Silesian
University
of Technology

DOCTORAL THESIS

Using a Camera to Determine Human Gaze Point

Mohd Faizan ANSARI
Student identification number: 4948

Discipline: Information and Communication Technology
Specialization: Informatics

SUPERVISOR
Pawel Kasprowski, PhD, DSc
DEPARTMENT: Applied Informatics
Faculty of Automatic Control, Electronics and Computer Science

Gliwice 2025

Author's Declaration

I, **Mohd Faizan Ansari**, declare that this thesis titled, *"Using the Camera to Determine Human Gaze Point"*, and the work presented in it are my own. I confirm that this work was conducted entirely or primarily during my candidature for a research degree at Silesian University of Technology. Any part of this thesis previously submitted for a degree or other qualification at this or any other institution has been clearly identified. All references to the published work of others are properly attributed and clearly cited. All quotations from the works of others are clearly identified, and the sources are appropriately credited. Except for these quotations, this thesis is entirely my original work. I have acknowledged all major sources of assistance and support. For work conducted jointly with others, I have explicitly stated what contributions were made by others and what I contributed myself.

Author's Signature: _____

Date: _____

Dissertation Copyright Authorization

I hereby grant authorization for this thesis to **Silesian University of Technology**. The University and Library are permitted to incorporate, duplicate, and use it in various formats (paper, CD-ROM, other digital media) without restrictions on location, duration, or frequency. In accordance with the Copyright Law, users are permitted to search online, read, download, or print the thesis.

Acknowledgements

Completing this PhD journey has been one of the most challenging yet rewarding endeavors of my life, and it would not have been possible without the unwavering support, guidance, and inspiration of countless individuals. To them, I owe my deepest gratitude.

First and foremost, I sincerely thank my supervisor, Dr. hab. inż. Paweł Kasprowski, PhD, DSc, for his invaluable guidance throughout this journey. Your expertise, dedication, and encouragement helped me navigate even the toughest phases of this research. Thank you for always taking the time to carefully review my drafts, for pushing me to think critically, and for trusting my ideas when I struggled to believe in them myself. Your thoughtful questions and advice not only strengthened this thesis but also taught me how to approach problems with clarity and confidence. Beyond your academic support, your understanding during moments of stress reminded me that great mentors care about their students as people, not just researchers. Much of what I've accomplished here is thanks to your guidance, and I will always appreciate the role you played in my growth.

To my beloved family members,thank you for being my anchor. Your sacrifices, from the smallest gestures to life-changing decisions, gave me the privilege of pursuing this dream. Mom and Dad, you taught me the value of hard work and curiosity long before I stepped into a lab, and your endless encouragement, whether through care packages, prayers, or reassuring words, carried me through moments of exhaustion.

My friends, old and new, deserve special recognition for keeping me grounded., thank you for the coffee runs, the laughter-filled breaks, and the relentless reminders that life exists beyond data and deadlines. Your ability to listen to my rants about my models one minute and distract me with travel plans or movie nights the next kept me sane. To my academic colleagues, thank you for brainstorming sessions, camaraderie during conferences, and shared triumphs and frustrations: this journey would have been much lonelier without you.

I am also indebted to Silesian University of Technology for providing the resources, infrastructure, and intellectual community that nurtured this research. Special thanks to my department for their financial support, which allowed me to focus fully on this work. To the participants of my study, thank you for trusting me with your stories and insights—your contributions are the heartbeat of this thesis.

Finally, to the countless individuals who supported me indirectly—teachers who ig-

nited my passion for this field, mentors who shaped my thinking, and even strangers who offered kindness during moments of stress—thank you. This milestone is not mine alone; it is a mosaic of the love, wisdom, and generosity of everyone who stood by me.

Abstract

Gaze estimation, a critical aspect of human-computer interaction (HCI), has various applications in areas such as augmented reality, virtual reality, gaming, behavior analysis, health care and assistive technology. Traditionally, eye trackers have been used within limited controlled laboratory environments, requiring the involvement of skilled professionals and complicated setup procedures. However, in recent years, scientific research on easier-to-use eye trackers that require no additional hardware—other than the standard cameras in computers, tablets, and mobile devices—has aimed to make this technology accessible to a broader audience. But these trackers have additional obstacles, such as low-resolution images produced by a standard webcam, variable background lighting conditions, individual appearance variances, changes in head pose, and many more. Recent studies in the field have concentrated on each of these shortcomings in order give better gaze estimate performances in the real-world environment. This doctoral dissertation investigates the field of gaze estimation using standard cameras, with the goal of improving affordability of gaze tracking systems. The primary objective of this study is to utilize standard webcams in order to make gaze estimation technology available to a wider audience.

The purpose of this study is to address the issue of gaze tracking using a standard webcam. Initially, a comprehensive study of the studies in the respective field was performed, carefully analyzing the pros and cons of each proposed idea. Special emphasis is placed on the challenges associated with the use of standard cameras, including lower resolution, head pose, and varying lighting conditions. In this work, appearance based gaze estimation method was used. These methods learn direct mapping from the image itself to predict the point of gaze (POG) or gaze direction. In addition, cutting-edge deep learning methodology was used in this work. Training and testing on different network architectures was performed, especially the convolutional neural network (CNN) for gaze estimation. This work involved optimizing different hyperparameters of the various networks to get the most effective model for predicting gaze. In addition, image-processing techniques were used to enhance the results. Furthermore, a person-specific model was built and shows that a carefully tuned network can outperform the model trained with multiple persons data. Moving forward, transfer learning approach with limited data environment was tested and compared with model without pretrained weights. Results show that transfer learning models perform better in limited data environment, also these mod-

els converge faster and stabilize earlier as compared to model without pre-trained weights.

Abstract

Estymacja spojrzenia, będąca kluczowym aspektem interakcji człowiek–komputer (HCI), znajduje zastosowanie w takich obszarach jak rzeczywistość rozszerzona, rzeczywistość wirtualna, gry komputerowe, analiza zachowań, opieka zdrowotna oraz technologie wspomagające. Tradycyjnie do tego celu wykorzystywano eye trackery w kontrolowanych warunkach laboratoryjnych, co wymagało udziału wykwalifikowanych specjalistów oraz skomplikowanej konfiguracji. Jednak w ostatnich latach badania naukowe skupiły się na tworzeniu prostszych w użyciu systemów śledzenia spojrzenia, które nie wymagają dodatkowego sprzętu — poza standardową kamerą wbudowaną w komputer, tablet czy urządzenie mobilne — w celu upowszechnienia tej technologii wśród szerszego grona odbiorców. Takie podejście wiąże się jednak z dodatkowymi trudnościami, takimi jak niska rozdzielczość obrazu z kamery internetowej, zmienne warunki oświetleniowe w tle, różnice w wyglądzie poszczególnych użytkowników, zmiany pozycji głowy i wiele innych. Współczesne badania w tej dziedzinie koncentrują się na analizie tych wyzwań, aby poprawić dokładność estymacji spojrzenia w warunkach rzeczywistych.

Niniejsza rozprawa doktorska podejmuje temat estymacji spojrzenia z wykorzystaniem standardowych kamer, dążąc do zwiększenia dostępności i przystępności cenowej systemów śledzenia wzroku. Głównym celem pracy jest zastosowanie standardowych kamer internetowych w celu udostępnienia technologii estymacji spojrzenia szerszemu gronu użytkowników. Celem badań było rozwiązanie problemu śledzenia spojrzenia przy użyciu standardowej kamery internetowej. Na początku przeprowadzono szczegółową analizę istniejących badań w tej dziedzinie, dokładnie oceniając zalety i wady poszczególnych podejść. Szczególną uwagę poświęcono wyzwaniom związanym z wykorzystaniem standardowych kamer, takim jak niska rozdzielczość, zmienna pozycja głowy czy zmienne warunki oświetleniowe. W niniejszej pracy zastosowano metodę estymacji spojrzenia opartą na wyglądzie (appearance-based), która polega na bezpośrednim odwzorowaniu obrazu na przewidywany punkt spojrzenia (POG) lub kierunek patrzenia.

W pracy zastosowano nowoczesne metody uczenia głębokiego. Przeprowadzono trening i testy z użyciem różnych architektur sieci neuronowych, w szczególności konwolucyjnych sieci neuronowych (CNN). Dokonano optymalizacji różnych hiperparametrów tych sieci w celu uzyskania najbardziej efektywnego modelu do przewidywania kierunku spojrzenia. Dodatkowo zastosowano techniki przetwarzania obrazu w celu poprawy wyników. Ponadto

zbudowano model dostosowany do konkretnej osoby (person-specific), który wykazał, że odpowiednio dostrojona sieć może przewyższyć model trenowany na danych pochodzących od wielu osób. W dalszej części pracy zastosowano podejście transfer learning w środowisku z ograniczoną ilością danych i porównano je z modelem trenowanym bez wstępnie wyuczonych wag. Wyniki pokazują, że modele z transfer learningiem osiągają lepsze wyniki przy ograniczonych danych, a także szybciej się uczą i stabilizują w porównaniu z modelami bez pretrenowanych wag.

Contents

Chapter 1

Introduction

In today's world, the way humans and technological systems interact depends on their ability to sense and engage with the environment. Humans depend on their sensory organs, like eyes, ears, and skin, to collect information and move through their environment. Similarly, machines rely on a range of sensors, including cameras, microphones, accelerometers, GPS, and LIDAR, to understand and react to their environment using visual, auditory, and other inputs.

Advancement in sensor technology have completely transformed our capacity to gather and analyze data from our surroundings. These technologies not only allow machines to collect information, but also give them the ability to make well-informed decisions and interact intelligently with their surroundings. For instance, cameras have a wide range of applications beyond capturing images. They are utilized in facial recognition and eye tracking, which greatly improve human-computer interaction (HCI) and the performance of autonomous systems [11]. Eye tracking, a specialized application of sensor technology, showcases its transformative potential through its ability to accurately measure and analyze gaze behavior.

Human-computer interaction (HCI) research aims to develop interfaces that replicate the natural way humans communicate. Understanding user intent and attention is greatly influenced by eye gaze, which is a vital aspect of nonverbal communication. Eye-tracking technology enables the analysis of gaze patterns, offering valuable insights into user behavior and cognitive processes. This technology has a wide range of applications in various fields such as psychology [100], marketing [17], virtual reality [112, 122], healthcare [56], communication skills [127], gaming [10] and assistive technologies [96]. Through the analysis of eye movements, researchers have the ability to create interfaces that are highly responsive to the needs of users, resulting in improved usability and an enhanced overall user experience.

Eye gaze tracking studies analyze the movement of a person's eyes. These eye movements then can be used to analyze person's attention, thinking, and mood [153]. Gaze estimation studies utilize specialized devices known as eye-trackers, which estimate where

a person is looking at or identify a point in the surrounding environment where the person gaze is directed [161]. The point of gaze can be located on a 2D plane, such as the display of an electronic device like a (laptop, desktop, tablet, or mobile screen) [23], or it can be tracked in a 3D space [105, 113], which is referred to as the point of regard (PoR) [58]. Currently, a wide variety of commercial eye trackers are available on the market, each with software specifically built for eye tracking research and practical applications [1, 126]. These commercial eye trackers provide excellent performance for marketing and scientific research, along with comprehensive documentation and support for calibration and usage.

Although commercial eye trackers have impressive performance, they encounter considerable obstacles when it comes to accessibility and scalability. Trained operators are necessary to operate them and they can be quite expensive, which limits their accessibility to the general public. There is a strong demand for cost-effective and user-friendly eye tracking solutions that can provide high performance without requiring specialized hardware or software, similar to commercial models. These solutions have the potential to make eye tracking technology more accessible to a wider range of people.

This thesis aims to investigate the potential of using common devices such as laptops, tablets, and smartphones, along with basic webcams, for eye tracking purposes, even with limited resources. We believe that a system utilizing these widespread devices has the potential to completely transform user interaction. For example, an eye-tracker could be used as a unique control mechanism similar to a mouse [81, 142]. It could also assist in analyzing how people interact with products, even from a distance [52]. If this system proves to be effective, it could potentially replace expensive eye-tracking tools, allowing for eye-tracking capabilities on mobile applications and personal computers.

Our objective is to analyze person's gaze (where a person is looking on a screen) using standard web cameras commonly found on regular computers and under natural lighting conditions. In this thesis, our focus is on laptops and desktops due to the fact that these devices generally have lower-quality web cameras compared to modern smartphones which comes with high-quality built-in cameras generally. Developing an eye tracker that can effectively operate with low-quality images presents a considerable obstacle. The accuracy of tracking a person's gaze becomes increasingly challenging when the eye is not easily visible.

The potential applications using this system could be vast, with no limits other than the imagination of developers and researchers. For instance, in educational settings, using cost-effective eye-tracking could offer fresh insights into student engagement and learning patterns. In healthcare, it has the potential to aid in the diagnosis and monitoring of neurological conditions. The potential is huge, reaching beyond our current imagination, providing a glimpse into a future where eye tracking easily fits into our everyday technology. Although this thesis primarily focuses on gaze estimation using standard webcams,

it is important to acknowledge that the potential implications and possibilities of this technology are enormous and go beyond the scope of this research.

1.1 The Objectives of Thesis

The goal of this PhD dissertation is to investigate whether a person owning a standard webcam, typically on a laptop or desktop, can perform gaze estimation on their machine without the need for any additional software or hardware. This thesis explores gaze estimation possibilities using a standard webcam because webcams generally do not provide high-resolution images (due to the low-quality camera). This makes gaze estimation where the person's eyes are not clearly visible very challenging and difficult. The requirement of having a standard camera is already met, since most commercial laptops and desktop devices come with an in-built webcam. Developing an eye tracker that can run in real-time on these devices will open up new possibilities and developments in the field of HCI.

In this thesis, state-of-the-art deep learning and computer vision techniques are used to train gaze estimation models. These latest techniques were used because they have been tested and have achieved significant success in many computer vision tasks such as object detection, object tracking, object segmentation, face recognition, and more. Furthermore, researchers have also achieved valuable results in eye tracking using deep learning techniques compared to conventional methods in recent years. Due to the success of deep learning in various fields, it has become a very popular technique, with researchers using and publishing new deep learning methods every day.

With the above-mentioned goal in mind, this research focused on examining the limitations of eye trackers and various factors, either technical or environmental, that can affect eye tracker performance. We made our first hypothesis.

Hypothesis 1 (H1) : *It is feasible to develop a CNN-based model that classifies a person's gaze into screen regions using low-resolution, low-quality eye and face images captured by a standard webcam.*

This hypothesis suggests that one can estimate a person's gaze direction on a screen by dividing the screen into different zones and using a standard web camera and convolutional neural networks (CNNs). The idea depends on the belief that deep learning techniques, particularly convolutional neural networks, have the capability to identify complex features in facial and eye images that might not be noticeable to the human eye. If this hypothesis is true, it has the potential to enhance the accessibility and affordability of gaze estimation technology by reducing dependence on expensive, specialized eye-tracking equipment. This could open up new possibilities for using gaze estimation in everyday applications and devices. We aim to answer the following questions with this hypothesis:

- Will some data pre-processing technique affect the accuracy?
- Is it possible to classify a person's gaze across different regions of the screen?

In real life, gaze estimation is a regression problem compared to classification. In a regression technique, the goal is to predict continuous values that represent the gaze point's coordinates or angles. Since humans naturally shift their gaze smoothly and continuously across different points on a screen, developing a model that can predict continuous gaze coordinates—rather than discrete regions—is more applicable to real-world scenarios.

In geneal, gaze estimation methods are trained on multiple people data, whereas training a model for an specific person requires only data from a specific user. Training a person specific model can have advantage in which people want to have more precise prediction for individual user. This leads to our second hypothesis:

Hypothesis 2 (H2): *A model trained on low-quality eye and face images from a standard webcam, tailored to a specific individual, can achieve gaze estimation accuracy comparable to that of state-of-the-art eye trackers.*

This hypothesis suggests that developing a gaze estimation model specific to an individual, as opposed to utilizing a model trained on a diverse dataset, could lead to enhanced accuracy in results. Individual variations in facial characteristics and eye movement pose challenges for generalized models, impacting their overall efficacy across different groups. Working on data from an individual allows the model to more effectively represent their distinct attributes. This methodology has the potential to enhance gaze estimation precision by utilizing only low-resolution images captured from a standard desktop webcam, thereby ensuring accessibility and ease of use across various computer systems without the need for additional hardware. If true, this could result in enhanced personalized and efficient eye-tracking solutions applicable across various domains. Furthermore, we aim to answer the following questions:

- How does the performance of a person-specific model compare to that of models trained on multi-user datasets when using low-resolution webcam images?
- What level of accuracy can a person-specific model trained on low-quality data achieve relative to models trained on multi-person datasets?

Transfer learning has gained significant attention in gaze estimation research. It is an approach where knowledge from one task is transferred to another. For example, a model trained for face classification can be used to train a model for gaze estimation. This leads to our third hypothesis:

Hypothesis 3 (H3):*Gaze estimation using low quality images collected from webcams that utilize models pre-trained with data of multiple users and fine-tuned for a specific person using transfer learning requires less data, and converge faster during fine-tuning compared to models trained from scratch for the specific person.*

Utilizing transfer learning has the potential to enhance gaze estimation by developing models tailored to individual users, particularly in scenarios where data resources are limited. Utilizing standard laptop webcams for gaze data collection presents an opportunity for enhanced accessibility and affordability in the technology. It is our hypothesis that models using transfer learning will show better performance compared to those that do not, especially regarding faster training, reduced costs, and enhanced stability when working with smaller datasets. The idea suggests that transfer learning has the potential to enhance the efficiency and usability of real-time applications in domains such as human-computer interaction, assistive technologies, and personalized user experiences. This investigation aims to answer the following questions:

- Do transfer learning models offer advantages over models trained without pre-trained weights?
- How many images are sufficient to achieve good results?

Overall, the goal of this PhD dissertation is to investigate and develop a gaze tracking methods that maximizes its ability to learn from low quality webcam data, aiming to deliver performance comparable to that of high-end eye-trackers.

1.2 Scope of Thesis

In this thesis, the first chapter delves into the introduction of gaze estimation. Chapter 2 presents theoretical foundations and basics of deep learning, specifically introducing artificial neural networks and convolutional neural network and their fundamental concepts. Chapter 3 reviews the current state-of-the-art in gaze estimation.

Chapter 4 introduces a method for gaze estimation using unmodified web camera and Convolutional Neural Networks (CNNs). In addition, we have made the dataset publicly available for use by other researchers. Parameters of the CNN were fine-tuned, and gaze prediction was treated as a classification task across different screen positions.

Chapter 5 focuses on person-specific gaze estimation from webcams. Here, we collected a new dataset and approached gaze prediction as a continuous rather than a classification problem using CNNs. Parameters of the CNN were adjusted, and results were compared with alternative methods.

Chapter 6 explores the use of transfer learning for gaze prediction and compares its performance with model without pre-trained weighs. Furthermore, we also introduce dataset for gaze estimation task from low quality webcam.

Finally, Chapter 7 concludes the thesis, summarizing the key findings and proposing directions for future research in gaze estimation.

Chapter 2

Theoretical Background

Understanding where a person is looking is a crucial focus of study in the fields of computer vision and human-computer interaction [78]. This technology has the potential to completely transform industries such as augmented reality [121], driver assistance, and user experience design. Even so, the issue is still complicated despite tremendous progress made in the last ten years.

The human eye is a complex organ, and its appearance is influenced by various factors such as head position, surrounding objects, image quality, and lighting conditions. Estimating gaze direction becomes quite a challenge due to these variables.

In addition, researchers are still having problems with fundamental questions. For example, what are the key visual cues that can indicate where a person is looking in a natural environment, without any prior knowledge? And how can these cues be effectively processed in real-time to deliver immediate results? Tackling these challenges is crucial in order to create reliable and useful gaze estimation systems.

Despite significant advancements, there remains a considerable amount of work to be undertaken in order to fully understand the complex processes of determining the direction of an individual's gaze.

This chapter presents foundational knowledge about the human eye and its functions. A comprehensive understanding of the mathematical foundation of artificial neurons is crucial in this context. The chapter dives into working of artificial neural networks (ANN), especially focusing on Convolutional Neural Networks (CNN), which are backbone of most of gaze estimation research. In addition, it highlights transfer learning, a powerful machine learning technique that allows models to leverage knowledge gained from one task to improve performance on another.

2.1 Introduction to the Human Eye and Its Movements

In order to understand the workings of eye trackers, it is crucial to have an understanding of a human eye and its structure.

The human eye is a fascinating organ that allows us to see and understand the world around us. Light is directed onto the retina, which is located at the back of the eye. The retina transforms incoming light into signals that can be processed by our brain, resulting in the formation of visual images we see. Figure 2.1 illustrates the structure of the human eye.

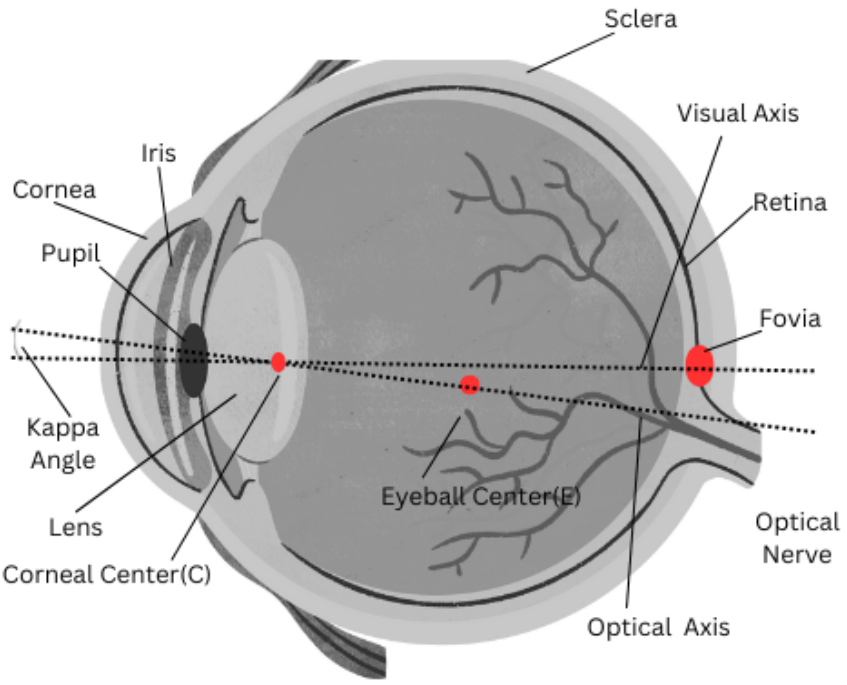


Figure 2.1: Basic structure of human eye

Our eyes are constantly in motion, allowing us to observe a variety of objects and scenes. Below are the various eye elements, each serves a distinct purpose:

- **Sclera:** The sclera is the strong, white outer covering of the eye, also known as the "white of the eye." It gives structural support and protection to the inner components of the eye.
- **Cornea:** The cornea is the transparent outermost layer of the eye, which plays an

important role in focusing light onto the retina.

- **Pupil:** The black, round hole in the middle of the eye is the pupil. It changes size to control how much light enters the eye.
- **Iris:** Surrounding the pupil is the iris, the colored portion of the eye that is visible from the outside. The iris functions similarly to a camera's diaphragm, controlling the amount of light that enters the pupil.
- **Lens:** Located behind the iris, the lens is a flexible and transparent structure that directs light to the retina. The lens changes shape to enable the eye to focus on objects at various distances, a process known as accommodation.
- **Retina:** The retina is the innermost layer of the eye, covering the inner surface of the eyeball. It is where the actual image is formed and converted into electrical signals that the brain can process.
- **Fovea:** The fovea is a small, central region of the retina that plays a critical role in high-acuity vision. It is often referred to as the fovea centralis or simply the fovea.

Figure 2.1 illustrates two axes: the optical axis passes through the center of the eyeball (E) and the center of the pupil. The visual axis, on the other hand, passes through the fovea and a point called the corneal center (C) [41]. The fovea is not on the optical line, which is why there is this difference. Another crucial aspect of human visual system is eye movement. It is known that human visual attention tends to focus on objects along the visual axis, where their images fall on the fovea. As a result, eye movements should be interpreted as shifts in focus. In general, eye movements can be broadly categorized into three types [37]:

- **Fixations:** Fixations are instances when the eyes remain relatively still, and the focus of the attention is directed towards a particular place. Fixations involve the processing of visual information and the collection of specific features. For example, engaging in reading, observing a scene, or evaluating an object involves a sequence of fixations.
- **Saccades:** Saccades are rapid, jerky eye movements that shift the gaze from one point to another. They play an important role in examining the visual surroundings, allowing individual to investigate and concentrate on specific objects or feature of interest in their visual surroundings.
- **Smooth Pursuit:** A smooth pursuit is a sort of eye movement that requires tracking a moving object in a smooth manner in order to keep one's attention on a particular thing. For example, a person is able to effortlessly follow a target, such as a moving car or a flying bird, when they have the ability to perform smooth pursuit motions.

These are the primary eye movements. However, other movements include blinking, an involuntary action that keeps the eyes moist, cool, and clean [58]. Microsaccades are tiny, involuntary movements that occur during fixations, helping to stabilize the visual scene and prevent sensory adaptation. It's crucial to identify and exclude these movements from training and testing procedures for an eye-tracking system to operate effectively.

2.2 Artificial Neuron

Artificial neurons have an input, activation, and output. When these artificial neurons are connected in layers, it forms what is called a neural network. In this setup, each artificial neuron gets input from the neuron in the previous layer and passes its output to the neuron in the next layer [14]. Figure 2.2 shows an artificial neuron. For example, when an artificial neuron is given an input with many dimensions let's say

$$X = (x_1, x_2, x_3, x_4, \ldots, x_N)$$

it calculates a weighted sum of the input signals using mathematical operations like this:

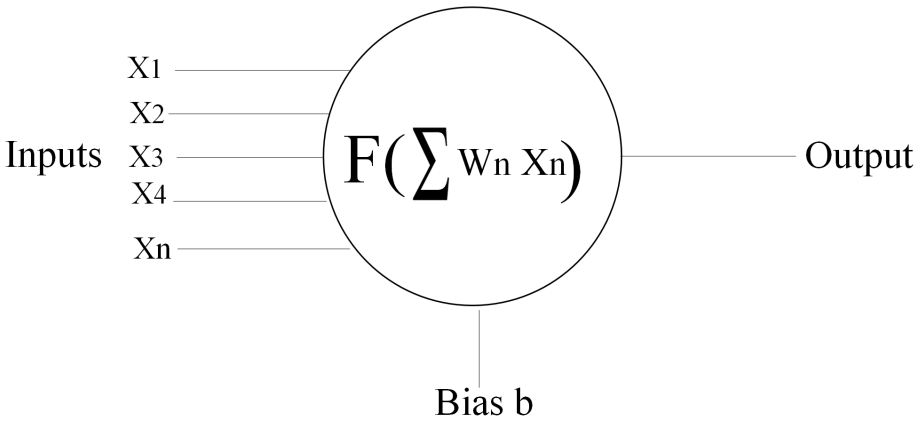


Figure 2.2: Basic structure of an artificial neuron

$$Y = f(W^T X + b) \tag{2.1}$$

The weights associated with the inputs are denoted by

$$W = (w_1, w_2, w_3, w_4, \ldots, w_N)$$

Where b represents the bias. The activation function, denoted by f, can be either linear or non-linear, although it is generally non-linear. Activation functions are applied to the weighted sum of the inputs to generate the output Y. There are several activation functions

available, but only a few have gained widespread popularity and are commonly used in artificial neurons. These popular activation functions are:

$$Y = \text{sigmoid}(z) \tag{2.2}$$

$$Y = \tanh(z) \tag{2.3}$$

$$Y = \max(0, z) \tag{2.4}$$

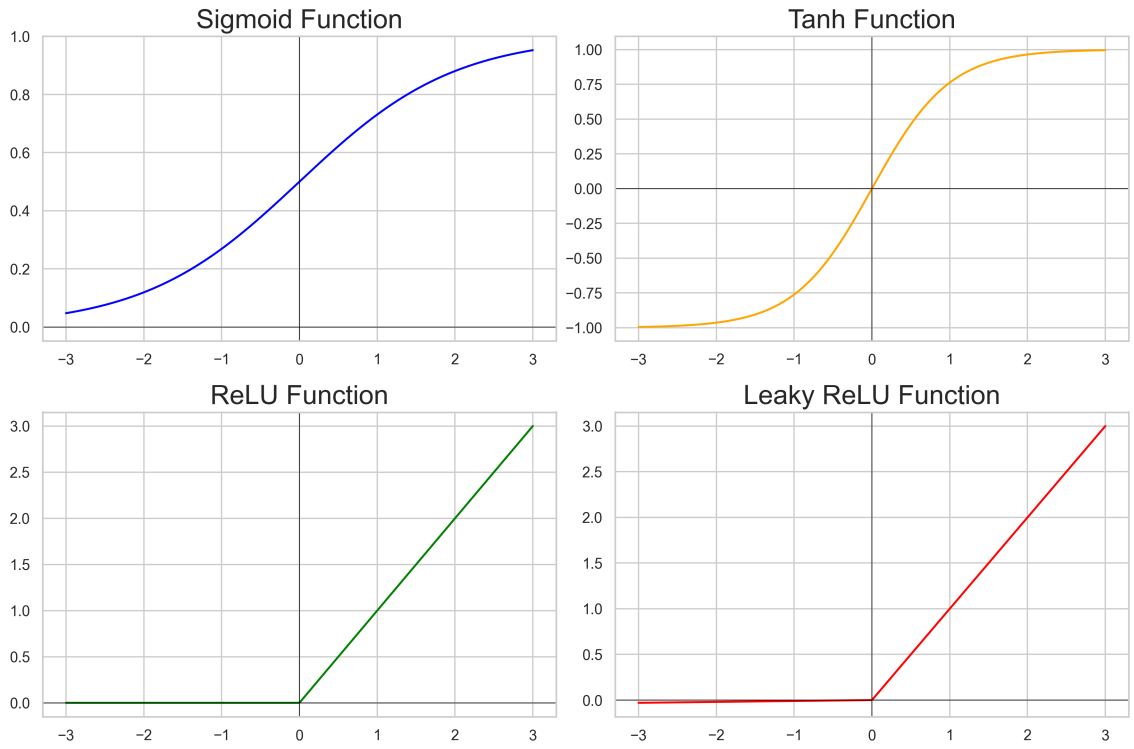


Figure 2.3: Commonly used activation functions

The z denotes the weighted sum of inputs in above equations. Activation function in equation 2.2 is known as sigmoid function, this function maps any real value number between 0 to 1. Activation function in equation 2.3 is called hyperbolic tangent function, range of this function is between -1 to 1. This function is zero cantered function unlike sigmoid. Equation 2.4 represents the rectified linear unit (ReLU) activation function. ReLU function introduce non-linearity into the model, which makes model to understand complicated pattern and correlations in the data. Because of its simplicity and efficacy, it has become one of the most commonly used activation functions in artificial neurons. Figure 2.3 shows the visual representation of commonly used activation functions.

2.3 Artificial Neural Networks

An artificial neural network (ANN) is a system made up of connected units called neurons. These networks are used in a variety of disciplines, including image processing, pattern recognition, disease and financial prediction, language comprehension, and and speech recognition.

An ANN looks like a grid of neurons connected in rows and columns. The first row is called the input layer, and the last row is the output layer. The links between neurons have values called weights, which act as adjustable settings.

In order to minimize the difference between the network's predictions and the correct responses, these weights are incrementally adjusted during the training process. This enables the network to acquire the ability to provide more precise results as time progresses.

2.3.1 Architecture of Artificial Neural Network

An artificial neural network (ANN) typically consists of an input layer, an output layer, and one or many hidden layers between the input layer and the output layer. Figure 2.4 represents the simple architecture of an artificial neural network with one hidden layer. An ANN with multiple layers is a specific type of architecture that consists of many interconnected nodes, or artificial neurons. The network consists of various layers, or neurons. There are three layers in this system: the input layer, the hidden layers, and the output layer. Each layer in the network has a weighted link between its neurons and the neurons in other layers. Three fundamental operations are involved in artificial neural networks with hidden layer.

- In the beginning, start with small values for the weights of each connection in the ANN. These weights represent how strong the connection is between neurons in the network. Positive weights indicate activation, while negative weights indicate suppression. Since the ANN operates using matrices, the weights in the ANN are defined as follows:

$$W_{i=}w_{i1}, w_{i2}, w_{i3}, \ldots, w_{iN}, \quad i = 1, 2, 3, \ldots, N$$

- Once the weights have been assigned, the next step is to calculate the weighted sum of the input." This requires multiplying each input value by its corresponding weight and then summing them up. This operation follows a linear pattern, where a bias or threshold value is included.It is defined as follows:

$$Y_j = \sum_{i=1}^{N} w_{ji} x_i + b_j$$

Where is b is the bias or threshold.

- In the third step, we introduce non-linearity into the weighted sum Y_i. Activation functions are used to achieved this. Activation functions play a crucial role in enabling ANN networks to grasp and convey complex relationships within data. Here is the definition:

$$Y_j' = f(Y_j)$$

Where f(.) is defined as the non-linear activation function.

