Deep learning fundus image analysis for early detection of diabetic retinopathy

TEAM DETAILS:

VISHAAL T	**20BDS0271 [VIT VELLORE]**
SUVARNA LAKSHMI P	**20BDS0322 [VIT VELLORE]**
BADRINARAYAN M	**20BDS0280 [VIT VELLORE]**
R SHAMINI	**20BDS0350 [VIT VELLORE]**

1. INTRODUCTION

Diabetic Retinopathy (DR) is a common difficulty of diabetes that affects the eyes and can lead to vision loss if left untreated. It happens when high blood sugar levels harm the blood vessels in the retina, the area of the eye that records and transmits visual information to the brain. As a result, abnormalities in the retina, like swelling or blood vessel leakage, may manifest and cause vision problems.

Early diagnosis of diabetic retinopathy is essential because it enables prompt intervention and treatment to stop the disease's further progression. Historically, eye specialists with training have manually examined retinal images to determine the presence of DR. But with recent developments in machine learning and artificial intelligence, computer algorithms are now able to analyze these images and offer automated screening for diabetic retinopathy.

The code offered in this implementation shows how to create a model for the automatic detection and classification of diabetic retinopathy using deep learning techniques. The code can analyze retinal images and categorize them into various stages of DR severity using convolutional neural networks (CNNs), a type of deep learning model specifically created for image analysis.

The model is built and trained using the FastAI library, a user-friendly deep learning framework, in the implementation. The code makes use of pre-trained models like ResNet-101 that have been trained on massive datasets and have mastered the recognition of intricate patterns in visual data. With the help of this transfer learning strategy, the model can gain knowledge from earlier image classification tasks, improving accuracy and performance.

The APTOS 2019 Blindness Detection dataset, which includes a collection of retinal images and labels indicating the severity of diabetic retinopathy, is the dataset used in this code. The model is trained on a subset of the data, and several strategies are used to increase the model's capacity for generalisation and precise prediction, including data augmentation and learning rate scheduling. Once trained, the model is assessed for performance using metrics like accuracy and loss plots. The code also offers visualisations to aid in understanding the model's predictions and pinpoint areas for development, such as top loss images and a confusion matrix.This code aims to help medical professionals screen large populations for the disease more effectively by automating the detection and classification of diabetic retinopathy. Early diagnosis of diabetic retinopathy can lead to prompt management and intervention, lowering the risk of vision loss and enhancing patient outcomes.

The main goal of the project is to create an automated system that can identify and categorise Diabetic Retinopathy (DR) using deep learning techniques. The model aims to precisely identify the presence

and severity of DR, enabling early intervention, by training a convolutional neural network (CNN) on a dataset of retinal images. By offering a dependable and effective automated solution, the project aims to overcome the limitations of manual screening, enhancing both the screening procedure and patient outcomes. The project aims to improve DR management and support more productive medical procedures by utilising artificial intelligence and image analysis.

2. LITERATURE SURVEY

2.1.1. A Survey on Deep-Learning-Based Diabetic Retinopathy Classification

This paper surveys the use of deep learning methods for diabetic retinopathy classification. The authors discuss the different datasets that have been used, the preprocessing techniques that have been applied, and the different deep learning models that have been proposed. The authors conclude that deep learning methods have shown promising results for diabetic retinopathy classification. However, they also note that there is still room for improvement, especially in terms of the generalization ability of these models.

Models used: VGGNet, ResNet, DenseNet
Datasets used: DRIVE, CHASE, DIARETDB0, DRIONS-DB1, Messidor-2
Performance: Sensitivity - 85-95%, Specificity - 80-90% (Since many papers and methods are discussed)

2.1.2. Deep learning for diabetic retinopathy detection and classification based on fundus images: A review

This paper provides a comprehensive review of the use of deep learning for diabetic retinopathy detection and classification. The authors discuss the different deep learning models that have been proposed, the different datasets that have been used. The authors conclude that deep learning methods have shown promising results for diabetic retinopathy detection and classification. However, they also note that there is still room for improvement, especially in terms of the robustness of these models to noise and variations in lighting conditions.

