Major_Project

INTRODUCTION

Diabetic Retinopathy (DR) is a progressive microvascular disorder that affects the retina, leading to visual impairment and blindness among working-age adults in the Western population. Itis the most common microvascular complication associated with diabetes mellitus. Prolonged exposure to high blood glucose levels and metabolic abnormalities contributes to the development and progression of DR. To prevent the vision-threatening damage caused by DR, strict glycaemic control, early detection, and appropriate management are essential. This study examines thepathogenesis, diagnosis, and management of DR, emphasizing the collaborative efforts of an interprofessional team in evaluating and treating patients with this condition. By understanding the underlying mechanisms, implementing timely interventions, and coordinating comprehensive care,healthcare professionals can effectively address the challenges posed by DR and improve patient outcomes.

Objectives

1. Implementing CLAHE for Image Enhancement
2. Implementation of DCGAN on Enhanced Images
3. Using GAN-FID to Compare GAN Images
4. Application of DENSENET-201 For Classification on CLAHE Images
5. Comparison between Proposed Model (DENSENET-201) and Existing Model (ALEXNET)

Methodology

- Image Enhancement
- Data Augmentation
- Classification using Deep Learning Models
-
    - DenseNet-201
    - AlexNet

PROPOSED SYSTEM

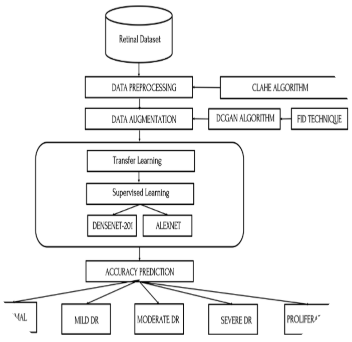


Fig 2.1 System Architecture

The system architecture provides a detailed description of the steps involved in identifying Diabetic Retinopathy (DR) using the mentioned approach.

Retinal Dataset:

Start by obtaining a retinal image dataset, such as those found on platforms like Kaggle. This dataset includes retinal images along with labels indicating the presence and severity of diabetic retinopathy. Additionally, real-time data was gathered from LVPEI Hospital and AIMER.

Data Preprocessing using CLAHE Algorithm:

Data preprocessing is a critical step to prepare the retinal images for analysis. Utilize the CLAHE (Contrast Limited Adaptive Histogram Equalization) algorithm to enhance contrast and image details. This process standardizes imag quality and improves the model's ability to detect features associated with diabetic retinopathy.[2]

Data Augmentation using GAN:

Enhance the diversity and quality of the training dataset through data augmentation using a GenerativeAdversarial Network (GAN). The use of GANs involves generating synthetic data samples that closely resemble the original dataset. This diversification process aims to create additional data points to improve the model's ability to generalize. The augmentation process specifically focuses on improving the Fréchet Inception Distance (FID), a metric that quantifies the similarity between real and generated data by considering the distribution of feature vectors extracted from InceptionNet. Reducing the FID indicates a successful data augmentation process, as it means that the generated data closely matchesthe real data in terms of statistical characteristics.

Feature Extraction with DENSENET-201 and ALEXNET:

o Utilize Pre-trained Models: Employ pre-trained DENSENET-201 and ALEXNET models to extract hierarchical features from retinal images. o Hierarchical Feature Representation: Leverage the hierarchical feature representations learned by DENSENET-201 and ALEXNET to capture complex patterns in the retinal images related to diabetic retinopathy.[14]

Classification:

o Model Training: Train a classifier, such as a fully connected neural network, on top of the features extracted by DENSENET-201 and ALEXNET. o Output Prediction: Use the trained classifier to predict the severity levels of diabetic retinopathy for each input image, classifying them into categories such as normal, mild, moderate, severe, or proliferative diabetic retinopathy based on the predicted output probabilities The success of this architectural approach relies on the quality and size of the retinal dataset, theeffectiveness of data preprocessing steps, the diversity introduced through data augmentation. Regular monitoring and evaluation of the model's performance are essential to ensure it can effectively identify diabetic retinopathy in retinal images