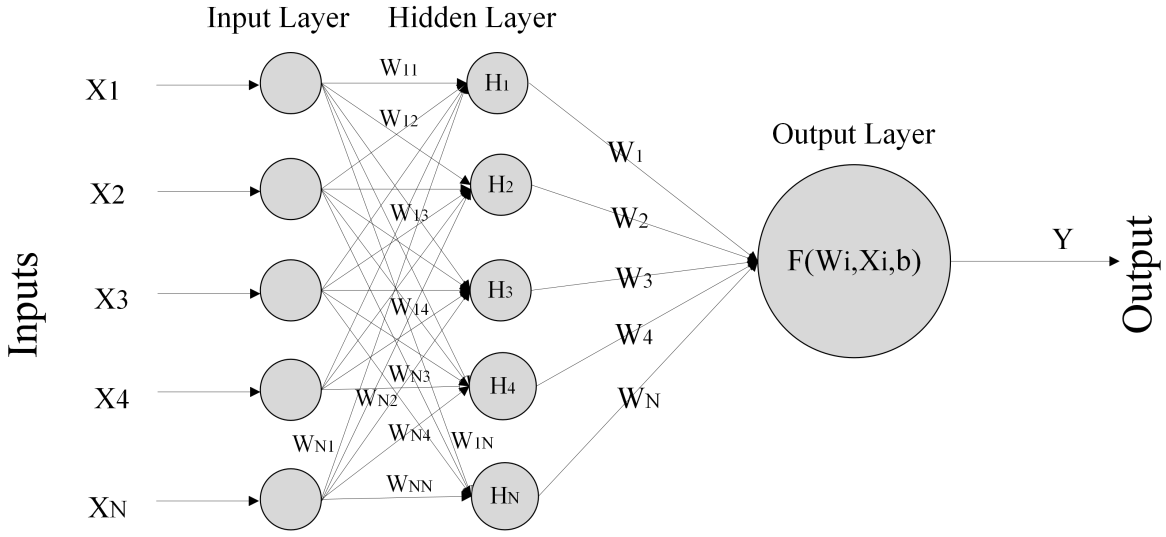


Figure 2.4: Artificial neural network with one hidden layer

There are no set guidelines to find the number of layers or neurons to include in each layer of an ANN. One can make decisions based on their personal experience, knowledge gained from reading, or through the process of experimentation with various options.Since there's no universal solution, what works best for you will depend on your specific situation.

2.3.2 Forward and Backward Propagation in ANN

The weights in a multilayer ANN can be optimized based on specific rules. The backpropagation algorithm is used to optimize these weights, helping in achieving the desired output. Before we delve into the details of the backpropagation algorithm, let's first gain a clear understanding of how the ANN performs forward propagation.

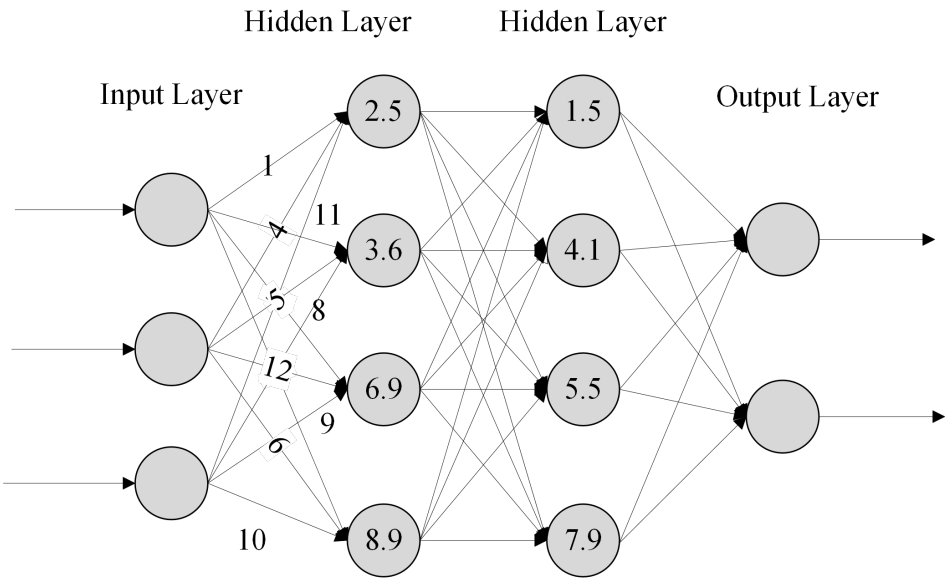


Figure 2.5: Forward pass with two hidden layer

Forward Propagation

In forward propagation, the input values are influenced by various weights and biases. As shown in Figure 2.5, when the input layer connects with the first hidden layer, each neuron is affected by its distinct set of weights and biases. The same happens when the first hidden layer connects to the second hidden layer.

Each hidden layer has different activation values due to the variability of weights and biases among connections. The values shown for the hidden states in the figure are just examples, not actual values. The activation values pass through each layer until they arrive at the output layer , as shown in Figure 2.5. The output layer then gives a final result. An ANN may have many outputs depending on the task; however (as shown in Figure 2.5, it has two outputs), and it frequently gives only one.

During the forward propagation in ANN, the information flows from the input layer to the hidden layer. For example, consider an ANN where $x_i^l$and b_i^l represent the input values and bias for the hidden unit i of layer l. Mathematically, the resulting output from the hidden layer can be calculated and defined as follows:

$$Y_i^l = f\left(\sum_j^N w_{ij}x_i + b_i^l\right)$$

To simplify, let's suppose b(bias)=x_0 and w_{i0}(weight)=1,then equation for forward propagation from layer l to layer l + 1 can be stated as follows:

$$h_i^{l+1} = f\left(\sum_{j=1}^{M_l} w_{ij}^l h_j^l\right), \quad i = 1, 2, 3, \ldots, M_l$$

Here, m_l represents the number of neurons in layer l, and w_{ij}^{l} is the matrix of weights from layer l to l+1. The output layer will finally produce the following results:

$$Y_i = f\left(\sum_{j=1}^{M_{m-1}} w_{ij}^{m-1} h_j^{m-1}\right), \quad i = 1, 2, 3, \ldots, M_0 \tag{2.5}$$

Here, M_0 represents the number of output units in the ANN. m is the total number of layers in the ANN. Y_i is the output generated from the i^{th} output unit.

Backpropagation

After obtaining the results from forward propagation, as shown in equation 2.5, next step involves applying backpropagation algorithms to adjust the weights during the training process of ANN. The objective is to ensure that the output of the ANN closely matches the output of the training sample. The backpropagation algorithm is used to obtain output values that mimic the real output, while achieving an exact match is difficult. This strategy is similar to other machine learning algorithms, in which we optimize the weights of ANN to minimize the cost function$(Y' - Y)$ through multiple training iterations. Here, Y represents the actual output, while Y' represents the expected output after forward propagation.

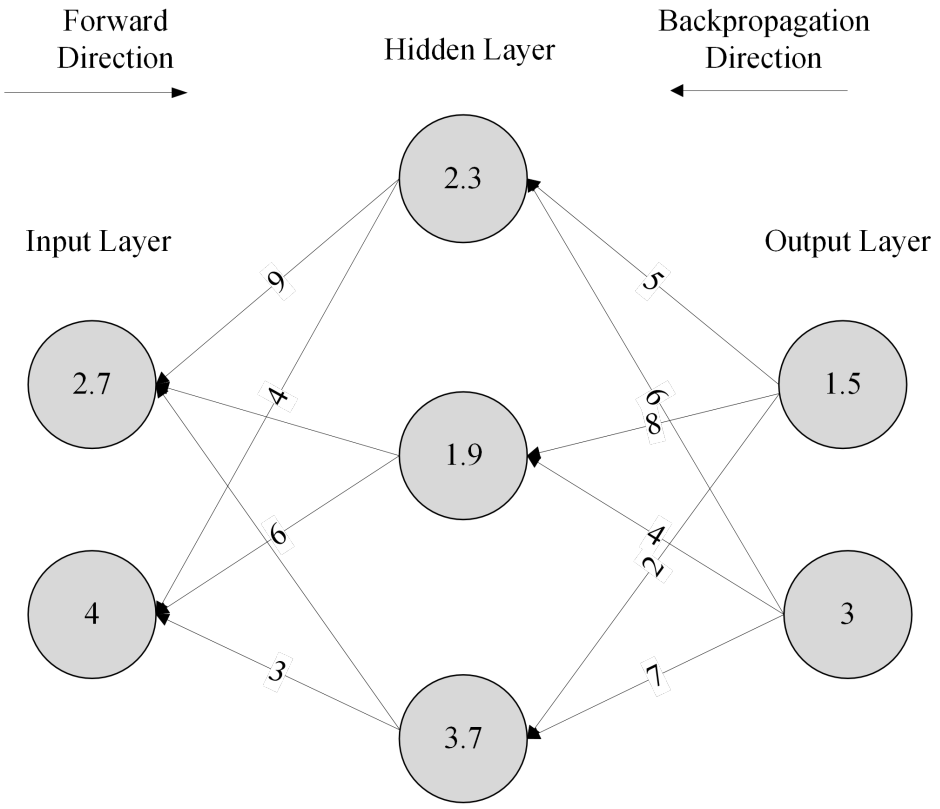


Figure 2.6: Backpropagation pass with one hidden state

The error is propagated over the entire network in accordance with the principles mentioned above in order to modify the weights and biases. The error moves from the output layer to the second hidden layer, then from the second hidden layer to the first hidden layer, and finally from the first hidden layer to the input layer, as shown in Figures 2.6. The numbers shown in the figure are just random numbers just to give an idea of how error propagated from the output layer to the input layer. These rules are applied between each pair of layers, allowing the adjustment of weights and biases for each neuron within the layer. This iterative procedure is performed for the entire network until the weights and biases are optimized, resulting in final outputs that closely approximate the actual outputs.

We can define the backpropagation error as follows:

Let's assume that Y_i^s is the output of the output unit i and the training sample s. Therefore, appropriate weights can be calculated by minimizing the error E as follows:

$$E\left(w\right) = \frac{1}{2}\sum_{i,s}\left(A_i^s - Y_i^s\right)^2 = \frac{1}{2}\sum_{i,s}\left(A_i^s - f\left(\sum_{j=1}^{M_{m-1}} w_{ij}^{m-1}h_j^{m-1}\right)\right)^2, \tag{2.6}$$

minimize $E\left(w\right)$ for $i = 1, 2, 3, \ldots, M_0$.

The variable w_{ij} represents a continuous, nonlinear, and differentiable function. To find the minimum, we adjust the weight in the direction of the negative gradient until we reach optimal values that meet the user-defined conditions. The gradient direction involves computing the partial derivative of the continuous nonlinear function. Let's assume the revised weight after t iterations is w_{ij}^t. If $\nabla E(w) \neq 0$, then the updated weight after $t+1$ iterations is found by taking the partial derivative of Equation (1.13) with respect to w_{ij}^t as follows:

$$\nabla E(w) \neq \frac{\partial E}{\partial w_{ij}^t} \tag{2.7}$$

$$w_{ij}^{t+1} = w_{ij}^t - \tau\nabla E(w_{ij}^t) \tag{2.8}$$

2.4 Convolutional Neural Network

Convolutional Neural Networks (CNN) are a special class of deep learning models; particularly well-suited for tasks involving spatial data, such as images and videos. CNNs are specifically designed to learn spatial hierarchies of features through backpropagation. Due to their feature extraction capabilities, CNN have become the backbone of many modern computer vision tasks, achieving state-of-the-art results in areas such as image recognition, object detection, and segmentation. Unlike traditional neural network, CNN use convolutional layers to automatically and adaptively learn spatial features from the

input data. This is achieved by utilizing various building blocks, including convolution layers, pooling layers, and fully connected layers.

2.4.1 Architecture of CNN

A standard CNN architecture consists of several layers each of its each serving a specific purpose and created upon mathematical concepts.

- **Input Layer:** The input layer consists of the raw pixel values of the image. For example, a 224 × 224 pixel RGB image would be represented as a tensor of shape 224 × 224 × 3.
- **Convolutional Layer:** The convolutional layer acts as the fundamental component of a CNN. Convolution operations are used to extract features from the input image by employing a set of learnable filters, also known as kernels. In mathematical terms, the convolution operation is defined as follows:

$$(I * K)(x, y) = \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} I(x + i, y + j) \cdot K(i, j) \tag{2.9}$$

  - Where:
  - The input image is denoted as I.
  - The filter (kernel) size is M ×N.
  - The coordinates (x, y) represent the current position of the filter on the input image.
  - The convolution operation is denoted by the symbol *.

  The resulting feature map shows different aspects of the input image, for example edges, textures, and patterns.

- **Activation Function:** Following the convolution operation, an activation function is applied to introduce non-linearity to model. The Rectified Linear Unit (ReLU) in equation 2.4 is widely used and highly popular.
- **Pooling Layer:** Pooling layers are useful for reducing the spatial dimensions of feature maps, which results in decrease computational load as well as helps to prevent overfitting. One of the most frequently used pooling operations is max pooling, which is defined as:

$$P(x, y) = \max\{I(x + i, y + j) \mid 0 \leq i < s, 0 \leq j < s\} \tag{2.10}$$

  - where

- $P(x, y)$ is the pooled feature map value at position (x, y).
- I is the input feature map.
- s is the size of the pooling window,for example 2×2

- **Flatten Layer:** In this layer, before sending feature map to fully connected layer, feature maps are flatten. This layer flattens the 3D feature maps into a 1D vector, preparing it for the fully connected layer.

- **Fully Connected Layer:** After a series of convolutional and pooling layers, the fully connected layers are responsible for executing the high-level reasoning. The feature maps are transformed into a vector by flattening them, and this vector is subsequently passed through one or more fully connected layers. The functioning of a fully connected layer is described by equation **??**

- **Output Layer:** The output layer can vary depending on the problem statement. When dealing with a classification task, it is common practice to utilize a classification layer. If we are dealing with a regression task, it is suitable to use a regression layer in the CNN.

  In classification tasks, the output layer usually uses a softmax activation function to generate probability distributions across the output classes. The operation of a output layer can be expressed as:

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \tag{2.11}$$

  - where:
  - $\sigma(z)_j$ represents the probability of the j-th class given the input z.
  - z_j is the score (logit) for the j-th class
  - K is the total number of classes.

  The output of a regression layer can be expressed as equation 2.4.1.

$$\hat{y} = f(x; \theta) = W \cdot h + b \tag{2.12}$$

  - where:
  - $\hat{y}$: predicted output (regression result)
  - x: input to the network
  - $f(x; \theta)$: function modeled by the neural network with parameters θ
  - h: output from the penultimate layer (features before regression layer)

- W: weights of the regression layer
- b: bias term

2.5 Transfer Learning

Transfer learning is a powerful machine learning technique that allows models to leverage knowledge gained from one task to improve performance on another task. It involves reusing a pre-trained model as a starting point for a new model, saving time and resources. It utilizes the knowledge acquired from solving one problem and applies it to another problem that is related but different. This method is especially useful when dealing with limited data in the target domain by utilizing the rich, learned representations from the source domain.

2.5.1 Problem Formulation

Let's consider $\mathcal{D}_S = \{(x_i^S, y_i^S)\}_{i=1}^{n_S}$ is the dataset from the source domain, where $x_i^S \in \mathcal{X}_S$ and $y_i^S \in \mathcal{Y}_S$ represent the input and output pairs, respectively. Similarly, let's consider $\mathcal{D}_T = \{(x_i^T, y_i^T)\}_{i=1}^{n_T}$ is the dataset from the target domain, where $x_i^T \in \mathcal{X}_T$ and $y_i^T \in \mathcal{Y}_T$.

In transfer learning, the goal is to improve the learning of the target predictive function $f_T : \mathcal{X}_T \to \mathcal{Y}_T$ by leveraging knowledge from the source domain.

2.5.2 Types of Transfer Learning

Transfer learning can be divided based on their relationship between source and target domain.

- **Inductive Transfer Learning:** Inductive transfer learning is useful when target domain is different from source domain. However, this technique is effective when there is large amount of data is available for the source task.
- **Transductive Transfer Learning:** This technique is useful when target task same as source task but the data distribution is different.
- **Unsupervised Transfer Learning:** This technique is useful when both target and source task are different and no label data is available for target task.

2.5.3 Transfer Learning with Convolutional Neural Networks (CNN)

Transfer learning is widely used in computer vision tasks, including image classification, object detection, and segmentation. Due to its feature extraction ability CNN have

been become the first choice for transfer learning approach. Below are the simple steps how CNN works.

Working of CNN

Step 1: Load a Pre-Trained Model: Utilize a CNN that has been pre-trained on a large dataset, such as ImageNet. ImageNet is an extensive collection of millions of images spanning thousands of categories. Some well-known pre-trained CNN architectures are VGG, ResNet, Inception, and MobileNet.

Step 2: Modify the Model

- **Freeze Layers:** In order to ensure that the early layers of the pre-trained model remain unchanged during training, it is recommended to freeze their weights. These layers include common features such as edges and textures which have practical use in various tasks.
- **Replace the Final Layer:** Update the last fully connected layer(s) that match the number of classes in the task you want to perform.

Step 3: Train the Model on the Target Dataset

- **Feature Extraction:** Only train the new layers on the target dataset; do not change the layers.
- **Fine-Tuning:** There is the possibility of unfreezing some of the later layers of the pre-trained model and then training them on the target dataset alongside the newly added layers for training purposes.

Looking into Mathematical Insights

Pre-trained Model: Let's consider θ_S as the parameters of the pre-trained model f_S. The model goes through training using a source dataset $\mathcal{D}_S = \{(x_i^S, y_i^S)\}_{i=1}^{n_S}$ in order to minimize the loss function L_S:

$$\theta_S = \arg\min_{\theta} \frac{1}{n_S} \sum_{i=1}^{n_S} L_S(f(x_i^S; \theta), y_i^S)$$

Feature Extraction: For the target task, utilize the pre-trained model as a feature extractor. Representing the features extracted by the pre-trained CNN is done using $g(x; \theta_S)$. Next, include extra layers of $h(x; \theta_T)$ to effectively map these features to the desired target classes. The model can be represented as follows:

$$f_T(x) = h(g(x; \theta_S); \theta_T)$$

Training the new layers θ_T requires utilizing the target dataset $\mathcal{D}_T = \{(x_i^T, y_i^T)\}_{i=1}^{n_T}$:

$$\theta_T = \arg\min_{\theta} \frac{1}{n_T} \sum_{i=1}^{n_T} L_T(f_T(x_i^T; \theta), y_i^T)$$

Transfer Learning in Gaze Estimation

Transfer learning can be effective in gaze estimation tasks, where the goal is to determine where a person is looking based on images of their eyes or face. Given the limited availability of large annotated datasets specific to gaze estimation, transfer learning helps leverage pre-trained models on related tasks (e.g., face recognition, eye detection) to improve performance.

Steps in Transfer Learning for Gaze Estimation

- **Pre-Trained Model Selection:** Select a pre-trained CNN model that has been trained on a huge dataset specifically designed for face or eye detection. Popular models include VGG, ResNet, and Inception and so on. These models are often pre-trained on datasets like ImageNet or face-specific datasets such as VGGFace.
- **Feature Extraction:** Utilize the pre-trained model to extract features from eye or face images. This step utilizes the acquired knowledge from the source task.
- **Fine-Tuning:** Refine the pre-trained model using the gaze estimation dataset. One approach is to retrain the later layers of the model on the new dataset while keeping the early layers fixed or changing their learning rate. This helps to maintain the learned features.

Example workflow for Gaze Estimation

Source Task: Face Recognition

- Pre-trained Model: Consider utilizing a Convolutional Neural Network (CNN) such as ResNet50 that has been pre-trained on either the ImageNet dataset or a face recognition dataset.
- The source dataset includes face images that are labeled with identity information.

Target Task: Gaze Estimation

- Includes images of eyes or faces along with labels indicating the direction of gaze.

Chapter 3

Methods for Gaze Estimation: State of the Art

This chapter walks through the state-of-the-art methods in gaze estimation and presents a comprehensive overview. Furthermore, it examines various approaches to framing the problem and explains the technical terms used in this field. Additionally, this chapter provides an overview of publicly available data and discusses the applications of gaze estimation in various domains. The rest of the chapter is organized as follows: it begins with a review of gaze estimation methods, covering both conventional and the latest techniques. This is followed by a discussion of the currently available publicly accessible datasets. Next, the chapter presents real-world use cases of gaze estimation.

3.1 Introduction

Gaze estimation has a long history, and it has been studied for decades. Due to this, gaze estimation has established itself in several disciplines. The concept originated in computer vision-related assistive technology [18, 30], and spread to Human Computer Interaction (HCI) [78], behavior analysis [152], Augmented Reality (AR) and Virtual Reality(VR) [121, 12], egocentric vision [124], biometric systems [75], health care [56] and other fields [38, 103]. Figure 3.1 provides a concise timeline of the groundbreaking gaze estimation approaches together with significant achievements. Although this only includes the most frequently mentioned events in the chronology, there is a long and illustrious history further down the lane. Back in 1879, first gaze estimation experiment was conducted [76]. Later on, its necessity in human visual attention modeling pushed the research in gaze estimation forward. In 1908, first eye tracker was produced. Furthermore, Purkinje images [31], Bright Pupil [102] and infrared (IR) [2] based eye tracker were developed. But these eye trackers are expensive and require specific laboratory settings which limit their use in real world environment.

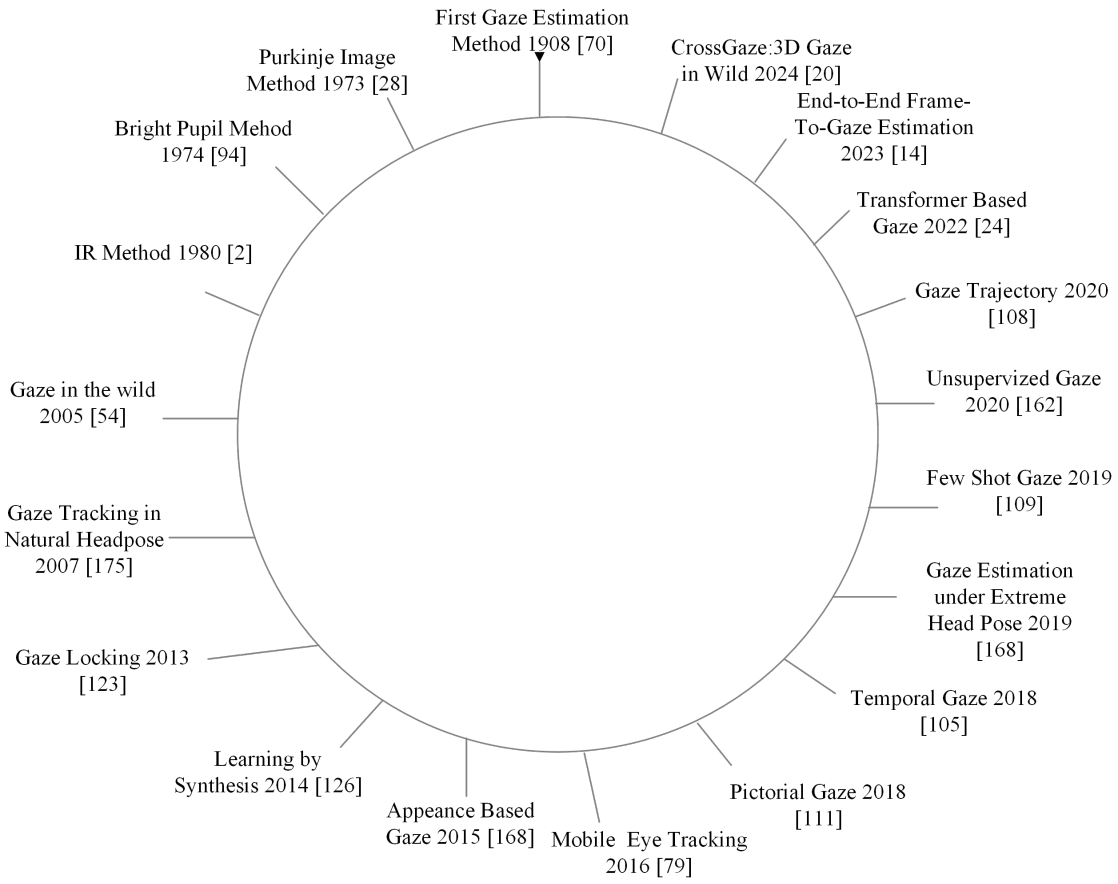


Figure 3.1: Gaze estimation research timeline

Most traditional gaze estimating methods use handmade low-level features such as color [58], shape [58, 60], appearance [131] and specific geometrical heuristics [134] to handle generic unconstrained settings [134, 183] in order to overcome their limits. In 2015, a paradigm shift based on deep learning has been observed in gaze estimation [176, 119, 85, 14, 20], with other computer vision task. Over the course of the past few years, the difficulties that are connected with variations in illumination conditions, camera setup, eye-head interaction, and other related factors have been significantly decreased thanks to the availability of massive training datasets and models based on deep learning. Despite this, the performance gain comes at the expense of large-scale annotated data, which is a costly resource to acquire. In recent times, there has been a growing interest in deep learning with minimal annotation [118, 36, 170].

In present time, most common techniques for eye tracking involve an infrared light (IR) source and high-quality cameras. There are numerous commercial models for eye tracking available, most of them either in the form of eye wear [138, 71] or table-mounted devices [137, 139, 128]. Furthermore, there is also open-source software that allows end-users to use custom hardware [5]. On the contrary, webcam-based eye trackers do not require any specialized hardware; they make use of a standard webcam and natural light for human gaze estimation. This thesis emphasizes eye tracking methods that can be

utilized on a desktop or laptop with a standard webcam. The following chapter delves into an examination of the findings by researchers in this field.

3.2 Methods for Gaze Estimation

Categorizing gaze estimation methods can be quite challenging because of the lack of clear distinctions between different approaches and the inconsistent use of terminology within the research community [41]. A well-known survey that provides a classification framework for these methods, dividing them into three main categories: 2D regression-based, 3D model-based, and appearance-based [58]. This study utilizes a classification system and examines three main methodologies: model-based, feature-based, and appearance-based techniques. Although it can be helpful to simplify things for the purpose of comparison and analysis, it's worth noting that the lines between these categories are often blurry, making it challenging to establish a clear classification [41].

- **Model based method:** Model-based gaze methods use parameters of a 2D or 3D eye model as input. When working in 2D, the iris edge model can be defined by these parameters [147]. However, in 3D, additional information such as the 3D position of the eyeball center [23] and other face landmarks [73] can be included.
- **Feature Based Method:** In the feature-based approach, researchers focus on identifying and utilizing special features associated with the eyes or face in order to make predictions about where a person is looking. These features may consist of the pixel positions of various key points such as the inner eye corner, iris center, eye lid, and pupil position [13, 57]. Computer vision features like histogram of oriented gradients (HOG) and local binary patterns (LBP) are commonly used in the field [33, 99, 90, 109]. Some researcher also used technique like feature grouping and summarize pixel intensities [42, 92, 94, 167]
- **Appearance Based Method:** The appearance-based method focuses on analyzing the visual appearance of the eyes, face, or relevant regions in the image or video frames. Pixel intensities are utilized to generate a map for gaze estimation [41].

The following sub sections provide an in-depth analysis of recent advancements in each of these areas.

3.2.1 Model based Gaze Estimation

In general, model-based gaze estimation methods use a geometric 3D eyeball model as the fundamental basis for determining gaze direction. The primary goal of these methods is to recognize eye features from 2D images to accurately estimate the model's parameters.

Furthermore, model-based methods can be divided into two approaches: the iris contour model and the eyeball model [41]. The iris contour model, also known as the one-circle model, fits an ellipse around the iris region. In eyeball model, the objective is to find the eye center [41]. Eyeball model methods work by locating center of the eyeball and then drawing a line from there to the center of the iris to determine where a person is looking.

Applying the direct least squares method to fit ellipses to a set of points [45] has greatly contributed to the progress of iris contour models in the area of gaze estimation. By leveraging this approach and understanding that the circular shape of the iris can sometimes appear as an ellipse in camera images, researchers have been able to develop various gaze tracking techniques. In [147, 149], the authors presented an iris contour algorithm that utilizes edge detection to identify the pixels comprising the iris boundary. Next, an ellipse is fitted to these points. Afterwards, the ellipse is projected back into 3D space in order to determine the contour circle of the iris. The normal vector of the iris is utilized as the gaze direction. Their system generally has an average error of approximately 1 degree.

Wu et al. proposed two circle algorithms. They suggested that the elliptical iris contours for both eyes are either on the same plane or on parallel planes in 3D. Based on this assumption, they can estimate the gaze direction without requiring camera calibration. Fukuda et al. [46] presented subpixel approaches for precisely predicting the iris contour in low-resolution images. Using their approach, they achieved an average error of 3.35 degrees. The authors in [104] employed a support vector machine (SVM) to remove irrelevant edge segments before fitting ellipses, resulting in an accuracy of 3.48 degrees. Huang et al. [70] utilized a randomized Hough transformation to fit iris contours. Moreover, Zhang et al. [172] suggested an enhanced RANSAC algorithm, which achieved an accuracy of 0.8 degrees in one direction.

Ishikawa et al. [73] use an Active Appearance Model (AAM) to track the face. They calculate anatomical constants, such as eye geometry, using the user's face size and the locations of the eye corners. After this calibration, they employ template matching to identify the iris and edge-based techniques to fine-tune its position to determine the iris center. Their method achieves an average error of 3.2 degrees. Xie and Lin [164] use a simple one-target calibration to find the eyeball center position and other specific parameters, achieving a 2-degree accuracy in one direction by using the iris center and eye corner coordinates in the image to calculate the gaze geometrically.

Chen and Ji [23] use a basic face model that includes several facial reference points, such as the nostrils, the inner and outer corners of the eyes, and one of the eyeball centers. They track these face points and use 3D modeling to calculate gaze accuracy, achieving an error of 2.7 degrees. Wang et al. [150] explicitly fit a deformable eye-face model over time, conducting a combined optimization of person-specific eye position and visual axis offset parameters. Wood et al. [157] immediately apply the morphable model to the eye

region to optimize both appearance and illumination. This methodology not only provides an estimation of gaze direction but also enables the redirection of gaze, as demonstrated in their later investigation [158].

Yamazoe et al.[169, 169] divide the pixels in the eye image into three categories: skin, sclera, and iris. They then use these segmented areas to determine the most likely eye position by minimizing projection errors for a given candidate. They reported an accuracy of approximately 9 degrees. Xiong et al. [165] customize a face model for each individual and manually designate the position of the eyeball. They then project the identified coordinates of the iris center onto the pre-established 3D model. Wu et al. [162] use a particle filter (PF) to monitor the contours of the iris and eyelids, employing various appearance metrics to calculate the probability of a specific particle (candidate). Their experimental findings achieved an average error of 3.5 degrees.

3.2.2 Feature based Gaze Estimation

In a typical computer vision process, the system first identifies features in an input image. Algorithms then use these features to estimate more complex aspects. For gaze estimation, methods like the ones mentioned above focus on finding features such as the center of the iris or the area around the edge of the iris. Hansen and Ji [58] classify these methods based on this feature detection. However, there's an important distinction. When methods use these features directly for estimating gaze, instead of fitting them into a predefined geometric model, they are termed "feature-based gaze estimation" methods [173].

In the presence of infrared light, a common method is to examine the difference between the computer's estimation of the center of pupil (PC) and the reflection of the infrared light off cornea, known as corneal reflection [41]. This difference, called the PC-CR vector, can often accurately indicate where you're looking, provided your head is facing the screen and remains still. Cerrolaza et al. evaluate and validate the effectiveness of these methods [21].

The authors in [57, 59] used an Active Appearance Model (AAM) and mean-shift techniques to track the movement of the eyes and determine the central position of the pupil and the corners of the eye. Additionally, they employed a Gaussian Process (GP) to estimate gaze by training it with the pupil center and eye corneal vector (PC-EC vector) as its input. They achieved an accuracy of 1.6 degrees and validated the accuracy of the eye tracker by testing their system within an eye-typing interface. Zhu et al. [181] proposed a set of techniques aimed at accurately detecting the center of the iris and the eye corner with subpixel precision. The researchers utilized a two-dimensional linear mapping technique to estimate the gaze positions based on the feature vectors. Their experiments demonstrated an accuracy of approximately 1.2 degrees.

In their study, Huang et al. [68] successfully identified corneal reflection points, also known as glints, from the display of a mobile phone. These glints were used to extract relevant features, which were then analyzed using Gaussian Processes. Although the effectiveness of this approach has been demonstrated, there may be challenges when generalizing the findings to outdoor environments and varying head-to-screen distances. These factors could impact the applicability of the results.

Valenti et al. [144, 145] present a novel approach for identifying the positions of eye corners. This method is integrated with a state-of-the-art eye center locator to calculate the EC-PC vector. Inspired by the work of Zhu et al. [181], they utilized a 2D linear mapping for gaze estimation. In their later work [143], the researchers integrated a head pose estimator and used the resulting transformation matrix to normalize the eye regions. The eye location information was enhanced and later used to improve the accuracy of head pose estimation through a feedback loop. To address the issue of gaze estimation during head movements, they proposed a method where known calibration points are dynamically adjusted to align with the monitor coordinates whenever a change in head pose is detected. This allows the system to recalibrate itself, ensuring accurate gaze estimation even with head movements. With these improvements, the researchers were able to achieve an average error range of 2 to 5 degrees in two different experimental tasks.

Cheung et al. [28] employed Active Shape Models (ASM) to analyze pre-processed images normalized using local sensitive histograms. By implementing novel methodologies aimed at pinpointing the exact location of the iris center and eye corners, the researchers achieved a remarkably low margin of error. Specifically, under controlled conditions with the subject's head stationary, the average error was as low as 1.28 degrees. However, with the introduction of head movements, the margin of error increased slightly to 2.27 degrees. Ince and Kim [72] utilized a specialized technique to track iris movement by quantifying the displacement of the iris center across successive camera frames, enabling them to determine gaze direction. Their system demonstrated a commendable accuracy, achieving a precision of 3.23 degrees for both horizontal and vertical measurements.

Similarly, Nguyen et al. [108] used a comparable methodology, leveraging the center-bias effect, which suggests that gaze distribution tends to concentrate towards the screen's center[79]. Their system operates without calibration by continuously calculating the average iris center position over time. Using the difference between the current and average positions, the system estimates gaze direction, resulting in a cumulative error of 3.43 degrees in both horizontal and vertical dimensions.

Lu et al. [94, 93] proposed extracting either 8-dimensional or 15-dimensional intensity features from the eye region, resizing the grayscale eye image to 2×4 or 3×5 pixels, respectively. They introduced a novel subpixel alignment method for accurate gaze point estimation in the eye region and employed adaptive linear regression (ALR) to enhance gaze estimation accuracy. Their approach achieved an impressive accuracy of up to 0.62

degrees. Skodras et al. [130] monitored various anchor points, including eye corners, eyelid control points, and the iris center, during both motion and stationary conditions. They computed vectors based on the relative positions of these points and used linear regression to map these vectors effectively to the gaze point, resulting in an accuracy of 2.96 degrees.