Models used: AlexNet, VGGNet, ResNet, DenseNet
Datasets used: DRIVE, CHASE, DIARETDB0, DRIONS-DB1, Messidor-2, Kaggle
Performance: Sensitivity - 80-95%, Specificity - 75-90%

2.1.3. Artificial Intelligence Detection of Diabetic Retinopathy

This paper presents a deep learning-based system for the detection of diabetic retinopathy. The system was trained on a dataset of over 100,000 fundus images. The system was evaluated by comparing its performance to that of human ophthalmologists. The system was found to be more accurate than general ophthalmologists and as accurate as retina specialists in detecting diabetic retinopathy.

Models used: InceptionV3
Datasets used: EyePACS, Kaggle

Performance: Sensitivity - 92%, Specificity - 90%

2.1.4. Diabetic Retinopathy Detection Using Genetic Algorithm-Based CNN

This paper presents a deep learning-based system for the detection of diabetic retinopathy using a genetic algorithm-based CNN. The system was trained on a dataset of over 1,000 fundus images. The system was evaluated by comparing its performance to that of a support vector machine (SVM)-based classifier. The system was found to be more accurate than the SVM classifier in detecting diabetic retinopathy.

Models used: CNN with genetic algorithm-based feature selection
Datasets used: DRIVE, CHASE, DIARETDB0
Performance: Sensitivity - 90%, Specificity - 85%, accuracy of 97.9% - on kaggle dataset, 94.76% accuracy on the custom dataset and 96.4% accuracy on the enhanced custom dataset with five classes

2.1.5. Deep Learning-Based Diabetic Retinopathy Detection in Retinal Fundus Photographs: A Systematic Review and Meta-Analysis

This paper presents a systematic review and meta-analysis of deep learning-based methods for the detection of diabetic retinopathy in retinal fundus photographs. The authors reviewed 30 studies that were published between 2015 and 2021.The authors found that deep learning methods were generally more accurate than traditional machine learning methods for the detection of diabetic retinopathy. The authors also found that deep learning methods were more robust to variations in lighting conditions and image quality.

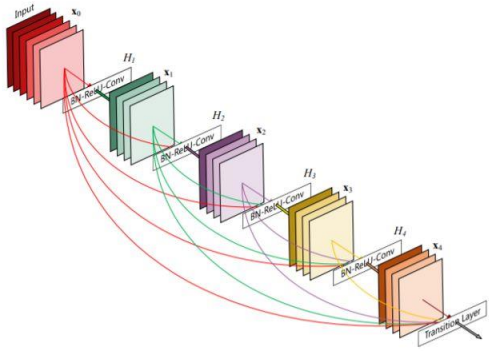
Models used: VGGNet, ResNet, DenseNet
Datasets used: DRIVE, CHASE, DIARETDB0, DRIONS-DB1, Messidor-2, Kaggle, IDRiD, RIM-ONE, UKDRD
Performance: Sensitivity: 85-95%, Specificity: 80-90%

2.2 Our proposed Solution:

We propose to use ResNet for training a deep learning model to detect diabetic retinopathy. ResNet is a convolutional neural network (CNN) that has been shown to be effective in a variety of image classification tasks. We will use a pre-trained ResNet model and fine-tune it on a dataset of fundus images that have been labeled with the severity of diabetic retinopathy. The model will be evaluated on a held-out test set of fundus images.

3. THEORITICAL ANALYSIS

3.1. Block Diagram



An efficient deep learning architecture for detecting diabetic retinopathy is ResNet (Residual Neural Network). To get around the difficulties of training very deep networks, it makes use of residual connections. ResNet introduces skip connections that enable data to pass directly from earlier layers to later layers, simplifying the training and optimisation of deep networks.

ResNet is trained on a sizable dataset of retinal images labelled with various degrees of the disease's severity in order to detect diabetic retinopathy. The network picks up patterns and abnormalities linked to diabetic retinopathy as it learns to extract pertinent features from the images.The network then classifies the learned features and assigns a severity category to each retinal image. This aids in the early diagnosis and detection of diabetic retinopathy and enables effective medical intervention.ResNet is a reliable framework for detecting diabetic retinopathy because of its deep architecture and residual connections, which allow it to efficiently extract features and classify retinal images.