IMPLEMENTATION OF MODULES

Dataset Description

1. Kaggle Datasets: The project leverages two Kaggle datasets, each containing over 35,000 images sourced from the Kaggle platform. These datasets collectively provide a diverse range of diabetic retinopathy images for model training and evaluation. The images are meticulously categorized into five distinct classes:

- No DR (Diabetic Retinopathy)
- Mild DR
- Moderate DR
- Severe DR
- PDR (Proliferative Diabetic Retinopathy)

2. Real-Time Datasets: Two real-time datasets were obtained from LV Prasad Eye Hospital and AIMER (Artificial Intelligence in Medical Epidemiological Research) to supplement the Kaggle datasets. These datasets, combined with the Kaggle data, offer a comprehensive collection of diabetic retinopathy images for model training and validation.

- LV Prasad Eye Hospital Dataset:
- This dataset comprises approximately 400 images, classified into four classes:
- Mild DR
- Moderate DR
- Severe DR
- PDR (Proliferative Diabetic Retinopathy)

- AIMER Dataset: With a collection of around 2500 images, the AIMER dataset enriches the model training process with additional real-world samples. The images in this dataset are categorized into four classes: Mild, Moderate, Severe, and PDR. **Combining Datasets:** The combination of Kaggle datasets, each with over 35,000 images, along with the real-time datasets from LV Prasad Eye Hospital and AIMER, forms a robust dataset for training and evaluating the diabetic retinopathy classification model. This diverse dataset facilitates the development of a model capable of accurately categorizing the severity of diabetic retinopathy across various scenarios.

Description of Technology used

1. Programming Language: − Python: A high-level, versatile programming language widely used for various purposes including web development, data analysis, machine learning, and more.
2. Development Environments/Tools:

- Jupyter: An open-source web application that allows you to create and share documents containing live code, equations, visualizations, and narrative text.
- Colab: Google Colaboratory, a cloud-based Jupyter notebook environment that enables free access to GPU and TPU resources for executing code.
- Streamlit: A Python library that allows you to create interactive web apps for machine learning and data science projects with simple Python scripts.
- cv2 (OpenCV): Open-Source Computer Vision Library, a library of programming functions mainly aimed at real-time computer vision.
- os: Python module providing a portable way of using operating systemdependent functionality.
- opendatasets: A Python library for downloading and working with datasets from online sources.
- shutil: Python module providing a higher-level interface for file operations.

3. Data Manipulation and Analysis:

- Pandas: A powerful Python library for data manipulation and analysis, particularly useful for working with structured data.
- numpy: A fundamental package for scientific computing with Python, providing support for multi-dimensional arrays and matrices.

4. Image Processing and Computer Vision:

- cv2 (OpenCV): Used for image processing and computer vision tasks.
- clahe: Contrast Limited Adaptive Histogram Equalization, a method for improving the contrast of an image.

5. Deep Learning:

- torch: PyTorch, an open-source deep learning platform that provides a seamless path from research prototyping to production deployment.
- torchvision: A package consisting of popular datasets, model architectures, and common image transformations for computer vision tasks in PyTorch.
- dcgan: Deep Convolutional Generative Adversarial Networks, a type of generative model for generating realistic images.

6. Machine Learning Frameworks:

- TensorFlow: An open-source machine learning framework developed by Google for building and training machine learning models.
- Keras: A high-level neural networks API, written in Python and capable of running on top of TensorFlow, Theano, or Microsoft Cognitive Toolkit (CNTK).

7. Optimization Algorithms:

- Adam Optimization: An optimization algorithm used for training deep learning models, particularly effective for models with large datasets and parameters.

8. Pre-trained Models:

- Densenet-201: A convolutional neural network architecture that has achieved state-of-the-art performance on various image classification tasks.
- AlexNet: A deep convolutional neural network architecture that was one of the first models to demonstrate significant improvement over traditional methods on the ImageNet dataset.

9. Data Visualization:

- matplotlib: A comprehensive library for creating static, animated, and interactive visualizations in Python.

