A comprehensive comparison analysis of scholarly investigations on deep neural network for Human Iris detection

MSc Research Project
Msc Data Analytics

Debmalya Deb
Student ID: x21242101

School of Computing
National College of Ireland

Supervisor: Aaloka Anant

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Debmalya Deb
Student ID:	x21242101
Programme:	Msc Data Analytics
Year:	2023
Module:	MSc Research Project
Supervisor:	Aaloka Anant
Submission Due Date:	14/12/2023
Project Title:	A comprehensive comparison analysis of scholarly investigations on deep neural network for Human Iris detection
Word Count:	6580
Page Count:	21

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	14th December 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

A comprehensive comparison analysis of scholarly investigations on deep neural network for Human Iris detection

Debmalya Deb
x21242101

Abstract

The research in Iris and eye gaze detection was initiated long ago and since then the research community has been trying to solve many critical case studies for example cheating on online examinations, digital biometrics, driver's eye fatigue detection during driving and many more which additionally has many business impacts. This paper is a comparison analysis of the performance and implementation of a high computing transfer learning model, Residual Network(ResNet) with a comparatively less compute unit consumption model, Convolution Neural Network(CNN). Images were captured from open-sourced videos and along with that, the real-time images were also been taken with the help of a built-in webcam. The taken images were annotated and stored in JSON documents and soon after they were augmented to get multiple features and scenarios. The augmented images were fed through the deep-learning models for evaluation and detection analysis. In the evaluation metrics, this study employed Mean Square Error(MSE) as validation and test loss, and some additional metrics like Root Mean Square Error(RMSE) and Mean Absolute Error(MAE). Finally, the project concluded that the high-computing ResNet model outperformed the less-computing CNN model regarding evaluation metrics and Iris detection analysis.

Keywords: Iris detection, ResNet mode, CNN model, Key point Annotations, Data Augmentation,

1 Introduction

The world of science is in the era of Digitisation and artificial Intelligence where the majority of the research community is moving towards computer vision, robotics, data science, machine learning, automatic detection and tracking etc. In this phase, the study and research on Human Iris Detection have a noticeable impact on curing many important factors concerning its technical and business significance.

The inventionDaugman (Jan. 2004) of the Iris recognition algorithm was invented by John G Daugman for the biometric identification of individuals rapidly and reliably with the help of the randomisation behavioural qualities of the iris of an eye and patient in the year 2013 having a United States patient no of 5,291,560[1]. Recent study saysRuiz-Beltrán

[1]https://www.invent.org/inductees/john-g-daugman#:~{}:text=John%20Daugman%20invented%20iris%20recognition,the%20iris%20of%20an%20eye

et al. (Sep2023) the iris recognition procedure for bio-metrics identification is based on some analysis and study on deep neural networks and mathematics which has many applications for instance phones, vaults, and computer unlocking systems. This study can have a noteworthy impact in the area of biometric identification via iris detection.

The adaptation of computer vision technology to determine the human Iris or eye gaze behaviours has been started for a long but the research community has noticed that this technological adaptation has increased significantly, especially during the Covid-19 and post-Covid-19 era due to many various aspects. To connect the statement the paper would like to give some examples of various case studies for example involvement of students in online classes, the detection of student's fraudulent activity during the online assessment, iris detection in a car if the driver sleeps during driving by installing some detection mechanism which would track and give some alarm to the driver and many more.

Taking another reference, in recent days, the whole education system has been partially moved towards online learning and the institution heads are discussing the most practical and useful ways for better student-teacher engagement. This research could put some additional importance towards those applications to improve these systems [2].

Besides the above the data scientists have taken this technology to a 3D eye detection and tracking model where the community could see its various technical and business significance to improve human life because eyes are the window of the mortal soul[3]. The researchers propose on "cooperative eye hypothesis" which says how evolvement of the human eye has increased the size and brightness of the white sclera for more useful transmission. The new application of human-machine relations using the machine is using several in-depth sensing cameras to catch eye gaze. Some of them are Attention tracking in retail, Eye detection for robotics, Automotive eye tracking, Automotive virtual co-pilot, Behavioral Research, Eye detection in advertisements and many more. The research discovers numerous motivational aspects behind this project which justifies this thesis.

This research mainly concentrates on and compares the two popular deep learning models on Iris detection. Although the research objective of this analysis is more technical and research-oriented, nevertheless this paper would like to bring out some of its business values as well. The study was broken down into several parts for example data collection, image collection, data/image annotation, data labelling, augmentation, data partitioning and finally model building. The paper took some open-sourced videos from a well-known platform[4]. From that, the videos were selected based on the subject's eye movements and considering the other multiple scenarios as well. Soon after, the images were taken using Python OpenCV, annotated(captured the annotation into separate JSON files) and labelled with Python LabelMe. This study also took care of the image's uniqueness by giving them an individual Universally Unique IDentifier(UUID). Finally, the images were fed to the deep learning models and compared their performance.

1.1 Research Question

To what extent can a high computational transfer learning model, Residual Network(ResNet) outperform a less computational model, Convolution Neural Network(CNN) for Human Iris Detection and how could it be employed as a business model to translate the computer

[2]`https://edtechmagazine.com/higher/article/2020/12/3-ways-increase-student-engagement-online-lear`

[3]`https://eyeware.tech/blog/top-7-use-cases-for-3d-eye-tracking/`

[4]`https://www.kaggle.com/code/mmmarchetti/deep-fake-chalenge/input`

vision concerns?

This thesis paper researched how well a Residual Network(ResNet) model outperformed a simple Convolution Neural Network(CNN) model in terms of evaluation metrics and visual interpretation of iris detection. But what was the need for the legacy CNN model in the era of high-performance transfer learning models for computer vision, objection and image detection? This case study found the ResNet model had taken a huge computational resource for model training, testing and deployment compared to the legacy CNN model even though the ResNet model surpassed the CNN model with high resources. Still, a CNN could achieve a good score in terms of performance.