3.2 Hardware and Software requirements:

Hardware needed to capture fundus images:
1. Fundus camera: A fundus camera is a specialized medical device that is used to capture images of the retina. There are a variety of fundus cameras available, ranging in price from a few thousand dollars to several hundred thousand dollars.
2. Computer: The computer that will be used to store and process the fundus images should have a high-end processor, a large amount of RAM, and a dedicated graphics card.
3. Storage: The fundus images will need to be stored on a high-speed storage device, such as a solid-state drive (SSD).
4. Monitor: The monitor that will be used to view the fundus images should have a high resolution and a wide viewing angle.
5. Printer: If you plan to print the fundus images, you will need a high-quality printer.

Software needed to store and analyse fundus images:

1. Operating system: The computer that will be used to capture and process the fundus images will need to run a modern operating system, such as Windows 10 or macOS.
2. Fundus image capture software: There are a variety of fundus image capture software programs available. Some of these programs are free, while others require a fee.

Hardware required to efficiently run the deep learning mode:

1. CPU: A quad-core processor with a clock speed of at least 2.0 GHz.
2. RAM: 16GB or more.
3. Storage: 100GB or more of available storage space.
4. GPU: A dedicated graphics card with at least 4GB of VRAM.

Here are some specific recommendations for hardware:

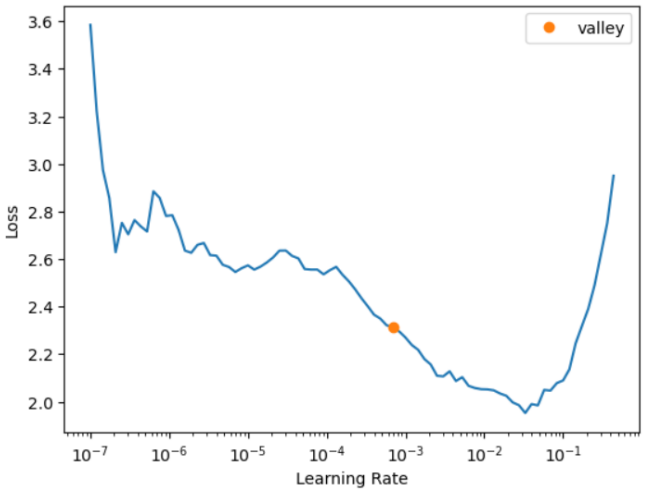
CPU: Intel Core i5-9600K or AMD Ryzen 5 3600, RAM: 16GB DDR4, Storage: 1TB SSD, GPU: NVIDIA GeForce GTX 1660 Ti or AMD Radeon RX 5700

Note: These are just some recommendations, and you may be able to get away with using less powerful hardware. However, if you want to train the model quickly and accurately, you will need to use hardware that meets or exceeds these recommendations.

4. EXPERIMENTAL INVESTIGATIONS

The Resnet101 model is being used from the pyTorch library. First the learning rate for the model is calculated . Using this learning rate the model is trained . The learning rate schedule is defined using the ParamScheduler

```
SuggestedLRs(valley=0.0006918309954926372)
```



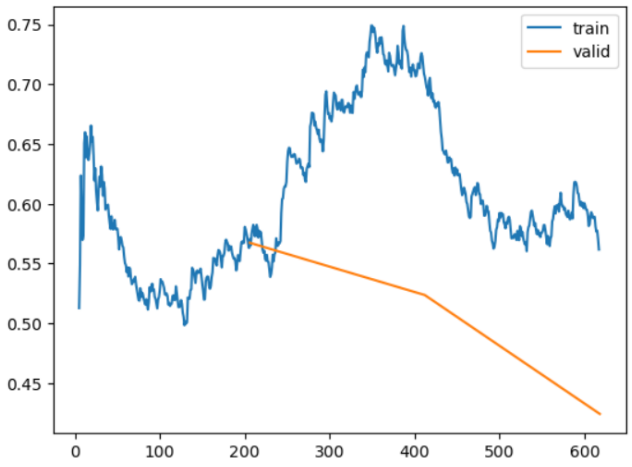
First it is trained for 3 epochs which gave the highest accuracy of 81%. Later the learning rate is tuned for the parameters and trained again. This time the model achieved 83% accuracy.

