Enhancing Ocular Disease Diagnosis in Fundus Images using Deep Learning

Md. Marop Hossain
ID#2013982042
Department ECE, Major CSE
marop.hossain@northsouth.edu

Abstract

Artificial intelligence (AI) based on deep learning (DL) is an emerging technology widely adopted in image recognition, speech recognition, and natural language processing. Deep learning is in the beginning stage of impacting healthcare, mostly in the field of ophthalmology. These methods have achieved outstanding performance in detecting ocular diseases such as diabetic retinopathy, glaucoma, diabetic macular degeneration, cataracts, and age-related macular degeneration. In this project, I used custom CNN, and four pre-trained models—EfficientNetV2S, DenseNet-121, ResNet-50, and ResNeXt-50—were trained and evaluated on the ODIR dataset. The EfficientNetV2S model achieved the highest categorical accuracy at 100% with lower misclassified testing images, followed by ResNet-50 (99.96%), DenseNet-121 (99.16%), and ResNeXt-50 (94.35%). The proposed system leverages these models to identify and classify ocular diseases, making diagnosis more accessible, accurate, and efficient. This advancement has the potential to significantly enhance early detection and treatment, ultimately improving patient outcomes and transforming the practice of ophthalmology.

Keywords—deep learning, ophthalmology, artificial intelligence, EfficientNetV2S, ResNet50, DenseNet121, ResNeXt50, CNN

I. INTRODUCTION

A. BACKGROUN AND MOTIVATION

The eye is a vital organ for a human being that plays a critical role compared to other organs of the human body. Eye diseases can cause vision loss or blindness, which can have a profound impact on a person's quality of life. There are a lot of eye diseases, such as diabetes, glaucoma, cataracts, age-related macular degeneration, hypertension, pathological myopia, and abnormalities. Globally, 600 million people will have diabetes by 2040, with a third having DR [1]. It is projected that 288 million patients may have some forms of AMD by 2040 [2]. The global prevalence of glaucoma for people aged 40–80 is 3.4%, and by the year 2040 it is projected there will be approximately 112 million affected individuals world-wide [3]. To prevent such eye diseases, early detection and timely treatment can help reduce the risk of great loss of vision. Most eye care institutions are not well off in many underdeveloped nations. Furthermore, reliable medical treatment and ophthalmologists are scarce in rural areas. So, it becomes quite tough for the people of rural areas to carry the expenses of better treatment. As the population is increasing, the number of patients with eye diseases is also rapidly increasing. So, it's a community's or government's obligation to improve eye care facilities for its citizens. A system that can detect eye diseases from retinal fundus images can be developed using digital image processing and machine learning. The system will take the retinal fundus images as input. Then, from the fundus images, the system can extract and simplify the features of specific eye diseases.

B. PURPOSE AND GOAL OF THE PROJECT

The purpose of this project is to develop a system using deep learning through CNN that will use the retinal fundus image to identify, extract, and evaluate disease-specific characteristics. This system will help to early detect eye diseases, which allows patients to maintain a good quality of vision while avoiding serious vision loss and blindness at an early age

To achieve the goal required objectives taken:

- Determine a deep learning model that can accurately identify, extract, and evaluate disease-specific characteristics from retinal fundus images
- Create a system that is easy to use and accessible to healthcare professionals, so that they can quickly and accurately diagnose eye diseases
- Help to improve the early detection and treatment of eye diseases, which can lead to better patient outcomes. And early detection of disease

II. RESEARCH LITERATURE REVIEW

Ocular Disease Detection Using Advanced Neural Network Based Classification Algorithms [4] paper presents an in-depth exploration of the challenges in early screening and diagnosis of ocular diseases from fundus images, highlighting the need for computer-aided automated detection systems. To address this, they develop and evaluate four deep learning-based models—Resnet-34, Efficient-Net, MobileNetV2, and VGG-16 – for ocular disease detection. These models are trained on the Ocular Disease Intelligent Recognition (ODIR) dataset, comprising 5000 fundus images belonging to 8 different ocular disease classes. These categories include eight disease classes that are Normal (N), diabetes (D), cataract (C), glaucoma (G), age-related macular degeneration (A), myopia (M), hypertension (H), and other abnormalities/diseases (O). The VGG-16 model achieved the highest accuracy of 97.23%, followed by Resnet-34 (90.85%), MobileNetV2 (94.32%), and Efficient-Net (93.82%). The study concludes that these models can contribute to the development of a real-time ocular disease diagnosis system.

Convolutional Neural Network Modeling for Eye Disease Recognition [5] addresses the pressing issue of eye diseases in Bangladesh, where limited access to healthcare services in rural areas contributes to a lack of awareness and early diagnosis. This paper suggests an osteopathic expert system that uses convolutional neural networks (CNNs) to identify common eye illnesses, including cataract, chalazion, and squint. A dataset of disease and disease-free eye images was collected, pre-processed, and augmented to develop and

evaluate the performance of six CNN models: VGG16, VGG19, Mobile-Net, Xception, InceptionV3, and DenseNet121. The results show that the Mobile-Net model achieves the highest accuracy of 97.49%, surpassing other recent works. The study aims to contribute to early disease detection and enhance healthcare accessibility in Bangladesh.

Deep Transfer Learning Models for Medical Diabetic Retinopathy Detection [6] addresses the growing concern about diabetic retinopathy (DR), a significant diabetic eye disease leading to blindness. With the increasing number of diabetic patients projected to reach 552 million by 2034, the study explores the potential of artificial intelligence (AI) and deep learning (DL) techniques for early detection of DR, aiming to enhance recovery chances and minimize vision loss. The authors investigate various deep transfer learning models using the Asia Pacific Tele-Ophthalmology Society (APTOS) 2019 dataset, which contains five classes representing different stages of DR severity. The dataset contains 3662 images and consists of 5 classes such as no diabetic retinopathy, mild non-proliferative retinopathy, moderate non-proliferative retinopathy, severe non-proliferative retinopathy, and proliferative diabetic retinopathy (PDR). The results indicate that Alex-Net achieved the highest testing accuracy at 97.9%, along with performance metrics such as precision, recall, and F1 score. The use of augmentation techniques to address overfitting and the choice of specific CNN models like Alex-Net, ResNet18, Squeeze-Net, Google-Net, VGG16, and VGG19 are highlighted.

