Project Report: Ocular Disease Recognition

Daniel Song, Justin Hwang, and Kirtitej Gandham

https://github.com/dsong124/Predicting-Ocular-Disease

Problem Statement: This project aims to develop a deep learning-based classification model that can quickly detect ocular diseases through fundus images using the ODIR-5K dataset. In ophthalmology, early fundus screening is an effective way to prevent blindness caused by ophthalmic diseases such as glaucoma and ocular hyperextension. Being able to detect any changes is crucial for effective treatment, yet manual diagnosis is time-consuming and inconsistent, as it heavily depends on the doctor's experience in the field. In this dataset, we will utilize convolutional Neural Networks (CNNs) and aim to improve both diagnostic accuracy and efficiency within ophthalmology.

Strategic Aspects: Creating a robust model involves overcoming several challenges, such as dataset quality and variability, class imbalance, and interpretability of AI decisions. Images with different lighting conditions, resolution variations, and artifacts are all included in the ODIR-5K dataset. To improve the robustness of the model, preprocessing methods like contrast adjustment, noise reduction, and augmentation will be applied. It is also possible that some illnesses are underrepresented in the dataset. This problem will be lessened with the use of strategies like weighted loss functions and data augmentation. Also, making sure that models produce outputs that are both interpretable and clinically relevant is a major challenge when implementing AI in healthcare. In order to identify the areas of fundus images that influence model predictions, we intend to incorporate explainable AI techniques like Grad-CAM. There are also ethical and practical considerations, such as bias, clinical standards, and integration into healthcare systems. Because AI is capable and known for learning from past data, bias in disease detection across various demographic groups may exist. Therefore, using modeling techniques and balanced training data, we will reduce bias from the data. As for the clinical standards, to guarantee safety and dependability, AI must, at the very least, comply with ethical standards and healthcare laws. The model will have to be assessed using these metrics before ever applying. Finally, for integration into healthcare systems, it would be simpler for clinics to perform initial screenings before referring patients to specialists who would use applications.

Novelty and Importance:

Early detection of eye diseases through automated AI screening can help prevent vision loss, providing timely evaluations before conditions worsen. This approach improves the availability of diagnostic services, especially in areas with limited access to specialists. Unlike manual assessments, which can be subjective and inconsistent, AI-driven diagnosis provides objective and reproducible results. With the number of patients being much greater than the number of

ophthalmologists, AI screening is needed for such large-scale fundus screening. In order to avoid delays in treatment for patients, deep learning models are used to detect what manual examinations would pick up in much less time.

Dataset details:

- **Source**: Ocular Disease Recognition dataset
- **Type**: Fundus images from both the left and right eye with disease labels.
- **Classes**: Multiple disease categories, including normal, diabetic retinopathy, glaucoma, and cataracts (labeled as D, G, C, etc).
- **Features:** Image-based features, including color, texture, and other flaws that indicate ocular disease symptoms or diagnosis.