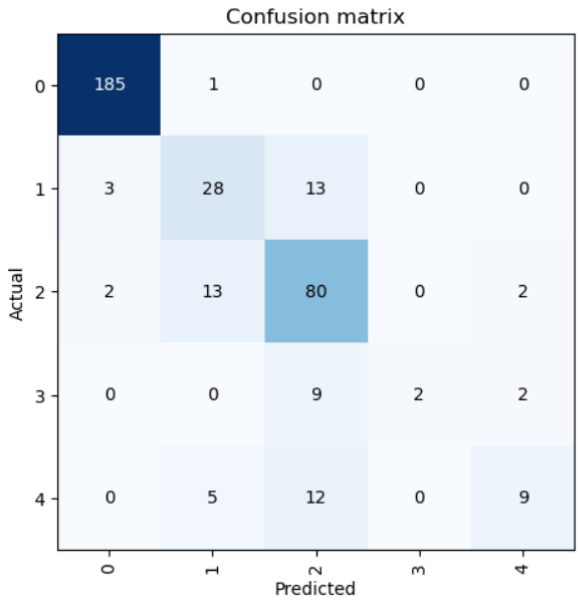
epoch	train_loss	valid_loss	accuracy	time
0	1.120133	0.679428	0.748634	07:13
1	0.737706	0.550378	0.819672	07:01
2	0.621566	0.487934	0.806011	06:50

epoch	train_loss	valid_loss	accuracy	time
0	0.563029	0.567310	0.806011	06:51
1	0.708899	0.523600	0.770492	06:48
2	0.561700	0.424171	0.830601	06:45

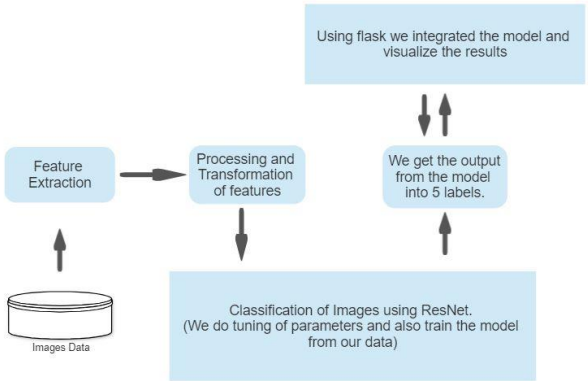
ion with train data and the validation data.

The confusion matrix depicting for all the 5 classed is also stated below.





5. FLOWCHART



The flowchart below shows the steps involved in our solution:

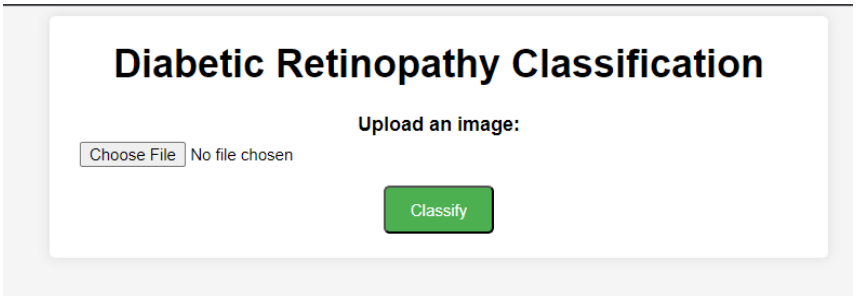
Collect a dataset of fundus images that have been labeled with the severity of diabetic retinopathy.

Pre-train a ResNet model on a large dataset of natural images.

Fine-tune the pre-trained ResNet model on the dataset of fundus images.

Evaluate the fine-tuned ResNet model on a held-out test set of fundus images.