Deep Learning for Ocular Disease Recognition: An Inner-Class Balance [7] paper mentioned that it can be challenging for doctors to identify eye disorders early enough using fundus pictures. Finding eye diseases by hand takes a long time and can involve mistakes. So, we need a computer system to help find eye problems in pictures. This study uses deep learning, which is a type of computer program, to find eye diseases. For this study, they used state-of-the-art image classification algorithms, such as VGG-19, to classify the ODIR dataset, which contains 5000 images of eight different classes of the fundus. These classes represent different ocular diseases. However, the dataset within these classes is highly unbalanced. To resolve this issue, the work suggested converting this multiclass classification problem into a binary classification problem and taking the same number of images for both classifications. Then, the binary classifications were trained with VGG-19. The accuracy of the VGG-19 model was 98.13% for the normal (N) versus pathological myopia (M) class; the model reached an accuracy of 94.03% for normal (N) versus cataract (C), and the model provided an accuracy of 90.94% for normal (N) versus glaucoma (G). All of the other models also improve accuracy when the data is balance.

III. METHODOLOGY

A. System Design

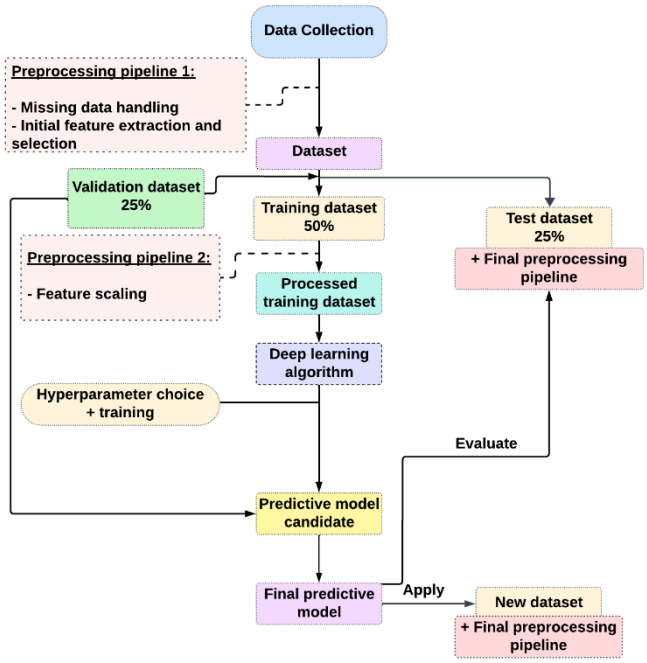


Figure: System Design Diagram

Finding best suitable data: I used the ODIR dataset combined with 4 classes such as cataract (C), normal (N), diabetes (D), and glaucoma (G) as the training and testing process.

Preprocessing pipeline 1: The next step was to preprocess the data. This involved cleaning the data, removing any missing data, and performing initial feature extraction and selection. The goal of preprocessing was to prepare the data for training the neural network model, and we used image hashing to identify duplicate images. After finding duplicate images, we removed them from the dataset.

Dataset Splitting: I split the dataset into three subsets: training, validation, and test sets. The **training set (50%)** was used to train the model, a **validation set (25%)** was used to monitor the model's performance during training, and a **test set (25%)** was used to evaluate the model's performance on unseen data.

Divided Set	Image No
Training set (50%)	2108
Validation set (25%)	1052
Testing set (25%)	1057
Total image count	4217

Table: Dataset Distribution

Preprocessing pipeline 2: I preprocessed the training set once again using a second preprocessing pipeline. This pipeline involved feature scaling, which was a technique that normalized the features to have a common scale. This helped the neural network model learn more effectively.

Deep learning algorithm: The next step was to choose a deep learning algorithm to train the model. A variety of deep learning algorithms could be used for image classification, but convolutional neural networks (CNNs) were particularly well-suited for this task. CNNs are a type of neural network

that are able to learn spatial features from images. I used CNN and four deep learning-based models for targeted ocular disease diagnosis. For this project, I trained cutting-edge classification algorithms such as **EfficientNetV2S, DenseNet-121, ResNet-50,** and **ResNeXt-50** on the ODIR dataset consisting of **4217 fundus images** that belonged to **4 different classes** such as **normal, cataract, glaucoma,** and **diabetic retinopathy.** Each of these classes represented a different ocular disease.

CNN model architecture involved convolutional layers, activation functions (ReLU and softmax), max-pooling layers, dropout layers, hidden fully connected layers, and the output layer. The model also used the Adam and RMSProp optimizers and a categorical cross-entropy loss function. Hyperparameters such as batch size, learning rate, and the number of epochs were also used.