3.2.3 Appearance based Gaze Estimation

In contrast to other methods that rely on models or features, appearance-based approaches utilize the image pixels as the input data. These methods utilize the pixel intensities from eye images to estimate gaze. Appearance-based methods bypass the need to detect eye feature points by directly mapping eye images to gaze targets. Appearance-based methods have the advantage of not relying on explicit shape extraction stages, allowing them to effectively handle low-resolution images. An initial exploration in this field was undertaken by Baluja and Pomerleau [16]. They used a neural network to directly calculate the precise location of on-screen gaze in pixels. In addition, they show that training the system with inputs from different head poses enables it to even accommodate slight head movements.

Tan et al. proposed local linear interpolation for gaze estimation [136]. In their study, they used the nearest-neighbor method, which relies on pixel similarity. The method involves utilizing hundreds of labeled samples along with their corresponding labels. These samples are weighted to generate screen coordinate estimates.

One significant improvement was observed when a large amount of training data was collected using a multi-view camera system and then synthesized using Structure-from-Motion [134]. Ono et al.[111] conducted a comprehensive analysis of eye images, considering the effects of gaze direction, eye appearance, and image cropping shifts. They used a decomposition technique to identify the three most similar training samples. Subsequently, they utilized Likelihood-based Linear Interpolation (LLI) to enhance the accuracy of gaze estimation. Their method achieved an impressive accuracy of 2.4 degrees.

Lu et al. [92] examined the effect of head pose on gaze bias and how it affects the accuracy of gaze estimation. They discovered that the final gaze estimate results can be enhanced by compensating for this bias. In another study, authors also presented a technique for generating synthetic eye images based on a specific head pose [95]. They proposed incorporating extra calibration samples to enhance the precision of the generated images. Sugano et.al [135] utilized a technique called LLI (Limited Latency Interaction) to consider the impact of head movements on the experimental tasks. The eye images are organized into clusters based on the corresponding head pose. For interpolation, samples are selected only from clusters that have the same head pose as the current sample. The system used in this study adopts an iterative learning approach, where the authors gathered knowledge from user interaction through mouse clicks. This process enables the system

to constantly improve its parameters and adjust to new head poses by incorporating necessary clusters as needed.

Previous studies have also used the Kinect sensor for appearance-based gaze estimation. This is primarily because the Kinect sensor offers valuable depth information, which is crucial for accurately estimating the pose of the head [29, 50]. Funes Mora and Odobez [47] used the Kinect as a capture sensor to address the issue of head pose variations. To overcome this challenge, they implemented a technique that involved rotating the face to a frontal position. The technique of frontolyzing faces has been further adapted to be applicable to standard RGB cameras. However, its implementation requires accurate detection of facial landmarks, as found by Jeni and Cohn in their study[77].

Alnajar et al. [7] propose a calibration-free method for gaze estimation. Their method is based on the assumption that individuals tend to exhibit similar gaze patterns when presented with identical stimuli. This approach eliminates the need for time-consuming and potentially biased calibration procedures, making it a promising alternative for gaze estimation in various applications. Initially, the researchers compute the initial gaze points for a user without calibration. Then, a transformation is generated to align the user's gaze pattern with that of other users. For initial gaze estimation, authors commonly use two approaches. The first approach involves reconstructing the current eye appearance by leveraging nearest neighbors from the training set, which includes samples from other users. The second approach projects the eye appearance onto a 2D manifold to identify the most similar samples.

Lai et al. [87] used the Random Forest algorithm to explore the neighborhood structure concerning joint head pose and eye appearance features (HPEA). The gaze estimation process involves using linear interpolation techniques that leverage neighboring data points within the random forest. The authors introduced a system that uses a Viola-Jones eye detector and optical flow (OF) technique to detect and monitor the eye in camera images [106, 107]. By doing the integration of these two methods, the authors successfully identify and monitor the eye within the camera image, a vital process in a range of applications including gaze tracking and human-computer interaction.

Once the eye image has been extracted, it is inputted into a Gaussian Process (GP) model to accurately determine the location of the gaze point. In their study, the authors show that by calibrating the system with various head poses multiple times, they are able to achieve head pose invariance. Sugano et al. utilize saliency information to automatically calibrate a gaze tracker [133]. This calibration takes place while the individual is watching a video clip. Throughout the process, the authors choose a distinct approach. Instead of relying on predetermined gaze positions, the researchers train the GP by utilizing gaze probability maps that are created by combining various saliency maps.

3.3 Deep Learning based Gaze Estimation Methods

Advancements in gaze estimation research have made significant progress thanks to the accessibility of large-scale datasets and the use of state-of-the-art methodologies inspired by various areas in Computer Vision[53]. Deep learning techniques have greatly accelerated the progress of gaze estimation, resulting in the development of several innovative approaches. In this section, we aim to provide an in-depth exploration of gaze estimation techniques that harness the power of deep learning methodologies, highlighting their potential and recent advancements in achieving more accurate and robust gaze estimation systems that have captured the interest of researchers in the field. Our contribution is well-timed, closely aligned with the current surge of interest in gaze estimation research. In this section, we aim to provide an in-depth exploration of gaze estimation techniques that harness the power of deep learning methodologies, highlighting their potential and recent advancements in achieving more accurate and robust gaze estimation systems.

3.3.1 Convolution Neural Network based Methods

Numerous approaches have adopted convolutional neural network (CNN)-based architectures [176, 178, 119, 177, 85], focusing on obtaining an end-to-end spatial representation to predict gaze accurately. These models typically derive from well-established CNN architectures in computer vision, such as AlexNet [86], VGG [129], ResNet [62], and DenseNet [67].

Generally, these CNN methods are categorized as single-stream [176], multi-stream [85], and prior-based networks [119]. Single-stream CNNs are trained using a single RGB image stream, often a face or a patch around the left or right eye. In contrast, multi-stream CNNs utilize multiple streams, such as both face and eye patches, integrating prior knowledge from eye anatomy or geometrical constraints.

The pioneering work on deep learning-based gaze estimation was introduced by Xucong et al. [176]. They developed CNN architecture that utilizes grayscale eye images as input to effectively predict gaze vectors. They used a LeNet-based Convolutional Neural Network (CNN) architecture consisting of five convolutional layers and two fully connected layers. Additionally, a 13-layer CNN network derived from the VGG network was utilized.

In another study, X Zhang et al. introduced the concept of full-face based gaze estimation [177]. They used a spatial mechanism to identify significant facial feature locations using a CNN architecture. This spatial mechanism includes three 1×1 convolutional layers followed by rectified linear unit (ReLU) activation. A spatial weight module learns the weight matrix by element-wise multiplication with the original activation function, allowing the model to assign greater importance to specific regions and reduce noise impact in input images. The authors evaluated accuracy using the l1 distance metric for both 2D and 3D gaze estimation tasks. For 2D gaze estimation in screen coordinates, l1 distance

measured the difference between predicted and ground truth values. Similarly, for 3D gaze estimation, l1 distance computed differences in predicted and ground truth angle vectors.

Neeru Dubey et al. proposed Ize-Net for coarse to fine gaze representation [67]. Their methodology partitions recorded gaze locations into distinct "gaze zones" and integrates convolutional and primary capsule layers in the network architecture. In a separate study, researchers introduced dilated convolutional layers [24], which maintain spatial resolution of inputs while expanding the receptive field size without increasing model parameters. Dilated convolutions are applied to process facial and eye regions before inferring gaze direction. The authors used cross-entropy loss in the label space as part of their approach.

The authors introduced a pictorial gaze network for gaze estimation, which transforms eye images into gazemaps and predicts gaze direction from these gazemaps [119]. Gazemaps serve as an intermediate model designed to mimic the structure and function of the human eyeball and iris. The gaze prediction process is divided into two main phases. Initially, gazemaps are predicted using a 3-module hourglass network. Cross-entropy loss is used between predicted and ground truth gazemaps across all pixels in the dataset. In the subsequent phase of the study, DenseNet, a deep learning architecture, is utilized to accurately forecast the gaze vector based on information from the gazemaps.

In a related approach, Krafka et al. proposed a network that takes input from left eye, right eye, facial landmarks, and face images to predict the person's gaze on the screen [85]. Similarly, in another study, authors used a two-stream VGG network for gaze inference [43], using left and right eye images as input data. The neural network was trained using l2 loss function between predicted and ground truth values.

3.3.2 Temporal-based Methods

Since human gaze involves dynamic eye movements like fixation, saccades, smooth pursuits, blinks, and micro movements, the image frame over time correlates highly with gaze direction from previous time steps [54]. Consequently, several studies have incorporated eye movements and temporal information into gaze estimation to improve accuracy. The objective of these models is to predict a person's gaze based on sequences of frames. Researchers have utilized various state-of-the-art recurrent neural networks for this purpose, including LSTM [80] and GRU [115].

Palermo et al. proposed a multimodal recurrent CNN for 3D gaze direction [113]. Their model initially learns static features from all input frames in a sequence, followed by feeding these features into a many-to-one recurrent network to predict 3D gaze direction for the last frame of the sequence. Zhou et al. introduced a bidirectional recurrent network for temporal-based gaze estimation [180]. Bidirectional networks leverage both past and future frames to enhance prediction accuracy, particularly effective for low- to mid-quality images.

Another innovative approach, the Pinball LSTM, was introduced to integrate contextual and temporal information [80]. This method uses a bidirectional LSTM network for video-based gaze estimation, utilizing a sequence of 7 frames to predict gaze. The authors utilized pinball loss to measure the difference between predicted and ground truth values.

Wang et al. introduced the Dynamic Gaze Transition Network (DGTN) as an innovative model that utilizes a semi-Markov model to capture the dynamics of human eye movements[151]. Their model aims to provide a comprehensive understanding of the mechanisms governing gaze transitions. Initially, the network computes per-frame gaze using a Convolutional Neural Network (CNN), which is then refined by incorporating learned dynamic information.

Lemei Xiao et al. proposed the LSTM-CVFAF method, which focuses on predicting gaze direction in dynamic videos by integrating LSTM with convolution and associated video frame features [163]. This approach effectively captures spatial details from each video frame by incorporating learnable central prior knowledge. LSTM analyzes temporal changes in gaze behavior, facilitating an understanding of gaze movements over time. By combining spatial and temporal motion information, the method generates gaze prediction maps for dynamic videos, thereby improving accuracy in predicting the subject's gaze location throughout the video.

3.3.3 Transformer based Methods

The recent surge in interest in transformer models can be attributed to their exceptional performance across various computer vision tasks [54]. Within the realm of gaze estimation, two distinct categories of transformers have emerged, both based on the ViT (Vision Transformer) framework [54]. These transformer-based models for gaze estimation include GazeTR-Pure [26] and GazeTR-Hybrid [26]. The former is a pure transformer model, while the latter integrates both CNN and Transformer components.

Gaze estimation poses a challenge as a regression task, particularly in perceiving gaze solely through local patch-based correlations. Although these transformer-based approaches represent initial explorations, significant performance enhancements have not yet been widely documented in current literature. Further investigation is necessary to fully explore the potential of transformer architectures in improving gaze estimation, despite their promising capabilities.

The Disentangling Transforming Encoder-Decoder (DT-ED) framework, introduced by Seonwook et al., uses an encoder network to map input images into a latent space [117]. Within this space, the DT-ED model effectively disentangles three crucial factors essential for gaze analysis: gaze direction, head orientation, and visual characteristics of the eye region. This disentanglement is achieved by incorporating constraints related to gaze and head pose rotations. Additionally, a decoder reconstructs the transformed image

back to its original form, facilitating the estimation of gaze direction from the latent embedding.

In another study authors introduced a Self-Transforming Encoder-Decoder (ST-ED) network designed to process a pair of images, x_i and x_t, as inputs [179]. This network aims to separate the subject's invariant embeddings while incorporating pseudo-label conditions and embedding representations. The learning objective for transformation utilizes pseudo condition labels to accommodate additional factors when ground truth annotations are unavailable.

Aayush et al. proposed RITnet [22], a hybrid model that combines U-Net and DenseNet architectures within the framework of Fully Convolutional Networks (FCN). To balance performance and computational complexity, RITnet employs 5 Down-Blocks in the encoder and 4 Up-Blocks in the decoder. The bottleneck layer of the encoder block serves as the final layer. Each Up-Block establishes a skip connection to its corresponding Down-Block, enhancing representation learning. The model is trained using several loss functions: Standard Cross-Entropy Loss (CEL) categorizes pixels into background, iris, sclera, or pupil. Generalized Dice Loss (GDL) penalizes pixels based on the overlap between ground truth and predictions. Boundary Aware Loss (BAL) weights pixels according to their distance from neighboring pixels, reducing confusion in CEL boundaries. Surface Loss (SL) helps restore small areas and shapes by scaling based on distance.

The Unsupervised Gaze Redirection Network [170] aims to acquire a comprehensive eye representation for gaze redirection. This framework takes an eye patch as input and generates a redirected eye patch while preserving rotational disparities between the two patches. The proposed method utilizes gaze redirection as a pretext task to facilitate representation learning.

3.3.4 Webcam-Based Methods

Algorithms that rely on standard webcams differ significantly from classical eye-tracking methods, which typically use infrared (IR) cameras along with an additional IR light source to create a glint (a reflection on the eyeball) for precise eye tracking. Although IR-based systems offer high accuracy in detecting eye movements, they come with increased hardware costs. In our research, we use regular, unmodified webcams to detect facial and eye regions, making eye tracking more accessible and cost-effective. However, estimating gaze coordinates using webcams is considerably more challenging than with IR-based systems. This is mainly because the pupil is not always clearly visible in the webcam footage. Furthermore, variations in lighting, head pose, and eye orientation introduce additional complexity in accurately estimating a person's gaze using webcam-based methods.

In this thesis, we build and train a webcam-based gaze estimation model for gaze

estimation. There are some works that are similar to our work. For example, the authors in [91] designed a differential gaze estimation by training a differential neural network. However, the authors in this article used calibrated high-quality eye images to train their network, which is different from our method because, in our approach, we used only low-resolution images without calibration. In another work [88], the authors designed a hardware-friendly CNN model that uses a minimum computational requirement to efficiently estimate gaze on low-cost devices. However, they tested their model on MPIIgaze, the dataset that is a calibrated dataset in which the on-screen gaze position is converted to the 3D position in the camera coordinate system using a camera calibration procedure from the OpenCV library.

3.4 Datasets for Gaze Estimation

Given the progress in the gaze estimation domain, a wide range of datasets have come to light to address different gaze estimation tasks [54]. The collection methodology for datasets has evolved over time, starting from controlled laboratory settings [112] and expanding to unconstrained indoor environments [56, 105, 102, 131, 54], and even outdoor settings [127]. Recent datasets are more advance in term of complexity, bias and volume. These datasets are good for training and evaluation for gaze estimation. Below is the description of some of the datasets.

MPIIGaze: The MPIIGaze dataset was collected by Xucong et al [178]. The authors collected the dataset from 15 individuals in their everyday natural sitting positions in front of a laptop. The researchers obtained a total of 213,659 images throughout the course of three months. To collect data, a series of random points were displayed on the participants' laptop screen. Following this, researchers have also collected the MPIIFaceGaze [177] dataset in which the entire participant's face is considered. The underlying assumption is that by including the complete facial features of an individual, the prediction of their gaze maybe be calculated more accurately.

Eyediap: Eyediap [48] dataset is characterized by its large-scale nature, which encompasses a wide range of subjects and exhibiting various environmental conditions. The design of the system was specifically aimed at addressing the primary obstacles that are commonly encountered when dealing with head pose, person, and 3-D target. The dataset includes recordings obtained from both controlled laboratory environments and real-world settings, thereby offering a wide array of scenarios that can be analysed.

Gaze360: The Gaze360 [80] dataset is a collection of gaze-tracking data that has been designed for the purpose of 3D gaze estimation in images that are not subject to any constraints. This dataset ensures estimation of gaze direction in 3D space. The dataset include a total of 238 individuals who were observed in both indoor and outdoor settings. The dataset includes annotated 3D gaze information, which was collected under various

conditions involving a diverse set of head poses and distances.

ETH-XGaze: ETH-XGaze [174] dataset consists of a collection of more than one million high-resolution images capturing a diverse range of gaze patterns exhibited by individuals under challenging head poses. The dataset has been acquired from a total of 110 individuals using a specialized hardware configuration. This setup consists of 18 digital single-lens reflex (SLR) cameras and allows for the adjustment of illumination conditions. Additionally, a calibrated system has been used to record the ground truth gaze targets.

TabletGaze: The TabletGaze dataset [69], consists of a collection of data from 51 subjects. The dataset includes observations of individuals in four distinct postures and captures their gaze at 35 different locations. The data was collected using a tablet device within an indoor environment.

RT-GENE: The RT-GENE [43] dataset is a collection of diverse images capturing gaze and head pose in a natural setting. This dataset aims to tackle the challenge of annotating ground truth by using a motion capture system to measure head pose and mobile eye tracking glasses to track eye gaze. The dataset include recordings obtained from 15 participants. Among these participants, there are 9 males and 6 females, with 2 individuals being recorded twice. The dataset consists of a substantial number of images, with 122,531 labeled training images and an additional 154,755 unlabeled images. Notably, the unlabeled images are of the same nature as the labeled ones.

CAVE: The CAVE [131] dataset is a collection of publicly available gaze data. It include a total of 5,880 images from 56 individuals, each showing a diverse range of gaze directions and head poses. Dataset boasts a remarkable number of images and fixed gaze targets. Specifically, dataset have 5 head poses for each subject, with 21 gaze directions per head pose. The study sample consisted of individuals from various ethnic backgrounds, ensuring a diverse representation.

Table 3.1: A comparison of gaze datasets with respect to different attributes. The abbreviations are: In: Indoor, Out: Outdoor, Both: Indoor + Outdoor, Syn: Synthetic, Seq: Sequence, EB: Eye Blink, GE: Gaze Event, GC: Gaze Communication.

Dataset	**Year**	**Sub**	**Image Resolution**	**Label**	**Data Statistics**	**Env**
CAVE [131]	2013	56	5184×3456	3-D	5880	In
UT MV[134]	2014	50	1280×1020	3-D	64000	In
EYEDIAP [48]	2014	16	512×512	3-D	237 min	In
MPIIGaze[178]	2015	15	1280×720	3-D, 2-D	213,659	In
SyntheticEye[160]	2015	NA	120×80	3-D	11,400	Syn
GazeFollow	2015	130,339	N/A	3-D	122,143	Both

Dataset	Year	Sub	Image Resolution	Label	Data Statistics	Env
UnityEyes [159]	2016	NA	400×300	3-D	1,000,000	Syn
GazeCapture [85]	2016	1450	640×480	2-D	2,445,504	Both
TableGaze[69]	2017	51	1280×420	2-D	816 Seq, 300,000 img.	In
InvisibleEye [141]	2017	17	5×5	2-D	280,000	In
MPIIFaceGaze [177]	2017	15	1280×720	3-D	213,659	In
RT-GENE [43]	2018	15	1920×1080	3-D	122,531	In
RT-BENE [32]	2019	17	1920×1080	EB	243,714	In
NV Gaze [82]	2019	30	1280×960, 640×480	3-D, Seg	2,500,000	Both
Gaze 360 [80]	2019	238	4096×3382	3-D	172,000	Both
Vacation [40]	2019	206,776	640×360	GC	96,993	Both
HUST-LEBW [65]	2019	172	1280×720	EB	963	Both
OpenEDS-19 [51]	2019	152	640×400	GEN	252,690	In
OpenEDS-20 [114]	2020	90	640×360	3-D	8,960 Seq., 550,400 img.	In
ETH-XGaze [174]	2020	110	6000×4000	3-D	1,083,492	In
mEBAL [34]	2020	38	1280×720	EB	756,000	In
EVE [116]	2020	54	6000×4000	3-D	12,308,334	In
GW [84]	2020	19	1920×1080	GE	5,800,000	In
OPENNEEDS [39]	2021	44	128×71	3-D	2,086,507	VR
GOO [140]	2021	100	N/A	3-D	201,552	Both
LAEO [98]	2021	485	N/A	3-D	800,000	Both
GazeONCE [171]	2022	10000	N/A	3-D	24,282	N/A
RavenGaze [168]	2023	34	1280×720	2-D, 3-D	556,476	In

3.5 The Powerful Gaze: Applications of Gaze Estimation Across Fields

Gaze estimation has emerged as a valuable tool across multiple domains, enabling to predict individual's visual focus. Through the analysis of eye movements and facial features, valuable insights can be obtained regarding an individual's attention, intention, and cognitive state. Here, we have provided a selection of eye-gaze applications across various domains.

3.5.1 Human-Computer Interaction (HCI)

Hands-free interfaces: Gaze estimation enables seamless interaction with computers without the need for hands. Users can effortlessly control interfaces, navigate menus, and select options using their gaze. This is especially advantageous for individuals with disabilities or in circumstances where using the hands is not feasible [175].

Accessibility Tools: Gaze tracking is an invaluable assistive technology for people with limited mobility, providing them with accessibility tools. It enables individuals to use computers and other devices independently [175].

3.5.2 Virtual and Augmented Reality (VR/AR) technology

Improving User Experience: Gaze estimation enhances VR/AR experiences by allowing objects to be activated or highlighted according to the user's gaze. This results in a more organic and captivating engagement [175].

Foveated Rendering: This technique uses gaze data to focus computational resources on the area where the user is looking, improve graphics quality, and reduce the processing power required for the rest of the visual field [175].

3.5.3 Marketing and User Research

Exploring User Attention: Through the analysis of eye movements on websites or advertisements, researchers gain insights into what sparks a user's interest and how they navigate through information. These data are extremely valuable and can be used to improve website design, product placement, and advertising campaigns [175].

Product Design and Development: Gaze tracking provides valuable insight into user behavior, allowing designers to understand which features attract the most attention and how users interact with interfaces. This can result in the development of products that are easier to use and understand.

3.5.4 Healthcare

Diagnostic: Eye tracking is a valuable tool for evaluating neurological conditions such as Parkinson's disease and autism spectrum disorder (ASD). By studying gaze patterns and pupil dilation, researchers can gain insight into these conditions [146].

Communication Aids: Gaze estimation has the potential to greatly benefit people with conditions such as Amyotrophic Lateral Sclerosis (ALS) by allowing them to interact with their environment and control communication devices through their eye movements [146].

Rehabilitation: Eye tracking technology can be seamlessly incorporated into rehabilitation programs for stroke victims and individuals with visual impairments. This innovative approach helps in the recovery process by helping patients regain control of their eye movements and improve their visual function [146].

3.5.5 Education and Learning

Personalized Learning: Teachers can gain valuable insight into student learning by observing their gaze patterns while studying. This allows educators to identify the specific areas that students find challenging and tailor their instruction accordingly. This assists teachers in adapting their instruction to better suits the individual needs of each student [175] .

Realtime Feedback: Monitoring the gaze patterns of students in online classes can provide valuable insights to teachers regarding their level of engagement and comprehension. This information can assist teachers in measuring student engagement and comprehension of the lesson [35] .

3.5.6 Gaming and Entertainment

Eye Tracking Gameplay: Gaze tracking has the potential to completely transform the gaming industry by integrating eye movements into the overall gameplay experience. For example, in a gaming experience where players can precisely aim weapons, select targets, or activate special abilities through eye-tracking technology, enhancing both immersion and strategic depth. This introduces an additional level of strategic thinking and engagement to gaming experiences.

Data Driven Game Design: Studying the gaze patterns of players during gameplay can provide valuable insights for game creators [89]. This information is valuable for game developers as it allows them to enhance the gaming experience. By fine-tuning the difficulty, designing levels more intelligently, and personalizing the game for each player, developers can create better games.

Biometric Authentication in Games: Gaze patterns have emerged as a potential

form of authentication in games. This innovative approach allows players to easily log in or access specific features by directing their gaze towards designated targets [19].

Augmented Reality Experiences: AR experiences can use gaze data to enhance the user's view by adding extra information or content that aligns with their line of sight. This creates a highly engaging and customized experience [175].

Virtual Reality Fitness: Gaze tracking has the potential to transform VR fitness experiences. By simply looking at virtual objects or using eye movements to control character movement, users can enjoy a more immersive and engaging workout session [63].

3.5.7 Security and Law Enforcement

Lie Detection: Research is currently being conducted to explore the potential of eye movements in detecting lies during questioning [146].

Security Attention Analysis: Applications of gaze tracking extend to various settings, such as airports, where it can be used to identify individuals exhibiting suspicious behavior by analyzing their eye movements [146].

3.5.8 Art and Design

Interactive Art Installations: Artists can utilize gaze tracking technology to create art installations that respond to the viewer's gaze, resulting in a personalized and unique experience for everyone [175].

Eye tracking in Art History: Studying eye-tracking in art history allows researchers to gain insights into how individuals perceive and interpret artworks. By analysing where people direct their gaze when observing different pieces, valuable information can be obtained citezhang2021eye.

3.5.9 Sports and Athletics

Performance Analysis: Coaches can utilize gaze tracking to analyse how athletes observe opponents or objects during practice or games. This enables researchers to develop more effective strategies for achieving optimal performance [120].

Training Tools: Athletes can enhance their focus and decision-making skills through the utilization of gaze tracking in training simulations. It assists individuals in honing their ability to focus on significant targets or specific areas of the field [120].

Chapter 4

Gaze Tracking Using an Unmodified Web Camera and Convolutional Neural Network

In the previous chapter a comprehensive literature survey was conducted to understand the progress in gaze estimation. This chapter goes through the objective of this dissertation where a goal is to investigate whether a person owning a standard webcam, typically on a laptop or desktop, can perform gaze estimation on their machine without the need for any additional hardware. This chapter is based on the paper [8], which was published in Applied Sciences Journal in 2021 (DOI: https://doi.org/10.3390/app11199068).

Gaze estimation plays a significant role in understating human behavior and in human–computer interaction. Currently, there are many methods accessible for gaze estimation. However, most approaches need additional hardware for data acquisition which adds an extra cost to gaze tracking. The classic gaze tracking approaches usually require systematic prior knowledge. Moreover, they are fundamentally based on the characteristics of the eye region, utilizing infrared light and iris glint to track the gaze point. It requires high quality images and infrared cameras with particular environmental conditions and another light source. Recent studies on appearance-based gaze estimation have demonstrated the capability of neural networks, especially convolutional neural networks (CNN), to decode gaze information present in eye images and achieved significantly simplified gaze estimation [27]. In this chapter, a gaze estimation method that utilizes a CNN for gaze estimation that can be applied to various platforms without additional hardware is presented. An easy and fast data collection method is used to collect images of the face and eyes from an unmodified desktop camera. The proposed method registered good results; it shows that it is possible to predict the gaze with reasonable accuracy without any additional tools. This chapter involves the classification of gaze into predefined regions on screen.

4.1 Introduction

Gaze estimation has long been recognized as an important research topic since it has strong real-world applications, for instance, in human–computer interfaces [101, 66], gaze-based interfaces [25, 123], virtual reality [112, 122], health care [56], behavioural analysis [64], and communication skills [127]. Therefore, gaze estimation has become a well-established research topic in computer vision, especially in human–computer interaction (HCI) [74, 97] . Early gaze estimation techniques required strict conditions to calculate gaze points, such as stabilizing a person's head and controlling light conditions. These restraints bounded the applications to relatively restricted laboratory environments. For applying gaze estimation in natural environments, authors have proposed many methods to mitigate these restraints and have uplifted gaze estimation towards being calibration-free, without person-specific and light independent gaze tracking [15, 182].

Many eye-tracking methods on the market can be quite costly, as they typically require specialized hardware and software. Given the increasing demand for eye-tracking applications, there is a corresponding need for affordable and widely accessible methods to gather data on human gaze coordinates. Thus, the utilization of unmodified cameras that are already present in most computer setups can have a significant impact on the widespread adoption of eye-tracking technology. These cameras are readily accessible and do not incur any additional expenses.

This chapter explores the potential of using unmodified web cameras to estimate gaze coordinates. In modern times, it's common for computers to come with built-in cameras that have the potential to be used for gaze estimation. One issue that occurs is that standard web cameras are limited to capturing images in the visible light spectrum, with many of them coming with infrared filters. Another issue is that these cameras typically come with a wide-angle lens and provide little to no zooming capabilities. The result is that the image of the eyes appears to be small with a low resolution and is greatly influenced by the surrounding light conditions. Using an unmodified camera for eye tracking poses a significant challenge.

Algorithms that utilize unmodified cameras have a distinct advantage over classical eye tracking algorithms. In contrast to classical eye tracker, which utilize infrared cameras and an additional infrared light source to track the eyes by capturing the reflection from the eyeball, algorithms that use unmodified cameras work differently. Although infrared-light-enhanced cameras are highly accurate in detecting eyes, they do come with an added cost for eye-tracking purposes. Our research involved utilizing standard unmodified cameras to efficiently detect face and eye regions, resulting in a significant reduction in eye-tracking costs. Estimating the gaze coordinate from unmodified cameras can be challenging due to the difficulty in consistently locating the pupil. In addition, variations in lighting, head orientation, and eye angles can pose challenges when attempting to determine the

direction of a person's gaze.

For this research, an unmodified web camera was used to capture an image which was then processed by the CNN network. The network was able to generate data about the specific area on the screen where the person was directing their gaze.The CNN network was trained and tested using various architectures and combinations of face, eyes, and single eyes to accurately estimate the gaze coordinate from camera images. Our experimental results indicate that it is indeed feasible to accurately predict human gaze points using an unmodified camera.

The main contribution of this chapter is as follows.

- This chapter introduces a dataset for gaze estimation using unmodified camera.
  - This dataset is unique because it was collected "in the wild" by users on their own laptop without any supervision.
- The dataset was used to train different neural networks for gaze estimation.
  - Data was collected using a webcam, as a result, achieving high accuracy for precise gaze point estimation was challenging.
- However, the results show that it is possible to determine with good accuracy which of the 20 predefined screen areas the user is looking at.

4.2 Methods and Materials

This section delves into the research methodology, beginning with the procedures for collecting data. In addition, it provides a comprehensive overview of the essential data preprocessing steps required for successful deep learning applications. Next, the chapter proceeds to discuss the dataset utilized in the study. Lastly, the final subsection explores the different neural network architectures that were examined in the experiment.

4.2.1 Data Collection Methodology

This section focuses on data acquisition, a critical aspect of deep learning. The quality of acquired data directly impacts the training process for neural networks. Any outliers or deviations from the expected data format can negatively affect the entire training process.

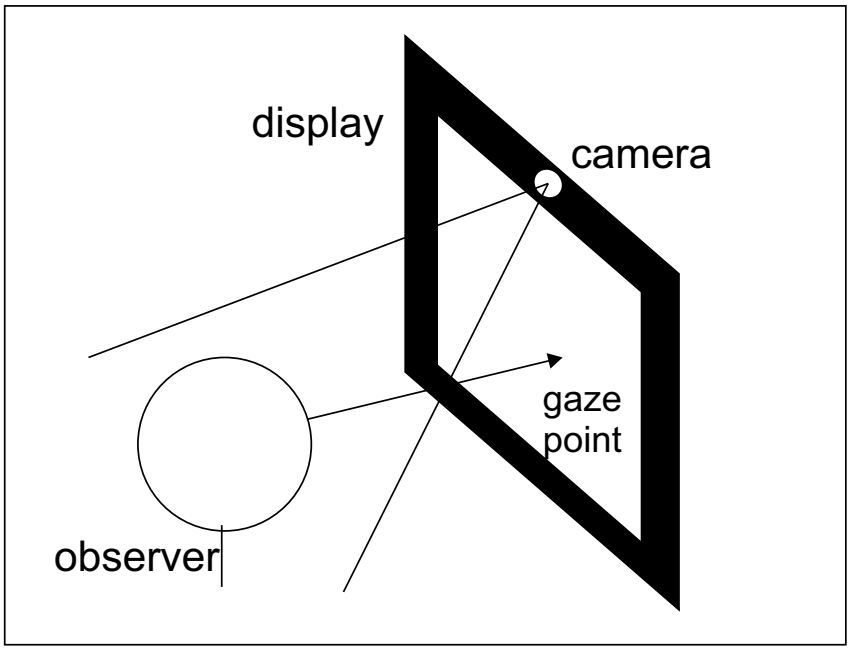


Figure 4.1: The experimental setup

Data were collected using the EyeTrackerDataCollector application, which is a desktop application registering images from a camera. No specific hardware requirements were made. The only hardware requirement was to have a webcam, preferably built into the top of the laptop screen (see Figure 4.1). This assumption allowed for the acquisition of images under real-world conditions and not typical laboratory conditions. These measures were designed to answer the question of whether anyone owning standard webcam equipment could use a classifier capable of determining where on the screen a person is focusing their attention. Test subjects were asked to pay attention to the lighting during image acquisition. A problem with the lack of robustness to this factor was noted at the outset. Failure to adjust the lighting resulted in very low-quality, dark even black, or completely overexposed images. The subject sat centrally facing the screen with the built-in camera at the height of the horizontal axis of the eyes at about 35 to 50 cm from the monitor. During every session, the subject was required to click 20 points displayed subsequently on screen in different evenly distributed locations (in a five by four grid). After each click, a series of three camera images was taken and stored together with information about the point's location.