ID	Patient Age	Patient Sex	Left-Fundus	Right-Fundus	Left-Diagnostic Keywords	Right-Diagnostic Keywords	N	D	G	C	A	H	M	O	filepath	labels	target	filename
0	69	Female	0_left.jpg	0_right.jpg	cataract	normal fundus	0	0	0	1	0	0	0	0	../input/ocular-disease-recognition-odir5k/ODIR-5K/Training Images/0_right.jpg	['N']	[1, 0, 0, 0, 0, 0, 0, 0]	0_right.jpg
1	57	Male	1_left.jpg	1_right.jpg	normal fundus	normal fundus	1	0	0	0	0	0	0	0	../input/ocular-disease-recognition-odir5k/ODIR-5K/Training Images/1_right.jpg	['N']	[1, 0, 0, 0, 0, 0, 0, 0]	1_right.jpg
2	42	Male	2_left.jpg	2_right.jpg	laser spot, moderate non proliferative retinopathy	moderate non proliferative retinopathy	0	1	0	0	0	0	0	1	../input/ocular-disease-recognition-odir5k/ODIR-5K/Training Images/2_right.jpg	['D']	[0, 1, 0, 0, 0, 0, 0, 0]	2_right.jpg
4	53	Male	4_left.jpg	4_right.jpg	macular epiretinal membrane	mild nonproliferative retinopathy	0	1	0	0	0	0	0	1	../input/ocular-disease-recognition-odir5k/ODIR-5K/Training Images/4_right.jpg	['D']	[0, 1, 0, 0, 0, 0, 0, 0]	4_right.jpg
5	50	Female	5_left.jpg	5_right.jpg	moderate non proliferative retinopathy	moderate non proliferative retinopathy	0	1	0	0	0	0	0	0	../input/ocular-disease-recognition-odir5k/ODIR-5K/Training Images/5_right.jpg	['D']	[0, 1, 0, 0, 0, 0, 0, 0]	5_right.jpg
6	60	Male	6_left.jpg	6_right.jpg	macular epiretinal membrane	moderate non proliferative retinopathy, epiretinal membrane	0	1	0	0	0	0	0	1	../input/ocular-disease-recognition-odir5k/ODIR-5K/Training Images/6_right.jpg	['D']	[0, 1, 0, 0, 0, 0, 0, 0]	6_right.jpg
7	60	Female	7_left.jpg	7_right.jpg	drusen	mild nonproliferative retinopathy	0	1	0	0	0	0	0	1	../input/ocular-disease-recognition-odir5k/ODIR-5K/Training Images/7_right.jpg	['D']	[0, 1, 0, 0, 0, 0, 0, 0]	7_right.jpg
8	59	Male	8_left.jpg	8_right.jpg	normal fundus	normal fundus	1	0	0	0	0	0	0	0	../input/ocular-disease-recognition-odir5k/ODIR-5K/Training Images/8_right.jpg	['N']	[1, 0, 0, 0, 0, 0, 0, 0]	8_right.jpg
9	54	Male	9_left.jpg	9_right.jpg	normal fundus	vitreous degeneration	0	0	0	0	0	0	0	1	../input/ocular-disease-recognition-odir5k/ODIR-5K/Training Images/9_right.jpg	['O']	[0, 0, 0, 0, 0, 0, 0, 1]	9_right.jpg
10	70	Male	10_left.jpg	10_right.jpg	epiretinal membrane	normal fundus	0	0	0	0	0	0	0	1	../input/ocular-disease-recognition-odir5k/ODIR-5K/Training Images/10_right.jpg	['N']	[1, 0, 0, 0, 0, 0, 0, 0]	10_right.jpg
11	60	Female	11_left.jpg	11_right.jpg	moderate non proliferative retinopathy, hypertensive retinopathy	moderate non proliferative retinopathy, hypertensive retinopathy	0	1	0	0	0	1	0	0	../input/ocular-disease-recognition-odir5k/ODIR-5K/Training Images/11_right.jpg	['D']	[0, 1, 0, 0, 0, 0, 0, 0]	11_right.jpg
13	60	Female	13_left.jpg	13_right.jpg	pathological myopia	pathological myopia	0	0	0	0	0	0	1	0	../input/ocular-disease-recognition-odir5k/ODIR-5K/Training Images/13_right.jpg	['M']	[0, 0, 0, 0, 0, 0, 1, 0]	13_right.jpg
14	55	Male	14_left.jpg	14_right.jpg	normal fundus	macular epiretinal membrane	0	0	0	0	0	0	0	1	../input/ocular-disease-recognition-odir5k/ODIR-5K/Training Images/14_right.jpg	['O']	[0, 0, 0, 0, 0, 0, 0, 1]	14_right.jpg
15	50	Male	15_left.jpg	15_right.jpg	normal fundus	myelinated nerve fibers	0	0	0	0	0	0	0	1	../input/ocular-disease-recognition-odir5k/ODIR-5K/Training Images/15_right.jpg	['O']	[0, 0, 0, 0, 0, 0, 0, 1]	15_right.jpg
16	54	Female	16_left.jpg	16_right.jpg	normal fundus	pathological myopia	0	0	0	0	0	0	1	0	../input/ocular-disease-recognition-odir5k/ODIR-5K/Training Images/16_right.jpg	['M']	[0, 0, 0, 0, 0, 0, 1, 0]	16_right.jpg