Hyperparameter choice and training: After I had chosen a deep learning algorithm, the next step was to select the hyperparameters for the model. Hyperparameters were parameters that controlled the training process, such as the learning rate and the number of epochs. I chose the hyperparameters using a process called grid search. Once the hyperparameters had been chosen, I trained the model on the training set.
Let's discuss the hyperparameter section that I used during the training process

i. EfficientNetV2S

Hypermeter	EfficientNetV2S
Data Augmentation	RandomFlip (horizontal), RandomRotation (0.1), RandomContrast (0.1)
Input shape	(160, 160, 3)
Base Model	EfficientNetV2S
Base Model Trainable	True
Re-scaling	1./255
Batch Normalization	Yes
Dense Layer Units	256
Dense Layer Activation	ReLU
Kernel Regularizer	L2(0.016)
Bias Regularizer	L1(0.006)
Activity Regularizer	L1(0.006)
Dropout Rate	40%
Optimizer	Adamax (LR=0.001)
Loss Function	Categorical Cross-Entropy
Number of Epochs	200
Early Stop Epochs (before overfitting)	88
Best Epoch (model return after complete execution)	78
Model Checkpoint	yes
Early Stopping	monitor='val_loss', mode='min', verbose=1, patience=10, min_delta=0.001
Learning Rate	ReduceLRonPlatea
Scheduler	u (monitor='val_loss, factor=0.2, patience=3, verbose=1, mode='min' min_lr=0.00001)
Learning Rate	initial: 0.001
CSV Logger	yes

ii. ResNet50

Hypermeter	EfficientNetV2S
Data Augmentation	RandomFlip (horizontal), RandomRotation (0.1), RandomContrast (0.1)
Input shape	(160, 160, 3)
Base Model	ResNet50 (pre-trained)
Base Model Trainable	True
Re-scaling	1./255
Batch Normalization	Yes
Dense Layer Units	256
Dense Layer Activation	ReLU
Kernel Regularizer	L2(0.001)
Bias Regularizer	L1(0.001)
Activity Regularizer	L1(0.001)
Dropout Rate	50%
Optimizer	Adamax (LR=0.0001)
Loss Function	Categorical Cross-Entropy
Number of Epochs	200
Early Stop Epochs (before overfitting)	82
Best Epoch (model return after complete execution)	72
Model Checkpoint	yes
Early Stopping	monitor='val_loss', mode='min', verbose=1, patience=10, min_delta=0.001
Learning Rate	ReduceLRonPlatea
Scheduler	u (monitor='val_loss, factor=0.2, patience=3, verbose=1, mode='min' min_lr=0.00001)
Learning Rate	initial: 0.0001
CSV Logger	Yes

iii. DenseNet121

Hypermeter	EfficientNetV2S
Data Augmentation	RandomFlip (horizontal), RandomRotation (0.1), RandomContrast (0.1), RandomCrop (height=160, width=160)
Input shape	(160, 160, 3)
Base Model	DenseNet121 (pre-trained)
Base Model Trainable	True
Re-scaling	1./255
Batch Normalization	Yes
Dense Layer Units	256
Dense Layer Activation	ReLU
Kernel Regularizer	L2(0.001)
Bias Regularizer	L1(0.001)
Activity Regularizer	L1(0.001)
Dropout Rate	40%
Optimizer	Adamax (LR=0.00001)
Loss Function	Categorical Cross-Entropy
Number of Epochs	300
Early Stop Epochs (before overfitting)	131
Best Epoch (model return after complete execution)	121
Model Checkpoint	Yes
Early Stopping	monitor='val_loss', mode='min', verbose=1, patience=10, min_delta=0.001
Learning Rate	ReduceLRonPlatea
Scheduler	u (monitor='val_loss, factor=0.2, patience=5, verbose=1, mode='min' min_lr=0.00001)
Learning Rate	initial: 0.00001
CSV Logger	yes

iv. ResNeXt50

Hypermeter	EfficientNetV2S
Data Augmentation	RandomFlip (horizontal), RandomRotation (0.1), RandomContrast (0.1), RandomCrop (height=160, width=160), RandomZoom (0.2)
Input shape	(160, 160, 3)
Base Model	ResNeXt50 (pre-trained)
Base Model Trainable	True
Re-scaling	1./255
Batch Normalization	Yes
Dense Layer Units	256
Dense Layer Activation	ReLU
Kernel Regularizer	L2(0.001)

Bias Regularizer	L1(0.001)
Activity Regularizer	L1(0.001)
Dropout Rate	50%
Optimizer	Adamax (LR=0.0001)
Loss Function	Categorical Cross-Entropy
Number of Epochs	250
Early Stop Epochs (before overfitting)	41
Best Epoch (model return after complete execution)	31
Model Checkpoint	yes
Early Stopping	monitor='val_loss', mode='min', verbose=1, patience=10, min_delta=0.001
Learning Rate	ReduceLRonPlatea
Scheduler	u (monitor='val_loss, factor=0.2, patience=3, verbose=5, mode='min' min_lr=0.00001)
Learning Rate	initial: 0.001
CSV Logger	yes

Custom CNN Model:

Hypermeter	EfficientNetV2S
Data Augmentation	RandomFlip (horizontal), RandomRotation (0.1), RandomContrast (0.1),
Input shape	(160, 160, 3)
Base Model	-
Base Model Trainable	-
Re-scaling	1./255
Batch Normalization	Yes
Dense Layer Units	256
Dense Layer Activation	ReLU
Kernel Regularizer	L2(0.016)
Bias Regularizer	L1(0.006)
Activity Regularizer	L1(0.006)
Dropout Rate	40%
Optimizer	Adamax (LR=0.001)
Loss Function	Categorical Cross-Entropy
Number of Epochs	200
Early Stop Epochs (before overfitting)	82
Best Epoch (model return after complete execution)	72
Model Checkpoint	yes
Early Stopping	monitor='val_loss', mode='min', verbose=1, patience=10, min_delta=0.001
Learning Rate	ReduceLRonPlatea
Scheduler	u (monitor='val_loss, factor=0.2, patience=3, verbose=1, mode='min' min_lr=0.00001)
Learning Rate	initial: 0.001
CSV Logger	yes

Evaluation: I evaluated the model on the **validation set (25%)** once it had been trained. This was done to assess the performance of the model and identify any areas where it could be improved. The model was then tuned by adjusting the hyperparameters and retraining the model. This process was repeated until the model was performing well on the validation set.