4.2.2 Data Preprocessing

Figure 4.2 presents the image processing steps. The original images as taken from the camera are presented on the left side of the figure, while the right side presents the images that were additionally sharpened using a function available in the OpenCV library. Step (1) is to take a picture with a webcam. Step (2) is to convert the image to grayscale. Step (3) is to cut out the face detected by the Viola–Jones classifier using a proper Haar cascade

[148]. This method was presented by Paul Viola and Michael Jones in 2001. The proposed solution is based on machine learning and uses Haar-like features. The cascade feature is trained on a large number of positive and negative images. It is then used to detect objects in other images. In the first stage, the algorithm needs a large number of positive images, i.e., images that will be targeted for detection, such as images containing a face, and a large number of negative images that do not contain the object to be detected in a later stage. It is important to keep the size of the images the same. To achieve this goal, the images must be scaled appropriately to equal size. It allows the algorithm to omit any unnecessary and meaningless factors that would also have to be considered during learning. Once trained, such a classifier returns a value of "1" when an object is found or "0" if an object is not found. To search for an object across the image, the classifier, in the form of a window, moves across the image in search of the object by checking each location. The classifier is designed to be easily expandable and modifiable. It allows searching for objects of different sizes. Such a solution translates into the classifier's efficiency because the size of the input image is not required to change. To find an object of unknown size in the image, the scanning procedure is repeated several times, and the scaling factor of the classifier is modified each time. We used the cascades available in the OpenCV library.

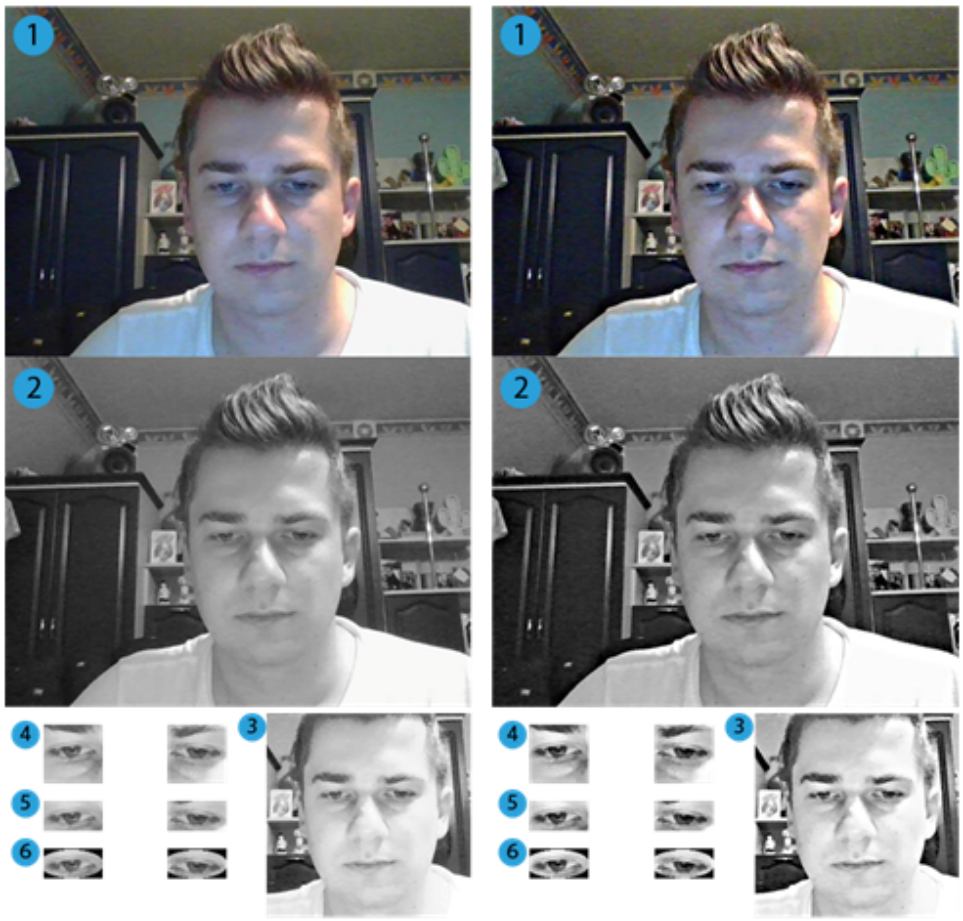


Figure 4.2: Stages of image processing.

The right and left eyes are found and marked by using another Haar cascade in step (4). The face is detected first, and then the eyes are detected in the face image. This approach reduced the susceptibility of misclassification of eyes in the image by detecting objects that are confusingly similar to eyes. Step (5) is to precisely cut out the eye itself if possible. Step (6) (the final set) is to mask the eye from the fifth stage with a white ellipse with a black border. Performing the fifth and final sixth stages help to reduce the influence of meaningless pixels surrounding the eye as much as possible. This approach increases the quality of the trained neural network. The last step before using the data to train the CNN is to resize the images. For proper operation, the neural network requires that each of the input images should be of the same size. Therefore, all eye images were rescaled to 60 x 30 pixels and all face images to 227 x 227 pixels using methods available in the OpenCV library.

Figure 4.3: An example of blink eye images for in red rectangle

The data prepared using this approach requires manual verification. The process involves thoroughly examining all gathered images to ensure their accuracy and relevance. In certain cases, the Haar cascade classifier may mistakenly identify a different element on the image that resembles an eye. In some instances, the individual conducting the examination may have directed their attention elsewhere. These cases should be disregarded. It is also possible that the individual blinks at the moment the picture is captured, resulting in the detection of a partially closed eye. It is important to manually remove these deviating images from the dataset before proceeding with the neural network training process. In the future, our goal is to enhance our capabilities in identifying deviations by implementing more advanced automated methods. Images that have been excluded from the dataset are highlighted in red and can be seen in Figures 4.3 and 4.4. In the first figure, it appears that the subject blinked while the image was being taken. However, in the second figure, there seems to be a mistake made by the Haar cascade classifier. It mistakenly identified an extra element in the image that closely resembled the eye feature.

The identified element appears to be a small piece of the individual's hair.

The final step of pre-processing is image value normalization. This process involves normalizing the pixel values of an image. The images in the dataset used for training and testing the neural network are normalized by dividing the pixel values by 255, which ensures that each pixel value falls within the range of 0 to 1.

Figure 4.4: An example of an extra element incorrectly detected as an eye by the classifier in red rectangle.

4.2.3 Datasets

Two datasets were created and utilized during the training process for the neural network. The initial dataset contained images obtained from a single person. There are around 6000 images in this dataset, which includes all the images obtained for the 20 points. It suggests that there are approximately 300 sets of images for each point, which include left and right eye images as well as a face image. The second dataset includes images obtained from four subjects, with one subject repeating the tests twice. The first time, they did not wear glasses (with lenses), and the second time, they wore glasses. In this set, the number of images is more than quadrupled as each tested person was asked to provide the same set of data.

4.2.4 Network Architecture of the Tested Model

This section details three designed models, each tailored to accept different input images. Despite variations in input, all models share fixed elements: output layer activation functions, hidden layer activation functions, weight initialization strategy, and loss function. These architectures represent the final, refined versions resulting from extensive parameter testing. Each network outputs a vector of 20 values, where each value signifies the probability of the input image corresponding to the user looking at a specific area within a pre-defined 5× 4 grid of 20 regions.

Every network consists of three kinds of components:

- Convolutional layer (Convolution): The layer's input is an image with some number of channels, and the layer creates another image with the number of layers equal to the number of filters. The number and the size of filters used to convert the image are two parameters of such layer.

- Pooling Layer (Subsampling): The layer's input is an image, and the output is the image reduced in both dimensions. Only Max Pooling layers were used, which reduced the image by representing the area of a given size by one pixel, which is the brightest. There is only one parameter for this layer—the size of the reduction. Only two sizes, 2×2 and 3×3, were used.

- Fully connected layer (FC): It is a classic neural network's layer that consists of some number of neurons, and every neuron received a weighted combination of all input values.

The Convolutional Neural Network (CNN) was trained using the ADAM optimizer [83] with a starting learning rate of 0.00015. ReLU (Rectified Linear Unit) activation [166] was employed for hidden layers, while Xavier initialization [55] provided weight initialization. The output layer utilized a normalized Softmax exponential function Equation (4.1) for activation.

$$\sigma(y_i) = \frac{e^{y_i}}{\sum_{j=1}^{K} e^{y_j}} \tag{4.1}$$

where y_i represents the output from the network, and $K = 20$ is the number of classes.

Negative Log-Likelihood (NLL) was taken as the loss function, which works well with the Softmax activation function for solving multi-class neural network learning and image classification problems considering a set of pixels as input. Equation (4.2) represents the mathematical formula for the NLL function:

$$l(\theta) = -\sum_{n=1}^{M} (Y log(\sigma(\theta^T X)) + (1 - Y) log(1 - \sigma(\theta^T X)) \tag{4.2}$$

where θ is the parameter of the function, Y denotes the output values, X represents the features of the image, and M is the number of samples.

An identical naming convention for the neural network layers was adopted for each experiment. The LRE symbol designation refers to layers using the right eye images, the LLE symbol refers to layers describing the left eye, and LF refers to layers pertaining to face images. The Convolution layers always have information about the size of the filter, and the numbers of filters are provided in the text or tables. The Subsampling layers have information about the reduction size. The number of neurons for each Fully Connected layer is explained in the corresponding text. The FC symbol indicates fully connected layers where, for example, FC-RE refers to the fully connected layer for the right eye.

One Eye Image as a Neural Network Input

For this experiment, a network architecture was set up to take in an image of a single eye as one of its parameters. Figure 4.5 displays the architecture. The neural network is composed of three convolutional layers, a subsampling layer with a filter size of 2×2,

and a fully connected layer. The initial convolutional layer (LLE1) consists of 96 filters, followed by LLE3 with 384 filters, and LLE4 with 256 filters. The FC1 layer consists of 256 neurons, while the final layer produces a vector of 20 values. Two models were trained: one using left eye images and the other using right eye images, referred to as ARCH-LE and ARCH-RE respectively.

Face Image as Neural Network Input

The neural network architecture designed for the second experiment utilizes face images as its parameters. Figure 4.6 displays the schema of the architecture. The neural network consisted of five convolutional layers, three subsampling layers with a filter size of 3× 3, and two fully connected layers called FC1 and FC2. Table 1 provides the details of the neural network layers. The fully connected layers had 4096 neurons for FC1 and 1000 neurons for FC2. In the following sections, we will use the term ARCH-F to refer to this solution.

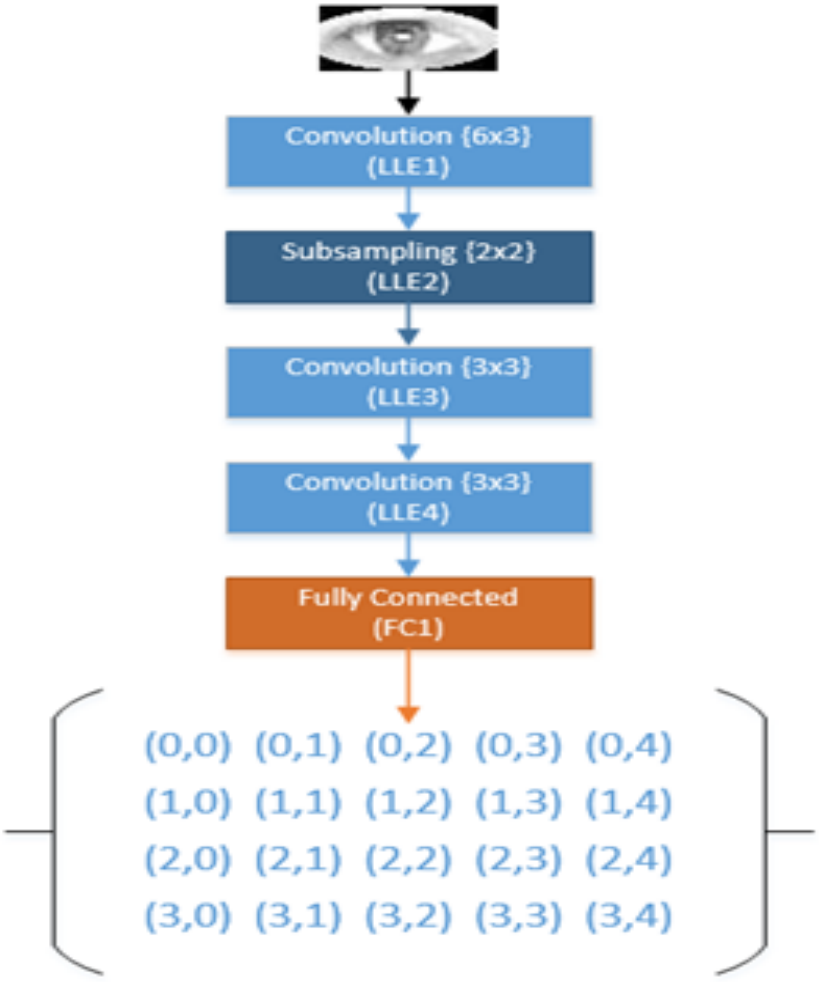


Figure 4.5: Network architecture for processing images with one eye (ARCH-LE and ARCH-RE).

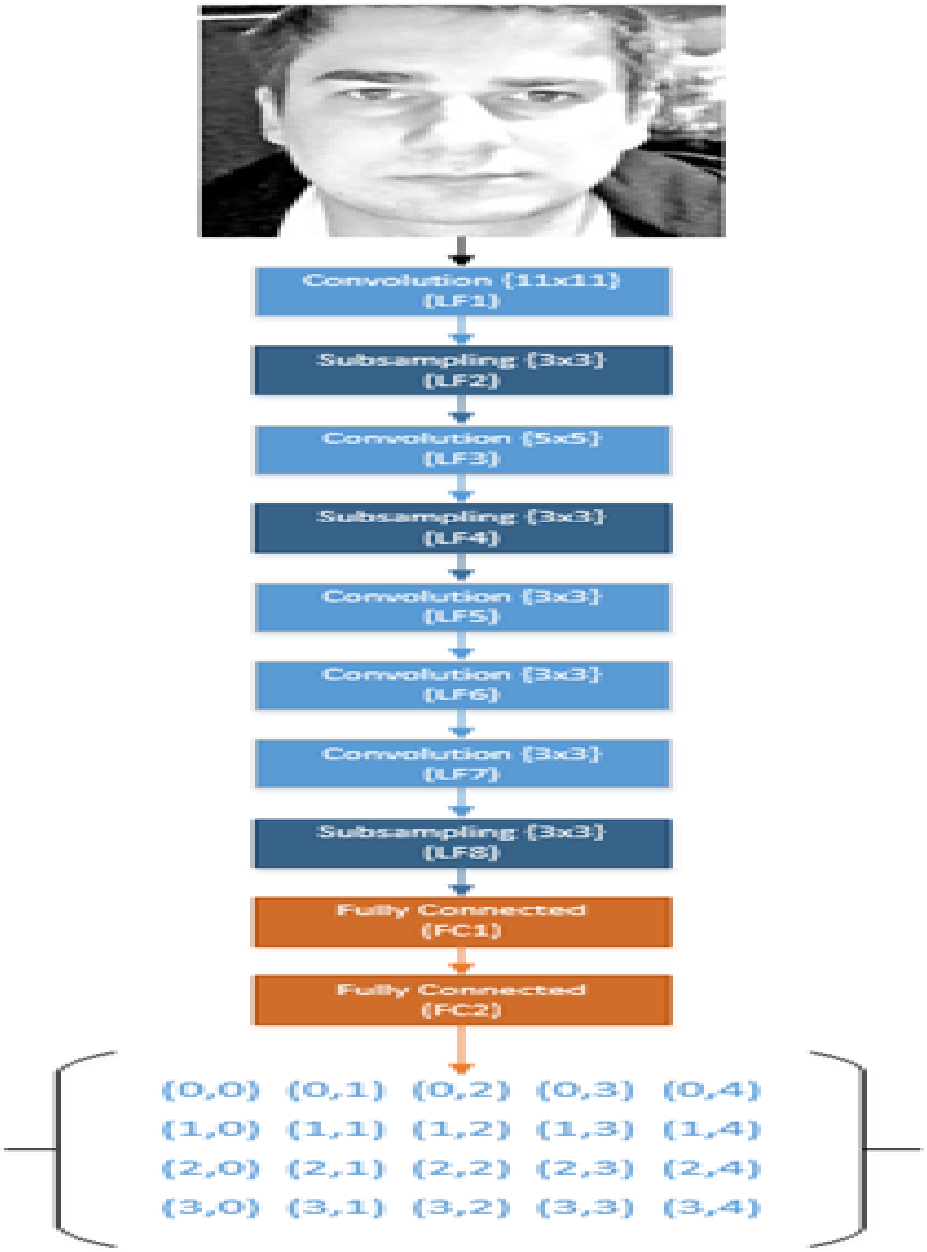


Figure 4.6: Network architecture for processing face images (ARCH-F).

Table 4.1: Characteristics of convolutional layers for an experiment using face image as neural network inputs

Layer	**Number of Filters**	**Filter Size**	**Step**
LF1	96	11×11	4
LF3	256	5×5	1
LF5, LF6	384	3×3	1
LF7	256	3×3	1

Both Eye Images as a Neural Network Input

For this experiment, a network architecture was prepared to accept left and right eye images as its input parameters. The schema is shown in Figure 4.7. The neural network consists of three convolutional layers, one subsampling layer 2× 2, one concatenation layer, and three fully connected layers.

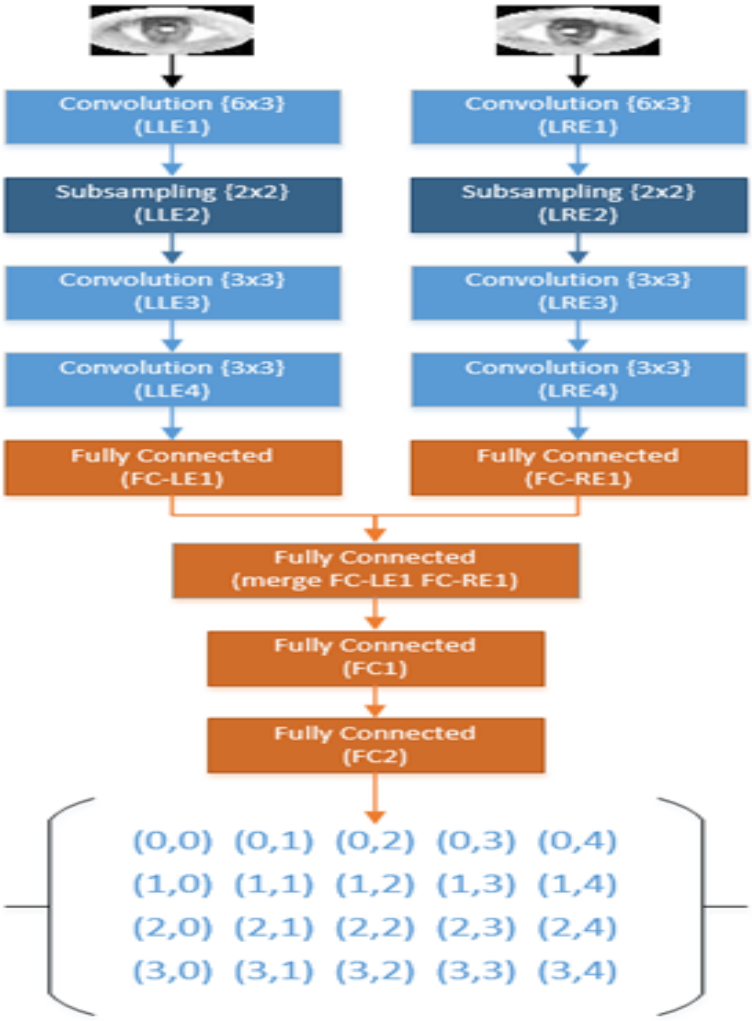


Figure 4.7: Network architecture for the network using both eyes images.

The fully connected layers consist of a layer for each input image, i.e., FC-LE1 and FC-RE1 with 1024 neurons, the concatenate layer, and two layers: FC1 with 2048 neurons and FC2 with 1024 neurons. The details of the convolutional layers of the neural network are presented in Table 2. The output of the developed neural network architecture is a set of twenty points representing the position of the area on which the subject focuses his/her gaze. This solution will be referred to as ARCH-LRE.

All three presented network architectures are a little bit different in terms of topology. For instance, one eye image architecture contains only three convolutional layers, one subsampling layer, and one fully connected layer (Figure 4.5). Face image architecture

is more complicated and contains two sets of alternate convolutions and subsampling layers and then again three convolutional layers, one subsampling layer, and finally two fully connected layers (Figure 4.6). For both eyes architecture (Figure 4.7), the network has a similar architecture as one eye network, but the outputs from the left eye and right eye stacks are concatenated at the end and sent to two fully connected layers. All three architectures have a different number of parameters because they all have a different number of layers. The proposed architectures were chosen from a set of several possibilities that were initially studied. These networks are not generic, and the search for possible better architectures requires further studies.

Table 4.2: Characteristics of convolutional layers for an experiment using both eyes images as neural network inputs

Layer	**Number of Filters**	**Filter Size**	**Step**
LLE1, LRE1	2×96	6×3	2
LLE3, LRE3	2×384	3×3	1
LLE4, LRE4	2×256	3×3	1

4.3 Experiments and Results

This section provides an overview of the research conducted and the findings obtained. It begins by outlining the methodology used to obtain the results. The following sections describe each of the experiments performed. The last section highlights the results and presents the conclusions derived from the research.

4.3.1 Research Methodology

The training set and the test set were determined through an automated and random process, with 70% of the data allocated for the training dataset and the remaining 30% for the test dataset. In terms of numbers, this equates to around 4200 images of left and right eyes and faces for the training set. Additionally, there are 1800 images allocated for the test set of Dataset.

Each neural network configuration was repeatedly changed in terms of architecture or hyperparameters and then retested. All experiments were performed on a laptop with Intel Core i7 and 16 GB RAM hardware specifications. The prediction quality was determined by the precision index built into the DeepLearning4j library calculated from the classification accuracy of the test set.

4.3.2 Results

In this section, we will discuss the outcomes of the tests conducted using the neural network architectures that were specifically designed for the provided data. Our primary objective in this research was to identify the most effective architecture for accurately classifying the focus point of gaze. The models that were developed during the experiments were integrated into the EyeTracker application, which was specifically designed for this purpose. This allowed for real-time testing of their performance.

The experiments were carried out on two datasets. In the study, there were two datasets used. The first dataset (D1) consisted of images from one person, while the second dataset (D2) included images from four different individuals. Additionally, one individual participated in the test twice, once wearing contact lenses and once wearing glasses, as explained in Section 4.2.2. All network architectures were trained for up to 70 epochs.

The first experiment was aimed at checking if the model will work better after specific preprocessing. The first subset consisted of original grayscale images, and the second of the images was additionally sharpened by using an algorithm available in the OpenCV library.

This experiment was performed based on images from the dataset containing data collected from one person (D1) and using ARCH-LRE architecture. The results of the experiment are presented in Table 4.3 (columns two and three).

Table 4.3: Results for original and sharpened images (the latter for both D1 and D2)—averaged accuracy for the range of training epochs

Epoch	**Original Images** (D1)	**Sharpened Images** (D1)	**Sharpened Images (D2)**
1–10	26.01%	27.52%	16.25%
11–20	41.25%	45.92%	30.73%
21–30	56.48%	59.12%	43.04%
31–40	67.32%	68.34%	53.06%
41–50	73.62%	72.70%	54.46%
51–60	76.52%	77.68%	63.28%
61–70	77.24%	81.61%	73.28%

The second experiment compared the results for two datasets, D1 and D2. Since the first experiment showed that the network performs better with sharpened images. The results are shown in the Table 4.3.

The following experiment conducted the learning process of a neural network using an architecture that utilizes complete face images as input parameters. In Figure 4.6,

we present the schema of the ARCH-F architecture used in this experiment. Table 4.4 presents the results of the experiment for Datasets 1 and 2.

Table 4.4: Results for ARCH-F architecture—averaged accuracy for the range of training epochs.

Epoch	**Dataset 1**	**Dataset 2**
1–10	11.80%	7.42%
11–20	15.20%	38.07%
21–30	22.74%	72.66%
31–40	39.88%	80.02%
41–50	61.81%	83.99%
51–60	77.00%	66.61%
61–70	81.81%	80.81%

In the last two experiments, we aimed to assess whether it is possible to achieve similar results by using images from only one eye, compared to a more complex network that uses both eyes. The results of these experiments, performed on both left and right eyes in both datasets, are summarized in Table 4.5.

Table 4.5: Results for ARCH-L and ARCH-R architectures—averaged accuracy for the range of training epochs.

	Left Eye		**Right Eye**	
Epoch	**Dataset 1**	**Dataset 2**	**Dataset 1**	**Dataset 2**
1–10	34.72%	44.00%	35.93%	36.22%
11–20	68.51%	64.29%	63.86%	59.49%
21–30	80.86%	72.75%	75.85%	67.94%
31–40	84.71%	75.17%	78.65%	71.95%
41–50	87.15%	77.40%	80.58%	74.34%
51–60	87.77%	78.03%	81.51%	75.37%
61–70	88.55%	79.53%	82.30%	75.93%

4.3.3 Discussion

The experiments in section 4.3.2 demonstrate the possibilities for achieving reliable eye-tracking results in diverse environments, even when using low-quality web cameras. Certainly, the results are far from perfect—we only assessed if one of the 20 areas could be identified using eye images—but this outcome might meet the requirements for various human-computer interface (HCI) applications. In the initial experiment, it was found that

sharpening the acquired image had a beneficial effect on subsequent classification, resulting in an increase in accuracy from 77% to 81%. As a result, all subsequent experiments utilized the sharpened versions of the images. As expected, the results for Dataset 2 were consistently inferior to those of Dataset 1 across all experiments. It is worth noting that these results are also quite impressive, with accuracy ranging from 72% to 80% depending on the architecture. It's important to note that when randomly guessing among 20 classes, the accuracy is typically around 5%. Therefore, the results obtained are significantly better than random chance.

It is worth noting that the architecture that yielded the highest accuracy (over 88%) was the one that exclusively utilized images of the left eye (ARCH-L) (Table 4.5). It is likely that the main reason for this phenomenon was the complexity of the network that utilizes two eye images and concatenates layers (ARCH-RL), coupled with the insufficient training of only 70 epochs.

Another interesting finding was the good accuracy of the model that uses face images instead of eyes. The model obviously has more data to analyze, but a large percentage of this data is probably irrelevant. Due to more extensive input, this model was also more challenging to train; the training process lasted significantly longer than for other models. However, good results suggest that eye detection algorithms may be omitted in some applications.

4.4 Conclusion

This chapter explores the potential of using an unmodified camera to track a person's gaze. Due to the high cost of commercial eye tracking devices, their availability is quite limited. Our research introduces a cost-effective eye-tracking method that can be seamlessly integrated into standard desktop or laptop computers, eliminating the need for any extra equipment. A commonly available CNN network was utilized, making use of unmodified webcams that are typically found on computers. The images obtained from the unmodified cameras have poor quality and are highly affected by variations in lighting. Consequently, achieving satisfactory results proved to be quite a formidable task. Nevertheless, this study demonstrates that by using carefully crafted topology and fine-tuning hyper-parameters, the CNN network can yield results that have potential applications in real-world scenarios. While this study achieved noteworthy findings, it is important to acknowledge the limitations that are also present. For example, the method used to calculate the exact gaze point did not involve treating it as a regression problem. Instead, it was simplified by classifying the gaze point into one of 20 areas. This approach makes it challenging to compare with other methods. In addition, the study had a limited number of participants and a small amount of data. It is widely recognized that deep learning networks perform and generalize more effectively when trained on large datasets. In addi-

tion, the study did not utilize any publicly available datasets for benchmarking purposes. Next chapters intend to tackle the aforementioned limitations.

4.5 Contributions

After conducting all experiments presented in this chapter, the first hypothesis was confirmed that it is feasible to develop a CNN-based model that classifies a person's gaze into screen regions using low-resolution, low-quality eye and face images captured by a standard webcam.

Chapter 5

Person-Specific Gaze Estimation from Low-Quality Webcam Images

In the previous chapter, we explored the feasibility of predicting a person's gaze using limited resources by mapping their gaze to 20 predefined regions on the screen. In this chapter, we extend that work by developing a model capable of estimating gaze continuously across the entire screen, rather than restricting it to fixed regions. We approach gaze estimation as a regression problem, which more closely reflects real-world scenarios. Furthermore, we developed person-specific models that achieved performance comparable to high-end eye trackers. Unlike generalized models trained on data from multiple individuals, a person-specific gaze estimation model is trained for a single user. This personalized approach allows for more accurate gaze predictions tailored to individual characteristics. Importantly, our method relies solely on low-quality images captured from a standard desktop webcam, making it applicable to any computer system without the need for additional hardware.

We began by collecting a dataset of face and eye images using a webcam. Various convolutional neural network (CNN) configurations were then tested, including different learning rates and dropout rates. Our experiments show that person-specific models outperform generalized models when tuned with appropriate hyperparameters. Specifically, we achieved the following Mean Absolute Errors (MAE) in pixels: 38.20 for the left eye, 36.01 for the right eye, 51.18 for both eyes combined, and 30.09 for full-face images. These correspond to angular errors of approximately 1.45°, 1.37°, 1.98°, and 1.14°, respectively. This chapter is based on the paper [9], which was published in Sensors Journal in 2023 (DOI: https://doi.org/10.3390/s23084138).

5.1 Introduction

Various eye trackers are available today, with desktop models offering 1–2° accuracy and high-end commercial trackers achieving around 0.5° [49, 132]. Despite this progress, challenges remain, including high costs [85], intrusive hardware [85], and performance limitations in real-world settings [48, 134, 176]. These obstacles continue to hinder the widespread adoption of eye-tracking technology across devices like desktops, laptops, and smartphones. Our main objective is to overcome these constraints and ensure that eye-tracking technology is accessible to all, regardless of limited resources.

We maintain a well-founded confidence in our ability to achieve this significant milestone. Therefore, the goal is to develop aystem that can operate on both desktop and mobile devices, eliminating the need for additional instruments. Our ultimate aim is to achieve an accuracy level that is comparable to that of commercial devices. For this specific task, we opted for desktop or laptop devices. This decision was based on the fact that mobile devices currently have exceptional built-in cameras that are capable of capturing high-quality images. Our objective is to develop an eye-tracking algorithm capable of functioning on low-quality images, such as those captured by standard cameras on laptops or desktop devices.

Recent advances in deep learning techniques have led to significant transformation in various fields. This includes computer vision tasks like image recognition[86], object detection [125], object tracking [156], and segmentation [61]. Unfortunately, progress in eye tracking is hindered by the scarcity of available data [176, 69, 134]. In this work, we attempted gaze estimation for a specific person and collected our own dataset using desktop application. To accomplish this, we developed our own dataset by utilizing a desktop application. We experimented with various combinations of the left eye, right eye, and face regions to determine their effectiveness in estimating gaze. In addition, we conducted experiments with various CNN architectures and fine-tuned their parameters to achieve the most optimal model.

The main contribution of this chapter as follows:

- We have gathered a dataset of eye and face images that have low-quality, which can be utilized for training deep CNN models.
- We conducted an analysis of various hyperparameters of the CNN network and found that their values greatly influence the results.
- We demonstrated that the model developed for a single individual yields significantly fewer errors compared to the general models trained on data from multiple users. Furthermore, the error rate of this model is comparable to that of commercial eye trackers, measuring below 2 degrees of visual angle.

5.2 Methods and Materials

This section provides details about the data collection process for the experiment, as well as an explanation of the essential data preprocessing steps needed for deep neural networks. In addition, this section includes details about the dataset and an overview of the network architectures utilized in this study.

5.2.1 Data Collection

This section present data collection procedure which places significant role in deep learning techniques.

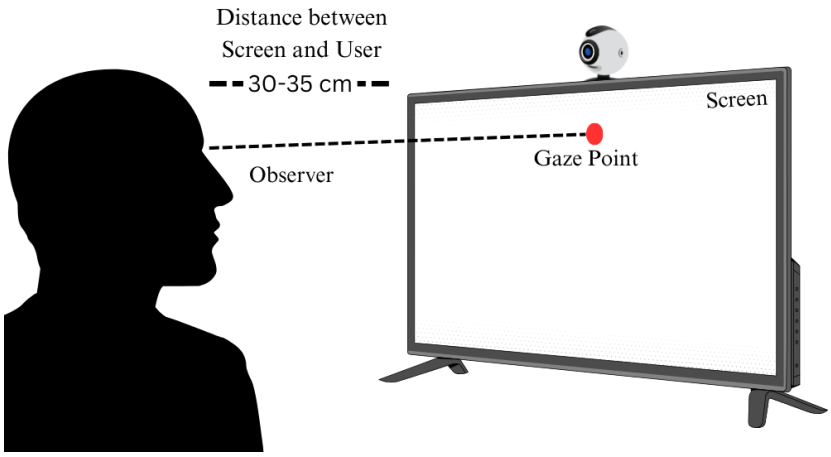


Figure 5.1: The experimental setup

The data used for the experiment were collected using a desktop application called DataCollector. This application is designed to collect images from the built-in camera of a laptop. In this research, we utilized the MSI GF75 Thin laptop with a Core i7 9th Gen processor. The resolution of the laptop camera during the data collection was 640 x 480 pixels. All that was needed for this data collection procedure was a webcam, preferably positioned on the top of the desktop screen as shown in Figure 5.1 [8]. By implementing this simple procedure, we were able to gather data in real-world situations, rather than confining ourselves to a controlled laboratory setting. The participant was instructed to focus on various points displayed on the screen while sitting at a distance of 30 to 35 cm from the screen. The participant was instructed to carefully consider the lighting conditions during the data collection process, as variations in lighting can significantly impact the image quality. Not being mindful of the lighting conditions occasionally led to

subpar images, either completely dark or partially obscured (with one eye visible and the other not).

The data was collected in 22 sessions over several days, at various locations, and at different times of the day and night. Each session had a duration of one to two minutes. This approach was designed to ensure that the dataset was diverse and included a wide range of variability. During daytime, when the lighting conditions were optimal, the image quality was better. However, at night, the image quality was low due to the lack of light in the room and its surroundings. Thus, the dataset contains images captured under various lighting conditions, allowing for comprehensive testing of the model in different environments. The variability in the dataset contributes to the robustness of the models trained on it, resulting in more generalized results. Throughout each data collection session, the participant was instructed to select 54 specific points on the screen.

