Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.063059

ARTICLE

Study on Eye Gaze Detection Using Deep Transfer Learning Approaches

Vidivelli Soundararajan*, Manikandan Ramachandran* and Srivatsan Vinodh Kumar

School of Computing, SASTRA Deemed University, Thanjavur, 613401, Tamil Nadu, India

*Corresponding Authors: Vidivelli Soundararajan. Email: vidivelli@cse.sastra.edu;
Manikandan Ramachandran. Email: srmanimt75@gmail.com

Received: 03 January 2025; Accepted: 07 April 2025; Published: 19 May 2025

ABSTRACT: Many applications, including security systems, medical diagnostics, and human-computer interfaces, depend on eye gaze recognition. However, due to factors including individual variations, occlusions, and shifting illumination conditions, real-world scenarios continue to provide difficulties for accurate and consistent eye gaze recognition. This work is aimed at investigating the potential benefits of employing transfer learning to improve eye gaze detection ability and efficiency. Transfer learning is the process of fine-tuning pre-trained models on smaller, domain-specific datasets after they have been trained on larger datasets. We study several transfer learning algorithms and evaluate their effectiveness on eye gaze identification, including both Regression and Classification tasks, using a range of deep learning architectures, namely AlexNet, Visual Geometry Group (VGG), InceptionV3, and ResNet. In this study, we evaluate the effectiveness of transfer learning-based models against models that were trained from scratch using eye-gazing datasets on grounds of various performance and loss metrics such as Precision, Accuracy, and Mean Absolute Error. We investigate the effects of different pre-trained models, dataset sizes, and domain gaps on the transfer learning process, and the findings of our study clarify the efficacy of transfer learning for eye gaze detection and offer suggestions for the most successful transfer learning strategies to apply in real-world situations.

KEYWORDS: Eye gaze detection; transfer learning; deep learning; AlexNet; VGG; InceptionV3; ResNet; domain adaptation; fine-tuning

1 Introduction

Identifying eye movements or pupil locations from pictures or frames received through live video processing is called eye gaze detection or eye tracking. Numerous businesses and day-to-day applications employ this technology extensively. It is used to provide virtual control of systems through gaze and blinks in smartphone cameras for bio-metric verification and in computer services for people with impairments. The core of eye gaze detection is composed of deep learning and image recognition algorithms. To create effective detection systems, pre-existing algorithms can be adapted or changed to meet particular needs. Apart from its application in authentication and assistive technologies, eye gaze tracking finds utility in human behavior research, evaluating the usability of user interfaces, and developing assistive technology for those with disabilities. Researchers like Asmetha Jeyarani et al. [1] have incorporated eye tracker biomarkers in deep learning algorithms to detect Autism Spectrum Disorder (ASD). Furthermore, eye gaze detection provides insights into how viewers respond to commercials and other visual stimuli, which is helpful for marketing studies. Robust eye gaze tracking or detection is highly advantageous in many domains. For instance, labor and time can be saved by using facial and eye movement tracking to spot mistakes in online test invigilation. Similar to this, eye tracking has been demonstrated by Rahman et al. [2], which can be

employed in educational virtual reality environments to identify inattentive pupils. Eye gazing data research allows businesses to make design improvements based on customer attention patterns, which improves user experience and marketing. Moreover, visual cues in vision-based navigation guide and control object motions, opening up new applications for robotics, self-driving automobiles, and uncrewed aerial vehicles (UAVs). Vision-based navigation may be responsive and dynamic in a variety of sectors because computer vision algorithms extract pertinent information from visual input. Among these traits are route planning, obstacle detection, and landmark identification. The ability to capture and evaluate pupil positions or eye motions from still or moving images has made eye gaze detection, or eye tracking, an essential tool in many fields.

Eye gaze detection technology has attracted much interest from a variety of industries and academic fields, such as psychology, human-computer interaction, marketing, and healthcare. Researchers and practitioners may learn a great deal about cognitive processes, behavioral patterns, and user interactions by studying where people focus their eyes. The utilization of eye gaze detection techniques in various fields has risen, but the existing models are often computationally intensive and require large amounts of data to train. Issues like occlusions, blur, varying light conditions and other real-world difficulties contribute to lack of robustness and accuracy of the said models. Due to the aforementioned reasons, exploring various transfer learning approaches via the existing lightweight pre-trained models is considered significant in order to improve usability, reliability and accessibility. With an emphasis on its applicability in various real-world circumstances, this paper explores the advantages, techniques, and applications of eye gaze recognition.

We have made an in-depth analysis of the current eye gaze detection systems that utilize deep learning and image recognition methods and the enhancements in user experience and market changes as the result of their advancements. In order to know the flexibility and effectiveness of gaze detection in various fields, we have also analyzed the new research and case studies relating to them. This opens up new avenues for comprehension and advances research and development in human-computer interaction. We hope that this work will provide a complete understanding of the capabilities and significance of eye gaze detection in the modern environment.

2 Related Works

The recent developments in deep learning algorithms have promoted hugely the remarkable progress that the field of eye gaze detection has currently achieved. Many manually trainable and pre-trained models are made available on the market that offers adaptable solutions, that is, can be curated to the specific requisites of the usage of eye detection tasks in various applications.

2.1 Approaches in Various Applications

In addition to traditional model-based solutions, novel techniques have been developed and proposed in order to address specific challenges in eye gaze identification. For example, Morimoto et al. [3] proposed employing CCD sensors to record and process ocular data instead of traditional image-based approaches. This method is a fast, robust and inexpensive proposal for an eye gaze detection system. Chen et al. demonstrated a real-time human-computer interaction (HCI) system [4]. It uses eye gaze data to compare the distribution of the sclera area and the relative distance between eyelashes. Their approach proved that when geometry and deep learning algorithms are used with Mask Region-based Convolutional Neural Network (R-CNN) for segmentation, they may achieve exceptional accuracy. Another intriguing method is S2LanGaze, presented by Sun et al. [5], which uses semi-supervised learning for eye gaze identification in glass wearers. S2LanGaze beats typical Convolutional Neural Network-based (CNN-based) algorithms when considering the efficiency of the methods. While their model performed great while training and with datasets that

resemble close conditions to the training dataset, it still requires improvements on prediction of unseen data and/or real-world settings. Nonetheless, this proposal offers a promising avenue for future study in eye gaze detection by including auxiliary tasks in the learning process.

de Lope et al. [6] together have compared a collection of six well-known pre-trained models, namely VGG19, ResNet50, InceptionV3, Xception, DenseNet and Inception-ResNet-V2. They have trained these models on the eye gaze dataset, which consists of images that contain both eyes of the individual. Their objective was to find the most efficient model among the aforementioned models for HAR (Human Activity Recognition), which in this case is the behavioral analysis of the user's gaze ethograms that is obtained by the user's gaze fixation on the computer screen. This could significantly help personnel in the fields of psychology and healthcare (in some cases). Kong et al. [7], along with fellow researchers, have proposed a system/methodology for eye tracking in smartphones and utilizing it to perform different activities. They have trained a state-of-the-art model by Valliappan et al. [8] using GazeCapture eye dataset to attain the said purpose. The model was deployed in iPhone 12 on EyeMU, which is a javascript application. The system is structured in a highly gated and double-trigger approach, that is, via Mediapipe's face mesh model. First, it checks if there is a user present and, if yes, their orientation. If not, the application is put to sleep. Then, the target fixation check is done on the user's gaze. If the target fixation is positive, then the user's gaze is utilized for performing the specified activities, and if the user is looking elsewhere, the motion detector is deactivated. They have achieved an accuracy of 97.2%, which is excellent, but the mean Euclidean error in prediction is 1.7 cm, which is over the desired threshold. The EyeMU's battery usage also appears to be high since it is a JavaScript application. However, with further advancements in the field, these limitations could be tackled. Furthermore, Zhang et al. [9] proposed a novel strategy for achieving robust eye identification by adopting a wholly acquired gaze-altering route. Their method is more effective at differentiating eye features and predicting the eye gaze direction since it combines object patches and extracts information using convolutional neural networks.