Figure 1: Raw Dataset Snippet

Data Cleaning/Preprocessing

ID	target	filename
0	[1, 0, 0, 0, 0, 0, 0, 0]	0_right.jpg
0	[0, 0, 0, 1, 0, 0, 0, 0]	0_left.jpg
1	[1, 0, 0, 0, 0, 0, 0, 0]	1_right.jpg
1	[1, 0, 0, 0, 0, 0, 0, 0]	1_left.jpg
2	[0, 1, 0, 0, 0, 0, 0, 0]	2_right.jpg
3	[1, 0, 0, 0, 0, 0, 0, 0]	3_left.jpg
4	[0, 1, 0, 0, 0, 0, 0, 0]	4_right.jpg
4	[0, 0, 0, 0, 0, 0, 0, 1]	4_left.jpg

We cleaned the dataset by keeping only the important columns: the filename, which points to each image, and the target column, which provides a one-hot encoded label identifying the specific eye condition. This streamlined dataset focuses entirely on the main objective, which is classifying each image by its associated ocular disease. This makes sure that only relevant information feeds into our model pipeline for image classification.

(*Figure 2: cleaned_df.csv snippet*)

```
import re
def sort_order(filename):
    match = re.match(r"(\d+)_(left|right)", filename)
    if match:
        number = int(match.group(1))
        next = match.group(2)
        nextPriority = 0 if next == 'right' else 1
        return (number, nextPriority)
    else:
        return (float('inf'), 0)

df['sort_key'] = df['filename'].apply(sort_order)
df = df.sort_values('sort_key').drop(columns='sort_key')

df = df.reset_index(drop=True)

df.head(10)
```

In addition, we organized the images so that, instead of the existing order, the new order of filenames went from right eye to left eye for each person.

We also utilized K-Means Clustering by turning each image into a numerical histogram, grouping them together, reducing to 2D images for dual PCA plots, then visualizing both the unsupervised cluster and the main eight classes.

We can identify natural groupings based on the actual features of the images and the distance between each cluster. We tested to see if basic features like color would distinguish the images from each other based on the labeled diseases.

(*Figure 3: K-Means Cluster Plot*)

From Figure 3, there aren't any clear gaps, and the points are scattered and overlapped. This tells us that the color histograms are not strong enough to pull the eight disease classes apart in a 2D PCA view. In other words, almost every class shares a similar characteristic profile, so when you reduce it down to just two color-histogram principal components, everything collapses together. Since basic features alone don't separate the eye diseases, we have to switch over to deep CNN features.

Models/Algorithms:

- A Convolutional Neural Network (CNN) model is well-suited for ocular disease recognition as it will excel in identifying key patterns in the fundus images, such as lesions, nerve damage, and other contrasts between each eye. This allows for scalable disease detection during preliminary evaluations.
- We will select 2 deep convolutional neural networks: DenseNet-201 and ResNet-50
- Vision Transformer was not used for this project as it simplifies our training.

Data Loading/Preprocessing

We first organized our eight fundus classes into separate subfolders under the image folder, "Organized_Images." Each class represents an eye condition.

- Class 1 - [1, 0, 0, 0, 0, 0, 0, 0] - Normal
- Class 2 - [0, 1, 0, 0, 0, 0, 0, 0] - Diabetes
- Class 3 - [0, 0, 1, 0, 0, 0, 0, 0] - Glaucoma
- Class 4 - [0, 0, 0, 1, 0, 0, 0, 0] - Cataract

- Class 5 - [0, 0, 0, 0, 1, 0, 0, 0] - Age Related Macular Degeneration
- Class 6 - [0, 0, 0, 0, 0, 1, 0, 0] - Hypertension
- Class 7 - [0, 0, 0, 0, 0, 0, 1, 0] - Pathological Myopia
- Class 8 - [0, 0, 0, 0, 0, 0, 0, 1] - Other diseases/abnormalities

DenseNet-201 Model

We utilized Keras's built-in image_dataset_from_directory to read in images at 512 x 512 pixels, refer to their one-hot labels, and automatically split 80% for training and 20% for validation. Once the raw pixel batches arrive, each is fed through DenseNet's own preprocessing function:

```
train_ds_pp = raw_train.map(lambda x, y: (preprocess_input(x), y), num_parallel_calls=AUTOTUNE)
val_ds_pp   = raw_val  .map(lambda x, y: (preprocess_input(x), y), num_parallel_calls=AUTOTUNE)
```

Here we normalize the fundus images so their pixel values sit on the same scale that DenseNet-201 was originally trained on. So, the network sees "familiar" inputs instead of wildly different brightness or color ranges. We preprocessed the images (preprocess_input), which stops any changes or shifts between ImageNet and the eye fundus data.