Module Description

1 CLAHE Algorithm

CLAHE, which stands for Contrast Limited Adaptive Histogram Equalization, is a technique used to enhance the contrast and improve the overall quality of images. It's particularly useful in medical image processing, including the analysis of retinal images for conditions like Diabetic Retinopathy. Here's how the CLAHE module works:

- **Prep-rocessing Step**: CLAHE is usually applied as a preprocessing step before feeding the images into machine learning models, such as convolutional neural networks (CNNs), for automated analysis and diagnosis.
- **Histogram Equalization:** The CLAHE technique is based on the concept of histogram equalization. The histogram of an image represents the distribution of pixel intensity values. In traditional histogram equalization, the goal is to spread out the intensity values in the image so that the full range is utilized, enhancing contrast.
- **Adaptive Approach:** What sets CLAHE apart is its adaptability. Unlike standard histogramequalization, CLAHE divides the image into smaller blocks or tiles. It computes a separate histogram and equalization for each of these tiles. This adaptive approach allows CLAHE toenhance local contrast.
- **Contrast Limiting:** To prevent over-amplification of noise and outliers in small regions, CLAHE includes a contrast limiting parameter. This parameter limits the amplification of the local contrast in each tile. It ensures that the contrast enhancement remains controlled and doesn't lead to artifacts or excessive noise. Histogram Calculation: For each tile, CLAHE calculates a histogram of pixel intensity values. The histogram reflects the distribution of intensity values within that local region.
- **Equalization**: After the histograms are calculated, CLAHE performs histogram equalizationon each tile separately. This equalization process redistributes the pixel intensity values in away that enhances local contrast

2 GAN Augmentation Technique

A GAN (Generative Adversarial Network) is a type of artificial neural network used in machine learning for generating new data, often in the form of images. DCGAN, which stands for Deep Convolutional Generative Adversarial Networks, is a type of generative model that uses deep convolutional neural networks (CNNs) to generate new images. GANs are composed of twomain parts: a generator and a discriminator. The GAN FID (Fréchet Inception Distance) is a metricused to assess the quality of the generated data compared to real data. Here's how the GAN and GAN FID module typically work: DCGAN Module:

1. Generator: The generator network takes random noise as input and learns to generate realistic images that resemble the training data. It typically consists of convolutional layers followed by batch normalization and ReLU activation functions. The final layer usually employs a tanh activation function to generate pixel values in the range [-1, 1].
2. Discriminator: The discriminator network is a binary classifier that learns to distinguish between real images from the dataset and fake images generated by the generator. It also consists of convolutional layers followed by batch normalization and LeakyReLU activation functions. The output of the discriminator is a probability indicating the likelihood that the input image is real.

Training Process:

The training of DCGAN involves training the generator and discriminator networks simultaneously in an adversarial manner:

1. Generator Training: The generator takes random noise as input and generates fake images. These generated images are then fed into the discriminator along with real images from the dataset. The generator aims to generate images that are indistinguishable from real images, fooling the discriminator.
2. Discriminator Training: The discriminator is trained to distinguish between real and fake images. It learns to assign high probabilities to real images and low probabilities to fake images. The discriminator's objective is to correctly classify images from the dataset while also correctly identifying fake images generated by the generator.
3. Adversarial Training: The generator and discriminator are trained iteratively, with each network updating its parameters to outperform the other. The generator aims to improve its ability to generate realistic images that can fool the discriminator, while the discriminator aims to become better at distinguishing between real and fake images.

GAN FID Module:

- Fréchet Inception Distance (FID): FID is a metric used to assess the quality of generated data. It quantifies the similarity between the distribution of feature vectors extracted from apre-trained InceptionNet applied to real and generated data. A lower FID value indicates thatthe generated data closely matches the real data in terms of statistical characteristics.
- FID Calculation: To calculate the FID, feature vectors from both the real and generated data are computed using a pre-trained InceptionNet. These feature vectors capture high-level information about the data. 35
- Statistical Comparison: The FID metric then compares the feature vector distributions fromreal and generated data. It takes into account the mean and covariance of these distributions.A smaller FID indicates a closer match

between the two distributions.