2.2 Deep Learning-Based Methods

Deep learning-based techniques have changed eye gaze detection, allowing for more accurate, precise and practical detection models in a wide range of domains. Numerous surveys and reviews are continuously undertaken by researchers [10–14], which provides us with an in-depth comparative analysis along with a solid theoretical foundation for the development of more advanced models and techniques [15–19]. Furthermore, researchers have developed specialized models to address industry-specific needs [20–23]. Wearable technologies such as head-mounts and smart glasses are becoming increasingly significant for eye tracking mainly due to their ease of use and portability. Nsaif, Wan, and Shehu et al. [24–26] have introduced various techniques and have conducted comparative analyses of the existing methodologies to advance this field. Liu et al. [27] have conducted substantial research comparing and evaluating various deep-learning approaches across a variety of datasets and settings. Their findings will influence future studies in the field since they provide critical information on the usefulness of various models and estimating methodologies in a range of contexts. Krafka et al. [28], along with the corresponding authors, have developed the first large-scale crowdsourced dataset, known as GazeCapture, for eye tracking in mobile devices. They have used the AMT (Amazon Mechanical Turk) as a platform for this purpose. They have also proposed a novel CNN-based model called iTracker. This model was trained by using the GazeCapture dataset, and it has been evaluated under both unconstrained and calibrated conditions. Since the inference time of predictions is not the primary concern in this work but rather the accuracy of predictions, the said model achieved remarkable results, considering the errors being as low as 1.04 and 1.69 cm in smartphones and tablets, respectively. This work has set a benchmark/standard for the works that followed in this domain.

Gunawardena et al. [29], together with fellow researchers, have proposed ensemble methods for eye tracking using smartphones and evaluated them on the grounds of various performance and loss metrics. This method included the combination of CNN with Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), respectively. These models were also compared with iTracker method. Both CNN + LSTM and CNN + GRU have achieved admirable results while being deployed in real-time. However, the inference time for the predictions averages 4~4.5 s in these models, which is not a feasible metric.

Badgujar et al. [30] addressed major safety issues by developing a gaze-tracking system that allows drivers to detect instances of driving attention. This application illustrates how eye gaze detection may improve road safety and minimize the number of incidents caused by distracted drivers. Furthermore, advances in eye gaze detection have made a significant impact in resolving problems for some clusters of users, such as those who wear spectacles. Eom et al. [31] presented a tailored approach that can reliably monitor eye movements in a variety of lighting situations, accommodate glasses wearers, and deliver strong performance in real-world settings. Although the proposed method appears to work well with glass wearers, when the non-glass wearers' metrics were compared to those of the glass wearers' metrics, there appears to be a considerable amount of errors present in the latter scenario, which needs more tweaking.

In conclusion, the wide variety of techniques and methods used in eye gaze detection highlights the importance of this topic in a number of applications. Eye gaze detection systems that are more precise, effective, and easily accessible are becoming a reality because of the ongoing improvements in the integration of deep learning techniques with creative solutions.

3 Dataset Preparation

Fig. 1 displays the sample images from the SBVPI dataset [32] that we utilized to build a robust eye gaze recognition algorithm. This dataset is a collection of eye images that were obtained from 55 subjects using a DSLR (Digital Single-Lens Reflex) camera set at high resolution. The acquisition of this dataset was primarily focused on sclera recognition but also intended to use periocular and iris recognition techniques. A total of 1398 images (usable eye images—regarding our objective) were obtained from this dataset for training our models.

In order to efficiently enhance the amount of dataset for training deep learning models, we employed augmentation approaches. Subsets for testing, validation, and training were then created from the supplemented dataset, maintaining the ratio at 70:20:10.

For independent model testing after training, we utilized an additional dataset (Fig. 1) with 10,532 images that we got from Kaggle, primarily intended for training models which are focused on controlling a wheelchair via eye gaze. Additionally, this dataset was expanded to offer a comprehensive assessment environment. We also created a bespoke testing dataset in order to replicate various deployment situations and evaluate the flexibility of the model. There were 360 augmented photos in this collection, each with both the left and right eye. The purpose of purposefully including a range of eye angles was to assess how well the model performed in unusual circumstances.

We ensured thorough evaluation and refinement of our eye gaze detection algorithms through the utilization of several datasets and a systematic testing methodology. This tactic makes it simpler to assess and fine-tune the algorithm in-depth in order to increase its robustness and functionality across a range of real-world scenarios.

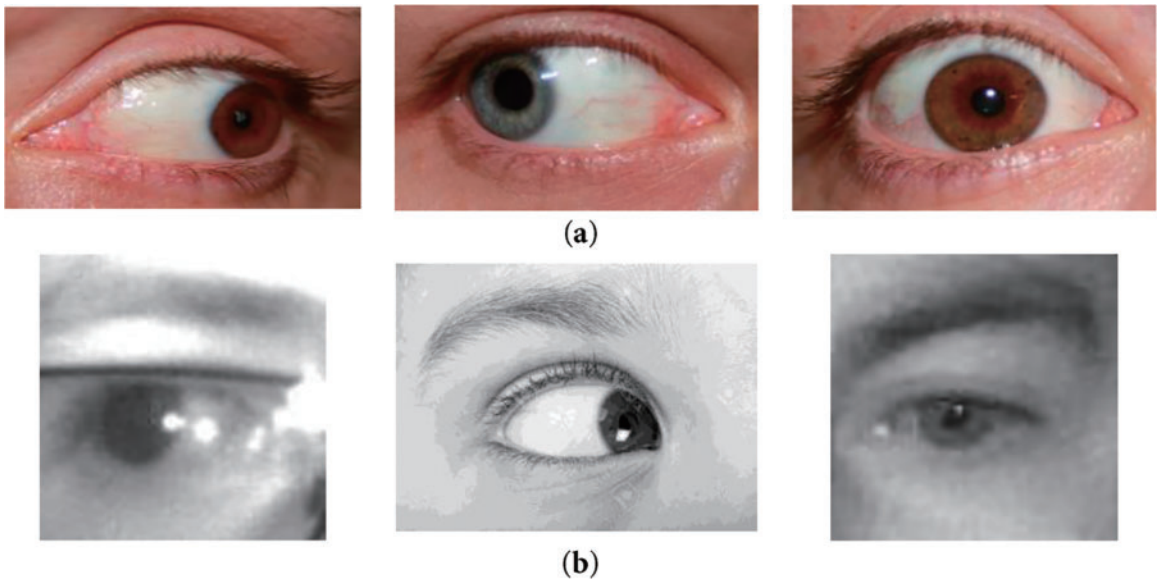

Figure 1: Sample images from the datasets. **(a)** SBVPI dataset (Reprinted with permission from [32–35]); **(b)** Kaggle Dataset (Reprinted with permission from [36])

Pre-Processing and Augmentation

To guarantee that our dataset closely reflects real-world settings, we use a variety of preprocessing approaches in this article, providing a strong basis for reliable eye gaze identification. All photos are first downsized to fit the input dimensions given by the corresponding models, which for ResNet-101 are 224×224. We then divide each pixel's value by 255.0 to normalize the image's pixel values to the range $[0, 1]$. Furthermore, we map the labels to their corresponding numerical values using a predetermined dictionary, which is a crucial step in the preparation of the dataset. Then, we do one-hot encoding on the labels.

Next, we apply Albumentations functions to improve the model's flexibility in different scenarios and the variety of the dataset. Albumentations is a popular image augmentation library for Python that is very helpful in computer vision and deep learning applications. It provides a number of photo-enhancement techniques, including geometric alterations, color tweaks, and noise addition. These changes increase the diversity of the training set, which improves the performance and generalization of the model.

