Deep Learning Fundus Image Analysis for Early Detection of Diabetic Retinopathy

---

Index

---

1. Introduction & Objective

A. Introduction

Diabetic Retinopathy (DR) is a serious complication of diabetes that affects the retina and can lead to vision loss if not detected early. Manual screening of retinal images requires specialized medical expertise and is time-consuming. With advancements in Deep Learning and Computer Vision, automated detection systems can assist in early diagnosis and large-scale screening.

This project presents a Deep Learning-based web application that detects and classifies diabetic retinopathy severity using fundus retinal images. The system utilizes the Xception convolutional neural network architecture through transfer learning and provides predictions via a user-friendly web interface built using Flask.

B. Objectives

- To develop an automated DR detection system using Deep Learning
- To classify retinal images into five severity levels
- To implement transfer learning using Xception
- To build a web-based application for image upload and prediction
- To integrate user authentication using Firebase
- To display severity levels using color-coded visualization

---

2. Technology Stack & Dependencies

A. Programming Language

a. Python

B. Deep Learning Framework

a. TensorFlow
b. Keras

C. Model Architecture

a. Xception (Pre-trained on ImageNet)

D. Web Framework

a. Flask

E. Frontend

a. HTML
b. CSS
c. JavaScript

F. Database

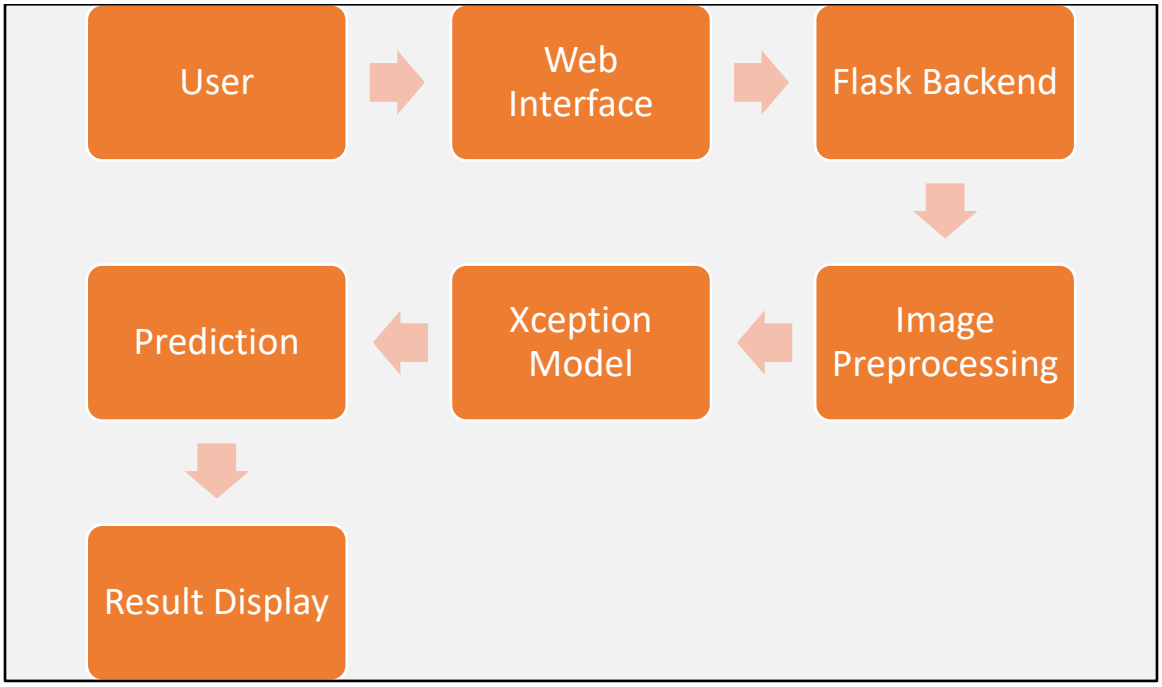
a. Firebase Realtime Database

G. Key Python Libraries

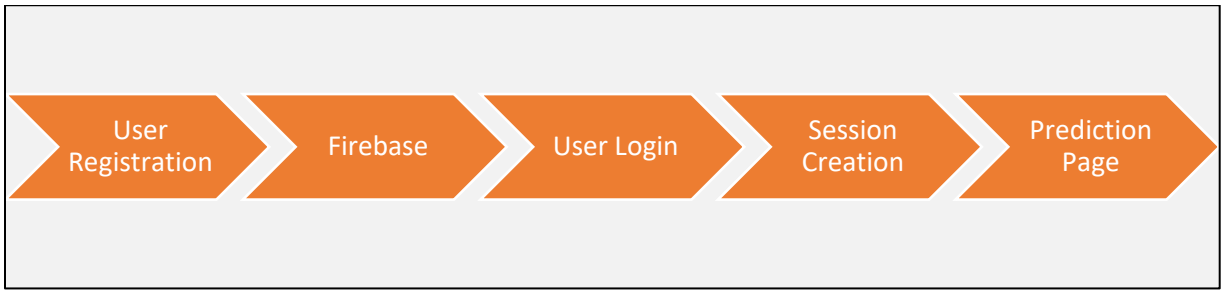
a. Numpy
b. Pandas
c. Matplotlib
d. Tensorflow
e. Keras
f. Flask
g. pyrebase

3. System Architecture & Workflow

A. System Architecture



B. Authentication Flow



C. Workflow

- User logs in through the web interface.
- User uploads a retinal image.
- Image is sent to Flask backend.
- Image is resized and preprocessed.
- Processed image is fed into trained Xception model.
- Model outputs probability scores.
- Highest probability class is selected.
- Result is displayed with severity color coding and confidence score.