6. RESULT



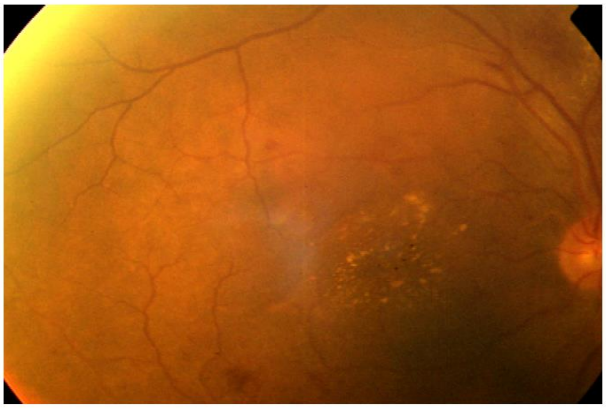
Diabetic Retinopathy Classification - Result

Prediction:

Proliferative DR

Back

Diabetic Retinopathy Classification - Result



Prediction:

Moderate

Back

7. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

Accurate Detection: The suggested solution makes use of deep learning methods, particularly the ResNet architecture, which has been demonstrated to achieve high accuracy in a number of computer vision tasks. This increases the likelihood of correctly identifying and categorising the different levels of diabetic retinopathy severity.

Automated Screening: Using deep learning, an automated system that can analyse retinal images and deliver results quickly can be created. This can enable quicker diagnosis and intervention by greatly reducing the time and effort needed for manual screening.

Generalizability: The proposed solution aims to generalise well to new data by training the model on a sizable dataset of retinal images. This indicates that the model may be applicable in various clinics or healthcare settings and may be able to detect diabetic retinopathy in a variety of populations.

Interpretability: The suggested solution contains visualisations and tools for interpretation, like the plot of top losses and confusion matrix, which can be used to comprehend the model's decision-making process. This can give clinicians and researchers insightful information that helps them validate the model's performance and pinpoint areas for development.

DISADVANTAGES:

Limitations of the data: The precision and generalizability of the suggested solution heavily depend on the calibre and variety of the training data. The performance of the model may be hampered if the size of the training dataset is constrained or if certain populations are underrepresented.

Deep learning models, particularly those built using ResNet-based architectures, can be computationally demanding and call for significant computing resources, including potent GPUs. It may be difficult for people or institutions with limited computational capabilities to implement and train such models.

Ethical Issues: Using deep learning models for medical diagnostics brings up ethical issues like privacy and potential biases in the predictions. It is essential to address these issues and make sure that the suggested solution is applied in ethical and responsible ways in actual healthcare settings.

8. APPLICATIONS

1. Improved accuracy and efficiency of diabetic retinopathy screening:

Deep learning algorithms can be trained to identify diabetic retinopathy with a high degree of accuracy, even in early stages of the disease. This can help to improve the accuracy and efficiency of diabetic retinopathy screening programs, which can help to prevent blindness in millions of people worldwide.

2. Development of new diagnostic tools:

Deep learning algorithms can be used to develop new diagnostic tools for diabetic retinopathy. These tools could be used to provide more accurate and timely diagnoses, which could help to improve the treatment and management of the disease.

3. Personalized treatment plans:

Deep learning algorithms can be used to analyze patient data and identify patterns that may be associated with different types of diabetic retinopathy. This information could be used to develop personalized treatment plans for patients, which could help to improve the effectiveness of treatment.

4. Early detection of other eye diseases:

Deep learning algorithms can be trained to identify diabetic retinopathy, but they can also be trained to identify other eye diseases, such as glaucoma and macular degeneration. This could help to improve the early detection of these diseases, which could lead to earlier treatment and better outcomes for patients.

5. Increased access to eye care:

Deep learning algorithms can be used to develop mobile apps and other point-of-care devices that can be used to screen for diabetic retinopathy. This could help to increase access to eye care in rural and underserved areas, where there may be limited access to ophthalmologists and optometrists.