The following are a few typical augmentation features offered by albumentations:

- **Random cropping:** Crops the picture to a predetermined size after choosing a random area of the picture.
- **Random brightness and contrast adjustments:** Varies the image's contrast and brightness at random.
- **Random gamma adjustments:** Applies gamma correction to the image, changing its brightness.
- **Random blur and sharpening:** Arbitrarily sharpens or adds blur to the picture.
- **Random noise addition:** Adds arbitrary noise to the picture, like salt-and-pepper or Gaussian noise.
- **Colour jittering:** This allows us to alter the brightness and contrast along with hue and saturation changes.

Deep learning models may utilize a more varied and reliable training dataset by applying and combining various augmentation methods sequentially, producing enhanced copies of the original pictures. Albumentations are widely utilized in a variety of computer vision and machine learning applications because of their adaptability, effectiveness, and simplicity of integration into current processes.

To get the most significant results from deep learning models, training data must be fed into them. The process of augmenting a dataset not only increases its quality but also its quantity, providing the model with a more extensive training set. This method strengthens the model's resilience and flexibility to a broader range of input patterns while simultaneously reducing the chance of overfitting. In essence, these augmentation strategies make the dataset 15 times larger than it was initially and much more adaptable for optimal model training, which significantly improves the efficacy and efficiency of the training process.

4 Transfer Learning Approaches

4.1 Problem Setting

This work is aimed at evaluating and fine-tuning various pre-trained algorithms for Eyegaze detection, namely VGG16, ResNet101, AlexNet and InceptionV3, which are some of the staple pre-trained CNN algorithms. The aforementioned models are trained with SBVPI eye dataset, which consists of RGB, sclera, iris, pupil, periocular and vascular eye images, and we extract the RGB images from the said dataset for training our models.

The models are tasked with performing both multi-class classification and regression simultaneously. The classification task has three possible outputs, namely Left, Right and Straight, along with an automatic regression-based image cropping at a fixed offset rate, which is done in order to avoid as much background noise from the input eye images as possible and to center the eye in each image further. The models are trained with the objective of performing robustly with improved generalization across the unseen data.

4.2 Model Architectures

For eye gaze prediction, which includes a classification task of 3 classes, namely Left, Right and Straight, and a regression task in order to enhance the localization of the eyes in the images, we applied four cutting-edge deep learning models: AlexNet, VGG-16, InceptionV3, and ResNet-101. These models are well-known for their proficiency in photo classification tasks and are utilized extensively in a wide range of computer vision applications. We applied these models to assess their performance and fit for the challenging task of eye gaze prediction. Each model has unique benefits and traits that suit it for different kinds of data and activities. Because of its creative architecture, AlexNet excels in processing hierarchical patterns in images. VGG-16 offers a trustworthy baseline for comparison because of its well-known simplicity, consistent architecture, and defined design principles. However, the distinctive residual connections of ResNet101 allow the training of intense networks, which might be helpful in identifying complex patterns in eye gaze data. Known for its efficient design, InceptionV3 excels at tasks like classifying pictures and identifying objects with remarkable precision. Its depth and complexity allow it to catch fine visual details, while regularization methods like dropout help it become more broadly applicable. The scientific community's confidence in its reliability and effectiveness is further evidenced by its extensive application in transfer learning. By contrasting these models, we want to learn more about the benefits and drawbacks of each for eye gaze prediction tasks and ultimately aid in the development of trustworthy and precise eye gaze detection systems.

4.2.1 AlexNet

After becoming victorious in the 2012 ImageNet Large Scale Visual Recognition Challenge (ILSVRC), the deep convolutional neural network architecture known as AlexNet garnered much attention. Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton, and others designed it. Its structure consists of several convolutional layers that are succeeded by completely connected layers [37]. The layers of AlexNet's architecture are as follows:

Number of Parameters: ~61 million parameters.

Input Layer: The network takes input images of size 227 × 227 pixels with three color channels (RGB).

Convolutional Layers: AlexNet begins with five convolutional layers, with a max-pooling layer positioned after each. From the input picture, these convolutional layers learn to extract features. Because the convolutional layers' filters have limited receptive fields—such as 3 by 3—they may identify specific patterns inside the picture.

Activation Function: An activation function known as a rectified linear unit (ReLU) is used after every convolutional layer and fully connected layer. ReLU gives the network non-linearity, which helps it recognize intricate patterns.

Normalization: Following the first and second convolutional layers is the use of Local Response Normalization (LRN). In order to facilitate improved generalization, LRN helps standardize the responses of nearby neurons.

Pooling Layers: After the convolutional layers, the max-pooling layers downsample the feature maps, keeping significant features while decreasing their spatial dimensions. Max-pooling contributes to the representation's resistance to slight input distortions and translations.

Fully Connected Layers: Three fully linked layers make up the network after the convolutional and pooling layers. These layers translate the features to the output classes by combining the knowledge gained by the convolutional layers. A third fully connected layer of 1000 neurons, representing the 1000 classes in the ImageNet dataset used for training, comes after the first two fully connected layers, each with 4096 neurons.

Output Layer: The output layer in the ImageNet dataset consists of one thousand neurons, or one class each. The raw output is converted into class probabilities by the output layer using a softmax activation function. The general architecture of AlexNet is displayed in Fig. 2.

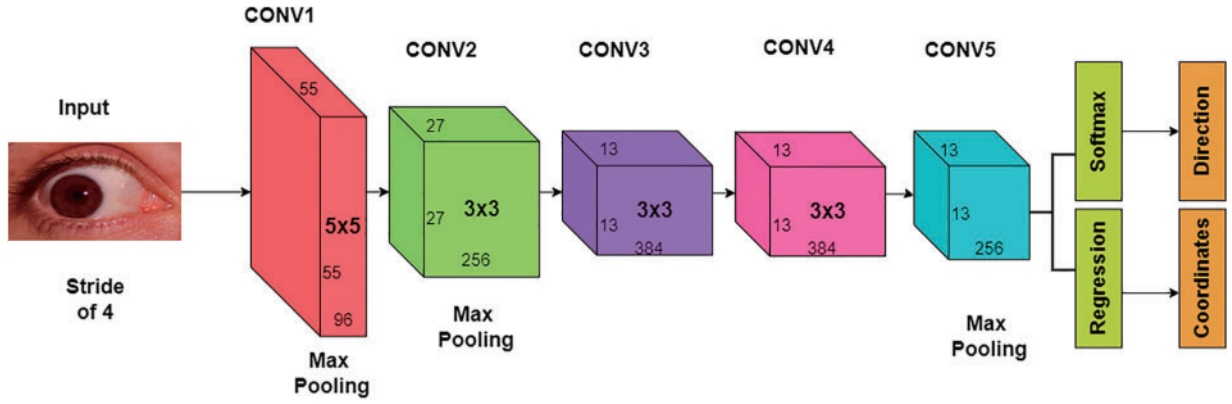

Figure 2: AlexNet architecture

4.2.2 VGG16 Architecture

The Visual Geometry Group (VGG) of University of Oxford developed and proposed VGG16, which is a convolutional neural network architecture. It is a well-known algorithm due to its ease of use, along with its success in image classification. This provides a comprehensive description of the VGG-16's architecture:

Number of Parameters: ~138 million parameters.

Input Layer: The images are inputted into the VGG-16 network at the resolution of 224 × 224 pixels, and with the 3 color channel configuration, that is, RGB.

Convolutional Blocks: A total of thirteen convolutional layers are stacked on top of one another and are represented by a sum of five blocks that comprise VGG-16. So, many convolutional layers are present in each block, and they are either followed by or follow max-pooling layer(s).