Predictive model candidate: In this case, I used different types of hyperparameters, a training set, and a validation set of all the models, as I will calculate each model's best performance on the validation set and training set. I also calculated the validation and training losses, precision, recall, and f1-score. In these steps, I also knew that the current model is either face overfitting, underfitting, or balance fit or not.

Final Predictive Model: The model that performs best on the validation set and testing will be selected as the predictive model candidate.

In this step, I tested all the models on a dataset of **1057 images,** as I could calculate the model performance and the number of misclassified test images in the testing set. If I get better accuracy and a lower number of misclassified images in the testing set, I will choose the best model for the tasks where it is important to correctly classify all of the images and minimize the number of errors. It is also important to note that the prediction accuracy of the test data is not the only factor to consider when choosing the best model. Other factors, such as the model's training time and inference speed, may also have been important.

B. Required Software

Tools	Why I selected this tool
Python [8]	Python is a versatile programming language that is well-suited for deep learning. It is also relatively easy to learn and use.
TensorFlow [9]	TensorFlow is strong, flexible, and supported by the community. It offers many resources and features for building and training neural networks
Keras [10]	It makes it easier to build and train deep learning models
NumPy [11]	Data preparation and analysis in eye disease detection were made more accessible using NumPy
Pandas [12]	Pandas due to their effective data handling, numerical computing, and deep learning integration.
Matplotlib [13], Seaborn [14]	These libraries provide me some powerful tools for data visualization which can I use them to visualize fundus image, explore the distribution of image feature, and analyze model performance such as plotting fundus image, visualizing model predications etc.,
Scikit-learn [15]	It provides me wide range of deep learning algorithms and tools for model evaluation and performance metrics such as preprocessing, model evaluation, Hyperparameter tuning, and model selection.
Google Collab pro	It provides solid GPUs and TPUs, available with Google Collab Pro, which helps train deep-learning models

C. Model Explanation

I used four deep learning model in this project such **EfficientNetV2S, ResNet50, DenseNet121**, and **ResNeXt50**. Let's break down each of these models…

EfficientNetV2S [16]: This cutting-edge convolutional neural network (CNN) architecture is known for its extraordinary performance and efficiency. It achieves this by consistently scaling the network's depth, breadth, and resolution in a balanced way using a revolutionary compound scaling approach. This guarantees that the model is increasingly complicated the deeper it goes, enabling it to capture complex patterns and features in images effectively. Because of its

particular optimization for smaller devices, the "V2S" variation is appropriate for resource-constrained contexts like edge computing and mobile devices. EfficientNetV2S is perfect for applications like image classification and feature extraction, including identifying eye illnesses from ocular pictures, because it is small and still retains remarkable accuracy.

ResNet50 [17]: ResNet50 is a deep convolutional neural network characterized by its ingenious use of residual connections, or skip connections, which help alleviate the vanishing gradient problem and enable the training of intense networks. ResNet50 consists of 50 layers and is renowned for its ability to learn intricate hierarchical features from images. Each residual block in ResNet50 contains multiple convolutional layers, and the shortcut connections allow the network to bypass specific layers, facilitating the flow of information through the network. This design enables ResNet50 to effectively capture fine-grained details and subtle patterns in images, making it well-suited for complex tasks such as image recognition and object detection.

DenseNet121 [18]: DenseNet121 is a notable convolutional neural network architecture characterized by its densely connected layers, with each layer directly receiving input from all preceding layers. This dense connectivity pattern encourages feature reuse, promotes gradient flow throughout the network, and facilitates effective feature propagation and diversity. With 121 layers, DenseNet121 is highly regarded for its efficient parameter utilization and robust feature learning capabilities. Its strength lies in capturing complex spatial dependencies and semantic information in images through dense inter-layer communication, making it particularly well-suited for tasks that demand precise feature extraction and classification, such as medical image analysis and disease diagnosis.

ResNeXt50 [19]: ResNeXt50 is an extension of the ResNet architecture that introduces the concept of "cardinality," representing the number of parallel pathways within each residual block. By combining multiple paths with different weights, ResNeXt50 improves model expressiveness and feature diversity without significantly increasing computational complexity. This innovative design enables ResNeXt50 to effectively capture a wide range of features and spatial dependencies in images, leading to enhanced performance in tasks such as image classification and object detection. With its adaptable and scalable architecture, ResNeXt50 provides a robust framework for extracting distinctive features and addressing challenging visual recognition tasks, including the detection of subtle abnormalities in medical images such as retinal scans for eye disease diagnosis.

D. Data collection

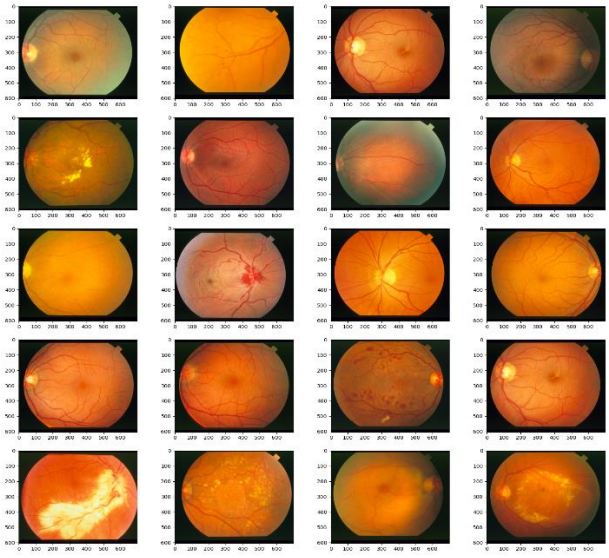
I collected a large dataset of **Ocular Disease Intelligent Recognition (ODIR) [20]** is a structured ophthalmic database of **5,000 patients** with age, color fundus photographs from left and right eyes and doctors' diagnostic keywords from doctors.

