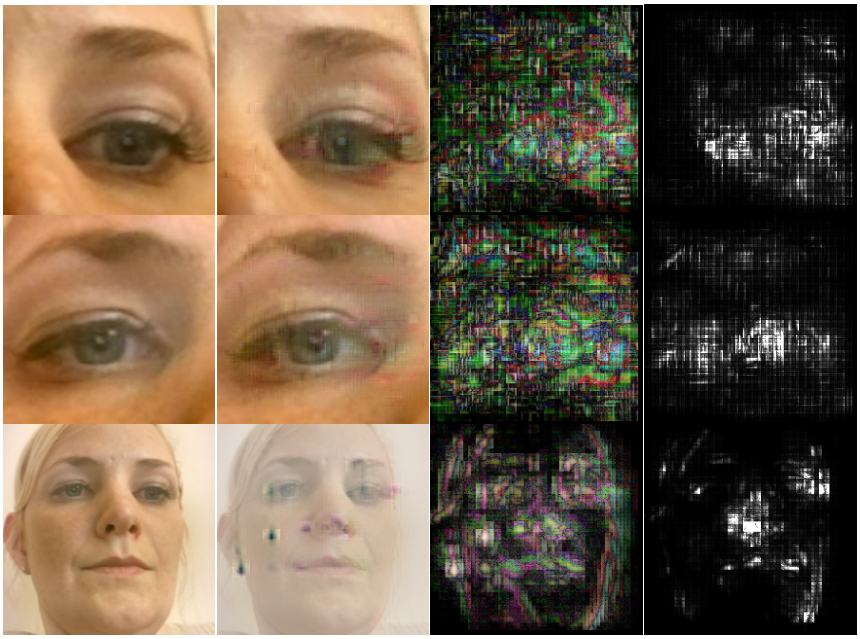

Figure 4: Left eye branch (first row), right eye branch (second row), face branch (third row). Deep dream is used to move the gaze from right to leftwards.

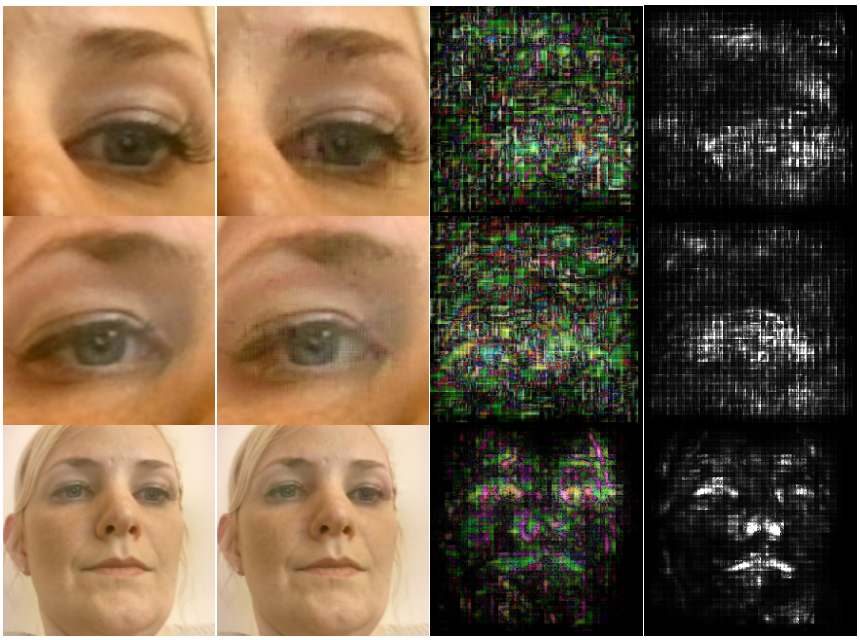

Figure 5: Results for the whole iTracker model. Left-eye image (first row), right-eye image (second row), and face images (third row). Deep dream is used to move the gaze from right to leftwards.