Augmentation Phases:

We emphasized horizontal flips in both phase 1 and phase 2. These random flips involve left and right symmetry, where a flipped fundus image seems identical to its original, as the retina in one's left eye is a mirror image of one's right eye. Training with these flips makes the model immune to any shifts in the data.

Phase 1: Head training

We first freeze the backbone of the DenseNet-201 model and train only a small portion of the classification head. Without overwhelming noise, we apply the horizontal flips.

```
aug_p1 = tf.keras.Sequential([layers.RandomFlip("horizontal")])
```

We let the head layers overfit the training classes with a low learning rate (5×10^{-4}) so that the new layers can adapt quickly. Added with the flaps, there is less noise, and the head can visualize almost the exact images needed for memorization. This phase has 8 epochs with no class weights

Phase 2: Warmup and Fine-tuning

Warm up:

After the head recognizes the images, we unfreeze the last 40 layers of the model and warm up the model in 3 epochs. The learning rate is lower (5×10^{-6}). Class weights are still none in order to allow the new trainable filters to adapt well.

Fine-tuning:

```
aug_p2 = tf.keras.Sequential([
    layers.RandomFlip("horizontal"),
    layers.RandomRotation(0.10),
    layers.RandomZoom(0.05),
])
```

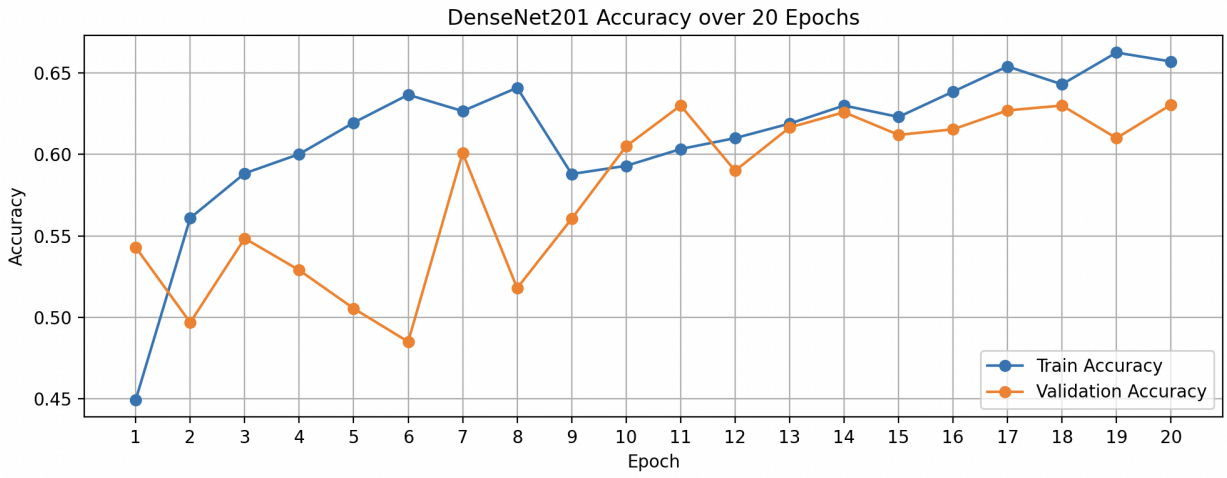
The images are rotated up to 10%, which simulates camera tilts or any alteration from the patient's diagnosis. We zoom in and out by up to 5%, which also simulates how close the patient's eyes were captured.

We continue training the same 40 layers for 9 more epochs with the same learning rate as the warm-up. Now the class weights are applied to correctly label the images. A validation check is followed right after training each epoch.