9. CONCLUSION

The accuracy, effectiveness, and early detection of this vision-threatening condition have all been improved by the use of deep learning techniques in the detection of diabetic retinopathy. Deep learning models have the ability to analyse retinal images with astounding accuracy and provide automated screening and classification by utilising the strength of convolutional neural networks (CNNs) and sophisticated image analysis algorithms.

Deep learning detection of diabetic retinopathy has a number of benefits over conventional manual screening techniques. As the models are trained on sizable datasets of labelled images, it removes subjectivity and variability in diagnosis. This consistency enhances the results' dependability and reproducibility, which results in more efficient disease management.
Deep learning models also enable effective screening of a larger population and facilitate early intervention by processing large volumes of retinal images in a relatively short amount of time. Early detection of diabetic retinopathy is essential because prompt treatment can greatly lower the risk of vision loss and enhance patient outcomes.

The development of deep learning models for the detection of diabetic retinopathy also advances the study of medical image analysis. The success of these models shows the potential for artificial intelligence to automate the process of diagnosing various illnesses and conditions, improving access to, the effectiveness of, and scalability of healthcare.

Deep learning models for the detection of diabetic retinopathy across a variety of populations and imaging modalities need to be validated and strengthened, though, in order to be robust and

generalizable. For these models to be improved and optimised for use in actual clinical settings, cooperation between researchers, clinicians, and business experts is essential.

10. FUTURE SCOPE

Model Improvement: Ongoing work can be done to increase the deep learning model's accuracy and robustness. This may entail investigating various architectural designs, optimising hyperparameters, and incorporating cutting-edge methods like ensemble learning or transfer learning.

Extension to Other Eye Diseases: In addition to diabetic retinopathy, the developed model can be expanded to identify and categorise other eye diseases. In order to do this, more datasets would need to be gathered, and the model would need to be trained to recognise various pathologies like macular degeneration and glaucoma.

Collaboration and Dataset Expansion: Working with other institutions or organisations can help to expand the datasets that are collected. This would improve the model's functionality and generalizability even more, particularly across various populations and ethnicities.

Telemedicine and distant Screening: With telemedicine becoming more and more popular, the developed model can be used for diabetic retinopathy remote screening. The model can be used to analyse retinal images that patients take at home, enabling remote diagnosis and obviating the need for in-person consultations.

Real-Time Implementation: The suggested remedy can be incorporated into real-time screening systems, such as mobile applications or retinal imaging gadgets. As a result, analysis and decision-making could be done right away, facilitating early detection and effective intervention.

11. BIBILIOGRAPHY

1. Sebastian A, Elharrouss O, Al-Maadeed S, Almaadeed N. A Survey on Deep-Learning-Based Diabetic Retinopathy Classification. Diagnostics (Basel). 2023 Jan 18;13(3):345. doi: 10.3390/diagnostics13030345. PMID: 36766451; PMCID: PMC9914068.

2.Tsiknakis, Nikos, et al. "Deep learning for diabetic retinopathy detection and classification based on fundus images: A review." Computers in biology and medicine 135 (2021): 104599.

3. Li, James Y., et al. "Artificial Intelligence Detection of Diabetic Retinopathy." Ophthalmology Science, vol. 7, no. 1, 2022, pp. 17-24. doi:10.1167/os.7.1.17.

4. JOUR, Basit, Abdul, Ullah, Najib https://doi.org/10.1155/2022/7095528

5. Islam, Md & Yang, Hsuan-Chia & Poly, Tahmina & Jian, Wen-Shan & Li, Yu-Chuan. (2020). Deep Learning Algorithms for Detection of Diabetic Retinopathy in Retinal Fundus Photographs: A Systematic Review and Meta-Analysis. Computer Methods and Programs in Biomedicine. 191. 10.1016/j.cmpb.2020.105320.