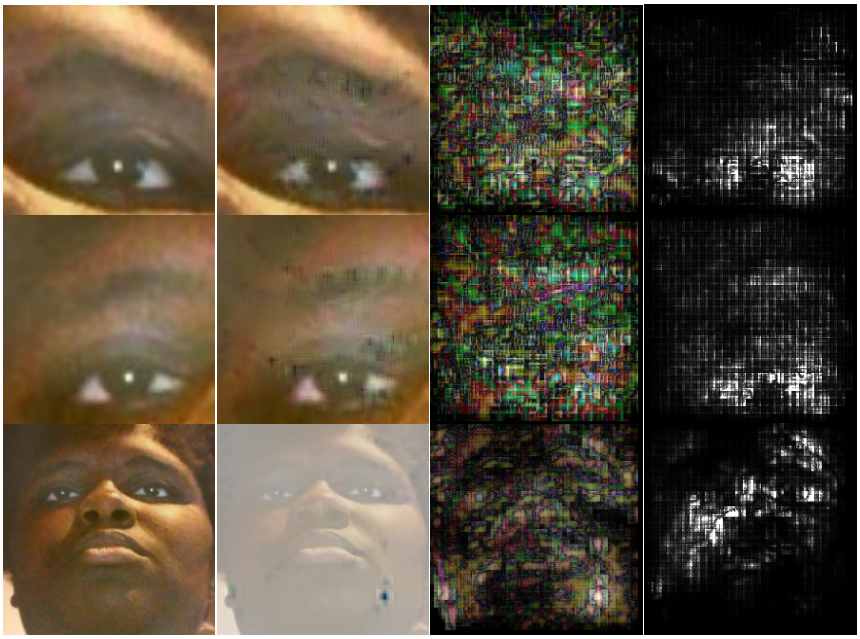

Figure 6: Left eye branch (first row), right eye branch (second row), face branch (third row). Deep dream is used to move the gaze from up to downwards.

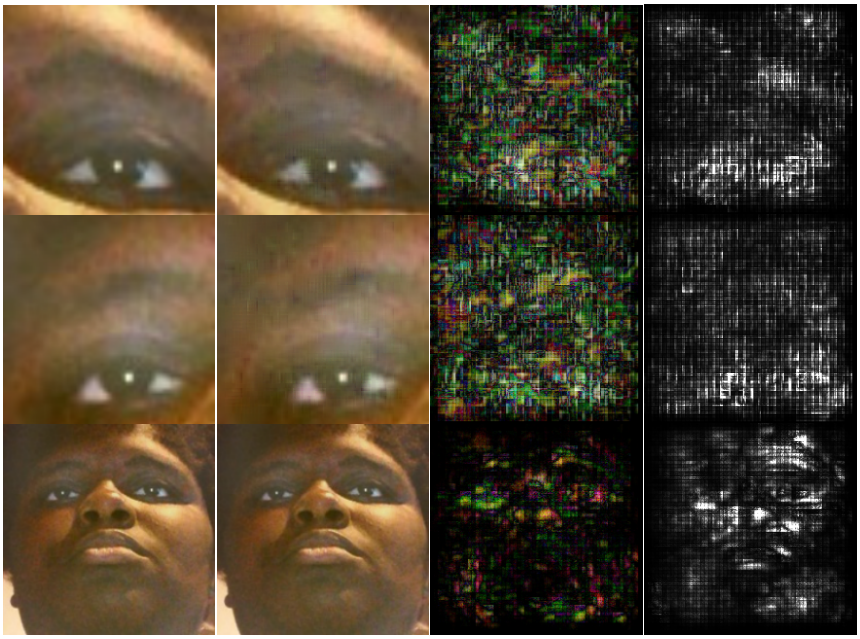

Figure 7: Results for the whole iTracker model. Left-eye image (first row), right-eye image (second row), and face images (third row). Deep dream is used to move the gaze from up to downwards.

	Left-eye branch	Right-eye branch	Face branch	iTracker left eye image	iTracker right eye image	iTracker face image
Avg summed DeepDream difference	684.97	850.48	161.16	898.88	904.07	645.28
Avg summed SmoothGrad	6.85	4.26	0.17	4.18	4.16	0.09

Table 1: Averaged across size 20 image set, we compute a DeepDream difference matrix, and sum all elements inside. We do the same with the image's SmoothGrad matrix.

8 Attributions

	Percentage	Work
Sidharth	33%	Codebase setup, methods / abstract / intro, creating branch models, pretraining, numerical difference comparison
David	33%	DeepDream: research, implementation, producing results, visual interpretation of results
Halim	33%	SmoothGrad: research, implementation, testing, producing results, conclusion