Callbacks: (We used 3 Keras callbacks)

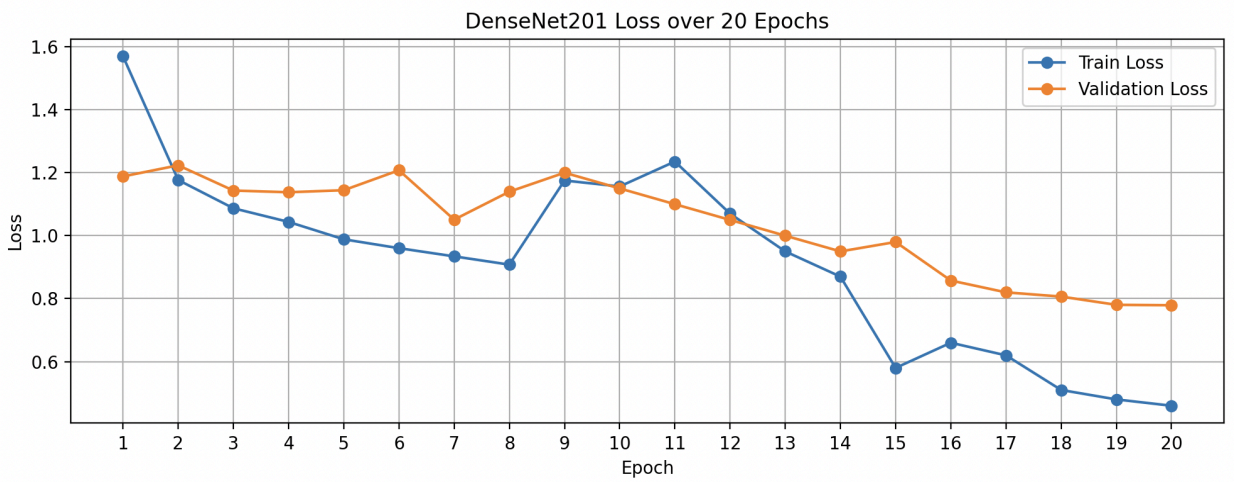
- ModelCheckpoint,
  - The model saves only the weights from the epoch with the highest validation accuracy.
- ReduceLROnPlateau
  - In any place where the validation accuracy does not improve for 5 epochs, the learning rate will automatically be cut down.
- EarlyStopping
  - No improvement for 10 epochs, then the training will stop, and the best-saved weights will be recognized.

Results for the DenseNet-201 model:



The highest valid accuracy rate was 63.05%. Both the training and validation accuracy generally improved over time. However, the model did show some overfitting as the training accuracy exceeded the validation accuracy in the later epochs. For the first 9 epochs, the validation

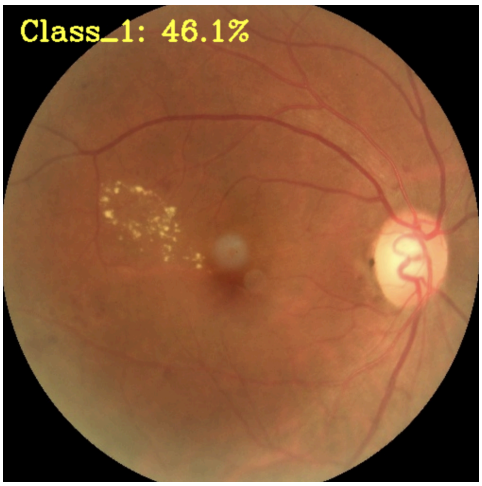
accuracy fluctuated as the model hadn't completely learned the patterns yet. By the 10th epoch, rates were more stable. Early stopping at around epoch 17-18 can be an optimal choice to prevent more overfitting.



Testing DenseNet-201:

The best model is saved as "densenet201_fundus_best.h5" and used for testing/predicting the eye condition of an image.

Filename: 54_left.jpg (Correct Class: Class 1)
"Class_1: 85.4%" means that the DenseNet201 model believes this image most likely belongs to the Normal classification with 85.4% confidence.