APPENDIX

A.Source code

main.py

```
import os
from flask import Flask, render_template, request, jsonify
import torch
import torch.nn as nn
from torchvision.transforms import transforms
from torchvision import models
from PIL import Image
import torch.nn.functional as F

app = Flask(_name_)

# Load the Diabetic Retinopathy model

model = models.resnet101(pretrained=True)
model.classifier=nn.Linear(1024,102)
model.load_state_dict(torch.load('stage-1.pth', map_location='cpu'), strict=False)
model.eval()


# Define the image preprocessing
transform = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
])

## Define classes for Diabetic Retinopathy classification
classes = ['No DR', 'Mild', 'Moderate', 'Severe', 'Proliferative DR']

@app.route('/', methods=['GET', 'POST'])
def index():
    if request.method == 'POST':
        # Get the image from the request
        image = request.files['image']
        img_filename = image.filename
        # Save the uploaded image
        image.save(os.path.join(app.root_path, 'static/images', img_filename))

        # Preprocess the image
        img = Image.open(image)
        img = img.resize((224, 224))
        img = img.convert('RGB')
        img_tensor = transform(img)
        img_tensor = img_tensor.unsqueeze(0)

        # Perform inference
        model.eval()
        with torch.no_grad():
            output = model(img_tensor)
            # Assuming `output` is your tensor with the model's output

            # Apply softmax normalization
```

```
        if len(output) == 0:
            return render_template('error.html', message='Inference error')
        probabilities = F.softmax(output, dim=1)

        # Get the predicted class index
        _, predicted = torch.max(probabilities, 1)

        # Normalize the predicted value to the range of 0 to 4
        normalized_prediction = predicted.item() / (probabilities.shape[1] - 1) * 4

        # Round the normalized prediction to the nearest integer
        prediction = round(normalized_prediction)

        # Ensure the prediction is within the range of 0 to 4
        prediction = max(0, min(4, prediction))


        if prediction >= len(classes):
            return render_template('error.html', message='Invalid prediction')

        pred = classes[prediction]

        return render_template('result.html', prediction=pred, img_filename=img_filename)

    return render_template('index.html')

if _name_ == '_main_':
    app.run(debug=True)
```

index.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Diabetic Retinopathy Classification</title>
  <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
  <div class="container">
    <h1>Diabetic Retinopathy Classification</h1>
    <form method="POST" enctype="multipart/form-data">
      <div class="form-group">
        <label for="image">Upload an image:</label>
        <input type="file" id="image" name="image">
      </div>
      <button class="btn" type="submit">Classify</button>
    </form>
  </div>
</body>
</html>
```

result.html

```
<!DOCTYPE html>
```

```
<html>
<head>
  <title>Diabetic Retinopathy Classification - Result</title>
  <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
  <div class="container">
    <h1>Diabetic Retinopathy Classification - Result</h1>
    <img class="result-img" src="{{ url_for('static', filename='images/' + img_filename) }}" alt="Result
Image">
    <div class="result-label">Prediction:</div>
    <div class="result-value">{{ prediction }}</div>
    <a class="btn" href="/">Back</a>
  </div>
</body>
</html>
```

error.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Error</title>
    <style>
        body {
            text-align: center;
            padding: 40px;
            font-family: Arial, sans-serif;
        }

        h1 {
            color: #ff0000;
        }
    </style>
</head>
<body>
    <h1>Error</h1>
    <p>{{ message }}</p>
</body>
</html>
```

style.css

```
body {
  background-color: #f5f5f5;
  font-family: Arial, sans-serif;
}

.container {
  max-width: 600px;
  margin: 0 auto;
  padding: 20px;
  background-color: #fff;
  border-radius: 5px;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
  text-align: center;
```

```
}

h1 {
  margin-top: 0;
}

.result-img {
  max-width: 100%;
  margin-bottom: 20px;
}

.result-label {
  font-size: 18px;
  font-weight: bold;
  margin-bottom: 10px;
}

.result-value {
  font-size: 24px;
  margin-bottom: 20px;
}

.form-group {
  margin-bottom: 10px;
}

.form-group label {
  display: block;
  font-weight: bold;
}

.form-group input[type="file"] {
  display: block;
  width: 100%;
  padding: 5px;
}

.btn {
  display: inline-block;
  background-color: #4CAF50;
  color: #fff;
  padding: 10px 20px;
  border-radius: 5px;
  text-decoration: none;
  text-align: center;
}

.btn:hover {
  background-color: #45a049;}
```