1.2 Report Structure

In the upcoming sections, this paper is divided into the following sections. Section 2 describes the literature review of the related works concerning computer vision and Iris Detection which highly inspired this study, section 3 contains research methodology like what machine learning approach was used for this experiment, section 4 contains the design specification for an instance of research flow diagram along with the ResNet and CNN model architecture, section 5 consists of implementation like the experimental setup following by dataset preparation, data pre-processing, modelling building and deployment, section 6 is a model evaluation with MSE as a validation and test loss and some additional metrics like RMSE and MAE, section 7 is discussion and finally section 8 is the conclusion and future work and lastly References.

2 Related Work

Over decades, the research community has been employed to improve iris and eye gaze detection and tracking to impact various fields, for example, the online education system, digital bio-metrics, cheating and fraudulent detection in online assessment, eye fatigue detection in automotive etc. The below paragraphs will be based on the research papers which have motivated this study and their contribution to iris and eye gaze detection from technical and business perspectives.

The ResNet model was presented by He et al. (2016) in their report "Deep Residual Learning for Image Recognition". It was one of the best image detection transfer learning and won the competition(ILSVRC 2015 and COCO 2015) by beating other high computational models like Image-Net detection, Image-Net localisation, COCO detection, COCO localisation etc.

ResNet-V2 and CNN are widely used deep neural network models for computer vision, image detection, deep fake detection, iris detection etc. Guefrachi1 et al. (2023) extracted video frames using DLIB and OpenCV and converted videos into frames of images. This study employed VGG16 in addition to 13 convolutional models for deep fake image detection. It is to be known that the ResNet model is inspired by VGG16 only. The study achieved model accuracy of around 94% with excellent other metrics like precision, recall etc in terms of deep fake video detection.

Another research paperGuefrechi et al. (2022) used openCV and DLIB libraries for image feature extraction, converted the images from videos, and then fed them to a fine-tuning high computational transfer learning model, InceptionResnet-V2 alongside a fully connected neural network, CNN. The inceptionV2 layer was designed by layers starting from SoftMax, Dropout, Average pooling, 5x inception Resnet-C, Reduction B and many more. At the end of the layer, it is connected to its input layer. ResNet and Inception were the heart of all networks utilised in image recognition and detection.

In terms of iris classification, a studyArora et al. (2023) performed an analysis with weighted average ensemble machine-learning techniques to predict the outcomes with more perfectionDang et al. (2022) by deploying a collaboration of various transfer learning models one of which was resnetV2 along with the CNN model. This study was performed with 3 different datasets and three times the ensemble model was employed, the accuracy of each model was calculated and the weighted average was figured out. It was observed that while the other transfer learning models including the Resnet performed better, the CNN model could not even achieve the accuracy of 10%. According to the paper in the 3 datasets, the CNN model performance was 6.40%, 6.29% and 4.07% respectively. Hence, this study supported this research paper and influenced to exploration of how the iris detection would have been better with a less performance CNN model.

Few authorsM et al. (2021) used transfer learning models like ResNet50, and InceptionV2 along with that traditional CNN model in the detection and recognition of COVID-19 in Chest CT scan images. The infected and non-infected X-ray scans were fed through these transfer learning models for classification. In the paper Guefrechi et al. (2021) on demand of a high-scaled dataset, data augmentation was performed on COVID-19 X-ray images before passing those to deep neural networks.

Shanmugapriya et al. (Mar 2023) found recent integration improvements in the biometric recognition system which is more user-friendly and secure compared to the traditional method and many applications use this boost technologyA. Kumar. (2023). The research was divided into Iris image generation, data pre-processing, template generation and classification. 756 iris images were collected from 106 eyes, given by the "Chinese Academy of Sciences Institute of Automation(CASIA)" [5]. Haar cascaded classifier and OpenCV were used for data preprocessing which further helped with template creation and located the pupil position with proper coordinates. The research assessed the Hough Circle transformation while searching the circular pupil in the eye from the image. This detection study was performed on the comparison analysis between Logistic Regression(LR) and Convolutional Neural Network(CNN) and it was observed that the CNN model outperformed the LR model without maxpooling and 100% with maxpooling. This research seemed to be slightly overfitting and the reason could be less amount of dataset. However, this study motivated this research project on model building part as the CNN model was used.

Qiu et al. (2020) focused on continuous and comprehensive tracking systems using eye gaze tracking and detection on robotic cataract surgery which is one of the major syndromes in people aged more than 40, especially in the USCongdon et al. (2004) to provide better accuracy surgical atmosphere and a warranted business model. The iris tracking could provide a better iris coordinate location before the surgery starts, during the incision period when the patient's cornea has been injected. The study began employing the iris feature extraction network in the medical data collected by the researchers. It was segregated between low-level and high-level features based on the different iris layers. The study was composed using a 7 intense convolution layer-based fully connected CNN model. The CNN layer produced a linear map and a ReLU activation function was used to generate the map. This research was performed by giving a specific concentration of bottleneck layers which was very significant in medical iris detection. In Ophthalmic robotic surgery, the CNN model played a desired role with around 87% accuracy which

[5] `http://biometrics.idealtest.org/#/`

pretty much influenced this research study but this paper wants to broaden this to the next level to the ResNet model as new transfer learning models are specially built for computer vision on tracking and detection.

Panwar and Pooja (2022)discussed and reviewed the evolution of biometric systems and how they had been changing from traditional finger prints, and voice recognition to computer vision and iris or eye gaze detection to get rid of fraudulent activities and impostersKhade and Thepade (2019). In the below Figure 1 a wonderful evolution of the iris biometric authentication had been noticed starting from 1985, how iris detection in biometrics took place and by 2021 how machine learning and deep learning algorithms gradually have taken into consideration by the research communitiesKhade et al. (2021).

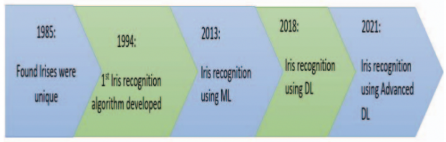

Figure 1: Evolution of growth of Iris bio-metric detection(Khade et al.; 2021)

In this study, the researchers used mobile-captured images and open-sourced internet images for the dataset and then the image data was preprocessed by pupil localisation, eye field detection, eyelid localisation, eyelash localisation, noise filtering etc. To classify and recognise the iris they reviewed multiple machine learning and deep learning algorithms just as Support Vector Machine(SVM), Direct Linear Discriminant Analysis, Decision tree, Random forest, logistic regression, transfer learning models, CNN models and many more. However, when it comes to large datasets the analysts suggested going with deep learning over machine learning.