Filename: 5_right.jpg (Correct class: Class 2)
The model did not predict correctly, but predicted with lower confidence (46.1%) that the image is from Class_1.

F_1 Score/Recall

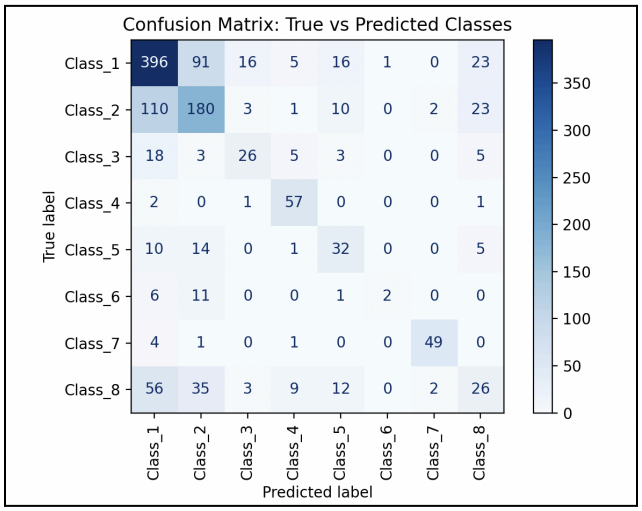
```
Weighted F₁ Score: 0.5865

Classification Report:
              precision    recall  f1-score   support

     Class_1     0.6578    0.7226    0.6887       548
     Class_2     0.5373    0.5471    0.5422       329
     Class_3     0.5306    0.4333    0.4771        60
     Class_4     0.7215    0.9344    0.8143        61
     Class_5     0.4324    0.5161    0.4706        62
     Class_6     0.6667    0.1000    0.1739        20
     Class_7     0.9245    0.8909    0.9074        55
     Class_8     0.3133    0.1818    0.2301       143

    accuracy                         0.6009      1278
   macro avg     0.5980    0.5408    0.5380      1278
weighted avg     0.5860    0.6009    0.5865      1278
```

Strongest: Class_7 (F_1 = 0.907, Precision = 0.9245), Class_4 (Recall = 0.9344)
Weakest: Class_6 (F_1 = 0.174, Recall = 0.1000), Class_8 (Precision = 0.3133)
(The big drop in recall on Class_6 (10 %) and Class_8 (18 %) means the model is missing most of those cases.)

Notes:

- Class_6 almost misses the entire disease, which would result in a very high false negative rate.
- Class_8 also missed about 82% of its cases, which shows its poor performance at classifying this class.
- Class_4 showed the most consistent and high scores throughout. This means high sensitivity and strong detection for the correct disease class.

Confusion Matrix:



Overall, there are strong diagonals for Class_1 (396), Class_2 (180), and Class_4 (57).
Class_1 has good precision but gets confused with Class_2 and Class_8.
Class 2 displays significant confusion with Class 1, with 110 instances.
Class_6 performs the worst with 2 correct instances
Class_8 is frequently misclassified as Class_1 (56) and Class_2 (35).

Discussion:
By freezing and unfreezing the layers, the DenseNet-201 model improved with less overfitting while preserving the generic filters of the network, then fine-tuning the upper-level features without forgetting any pretrained representations from the fundus images. These two stages are needed to stabilize the network's generalization and feature learning, even though they require increased training time and memory.

