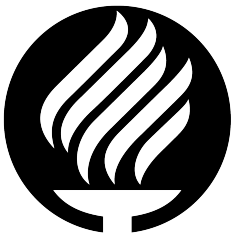
Instituto Tecnológico y de Estudios Superiores de Monterrey
Campus Querétaro
Departamento de Computación y Mecatrónica



Tecnológico de Monterrey

Gaze-Tracking with Convolutional Neural Networks

by

Alberto Castañeda Arana

Proyecto Integrador para el Desarrollo de Soluciones Empresariales

Ingeniería
en
Sistemas Computacionales

Asesor: Dr. José Antonio Cantoral Ceballos

Santiago de Querétaro, Querétaro, México
15/06/2022

List of Figures

List of Tables

Contents

Chapter 1

Introduction

1.1 Gaze-Tracking

Applications of estimating the human eye gaze direction, or Gaze-Tracking, can be found in multiple study areas such as health care [3], virtual-reality [10], and human-computer interaction [9]. Devices that utilize gaze-tracking methods for human-computer interaction are often referred to as Eye Trackers. While there is commercial use for Eye Trackers, they have not reached the accuracy needed to allow users with fine-motor disabilities to interact with a computer by using their own eyes as a replacement for a traditional mouse device.

Conventional methods for gaze-tracking require additional devices such as infrared light in order to make the gaze estimations. However, these methods are prone to error in poor light conditions.

Research in computer vision and Deep Learning have allowed improvement in Gaze-Tracking systems by allowing better predictions while also being less intrusive for users with no additional devices required.

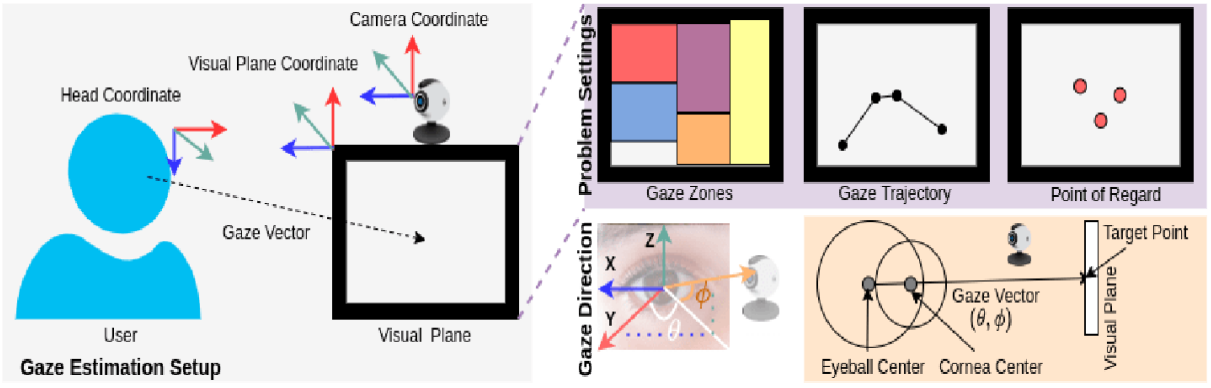


FIGURE 1.1: Overview of Gaze-Estimation setups. Recovered from [2]