The above-discussed research papers positively influenced this study and permitted to a certain extent to explore some unique approaches. The the above papers this study discovered how the transfer learning models perform well in computer vision detections and trackings whereas the CNN models could not achieve good results compared to the pre-trained models. On the other hand, the dataset creation step, capturing images from videos and the author's image with a webcam followed by using labelME for annotation and putting all the information into a JSON document etc all were highlighted in the above research papers. To all the papers mentioned above this study tried to make a better performance CNN model by customised its deep neuron layers which could give a tough fight in terms of model performance for iris detection.

3 Research Methodology

The thesis methodology section wraps the entire project steps starting from the experimental setup, and data gathering until the knowledge is achieved out of it. Therefore this part of the report consists of experimental setup, data gathering, data selection, data pre-processing, data transformation, data mining and finally evaluation. This means this

[6]KDD Miro Dashboard `https://rb.gy/bdjtvl`

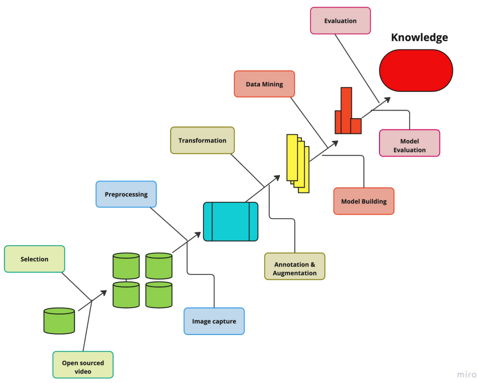

Figure 2: KDD approach for Iris Detection[6]

study strictly obeys the comprehensive strategy of discovering learning facts in data science which is Knowledge Discovery in Database in short KDD approach[7].KDD approach underlines high-level data analysis based on its 5 different classes for example Selection, Preprocessing, Transformation, Data Mining and interpretation or Evaluation as per Figure 2. The KDD architectural diagram refers to how this particular project utilised it.

4 Design Specification

Figure 3 represents the architectural diagram or flow diagram of the entire thesis paper. This study was divided into 5 different stages Data collection, Data pre-processing, Modelling, Evaluation and Deployment.

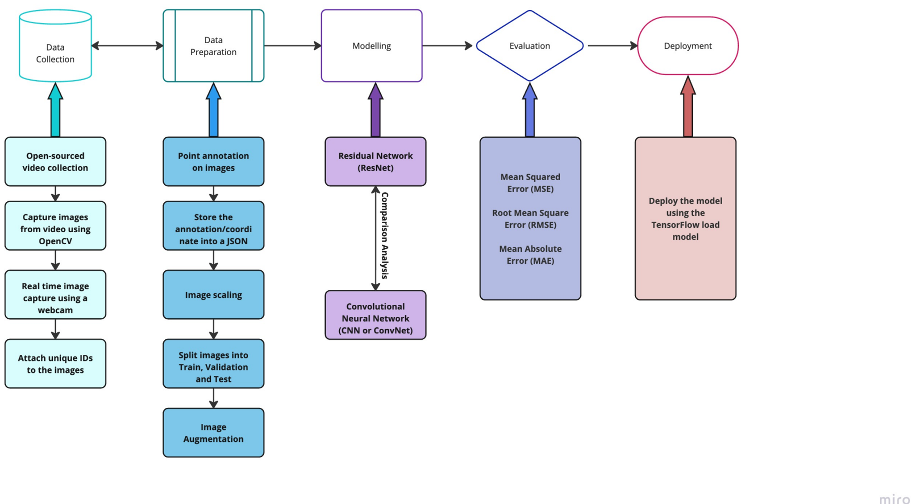

Figure 3: Iris detection model flow diagram [8]

[7]`https://www2.cs.uregina.ca/~dbd/cs831/notes/kdd/1_kdd.html`
[8]Miro Dashboard of Flow diagram `https://rb.gy/bdjtvl`

To prove that, this empirical analysis follows every category of the KDD approach, at the beginning of the study some open-sourced(Kaggle[9]) short videos were selected based on the different iris and eye gaze movement scenarios as mentioned above which comes under data selection. The snaps were being captured from every video and a unique ID was being attached as well with every snap, this comes under data preprocessing. Also, to strengthen the dataset with the help of an inbuilt webcam, the author's image was taken, keeping all different scenarios concerning the iris and eye gaze movements. While capturing the snaps the study made sure that all the images were taken in 450 x 450 frames to balance proportionate the data frame throughout the project. With the help of labelME, all images were labelled and iris coordinated and locations were stored in a JSON file with a unique identifier. All the processed images were then split into train, test and validation datasets with their annotations and then they were augmented in a way of rotating, flipping, twisting, mirroring etc. And these all come under data transformation. Then the augmented images were fed to a traditional CNN model and high-configuration ResNet model for research analysis these steps belong to data mining. In the evaluation stage both of the models were evaluated using the validation and test datasets and the iris detection was checked using the test dataset. In the evaluation metrics, the study took Mean Squared Error(MSE), Root Mean Squared Error(RMSE) and Mean Absolute Error(MAE). In the final stage, the model has been deployed in Hierarchical Data Format version 5 (HDF5) or .h5 format for further investigation

To extend this study and go more in-depth to give this study a justifiable business model the researcher tried for real-time detection by deploying the model into software. Still, unfortunately, it was not up to the mark as desired that's why the study was mainly focused on the research comparative analysis of two deep learning approaches and kept the real-time tracking in its future scope.