This dataset is meant to represent a **'real-life'** set of patient information collected by Shang gong Medical Technology Co., Ltd. from different hospitals/medical centers in China. In these institutions, fundus images are captured by various cameras in the market, such as Canon, Zeiss and Kowa, resulting in varied image resolutions. Annotations were labeled by trained human readers with quality control management. They classify patient into ***eight labels*** including:

i. *Normal (N),*
ii. *Diabetes (D),*
iii. *Glaucoma (G),*
iv. *Cataract (C),*
v. *Age related Macular Degeneration (A),*
vi. *Hypertension (H),*
vii. *Pathological Myopia (M),*
viii. *Other diseases/abnormalities (O)*

This dataset also has ***8 classes***, but I will use ***4 classes*** for my project, such as **normal**, **glaucoma**, **cataract**, and **diabetic retinopathy.** Diabetic retinopathy also has **four subcategories**, such as

- *mild non-proliferative retinopathy,*
- *proliferative retinopathy,*
- *diabetic retinopathy, and*
- *moderate non-proliferative retinopathy.*



Classes	Image No.
Cataract	1038
Diabetic retinopathy	1098
Glaucoma	1007
Normal	1074
Total Image count	4217

Table: Image Distribution

IV. Result

On an ocular disease project, I investigated the performance of five deep learning models: **CustomCNN**, **EfficientNetV2S, ResNet50, DenseNet121,** and **ResNext50**. I trained all models with data augmentation and evaluated their performance on a held-out validation set.

Model	Training					
	BE*	Precision	Recall	F1-score	CA*	Loss
CustomCNN	72/200	0.91	0.71	0.77	0.82	0.76
EfficientNetV2S	78/200	1.00	0.99	0.99	1.00	0.18
ResNet50	73/200	0.99	0.99	0.99	0.99	0.34
DenseNet121	121/300	0.99	0.99	0.99	0.99	0.47
ResNeXt50	31/250	0.95	0.92	0.93	0.94	0.78

Table: Performance accuracy during training of four models

CA* = *Categorical Accuracy*, ***BE**** = *Best Epoch*

	CustomCNN	EfficientNetV2S	ResNet50	DenseNet121	ResNeXt50
CA*	82.00%	100%	99.96%	99.16%	94.35%
VCA*	72.00%	98.58%	98.39%	84.03%	64.92%
AUC PD*	5.11%	7.67%	3.98%	2.79%	11.11%
APD*	13.02%	1.42%	1.56%	15.24%	31.19%
TTI*	1057	1057	1057	1057	1057
MTI*	327	23	32	163	377
PATS*	69.06%	97.82%	96.98 %	84.59%	64.33%

Table: accuracy comparison of training, testing, and validation sets of five models

CA* = *Categorical Accuracy*, ***VCA**** = *Validation categorical accuracy*, ***AUC PD**** = *AUC percentage difference*, ***APD**** = *Accuracy percentage difference*, ***TTI**** = *Total testing images*, ***MTI**** = *Misclassified test images*, ***PATS**** = *Prediction accuracy on the test set*

A. CustomCNN

CustomCNN training and validation accuracy:

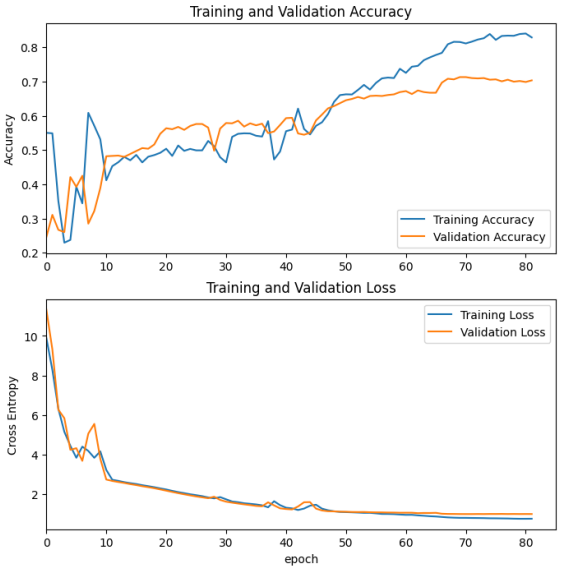


Figure: Training and validation accuracy Graph of CustomCNN

CustomCNN confusion matrix:

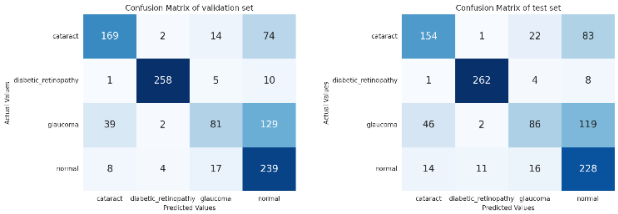


Figure: confusion matrix of CustomCNN

CustomCNN classification report:

	Precision	Recall	F1-score	Support
Cataract	0.7163	0.5923	0.6484	260
Diabetic Retinopathy	0.9493	0.9527	0.9510	275
Glaucoma	0.6719	0.3329	0.4514	253
Normal	0.5205	0.8476	0.6450	269
Micro Avg	0.9606	0.6906	0.6906	1057
Macro Avg	0.7145	0.6831	0.6740	1057
Weighted Avg	0.7165	0.6906	0.6791	1057
Samples Avg	0.6906	0.6906	0.6906	1057

Table: Classification report of CustomCNN

B. EfficientNetV2S

EfficientNetV2S training and validation accuracy:

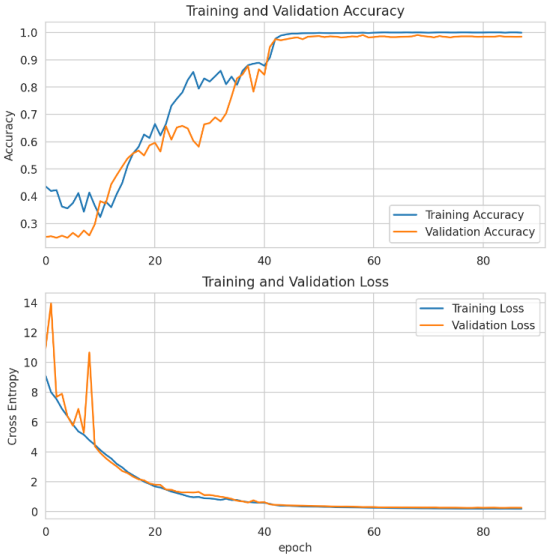


Figure: Training and validation accuracy Graph of EfficientNetV2S

EfficientNetV2S confusion matrix:

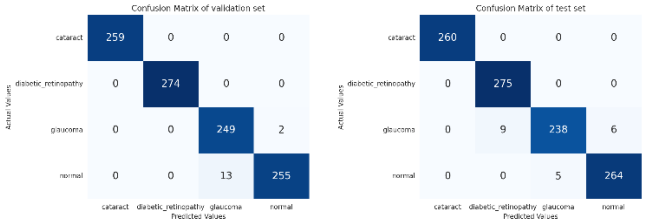


Figure: confusion matrix of EfficientNetV2S

EfficientNetV2S classification report:

	Precision	Recall	F1-score	Support
Cataract	1.0000	0.9962	0.9981	260
Diabetic Retinopathy	0.9683	1.0000	0.9839	275
Glaucoma	0.9714	0.9407	0.9558	253
Normal	0.9740	0.9704	0.9704	269
Micro Avg	0.9782	0.9782	0.9782	1057
Macro Avg	0.9784	0.9777	0.9779	1057
Weighted Avg	0.9783	0.9782	0.9781	1057
Samples Avg	0.9782	0.9782	0.9782	1057

Table: Classification report of EfficientNetV2S

C. ResNet50

ResNet50 training and validation accuracy:

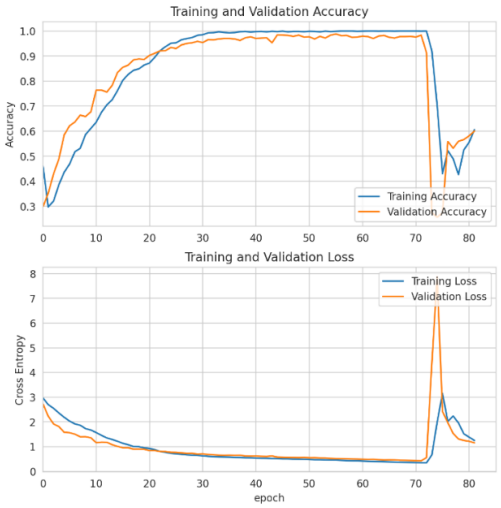


Figure: Training and validation accuracy Graph of ResNet50

ResNet50 confusion matrix:

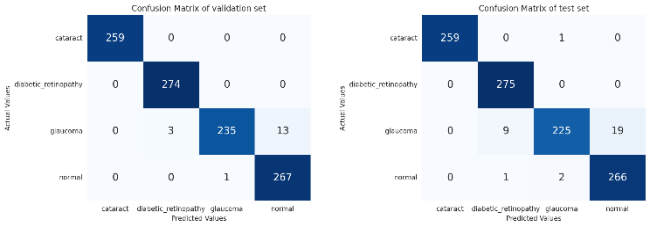


Figure: confusion matrix of ResNet50

ResNet50 classification report:

	Precision	Recall	F1-score	Support
Cataract	1.0000	0.9962	0.9981	260
Diabetic Retinopathy	0.9649	1.0000	0.9821	275
Glaucoma	0.9868	0.8893	0.9356	253
Normal	0.9333	0.9888	0.9603	269
Micro Avg	0.9697	0.9697	0.9697	1057
Macro Avg	0.9713	0.9686	0.9690	1057
Weighted Avg	0.9708	0.9697	0.9693	1057
Samples Avg	0.9697	0.9697	0.9697	1057

Table: Classification report of ResNet50

D. DenseNet121

DenseNet121 training and validation accuracy:

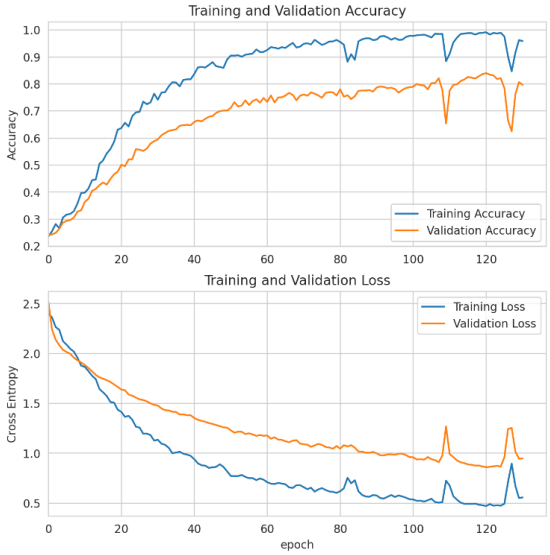


Figure: Training and validation accuracy Graph of DenseNet121

DenseNet121 confusion matrix:

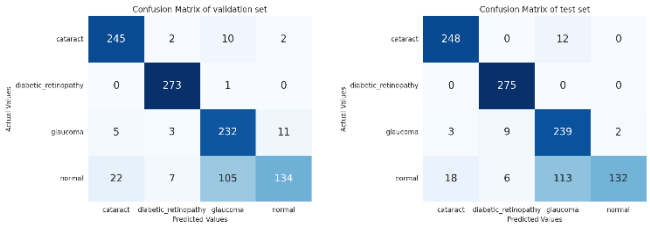


Figure: confusion matrix of DenseNet121

DenseNet121 classification report:

	Precision	Recall	F1-score	Support
Cataract	0.9219	0.9538	0.9376	260
Diabetic Retinopathy	0.9483	1.0000	0.9735	275
Glaucoma	0.6566	0.9447	0.7747	253
Normal	0.9851	0.4907	0.6551	269
Micro Avg	0.8458	0.8458	0.8458	1057
Macro Avg	0.8780	0.8473	0.8352	1057
Weighted Avg	0.8813	0.8458	0.8360	1057
Samples Avg	0.8458	0.8458	0.8458	1057

Table: Classification report of DenseNet121

E. ResNeXt50

ResNeXt50 training and validation accuracy:

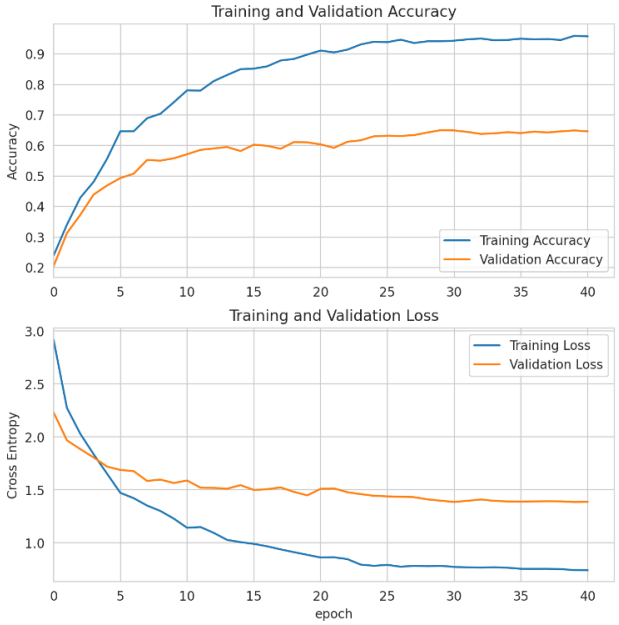


Figure: Training and validation accuracy Graph of ResNeXt50

ResNeXt50 confusion matrix:

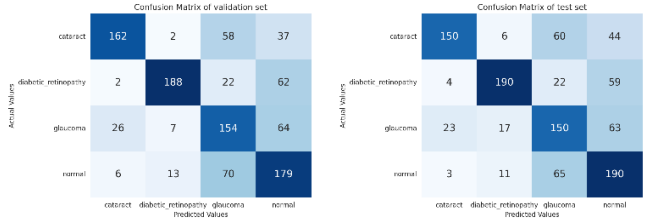


Figure: confusion matrix of ResNeXt50

ResNeXt50 classification report:

	Precision	Recall	F1-score	Support
Cataract	0.8333	0.5769	0.6818	260
Diabetic Retinopathy	0.8482	0.6909	0.7615	275
Glaucoma	0.5051	0.5929	0.5455	253
Normal	0.5337	0.7063	0.608	269
Micro Avg	0.6433	0.6433	0.6433	1057
Macro Avg	0.6801	0.6418	0.6492	1057
Weighted Avg	0.6824	0.6433	0.6511	1057
Samples Avg	0.6433	0.6433	0.6433	1057

Table: Classification report of ResNeXt50

Analysis

I evaluated the performance of Five deep learning models (**CustomCNN**, **ResNet50, EfficientNetV2S, DenseNet121,** and **ResNeXt50**) for classifying ocular disease diagnosis. I tested the models on a dataset of **1057 fundus images** labeled with one of four classes: **cataract, diabetic retinopathy, glaucoma, and normal.**

As you saw, [*Table: accuracy comparison of training, testing, and validation sets of five models*] **EfficientNetV2S** achieved the highest accuracy on the testing and validation sets, which were **97.82%** and **98.58%** respectively**.** ResNet50 also performed well, with a testing accuracy of **96.98%** and a validation accuracy of **98.39%**. DenseNet121 and ResNeXt50, CustomCNN, on the other hand, performed significantly worse, with testing accuracy of **84.58%** and **64.33%, 69.06%** and validation accuracy of **84.03%** and **64.92%**, **72.00%,** respectively.

Also, we can see that ResNet50 and EfficientNetVS2 outperformed the other models in our project, with categorical accuracies of **99.95%** and **100%**, respectively. This indicates that they could classify almost all of the images in the dataset correctly. DenseNet121 performed slightly worse, with a categorical accuracy of 99.15%, while ResNeXt-50, CustomCNN had the lowest categorical accuracy at 94.35% and 82.00%.

EfficientNetV2S misclassified **23 test images**, resulting in a prediction accuracy of 97.82%.
ResNet50 misclassified **32 test images**, resulting in a prediction accuracy of 96.97%.
DenseNet121 misclassified **163 test images**, resulting in a prediction accuracy of 84.58%.
ResNeXt-50 misclassified **377 test images**, resulting in a prediction accuracy of 64.33%.
CustomCNN misclassified **327 test images,** resulting in a prediction accuracy of 69.03%

EfficientNetV2S had the best precision and recall values [*Table: accuracy comparison of training, testing, and validation sets of five models*], followed by ResNet50 and DenseNet121. ResNeXt-50 had the lowest precision and recall values.