The points were selected at random, allowing the user to click any point on the screen. However, it was suggested to maintain an equal distribution of points across the entire screen. Every time a click occurred, a collection of ten images capturing the person's face and their gaze location were stored. The dataset used for this research is unique compared to other universal datasets that are collected from multiple users to train the model. In this case, the dataset was specifically collected for an individual user.

5.2.2 Data Preprocessing

Once the data was collected, the subsequent task involved data preprocessing, a crucial step in any machine learning or deep learning approach. Here is Figure 5.2, which shows all the preprocessing steps. The first step in Figure 5.2a was to take an image from the webcam, and the next step in Figure 5.2b was to detect face using the Viola–Jones (VJ) classifier that uses the Haar cascade [148]. All incorrectly-classified images were manually removed. In the next step, the left eye and right eye (Figure 5.2c) were detected using another Haar cascade. This approach reduced the chance of misclassifying eyes in images by detecting objects that are similar to eyes [8]. The average dimensions for the left eye image were 55.12 × 55.12 pixels with standard deviation 8.76; for the right eye 55.89 × 55.89 pixels with standard deviation 7.23; and for the face image 223.24 × 223.24 pixels with standard deviation 23.29. Before sending data to the CNN, all eye images were resized to 64 × 64 pixels and face images to 224 × 224 pixels.

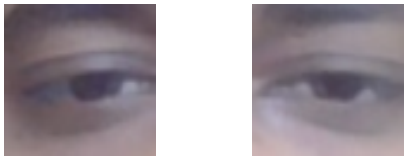
The data collected using this method required manual checking and verification before being inputted into the neural network. One of the tasks involved is thoroughly examining all images to ensure their information is correct. There have been instances where the Haar cascade for eyes has detected objects having a resemblance to eyes. We carefully reviewed these images and took the necessary steps to remove them from the dataset. In addition, on certain occasions, individuals would blink or close their eyes while being recorded. We

manually removed such images from the dataset before training.

The dataset was collected for the purpose of person-specific gaze estimation. We collected a total of 11,800 images for the experiment. In addition, we created an additional dataset by masking the area around the eyes with a white ellipse that has a black border Figure 5.2d. We implemented this procedure to assess the impact of minimizing the influence of surrounding pixels. The dataset was divided into an 80/20 ratio for training and testing various models.

(a) Step 1: Original image.

(b) Step 2: Face detection.



(c) Step 3: Eye detection.

(d) Step 4: Eye masking.

Figure 5.2: Pre-processing steps.

The dataset was collected for the purpose of person-specific gaze estimation. We collected a total of 11,800 images for the experiment. In addition, we created an additional dataset by masking the area around the eyes with a white ellipse that has a black border Figure 5.2d. We implemented this procedure to assess the impact of minimizing the

influence of surrounding pixels. The dataset was divided into an 80/20 ratio for training and testing various models.

5.2.3 Convolutional Neural Network Architecture

This section presents various architectures that have been utilized in the study. We opted for the CNN network architecture as it is currently state of the art solution for image data. We examined two different architectures: one with a single image as input and another with two images as input. For one-image input, we utilized various types of images including the left eye, right eye, both eyes and masked eyes, and full-face images. When working with a two-image input, we merged the images from the left and right eye. The network output consisted of a vector containing two values that indicated location where a person was looking on the screen. Each network that was tested included five different types of layer:

- Convolutional layers are responsible for learning the feature map from the input image.
- Pooling layers that decrease the size of the image.
- Batch normalization layers are employed to ensure stability in the neural network.
- Dropout layers are employed to mitigate the risk of over-fitting during the training process.
- Fully connected layers that compute the ultimate output.

The network was trained using the ADAM optimizer [83] with a variety of learning rates to fine-tune the network parameters. Once the optimal learning rate was discovered, the network was re-trained using this value. For the activation function in the convolutional and fully connected layers, we opted for the rectified linear unit (ReLu) [46]. The weight initialization for filters was set to the default option provided by the Keras implementation. A loss function, specifically the MAE function, was utilized during training. Equation 5.3 represents a mathematical formula for MAE.

$$\text{MAE} = \frac{\sum_{i=1}^{N} |y - \overline{y}|}{N}, \tag{5.1}$$

In equation, actual value is represented by y , the predicted value is represented by $\overline{y}$ and the total number of examples represented by N.

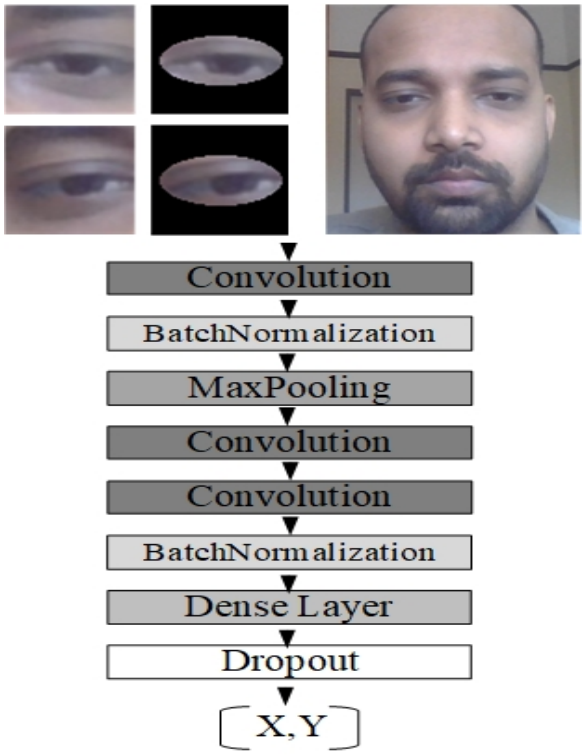


Figure 5.3: Architecture that takes normal eyes, mask eyes, and face image as input.

An identical naming convention was used for experiments involving a single eye, double eye, face, single eye with ellipse (mask), and both eyes with ellipse. The symbols LE, RE, FF, BE, LEM, REM, and BEM are used to represent different inputs for the eyes and face. LE represents the left eye, RE represents the right eye, FF represents the full face, BE represents both left and right eyes, LEM represents the left eye with an ellipse (LeftEyeMasked), REM represents the right eye with an ellipse (RightEyeMasked), and BEM represents both left and right eyes with a mask (BothEyesMasked).

The architecture for a single eye and face in CNN includes 3 convolutional layers, 1 pooling layer, 2 batch normalization layers, 1 fully connected layer (also called a dense layer), 1 dropout layer, and 1 output layer. The output layer consists of 2 neurons that represent the X and Y coordinates of a person's gaze on the screen. The CNN architecture for both eyes include multiple layers such as convolutional layers, max pooling layers, batch normalization layers, dropout layers, concatenate layers, dense or fully connected layers, and a final dense layer (the output layer).

The architecture used in this study, as shown in Figure 5.3 Figure 5.4 a depicts the network architecture, which accepts either a single eye image (with or without a mask) or a face image as input. The network architecture in Figure 5.4 takes the left eye and right eye images as input. Figure 5.4 shows the merging of LE and RE, which combines the outputs from the left and right networks. The results from the concatenation layer are passed to the dense layers and ultimately to the output layer.

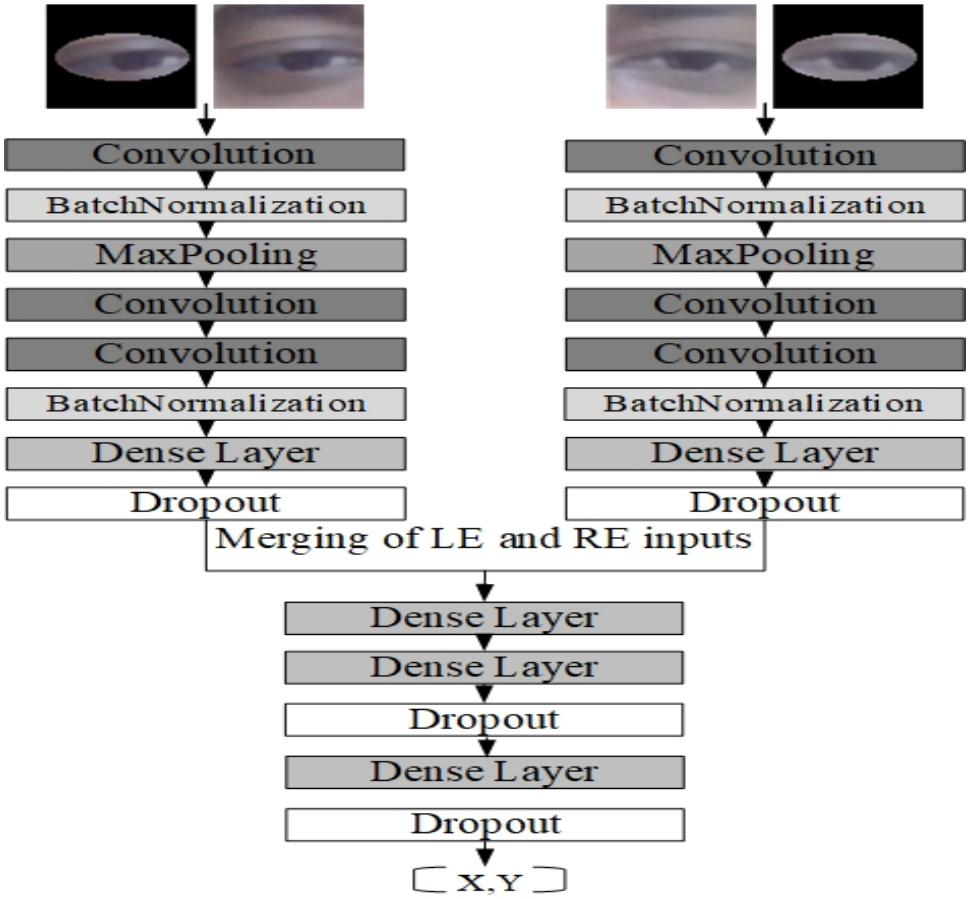


Figure 5.4: Architecture that takes both left and right eyes (with or without mask) as a single input.

5.2.4 Parameter Tuning

This section presents various parameters that are tuned during the training of the CNN network. We experimented with different parameter values and utilized the Keras tuner library [110] to find the optimal ones. We carefully monitored the learning rate, dropout, and various kernel sizes for both facial and eyes. We kept the number of filters (kernels) in the convolutional layers and the number of neurons in the dense layer's constant throughout our experiments.

We examined various dropout values and filter sizes for each learning rate in that experiment. After extensive parameter tuning, we successfully re-trained the model using the optimal values. Table 4.2 displays the list of tuned parameters along with their corresponding values. For example, a range of learning rate values were selected, starting from 0.1 and going down to 0.0001. Similarly, the dropout rate was varied, ranging from 0.1 to 0.5. For the face experiment, we chose filter sizes of 3×3, 5×5, and 7×7. As for the eye experiments, we selected filter sizes of 3×3 and 5×5 . We did not include a 7×7 kernel for eyes as the size of the eye image was 64×64 pixels. We set the number of filters for the first layer at 32, for the second layer at 96, for the third layer at 160, and

the number of dense units at 64. Table 4.2 shows the number of filters in each layer: the first layer is represented by ConV1, the second layer by ConV2, and the third layer by ConV3.

Table 5.1: List of networks' parameters together with their values.

Parameter Name	**Values**
Learning Rate	(0.1, 0.01, 0.001, 0.0001)
Dropout	(0.1, 0.2, 0.3, 0.4, 0.5)
Kernel Size	(3×3), (5×5), (7×7)
ConV1 Filter No.	32
ConV2 Filter No.	96
ConV3 Filter No.	160
Dense Unit	64

5.2.5 Results

This section assesses the CNN network's performance on the person-specific dataset for gaze estimation. The models were trained using 80% of the data and tested using the remaining 20%. The experiments were conducted on a desktop computer equipped with an AMD Ryzen 7 3700X 8-core processor and 16 GB of RAM. In addition, all models were constructed using the TensorFlow library. The quality of the predictions made by the trained models was assessed by calculating the Mean Absolute Error (MAE) value in pixels.

Training a model on a local machine for eye and face on a desktop processor proved to be quite time-consuming. The model using eye images took an average of 12 hours to train, while the model using face images took over 48 hours. Nevertheless, when the model was transferred to a machine equipped with a graphics processing unit (GPU) for training, there was a notable decrease in training time. The eye models were trained in a remarkably short time of just 30 minutes, while the face models took a slightly longer but still impressive average of 1 hour and 30 minutes. Considering the time difference between training the model on a local machine and a machine with GPU, it was a wise decision to shift the model training to an external source. This move significantly decreased the training time. To use the suggested method on a laptop or desktop, person need to gather images in multiple sessions using the DataCollector application and then train the model with them. Training the model on the images initially may require time, but once it's trained, it becomes capable of gaze estimation.

Our primary objective for this experiment was to discover the most effective model for predicting a person's gaze using low-quality images. Two different experiments were conducted using the same dataset: one where the neighborhood of an eye was not masked,

and another where it was masked. The OpenCV library was used to mask the eye region. All networks were trained for up to 100 epochs. The results for both experiments are displayed with varying learning rates, dropout values, and filter sizes. Tables 5.2–5.5 display the pixel errors for the left and right eyes, both with and without a mask.

Table 5.2: Results of LE and and RE with and without a mask (learning rate = 0.0001).

Exp. Name	LE	LE	LE	LE	LE	RE	RE	RE	RE	RE
Filter Size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**41.24**	42.55	44.92	50.30	55.37	**40.89**	42.97	46.83	48.39	47.37
Exp. Name	LE	LE	LE	LE	LE	RE	RE	RE	RE	RE
Filter Size	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**38.20**	41.68	47.37	47.58	52.39	**36.01**	40.47	41.93	45.83	47.97
Exp. Name	LEM	LEM	LEM	LEM	LEM	REM	REM	REM	REM	REM
Filter Size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**50.26**	55.87	58.08	62.01	64.33	52.57	**51.86**	49.12	55.58	59.89
Exp. Name	LEM	LEM	LEM	LEM	LEM	REM	REM	REM	REM	REM
Filter Size	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**44.32**	48.26	50.07	52.77	58.54	**42.34**	45.34	49.28	51.78	51.13

The first experiment compared the error for eyes with and without masking as input to the network to see if masking eye-neighboring pixels affects results. The best results are presented in Tables 5.2–5.5 for different learning rates and filter sizes, with respect to both eyes with and without the mask. All the results are presented as MAE in pixels.

Similarly, the second experiment was intended to compare the results of the combination of both eyes with and without a mask. The best results for different learning rates and filter sizes are presented in Table 5.6. Finally, the results for the full face as a single input are shown in Table 5.7 with different learning rates and filter sizes.

Tables 5.2–5.5 indicate that the network performed well in both single-eye experiments, with and without a mask, when using a learning rate of 0.0001, a filter size of 5x5, and a dropout rate of 0.1. The best results are displayed in bold. In our analysis, it was observed that the network with a learning rate of 0.1 exhibited the poorest performance. Based on this experiment, it becomes clear that the learning rate plays a crucial role in determining the performance of the network.

The objective of the second experiment was to determine if it was possible to achieve comparable results to using single-eye input by using two-eyed images as input in a more

Table 5.3: Results of LE and RE with and without a mask (learning rate = 0.001).

Exp. Name	LE	LE	LE	LE	LE	RE	RE	RE	RE	RE
Filter Size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**43.42**	45.04	50.09	49.35	53.40	**41.85**	44.15	47.63	47.27	54.64
Exp. Name	LE	LE	LE	LE	LE	RE	RE	RE	RE	RE
Filter Size	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**42.10**	46.86	47.37	52.48	57.46	**37.74**	41.95	45.10	43.78	48.53
Exp. Name	LEM	LEM	LEM	LEM	LEM	REM	REM	REM	REM	REM
Filter Size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**55.53**	56.36	59.48	62.03	61.07	**52.58**	53.98	58.14	59.59	63.36
Exp. Name	LEM	LEM	LEM	LEM	LEM	REM	REM	REM	REM	REM
Filter Size	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**48.93**	50.94	54.27	51.90	57.44	**43.95**	48.52	51.76	52.02	56.79

complex network. We discovered a comparable pattern to the single eye: the most crucial parameter after the filter size is a small learning rate. Unfortunately, the results did not meet expectations compared to the single eye, and the training process took significantly more time than in the single-eye experiments.

The last experiment involved using full-face images as input to the network to predict gaze. The findings for full-face images were unexpectedly good and comparable to eyes. In our experiment, we found that using a 7 × 7 filter for the face image produced the best results. This indicates that choosing for a larger filter size is more effective than using a smaller filter size, such as 3 × 3.

In addition, we conducted experiments using varying percentages of data in the training set to observe the impact of changing the amount of training data on accuracy. Table 5.8 represent all experiments using a 3 × 3 filter size. It is evident that as the percentage of data in the training set was reduced, there was a corresponding decrease in accuracy. We utilized different percentages of images for training purposes, specifically 80%, 60%, 40%, and 20%.

Based on the data presented in Table 5.8, it is evident that the results obtained using 20% of the data are the worst. The reason for this is that the training images (1880) are fewer in number compared to the test images (2360). We conducted an additional experiment using a lower dropout rate (0.005) to assess whether there would be any additional enhancements in accuracy. Unfortunately, it was discovered that the accuracy

Table 5.4: Results of LE and RE with and without a mask (learning rate = 0.01).

Exp. Name	LE	LE	LE	LE	LE	RE	RE	RE	RE	RE
Filter Size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**47.13**	49.59	50.53	52.64	55.82	51.11	**51.03**	57.15	57.17	54.79
Exp. Name	LE	LE	LE	LE	LE	RE	RE	RE	RE	RE
Filter Size	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**44.15**	50.69	49.53	54.63	61.59	**44.26**	48.73	51.54	50.94	58.63
Exp. Name	LEM	LEM	LEM	LEM	LEM	REM	REM	REM	REM	REM
Filter Size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**55.77**	60.04	65.93	69.06	66.52	**54.09**	66.11	59.81	61.64	68.88
Exp. Name	LEM	LEM	LEM	LEM	LEM	REM	REM	REM	REM	REM
Filter Size	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**52.95**	67.42	72.61	63.52	66.90	**52.66**	59.46	74.77	67.51	69.32

Table 5.5: Results of LE and RE with and without a mask (learning rate = 0.1).

Exp. Name	LE	LE	LE	LE	LE	RE	RE	RE	RE	RE
Filter Size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**95.94**	280.29	272.36	354.21	279.89	**93.06**	389.72	308.64	184.71	250.27
Exp. Name	LE	LE	LE	LE	LE	RE	RE	RE	RE	RE
Filter Size	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	140.92	**131.15**	183.77	260.94	369.40	**111.65**	172.65	291.42	389.73	277.74
Exp. Name	LEM	LEM	LEM	LEM	LEM	REM	REM	REM	REM	REM
Filter Size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**216.23**	203.84	196.29	248.68	333.17	**104.11**	174.99	181.81	395.73	382.74
Exp. Name	LEM	LEM	LEM	LEM	LEM	REM	REM	REM	REM	REM
Filter Size	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**96.16**	173.60	184.90	327.07	366.27	**150.37**	278.38	319.84	350.18	393.51

did not show any further improvement.

Table 5.6: Results of BE with and without a mask.

	Learning Rate = 0.0001					**Learning Rate = 0.001**				
Exp. Name	BE	BE	BE	BE	BE	BE	BE	BE	BE	BE
Filter Size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**53.08**	57.85	62.97	61.37	70.59	59.89	58.06	61.05	67.23	69.32
Filter Size	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	53.85	53.46	56.36	64.00	70.28	**51.18**	55.44	56.72	64.06	62.69
Exp. Name	BEM	BEM	BEM	BEM	BEM	BEM	BEM	BEM	BEM	BEM
Filter Size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	66.39	66.02	73.64	73.45	85.30	64.19	71.90	76.04	78.35	82.71
Filter Size	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**60.83**	62.75	63.90	69.83	73.56	**62.06**	64.87	66.76	70.06	71.80
	Learning Rate = 0.01					**Learning Rate = 0.1**				
Exp. Name	BE	BE	BE	BE	BE	BE	BE	BE	BE	BE
Filter Size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	102.16	91.11	115.09	113.72	126.63	3680.40	368.50	373.74	368.68	368.62
Filter Size	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**94.33**	106.30	125.27	102.14	121.40	370.49	370.51	370.49	372.40	**370.46**
Exp. Name	BEM	BEM	BEM	BEM	BEM	BEM	BEM	BEM	BEM	BEM
Filter Size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	199.04	153.00	173.98	268.43	285.19	368.70	368.75	368.66	368.67	368.70
Filter Size	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**127.14**	180.67	175.08	233.90	337.41	366.14	**366.10**	368.10	**366.10**	366.90

Given the prevalence of presenting gaze errors in degrees rather than pixels, we have recalculated the errors to provide a more accurate representation of our model's performance. Once the distance of a person from the camera and the screen size are known, it is possible to convert the error in pixels to degrees using the following formula:

$$Error_{deg} = \tan^{-1} \frac{E_{cm}}{Dist}, \tag{5.2}$$

Where E_{cm} represents the error in centimetres resulting from the conversion from

pixels, and Dist denotes the distance in centimetres between the screen and the viewer's eyes. The distance between the screen and the person can vary, as there is currently no way to adjust it without a chin rest. Therefore, we could only make an approximation of this distance. Following the recalculation, the minimum error for LE is approximately 1.45 degrees. The minimum error for the RE is approximately 1.37 degrees. The minimum error for both eyes (BE) is approximately 1.98 degrees. With regards to full-face (FF), the minimum error is approximately 1.14 degrees. Figure 5.5 shows the error of the top models in terms of pixels.

Table 5.7: Results of full-face.

	Learning Rate = 0.0001					**Learning Rate = 0.001**				
Exp. Name	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
Filter Size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	39.38	42.29	43.26	47.76	64.33	40.24	45.49	54.17	54.42	64.33
Filter Size	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	37.87	43.04	39.04	49.59	59.02	**37.75**	55.84	62.60	61.84	77.63
Filter Size	7 × 7	7 × 7	7 × 7	7 × 7	7 × 7	7 × 7	7 × 7	7 × 7	7 × 7	7 × 7
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**30.09**	34.12	43.60	47.17	52.73	55.52	60.83	54.71	54.03	59.80
	Learning Rate = 0.01					**Learning Rate = 0.1**				
Exp. Name	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
Filter Size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	47.48	76.70	273.03	81.23	90.77	**380.65**	380.65	380.64	380.65	380.65
Filter Size	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5	5 × 5
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**43.24**	97.59	169.95	57.20	80.20	380.65	380.65	380.65	380.65	380.65
Filter Size	7 × 7	7 × 7	7 × 7	7 × 7	7 × 7	7 × 7	7 × 7	7 × 7	7 × 7	7 × 7
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	53.19	77.66	78.16	64.05	82.59	380.65	380.65	380.65	380.65	380.65

Where E_{cm} represents the error in centimetres resulting from the conversion from pixels, and Dist denotes the distance in centimetres between the screen and the viewer's eyes. The distance between the screen and the person can vary, as there is currently no way to adjust it without a chin rest. Therefore, we could only make an approximation of this distance. Following the recalculation, the minimum error for LE is approximately 1.45 degrees. The minimum error for the RE is approximately 1.37 degrees. The minimum

error for both eyes (BE) is approximately 1.98 degrees. With regards to full-face (FF), the minimum error is approximately 1.14 degrees. Figure 5.5 shows the error of the top models in terms of pixels.

Table 5.8: Results of different training sets on face data.

	Learning Rate = 0.0001					**Learning Rate = 0.001**				
Exp. Name	FF(80)	FF(80)	FF(80)	FF(80)	FF(80)	FF(80)	FF(80)	FF(80)	FF(80)	FF(80)
Filter Size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	41.88	**41.87**	43.81	47.89	60.35	**45.90**	52.25	52.17	59.46	75.87
Exp. Name	FF(60)	FF(60)	FF(60)	FF(60)	FF(60)	FF(60)	FF(60)	FF(60)	FF(60)	FF(60)
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**42.88**	44.94	49.04	53.55	60.35	**47.33**	54.69	55.17	60.23	74.87
Exp. Name	FF(40)	FF(40)	FF(40)	FF(40)	FF(40)	FF(40)	FF(40)	FF(40)	FF(40)	FF(40)
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	49.37	**48.70**	54.30	52.43	58.83	**50.60**	59.03	63.97	57.37	70.62
Exp. Name	FF(20)	FF(20)	FF(20)	FF(20)	FF(20)	FF(20)	FF(20)	FF(20)	FF(20)	FF(20)
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**56.17**	59.29	67.03	68.24	71.01	63.36	**61.89**	76.18	74.96	71.11
	Learning Rate = 0.01					**Learning Rate = 0.1**				
Exp. Name	FF(80)	FF(80)	FF(80)	FF(80)	FF(80)	FF(80)	FF(80)	FF(80)	FF(80)	FF(80)
Filter Size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**58.01**	81.48	71.18	125.93	150.27	**377.40**	377.40	377.41	377.40	377.41
Exp. Name	FF(60)	FF(60)	FF(60)	FF(60)	FF(60)	FF(60)	FF(60)	FF(60)	FF(60)	FF(60)
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**59.21**	83.88	70.45	127.45	148.76	377.41	**377.40**	377.41	377.40	377.41
Exp. Name	FF(40)	FF(40)	FF(40)	FF(40)	FF(40)	FF(40)	FF(40)	FF(40)	FF(40)	FF(40)
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**53.98**	114.24	125.67	82.95	136.86	382.99	**382.99**	382.99	382.99	382.99
Exp. Name	FF(20)	FF(20)	FF(20)	FF(20)	FF(20)	FF(20)	FF(20)	FF(20)	FF(20)	FF(20)
Dropout	0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
Error	**61.68**	112.39	263.82	139.86	389.31	**465.23**	465.34	464.79	465.34	465.34

Given the prevalence of presenting gaze errors in degrees rather than pixels, we have recalculated the errors to provide a more accurate representation of our model's performance. Once the distance of a person from the camera and the screen size are known, it is possible to convert the error in pixels to degrees using the following formula:

$$Error_{deg} = \tan^{-1} \frac{E_{cm}}{Dist}, \tag{5.3}$$

Where E_{cm} represents the error in centimetres resulting from the conversion from

pixels, and Dist denotes the distance in centimetres between the screen and the viewer's eyes. The distance between the screen and the person can vary, as there is currently no way to adjust it without a chin rest. Therefore, we could only make an approximation of this distance. Following the recalculation, the minimum error for LE is approximately 1.45 degrees. The minimum error for the RE is approximately 1.37 degrees. The minimum error for both eyes (BE) is approximately 1.98 degrees. With regards to full-face (FF), the minimum error is approximately 1.14 degrees. Figure 5.5 shows the error of the top models in terms of pixels.

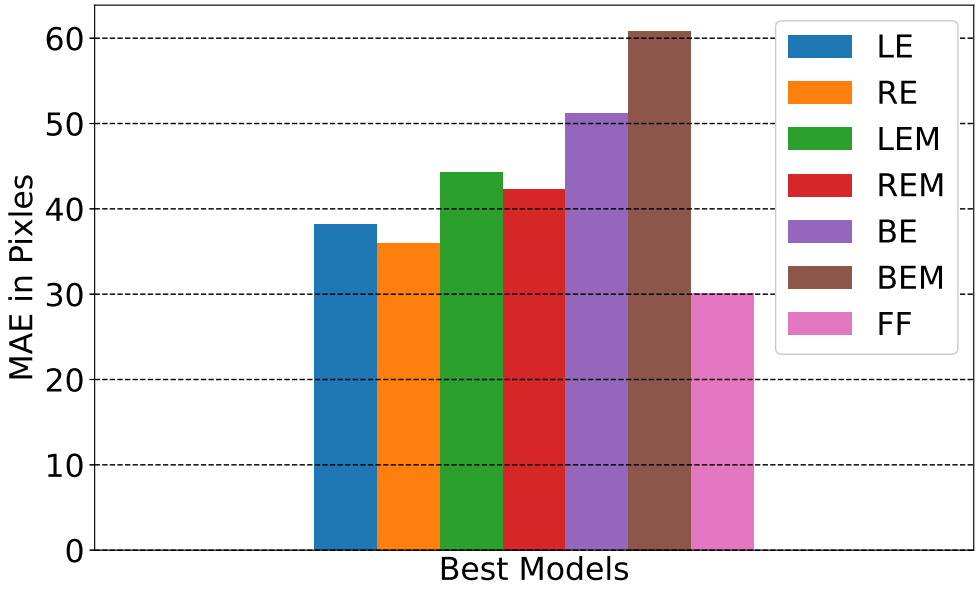


Figure 5.5: MAE in pixels for best models.

5.2.6 Discussion

The experiments demonstrate the potential to achieve performance comparable with commercial eye trackers, as mentioned in the introduction, ranging from 0.5 to 2 degrees. The error for RE images was 1.37 degrees, while for FF images it was 1.14 degrees. These results outperformed other webcam-based models found in the literature. It is interesting to note that in paper [118], the authors were able to achieve an error of 3.14 degrees on the MPIIGaze dataset. Similarly, in [3], the authors achieved an error of 3.94 degrees, while in [44], the reported error was 4.8 degrees. In addition, in [119], the authors were able to achieve a measurement of 4.5 degrees error in MPIIGaze and 10.3 degrees for the EYEDIAP dataset and a margin of error on MPIIGaze. In a related study [6], the authors were able to achieve an error of 2.8 degrees on the MPIIGaze dataset and 3.05 degrees on the EYEDIAP dataset. In a recent study, researchers found that there was an error of

3.34 degrees in the EYEDIAP dataset, as mentioned in [113].

Based on our findings, it is evident that our results surpassed those of previous research. It is important to note that the results were obtained using a model that is specific to one person and only applicable to that person's eye/face images. However, our research has shown that by using data from a standard webcam, we can create a person-specific model that achieves accuracy similar to that of commercial eye trackers. Table 5.9 presents a comparison between our proposed method and other cutting-edge techniques. It is worth mentioning that our method was trained using data from a single user, while other methods utilized data from multiple individuals. This comparison demonstrates that training the model with a single user can result in higher accuracy compared to models trained on data from multiple individuals.

Table 5.9: Angular errors of our method in comparison with other state-of-the-art methods.

Method	**Error in Degrees**
FAZE [118]	3.14
L2CS-Net [3]	3.94
RT-Gene [44]	4.8
Deep Pictorial Gaze [119]	4.5
Multi-stream CNN [6]	2.8
Recurrent CNN [113]	3.34
Ours (with RE)	**1.37**
Ours (with FF)	**1.14**

In the first experiment, the network's performance was tested on single eyes with and without a mask. The results, as shown in Tables 5.2–5.5, indicate that masking the area surrounding the eyes did not improve the network's performance compared to eyes without masking. The highest achieved result was for LE, with an MAE of 38.20. This was accomplished using a learning rate of 0.0001, a filter size of 5x5, and a dropout rate of 0.1. The best result for LEM was 44.32 using the identical set of parameters. In a similar way, the best value for RE was found to be 36.01, while for REM it was 42.34, both achieved using the same parameter settings. Through various experiments with single-eye input, it is evident that the network for RE consistently outperforms the others, achieving an impressive result of 36.01. This indicates that the network has the ability to learn effective representations in its middle layers, resulting in slightly better performance compared to LE. On the other hand, we came across some intriguing findings. For instance, in Table 3, we noticed that a 3x3 filter outperformed a 5x5 filter (LE at dropout 0.4). Additionally, REM yielded better results than RE at dropout rates of 0.1, 0.2, and 0.3, when using a filter size of 5x5.

The results obtained from utilizing both eyes were quite satisfactory. For the BE network, the best result was achieved with a learning rate of 0.001, filter size of 5x5, and dropout of 0.1. This resulted in an MAE of 51.18, which outperformed the results obtained with a learning rate of 0.0001 and a filter size of 3x3. Unfortunately, the results were not as good as those achieved with single images. It seems that the network was more challenging to train for both eyes, likely because of its increased complexity. Using an extremely low learning rate of 0.0001, the network experienced a sluggish convergence, resulting in the inability to reach the global minimum within 100 epochs. Based on the experiments conducted, it was found that the network performed exceptionally well when using a learning rate of 0.0001, a filter size of 5x5, and a dropout rate of 0.1.

One of the most unexpected findings from the experiments was the outcome when the model utilized full-face images (FF) as input to the network. The CNN for FF achieved a value of 37.87 MAE, which is comparable to the single-eye input of 36.01 (RE at 0.0001 learning rate, filter size 5x5, and dropout 0.1). In addition, the network achieved an improved MAE score of 30.09 when trained with a larger filter size of 7 x 7. This intriguing discovery demonstrates that utilizing a larger filter size yields a substantial enhancement. The reason for this is that the face images were larger, which allowed the network to use a larger filter size to capture more significant features. This ultimately led to improved results for the network. Just like other models, the one with a learning rate of 0.1 performed the worst. It consistently produced similar values for all filter sizes. Despite the findings, a positive outcome for FF images indicates the potential to accurately anticipate a person's gaze using an entire face image. One possible explanation for the surprisingly good performance of the FF model is the consistent use of the same face from the same person during both the training and testing stages.

5.3 Conclusion

Our experimental results indicate that the model developed for a single person achieves significantly lower errors compared to general models trained on data from multiple users. We trained various CNN models using different parameter configurations on a dataset of images captured from a laptop webcam. The findings in this chapter demonstrate the effectiveness of a well-optimized CNN architecture in achieving good gaze estimation results using a standard camera. These findings have significant implications for real-world applications.

The presented results are promising, yet there are many limitations correlated with this study. For example, we showed results for only one person, and we are still determining how well CNN will perform for other people since different people have different eye and facial appearances, and different image appearances affect network performance. We also performed all the experiments on a shallow CNN.

In the next chapter, we will overcome these limitations by developing and testing personalized models for each user, along with a comprehensive model that can be applied to all users. Our goal is to broaden our data collection to encompass a wider range of subjects. This will enable us to analyze both generalized and individualized models using transfer learning techniques. In upcoming chapter, we will delve into the transfer learning approach, where models are initially trained on extensive datasets and subsequently fine-tuned to adapt to individual users.