4.1 Neural Network Model Architecture

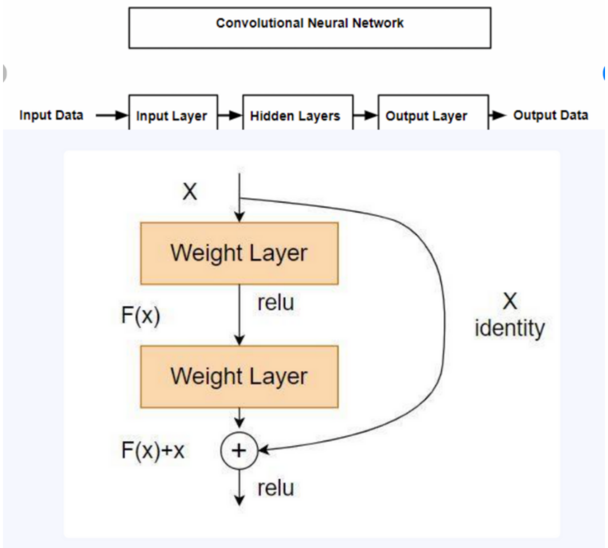

Figure 4: ResNet and CNN Block Diagram

[9] `https://www.kaggle.com/code/mmmarchetti/deep-fake-chalenge/input`

When we keep on increasing the dense deep learning layers on the models the chances of Vanishing Gradient Decent(when the gradient points reach zero) and Exploding Gradient Decent(gradient becomes too high) problems also perpetually increase which leads to bad accuracy of the model for these reasons only dropout layer is needed to drop the dense layers from the network layers and this phenomenon happens mainly with CNN network. Figure 4 describes the simple connection logic of a Resnet and a CNN model. For ResNet model shows how two blocks connect to each other in a ResNet network and how they skip, this phenomenon is called ”skip connection” and this is because to avoids the above-mentioned phenomenon. Without this skip the input passes through multiple layers and weights which causes the bias in the model [10]. In terms of the CNN model, the model needs dropout logic to skip the deep neurons to avoid all those phenomena. This way the ResNet works well over the CNN model even with intensive deep network layers without any Gradient descent and dead neuron problems.

The ResNet model architecture is inspired by VGG-19 and the plain 34 layers model but then it connects and skips those layers which differentiates the particular model from others. Figure 5 is a visual representation of ResNet.

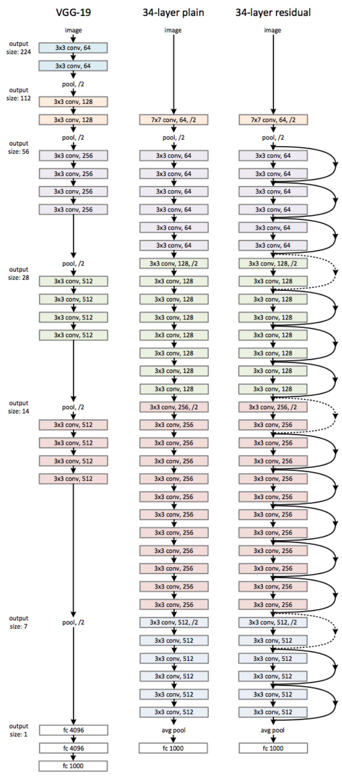

Figure 5: ResNet and simple CNN network Architecture Diagram

Keras supports many ResNet versions just as ResNet50, ResNetV2, ResNet101, ResNet101V2, ResNet152, and ResNet151V2 and from this the study implemented ResNet152V2. These two or three digits suggest the number of layers in the ResNet network layer.

[10]https://www.codingninjas.com/studio/library/resnet-architecture

This study chose ResNet-V2 over ResNet-V1 in the iris detection. As per Figure 6, the ResNet-V2 emphasises the Batch Normalisation and ReLu activation function to the input function. This happens before the input gets multiplied by the weight matrix.

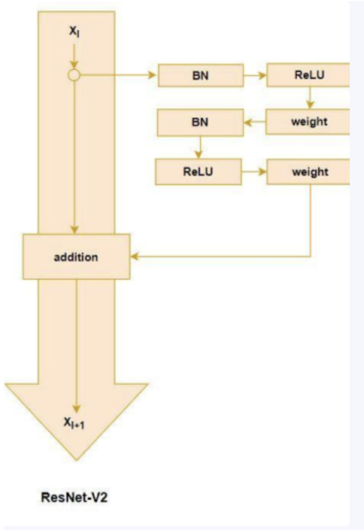

Figure 6: ResNet-V2 Block Diagram

5 Implementation

5.1 Experimental Setup

There was a minor experimentation setup to be mentioned while capturing the author's images to include in the dataset. The study used the built-in laptop webcam to take images using Python OpenCV. This webcam was also used for real-time monitoring checking even though that was not up to the mark as expected.

5.2 Data Collection

In the Data Collection stage of the research study as mentioned above some open-sourced short videos were collected suitable for the iris detection. Along with that to strengthen the dataset and to explore a bit more with that author's images were also being captured using Python OpenCV.

5.2.1 Images from the open-sourced video

In the first phase of data collection, around 20 videos were selected which were more appropriate for Iris Detection. Regarding the snaps, a Python code was written with taking help of OpenCV which helped to check using a VideoObject cap(enabled to check whether the video file is available or not if not then print an error statement) and then to frame the images and store them in a desired file path. Each frame was shaped into a specific measurement(image_width=450 image_height=450) and a unique identifier name was generated using the Python UUID module. The number of images had been chosen to 5 per video file in some cases 10 depending upon the variation of the eye gaze movements of the video content. The resized image frames were saved into a specific file path in JPEG format. All the video capture objects and OpenCV windows were closed as soon as the image collection was being done.

5.2.2 Images from the inbuilt webcam

In the second phase of image collection, the real-time images were captured using OpenCV and the device's inbuilt webcam. Here as well a video capture object 'cap' had been used and the difference from the above code was here it was to check whether the webcam window was opened or shut. A circle of loop was initialised in a specific number of times making sure a slit pause of 0.5 seconds between each frame for the real-time image capture. All the image sizes were taken as similar to the above-mentioned height and width to proportionate the size of the images in the dataset and stored in the same place where the other images were being stored. The image-capturing test was executed considerable times with different variations of iris actions and eye gaze movements and the trials were conducted by setting 15 to 20 images per loop and trial.

5.3 Data Pre-Processing

In the next phase before modelling the most important stage was divided into two parts, Data Annotation and Labelling and Data Augmentation.