EfficientNetV2S vs. ResNet50 for Glaucoma and Normal Image Prediction [EfficientNetV2S confusion matrix and ResNet50 confusion matrix]. I compared the performance of our two best state-of-the-art deep learning models, EfficientNetV2S and ResNet50, on a four-class classification task of normal, cataract, diabetic retinopathy, and glaucoma images. Using the validation set, ResNet50 correctly predicted cataract, diabetic retinopathy, and normal class images but misclassified 16 glaucoma images. EfficientNetV2S, on the other hand, correctly predicted all cataract, diabetic retinopathy, and glaucoma images but misclassified 14 normal images. Based on these results, we concluded that ResNet50 is not well suited for predicting glaucoma images, while EfficientNetV2S could be better suited for predicting normal images.

V. CONCLUSION

The ocular diseases project focused on improving ocular disease diagnosis using advanced deep-learning techniques. I employed the Ocular Disease Recognition (ODIR) dataset, containing 5,000 patient records with fundus images, reflecting real-world scenarios. I primarily focused on three common diseases: diabetes retinopathy, glaucoma, and cataracts. I implemented Convolutional Neural Networks (CNNs) and four pre-trained models, including EfficientNetV2S, ResNet-50, DenseNet-121, and ResNeXt-50, to classify fundus images into specific disease categories. EfficientNetV2S was the best-performing model on the validation and testing sets after hyperparameter tuning, and the testing set confusion matrix shows that it has lower misclassified images than the other models. ResNet50 is close to performing well, with good testing and validation accuracy and the lowest misclassified images in the testing set. Also, we saw that EfficientNetV2S had the best precision and recall values, followed by ResNet50 and DenseNet121. ResNeXt50 had the lowest precision and recall values. However, DenseNet121, another deep learning model, achieved the highest overall accuracy on the validation set but struggled in the testing phase due to many misclassifications. ResNeXt-50, a variant of ResNet50, performed slightly worse than ResNet50, and our best two models, ResNet50 and EfficeintNetV2S's confusion matrix, also show that ResNet50 is not well suited for predicting glaucoma images, while EfficientNetV2S is not well suited for predicting normal images. In the future, we must fine-tune and ensemble these two models to improve their performance. The new model will be able to predict all four classes of images correctly.

Finally, we concluded that ResNet50 and EfficientNetV2s are our two-best deep-learning models well-suited for classifying eye diseases.

REFERENCES

[1] Yau JW, Rogers SL, Kawasaki R, et al Global prevalence and major risk factors of diabetic retinopathy. Diabetes Care 2012;35:556–64.

[2] Wong WL , Su X , Li X , et al Global prevalence of age-related macular degeneration and disease burden projection for 2020 and 2040: a systematic review and meta-analysis. Lancet Glob Health 2014;2:e106-16.

[3] Tham YC, Li X, Wong TY, et al Global prevalence of glaucoma and projections of glaucoma burden through 2040: a systematic review and meta-analysis. Ophthalmology 2014;121:2081–90.

[4] Dipu, N. M., Alam Shohan, S., & Salam, K. (2021, August 18). Ocular Disease Detection Using Advanced Neural Network Based Classification Algorithms. *ASIAN JOURNAL OF CONVERGENCE IN TECHNOLOGY*, *7*(2), 91–99. https://doi.org/10.33130/ajct.2021v07i02.019

[5] Siddique, M. A. A., Jannatul Ferdouse, Md. Tarek Habib, Md. Jueal Mia, & Mohammad Shorif Uddin. (2022, July 11). Convolutional Neural Network Modeling for Eye Disease Recognition. *International Journal of Online and Biomedical Engineering (IJOE)*, *18*(09), 115–130. https://doi.org/10.3991/ijoe.v18i09.29847

[6] Khalifa, N., Loey, M., Taha, M., & Mohamed, H. (2019). Deep Transfer Learning Models for Medical Diabetic Retinopathy Detection. *Acta Informatica Medica*, *27*(5), 327. https://doi.org/10.5455/aim.2019.27.327-332

[7] Khan, M. S., Tafshir, N., Alam, K. N., Dhruba, A. R., Khan, M. M., Albraikan, A. A., & Almalki, F. A. (2022, April 28). Deep Learning for Ocular Disease Recognition: An Inner-Class Balance. *Computational Intelligence and Neuroscience*, *2022*, 1–12. https://doi.org/10.1155/2022/5007111

[8] *Welcome to Python.org*. (2024, May 8). Python.org. https://www.python.org/

[9] *TensorFlow*. (n.d.). TensorFlow. https://www.tensorflow.org/

[10] Team, K. (n.d.). Keras documentation: Keras 3 API documentation. https://keras.io/api/

[11] *NumPy Tutorial*. (n.d.). https://www.w3schools.com/python/numpy/default.asp

[12] *Getting started — pandas 2.2.2 documentation*. (n.d.). https://pandas.pydata.org/docs/getting_started/index.html

[13] Matplotlib — Visualization with Python. (n.d.). https://matplotlib.org/

[14] *API reference — seaborn 0.13.2 documentation*. (n.d.). https://seaborn.pydata.org/api.html

[15] scikit-learn: machine learning in Python — scikit-learn 1.4.2 documentation. (n.d.). https://scikit-learn.org/stable/

[16] Team, K. (n.d.). *Keras documentation: EfficientNetV2 B0 to B3 and S, M, L*. https://keras.io/api/applications/efficientnet_v2/

[17] Team, K. (n.d.). *Keras documentation: ResNet and ResNetV2*. https://keras.io/api/applications/resnet/

[18] Team, K. (n.d.). *Keras documentation: DenseNet*. https://keras.io/api/applications/densenet/

[19] P. (n.d.). *GitHub - pth051001/ResNeXt50-Implementation-and-Transfer-Learning-with-ResNet50V2*. GitHub. https://github.com/pth051001/ResNeXt50-Implementation-and-Transfer-Learning-with-ResNet50V2

[20] *ODIR-2019 - Grand Challenge*. (n.d.). grand-challenge.org. https://odir2019.grand-challenge.org/