4. Key Components & Code Explanation

A. Data Collection

- The dataset used for this project was collected from Kaggle. The dataset contains labeled fundus retinal images categorized into different stages of Diabetic Retinopathy.
- Source Platform
  Kaggle
- Dataset Type
  Image Dataset
- Number of Classes
  5 Classes
  - 0 – No Diabetic Retinopathy
  - 1 – Mild DR
  - 2 – Moderate DR
  - 3 – Severe DR
  - 4 – Proliferative DR
- The dataset includes high-resolution retinal images that capture blood vessels and pathological changes in the retina. The dataset was collected from Kaggle, stored in Google Drive for persistence, and accessed in Google Colab for model training.

B. Data Preprocessing

- Images are resized to 299×299 pixels to match Xception input requirements.
- Xception's preprocess_input function is applied to normalize pixel values.
- Data augmentation is performed using ImageDataGenerator to improve generalization.
- Example:
  - Image resizing
  - Pixel normalization
  - Batch generation

C. Model Architecture (Xception)

- The model uses Xception as a base feature extractor. The top classification layers are removed and replaced with custom Dense layers.
- Structure:
  - Input Layer (299×299×3)
  - Xception Base Model (pre-trained, frozen layers)
  - Global Average Pooling Layer
  - Dense Layer (ReLU activation)
  - Output Layer (Softmax activation – 5 classes)
- Loss Function
  - Categorical Crossentropy
- Optimizer
  - Adam

D. Model Training & Evaluation

- Model Training
  - Transfer learning approach is used.
  - Base Xception layers are frozen.
  - Custom classifier layers are trained.
- Model Evaluation
  - Accuracy
  - Loss curves
  - Confusion Matrix
  - Confidence Score

E. Model Loading

- The trained model is saved as .h5 file and loaded in Flask application.
- Example:
  - model = load_model("Updated-Xception-diabetic-retinopathy.h5")

F. Prediction Functions

- Uploaded image is:
  - Resized to 299×299
  - Converted to array
  - Expanded dimension
  - Preprocessed using preprocess_input
  - Passed to model.predict()
- Predicted class is determined using:
  - np.argmax(prediction)
  - Confidence score is calculated using:
  - np.max(prediction) × 100

G. User Interface (File Upload)

- Frontend features:
  - Drag and Drop Image Upload
  - Browse File Button
  - Image Preview
  - Analyze Button
  - Re-upload Button
- Uploaded image is sent using:
  - enctype="multipart/form-data"
  - Flask receives image using:
  - request.files["file"]

H. Main Application Logic

- Login System
  - Validates email and password from Firebase
    Creates user session
- Prediction Route
  - Receives image
  - Processes image
  - Predicts class
  - Renders result with severity color coding
- Session Handling
  - Prevents unauthorized access to prediction page

5. How to Run the Application

- Step 1
  Install required libraries:
  pip install tensorflow flask numpy pyrebase4
- Step 2
  Place trained model file in project directory.
- Step 3
  Run Flask application:
  python app.py
- Step 4
  Open browser:
  http://127.0.0.1:5000
- Step 5
  Login and upload retinal image for prediction.

---

6. Output & Results

- The system classifies images into five categories:
  - No DR
  - Mild DR
  - Moderate DR
  - Severe DR
  - Proliferative DR
- The result page displays:
  - Uploaded image
  - Predicted class
  - Confidence score
  - Color-coded severity level
- Model achieved satisfactory classification accuracy on test dataset.

---

7. Conclusion & Future Enhancements

- Conclusion

  The developed system demonstrates the effectiveness of Deep Learning in automated diabetic retinopathy detection. The integration of transfer learning and web technologies enables an accessible and scalable diagnostic support tool. The color-coded output enhances interpretability for users.

- Future Enhancements
  - Implement Grad-CAM visualization for highlighting affected retinal regions
  - Use more advanced architectures like EfficientNet
  - Deploy application on cloud platform
  - Implement encrypted password storage
  - Add mobile application version
  - Integrate real-time camera-based screening

➢ Project demo video link:

https://drive.google.com/file/d/1NjrCZ0Y0pTuJRl1x46uCGzZWg4q9eNGu/view?usp=drive_link

➢ Github Repo Link:

https://github.com/36bhaskar2004/Deep-Learning-Fundus-Image-Analysis-for-Early-Detection-of-Diabetic-Retinopathy.git

➢ Dataset link:

https://drive.google.com/drive/folders/19pf9nNM50mmfErRCEMcv9kpKv2LX0Ik5?usp=drive_link

- **Trained Model link:**

https://drive.google.com/file/d/1Zur2cKEgLjBDG_LkbBmQgRUoJg-zbLXT/view?usp=drive_link

- **Documentation link:**

https://docs.google.com/document/d/1bT7q3X6M-fLAfMfPNKwpGpyOvFK3qC6w/edit?usp=drive_link&ouid=100233716855641038994&rtpof=true&sd=true