- **Block 1:** Presented with 64 filters each, there are two convolutional layers present here, along with a 3 × 3 kernel size. They come after max-pooling layer with a 2 × 2 window size and a stride of 2.
- **Block 2:** In this block, there are, again, two convolutional layers with 128 filters and 3 × 3 kernel sizes present, which comes after max-pooling.
- **Block 3:** Three convolutional layers with 256 filters and 3 × 3 kernel sizes each come after max-pooling.
- **Block 4:** Three convolutional layers with 512 filters and 3 × 3 kernel sizes come after max-pooling.
- **Block 5:** Three 512-filter convolutional layers with a 3 × 3 kernel size each are followed by max-pooling.

Activation Function: Following each convolutional layer comes the rectified linear unit (ReLU) activation function. The network gains non-linearity via ReLU, which facilitates the recognition of complex patterns.

Fully Connected Layers: VGG-16 contains three fully connected layers with 4096 neurons each, following the convolutional blocks. These layers translate the features to the output classes by combining the knowledge gained by the convolutional layers.

Dropout: To avoid over-fitting, dropout regularization is done to the ultimately linked layers. During training, it randomly removes a portion of neurons in an effort to lessen neuronal co-adaptation.

Output Layer: The last completely linked layer is subjected to a softmax activation function at the output layer. The raw output is transformed into class probabilities.

The deep convolutional layers of the VGG-16 architecture, which use tiny 3 × 3 filters, are followed by max-pooling layers for spatial downsampling. As seen in Fig. 3, our eye gaze detection program has extensively embraced this straightforward and consistent design, which facilitates efficient feature learning.

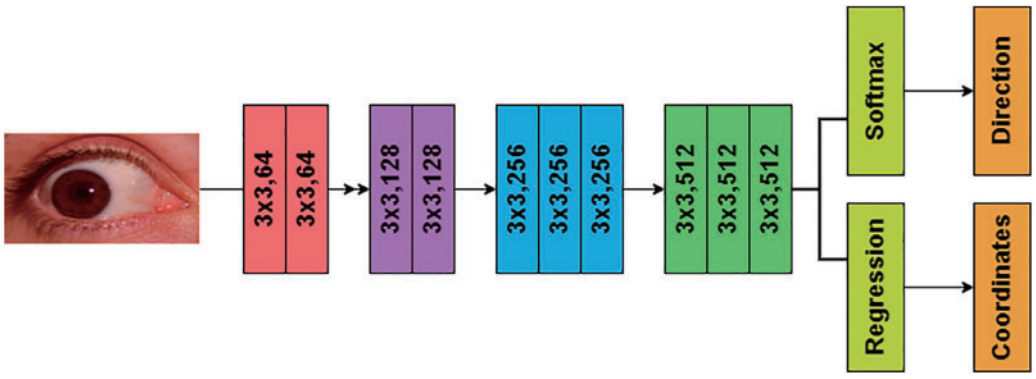

Figure 3: VGG-16 architecture

4.2.3 ResNet-101 Architecture

Convolutional neural networks like ResNet-101 are part of the ResNet (Residual Network) family of neural network architectures. He et al. [38] presented ResNet in their groundbreaking article "Deep Residual Learning for Image Recognition", published in 2015. An expanded version of the original ResNet model with 101 layers is called ResNet-101.

The detailed description of the ResNet-101 architecture:

Number of Parameters: ~45 million parameters.

Input Layer: Similar to other CNN designs, ResNet-101 accepts input pictures with three RGB color channels that are fixed in size, usually 224 by 224 pixels.

Convolutional Layers: In order to lower spatial dimensions, ResNet-101 begins with a few initial convolutional layers and then max-pooling. From the input pictures, the convolutional layers acquire the ability to extract low-level information.

Residual Blocks: The creation of residual blocks is the main ResNet breakthrough. These blocks have shortcut connections or skip connections, which omit one or more convolutional layers. ResNet solves the vanishing gradient issue by including these skip connections, which makes it possible to train extremely deep networks.

Basic Residual Block: There are two convolutional layers present in every basic residual block of ResNet-101, and they are paired with Batch Normalization and ReLU activation algorithms. The next stage is where the skip connection comes into picture. This allows the addition of input to the output. By the aforementioned skip connection, the residual mappings are learned by the network, which aids in optimization.

Bottleneck Residual Block: In order to lower the processing complexity at higher layers, bottleneck residual layers are used by the ResNet-101. A total of three convolutional layers are comprised here: an additional 1×1 layer for channel augmentation for feature extraction—3×3 layer, and a 1×1 layer for input channel reduction.

Identity Shortcut Links: Identity mappings, that is, the input being directly added to the output with no changes, are attained by the presence of Skip connections in each block of ResNet-101. While being trained, gradients may now easily travel around the network.

Global Average Pooling: Global Average Pooling is utilized in ResNet-101 in order to integrate spatial information after the convolutional layers and residual blocks across the feature maps. Throughout the pooling process, by averaging the values of each feature map, a foxed-length feature vector that encompasses the whole input picture is generated.

Fully Connected Layer: Last but not least, ResNet-101 features a fully linked layer that employs softmax activation for classification tasks. This layer maps the combined features to the output classes, producing class probabilities for the input image.

The deep stack of residual blocks that characterizes ResNet-101's design (Fig. 4) allows for the training of very deep networks while reducing the issue of the vanishing gradient. This architecture is still in widespread usage in the computer vision community, having produced state-of-the-art results on a variety of image identification tasks.

4.2.4 InceptionV3 (Google-Net)

Developed for image identification applications, Google researchers unveiled InceptionV3, a convolutional neural network architecture, in 2015. Here is an overview of its architecture:

Number of Parameters: ~23.8 million parameters.

Input Shape: InceptionV3 accepts three RGB color channels as input images, which are typically 299 by 299 pixels. This architecture is similar to earlier CNN models.

Number of Blocks: Convolutional layers, auxiliary classifiers, and inception modules are some of the components that comprise InceptionV3.

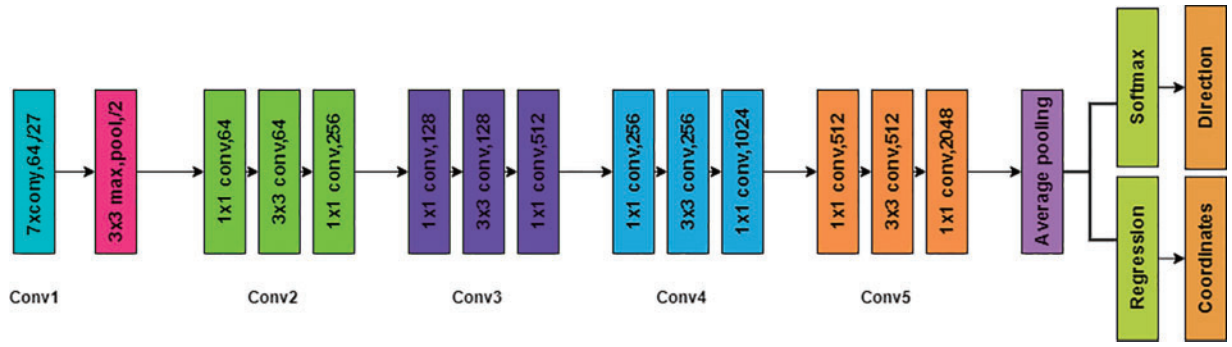

Figure 4: ResNet-101 architecture

Convolutional Layers: Max-pooling operations are carried out in order to decrease the spatial dimensions after the network begins with initial convolutional layers. These layers extract low-level characteristics from the input images.

Inception Modules: The basis of InceptionV3 is the presence of parallel convolutional branches with various filter sizes and processing, which consists of the Inception module. Thanks to these components, the network may effectively gather characteristics of various sizes.