5.3.1 Data Annotation and Labelling

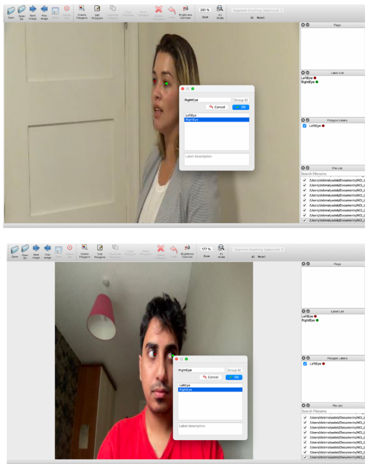

Figure 7: Point annotation using LabelME

The LabelMe[11] is an open-sourced data annotation tool for manual image polygonal annotation for object detection, segmentation and classification etc. LabelMe supports to the creation of various kinds of shape annotation for example points, circles, rectangles, line strips etc. This study employs the point annotation keeping the left eye annotation class name as "LeftEye" and "Green" in colour and the right eye annotation class name as "RightEye" and "Red" in colour. The annotations were saved directly into the respected JSON files straight from the app. With the labelME feature of Pascal Vol the whole

[11] http://labelme.csail.mit.edu/

annotation was converted into a Python script and stored in the respected repository files[12]. Figure 7 represents the image labelling of the left eye and right eye for the snaps taken from the videos and real-time as well. The labelMe window opens in a local host from a single line code in Python.

5.3.2 Data Augmentation and splitting

TensorFlow and OpenCV libraries were used to gather the dataset images with their corresponding annotations for this augmentation. Using the TensorFlow API the images were gathered up from the local directory and read. With a function, the images were read and resized into 450 x 450 format along with that it normalised the images by the proper pixel value and into 4 sections for proper graphical representation.

In the next stage, the images were manually loaded into the train, test and validation folders. In this regard, 75% of the images were sent to the training folder and 15% each to the test and validation folders respectively.

After moving the images, one more crucial part was to transmit the respected image's annotations(JSON files) to their individual train, test and validation folders. An iterative loop was written to repeat each corresponding folder of train, test and validation, then the script located the JSON annotations and finally with the Python "OS" library and with "os.replace"[13] it relocated the annotations to the respected directory by constructing new file path.

Now eventually, this thesis paper employed the Python Albumentaion[14] to conduct the data augmentation on images and with their affiliated annotations for the Iris detection. The script started with reading and loading the images and their corresponding key point JSON annotations from the respected directory just for train, testing and val.

In this regard just to define the keypoint annotations and labelling with an example with the help of Figure 8. It shows the "LeftEye" and "RightEye" classes with their coordinate points and "point" as their shape type.

```
[{'label': 'LeftEye',
  'points': [[196.4689265536723, 139.40677966101694]],
  'group_id': None,
  'description': '',
  'shape_type': 'point',
  'flags': {}},
 {'label': 'RightEye',
  'points': [[218.50282485875704, 133.19209039548022]],
  'group_id': None,
  'description': '',
  'shape_type': 'point',
  'flags': {}}]
```

Figure 8: The Iris Annotaions

The script then initialised an augmentation pipeline using the Albumentaion library by including procedures like vertical and horizontal flip, random cropping, adjusting the noise, brightness and contrast altering, RGB shift, and Gamma correction. The code iterated through every partitioned folder and their respective JSON annotations, loaded into the respected augmentation function by extracting the classes of the annotation

[12]https://datagen.tech/guides/image-annotation/labelme/#
[13]https://docs.python.org/3/library/os.html#os.replace
[14]https://pypi.org/project/albumentations/

and finally put those things together into the augmentation pipeline on an iterative loop. The resulting augmented images and their respective annotation preserved all the corresponding knowledge, especially the coordinate points that were successfully relocated with the image augmentation keeping the size as constant as mentioned above. The was code written in such a way so that it could handle all the exceptions during the augmentation procedure.

At the end of the augmentation stage, this study managed to pull out 8280 training images 1800 test images and 1800 validation images along with their annotations for model building.

5.4 Merging the Annotations with the images

The augmented images with their annotations were uploaded to Google Drive for the data mounting in Colab[15]. TensorFlow, OpenCV, Numpy, OS, Matplotlib etc were imported to work the neural network training, detection, working with annotations with the images and plotting etc. The images were decoded, normalised and resized after loading them into a function. The same steps were performed repeatedly for train, test and validation datasets. The images were resized to 250 x 250 in size which was previously 450 x 450. This process was applied with JSON annotations as well by loading into a function and specifying the "LeftEye", and "RightEye" coordinates and key points. The Python script shuffled and batched the images into the size of 16 proportionate with their labels and the dataset was zipped. Figure 9 shows the combined images with the labelled annotation and now the dataset is ready for train, validation and test.

The above steps were completely common for both of the below models.

Figure 9: The training dataset with the annotations

[15]`https://drive.google.com/drive/folders/1CCOM78mne2GxjN-h8xrW42R3vqSEtgjq?usp=drive_link`

5.5 Model Building

The comparison analysis first started its research from a transfer learning model, ResNet which employed a huge computational backup and time. Then the study wanted to explore a legacy model which was CNN to diminish the computational resource.

5.5.1 ResNet152V2 Model

This study employed ResNet152V2 as described above. For the script, this research imported essential Python TensorFlow libraries like Sequential from Tensorflow models, Input, Conv2D, Reshape, Dropout from TensorFlow layers and ResNet152V2 from TensorFlow application.

The model has sequential CNN layers with Keras API. The model architecture and input layer specify the image shape of (250,250,3). To investigate the complex features of the images the succeeding layer was a pre-trained transfer learning model based on ImageNet to deal with the ultimate complex features of iris detection, ResNet152V2. The following two CNN layers had 512 filters along with 3x3 kernel size, followed by ResNet152V2. Each of the neural network layers had having ReLU activation function and the 'same' padding. Same padding is a technique for processing input data it further helps to add additional rows and column feature pixels around the edges of the input data which is why the size of the output data exactly matches with the size of the input data[16]. The next convolutional layers had 256 filters along with 3x3 kernel size and stride value 2 used for downsampling tracked by one more CNN layer of consisting 256 filters and 2x2 kernel size and downsampling stride value of 2. For model regularisation, a dropout layer of a dropout rate of 0.05 was kept. The final layer had 4 filters with 2x2 kernel value and stride 2 along with that the output was flattened into 1D and resized with 4 elements. Figure 10 illustrates the architecture of the model summary.