5.4 Contributions

After conducting all experiments, the second hypothesis was confirmed that a model trained on low-quality eye and face images from a standard webcam, tailored specifically for an individual user, can achieve gaze estimation accuracy comparable to that of state-of-the-art eye trackers. Our experimental results demonstrated that person-specific models, even when using low-resolution images, were capable of delivering strong gaze estimation performance.

By fine-tuning model parameters for individual users, we successfully minimized the performance gap between webcam-based models and professional eye-tracking systems, thus validating the effectiveness of personalized training approaches for gaze estimation.

In addition to confirming our hypotheses, our experiments provided important insights regarding the use of face images for gaze estimation. Although face-based models proved more challenging to train compared to eye-based models, they demonstrated notable advantages in person-specific scenarios. In particular, our results from Chapter 5 showed that when the same person's face is used for both training and testing, face images actually outperformed eye-only models in terms of gaze estimation accuracy. This finding highlights the potential of using full face information to capture subtle contextual cues that contribute to more precise gaze prediction, especially in personalized models. Therefore, while training complexity may be higher, face-based models offer significant performance benefits in individualized settings, supporting their practical application in real-world gaze tracking solutions.

Chapter 6

Exploring Transfer Learning for Gaze Estimation: A Study on Model Adaptability

In the previous chapter, we introduced a person-specific gaze estimation model that achieved performance comparable to a generalized model. This chapter explores the use of transfer learning in gaze estimation, focusing on the development of personalized models specific for individual users. Our approach involves the collection of gaze data using standard laptop webcams, intended to work effectively within resource-limited settings, thereby enhancing accessibility and affordability. This chapter presents a comparative analysis of models using transfer learning with those that do not utilize pre-trained models. The analysis includes both eye and face images, evaluating their performance across datasets of different sizes. Our findings show that both methods produce comparable results; however, transfer learning presents notable advantages, including faster convergence, reduced computational expenses, and enhanced stability when working with smaller datasets. The results demonstrate the significant implications of transfer learning in gaze estimation, especially in scenarios involving limited datasets. This approach presents a more efficient and scalable solution, enabling real-time applications across domains such as human-computer interaction (HCI), assistive technologies, and personalized user experiences. This chapter is based on a paper which has been submitted to a journal and is waiting for publication.

6.1 Introduction

Traditionally, gaze estimation techniques required tightly controlled settings, such as fixed head positions or controlled lighting conditions, to accurately predict gaze direction. This constrained the technology to laboratory environments and specialized hardware setups. Recent efforts have aimed to address these limitations by utilizing standard web-

cams built into laptops and desktops, providing a more accessible solution that does not require additional hardware. However, such approaches often face challenges related to image quality, lighting variability, and camera angles, particularly when using standard consumer cameras.

In this context, transfer learning has emerged as a promising technique to improve the efficiency of gaze estimation models. Rather than training models from scratch, transfer learning enables the reuse of knowledge learned from large-scale datasets, which can then be fine-tuned for specific tasks like personalized gaze estimation. This approach not only accelerates model convergence but also reduces the computational resources required, making it a practical solution for deploying gaze estimation systems on devices with limited hardware capabilities. By leveraging pre-trained models, we can overcome some of the inherent challenges of gaze estimation using low-resolution, low-quality images from standard webcams, while still achieving results comparable to models trained from scratch.

In this chapter, we investigate the application of transfer learning in developing personalized gaze estimation models. Our goal is to explore how transfer learning can enhance the performance of gaze estimation while minimizing the need for extensive computational resources and specialized equipment. We focused on creating a dataset from scratch and fine-tune pre-trained deep learning models for gaze estimation task. Through this work, we aim to contribute to the broader adoption of gaze estimation technology by developing methods that are resource-efficient, accessible, and capable of functioning in diverse real-world scenarios.

Furthermore, this chapter investigates the performance of gaze estimation model under limited data scenarios using dataset containing 500, 400, 300, 200 and 100 images. This approach differs from our previous approach trained on a much larger dataset of approximately 12000 images collected from a single user, achieving excellent results [9]. The smaller dataset used in this study results in higher MAE values, as expected, but the objective is not to exceed the previous findings in terms of accuracy. Instead, we aim to explore the feasibility of gaze estimation in situations where training data is limited; as this is a typical situation in real world application, where collecting large dataset is not feasible. Our findings demonstrate that although accuracy decreases with smaller dataset, it is still possible to achieve satisfactory performance, highlighting the potential of implementing gaze estimation system in environments with limited resources. The trade-off between dataset sizes and model performance provide valuable insights for people working with limited data. To the best of our knowledge, previous studies have not explored the gaze estimation under limited data scenarios.

The main contributions of this chapter are as follows:

- **Comparative Model Evaluation:** We trained and assessed gaze estimation models for all participants, utilizing both transfer learning technique and model without

pre-trained weights. This comparative analysis offers valuable insights with respect to the effectiveness of transfer learning in improving model performance and adaptability.

- **Demonstration of Transfer Learning Advantages:** Our findings illustrate the advantages of employing transfer learning in gaze estimation, especially in contexts where data is limited. The findings indicate that transfer learning models achieve accuracy levels similar to those trained without pre-trained weights, while also demonstrating improved convergence and generalization among individuals.

- **Guidance for Practical Applications:** This study illustrates practical recommendations for the application of gaze estimation technologies in real-world contexts, emphasizing the role of transfer learning in enabling quicker deployment and customization across diverse fields such as healthcare, virtual reality, and assistive technologies.

- **Contribution to the Understanding of Model Performance:** This study advances our understanding of the efficient utilization of transfer learning in gaze estimation task. The inclusion of empirical evidence about model performance across varying dataset sizes significantly contributes to the present debate surrounding the optimization of machine learning techniques in gaze estimation.

- **Dataset Collection:** A dataset "WebGazeLowRes: A Webcam-Based Low-Resolution Gaze Dataset" was created, where WebGaze highlights data is webcam-based and focus on gaze estimation and LowRes represents the low resolution nature of images. The dataset consists of gaze data from 19 participants, providing a valuable resource for the training of deep learning models.

6.2 Methods and Materials

This section outlines the key steps that are a part of the research process. First, we outline the methodology used in data collection, emphasizing the methods utilized to gather data that would enhance the effectiveness of gaze estimation models. Next, we provide the data preprocessing techniques for transforming the raw data into a suitable format for deep learning models. Following this, we present a detailed overview of the datasets used in this study. Finally, we provide an overview of neural network architectures that were used in this study.

6.2.1 Data Collection

Data collection is a crucial aspect of deep learning, as the quality of the data used for training significantly impacts the overall performance of neural networks. In this study, we used simple Python based desktop EyeTrackerDataCollector application to capture images from a standard webcam. Participants were asked to look at the different parts of the screen and click on the screen. This simple approach allowed us to gather gaze locations data across the screen along with their face images. From the raw images, left eye, right eye and face regions were cropped along with their gaze coordinate on screen. The simplicity of this setup – requiring only a built-in or external webcam positioned at the top of the screen – enabled us to collect data under real-world conditions, rather than in a controlled laboratory environment. An abstract setup for data collecting is shown in Figure 6.1, where a participant looks at various points on the screen. The application captures the participant's face and gaze coordinates when they click on a screen.

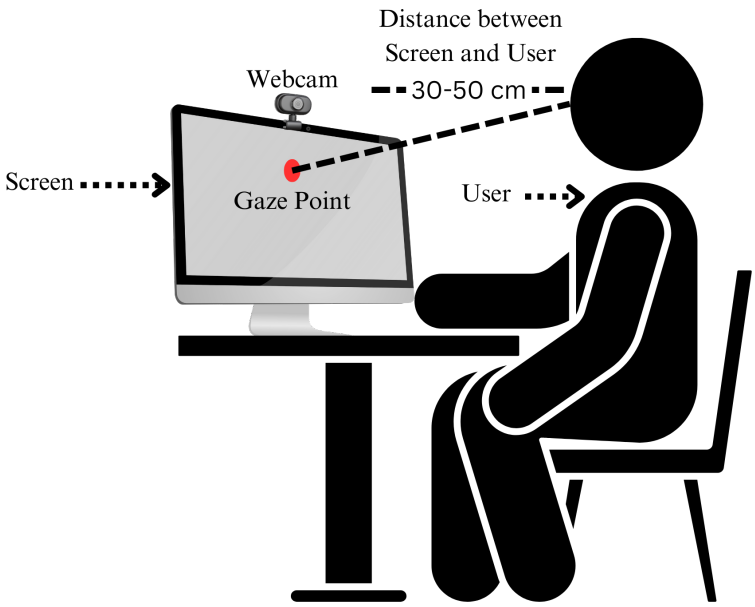


Figure 6.1: Basic data collection setup.

The data collection process involved participants sitting at a distance of approximately 30 to 50 cm from laptop screens, with the camera positioned at eye level. Total of 19 participant were involved in data collection. During the data collection we paid careful attention to lighting conditions, as poor lighting could result in low-quality images – either too dark or overexposed – which would negatively impact the network's training process. Each session involved the participants clicking on a series of points distributed across the screen. For every click, a series of images was captured along with the corresponding screen coordinates.

Characteristics of the Dataset

The collected dataset exhibits substantial diversity in terms of ethnicity, as participants involved in the data collection were from various regions, including Asia, Africa, and Europe. To further enhance this diversity, data were gathered across multiple sessions (2–3 per participant) conducted in different environments, under varying lighting conditions, and at different times of day. This approach ensured that the dataset captured a wide range of natural variations, thereby improving the robustness and generalizability of the trained models. Each session lasted between one and two minutes, with participants clicking on 54 points. Participants were free to click anywhere on the screen but encouraged to evenly distribute their clicks across the entire screen. The resulting dataset provided a set of images, allowing for an evaluation of the neural network models used in this research. The images were collected on laptop with a 1920 $\times$ 1080 screen resolution. The resolution of the captured images from EyeTrackerDataCollector application were different across participants. To ensure consistency and facilitate analysis, all images were resized to a uniform resolution during the preprocessing step. Specifically, eye images from all participants were resized to 64 $\times$ 64 pixels, while face images were resized to 224 $\times$ 224 pixels.

6.2.2 Data Preprocessing

Once the data collection process was complete, the next step was data preprocessing, essential for ensuring the quality of input data for deep learning models. Figure 6.2 outlines the entire preprocessing steps.

Step 1 as shown in Figure 6.2 involved capturing an image from the webcam. In the subsequent step 2, we used the Viola-Jones (VJ) classifier with a Haar cascade to detect the face in each image [148]. The VJ classifier was chosen for its ease of use and efficiency in detecting faces. Any images where faces were misclassified were manually reviewed and removed from the dataset.

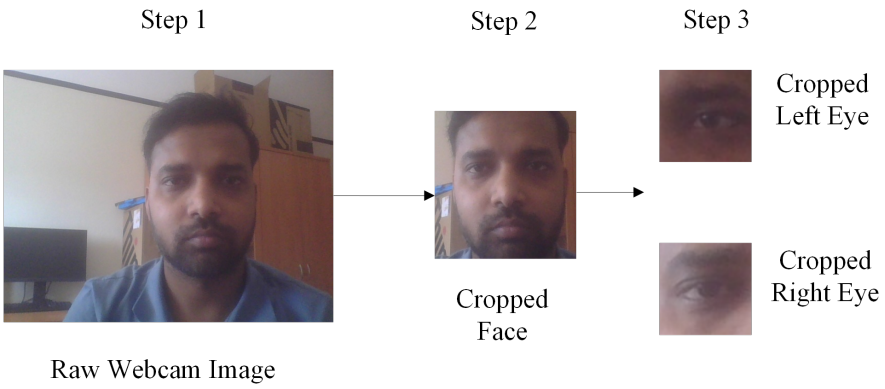


Figure 6.2: Preprocessing steps.

In step 3, a separate Haar cascade classifier [148] was used to detect the left and right eyes from the cropped face image as hows in Figure 6.2. This step minimized the risk of misclassifying other objects as eyes. All eye images were resized to 64 × 64 pixels, and face images were resized to 224 × 224 pixels before being fed into the neural network.

A manual check of all images was conducted to verify their appropriateness. Any instances where the Haar cascade falsely detected eye-like objects or where subjects blinked or closed their eyes during recording were removed to ensure the dataset's quality.

A total of 19 participants were included in the data acquisition process, specifically designed for person-specific gaze estimation. Processes were conducted individually for each participant to facilitate creation of person-specific models. In total 27,413, 29,082 32,395 images were collected for left eye, right eye and face for all participants. An example of the gaze target distribution for an individual participant is presented in Figure 6.3. This figure demonstrates the spatial distribution of the collected points, reflecting the diversity and coverage achieved during the data acquisition process.

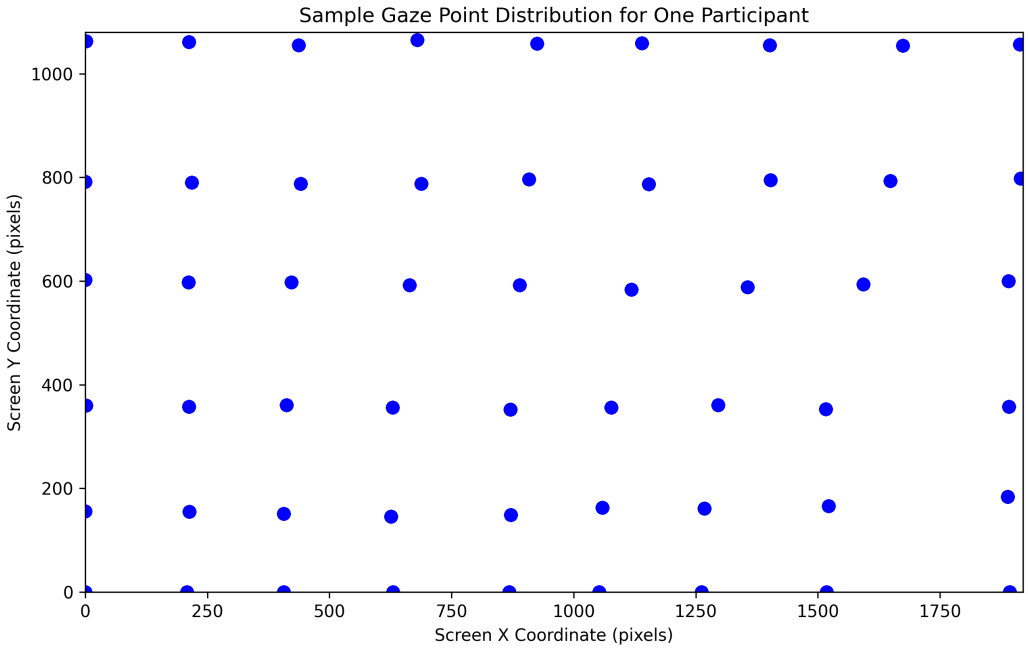


Figure 6.3: Example of points distribution for a single participant, illustrating the spatial arrangement of gaze estimation targets.

An overview of the number of images collected for each participant, categorized by Left Eye, Right Eye, and Face datasets, is provided in Table 6.1.

Table 6.1: Number of images for each participant for Left Eye, Right Eye, and Face datasets.

Participant	Left Eye	Right Eye	Face
1	544	1222	1441
2	2297	2193	2544
3	1387	1841	1929
4	1270	1170	1307
5	1768	1751	1804
6	843	1346	1528
7	1747	1607	2347
8	1231	1819	1858
9	983	1246	1581
10	1442	2556	1789
11	1088	1266	1295
12	1914	1055	1185
13	997	992	1005
14	1999	1928	2142
15	1219	1064	1284
16	960	961	981
17	985	1025	1028
18	432	316	554
19	1277	1644	1873

6.2.3 Network Architecture

This section outlines the network architecture used and evaluated in the study as shown in Figure 6.4. The architecture describes all models that have been trained with same hyper parameters. Each model outputs a vector of two values representing the gaze coordinates on the screen.

The network architecture consists of three convolutional layers, one pooling layer, two batch normalization layers, one fully connected layer, one dropout layer, and an output layer as shown in Figure 6.4. The output layer in the model consists of two neurons responsible for predicting the X and Y coordinates of an individual's gaze on the screen.

The network was trained utilizing the ADAM optimizer [83], we first trained network with different parameters to find the optimal values for best results. After finding the optimal value, we used them to train network for all participants. The Rectified Linear Unit (ReLU) [4] activation function was used in the hidden layers and the fully connected layer. Output layer contains the gaze coordinate of person's gaze. Mean Absolute Error(MAE) was used as loss function during training [155]. Equation 6.1 represents the mathematical

formula of MAE.

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|\,, \tag{6.1}$$

where y_i is the actual value, $\hat{y}_i$ is the predicted value, and N is the total number of examples.

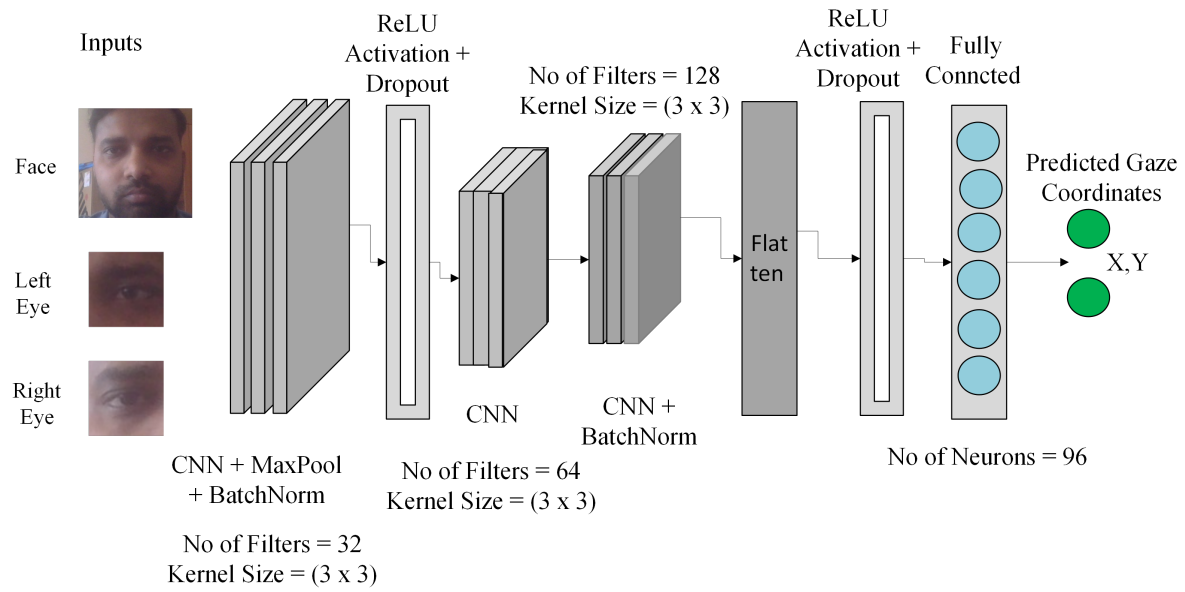


Figure 6.4: Architecture of CNN network which takes the Left Eye, Right Eye or Face images as input and the output layer with two neurons predicts the gaze coordinates.

A simple naming convention was adopted across all participants to ensure consistency in the experiments. To ensure clarity, the experiment involving the left eye was referred to as LeftEye, the experiment involving the right eye referred to as RightEye, and the experiment focused on the face referred to as Face. Furthermore, the transfer learning model is referred as TL, while the model without pre-trained weights is labeled as NT.

6.2.4 Research Methodology

The objective of these experiments was to evaluate the efficiency of transfer learning with respect to training without pre-trained models for gaze estimation using images of the left eye, right eye, and face.

First, a model was trained using the data from all participants for transfer learning approach, excluding the specific participant for whom the pre-trained model weights were later used. After this, the model was fine-tuned specifically for the excluded participant, who had not been included in the original training set for large model. When training a model for a specific particiapnt, participant data was divided into trainset and testset. The evaluation of models was conducted for various training dataset sizes, specifically

500, 400, 300, 200, and 100 images while keeping the same number of images in testset when doing evaluation to ensure a fair comparison. This consistent approach was used to ensure that evaluation of all models was done for same number of images in the testset. It was also ensured that gaze location was equally distributed along with images across training sets and testset to prevent the issue of the model overfitting to specific regions of the screen. It was essential to ensure the equal distribution of gaze points, as ignoring this aspect could result in a model showing strong performance in specific regions of the screen while under performing in others. This balanced approach improved the models' ability to generalize effectively across the entire screen. The Figure 6.5 compares the transfer learning approach with training from scratch for person-specific models.

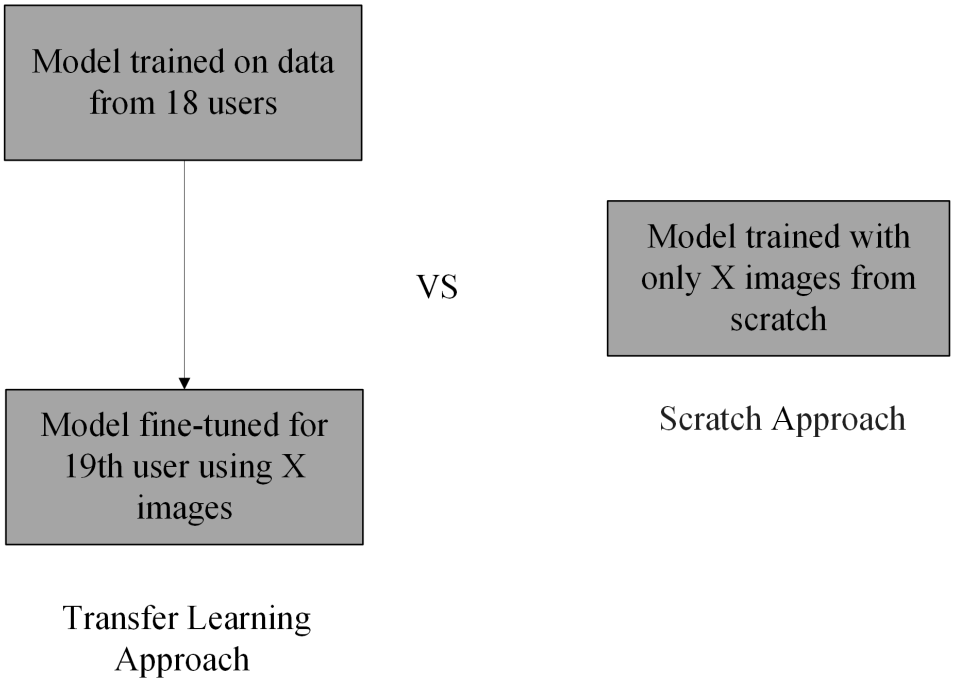


Figure 6.5: Schematic diagram for finetuning the model vs scratch model

The same strategy was used across all participants during the training of models for left eye, right eye, and face images.

The Validation Mean Absolute Error(MAE) was used to evaluate model's performance on unseen test data. The use of this consistent approach helped in fair comparisons among different models, participants, and dataset sizes, which allowed a comprehensive analysis of the efficiency of transfer learning in varying data availability conditions.

6.3 Experiments and Results

This section presents a complete overview of the conducted research, along with the relevant results obtained. It starts with a brief summary of how the results were obtained,

followed by in-depth descriptions of the specific experiments carried out. The last section describes the key findings and presents conclusions derived from the conducted research.

6.3.1 Left Eye Model

The performance for the LeftEye model was determined with different dataset sizes, comparing transfer learning with training without pre-trained models. The mean MAE was used to evaluate performance for each model.

In the LeftEye model, transfer learning demonstrated stable performance across various dataset sizes. The investigation of 500 images revealed an average MAE of 186.83, with a standard deviations(StD) of 41.10 as shown in Table 6.2. This metric showed a gradual increase, reaching 224.57 with a sample size of 100 images and a standard deviation of 45.99.

Table 6.2: Experiments of Left Eye models using Transfer Learning (TL) and Models Without Pre-trained Weights (NT). All values for MAE are given in pixels.

No. of Images	**Experiment with TL**		**Experiment with NT**	
	MAE	**StD**	**MAE**	**StD**
500	186.83	41.10	191.99	45.61
400	190.98	41.15	196.02	45.46
300	197.65	41.77	196.40	44.29
200	206.83	43.59	211.29	45.58
100	224.57	45.99	246.46	50.55

On the other hand, training without pre-trained models resulted in generally higher MAE scores. The MAE for 500 images was 191.99 which increase to 246.46 when checked with 100 images, with a higher standard deviation of 50.55 which we can be observed in Table 6.2. The results indicate that transfer learning models show low error rates and stable performance and even perform better when dataset is small.

A paired t-test [154] was used to determine if there is a statistically significant difference in gaze estimation accuracy between models that used pre-trained weights and and without pre-trained weights. The difference in accuracy was statistically significant ($p < 0.05$) only for 100 images shows not pre -trained models struggle to perform well on small datasets. While the difference did not reach statistical significance for other dataset sizes, transfer learning showed a lower average error for most experiments except for 300 images for left eye. This suggests that although the improvement may not be drastic, transfer learning presents a benefit in practical applications, particularly in scenarios where data resources are limited.

We reported all our results in pixel, however, it is common to presents gaze error in

degrees. If a distance of a person from the screen and the screen size is available (30-50 cm in our case), error in pixels can be calculated in degrees using the following formula:

$$E_{\text{deg}} = \arctan\left(\frac{E_{\text{cm}}}{\text{Dist}}\right), \tag{6.2}$$

where E_{cm} is the error in cm after converting from pixels and $Dist$ is the distance between the screen and the person's eyes in cm. The distance from the screen to the individual is variable due to unconstrained settings. As a result, we were restricted to providing an approximate estimation of this distance. For example, if we convert 186.83 pixels in to degree, first we need to convert pixels error into error in centimeters (cm) using the formula:

$$E_{\text{cm}} = \frac{E_{\text{pixels}}}{\text{PPI}} \cdot 2.54, \tag{6.3}$$

where pixels per inch (PPI) is the pixel density of the screen, which defines number of pixels per inch. The relationship between inches and centimeters is 1 inch = 2.54 cm. To calculate error in degrees for the error in pixels of 186.83, we first convert error in pixels in cm using eq. (6.3) and after using the PPI value of 127 according to our screen, we get the E_{cm} of approximately 3.73 cm. Next, we substitute these values in eq. (6.2), assuming distance between screen and user's eye is 50 cm. This gives the angular error E_{deg} of approximately 4.27°. Similarly, angular error for other values can be calculated using the same approach.

6.3.2 Right Eye Model Performance

For right eye, transfer learning, the MAE started at 164.51 for 500 images, increased slightly to 197.47 with 100 images, as shown in Table 6.3. The steady increase shows the model's capability to maintain stable performance, despite a decrease in dataset size, therefore showing the robustness of transfer learning in scenarios with limited data availability.

On the other hand, models that were not initialized with pre-trained weights showed a sharper increase in MAE, beginning at 172.36 with 500 images and moving to 232.94 with 100 images, as shown in Table 6.3. The rise observed indicates that models lacking pre-trained weights face difficulties in maintaining performance with limited datasets.

The performance of the right eye model shows a trend similar to the left eye, with transfer learning achieving better results across different dataset sizes in compared to models trained without the use of pre-trained models.

A paired t-test was also used for the right eye to determine the significance differences in results of both experiments. The p-values for 500, 400, 300, 200, 100 are 0.476, 0.402, 0.293, 0.105, 0.009 respectively. It can be seen that, p-value for 100 images is below the

standard threshold of 0.05, indicating a statistically significant difference between TL and NT for the right eye model. Similar to left eye, transfer learning models showed a lower average error for all experiments.

Table 6.3: Experiments of Right Eye models using Transfer Learning (TL) and Models Without Pre-trained Weights (NT). All values for MAE are given in pixels.

No. of Images	**Experiment with TL**		**Experiment with NT**	
	MAE	**StD**	**MAE**	**StD**
500	164.51	43.41	172.36	34.94
400	168.94	41.74	178.31	38.13
300	174.49	45.44	185.50	39.24
200	181.89	47.10	200.48	44.79
100	197.47	51.30	232.94	46.03

6.3.3 Face Model Performance

The Face model demonstrated variations when comparing transfer learning to training without pre-trained models as the dataset size reduced, yet both techniques showed comparable results.

The MAE for transfer learning model utilizing 500 images was observed to average 178.03 pixels, while for 100 images, it increased to 235.46 pixels. The standard deviation showed a stable range, consistently falling between 34.04 and 35.11, as shown in Table 6.4.

Nonetheless, the model that was trained without pre-trained weights showed good performance, achieving a MAE of 154.79 pixels with 500 images, which increased to 217.70 pixels with 100 images. The standard deviation has risen to 46.65 from 36.12, indicating an increase in variability, as shown in Table 6.4. This indicates that training without pre-trained models can produce satisfactory outcomes, particularly when large dataset is available. The model without pre-trained wights showed good results for face data as compared to transfer learning. However, transfer learning model demonstrates better stability in term of standard deviation compared to model without pre-trained weights.

Similar to previous experiments, a paired t-test was also used for the Face model to determine the significance of differences in both experiments. For the Face model p value for 500, 400, 300, 200, 100 images are 2.56e-07, 5.25e-07, 6.97e-05, 1,83e-05, 0.002 respectively. All differences occurred to be statistically significant for Face experiments.

The models trained without pre-trained weights showed lower MAE scores compared to transfer learning.At the same time the results of transfer learning are better clustered around the mean. We hypothesize that the reason for the worse results of the transfer learning model in terms of MAE is that it might have challenges with face data due to

the high variability in facial features across the dataset, limiting the model's ability to generalize. On the other hand, eye features show less variation and look more similar, enabling more robust generalization. This may explain why transfer learning is less effective for face data but provides better results for eye data.

Table 6.4: Experiments of Face models using Transfer Learning (TL) and Models Without Pre-trained Weights (NT). All values for MAE are given in pixels.

No. of Images	**Experiment with TL**		**Experiment with NT**	
	MAE	**StD**	**MAE**	**StD**
500	178.03	34.04	154.79	36.12
400	186.63	35.72	164.98	40.62
300	196.33	34.21	173.44	41.63
200	208.55	36.83	185.70	42.78
100	235.46	35.11	217.70	46.65

The overall findings from the left eye, right eye, and face models show that transfer learning provides enhanced performance relative to models that are trained without pre-trained weights (except for the face models), especially in the context of smaller datasets. Models utilizing transfer learning showed enhanced stability, characterized by a more gradual rise in error as the dataset size was reduced.

Models that were not trained with pre-trained knowledge may require a larger dataset to achieve performance levels comparable to those utilizing transfer learning. Additionally, they showed increased variability and decreased reliability in situations where data is limited. This highlights the benefits of transfer learning where quick implementation, adaptability, and effectiveness are crucial, particularly in the context of limited datasets.

Overall, transfer learning demonstrated notable benefits, including enhanced stability, and reduced MAE scores except for face experiments across various dataset sizes. Models that utilized transfer learning demonstrated greater consistency in data-scare environments, whereas models trained without pre-trained weights demonstrated considerably more errors, especially when the number of training images was reduced.

6.3.4 Further Analysis of Model Performance

This section presents additional experiments that access the model performance on decreasing and increasing the number of images in training set. These image counts differ from our initial experiment size which were is 500, 400, 300, 200 and 100 images.

Impact of Decreased Training Dataset Size on Model Performance

In addition to initial experiments, further testing was done to evaluate the model's performance with extremely limited data. Three participants were selected for the experiments, and the performance of the Left Eye, Right Eye, and Face models was evaluated using 50, 40, 30, 20, and 10 images.

Table 6.5: Experiments of Left Eye models using Transfer Learning (TL) and Models Without Pre-trained Weights (NT) with extremely limited number of images. All values for MAE are given in pixels.

No. of Images	**Experiment with TL**		**Experiment with NT**	
	MAE	**StD**	**MAE**	**StD**
50	292.09	29.53	329.70	39.45
40	299.57	42.05	357.40	47.31
30	381.01	46.91	466.71	74.83
20	426.56	34.53	619.72	56.57
10	561.57	57.99	700.59	72.65

Table 6.6: Experiments of Right Eye models using Transfer Learning (TL) and Models Without Pre-trained Weights (NT) with extremely limited number of images. All values for MAE are given in pixels.

No. of Images	**Experiment with TL**		**Experiment with NT**	
	MAE	**StD**	**MAE**	**StD**
50	328.57	54.73	355.69	63.22
40	334.46	62.68	421.94	103.31
30	358.46	72.38	643.26	123.51
20	377.54	47.49	734.47	66.28
10	548.48	78.63	748.44	92.07

The results indicate that, with an extremely low number of images, the models using transfer learning demonstrated superior performance compared to those trained without pre-trained weights. The results shown in Table 6.5 and Table 6.6 indicate that as the number of images decreases, the performance of both models deteriorated. However, the transfer learning model showed better MAE score, but the model without pre-trained weights struggled to learn effectively compared to the transfer learning model. Similar results were observed for face data as well as for those presented in Table 6.7.

Table 6.7: Experiments of Face models using Transfer Learning (TL) and Models Without Pre-trained Weights (NT) with extremely limited number of images. All values for MAE are given in pixels.

No. of Images	Experiment with TL		Experiment with NT	
	MAE	StD	MAE	StD
50	292.65	24.36	359.08	40.17
40	300.79	38.42	456.82	104.66
30	308.59	50.84	544.29	95.90
20	322.71	33.40	614.88	15.13
10	357.87	43.91	634.76	66.36

Impact of Increased Training Dataset Size on Model Performance

In order to evaluate the impact of dataset size on accuracy, we performed additional experiments using increased number of images in the training set. As shown in Table 6.8, an increase number of images in dataset size (873 for the Left Eye model, 803 for the Right Eye model, and 1218 for the Face model) resulted in improved MAE for the Left Eye, Right Eye, and Face models across both training configurations. These results lead us to the conclusion that the results can be further improved by increasing the number of training images.

Table 6.8: Experiments of the Left Eye, Right Eye, and Face models using more images as compared to initial images with Transfer Learning (TL) and Models Without Pre-trained Weights (NT). All values for MAE are given in pixels.