Auxiliary Classifiers: InceptionV3 includes additional classifiers in intermediate layers to solve the problem of vanishing gradients. These auxiliary classifiers provide additional regularization during training, which aids in the convergence of the model.

Description of Each Block: In every Inception module, Max-pooling operations and parallel convolutional branching, including 1×1, 3×3, and 5×5 convolutions, are often seen. By concatenating these branches, the output of the module is produced, which in turn enables the network to collect characteristics at various spatial scales.

Global Average Pooling: From several feature maps using global average pooling, which comes after the convolutional layers and the Inception modules, InceptionV3 integrates spatial data. After this procedure, in order to create a fixed-length feature vector that represents the whole input picture, the values of each feature map are averaged.

Fully Connected Layer: The last component of InceptionV3 is a fully connected layer with Softmax activation for classification tasks. This layer creates class probabilities for the input image by translating the combined features to the output classes.

The extensive stack of Inception modules that make up InceptionV3's architecture (Fig. 5) enables the network to collect features at various sizes effectively. The outstanding performance of InceptionV3 in image recognition tasks and its extensive use in the computer vision community can be attributed to these architectural aspects.

5 Experimental Analysis

We have gathered the results by making comparisons between the aforementioned pre-trained models, by which we acquire the best one among them and later compare it with other state-of-the-art ones. We have compared the above models on the grounds of various performance and loss metrics and have compiled the results in the below sections.

5.1 Experimental Setup

We used a PC, which comprises of Processor: AMD Ryzen 7 3750H with Radeon Vega Mobile Gfx (2.3 GHz base frequency, up to 4.0 GHz burst frequency); Graphics: AMD Radeon RX 5500M with 4 GB GDDR6 VRAM; Memory: 16 GB DDR4 RAM (expandable up to 64 GB); Storage: 512 GB NVMe SSD; Display: 15.6-inch Full HD (1920 × 1080 pixels) IPS-level panel with 144 Hz refresh rate; Operating System: Windows 11.

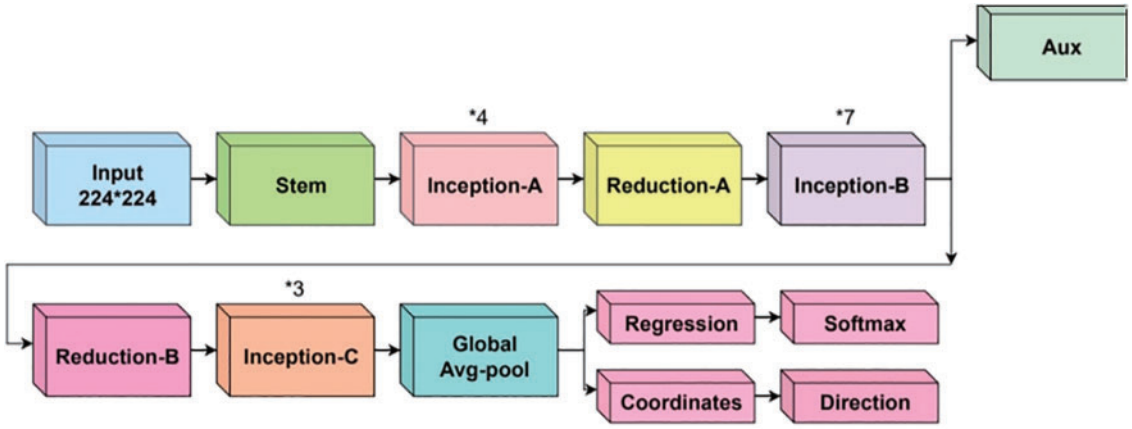

Figure 5: InceptionV3 architecture

Evaluation metrics (performance and loss metrics)

We have utilized classification accuracy and estimated classification loss for the predicted labels and localization loss for the predicted regression box coordinates. As mentioned in Section 4.1, the models perform classification and regression tasks simultaneously. So we utilize multi-task learning approach where we compute classification loss (categorical cross entropy loss) and regression loss (localization loss) jointly and total loss by summing them up for each batch while training. In our multi-class classification problem, the labels are generally one-hot encoded, so the positive classes alone keep their term at a loss. There is only one element different than zero in the target vector, so by discarding the elements in the summation which becomes zero due to target label, we can attain the categorical cross entropy loss function, which is a loss function of cross entropy for softmax multi-class function as shown in (1).

$$f(s)_i = \frac{e^{s_i}}{\sum_j^c e^{s_j}} \tag{1}$$

where, "s" represents output scores.

Cross Entropy Loss Formula:

$$CE = -\sum_I^C t_i log\left(f(s)_i\right) \tag{2}$$

Categorical Cross Entropy (Loss Function for Softmax Activation):

$$CCE = -log\left(\frac{e^{s_p}}{\sum_j^c e^{s_j}}\right) \tag{3}$$

Delta Coordinate Loss:

$$\delta_{coords} = \sum_{i=1}^{N} \left\| y|_{true,i}^{x,y} - \hat{y}|_{i}^{x,y} \right\|^2 \tag{4}$$

where, $y|_{true,i}^{x,y}$, $\hat{y}|_{i}^{x,y}$ are the true and predicted center coordinates of the *i*th bounding box, respectively.

Delta Size Loss:

$$\delta_{sizes} = \sum_{i=1}^{N} \left(\left\| y|_{true,i}^{w} - \hat{y}|_{i}^{w} \right\|^2 + \left\| y|_{true,i}^{h} - \hat{y}|_{i}^{h} \right\|^2 \right) \tag{5}$$

where, $y|_{true,i}^{w}$, $y|_{true,i}^{h}$ represent the true width and height of the *i*th bounding box, and $\hat{y}|_{i}^{w}$, $\hat{y}|_{i}^{h}$, represent the predicted width and height, respectively.

Localization Loss:

$$Localization\,Loss = \delta_{coords} + \delta_{sizes} \tag{6}$$

Total Loss:

$$L_{total} = L_{cross-entropy} + L_{localization} \tag{7}$$

Classification Accuracy:

$$Accuracy(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} 1\left(\widehat{y_i} = y_i\right) \tag{8}$$

Mean Absolute Error (for Regression):

$$MAE = \frac{1}{N \times 4} \sum_{i=1}^{N} \sum_{j=1}^{4} \left| y_{i,j}^{\wedge} - y_{i,j} \right| \tag{9}$$

5.2 Performance Assessment

We have evaluated the performance of various models on various datasets. The dataset, which was initially used for training, was divided into a ratio of 70:20:10 in order to provide 70% for training, 20% for testing, and the remaining 10% for validation. The dataset was augmented after being labeled, in order to multiply the initially available 1398 images from the licensed dataset by 12 times, resulting in a total of 16,776 images. So, 11,736 images were used for training, 3348 images were used for testing, and 1692 images were used for validation. We have also augmented the images from Kaggle Dataset of 44 by 30 times and have achieved a sum of 1320 images, which was used for validation purposes. This is done in order to make the models' training procedures more robust and work better on unseen images. Moreover, for more evaluation purposes, we have used our own dataset, which comprised 20 original images, which was again augmented by 8 times to produce 360 images.

We have obtained different loss metrics and accuracy metrics, as mentioned in previous section. We know that VGG-16 has 16 layers, InceptionV3 has 48 layers, ResNet-101 has 101 layers, and Alexnet has only 5 layers. This difference is evident in the results that we have obtained. Using more than one optimizer also helped us achieve better performance and in optimizing the model more. Furthermore, using different epochs also had a significant impact on the final optimized model. Using more epochs initially resulted in

over-fitting, and later, when we used relatively fewer epochs, the model attained under-fitting. We finally achieved optimal fitting of the epochs, which was 8, which produced neither under-fitting nor over-fitting. We also obtained validation loss for each epoch being run in training and then the respective classification loss and regression loss. This was done with both ADAM and ADAGRAD optimizers, and ADAM optimizer (Fig. 6) achieved the optimal curve out of the two.