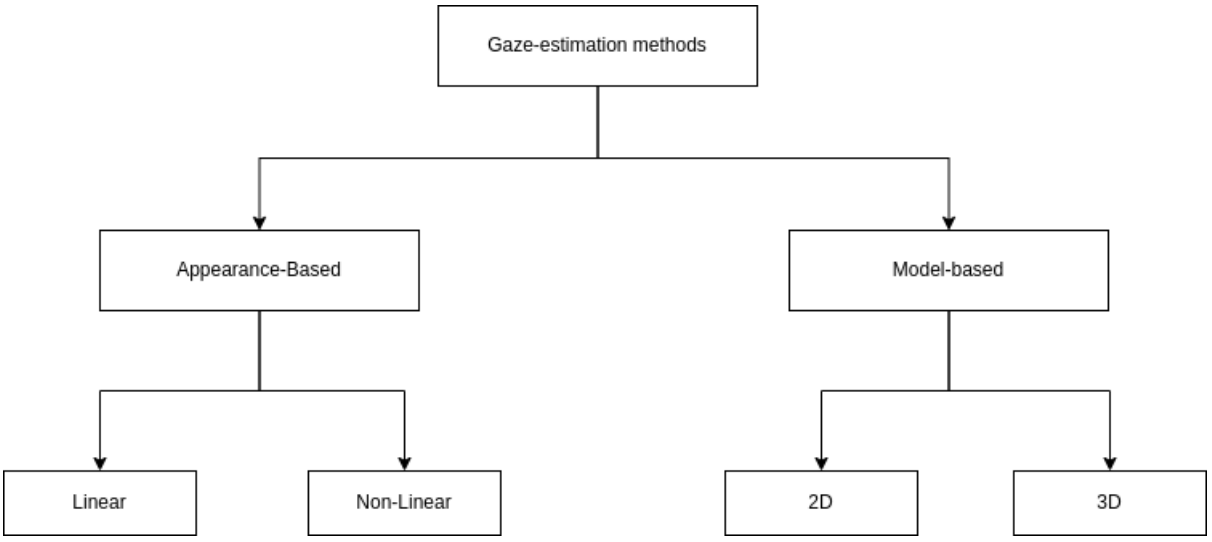


FIGURE 1.2: Overview classification of Gaze-Tracking methods

Gaze-Tracking with computer vision can be achieved with two different type of methods: Model-based methods and Appearance-based methods. These methods will be described in the following sections.

1.2 Model-based methods

Model-based methods, or sometimes referred to as feature-based methods, use the geometric form of the eyes to make the eye gaze prediction. As seen on Figure 1.2, these can be interpreted as 3D models. Model-based 3D Gaze-Estimation methods use 3D eyeball models and estimate the gaze direction using geometric eye features, such as the iris center and the eye corners [4].

Furthermore, most these methods require an infrared light source (IR) in order to create a refraction in the human eye which is then used to calculate the eye gaze direction by comparing the refraction position with the pupil position. The main drawback of model-based methods is that they can have a lower accuracy with low-resolution images and poor-condition lighting environments. Additionally, all of them have limited working distance.1

Some state-of-the-art commercial products that have adapted these techniques are Tobii Eye Tracker (`https://www.tobii.com/`), Eye Tribe (`https://imotions.com/hardware/the-eye-tribe-tracker/`) and EyeSee (`https://eyesee-research.com/`). While these products allow very accurate predictions for the Eye Tracking, they suffer from the previously mentioned model based methods disadvantages while also requiring users to purchase additional expensive equipment. Most Eye-Tracking products also require wearable glasses peripherals that can be intrusive for daily usage.

With the increase of interest and usage of Gaze-Tracking, there is also a growing need for new Eye-Trackers that are cheap and easy to use. Gaze-Tracking methods that utilize built-in or external cameras included in almost every computer can be used as a way of making eye tracking more accessible. Appearance-based approaches are beginning to be researched to achieve the previous goal.

1.3 Appearance-based methods

Recently, investigations for appearance based methods with Deep Learning has surfaced which have demonstrated high accuracy while only using on-the-shelf web cameras to capture the human eye appearance hence the method name.

Conventional appearance-based methods used regression in order to learn the mappings from appearance to human gaze. However, since these methods use the appearance of the eye, the main challenges are head motion and subject differences that alter the eye appearance on use.

To overcome the previous challenges, appearance-based Gaze-Estimation with Deep Learning has become a trend in research. Deep Learning based methods can extract high-level abstract gaze features from high-dimensional images, making them more robust and accurate than conventional methods. The mapping between eye-images and gaze direction can be achieved with various regression techniques, such as neural networks, local interpolation, or Gaussian process regression. Convolutional Neural Networks (CNNs) models are commonly used in computer vision tasks for its high accuracy and performance. Current state-of-the-art solutions have begun to adopt CNN based architectures, adopting the advantages of appearance-based methods and achieving high accuracy. CNN models used for Gaze-Tracking learn from RGB images which are preprocessed to segment the human eyes in training.