Model	Experiment with TL		Experiment with NT	
	MAE	StD	MAE	StD
Left Eye (873 images)	173.38	27.02	166.91	33.20
Right Eye (803 images)	166.95	34.78	158.60	36.98
Face (1218 images)	160.38	21.49	165.06	28.52

6.3.5 Convergence Analysis

This section provides an analysis of convergence, early stabilization, evaluation of learning curves and impact of dataset amount on convergence of transfer learning models and models without pre-trained weights.

Rate of Convergence

Although the results for transfer learning are not always significantly better, a notable benefit of transfer learning is its ability to quickly reduce validation loss (MAE) in the

early epochs. The graph in Figure 6.6 for the Left Eye model (similarly for the Right Eye and Face models in Figures 6.7 and 6.8) clearly shows that during the initial few epochs, the MAE for the transfer learning approach shows a significant decline, achieving a considerably lower error rate in contrast to the model trained from scratch.

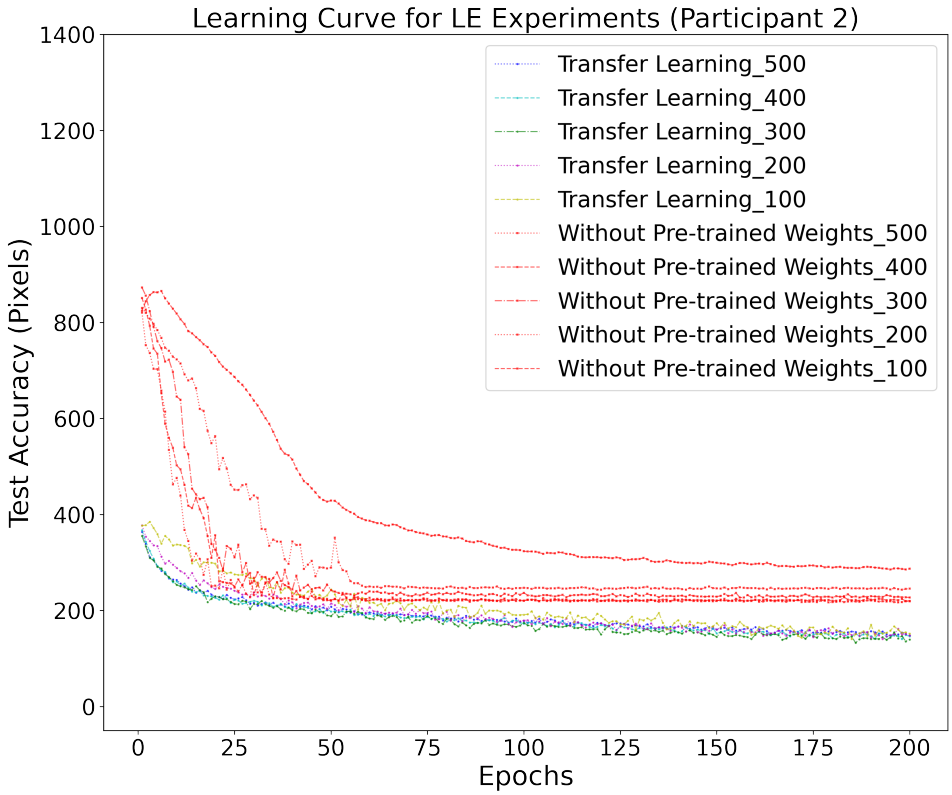


Figure 6.6: Learning curves for participant 2 (Left Eye model – LE) using TL vs NT (red) models. Similar pattern was observed for all other participants. It is visible from curves that transfer learning models converge faster than not pre-trained model. Also error for transfer learning models start from lower MAE compared to not pre-trained weights.

This trend can be explained by the pre-trained weights, which already represent critical features such as eye and facial structures derived from a comprehensive diverse dataset. This enables the model to fine-tune these weights using only a small amount of data and training duration.

In practical implementations, the fast convergence observed is essential as it minimizes both the duration of training and the utilization of computational resources. In contexts like gaze estimation for real-time systems or devices constrained by processing capabilities, early reduction of error results in accelerated deployment and enhanced data efficiency.

In contrast, models without pre-trained weights demonstrate a slower decrease in MAE (graph with red color) in Figure 6.6 6.7,6.8. These models have high error rates in the early epochs and only become better over time; it frequently takes many epochs for the error to start to drastically decrease.

Starting training from scratch requires more epochs to achieve comparable perform-

ance levels than that are achieved through transfer learning. In situations where data is scarce (for instance only 100-200 images), the model without pre-trained weights often reaches a plateau at a higher error rate, as it has to learn all knowledge solely from the raw data, lacking the advantage of pre-existing information.

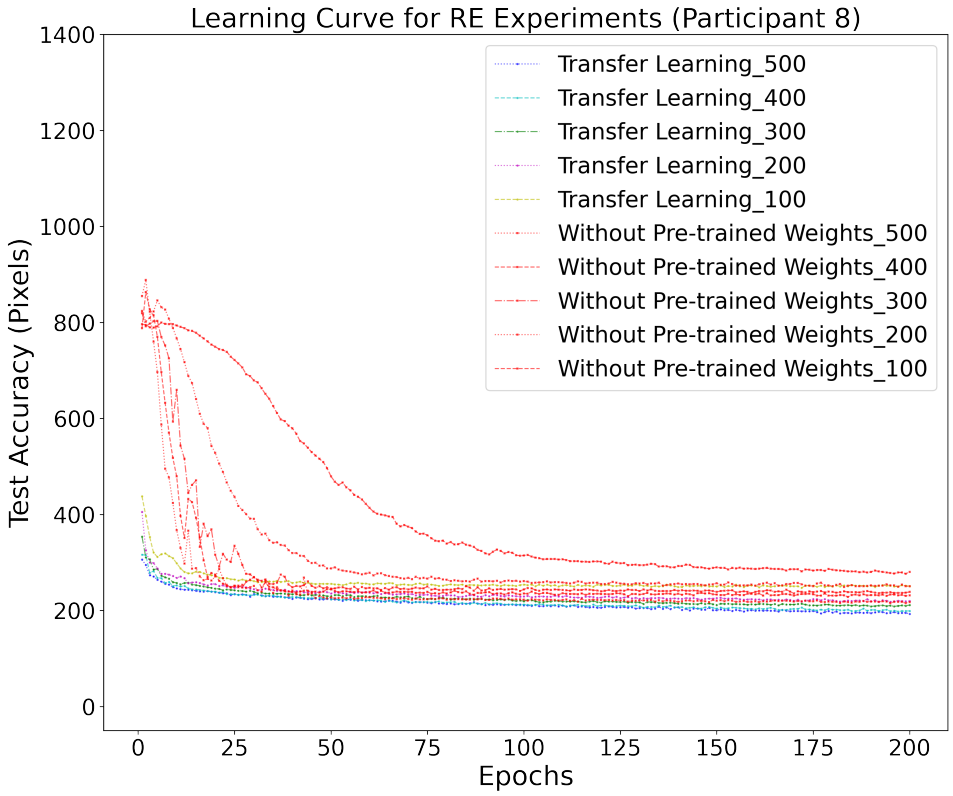


Figure 6.7: Learning curves for participant 8 (Right Eye model – RE) using TL vs NT (red) models.

Early Stabilization

The stabilization of transfer learning models takes place significantly earlier with respect to MAE. The graph in Figure 6.6 shows that following few epochs, the validation error associated with transfer learning stabilizes at a low level, showing minimal fluctuations. It indicates that the model has successfully learned its parameters and is no longer exhibiting significant overfitting or underfitting on the validation dataset.

The early stabilization of models indicates that transfer learning can reach optimal performance in fewer epochs, which is particularly useful for scenarios that require regular model retraining or adaptation (such as accepting new users or varying environments). The ability to stabilize early reduces the need for prolonged retraining efforts.

On the other hand, models trained without pre-trained weights require a longer time to achieve stabilization.The graph (with red color) in Figure 6.6 illustrates that the MAE for without pre-trained models shows continue fluctuations over time. After stabilization

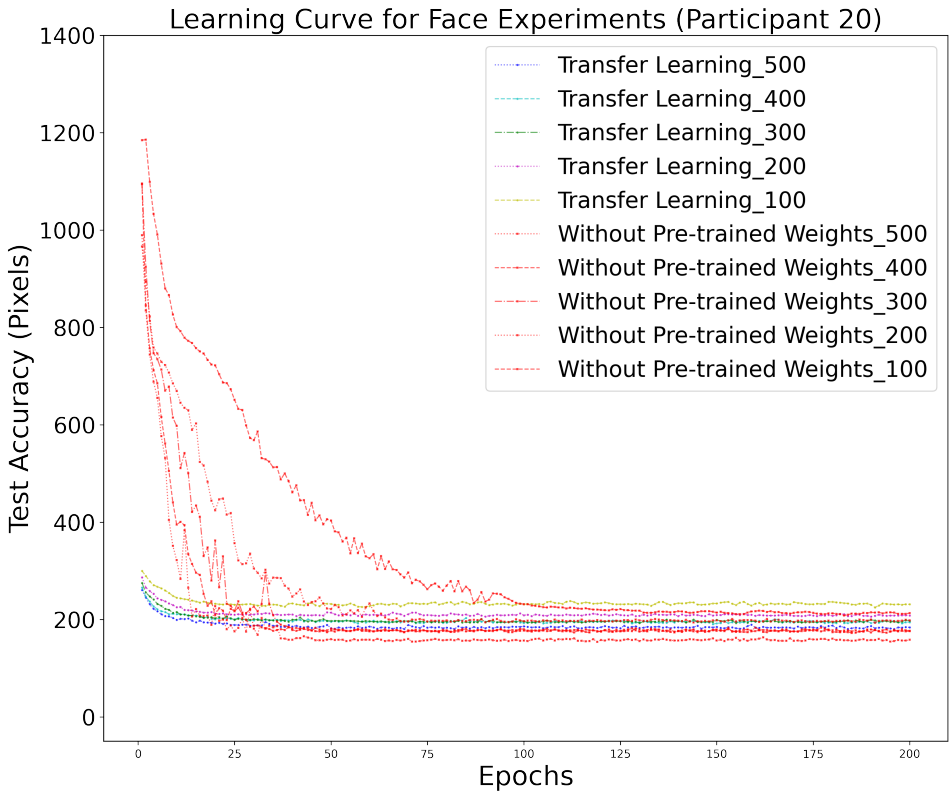


Figure 6.8: Learning curves for participant 19 (Face model) using TL vs NT (red) models.

of the not pre-trained model, it is observed that it maintains a higher error rate except for face model in comparison to the transfer learning model.

The longer stabilization periods in without pre-trained models increase the likelihood of encountering overfitting or under-fitting issues, particularly in scenarios where training data is scarce.

Evaluation of Learning Curves

The trajectory of the learning curve in transfer learning shows an initial decrease, which is then followed by a plateau phase. The steep curve shows that transfer learning give ability to rapidly acquire significant features from pre-trained weights, necessitating only slight fine-tuning to achieve a satisfactory performance on the validation dataset.

The steep learning curve demonstrates the effectiveness of transfer learning, providing it particularly suitable for scenarios with limited training times. The ability to show good performance at an early stage shows transfer learning can be advantageous for iterative development, in which models may require frequent updates with new data.

The initial learning curve for models without pre-trained weights shows a flatter trajectory. The models show a long period before demonstrating improvements in MAE, with the overall progress showing more gradual changes. This indicates that models without pre-trained weights face greater challenges during the initial phases, as they must learn

all features from the scratch.

In scenarios where computational efficiency and time are essential, models without pre-trained weights show limitations. The slower learning curve indicates that without pre-trained weights requires more epochs to achieve comparable performance levels, which leads to increased costs with regard to both time and computational resources.

Impact of Dataset Amount on Convergence

Transfer learning demonstrates consistent convergence efficiency when applied to various dataset sizes. The dataset, regardless of whether it consists of 500 images or merely 100, shows significant initial convergence within the first few epochs, later moved into an early stabilization phase. This behavior demonstrates the efficiency of transfer learning in adapting diverse data sizes.

The robustness of transfer learning in terms of variations in dataset size make it the preferred approach for tasks that involve limited data. This holds importance in gaze estimation, as gathering extensive participant-specific data can bring challenges.

In contrast, models trained without pre-trained weights show a greater dependence on the size of the dataset. In scenarios where larger datasets, such as 500 images, are available, models without pre-trained weights can ultimately reach MAE scores that are on level with those obtained through transfer learning. However, when working with smaller datasets (e.g., 100-200 images), models trained from scratch tend to converge at a slower rate and demonstrate increased variability as shown in Figure 6.6. This is demonstrated by the frequent fluctuations in MAE during the later epochs.

The high sensitivity of models without pre-trained weights to the size of the dataset makes them less advantageous in scenarios where data availability is limited. This highlights the significance of transfer learning in practical applications, particularly in scenarios where datasets can be limited, noisy, or challenging to obtain.

6.4 Discussion

The following section outlines key findings and the significance of this work for gaze estimation for practical applications as well as the impact of machine learning techniques such as transfer learning.

6.4.1 Importance of Transfer Learning in Data-Limited Environments

This study highlights the significant impact of transfer learning in scenarios where data availability is limited. As seen in Section 6.3, the error for TL models is lower than NT

models in both Left Eye and Right Eye experiments. In many real-world scenarios, collecting large datasets for each individual proves to be impractical, particularly in domains such as medicine and industry, where the collection of extensive gaze data for personalized models can be both labor-intensive and complex in nature. Transfer learning can addresses this limitation by allowing models to maintain good performance levels, even when trained on smaller datasets. Moreover, transfer learning enables quick deployment by fine-tuning using pre-trained models with minimum data, therefore lowering costs and enabling fast adaption across users and domains.

6.4.2 Enhancing Gaze Estimation Efficiency with Transfer Learning

This study contributes to gaze estimation and machine learning by demonstrating the effectiveness of transfer learning in overcoming common challenges. While the training time for transfer learning and scratch models was almost identical (with transfer learning often taking slightly less time), transfer learning models ability to quickly adapt to different individuals helps address challenges such as overfitting, especially in scenarios with limited datasets, as shown in Section 6.3. Additionally, transfer learning models converge faster with lower error take fewer epochs to achieve results comparable to NT models, as seen in Section 6.3.5. This highlights the ability of pre-trained models to generalize efficiently across a variety of users, improving reliability and robustness, and offers a practical solutions for real-world scenarios where accuracy and adaptability are critical.

6.4.3 Significance and Broader Implications of the Study

This chapter makes valuable contributions that go beyond gaze estimation. It draws attention to the wider possibilities of transfer learning as an effective machine learning method. Transfer learning makes it possible for models to produce comparable results with fewer resources as opposed to just focusing on increasing accuracy. Because of their efficiency, machine learning models are more useful and approachable for broader application.

By lowering the requirement for large datasets, the study contributes to the increased accessibility of gaze estimation in a variety of domains, including assistive technology, gaming, education, and healthcare. This approach may encourage a wider use of eye-tracking devices in settings with limited resources.

6.5 Conclusion

This chapter illustrates the efficiency of transfer learning in gaze estimation systems, offering significant insights into its practical applications and advantages. The results show that transfer learning maintains competitive performance relative to models trained without pre-trained weights.

The analysis demonstrated benefits of transfer learning, including faster convergence and early stabilization of validation error. Fast adaptation to different individuals reduces the risks of overfitting and enhances the reliability of gaze estimation systems. The findings indicate that transfer learning is capable of effectively fine-tuning pre-trained models using limited data, thereby enabling quick deployment in a variety of applications.

It is essential to acknowledge certain limitations of this study. Although the gaze estimating models showed encouraging outcomes, their error rates remain inferior to those of leading commercial eye-tracking software available in the market. Our results highlight improvements in making gaze estimation more accessible but show that there is room for further refinement to achieve comparable accuracy of high-end systems.

This research shows the effectiveness of transfer learning in gaze estimation task, highlighting the advantages of using pre-trained models to improve performance when data availability is limited. The use of transfer learning in this context provides insights into improving model performance and resource efficiency in gaze estimation, which are applicable to similar computer vision tasks.

6.6 Contributions

After conducting all experiments, the third hypothesis was confirmed that models utilizing transfer learning, pre-trained on large datasets, and fine-tuned with low-quality webcam images from individual users, can outperform models trained from scratch. Our results demonstrated that transfer learning not only improves gaze estimation accuracy in data-limited settings, but also accelerates faster convergence and enhances training stability.

By leveraging pre-trained knowledge and adapting it to user-specific gaze data, we successfully reduced the amount of required training data, making the overall gaze estimation pipeline more efficient and practical. These advantages are especially relevant for real-world applications where collecting large personalized datasets is often not feasible.

In addition to validating our hypothesis, we performed a detailed analysis of the performance of the model across different input numbers of images. Furthermore, as part of this work, we introduced a new dataset specifically collected using low-resolution webcams, designed to support research on person-specific gaze estimation with and without transfer learning. This dataset, along with our results, offers useful insights for creating

eye-tracking technologies that are easy to use, affordable, and adaptable for many people.

Chapter 7

Conclusions and Future Aspects

This PhD thesis explored challenging yet important domain of gaze estimation using standard, low-cost hardware such as webcams. In this thesis, multiple approaches were developed and analyzed to address the limitations of gaze estimation with webcams using state-of-the-art deep learning techniques, especially the CNN network. The findings highlight the potential to democratize gaze tracking technology by using commonly available hardware and limited computational resources. This chapter summarizes our contributions and insights that we learned during designing and implementation of each contribution.

7.1 Summary

Chapter 1 introduced the concept of gaze estimation, emphasizing its importance in fields such as human-computer interaction (HCI), healthcare, gaming, and education. This chapter outlined the objectives and scope of the research, focusing on accessible and cost-effective solutions.

Chapter 2 provided the theoretical foundation, covering the anatomy of the human eye, artificial neural networks, convolutional neural networks, and transfer learning. This background established essential concepts necessary for understanding and implementing gaze estimation methods.

Chapter 3 reviewed advanced methodologies for gaze estimation, including model-based, feature-based, and appearance-based methods, highlighting their respective advantages and limitations. Advanced deep learning techniques such as single-stream CNNs, multi-stream CNNs, prior-based networks, temporal models, and transformer-based methods were also discussed. Additionally, this chapter examined publicly available gaze datasets and real-world applications of gaze estimation.

Chapter 4 presented a CNN-based approach for gaze estimation using unmodified webcams. The study treated gaze prediction as a classification task across 20 screen regions and showed that reasonable performance can be achieved with careful fine-tuning

of network parameters. The impact of image sharpening on model accuracy was also investigated, revealing a slight improvement in overall results.

Chapter 5 extended the focus to person-specific gaze estimation. The study explored whether building individualized models could achieve better performance compared to models trained on data from multiple users. User-specific models were developed using eye and face data and were systematically compared to multi-user models.

Chapter 6 investigated the applicability of transfer learning for gaze estimation in limited data scenarios using webcam-collected datasets. Models trained with and without pre-trained weights were compared across varying dataset sizes. The study demonstrated that pre-trained models performed better in low-data scenarios but also converged and stabilized more quickly. Additional experiments with very small and larger datasets reinforced these findings.

7.2 Contributions

This section details the original contributions made through the research work presented in this thesis. The contributions are organized according to the major objectives addressed throughout the study.

- **Development of a CNN-Based Model that Classifies a Person's Gaze into Screen Regions:** A convolutional neural network (CNN)-based gaze estimation model was developed capable of predicting a user's gaze region using images captured from an unmodified, low-cost webcam. Unlike many existing systems that require specialized and expensive hardware, this work demonstrated that reliable gaze prediction could be achieved affordably. The model framed gaze prediction as a classification task across 20 predefined regions on the screen, offering a practical balance between precision and usability for real-world applications.Furthermore, we investigated the role of image preprocessing, particularly the impact of image sharpening, on model performance. Experimental results indicated that applying sharpening filters enhanced feature clarity, leading to a modest but consistent improvement in gaze prediction accuracy. This finding highlights the importance of lightweight, computationally inexpensive preprocessing steps in enhancing performance for webcam-based gaze estimation systems.

  Based on the findings presented in Chapter 4, Hypothesis 1 (H1) is confirmed. The development and evaluation of a CNN-based gaze estimation model using low-quality images from an unmodified webcam demonstrate that it is feasible to classify a user's gaze into screen regions without the need for specialized hardware. The model's performance across 20 predefined regions supports the hypothesis that accurate gaze classification can be achieved using low quality images collected from

webcam.

- **Introduction and Evaluation of Person-Specific Gaze Estimation Models:** A person-specific gaze estimation model was developed and evaluated, and it was demonstrated that models trained on data from individual users—when combined with hyperparameter optimization—can outperform those trained on multi-user datasets. The potential benefits of personalization in gaze estimation were highlighted, especially in scenarios where highly accurate gaze prediction is required.

  After conducting all the experiments and findings presented in Chapter 5, Hypothesis 2 (H2) is confirmed. The development and evaluation of a person-specific gaze estimation model showed that models tailored for individual users, especially when carefully optimized hyperparameter, outperform generalized models trained on multi-user datasets. This confirms the hypothesis that personalization enhances gaze prediction accuracy, making it particularly valuable in applications where high precision is essential.

- **Comprehensive Study and Proposal for Scalable Gaze Estimation Solutions Using Transfer Learning:** A detailed analysis of the impact of transfer learning on gaze estimation was conducted when only a limited amount of training data was available. By comparing models trained from scratch to models initialized with pre-trained weights, across different dataset sizes (500, 400, 300, 200, and 100 images), we showed that transfer learning significantly boosts performance, enables faster convergence, and leads to more stable training outcomes. This finding is particularly valuable for scenarios where large labeled datasets are difficult or expensive to collect.Further experiments extended the analysis to extremely small datasets (50, 40, 30, 20, and 10 images) and larger datasets beyond 500 images. These experiments validated that pre-trained models consistently maintain performance advantages even in highly constrained data environments. This work underscores the robustness of transfer learning strategies for webcam-based gaze estimation tasks.

  Through the above developments, this research proposed scalable solutions to make gaze estimation technology more accessible to a wider audience. By eliminating the dependency on specialized hardware, leveraging lightweight preprocessing, incorporating personalization, and utilizing transfer learning, the methods developed in this thesis collectively pave the way for practical application of gaze-based systems.

  In addition to the main contributions, this thesis also proposes a practical workflow for building an person-specific gaze estimation model based on experimental findings. The recommended steps are as follows:

  Initially, a general gaze estimation model should be trained using eye images col-

lected from multiple users. Experiments demonstrated that eye images are more reliable than face images for training a general model, due to the challenges in generalizing face-based features across individuals.

Once the general model is prepared, for any new user, a small amount of personalized data should be collected through simple image capture sessions. This user-specific data is then used to fine-tune the pre-trained general model. The fine-tuning process is lightweight and efficient, making it possible even on devices with limited computational power.

If the gaze estimation error remains high after initial fine-tuning, the users need to gather more personalized samples or continue training for additional epochs to further improve accuracy. The experiments conducted in this thesis confirmed that following this workflow results in better-performing person-specific models with significantly less effort compared to training models from scratch.

The experimental findings presented in Chapter 6 confirmed Hypothesis 3 (H3). The conducted experiments demonstrated that transfer learning significantly enhances gaze estimation performance, especially when training data is limited. Models initialized with pre-trained weights performed better than those trained without pre-trained weights across varying dataset sizes—from as few as 10 samples to over 500—showing improved accuracy, faster convergence, and more stable training behavior. These findings validate the hypothesis that transfer learning is an effective strategy for improving model robustness and scalability in webcam-based gaze estimation, particularly in data-constrained scenarios.

7.3 Future Aspects

Webcam-based or standard camera-based gaze estimation presents important possibilities for improving accessible and economical solutions in multiple fields. Nonetheless, there are many possibilities for improvement and innovation that may accelerate the advancement of this technology.

Expanding datasets to include a broader range of environmental conditions, such as varying lighting, diverse backgrounds, and dynamic scenarios, will improve the robustness of gaze estimation models. Including participants from different demographic groups, such as varying age ranges, ethnicity, and those with visual aids like glasses or contact lenses, will help ensure inclusivity and accuracy across diverse populations. Collecting data from real-time interactive settings, including gaming, virtual reality simulations, and remote learning environments, will provide valuable insights into dynamic gaze behaviors.

In terms of model innovation, exploring hybrid architectures that combine CNN with transformer-based models can capture temporal dependencies in gaze patterns, making

them suitable for sequential gaze prediction tasks. Moreover, transfer learning, adopting self-supervised or semi-supervised learning approaches can reduce reliance on large annotated datasets, allowing models to adapt more efficiently to new applications.

Since standard webcams generally generate low image quality compared to specialized gaze tracking cameras, future studies should concentrate on enhancing gaze estimation accuracy using low-resolution data. Methods like super-resolution preprocessing or feature enhancement integrated into the model architecture may reduce the effects of image quality.

Creating efficient lightweight models for gaze estimation on resource-limited devices is crucial. Techniques for optimization, including quantization, pruning, and knowledge distillation, effectively minimize computational demands, allowing gaze estimation to be performed on standard laptops, smartphones, and tablets while maintaining accuracy. This improvement will contribute to energy efficiency, facilitating longer use in portable systems.

There exists significant potential for the development of domain-specific applications using gaze estimation technology. In healthcare, gaze models present opportunities for adaptation in clinical diagnostics, particularly in the identification of neurological disorders such as Parkinson's disease or autism, as well as in the monitoring of patient recovery throughout rehabilitation processes. In education sector, gaze tracking has the potential to evaluate student engagement within e-learning platforms and improve training simulations in essential domains like aviation and medicine. Additionally, in marketing and consumer behavior analysis, gaze tracking has the potential to achieve real-time insights into user attention, thereby revolutionizing product design and advertising methodologies.

Webcam-based gaze estimation can be easily included into AR/VR systems, allowing immersive experiences without requiring further hardware. Webcam-based gaze tracking in e-learning platforms can improve user engagement analytics. Furthermore, integrating gaze estimation with additional modalities such as facial expressions or speech recognition may produce highly interactive systems that intuitively respond to human intent.

With advancement of gaze estimation technology, it is essential that ethical considerations are prioritized to guarantee its responsible and fair implementation. Providing user privacy and secure data management is crucial for achieving widespread adoption. Creating strong on-device processing frameworks that remove the necessity for data transfer to external servers can greatly reduce privacy issues. Implementing robust data security measures, including encryption and anonymization, will improve the protection of user information. Minimizing biases in datasets and algorithms is essential for achieving fair results among various demographic groups, which encourages fairness and inclusivity. Furthermore, the development of transparent algorithms and the formulation of ethical guidelines for the implementation of gaze estimation technologies will enhance trust and accountability, allowing their responsible application across diverse field.

Extensive research on gaze behaviors in various contexts and tasks can enhance the development of more customized and effective models. Through the integration of user-specific data over time, webcam-based systems may improve performance to align with individual behaviors.

By exploring these future directions, researchers can expand upon contribution of this thesis to develop scalable, accessible, and ethically responsible gaze estimation solutions, thereby transforming human-computer interaction across various sectors.

Bibliography

[1] Tobii AB. *Tobii Eye-Trackers*. Accessed: 2024-06-24. URL: https://www.tobii.com/.

[2] Richard V Abadi, David Carden and John Simpson. 'Listening for eye movements'. In: *Ophthalmic and Physiological Optics* 1.1 (1981), pp. 19–27.

[3] Ahmed A Abdelrahman, Thorsten Hempel, Aly Khalifa and Ayoub Al-Hamadi. 'L2CS-Net: Fine-Grained Gaze Estimation in Unconstrained Environments'. In: *arXiv preprint arXiv:2203.03339* (2022).

[4] AF Agarap. 'Deep learning using rectified linear units (relu)'. In: *arXiv preprint arXiv:1803.08375* (2018).

[5] Javier San Agustin and Martin Tall. *Gaze Tracking Library*. http://sourceforge.net/projects/gazetrackinglib/. 2017.

[6] Abid Ali and Yong-Guk Kim. 'Deep fusion for 3D gaze estimation from natural face images using multi-stream CNNs'. In: *IEEE Access* 8 (2020), pp. 69212–69221.

[7] Fares Alnajar, Theo Gevers, Roberto Valenti and Sennay Ghebreab. 'Calibration-free Gaze Estimation using Human Gaze Patterns'. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. IEEE. 2013, pp. 137–144.

[8] Mohd Faizan Ansari, Pawel Kasprowski and Marcin Obetkal. 'Gaze Tracking Using an Unmodified Web Camera and Convolutional Neural Network'. In: *Applied Sciences* 11.19 (2021), p. 9068.

[9] Mohd Faizan Ansari, Pawel Kasprowski and Peter Peer. 'Person-specific gaze estimation from low-quality webcam images'. In: *Sensors* 23.8 (2023), p. 4138.

[10] João Antunes and Pedro Santana. 'A study on the use of eye tracking to adapt gameplay and procedural content generation in first-person shooter games'. In: *Multimodal Technologies and Interaction* 2.2 (2018), p. 23.

[11] Automate. *Eye-Tracking Technology: The Future of Human-Computer Interaction*. Accessed: 2024-06-24. 2019. URL: https://www.automate.org/vision/blogs/eye-tracking-technology-the-future-of-human-computer-interaction.

[12] Ronald T Azuma. 'A survey of augmented reality'. In: *Presence: teleoperators & virtual environments* 6.4 (1997), pp. 355–385.

[13] David Bäck. *Neural network gaze tracking using web camera*. 2006.

[14] Haldun Balim, Seonwook Park, Xi Wang, Xucong Zhang and Otmar Hilliges. 'Efe: End-to-end frame-to-gaze estimation'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 2688–2697.

[15] Tadas Baltrusaitis, Amir Zadeh, Yao Chong Lim and Louis-Philippe Morency. 'Openface 2.0: Facial behavior analysis toolkit'. In: *2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018)*. IEEE. 2018, pp. 59–66.

[16] Shumeet Baluja and Dean Pomerleau. 'Non-intrusive gaze tracking using artificial neural networks'. In: *Advances in Neural Information Processing Systems* 6 (1993).

[17] Sylwester Białowąs and Adrianna Szyszka. 'Eye-tracking in marketing research'. In: *Managing Economic Innovations–Methods and Instruments* 1.69 (2019), pp. 91–104.

[18] Maria Borgestig, Jan Sandqvist, Richard Parsons, Torbjörn Falkmer and Helena Hemmingsson. 'Eye gaze performance for children with severe physical impairments using gaze-based assistive technology—A longitudinal study'. In: *Assistive technology* 28.2 (2016), pp. 93–102.

[19] Virginio Cantoni, Mirto Musci, Nahumi Nugrahaningsih and Marco Porta. 'Gaze-based biometrics: An introduction to forensic applications'. In: *Pattern Recognition Letters* 113 (2018), pp. 54–57.

[20] Andy Catruna, Adrian Cosma and Emilian Radoi. 'Crossgaze: A strong method for 3d gaze estimation in the wild'. In: *2024 IEEE 18th International Conference on Automatic Face and Gesture Recognition (FG)*. IEEE. 2024, pp. 1–5.

[21] Juan J Cerrolaza, Arantxa Villanueva and Rafael Cabeza. 'Taxonomic study of polynomial regressions applied to the calibration of video-oculographic systems'. In: *Proceedings of the 2008 symposium on Eye tracking research & applications*. 2008, pp. 259–266.

[22] Aayush K. Chaudhary, Rakshit Kothari, Manoj Acharya, Shusil Dangi, Nitinraj Nair, Reynold Bailey, Christopher Kanan, Gabriel Diaz and Jeff B. Pelz. 'RIT-Net: Real-Time Semantic Segmentation of the Eye for Gaze Tracking'. In: *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. IEEE. 2019, pp. 3698–3702.

[23] Jixu Chen and Qiang Ji. '3D gaze estimation with a single camera without IR illumination'. In: *2008 19th international conference on pattern recognition*. IEEE. 2008, pp. 1–4.

[24] Zhaokang Chen and Bertram E Shi. 'Appearance-based gaze estimation using dilated-convolutions'. In: *Asian Conference on Computer Vision*. Springer. 2018, pp. 309–324.

[25] Zhaokang Chen and Bertram E Shi. 'Using variable dwell time to accelerate gaze-based web browsing with two-step selection'. In: *International Journal of Human–Computer Interaction* 35.3 (2019), pp. 240–255.

[26] Yihua Cheng and Feng Lu. 'Gaze Estimation Using Transformer'. In: *arXiv preprint arXiv:2105.14424* (2021).

[27] Yihua Cheng, Haofei Wang, Yiwei Bao and Feng Lu. 'Appearance-based gaze estimation with deep learning: A review and benchmark'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2024).

[28] Yiu-ming Cheung and Qinmu Peng. 'Eye gaze tracking with a web camera in a desktop environment'. In: *IEEE Transactions on Human-Machine Systems* 45.4 (2015), pp. 419–430.

[29] Byeongkeun Choi, Juho Ahn, Jaesik Parl and In So Kweon. 'Appearance-based Gaze Estimation Using Kinect'. In: *Proceedings of the IEEE Conference on Ubiquitous Robots and Ambient Intelligence*. 2013, pp. 245–250.

[30] Fulvio Corno, Laura Farinetti and Isabella Signorile. 'A cost-effective solution for eye-gaze assistive technology'. In: *Proceedings. IEEE International Conference on Multimedia and Expo*. Vol. 2. IEEE. 2002, pp. 433–436.

[31] Tom N Cornsweet and Hewitt D Crane. 'Accurate two-dimensional eye tracker using first and fourth Purkinje images'. In: *JOSA* 63.8 (1973), pp. 921–928.

[32] Kevin Cortacero, Tobias Fischer and Yiannis Demiris. 'RT-BENE: A dataset and baselines for real-time blink estimation in natural environments'. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*. 2019, pp. 0–0.

[33] Navneet Dalal and Bill Triggs. 'Histograms of oriented gradients for human detection'. In: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*. Vol. 1. Ieee. 2005, pp. 886–893.

[34] Roberto Daza, Aythami Morales, Julian Fierrez and Ruben Tolosana. 'mEBAL: A multimodal database for eye blink detection and attention level estimation'. In: *Companion publication of the 2020 international conference on Multimodal interaction*. 2020, pp. 32–36.