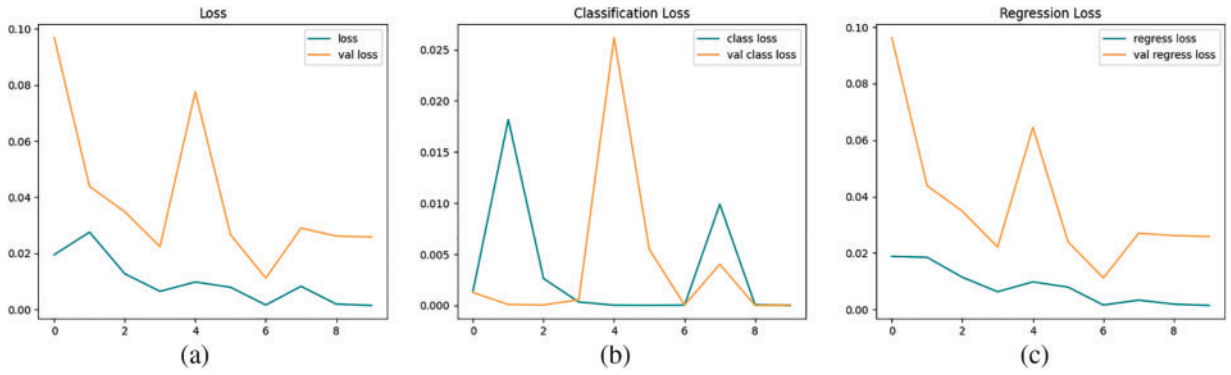

Figure 6: Loss graphs. **(a)** Total loss using ADAM optimizer in VGG16; **(b)** Classification loss using ADAM optimizer in VGG16; **(c)** Regression loss using ADAM optimizer in VGG16

5.3 Comparison within Pre-Trained Models

A comparison was made with multiple pre-trained models to find the one that performed well. The comparison models were ResNet101, VGG16, AlexNet and InceptionV3. Out of these, VGG-16 was found to outperform all the other pre-trained models and not just with simple margins. Moreover, it also seems to obtain the highest accuracy for each class along with the already mentioned overall advantage. We have calculated various accuracy metrics for classification and regression outputs of all the models in comparison, as shown in Tables 1 and 2.

Table 1: Training: The number of convolution blocks and the type of optimizers applied to the model

Dataset	Proposed model	Optimizer	Classification accuracy (with training data)	Classification accuracy (with validation data)	Mean absolute error (MAE) (with validation data)
SBVPI [32]	AlexNet	Adam	100%	97.5	0.05
		Adagrad	100%	96.4	0.07
	VGG16	Adam	100%	99.25	0.02
		Adagrad	100%	99	0.03
	ResNet-101	Adam	100%	98.1	0.06
		Adagrad	99%	98	0.05
	InceptionV3	Adam	100%	97.8	0.038
		Adagrad	99%	97.4	0.04

Table 2: Testing: The number of convolution blocks and the type of optimizer applied to the model

Dataset	Model	Optimizer	Classification accuracy	Mean absolute error (MAE) for regression
Kaggle dataset	AlexNet	Adam	64%	0.52
		Adagrad	54.1%	0.46
	VGG16	Adam	74.5%	0.18
		Adagrad	73.4%	0.21
	ResNet-101	Adam	69.9%	0.31
		Adagrad	37.5%	0.43
	InceptionV3	Adam	50.4%	0.42
		Adagrad	48%	0.40
Custom dataset	AlexNet	Adam	68.3%	0.38
		Adagrad	54.2%	0.42
	VGG16	Adam	84%	0.20
		Adagrad	64%	0.32
	ResNet-101	Adam	50%	0.44
		Adagrad	48.2%	0.41
	InceptionV3	Adam	40%	0.39
		Adagrad	38%	0.42

We have taken an average of the accuracy metrics that we have attained while comparing the models with different evaluation datasets by running multiple rounds of testing. Furthermore, VGG16, as mentioned previously, seems to achieve the desired results among the four models. It has the most optimal curve for the achieved class loss, regression loss and total loss, which aligns with validation class loss, regression loss and total loss, Fig. 6. This makes the model an ideal fit for eye gaze since it is neither over-fitted nor under-fitted. It also attained 0.9925 accuracies in training, and while testing, it had an average accuracy of 0.80 for the evaluation datasets, as provided in Fig. 7. This is far from the results of the other models, which average around the 0.58 mark in Table 2. Similarly, the average Mean Absolute Error results in 0.2 on unseen images for VGG16, while it is 0.45 for the other models in comparison. Some of the reasons that further help us understand why VGG16 has displayed better performance metrics among the given models across datasets are:

- AlexNet is a shallower model which cannot learn complex gaze patterns as effectively as its peers. At the same time, ResNet101 consists of a deep architecture with a large parameter count, making it prone to over-fitting when training with limited dataset.
- VGG16, due to its moderate depth and simple sequential architecture, allows it to generalize better, which results in its good performance across all the datasets, including unseen data.
- VGG16 performs best when combined with Adam optimizer, which has proven to perform better than Adagrad optimizer (which might indicate that the model suits Adam's adaptive learning rate convergence over Adagrad's monotonically decreasing learning rates).

5.4 Comparison with State-of-the-Arts

There are various state-of-the-art models/techniques for eye gaze estimation. We have compared VGG-16 (which produced the best results out of all the models that were compared, as mentioned in previous section) with CNN models, which have been used as a go-to for the same purpose. Again, VGG-16 seemed to perform better than some of the latter methods, as shown in Table 3.