Some of the drawbacks that appearance-based methods have been that they require more images for training than model-based methods, which are not easily retrieved. Additionally, since most appearance-based methods require deep-learning architectures such as Convolutional Neural Networks, they require higher computational power for training. Several works apply CNNs for gaze estimation, as they have shown to outperform traditional model-based approaches [15].

Chapter 2

State-of-the-art

2.1 Datasets

TABLE 2.1: Dataset comparison

Dataset	Total #	Size	Sub-jects #	Gaze Points #	Full Face	2D Gaze	3D Gaze	Ref
GazeCapture 2016 images	2.4M	136 GB	1,474	203	✓	✓	✗	[7]
MPIIGaze 2015 images	213K	2.1 GB	15	203	✗	✓	✓	[15]
CAVE 2013	6K images	2.2 GB	56	105	✓	✗	✓	[12]
EyeDiap 2014	94 videos	11.6 TB	16	105	✓	✓	✓	[12]
ETH-XGaze 2020	1.1M	130 GB	110	N/A	✓	✓	✓	[16]

Until recently, datasets for Gaze-Estimation models training were built manually by taking pictures of different users' faces by placing their head firmly in a stand. Training models require

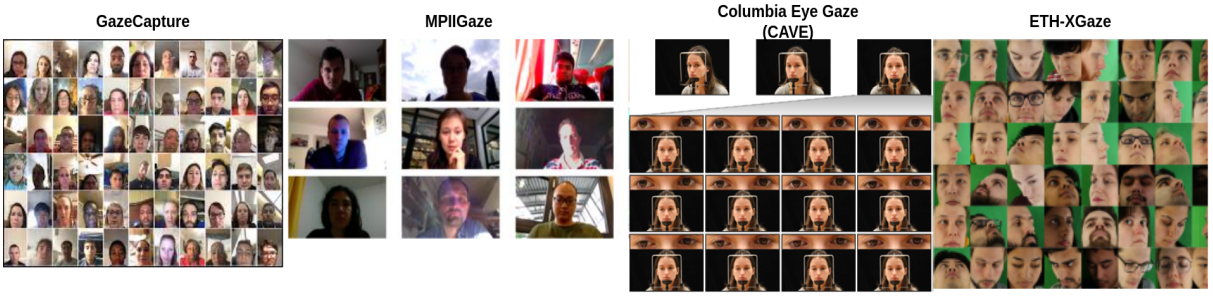


FIGURE 2.1: Samples from different datasets. Recovered from [7], [15], [12] and [16] respectively.

multiple high-quality images, which makes the previous methods slow and not ideal. To improve the previous methods' problem, multiple data sets containing images for eye tracking have been compiled with crowd founding to train new models. Figure 2.1 represents an overview of different state-of-the-art datasets, while Table 2.1 provides an analytical comparison of each of them.

2.1.1 GazeCapture

Krafka et al. built and released the GazeCapture dataset, which crowdfunded a mobile-based data set of approximately 1500 participants from a wide variety of backgrounds captured by an iOS application [7]. However, given that this dataset is built with mobile users, training with this dataset may lead to a huge bias towards mobile users which could lead to poor results with laptop or desktop use. Additionally, the authors provide pre-trained models utilizing the dataset available as open source (`https://github.com/CSAILVision/GazeCapture`).

2.1.2 MMPIIGaze

Another state-of-the-art dataset that has been widely used is the MMPIIGaze dataset created by Zhang et al. [15]. The dataset contains a total of 213,659 images from 15 participants. For each participant, there is a varied number of images ranging from 34,745 to 1,498. The dataset contains multiple features, our target being the 3D gaze target position related to camera. The dataset is taken from the participant's laptops, contrasting GazeCapture's images taken from mobile devices.