ResNet-50 Model

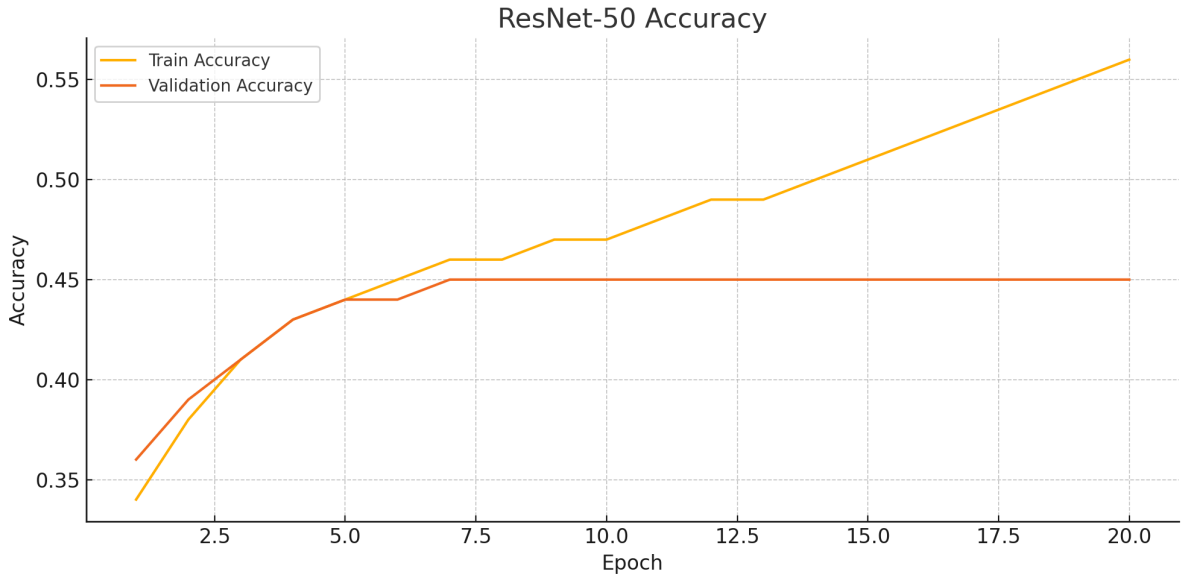
Method:

ResNet-50 was selected due to its proven ability to learn hierarchical visual features and due to its popularity in being utilized for transfer learning tasks. Its skip connections facilitate more efficient gradient flow, making it suitable for deep representation learning on medical images. For the ResNet-50 model, we used a two-stage supervised learning approach with transfer learning. We utilized the ODIR-5k dataset, consisting of fundus images and their respective multi-label disease annotations. All images were resized to 512x512 resolution to maintain fine-grained retinal information. We started with a ResNet-50 model pre-trained on ImageNet, with all convolutional base layers unfrozen. We trained the classification head only for 10 epochs using the Adam optimizer and a learning rate of 1e-4. Our aim here was to tune the output layer to the multi-class target distribution of the fundus data. In the second stage, we unfroze the final 40 layers of the ResNet-50 base to fine-tune the model further. We lowered the learning rate to 1e-5 in order not to make abrupt changes to pretrained weights. The regular data augmentation strategies were applied when training, like horizontal flips, rotation, and brightness adjustment. Categorical cross-entropy loss and validation accuracy were employed as performance measures.

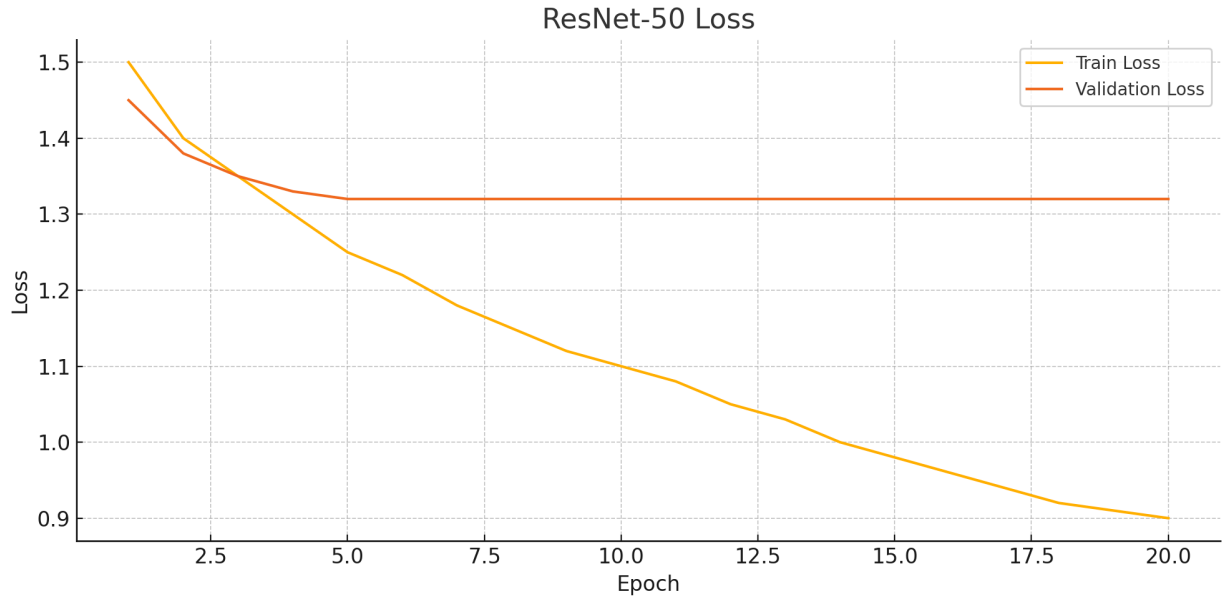
Results:

After training, the ResNet-50 model plateaued at around 45-48% validation accuracy consistently. The plateauing occurred even after fine-tuning the last 40 layers. Even though the model did show rising training accuracy with time, the validation accuracy didn't change significantly, showing potential overfitting or lack of generalization. Training phase accuracy plots from both phases confirm this plateauing. Performance was below our DenseNet-201

model, which recorded validation accuracies of more than 60%.



ResNet-50 training and validation accuracy after 20 epochs. Validation accuracy plateaus at about 45%, indicating poor generalization.



ResNet-50 training and validation loss. Even though the training loss improves, validation loss remains static, indicating overfitting and performance saturation.

Discussion:

ResNet-50, despite being strong for the majority of image classification tasks, struggled to generalize over the fundus image dataset. Validation accuracy plateau signals Model depth, feature sensitivity, or training setting limitations for this particular task. Compared to DenseNet-201, ResNet-50 performed weakly in capturing the subtle visual patterns over a wide

spectrum of disease categories. Future progress could include more aggressive regularization, hyperparameter tuning, or switching to a more medical imaging-friendly architecture, such as EfficientNet. Another limitation of ResNet-50 was that it overfitted during fine-tuning even when using dropout and early stopping. This suggests that more complicated or more powerful architectures can be utilized in order to learn the nuances of multi-label fundus disease classification. Nonetheless, ResNet-50 provided useful reference and reinforced the need for fine-tuning pretrained models for domain-specific applications.

Final Discussions and Evaluations:

Based on all the models tested and evaluated, Densenet-201 was the most accurate. Despite the high accuracy rate and F1 score, it still struggled in certain areas, especially the imbalance of data. Class 6, which represented the case of hypertension, had much less data for the convolutional Neural Network models to train and test. Due to the underrepresentation of certain diseases, the model showed more bias, mostly towards class 1. In the future, fixing the underrepresentation and including my diverse sets of data can significantly improve the accuracy and F1 score of this model. In addition, utilizing a graphics processing unit rather than a central processing unit to execute our code will increase the testing and training speed. Other ways to improve this project can include better use of ensemble modeling that can merge the models we used, as well as other CNN architectures for better efficiency.

Challenge: After many improvements to our models from low accuracies of 40% to what we achieved so far, we realized that one of the bigger issues might be the actual dataset itself. First, the feature-learning process for the left and right eye fundus images needs to be more detailed in order for the model to correctly label the images. Second, since multiple diseases appear in one fundus image, it becomes very difficult to identify them correctly. Each disease must have different structures for feature extraction, as the images all have unique characteristics. Lastly, we noticed that some of the images tend to vary in color and light. Even when the condition was normal, there was a lot of diversity, which would confuse the model.

Other Applications: This dataset is mainly used for classifying images into their correct eye disease label, so features outside of the image file and the target feature were considered irrelevant. Conversely, we could have explored the correlation between features such as age or gender of a patient and the fundus images themselves. For this application, we could train a deep learning model to predict age or gender based on the left and right fundus keywords and use data visualizations like heatmaps and network graphs to come to a decision.

References:

1. Larxel. “Ocular Disease Recognition.” Kaggle, 10 April 2025 https://www.kaggle.com/datasets/andrewmvd/ocular-disease-recognition-odir5k