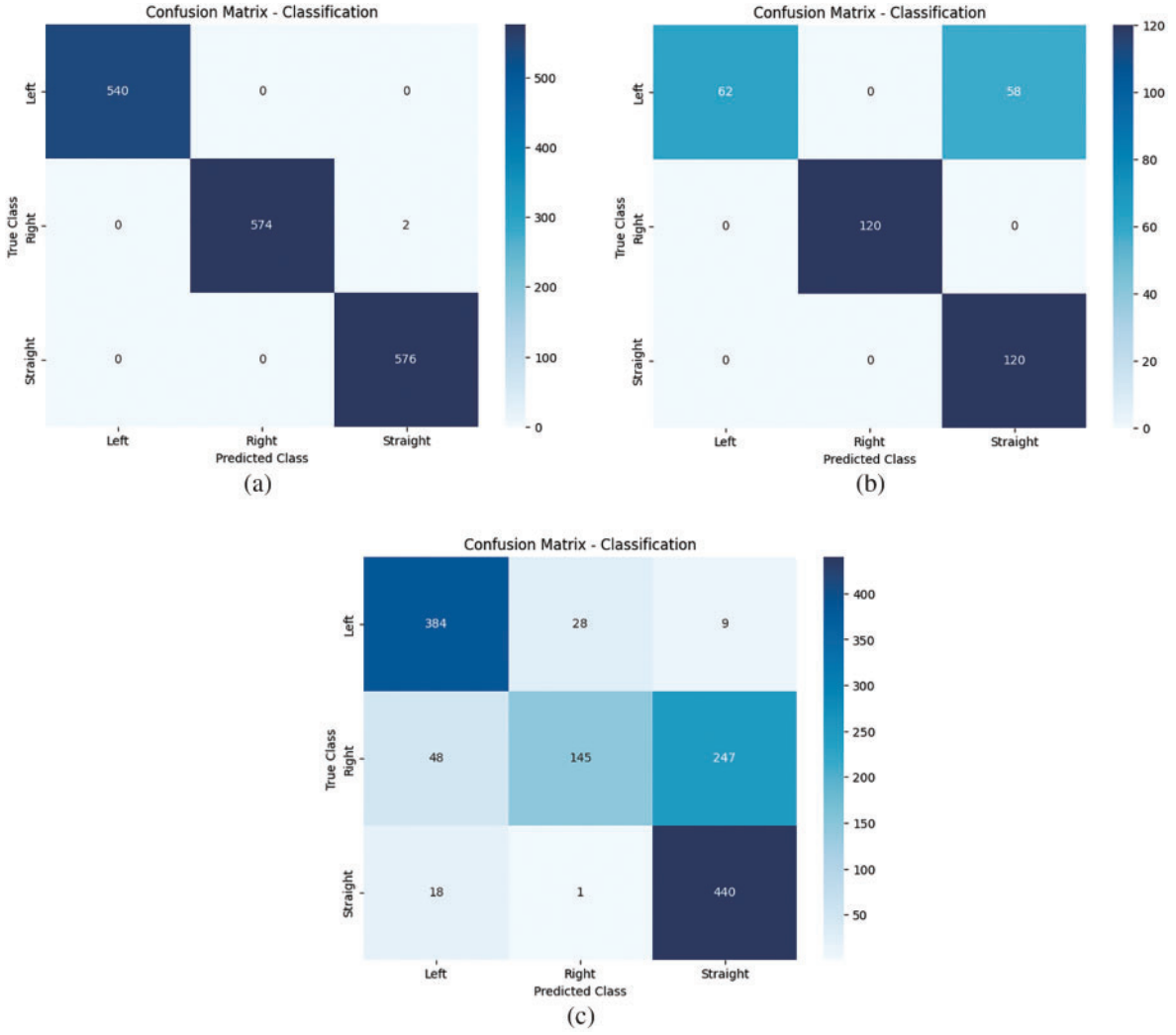

Figure 7: Confusion matrices for various datasets while training and testing using VGG-16, **(a)** Validation dataset (part of training), **(b)** Own dataset. **(c)** Kaggle dataset

As Table 3 presents, our VGG16 seems to achieve excellent results when compared with the existing state-of-the-art of CNN. Also, it performs well with unseen images with an accuracy of 84% (on a custom dataset that we prepared ourselves) and 75% (on the dataset which was attained from Kaggle), computing 79%–80% accuracy on average. We have achieved this by inducing dropout and regularization techniques for both our classification and regression layers and finding the optimal metrics after trial and error with various values.

Table 3: Comparison with existing work

S. No.	Year	Technique	Number of convolutional layers	Parameters	Filter sizes	Dataset used	Accuracy (in %)
1	2020	SegNet [39]	13	~138 million	3 × 3	SBVPI Dataset	94.9 ± 2.1
2	2020	DSeg (Unet-4P proposed) [40]	18	~0.12 million	3 × 3		98.2
3	2018	USS [41]	–	–	–		95.47
4	2024	OcularSeg [42]	–	~22.65 million	3 × 3		96.48
5	**2025**	**VGG16 (Proposed)**	13	~138 million	3 × 3		**99.25**

5.5 Failure Cases

VGG-16 faces difficulties while estimating the gaze from the images, which seem to be taken in dark/pitch black or gray-scale-like images, with a small quantity of blur, as shown in Fig. 8. In these cases, it has an accuracy of just 67%.

Figure 8: Failure cases

6 Conclusion

This paper provides the results of a comparison between various models for eye gaze detection. Moreover, the experiments and the results acquired have concluded that **VGG-16** produces desirable outcomes in training with an accuracy of 99% for classification and just 0.02 MAE (Mean Absolute Error) for regression. While evaluating it with the multiple unseen datasets, it achieved an average of 80% accuracy for classification and 0.19 MAE for regression. These results were compared with various pre-trained models using multiple optimizers and also with existing state-of-the-art ones, and they proved to perform well.

Our future work aims to utilize these studies and comparisons, as well as various explainable AI techniques, namely GradCAM and GradCAM++, in building a robust eye gaze detection system of our own. The objective of the said model is to be deployed on IoT vehicle automation applications, namely a wheelchair, which is to be operated only via eye gaze, for which the model is deployed on a wearable, namely a head-mount. This work is focused on the domain-IoT Vehicle automation using deep learning techniques

in the field of healthcare. It is primarily aimed at assisting people who are victims of Cerebral Palsy (CP) and also helping other individuals who are paralyzed due to accidents or by means of diseases. Cerebral Palsy is a condition that often affects children at birth or soon after it, where the victim is either entirely or partially paralyzed and is unable to perform any motor functions on their own. However, according to findings and statements from doctors and via real-life observation, in most cases, their IQ appeared to be decent, and this is a condition that can sometimes be overcome with the required professional assistance. So, our work is aimed at contributing towards the betterment of their lives and is intended to build their sense of confidence and instill a sense of freedom by helping them move.

Acknowledgement: The authors acknowledge the SASTRA Deemed University for providing resources to complete this article.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Vidivelli Soundararajan, Srivatsan Vinodh Kumar; data collection: Vidivelli Soundararajan; analysis and interpretation of results: Manikandan Ramachandran; draft manuscript preparation: Vidivelli Soundararajan, Srivatsan Vinodh Kumar. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The authors confirm that the data supporting the findings of this study are available within the article.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Asmetha Jeyarani R, Senthilkumar R. Eye tracking biomarkers for autism spectrum disorder detection using machine learning and deep learning techniques: review. Res Autism Spectr Disord. 2023;108(4):102228. doi:10.1016/j.rasd.2023.102228.
2. Rahman Y, Asish SM, Fisher NP, Bruce EC, Kulshreshth AK, Borst CW. Exploring eye gaze visualization techniques for identifying distracted students in educational VR. In: 2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR); 2020 Mar 22–26; Atlanta, GA, USA. p. 868–77. doi:10.1109/vr46266.2020.00009.
3. Morimoto CH, Koons D, Amir A, Flickner M. Pupil detection and tracking using multiple light sources. Image Vis Comput. 2000;18(4):331–5. doi:10.1016/S0262-8856(99)00053-0.
4. Chen H, Zendehdel N, Leu MC, Yin Z. Real-time human-computer interaction using eye gazes. Manuf Lett. 2023;35(10):883–94. doi:10.1016/j.mfglet.2023.07.024.
5. Sun Y, Zeng J, Shan S. Gaze estimation with semi-supervised eye landmark detection as an auxiliary task. Pattern Recognit. 2024;146(1):109980. doi:10.1016/j.patcog.2023.109980.
6. de Lope J, Graña M. Deep transfer learning-based gaze tracking for behavioral activity recognition. Neurocomputing. 2022;500(16):518–27. doi:10.1016/j.neucom.2021.06.100.
7. Kong A, Ahuja K, Goel M, Harrison C. EyeMU interactions: gaze + IMU gestures on mobile devices. In: Proceedings of the 2021 International Conference on Multimodal Interaction (ICMI'21); 2021 Oct 18–22; Montréal, QC, Canada. p. 577–85. doi:10.1145/3462244.3479938.
8. Valliappan N, Dai N, Steinberg E, He J, Rogers K, Ramachandran V, et al. Accelerating eye movement research *via* accurate and affordable smartphone eye tracking. Nat Commun. 2020;11(1):4553. doi:10.1038/s41467-020-18360-5.
9. Zhang S, Shen L, Zhang R, Yang Y, Zhang Y. Robust eye detection using deeply-learned gaze shifting path. J Vis Commun Image Represent. 2018;55(8):654–9. doi:10.1016/j.jvcir.2018.07.013.