2.1.3 Columbia Eye Gaze / CAVE

The CAVE dataset [12] consists of 5,880 images of 56 different participants. For each participant, there are 5 head poses with 21 gaze directions per head pose. The authors describe the dataset as having more fixed gaze targets than other publicly available gaze datasets. The participants are ethnically diverse and some of them are captured wearing glasses which makes this a strong candidate for learning. The dataset comes with already-segmented eye areas, so segmentation step can be skipped.

2.1.4 EyeDiap

The EYEDIAP dataset was designed to train and evaluate gaze estimation algorithms from RGB and RGB-D data. By using videos instead of pictures, it manages to be the biggest sized dataset of the state-of-the-art. Additionally, the recording methodology used is designed to

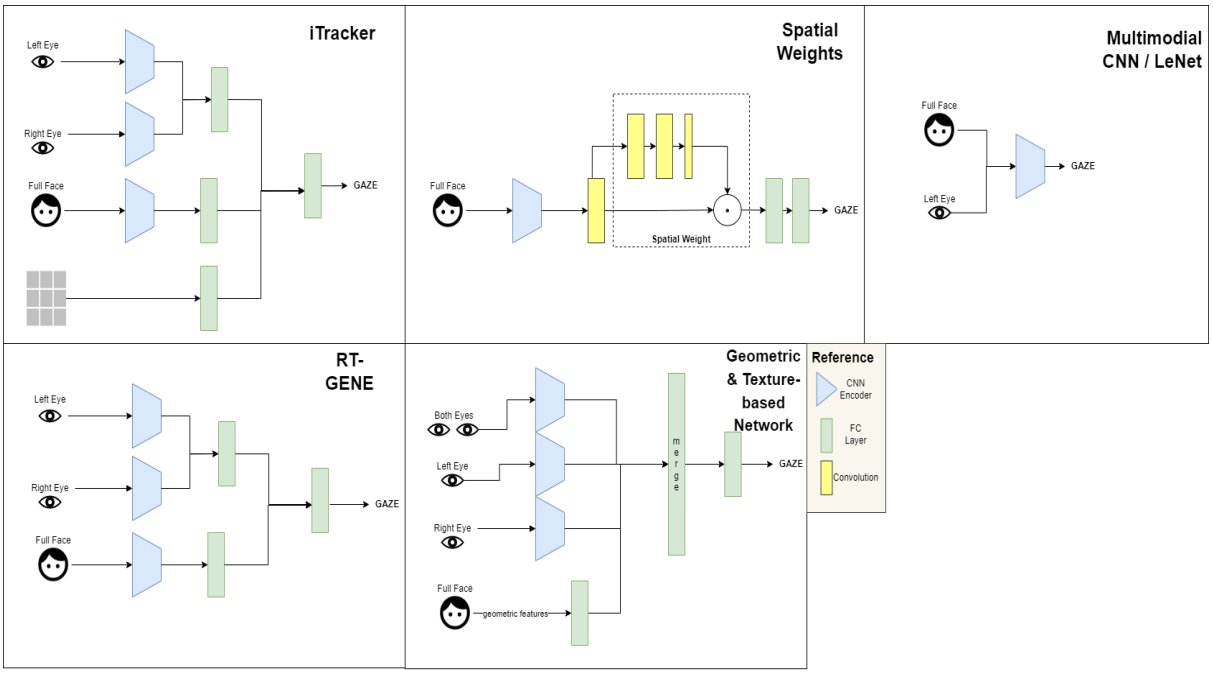


FIGURE 2.2: Generalized architectures of state-of-the-art models

systematically include, and isolate, most of the variables which affect gaze estimation algorithms: head pose and lightning variations.

2.1.5 ETH-XGaze

Finally, the ETH-XGaze dataset [16] consists of over 1,000,000 images of varying gaze under extreme head posers collected from 110 participants While the extreme varied head poses provide good opportunities for learning, for the case of this project the head poses that are not looking into the screen would not be useful so hey must be discarded.