[35] Mehmet Donmez. 'The Use of Eye-tracking Technology in Education'. In: (2023).

[36] Neeru Dubey, Shreya Ghosh and Abhinav Dhall. 'Unsupervised learning of eye gaze representation from the web'. In: *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2019, pp. 1–7.

[37] Andrew T Duchowski and Andrew T Duchowski. *Eye tracking methodology: Theory and practice*. Springer, 2017.

[38] Maria K Eckstein, Belén Guerra-Carrillo, Alison T Miller Singley and Silvia A Bunge. 'Beyond eye gaze: What else can eyetracking reveal about cognition and cognitive development?' In: *Developmental cognitive neuroscience* 25 (2017), pp. 69–91.

[39] Kara J Emery, Marina Zannoli, James Warren, Lei Xiao and Sachin S Talathi. 'OpenNEEDS: A dataset of gaze, head, hand, and scene signals during exploration in open-ended VR environments'. In: *ACM Symposium on Eye Tracking Research and Applications*. 2021, pp. 1–7.

[40] Lifeng Fan, Wenguan Wang, Siyuan Huang, Xinyu Tang and Song-Chun Zhu. 'Understanding human gaze communication by spatio-temporal graph reasoning'. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 5724–5733.

[41] Onur Ferhat. *A gaze estimation method and system for natural light cameras*. 2017.

[42] Onur Ferhat, Arcadi Llanza and Fernando Vilariño. 'A feature-based gaze estimation algorithm for natural light scenarios'. In: *Pattern Recognition and Image Analysis: 7th Iberian Conference, IbPRIA 2015, Santiago de Compostela, Spain, June 17-19, 2015, Proceedings 7*. Springer. 2015, pp. 569–576.

[43] Tobias Fischer, Hyung Jin Chang and Yiannis Demiris. 'RT-GENE: Real-Time Eye Gaze Estimation in Natural Environments'. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 339–357.

[44] Tobias Fischer, Hyung Jin Chang and Yiannis Demiris. 'Rt-gene: Real-time eye gaze estimation in natural environments'. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 334–352.

[45] Andrew W Fitzgibbon, Maurizio Pilu and Robert B Fisher. 'Direct least squares fitting of ellipses'. In: *Proceedings of 13th international conference on pattern recognition*. Vol. 1. IEEE. 1996, pp. 253–257.

[46] Takashi Fukuda, Kosuke Morimoto and Hayato Yamana. 'Model-based eye-tracking method for low-resolution eye-images'. In: *2nd Workshop on Eye Gaze in Intelligent Human Machine Interaction*. Vol. 7. 2011, pp. 10–19.

[47] Antonio Funes Mora and Jean-Marc Odobez. 'Geometric Generative Gaze Estimation (G3E) for Remote RGB-D Cameras'. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 2441–2448.

[48] Kenneth Alberto Funes Mora, Florent Monay and Jean-Marc Odobez. 'Eyediap: A database for the development and evaluation of gaze estimation algorithms from rgb and rgb-d cameras'. In: *Proceedings of the symposium on eye tracking research and applications*. 2014, pp. 255–258.

[49] Gregory Funke, Eric Greenlee, Martha Carter, Allen Dukes, Rebecca Brown and Lauren Menke. 'Which eye tracker is right for your research? performance evaluation of several cost variant eye trackers'. In: *Proceedings of the Human Factors and Ergonomics Society annual meeting*. Vol. 60. 1. SAGE Publications Sage CA: Los Angeles, CA. 2016, pp. 1240–1244.

[50] Tianmin Gao, Daniel Harari, Joshua Tenenbaum and Shimon Ullman. *When Computer Vision Gazes at Cognition*. Tech. rep. Cambridge, MA, USA: Center for Brains, Minds and Machines, 2014.

[51] Stephan J Garbin, Yiru Shen, Immo Schuetz, Robert Cavin, Gregory Hughes and Sachin S Talathi. 'Openeds: Open eye dataset'. In: *arXiv preprint arXiv:1905.03702* (2019).

[52] GazeHawk. *GazeHawk - Webcam Eye Tracking - Ad Testing and Optimization*. https://gazehawk.com/. Accessed: Mar. 26, 2024.

[53] Shreya Ghosh. 'Automatic Eye Gaze Estimation with Limited Supervision'. Thesis. Monash University, 2022. DOI: 10.26180/21302853.v1. URL: https://doi.org/10.26180/21302853.v1.

[54] Shreya Ghosh. 'Automatic Eye Gaze Estimation with Limited Supervision'. PhD thesis. Monash University, 2022.

[55] Xavier Glorot and Yoshua Bengio. 'Understanding the difficulty of training deep feedforward neural networks'. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2010, pp. 249–256.

[56] Alessandro Grillini, Daniel Ombelet, Rijul S Soans and Frans W Cornelissen. '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*. 2018, pp. 1–5.

[57] Dan Witzner Hansen, John Paulin Hansen, Mads Nielsen, Anders Sewerin Johansen and Mikkel B Stegmann. 'Eye typing using Markov and active appearance models'. In: *Sixth IEEE Workshop on Applications of Computer Vision, 2002.(WACV 2002). Proceedings*. IEEE. 2002, pp. 132–136.

[58] Dan Witzner 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* 32.3 (2009), pp. 478–500.

[59] Dan Witzner Hansen, Mads Nielsen, John Paulin Hansen, Anders Sewerin Johansen and Mikkel B Stegmann. 'Tracking Eyes Using Shape and Appearance.' In: *MVA*. 2002, pp. 201–204.

[60] Dan Witzner Hansen and Arthur EC Pece. 'Eye tracking in the wild'. In: *Computer Vision and Image Understanding* 98.1 (2005), pp. 155–181.

[61] Kaiming He, Georgia Gkioxari, Piotr Dollár and Ross Girshick. 'Mask r-cnn'. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2961–2969.

[62] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. 'Deep Residual Learning for Image Recognition'. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[63] Peter Hevesi, Jamie A Ward, Orkham Amiraslanov, Gerald Pirkl and Paul Lukowicz. 'Wearable eye tracking for multisensor physical activity recognition'. In: *International Journal On Advances in Life Sciences* 10.12 (2018), pp. 103–116.

[64] Sabrina Hoppe, Tobias Loetscher, Stephanie A Morey and Andreas Bulling. 'Eye movements during everyday behavior predict personality traits'. In: *Frontiers in human neuroscience* 12 (2018), p. 105.

[65] Guilei Hu, Yang Xiao, Zhiguo Cao, Lubin Meng, Zhiwen Fang, Joey Tianyi Zhou and Junsong Yuan. 'Towards real-time eyeblink detection in the wild: Dataset, theory and practices'. In: *IEEE Transactions on Information Forensics and Security* 15 (2019), pp. 2194–2208.

[66] Chien-Ming Huang and Bilge Mutlu. 'Anticipatory robot control for efficient human-robot collaboration'. In: *2016 11th ACM/IEEE international conference on human-robot interaction (HRI)*. IEEE. 2016, pp. 83–90.

[67] Gao Huang, Zhuang Liu, Laurens van der Maaten and Kilian Q. Weinberger. 'Densely Connected Convolutional Networks'. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[68] Michael Xuelin Huang, Jiajia Li, Grace Ngai and Hong Va Leong. 'Screenglint: Practical, in-situ gaze estimation on smartphones'. In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 2017, pp. 2546–2557.

[69] Qiong Huang, Ashok Veeraraghavan and Ashutosh Sabharwal. 'Tabletgaze: dataset and analysis for unconstrained appearance-based gaze estimation in mobile tablets'. In: *Machine Vision and Applications* 28 (2017), pp. 445–461.

[70] Shang-Che Huang, Yi-Leh Wu, Wei-Chih Hung and Cheng-Yuan Tang. 'Point-of-regard measurement via iris contour with one eye from single image'. In: *2010 IEEE International Symposium on Multimedia*. IEEE. 2010, pp. 336–341.

[71] iMotions. *iMotions Eye Tracking Glasses Module*. Accessed: 2024-12-08. n.d. URL: https://imotions.com/products/imotions-lab/modules/eye-tracking-glasses/.

[72] Ibrahim Furkan Ince and Jin Woo Kim. 'A 2D eye gaze estimation system with low-resolution webcam images'. In: *EURASIP Journal on Advances in Signal Processing* 2011 (2011), pp. 1–11.

[73] Takahiro Ishikawa. 'Passive driver gaze tracking with active appearance models'. In: (2004).

[74] Robert JK Jacob and Keith S Karn. 'Eye tracking in human-computer interaction and usability research: Ready to deliver the promises'. In: *The mind's eye*. Elsevier, 2003, pp. 573–605.

[75] Anil K Jain, Ruud Bolle and Sharath Pankanti. *Biometrics: personal identification in networked society*. Vol. 479. Springer Science & Business Media, 2006.

[76] Emile Javal. 'Essai sur la physiologie de la lecture'. In: *Annales d'Ocilistique* 80 (1878), pp. 97–117.

[77] L. A. Jeni and J. F. Cohn. 'Person-independent 3D Gaze Estimation using Face Frontalization'. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2016, pp. 1620–1628.

[78] Antony William Joseph and Ramaswamy Murugesh. 'Potential eye tracking metrics and indicators to measure cognitive load in human-computer interaction research'. In: *J. Sci. Res* 64.1 (2020), pp. 168–175.

[79] Tilke Judd, Krista Ehinger, Frédo Durand and Antonio Torralba. 'Learning to predict where humans look'. In: *2009 IEEE 12th international conference on computer vision*. IEEE. 2009, pp. 2106–2113.

[80] Petr Kellnhofer, Adria Recasens, Simon Stent, Wojciech Matusik and Antonio Torralba. 'Gaze360: Physically Unconstrained Gaze Estimation in the Wild'. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2019.

[81] Eun Yi Kim, Sin Kuk Kang, Keechul Jung and Hang Joon Kim. 'Eye mouse: mouse implementation using eye tracking'. In: *2005 Digest of Technical Papers. International Conference on Consumer Electronics, 2005. ICCE*. IEEE. 2005, pp. 207–208.

[82] Joohwan Kim, Michael Stengel, Alexander Majercik, Shalini De Mello, David Dunn, Samuli Laine, Morgan McGuire and David Luebke. 'Nvgaze: An anatomically-informed dataset for low-latency, near-eye gaze estimation'. In: *Proceedings of the 2019 CHI conference on human factors in computing systems*. 2019, pp. 1–12.

[83] Diederik P Kingma and Jimmy Ba. 'Adam: A method for stochastic optimization'. In: *arXiv preprint arXiv:1412.6980* (2014).

[84] Rakshit Kothari, Zhizhuo Yang, Christopher Kanan, Reynold Bailey, Jeff B Pelz and Gabriel J Diaz. 'Gaze-in-wild: A dataset for studying eye and head coordination in everyday activities'. In: *Scientific reports* 10.1 (2020), p. 2539.

[85] Kyle Krafka, Aditya Khosla, Petr Kellnhofer, Harini Kannan, Suchendra Bhandarkar, Wojciech Matusik and Antonio Torralba. 'Eye tracking for everyone'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2176–2184.

[86] Alex Krizhevsky, Ilya Sutskever and Geoffrey E Hinton. 'Imagenet classification with deep convolutional neural networks'. In: *Advances in neural information processing systems* 25 (2012), pp. 1097–1105.

[87] Chih-Chuan Lai, Yu-Ting Chen, Kuan-Wen Chen, Shen-Chi Chen, Sheng-Wen Shih and Yi-Ping Hung. 'Appearance-Based Gaze Tracking with Free Head Movement'. In: *Proceedings of the International Conference on Pattern Recognition (ICPR)*. Vol. 1. 2014, pp. 1869–1873.

[88] Joseph Lemley, Anuradha Kar, Alexandru Drimbarean and Peter Corcoran. 'Convolutional neural network implementation for eye-gaze estimation on low-quality consumer imaging systems'. In: *IEEE Transactions on Consumer Electronics* 65.2 (2019), pp. 179–187.

[89] Xiaozhou Li, Zheying Zhang and Kostas Stefanidis. 'A data-driven approach for video game playability analysis based on players' reviews'. In: *Information* 12.3 (2021), p. 129.

[90] Ke Liang, Youssef Chahir, Michele Molina, Charles Tijus and François Jouen. 'Appearance-based gaze tracking with spectral clustering and semi-supervised gaussian process regression'. In: *Proceedings of the 2013 Conference on Eye Tracking South Africa*. 2013, pp. 17–23.

[91] Gang Liu, Yu Yu, Kenneth Alberto Funes Mora and Jean-Marc Odobez. 'A differential approach for gaze estimation'. In: *IEEE transactions on pattern analysis and machine intelligence* (2019).

[92] Feng Lu, Takahiro Okabe, Yusuke Sugano and Yoichi Sato. 'Learning Gaze Biases with Head Motion for Head Pose-Free Gaze Estimation'. In: *Image and Vision Computing* 32.3 (2014), pp. 169–179.

[93] Feng Lu, Yusuke Sugano, Takahiro Okabe and Yoichi Sato. 'Adaptive linear regression for appearance-based gaze estimation'. In: *IEEE transactions on pattern analysis and machine intelligence* 36.10 (2014), pp. 2033–2046.

[94] Feng Lu, Yusuke Sugano, Takahiro Okabe and Yoichi Sato. 'Inferring Human Gaze from Appearance via Adaptive Linear Regression'. In: *Proceedings of the International Conference on Computer Vision (ICCV)*. Ed. by Dimitris N. Metaxas, Long Quan, Alberto Sanfeliu and Luc J. Van Gool. IEEE, 2011, pp. 153–160.

[95] Yi Lu, Yusuke Sugano, Takahiro Okabe and Yoichi Sato. 'Head Pose-free Appearance-based Gaze Sensing via Eye Image Synthesis'. In: *Proceedings of the IEEE International Conference on Pattern Recognition*. 2012, pp. 424–427. DOI: 10.1109/ICPR.2012.674.

[96] Robert Gabriel Lupu, Radu Gabriel Bozomitu, Alexandru Păsărică and Cristian Rotariu. 'Eye tracking user interface for Internet access used in assistive technology'. In: *2017 E-Health and Bioengineering Conference (EHB)*. IEEE. 2017, pp. 659–662.

[97] Päivi Majaranta and Andreas Bulling. 'Eye tracking and eye-based human–computer interaction'. In: *Advances in physiological computing*. Springer, 2014, pp. 39–65.

[98] Manuel J Marin-Jimenez, Vicky Kalogeiton, Pablo Medina-Suarez and Andrew Zisserman. 'Laeo-net: revisiting people looking at each other in videos'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3477–3485.

[99] Francis Martinez, Andrea Carbone and Edwige Pissaloux. 'Gaze estimation using local features and non-linear regression'. In: *2012 19th IEEE International Conference on Image Processing*. IEEE. 2012, pp. 1961–1964.

[100] Maria Laura Mele and Stefano Federici. 'Gaze and eye-tracking solutions for psychological research'. In: *Cognitive processing* 13 (2012), pp. 261–265.

[101] Raphael Menges, Chandan Kumar, Daniel Müller and Korok Sengupta. 'Gazetheweb: A gaze-controlled web browser'. In: *Proceedings of the 14th International Web for All Conference*. 2017, pp. 1–2.

[102] John Merchant, Richard Morrissette and James L Porterfield. 'Remote measurement of eye direction allowing subject motion over one cubic foot of space'. In: *IEEE transactions on biomedical engineering* 4 (1974), pp. 309–317.

[103] Melissa A Miller and Mark T Fillmore. 'Persistence of attentional bias toward alcohol-related stimuli in intoxicated social drinkers'. In: *Drug and Alcohol Dependence* 117.2-3 (2011), pp. 184–189.

[104] Mohammad Reza Mohammadi and Abolghasem Raie. 'Robust pose-invariant eye gaze estimation using geometrical features of iris and pupil images'. In: *20th Iranian Conference on Electrical Engineering (ICEE2012)*. IEEE. 2012, pp. 593–598.

[105] Susan M Munn and Jeff B Pelz. '3D point-of-regard, position and head orientation from a portable monocular video-based eye tracker'. In: *Proceedings of the 2008 symposium on Eye tracking research & applications*. 2008, pp. 181–188.

[106] Ba Linh Nguyen. 'Eye Gaze Tracking'. In: *Proceedings of the IEEE-RIVF International Conference on Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF)*. IEEE. 2009, pp. 1–4.

[107] Ba Linh Nguyen, Youssef Chahir, Michèle Molina, Charles Tijus and François Jouen. 'Eye Gaze Tracking with Free Head Movements Using a Single Camera'. In: *Proceedings of the ACM Symposium on Information and Communication Technology (SoICT)*. Ed. by Nguyen Trong Giang, Nguyen Thuc Hai and Huynh Quyet Thang. Vol. 449. ACM International Conference Proceeding Series. ACM. 2010, pp. 108–113.

[108] PhiBang Nguyen, Julien Fleureau, Christel Chamaret and Philippe Guillotel. 'Calibration-free gaze tracking using particle filter'. In: *2013 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE. 2013, pp. 1–6.

[109] Timo Ojala, Matti Pietikäinen and David Harwood. 'A comparative study of texture measures with classification based on featured distributions'. In: *Pattern recognition* 29.1 (1996), pp. 51–59.

[110] Tom O'Malley, Elie Bursztein, James Long, François Chollet, Haifeng Jin, Luca Invernizzi et al. *KerasTuner*. https://github.com/keras-team/keras-tuner. 2019.

[111] Yasuhiro Ono, Takahiro Okabe and Yoichi Sato. 'Gaze estimation from low resolution images'. In: *Lecture Notes in Computer Science* 4319 (2006), pp. 178–188.

[112] Benjamin I Outram, Yun Suen Pai, Tanner Person, Kouta Minamizawa and Kai Kunze. 'Anyorbit: Orbital navigation in virtual environments with eye-tracking'. In: *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications*. 2018, pp. 1–5.

[113] Cristina Palmero, Javier Selva, Mohammad Ali Bagheri and Sergio Escalera. 'Recurrent cnn for 3d gaze estimation using appearance and shape cues'. In: *arXiv preprint arXiv:1805.03064* (2018).

[114] Cristina Palmero, Abhishek Sharma, Karsten Behrendt, Kapil Krishnakumar, Oleg V Komogortsev and Sachin S Talathi. 'Openeds2020: Open eyes dataset'. In: *arXiv preprint arXiv:2005.03876* (2020).

[115] Seonwook Park, Emre Aksan, Xucong Zhang and Otmar Hilliges. 'Towards End-to-End Video-based Eye-Tracking'. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2020.

[116] Seonwook Park, Emre Aksan, Xucong Zhang and Otmar Hilliges. 'Towards end-to-end video-based eye-tracking'. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XII 16*. Springer. 2020, pp. 747–763.

[117] Seonwook Park, Shalini De Mello, Pavlo Molchanov, Umar Iqbal, Otmar Hilliges and Jan Kautz. 'Few-Shot Adaptive Gaze Estimation'. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2019, pp. 9368–9377.

[118] Seonwook Park, Shalini De Mello, Pavlo Molchanov, Umar Iqbal, Otmar Hilliges and Jan Kautz. 'Few-shot adaptive gaze estimation'. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 9368–9377.

[119] Seonwook Park, Adrian Spurr and Otmar Hilliges. 'Deep pictorial gaze estimation'. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 721–738.

[120] Stefan Pastel, Josua Marlok, Nicole Bandow and Kerstin Witte. 'Application of eye-tracking systems integrated into immersive virtual reality and possible transfer to the sports sector-A systematic review'. In: *Multimedia Tools and Applications* 82.3 (2023), pp. 4181–4208.

[121] Anjul Patney, Joohwan Kim, Marco Salvi, Anton Kaplanyan, Chris Wyman, Nir Benty, Aaron Lefohn and David Luebke. 'Perceptually-based foveated virtual reality'. In: *ACM SIGGRAPH 2016 emerging technologies*. 2016, pp. 1–2.

[122] Anjul Patney, Marco Salvi, Joohwan Kim, Anton Kaplanyan, Chris Wyman, Nir Benty, David Luebke and Aaron Lefohn. 'Towards foveated rendering for gaze-tracked virtual reality'. In: *ACM Transactions on Graphics (TOG)* 35.6 (2016), pp. 1–12.

[123] Jimin Pi and Bertram E Shi. 'Probabilistic adjustment of dwell time for eye typing'. In: *2017 10th International Conference on Human System Interactions (HSI)*. IEEE. 2017, pp. 251–257.

[124] Francesco Ragusa, Antonino Furnari, Sebastiano Battiato, Giovanni Signorello and Giovanni Maria Farinella. 'EGO-CH: Dataset and fundamental tasks for visitors behavioral understanding using egocentric vision'. In: *Pattern Recognition Letters* 131 (2020), pp. 150–157.

[125] Shaoqing Ren, Kaiming He, Ross Girshick and Jian Sun. 'Faster r-cnn: Towards real-time object detection with region proposal networks'. In: *Advances in neural information processing systems* 28 (2015).

[126] SR Research. *EyeLink Eye Trackers*. Accessed: 2024-06-24. URL: https://www.sr-research.com/.

[127] Derek R Rutter and Kevin Durkin. 'Turn-taking in mother–infant interaction: An examination of vocalizations and gaze.' In: *Developmental psychology* 23.1 (1987), p. 54.

[128] SensoMotoric Instruments GmbH. *Eye Tracking Solutions by SMI*. http://www.smivision.com/. 2017.

[129] Karen Simonyan and Andrew Zisserman. 'Very Deep Convolutional Networks for Large-Scale Image Recognition'. In: *arXiv preprint arXiv:1409.1556* (2014).

[130] Evangelos Skodras, Vasileios G Kanas and Nikolaos Fakotakis. 'On visual gaze tracking based on a single low cost camera'. In: *Signal Processing: Image Communication* 36 (2015), pp. 29–42.

[131] Brian A Smith, Qi Yin, Steven K Feiner and Shree K Nayar. 'Gaze locking: passive eye contact detection for human-object interaction'. In: *Proceedings of the 26th annual ACM symposium on User interface software and technology*. 2013, pp. 271–280.

[132] Lisa Spitzer and Stefanie Mueller. 'Using a test battery to compare three remote, video-based eye-trackers'. In: *2022 Symposium on Eye Tracking Research and Applications*. 2022, pp. 1–7.

[133] Yusuke Sugano, Yasuyuki Matsushita and Yoichi Sato. 'Appearance-based Gaze Estimation using Visual Saliency'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.2 (2013), pp. 329–341.

[134] Yusuke Sugano, Yasuyuki Matsushita and Yoichi Sato. 'Learning-by-synthesis for appearance-based 3d gaze estimation'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 1821–1828.

[135] Yusuke Sugano, Yasuyuki Matsushita, Yoichi Sato and Hideki Koike. 'An Incremental Learning Method for Unconstrained Gaze Estimation'. In: *Proceedings of the European Conference on Computer Vision (ECCV), Part III*. 2008, pp. 656–667.

[136] Kar-Han Tan, David J Kriegman and Narendra Ahuja. 'Appearance-based eye gaze estimation'. In: *Sixth IEEE Workshop on Applications of Computer Vision, 2002.(WACV 2002). Proceedings.* IEEE. 2002, pp. 191–195.

[137] The Eye Tribe. *The Eye Tribe*. http://theeyetribe.com/. 2017.

[138] Tobii. *Tobii Pro Glasses 3*. Accessed: 2024-12-08. n.d. URL: https://www.tobii.com/products/eye-trackers/wearables/tobii-pro-glasses-3.

[139] Tobii AB. *Tobii Eye-Trackers*. http://www.tobii.com/. 2017.

[140] Henri Tomas, Marcus Reyes, Raimarc Dionido, Mark Ty, Jonric Mirando, Joel Casimiro, Rowel Atienza and Richard Guinto. 'Goo: A dataset for gaze object prediction in retail environments'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 3125–3133.

[141] Marc Tonsen, Julian Steil, Yusuke Sugano and Andreas Bulling. 'Invisibleeye: Mobile eye tracking using multiple low-resolution cameras and learning-based gaze estimation'. In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1.3 (2017), pp. 1–21.

[142] Kentaro Toyama. 'Look, ma-no hands! hands-free cursor control with real-time 3d face tracking'. In: *PUI98* 24 (1998).

[143] Roberto Valenti, Nicu Sebe and Theo Gevers. 'Combining head pose and eye location information for gaze estimation'. In: *IEEE Transactions on Image Processing* 21.2 (2011), pp. 802–815.

[144] Roberto Valenti, Nicu Sebe, Theo Gevers et al. 'Simple and efficient visual gaze estimation'. In: *Workshop on Multimodal Interactions Analysis of Users in a Controlled Environment*. 2008.

[145] Roberto Valenti, Jacopo Staiano, Nicu Sebe and Theo Gevers. 'Webcam-based visual gaze estimation'. In: *Image Analysis and Processing–ICIAP 2009: 15th International Conference Vietri sul Mare, Italy, September 8-11, 2009 Proceedings 15*. Springer. 2009, pp. 662–671.

[146] Niilo V Valtakari, Roy S Hessels, Diederick C Niehorster, Charlotte Viktorsson, Pär Nyström, Terje Falck-Ytter, Chantal Kemner and Ignace TC Hooge. 'A field test of computer-vision-based gaze estimation in psychology'. In: *Behavior Research Methods* 56.3 (2024), pp. 1900–1915.

[147] Ronda Venkateswarlu et al. 'Eye gaze estimation from a single image of one eye'. In: *Proceedings Ninth IEEE International Conference on Computer Vision*. IEEE. 2003, pp. 136–143.

[148] Paul Viola and Michael Jones. 'Rapid object detection using a boosted cascade of simple features'. In: *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*. Vol. 1. Ieee. 2001, pp. I–I.

[149] Jian-Gang Wang, Eric Sung and Ronda Venkateswarlu. 'Estimating the eye gaze from one eye'. In: *Computer Vision and Image Understanding* 98.1 (2005), pp. 83–103.

[150] Kang Wang and Qiang Ji. 'Real time eye gaze tracking with 3d deformable eye-face model'. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 1003–1011.

[151] Kang Wang, Hui Su and Qiang Ji. 'Neuro-inspired Eye Tracking with Eye Movement Dynamics'. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 9831–9840.

[152] Michel Wedel and Rik Pieters. 'A review of eye-tracking research in marketing'. In: *Review of Marketing Research* (2017).

[153] Alex Wiebe, Anette Kersting and Thomas Suslow. 'Deployment of attention to emotional pictures varies as a function of externally-oriented thinking: An eye tracking investigation'. In: *Journal of behavior therapy and experimental psychiatry* 55 (2017), pp. 1–5.

[154] Stephanie Wilkerson. 'Application of the Paired t-test'. In: *XULAneXUS* 5.1 (2008), p. 7.

[155] Cort J Willmott and Kenji Matsuura. 'Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance'. In: *Climate research* 30.1 (2005), pp. 79–82.

[156] Nicolai Wojke, Alex Bewley and Dietrich Paulus. 'Simple online and realtime tracking with a deep association metric'. In: *2017 IEEE international conference on image processing (ICIP)*. IEEE. 2017, pp. 3645–3649.

[157] Erroll Wood, Tadas Baltrušaitis, Louis-Philippe Morency, Peter Robinson and Andreas Bulling. 'A 3d morphable eye region model for gaze estimation'. In: *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer. 2016, pp. 297–313.

[158] Erroll Wood, Tadas Baltrušaitis, Louis-Philippe Morency, Peter Robinson and Andreas Bulling. 'Gazedirector: Fully articulated eye gaze redirection in video'. In: *Computer Graphics Forum*. Vol. 37. 2. Wiley Online Library. 2018, pp. 217–225.

[159] Erroll Wood, Tadas Baltrušaitis, Louis-Philippe Morency, Peter Robinson and Andreas Bulling. 'Learning an appearance-based gaze estimator from one million synthesised images'. In: *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*. 2016, pp. 131–138.

[160] Erroll Wood, Tadas Baltrusaitis, Xucong Zhang, Yusuke Sugano, Peter Robinson and Andreas Bulling. 'Rendering of eyes for eye-shape registration and gaze estimation'. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 3756–3764.

[161] Haiyuan Wu, Qian Chen and Toshikazu Wada. 'Conic-based algorithm for visual line estimation from one image'. In: *Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings.* IEEE. 2004, pp. 260–265.

[162] Haiyuan Wu, Yosuke Kitagawa, Toshikazu Wada, Takekazu Kato and Qian Chen. 'Tracking iris contour with a 3D eye-model for gaze estimation'. In: *Computer Vision–ACCV 2007: 8th Asian Conference on Computer Vision, Tokyo, Japan, November 18-22, 2007, Proceedings, Part I 8*. Springer. 2007, pp. 688–697.

[163] Limei Xiao et al. 'Gaze Prediction based on Long Short-Term Memory Convolution with Associated Features of Video Frames'. In: *Computers and Electrical Engineering* 107 (2023), p. 108625.

[164] Jing Xie and Xueyin Lin. 'Gaze direction estimation based on natural head movements'. In: *Fourth International Conference on Image and Graphics (ICIG 2007)*. IEEE. 2007, pp. 672–677.

[165] Xuehan Xiong, Zicheng Liu, Qin Cai and Zhengyou Zhang. 'Eye gaze tracking using an RGBD camera: a comparison with a RGB solution'. In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*. 2014, pp. 1113–1121.

[166] Bing Xu, Naiyan Wang, Tianqi Chen and Mu Li. 'Empirical evaluation of rectified activations in convolutional network'. In: *arXiv preprint arXiv:1505.00853* (2015).

[167] Pingmei Xu, Krista A. Ehinger, Yinda Zhang, Adam Finkelstein, Sanjeev R. Kulkarni and Jianxiong Xiao. 'Turkergaze: Crowdsourcing Saliency with Webcam Based Eye Tracking'. In: *CoRR* abs/1504.06755 (2015). URL: http://arxiv.org/abs/1504.06755.

[168] Tao Xu, Bo Wu, Yuqiong Bai and Yun Zhou. 'Ravengaze: A dataset for gaze estimation leveraging psychological experiment through eye tracker'. In: *2023 IEEE 17th international conference on automatic face and gesture recognition (FG)*. IEEE. 2023, pp. 1–6.

[169] Hirotake Yamazoe, Akira Utsumi, Tomoko Yonezawa and Shinji Abe. 'Remote and head-motion-free gaze tracking for real environments with automated head-eye model calibrations'. In: *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE. 2008, pp. 1–6.

[170] Yu Yu and Jean-Marc Odobez. 'Unsupervised representation learning for gaze estimation'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 7314–7324.

[171] Mingfang Zhang, Yunfei Liu and Feng Lu. 'Gazeonce: Real-time multi-person gaze estimation'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 4197–4206.

[172] Wen Zhang, Tai-Ning Zhang and Sheng-Jiang Chang. 'Gazing estimation and correction from elliptical features of one iris'. In: *2010 3rd International Congress on Image and Signal Processing*. Vol. 4. IEEE. 2010, pp. 1647–1652.

[173] X. Zhang. 'Gaze estimation and interaction in real-world environments'. In: (2018). cit. on p. 25.

[174] Xucong Zhang, Seonwook Park, Thabo Beeler, Derek Bradley, Siyu Tang and Otmar Hilliges. 'Eth-xgaze: A large scale dataset for gaze estimation under extreme head pose and gaze variation'. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*. Springer. 2020, pp. 365–381.

[175] Xucong Zhang, Seonwook Park and Anna Maria Feit. 'Eye gaze estimation and its applications'. In: *Artificial Intelligence for Human Computer Interaction: A Modern Approach* (2021), pp. 99–130.

[176] Xucong Zhang, Yusuke Sugano, Mario Fritz and Andreas Bulling. 'Appearance-based gaze estimation in the wild'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 4511–4520.

[177] Xucong Zhang, Yusuke Sugano, Mario Fritz and Andreas Bulling. 'It's Written All Over Your Face: Full-face Appearance-based Gaze Estimation'. In: *Proceedings of the IEEE Computer Vision and Pattern Recognition Workshop (CVPRW)*. 2017.

[178] Xucong Zhang, Yusuke Sugano, Mario Fritz and Andreas Bulling. 'Mpiigaze: Real-world dataset and deep appearance-based gaze estimation'. In: *IEEE transactions on pattern analysis and machine intelligence* 41.1 (2017), pp. 162–175.

[179] Yufeng Zheng, Seonwook Park, Xucong Zhang, Shalini De Mello and Otmar Hilliges. 'Self-Learning Transformations for Improving Gaze and Head Redirection'. In: *arXiv preprint arXiv:2010.12307* (2020).

[180] Xiaolong Zhou, Jianing Lin, Jiaqi Jiang and Shengyong Chen. 'Learning a 3D Gaze Estimator with Improved iTracker Combined with Bidirectional LSTM'. In: *2019 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE. 2019, pp. 850–855.

[181] Jie Zhu and Jie Yang. 'Subpixel eye gaze tracking'. In: *Proceedings of Fifth IEEE International Conference on Automatic Face Gesture Recognition*. IEEE. 2002, pp. 131–136.

[182] Wangjiang Zhu and Haoping Deng. 'Monocular free-head 3d gaze tracking with deep learning and geometry constraints'. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 3143–3152.

[183] Zhiwei Zhu and Qiang Ji. 'Novel eye gaze tracking techniques under natural head movement'. In: *IEEE Transactions on biomedical engineering* 54.12 (2007), pp. 2246–2260.

Appendices

List of Abbreviations

ANN – Artificial Neural Network

BE – Both Eyes

CNN – Convolutional Neural Network

CV – Computer Vision

DL – Deep Learning

FC – Fully Connected

GPU – Graphics Processing Unit

LSTM – Long Short-Term Memory

MAE – Mean Absolute Error

ML – Machine Learning

MSE – Mean Squared Error

NT – No Transfer (Without Pre-trained Weights)

RE – Right Eye

LE – Left Eye

RMSE – Root Mean Squared Error

SGD – Stochastic Gradient Descent

TL – Transfer Learning

List of Figures

List of Tables