Model: "sequential"

Layer (type)	Output Shape	Param #
resnet152v2 (Functional)	(None, 8, 8, 2048)	58331648
conv2d (Conv2D)	(None, 8, 8, 512)	9437696
conv2d_1 (Conv2D)	(None, 8, 8, 512)	2359808
conv2d_2 (Conv2D)	(None, 4, 4, 256)	1179904
conv2d_3 (Conv2D)	(None, 2, 2, 256)	262400
dropout (Dropout)	(None, 2, 2, 256)	0
conv2d_4 (Conv2D)	(None, 1, 1, 4)	4100
reshape (Reshape)	(None, 4)	0

Total params: 71575556 (273.04 MB)
Trainable params: 71431812 (272.49 MB)
Non-trainable params: 143744 (561.50 KB)

Figure 10: ResNet152V2 model summary

5.5.2 CNN Model

The study conducted one more research with legacy CNN model which was less in computation energy and power. The first layer consisted of 250 x 250 images having 3 different colour channels Red Green Blue(RGB). The following 2 layers were having 64 filters in

[16] https://www.geeksforgeeks.org/what-is-the-difference-between-same-and-valid-padding-in-tf-nn-max-pool-of-tensorflow/

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 250, 250, 64)	1792
conv2d_1 (Conv2D)	(None, 250, 250, 64)	36928
max_pooling2d (MaxPooling2D)	(None, 125, 125, 64)	0
conv2d_2 (Conv2D)	(None, 125, 125, 128)	73856
conv2d_3 (Conv2D)	(None, 125, 125, 128)	147584
max_pooling2d_1 (MaxPooling2D)	(None, 62, 62, 128)	0
conv2d_4 (Conv2D)	(None, 62, 62, 256)	295168
conv2d_5 (Conv2D)	(None, 62, 62, 256)	590080
max_pooling2d_2 (MaxPooling2D)	(None, 31, 31, 256)	0
flatten (Flatten)	(None, 246016)	0
dense (Dense)	(None, 512)	125960704
dropout (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 4)	2052
reshape (Reshape)	(None, 4)	0

Total params: 127108164 (484.88 MB)
Trainable params: 127108164 (484.88 MB)
Non-trainable params: 0 (0.00 Byte)

Figure 11: CNN model summary

each with a ReLU activation function in each layer and max-pooling for downsampling. This pattern was followed the same but kept the filter hierarchy as 64, 128, and 256. The CNN model the flattened, executing one last dense layer with 512 filters, ReLU activation and dropout regularisation technique. Finally, the output was reshaped into 4 elements to match the desired result in the final element. Figure 11 demonstrates the model as explained above.

5.6 Deployment

Both of the models were deployed in an H5 format which was a Hierarchical Data Format(HDF) to maintain the code sequence[17]. To do that "load model" from the keras.model was imported and the deployed model were stored in a directory for future purpose.

6 Evaluation

The main purpose of this part was to evaluate how well a high computational transfer learning model works over a traditional deep learning model(CNN)? In this matter, this study assessed 3 parameters as its evaluation metrics to validate the fact, those were Mean Square Error(MSE), Root Mean Square Error(RMSE) and Mean Absolute Error(MAE).

This research now breaks into two case studies in terms of their evaluation.

6.1 ResNet152V2 Model / Case Study 1

6.1.1 Validation Dataset

In the first stage of evaluation, the study checked the model performance with its validation dataset. The script used Adam as an optimiser, keeping the learning late as 0.001 and MSE as its loss function. Not only MSE the paper additionally added RMSE and MAE to put weight towards the model assessment. The model was initially set for 200 epochs with an early stopping as a callback, this callback would check that if there was

[17] `https://docs.fileformat.com/misc/h5/`

no significant improvement in the loss function for consecutive 3 epochs then the training would stop and it would pull out the best validation loss with best epoch value.

The model got its best validation loss of 0.00018 with corresponding RMSE and MAE 0.018 and 0.014 respectively at the 9th epoch, referred to in Figure 12.

```
Best Validation Loss: 0.00017881562234833837
Best Epoch: 9
Corresponding RMSE: 0.017991825938224792
Corresponding MAE: 0.014049476012587547
```

Figure 12: ResNet-V2 model best validation loss

Figure 13 shows how the MSE started from the first epoch and how initially it was around 0.4 at the first epoch for the training data and then the validation loss was recorded pretty low. The other graph shows almost the same when for its RMSE value that in the beginning epoch for training data it was high compared to the validation dataset. But for both of the graphs, a significant inclination had been observed and finally, the both training and validation metrics matched at the best value epoch.

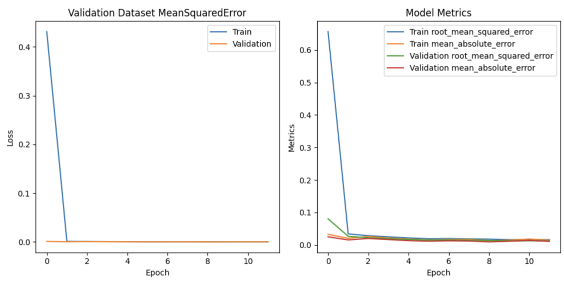

Figure 13: ResNet152-V2 model evaluation graph for validation dataset

6.2 Test dataset

In the next, the same above mentioned code had been gone through using the test data and after the 11th epoch, the loss suddenly got a sharp spike and the model build stopped because of the fruitful behaviour of early stopping.

As per Figure 14, the research found its best value test loss of 0.00014 with their corresponding RMSE and MAE values of 0.013 and 0.0101 respectively in the 9th epoch.

```
Best Test Loss: 0.00014739563630428165
Best Epoch: 9
Corresponding RMSE: 0.012985926121473312
Corresponding MAE: 0.010134689509868622
```

Figure 14: ResNet-V2 model best test loss

Figure 15 validates the above facts that from the initial epoch all the metrics were very low and were giving wonderful results and suddenly it changed its path and intended to rise up but then it stopped because the script had its callback set.