2.1.6 Discussion

As seen on Table 2.1, the dataset with the biggest scale is EyeDiap, by having terabytes of videos with distinct frames of information. While training with the EyeDiap dataset would create the biggest trained model, the huge size of its collection of frames in the videos would require high computational power and training time in order to fully train the model. While GazeCapture is also a great candidate, its small number of gaze points is insufficient for the purpose of gaze-estimation in a computer screen.

TABLE 2.2: 2D (cm) and 3D (°) Gaze-Estimation Benchmark of State-of-the-Art

Models \ Datasets	MPI-IGaze	Eye-Diap	GazeCapture (Tablet)	GazeCapture (Phone)	CAVE
Multimodal Convolutional Neural Network	13.9 °	10.5 °	N/A	N/A	N/A
iTracker	7.67 cm	10.13 cm	2.81 cm	1.86 cm	N/A
RT-Gene	5.36 cm	7.19 cm	N/A	N/A	N/A
Spatial Weights	4.2 cm	8.56 cm	N/A	N/A	N/A
Geometric & Texture-based Networks	N/A	N/A	N/A	N/A	2.22 °

2.2 Related Work

2.2.1 Multimodal Convolutional Neural Network

Zhang et al. proposed a six-layered CNN based on LeNet that takes the eye image as input and combines the head pose in the last fully connected layer of the network [15]. The results demonstrated a large performance gap between person-specific training results (RF model) with the proposed CNN model and revealed the potential of appearance-based gaze estimation.

2.2.2 Spatial Weight CNN

After the previous proposal, Zhang et al. proposed a spatial weights CNN based on the AlexNet architecture [8] that takes the full face image as input [17]. The spatial weights classify the importance of different facial areas in order to perform the Gaze-Estimation. This mechanism includes three additional 1 x 1 convolutional layer followed by a ReLU activation. Compared to other approaches only use eye regions, this method was demonstrated to be more robust against appearance variation such as head pose as well as illumination. The results achieved an error of 4.8-6.0°.

2.2.3 iTracker

Kannan et al. proposed a CNN architecture for an iOS based Gaze-Tracking system called iTracker [6]. The model's large neural network was constructed with multiple smaller neural networks, with four total inputs: a right eye image, a left eye image, a face image, and a face-grid. The face-grid is a 25x25 binary grid that indicates where the user face is placed relatively to the camera view.

This model achieved 1.66 centimeter error with no calibration needed. The GazeCapture dataset which is described in section 2.1 was used to train this model.

While achieving good results, this model is biased with mobile usage which makes it unusable for laptop or computer desktop usage. Additionally, the authors describe there being room for improvement for better segmentation of eyes. However, by adjusting the model's face-grid dimensions to simulate webcam dimensions and with a different dataset, the model could potentially be used for desktop/laptop use.

2.2.4 Geometric & Texture-based Networks

Jyoti et al. proposed an ensemble of networks that uses the full face, left and right eye [5]. A Deep Neural Network (DNN) is trained using the face geometric features. Secondly, three CNNs are trained for the texture based features i.e. the segment of the left eye, right eye, and the combined eyes. The model was trained with the publicly available Columbia Eye Gaze and TabletGaze datasets for experiments. When trained with the Columbia Eye Gaza dataset, experiments resulted in an error of 2.22°. Additionally, the authors emphasize on the importance of choosing different activation functions for different results by comparing the model outcome with ReLU vs Swish, where the former achieved better performance for geometric network while Swish showed better performance on the rest.

2.2.5 RT-GENE

Fischer et al. proposed a two-stream VGG network that takes the left and right as inputs [1]. This approach considers large camera-to-user distances and high variations in head pose found in natural environments. This approach allows automatic annotation of subject's ground truth gaze and head pose regardless of large camera-user distances, achieving an error of 7.7°. However, the approach setup requires additional devices such as mobile eye-tracking glasses (Kinect v2 RGB-D camera).

2.2.6 Discussion

As seen on Table 2.2, iTracker outperforms every other 2D estimation modeLs when using phone/tablet based datasets. However, when using regular datasets taken from computer dimensions, the performance of iTracker drastically becomes lower. On the other hand, Geometric & Texture-based networks has the best performance on 3D gaze estimation.

