

## Data Collection and Preprocessing Phase

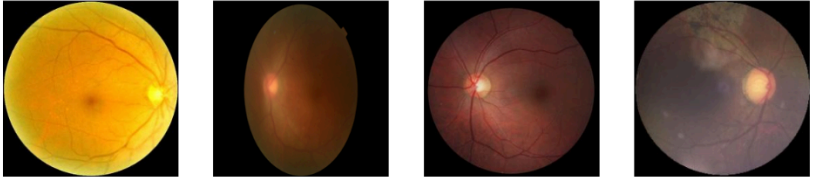
Date	17 April 2024
Team ID	738184
Project Title	Eye Disease Detection using Deep Learning
Maximum Marks	6 Marks

### Preprocessing

The images will be preprocessed by resizing, normalizing, augmenting, denoising, adjusting contrast, detecting edges, converting color space, cropping, batch normalization, and whitening data. These steps will enhance data quality, promote model generalization, and improve convergence during neural network training, ensuring robust and efficient performance across various computer vision tasks.

Section	Description
Data Overview	<p>Eye Disease Detection dataset contains 4 sub-folders each with 4 eye diseases like cataract, glaucoma, diabetic_rethinopathy, and normal. It contains a total of 4217 images.</p> <p>Each Eye Disease Folder Contains:</p> <p>Cataract Folder: 1038 Images</p> <p>Glaucoma Folder: 1007 Images</p> <p>Diabetic_Rethinopathy Folder: 1098 Images</p> <p>Normal Folder: 1074 Images</p>
Resizing	Each disease folder had images that were in different sizes. We have resized all the images into a size of 256x256.
Normalization	Normalized and rescaled all the images using ImageDataGenerator library from tensorflow.
Data Augmentation	Performed techniques like shearing, zooming and flipping using ImageDataGenerator library from tensorflow.
<b>Data Preprocessing Code Screenshots</b>	

<p>Loading Data</p>	<ul style="list-style-type: none"> <li>Loading the necessary libraries</li> </ul> <pre>[1] import tensorflow as tf</pre> <ul style="list-style-type: none"> <li>Creating the kaggle directory</li> </ul> <pre>[2] !mkdir -p ~/.kaggle     !cp kaggle.json ~/.kaggle</pre> <ul style="list-style-type: none"> <li>Loading Dataset into Collab using Kaggle API</li> </ul> <pre>[3] !kaggle datasets download -d gunavenkatdoddi/eye-diseases-classification</pre> <p>Warning: Your Kaggle API key is readable by other users on this system! To fix ·          Downloading eye-diseases-classification.zip to /content          99% 727M/736M [00:06&lt;00:00, 129MB/s]          100% 736M/736M [00:06&lt;00:00, 117MB/s]</p>  <ul style="list-style-type: none"> <li>Unzip the downloaded file</li> </ul> <pre>[4] !unzip 'eye-diseases-classification.zip'</pre> <p>Archive: eye-diseases-classification.zip          inflating: dataset/cataract/0_left.jpg          inflating: dataset/cataract/103_left.jpg          inflating: dataset/cataract/1062_right.jpg          inflating: dataset/cataract/1083_left.jpg</p>
<p>Resizing</p>	<p>Image Pre-Processing Steps</p> <p>1. Image Resizing</p> <pre>import os from PIL import Image  def resize_images(input_dirs, output_dir, target_size):     os.makedirs(output_dir, exist_ok=True) # Create output directory if not exists      for input_dir in input_dirs:         category_name = os.path.basename(input_dir)          for filename in os.listdir(input_dir):             if filename.endswith('.jpg') or filename.endswith('.png') or filename.endswith('.jpeg'):                 img_path = os.path.join(input_dir, filename)                 print(f"Processing image: {img_path}")</pre>

	<pre> try:     img = Image.open(img_path)     resized_img = img.resize(target_size, Image.ANTIALIAS)      output_filename = f"resized_{filename}" # New filename for resized image     output_path = os.path.join(output_dir, category_name, output_filename)     os.makedirs(os.path.dirname(output_path), exist_ok=True)     resized_img.save(output_path)      print(f"Resized and saved: {filename} -&gt; {output_path}") except Exception as e:     print(f"Error processing {filename}: {e}")  cataract='/content/dataset/cataract' diabetic='/content/dataset/diabetic_retinopathy' glaucoma='/content/dataset/glaucoma' normal='/content/dataset/normal' input_dirs = [cataract,diabetic,glaucoma,normal] output_directory = '/content/dataset/resized_img' target_image_size = (256, 256) resize_images(input_dirs, output_directory, target_image_size) </pre> <div> <div>Normal</div> <div>Cataract</div> <div>Diabetic_retinopathy</div> <div>Glaucoma</div> </div> 
Normalization	<p>Normalization and Image data Generator</p> <pre> ] from tensorflow.keras.preprocessing.image import ImageDataGenerator  train_datagen=ImageDataGenerator(rescale=1./255,                                 shear_range=0.2,zoom_range=0.2,horizontal_flip=True) test_datagen=ImageDataGenerator(rescale=1./255) </pre>
Data Augmentation	<p>Normalization and Image data Generator</p> <pre> ] from tensorflow.keras.preprocessing.image import ImageDataGenerator  train_datagen=ImageDataGenerator(rescale=1./255,                                 shear_range=0.2,zoom_range=0.2,horizontal_flip=True) test_datagen=ImageDataGenerator(rescale=1./255) </pre> <pre> [14] x_train=train_datagen.flow_from_directory(r'/content/dataset/resized_img/train',  target_size=(64,64),batch_size = 32,class_mode='categorical')  x_test=test_datagen.flow_from_directory(r'/content/dataset/resized_img/test',  target_size=(64,64),batch_size=32,class_mode='categorical') </pre> <p>Found 3376 images belonging to 4 classes. Found 841 images belonging to 4 classes.</p>