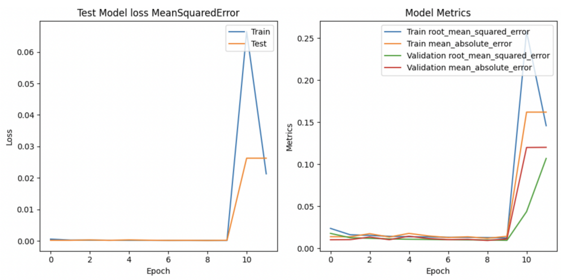

Figure 15: ResNet-V2 model evaluation graph for test dataset

6.2.1 Performance testing on Iris Detection

The model was iterating through the test data using an iterative function and a test sample was taken in a batch of 4 images together. The code utilised matplotlib library for the visualisation. For each detection, the code employed the key point annotation that had been used during the labelling process as shown in Figure 16. From the visual representation and judging the multiple scenarios, the study could see that most of the detections were nearly perfect but in very few scenarios the detection was slightly misjudged and due to that the key points would shift from the and most possible reason would be the eye gaze positions were pointing at the edges.

Figure 16: ResNet-V2 model detection performance

6.3 CNN Model / Case Study 2

This study found that when the same code was triggered for the CNN model less computation time was being taken in model building and testing. The research trained the

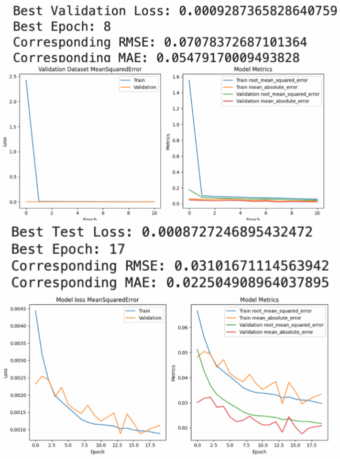

Figure 17: CNN model Evaluation Metrics for validation and testing

model by setting the epochs value to 200 with validation and test datasets same as like ResNet model mentioned above and got the performance metrics.

Figure 17 describes that the best epoch with the validation dataset was 8 while with the test it was 17 and the corresponding loss or MSE values were 0.00087 and 0.00092 respectively with the additional metrics which was quite a good result. But while testing the model a certain fluctuation of the evaluation metrics was observed even after a few experiments and due to early stopping it stopped in between epochs 17 to 20 as per Figure 17.

6.3.1 Iris detection Performance testing

The trained model was now looped through the test data with the help of an iterative function with a batch size of 4 just as like ResNet model mentioned above. The same matplotlib library and key point annotations were utilised for the visualisation. Figure 18, illustrates the CNN model's iris detection performance and the research observed that it detected the iris coordinates and the eye gaze movements with a slight shift and on average with every image more or less the same scenario was noticed.

6.4 Comparison Analysis

It was a very close call to choose the best-fit model and justify the research question as both of the models, concerning evaluation metrics were up to the mark. Even though with a fine close call the ResNet152-V2 model outperformed the CNN model. Finally altogether to compare and analyse both of the models it was noticeable that the ResNet model's MSE, RMSE and MAE values were 0.0001473, 0.0129859 and 0.0101346 respectively whereas for the CNN model, the metrics were 0.0008727, 0.0310116 and 0.0225049 respectively, shown in Figure 19. Hence all the above findings made it clear that the ResNet model was slightly better in terms of Iris Detection performance.

Figure 18: CNN model on Iris Detection

	ResNet152-V2		CNN
MSE	0.0001473		0.0008727
RMSE	0.0129859		0.0310116
MAE	0.0101346		0.0225049

Figure 19: Evaluation Metrics Comparison Table

7 Discussion

This research always aimed to justify its research question and tried to put a hold on the fact that the high computational ResNet model worked better than the traditional CNN model. The study used 150 layer-based V2 ResNet model along with that the study connected some Convolutional layers as well to straighten the model performance. On the other hand, the research wanted to analyse how a less computational model performs on its detection and for that traditional CNN model which had not high compute units to train was employed. The report was divided into 2 case studies one with the ResNet model and another with the CNN model and for both models, the researcher had to perform multiple trials to get the accurate and final desired model. For the CNN the experiment customised its layer for better modelling and tried to achieve a good evaluation score even though it was a less compute unit model.

The study noticed that based on evaluation metrics CNN model gave a tough fight to the intensive deep layer-based transfer learning model, ResNet152V2. During the comparison analysis of the evaluation metrics, this study found that the metrics score was not deep-margined. The ResNet model was crowned as a best-fit model in terms of evaluation metrics with a very marginal difference between MSE, RMSE and MAE.

Now when the paper looked at the visual differences in its iris detection the differences were noticeable. For the ResNet model, most of the detection was perfect, the key points were on top of the eye's pupil and when the eye gaze position changed the key point was also tracked and moved along with that. But along with that some of the detections were not accurate especially when the eye's pupils were at the edges which was majorly because this study did not find the images where the participants were looking at the extreme edges else few. But overall the ResNet model performed well for Human Iris Detection.

When it was time for the CNN model's iris detection even though the evaluation metrics score were exceptionally well, the detections were not up to mark as compared to the ResNet. For most of the images, the key point annotations were slightly out of the surface of the eye. But it needs to be mentioned that for some of the images, the CNN model also detected well.

8 Conclusion and Future Work

A transfer learning model or a pre-trained model is always high in terms of computing units because these models have intensive deep neural networks like how ResNet152V2 model has. This study thinks that if a less computing model like CNN could have been built by adding the CNN layers in a perfect way many more computer vision tasks like Iris detection could have been achieved.

This experiment had to handle many Python libraries one of which was OpenCV for image capturing and real-time image capturing to do that opening the built-in webcam was a challenge there was a big challenge to train the high computing deep learning models and deploying them. Taking into consideration every single challenge and trial the whole experiment was performed in 3 different Python Integrated Development Environments(IDE). The whole dataset collection, annotation and augmentation part was done in Jupyter Notebook, the model building was accomplished in Google Colab Pro and the Deployment and the real-time tracking trial were conducted in Pycharm.

The study was first started by taking only real-time images but that would be very specific to make the model a generative one the research then included other open-sourced images as well. The study faced some challenges to identify the different eye gaze movements and to do that this paper explored many small videos and after a long research selected some appropriate videos with several iris movement scenarios. This study also agreed that the dataset might be short because of the scarcity of multiple scenarios on eye gaze positions.