Many approaches similar to Multimodal Convolutional Networks have been proposed, using different general architectures such as Lenet, Resnet, and others. However, as of the date of

writing this paper, the application of GoogleNet for Gaze-Estimation models have not been proposed. GoogleNet is a deep convolutional neural network architecture developed by Szegedy et al. in 2014 [14]. GoogleNet has shown to outperform traditional deep convolutional neural networks such as LeNet. Having a history of outperforming these traditional methods, in this project a GoogleNet model approach for Gaze-Estimation is proposed as a potential alternative to state-of-the-art systems.

Chapter 3

Development

3.1 Objectives

For this research, the goal is to design and train new models for Gaze-Estimation that outperform current state-of-the art systems. Accuracy can be determined with evaluation metrics described in section 3.2, while the speed of the model must be fast enough in order to potentially use this gaze-estimation system in real-time on the average user computer. The models must be able to query predictions from a standard desktop computer and laptop webcam while handling the possible variance in head pose and lightning conditions.

3.2 Evaluation Metrics

There are two commonly used metrics for performance evaluation in Gaze-Tracking Systems: Euclidean distance and the angular error. Angular error is used when referring to 3D gaze estimation, while euclidean distance is used for 2D gaze estimation.

Assuming the gaze direction is $\boldsymbol{g} \in \mathbb{R}^{\mathbf{3}}$, and the estimated gaze direction is $\hat{\boldsymbol{g}} \in \mathbb{R}^{\mathbf{3}}$, the angular error can be calculated with the formula:

$$\iota_{\boldsymbol{angular}} = \frac{\boldsymbol{g} \cdot \hat{\boldsymbol{g}}}{\|\boldsymbol{g}\|\|\hat{\boldsymbol{g}}\|} \tag{3.1}$$

For 2D gaze estimation, the actual gaze position is denoted as $\boldsymbol{p} \in \mathbb{R}^{\mathbf{2}}$ while the estimated gaze position as $\hat{\boldsymbol{g}} \in \mathbb{R}^{\mathbf{2}}$. This would result in the following formula:

$$\iota_{\boldsymbol{Euclidean}} = \|\boldsymbol{p} - \hat{\boldsymbol{p}}\| \tag{3.2}$$

For the purpose of this research, the models will use 3D angular error for their predictions.

3.3 Proposed Models

Given that multiple models are being used for benchmarking and comparison, the following models will be trained:

- LeNet [Scratch]
- GoogleNet (Inception V3) [Scratch]
- GoogleNet (Inception V3) [Pre-Trained]
- Resnet-18 [Pre-Trained]
- Resnet-34 [Pre-Trained]
- Resnet-50 [Pre-Trained]

Pre-trained models are also taken into account, since the trade-offs between training time and accuracy can be analyzed. Pre-trained models are modified so its final classifier layer (usually a fully connected layer or FC) is modified for the context of the problem. In this case, the 3D gaze estimation would require an output of two dimensions. Additionally, a new convolutional layer can be added to the beginning of the model in order to handle different input sizes that the pre-trained model.

3.4 Framework

In order to build and train the previously proposed models, the PyTorch framework (https://pytorch.org/) will be used due to its simplicity and flexibility. Additionally, PyTorch provides multiple pre-trained models that can be used for transfer learning, including some models proposed in section 3.3. Results will be stored with the Tensorboard library (https://www.tensorflow.org/tensorboard) and converted to figures with the Matplotlib and Seaborn libraries.

Given the time constraints of this project, an existing PyTorch implementation of Gaze Estimation was chosen and extended for this project. The project was developed by GitHub user "Hysts" in 2018, and is available as open source (https://github.com/hysts/pytorch_mpiigaze. The repository includes scripts, processes, and tools to train different models with

configuration files. By default, the available datasets in this repository are MPIIGaze and MPIIFaceGaze, the latter being a modified version of MPIIGaze which was described in Figure 2.1. Additional models will be added to the repository, as well as some additional code changes such as the implementation of One Cycle scheduler, which will be described in subsection 3.6.1. The altered version of the repository, or the fork, can be found in the following repository: https://github.com/albertcastaned/pytorch_mpiigaze.