10. Klaib AF, Alsrehin NO, Melhem WY, Bashtawi HO, Magableh AA. Eye tracking algorithms, techniques, tools, and applications with an emphasis on machine learning and Internet of Things technologies. Expert Syst Appl. 2021;166(1):114037. doi:10.1016/j.eswa.2020.114037.
11. Pathirana P, Senarath S, Meedeniya D, Jayarathna S. Eye gaze estimation: a survey on deep learning-based approaches. Expert Syst Appl. 2022;199(13):116894. doi:10.1016/j.eswa.2022.116894.
12. Akinyelu AA, Blignaut P. Convolutional neural network-based methods for eye gaze estimation: a survey. IEEE Access. 2020;8:142581–605. doi:10.1109/ACCESS.2020.3013540.
13. Tsukada A, Shino M, Devyver M, Kanade T. Illumination-free gaze estimation method for first-person vision wearable device. In: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops); 2011 Nov 6–13; Barcelona, Spain. p. 2084–91. doi:10.1109/ICCVW.2011.6130505.
14. Al-Moteri MO, Symmons M, Plummer V, Cooper S. Eye tracking to investigate cue processing in medical decision-making: a scoping review. Comput Hum Behav. 2017;66(3):52–66. doi:10.1016/j.chb.2016.09.022.
15. Henderson J. Human gaze control during real-world scene perception. Trends Cogn Sci. 2003;7(11):498–504. doi:10.1016/j.tics.2003.09.006.
16. Beymer D, Flickner M. Eye gaze tracking using an active stereo head. In: 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition; 2003 Jun 18–20; Madison, WI, USA. p. II-451. doi:10.1109/CVPR.2003.1211502.
17. Chauhan NK, Singh K. A review on conventional machine learning vs deep learning. In: 2018 International Conference on Computing, Power and Communication Technologies (GUCON); 2018 Sep 28–29; Greater Noida, India. p. 347–52. doi:10.1109/GUCON.2018.8675097.
18. Sun L, Song M, Liu Z, Sun MT. Real-time gaze estimation with online calibration. IEEE Multimed. 2014;21(4):28–37. doi:10.1109/MMUL.2014.54.
19. Huang H, Xu Y, Hua X, Yan W, Huang Y. A crowdsourced system for robust eye tracking. J Vis Commun Image Represent. 2019;60(2):28–32. doi:10.1016/j.jvcir.2019.01.007.
20. Wang FS, Wolf J, Farshad M, Meboldt M, Lohmeyer Q. Object-gaze distance: quantifying near-peripheral gaze behavior in real-world applications. J Eye Mov Res. 2021;14(1). doi:10.16910/jemr.14.1.5.
21. Brunyé TT, Drew T, Kerr KF, Shucard H, Weaver DL, Elmore JG. Eye tracking reveals expertise-related differences in the time-course of medical image inspection and diagnosis. J Med Imaging. 2020;7(5):051203. doi:10.1117/1.JMI.7.5.051203.
22. Chen JC, Yu PQ, Yao CY, Zhao LP, Qiao YY. Eye detection and coarse localization of pupil for video-based eye tracking systems. Expert Syst Appl. 2024;236(8):121316. doi:10.1016/j.eswa.2023.121316.
23. Zhu D, Moore ST, Raphan T. Robust pupil center detection using a curvature algorithm. Comput Meth Programs Biomed. 1999;59(3):145–57. doi:10.1016/S0169-2607(98)00105-9.
24. Nsaif AK, Ali SHM, Jassim KN, Nseaf AK, Sulaiman R, Al-Qaraghuli A, et al. FRCNN-GNB: cascade faster R-CNN with Gabor filters and Naïve Bayes for enhanced eye detection. IEEE Access. 2021;9:15708–19. doi:10.1109/ACCESS.2021.3052851.
25. Wan ZH, Xiong CH, Chen WB, Zhang HY. Robust and accurate pupil detection for head-mounted eye tracking. Comput Electr Eng. 2021;93(12):107193. doi:10.1016/j.compeleceng.2021.107193.
26. Shehu IS, Wang Y, Athuman AM, Fu X. Remote eye gaze tracking research: a comparative evaluation on past and recent progress. Electronics. 2021;10(24):3165. doi:10.3390/electronics10243165.
27. Liu J, Chi J, Yang H, Yin X. In the eye of the beholder: a survey of gaze tracking techniques. Pattern Recognit. 2022;132(3):108944. doi:10.1016/j.patcog.2022.108944.
28. Krafka K, Khosla A, Kellnhofer P, Kannan H, Bhandarkar S, Matusik W, et al. Eye tracking for everyone. arXiv:1606.05814. 2016.
29. Gunawardena N, Ginige JA, Javadi B, Lui G. Deep learning based eye tracking on smartphones for dynamic visual stimuli. Procedia Comput Sci. 2024;246:3733–42. doi:10.1016/j.procs.2024.09.183.
30. Badgujar P, Selmokar P. Driver gaze tracking and eyes off the road detection. Mater Today Proc. 2023;72(1):1863–8. doi:10.1016/j.matpr.2022.10.046.

31. Eom Y, Mu S, Satoru S, Liu T. A method to estimate eye gaze direction when wearing glasses. In: 2019 International Conference on Technologies and Applications of Artificial Intelligence (TAAI); 2019 Nov 21–23; Kaohsiung, Taiwan. p. 1–6. doi:10.1109/taai48200.2019.8959824.
32. The SBVPI dataset. [cited 2025 Jan 1]. Available from: http://sclera.fri.uni-lj.si/.
33. Vitek M, Das A, Pourcenoux Y, Missler A, Paumier C, Das S, et al. SSBC 2020: sclera segmentation benchmarking competition in the mobile environment. In: 2020 IEEE International Joint Conference on Biometrics (IJCB); 2020 Sep 28–Oct 1; Houston, TX, USA. p. 1–10. doi:10.1109/ijcb48548.2020.9304881.
34. Vitek M, Rot P, Štruc V, Peer P. A comprehensive investigation into sclera biometrics: a novel dataset and performance study. Neural Comput Appl. 2020;32(24):17941–55. doi:10.1007/s00521-020-04782-1.
35. Vitek M, Bizjak M, Peer P, Štruc V. IPAD: iterative pruning with activation deviation for sclera biometrics. J King Saud Univ Comput Inf Sci. 2023;35(8):101630. doi:10.1016/j.jksuci.2023.101630.
36. Shah K. Eye-dataset. San Francisco, CA, USA: Kaggle; 2020. doi:10.34740/KAGGLE/DSV/1093317.
37. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. Adv Neural Inf Process Syst. 2012;25(6):1–9. doi:10.1145/3065386.
38. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2016 Jun 27–30; Las Vegas, NV, USA. p. 770–8. doi:10.48550/arXiv.1512.03385.
39. Rot P, Vitek M, Grm K, Emeršič Ž, Peer P, Štruc V. Deep sclera segmentation and recognition. In: Uhl A, Busch C, Marcel S, Veldhuis R, editors. Handbook of vascular biometrics. Cham, Switzerland: Springer International Publishing; 2019. doi:10.1007/978-3-030-27731-4_13.
40. Das S, De Ghosh I, Chattopadhyay A. An efficient deep sclera recognition framework with novel sclera segmentation, vessel extraction and gaze detection. Signal Process Image Commun. 2021;97(1):116349. doi:10.1016/j.image.2021.116349.
41. Riccio D, Brancati N, Frucci M, Gragnaniello D. An unsupervised approach for eye sclera segmentation. In: Mendoza M, Velastín S, editors. Progress in pattern recognition, image analysis, computer vision, and applications. Cham, Switzerland: Springer International Publishing; 2018. doi:10.1007/978-3-319-75193-1_66.
42. Zhang Y, Wang C, Li H, Sun X, Tian Q, Zhao G. OcularSeg: accurate and efficient multi-modal ocular segmentation in non-constrained scenarios. Electronics. 2024;13(10):1967. doi:10.3390/electronics13101967.