The study found that many key point annotations and coordinate data storing were a little bit time-consuming but it was also on the other hand more accurate. The research tried to establish some approaches as well but finally, it decided to stick to manual annotation only.

Image labelling was one of the challenges which had been flagged up before model building so it was rectified once found.

This experiment's future score would be real-time iris tracking to make it a business model. To explore the future scope, this study tried to perform a small test. The research study loaded the deployment model and with the webcam, the author tried an experiment on real-time tracking but unfortunately, it was not up to the mark. Figure 20 shows the experimental trial on real-time tracking.

Overall, the main motivation of this research project was to compare and analyse the performance of a high computational model and a low computational model for human iris detection to motivate some business applications. After the whole experiment, this study could comment that the high computational ResNet152-V2 model performed better than the plain CNN layer network model. Although, the CNN model scored nicely in terms of evaluation metrics results the ResNet model surpassed it but concerning the iris

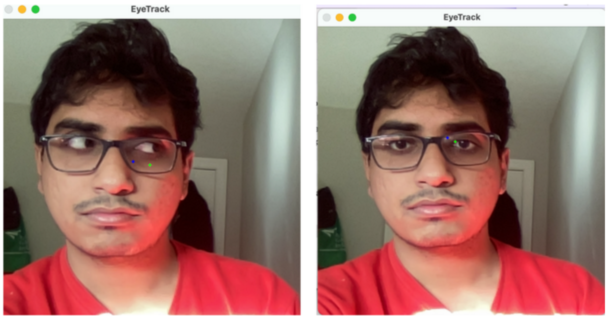

Figure 20: Real-time tracking with ResNet model

detention while testing the the ResNet model performed way better than the CNN.

Finally, this analysis was initiated to give its fruitful impact towards the research community of iris detection applications as mentioned in the introduction. This study found that a transfer learning pre-trained deep learning model could make a significant improvement towards iris detection business applications. As mentioned in the future scope, the next-generation iris detection model will be able to track real-time iris tracking and a complete machine learning tracking product will come out with real business prototypes and values.

References

A. Kumar., S. Jain, M. K. (2023). Face and gait biometrics authentication system based on simplified deep neural networks, *Int. J. Inf. Technol* **15**: 1005–1014.

Arora, A., Gupta, A., Jindal, B. and Gaurish Gupta, a. (2023). ”smart iris classification using weighted average ensemble learning”, *2023 International Conference on Disruptive Technologies (ICDT), Greater Noida, India,* (21): 124–130.

Congdon, N., Vingerling, J. R., Klein, B. E., West, S., Friedman, D. S., Kempen, J., O’Colmain, B., Wu, S. Y. and Taylor, H. R. (2004). ‘prevalence of cataract and pseudophakia/aphakia among adults in the united states, *Arch. Ophthalmol* **122**(4): 487–494.

Dang, T., T. T. Nguyen and, J. M. and A. W. C. Liew, a. (2022). ”ensemble learning based on classifier prediction confidence and comprehensive learning particle swarm optimisation for medical image segmentation”, *2022 IEEE Symposium Series on Computational Intelligence (SSCI), Singapore, Singapore,* (21): 269–276.

Daugman, J. G. (Jan. 2004). How iris recognition works, *EEE Transactions on Circuits and Systems for Video Technology* **14**(1): 21–30.

Guefrachi1, S., Jabra, M. B., Alsharabi, N. A., Othman, M. T. B., Alharabi, Y. O., Alkholidi, A. and Habib Hammam, a. (2023). Deep learning based deepfake video detection, *2023 International Conference on Smart Computing and Application (ICSCA), Hail, Saudi Arabia, 2023* pp. 1–8.

Guefrechi, S., Jabra, M. B. and H. Hamam, a. (2022). Deepfake video detection using inceptionresnetv2, *2022 6th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP), Sfax, Tunisia, 2022* pp. 1–6.

Guefrechi, Sarra, M. B. J., Ammar, A., Koubaa, A. and Hamam., H. (2021). ”deep learning based detection of covid-19 from chest x-ray images, *Multimedia Tools and Applications 80* (21): 31803–31820.

He, K., Zhang, X., Ren, S. and Sun, J. (2016). Deep residual learning for image recognition, *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA,* pp. 770–778.

Khade, S., Ahirrao, S., Phansalkar, S., Kotecha, K., Gite, S. and Thepade, S. (2021). Iris liveness detection for biometric authentication: A systematic literature review and future directions, *Inventions 2021* **6(4)**: 65.

Khade, S. and Thepade, S. (2019). Fingerprint liveness detection with machine learning classifiers using feature level fusion of spatial and transform. domain features,”, *In Proceedings of the 5th International Conference On Computing, Communication, Control, And Automation (ICCUBEA), Pune, India* **2-12**: 1–6.

M, B. J., Koubaa, A., Benjdira, B., Ammar, A. and Hamam, H. (2021). Covid-19 diagnosis in chest x-rays using deep learning and majority voting, *Applied Sciences 11* (6).

Panwar, V. and Pooja (2022). A review on iris recognition system using machine and deep learning, *2022 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), Greater Noida, India,* pp. 857–866.

Qiu, H., Li, Z., Yang, Y., Xin, C. and Bian, G. (2020). Real-time iris tracking using deep regression networks for robotic ophthalmic surgery, *IEEE Access Access, IEEE. 8:50648-50658 2020* pp. 425–429.

Ruiz-Beltrán, C. A., Romero-Garcés, A., González-García, M., Marfil, R. and Bandera, A. (Sep2023). Real-time embedded eye image defocus estimation for iris biometrics., *1Departamento Tecnologia Electronica, ETSI Telecomunicacion, University of Málaga, 29071 Málaga, Spain* **23**(17): p7491. 20p.

Shanmugapriya, D., Padmavathi, G. and Aysha, A. (Mar 2023). Detection of iris template attacks using machine learning and deep learning methods, *10th International Conference on Computing for Sustainable Global Development (INDIACom) Computing for Sustainable Global Development (INDIACom), 2023 10th International Conference* pp. 425–429.