The training will be accomplished using an NVIDIA 1060 GTX 3GB GPU, and a AMD Ryzen 5 1600 Six-Core Processor. For faster training, CUDA is configured to use the GPU as the training device.

3.5 Dataset preprocessing

For demonstration purposes, the small sized MPIIGaze dataset will be used for the training of this project's models. As with other Gaze-Estimation datasets, it contains multiple images of the user's eyes which must be transformed to serve as the input of the models.

3.6 Training

The training of the proposed models will be of a total of 15 epochs or cycles. After each epoch is complete, the training time, accuracy (angular error), loss value, and current learning rate will be simultaneously displayed on the terminal and saved as Tensorboard metrics.

As mentioned on previous sections, some models are pre-trained with just the classifier layer replaced. By doing this, faster training times and better performance are expected from these models,

To handle potential GPU or CPU crashes or application-level errors during training, checkpoints are set to be saved periodically in case there is need for reloading.

3.6.1 One Cycle Learning Rate

To further optimize the training of the models and get overall better performance, One Cycle Learning Rate for "super-convergence" described by Leslie Smith et al. in 2017 [13]. The learning rate of the training increases until it reaches a specified percentage of completion (PoC) and starts decreasing at the same rate until it completes another cycle of length PoC.

While PyTorch supports One Cycle scheduler for training, the original code retrieved supported default schedulers which requires additional code for One Cycle scheduler support.

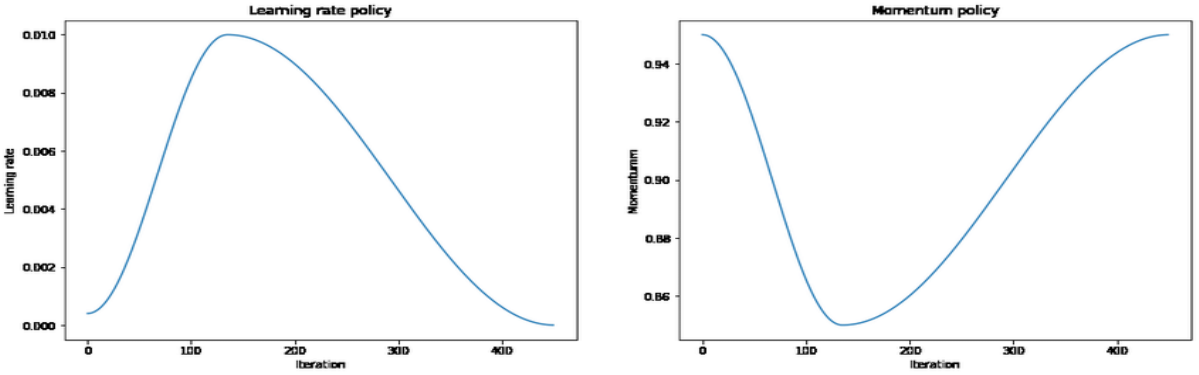


FIGURE 3.1: One Cycle Learning Rate. Recovered from [11].

By default, all models were set to have a PoC of 0.3 or 30% of completion percentage. The behavior of the change in learning rate is represented by Figure. TODO ADD .

3.7 Saving the model

Once the training has been completed, the models will be saved as .h5 files so that they can be transferred for further learning, and to use it with a demonstration application by using OpenCV. As a result, the model will predict the angular gaze of the user, while the OpenCV program will query them at constant intervals of time and displaying this 3D angle on screen.

Chapter 4

Results

Chapter 5

Conclusion

References

[1] Tobias Fischer, Hyung Chang, and Yiannis Demiris. "RT-GENE: Real-Time Eye Gaze Estimation in Natural Environments". In: Sept. 2018. DOI: 10.1007/978-3-030-01249-6_21.

[2] Shreya Ghosh et al. "Automatic Gaze Analysis: A Survey of Deep Learning based Approaches". In: *arXiv:2108.05479 [cs]* (Aug. 2021). arXiv: 2108.05479. URL: http://arxiv.org/abs/2108.05479 (visited on 03/31/2022).

[3] Alessandro Grillini et al. "Towards Using the Spatio-Temporal Properties of Eye Movements to Classify Visual Field Defects". In: *Proceedings of the 2018 ACM Symposium on Eye Tracking Research Applications*. ETRA '18. Warsaw, Poland: Association for Computing Machinery, 2018. ISBN: 9781450357067. DOI: 10.1145/3204493.3204590. URL: https://doi.org/10.1145/3204493.3204590.

[4] E.D. Guestrin and M. Eizenman. "General theory of remote gaze estimation using the pupil center and corneal reflections". In: *IEEE Transactions on Biomedical Engineering* 53.6 (2006), pp. 1124–1133. DOI: 10.1109/TBME.2005.863952.

[5] Shreyank Jyoti and Abhinav Dhall. "Automatic Eye Gaze Estimation using Geometric & Texture-based Networks". In: Aug. 2018, pp. 2474–2479. DOI: 10.1109/ICPR.2018.8545162.

[6] Harini D. Kannan. "Eye tracking for the iPhone using deep learning". eng. Accepted: 2018-01-12T20:59:18Z. Thesis. Massachusetts Institute of Technology, 2017. URL: https://dspace.mit.edu/handle/1721.1/113142 (visited on 03/27/2022).

[7] K. Krafka et al. "Eye Tracking for Everyone". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, June 2016, pp. 2176–2184. DOI: 10.1109/CVPR.2016.239. URL: https://doi.ieeecomputersociety.org/10.1109/CVPR.2016.239.

[8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Vol. 25. Curran Associates, Inc., 2012. URL: https://papers.nips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html (visited on 04/15/2022).

[9] Raphael Menges et al. "GazeTheWeb: A Gaze-Controlled Web Browser". In: *Proceedings of the 14th International Web for All Conference*. W4A '17. Perth, Western Australia, Australia: Association for Computing Machinery, 2017. ISBN: 9781450349000. DOI: 10.1145/3058555.3058582. URL: https://doi.org/10.1145/3058555.3058582.

[10] Benjamin Outram et al. "Anyorbit: orbital navigation in virtual environments with eye-tracking". In: June 2018, pp. 1–5. DOI: 10.1145/3204493.3209579.

[11] Miguel Romero et al. *Training Deep Learning models with small datasets*. Dec. 2019.

[12] B.A. Smith et al. "Gaze Locking: Passive Eye Contact Detection for Human?Object Interaction". In: *ACM Symposium on User Interface Software and Technology (UIST)*. Oct. 2013, pp. 271–280.

[13] Leslie N. Smith and Nicholay Topin. "Super-convergence: very fast training of neural networks using large learning rates". In: *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*. Ed. by Tien Pham. Vol. 11006. International Society for Optics and Photonics. SPIE, 2019, pp. 369–386. DOI: 10.1117/12.2520589. URL: https://doi.org/10.1117/12.2520589.

[14] Christian Szegedy et al. "Going deeper with convolutions". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1–9. DOI: 10.1109/CVPR.2015.7298594.

[15] Xucong Zhang et al. "Appearance-based gaze estimation in the wild". en. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Boston, MA, USA: IEEE, June 2015, pp. 4511–4520. ISBN: 978-1-4673-6964-0. DOI: 10.1109/CVPR.2015.7299081. URL: http://ieeexplore.ieee.org/document/7299081/ (visited on 04/06/2022).

[16] Xucong Zhang et al. "ETH-XGaze: A Large Scale Dataset for Gaze Estimation under Extreme Head Pose and Gaze Variation". In: *European Conference on Computer Vision (ECCV)*. 2020.

[17] Xucong Zhang et al. "It's Written All Over Your Face: Full-Face Appearance-Based Gaze Estimation". In: *arXiv:1611.08860 [cs]* (May 2017). arXiv: 1611.08860. URL: http://arxiv.org/abs/1611.08860 (visited on 03/31/2022).