The GAN FID module assesses the quality of the synthetic retinal images generated by the GAN. This metric helps ensure that the synthetic data closely resembles real retinal images, which is crucial for effective data augmentation.

The GAN and GAN FID modules work together to generate high-quality synthetic data that enhancesthe training dataset's diversity, ultimately contributing to improved model performance for tasks like Diabetic Retinopathy detection. The GAN FID module helps ensure that the synthetic data is of sufficient quality to benefit the training process.

3 AlexNet Model

1. Data Collection and Preprocessing:

- Gather labelled retinal images: Obtain a dataset of retinal images annotated with diabetic retinopathy labels. This dataset should ideally be diverse and representative.
- Resize, normalize, and augment images: Resize all images to a consistent resolution suitable for input into the neural network. Normalize pixel values to a range suitable for training. Augment the dataset by applying transformations like rotation, flipping, and cropping to increase robustness and generalization of the model.

2. Model Architecture:

- Implement AlexNet: Construct the AlexNet architecture using convolutional layers, max-pooling layers, and fully connected layers. Ensure proper configuration of filter sizes, strides, padding, and activation functions as specified in the original AlexNet paper.

3. Training:

- Split data into train, validation, and test sets: Divide the dataset into three distinct subsets: one for training the model, one for validating the model's performance during training, and one for evaluating the final trained model.
- Train the model: Utilize Stochastic Gradient Descent (SGD) with momentum and learning rate scheduling to optimize the model's parameters. Monitor training progress using the validation set to prevent overfitting by adjusting hyperparameters or employing regularization techniques.

4. Evaluation:

- Evaluate model performance: Assess the trained model's performance using the test set, which the model has not seen during training or validation. Compute various evaluation metrics such as accuracy, precision, recall, and Area Under the ROC curve (AUC-ROC) to gauge its effectiveness in detecting diabetic retinopathy.

5. Fine-Tuning and Optimization:

- Fine-tune model architecture and hyperparameters: Experiment with different configurations of the model architecture, such as varying the number of layers or adjusting layer sizes, to improve performance. Tune hyperparameters such as learning rate, momentum, and regularization strength through systematic experimentation.
- Consider transfer learning: Explore the possibility of leveraging pre-trained weights from models trained on large-scale image datasets like ImageNet to expedite training or enhance performance.

6. Deployment:

- Integrate the model: Embed the trained model into a web or mobile application, or deploy it on edge devices for real-time inference. Ensure compatibility with the target platform's hardware and software requirements.

7. Monitoring and Maintenance:

- Continuously monitor model performance: Deploy mechanisms to monitor the model's performance in production, including accuracy, latency, and resource utilization. Implement alerting systems to notify stakeholders of any anomalies or degradation in performance.
- Retrain periodically: Periodically retrain the model on new data to adapt to changing patterns or distributions in the input data. Incorporate mechanisms for data collection, labelling, and retraining into the deployment pipeline to ensure the model remains effective over time.

4 Densenet-201 Model

A 201-layer deep neural network, excels in image classification and object detection. It enhances information flow and feature reuse through dense blocks, fostering direct connections between layers for improved gradient flow. The architecture includes convolution, dense blocks with batch normalization and ReLU activation, and transition blocks for spatial dimension reduction. DenseNet-201 learns complex patterns effectively with pretraining on datasets like ImageNet for transfer learning. Supported in PyTorch and TensorFlow

Layer structure of Densenet-201 Model:

- **Initial Layers:**

1. Convolutional Layer: The first layer is usually a convolutional layer with a kernel size of 7x7 and stride. This layer extracts low-level features like edges and textures from the input image.
2. Pooling Layer: A pooling layer (often max pooling) follows the convolutional layer to reduce the spatial dimensions (height and width) of the feature maps. This helps control overfitting and computational costs.
3. Batch Normalization: Batch normalization layers are often placed after convolutional layers. They help stabilize the training process and improve gradient flow.

- **Dense Blocks:** DenseNet-201 typically consists of four dense blocks, each containing a sequence of convolutional layers. These blocks are the heart of the DenseNet architecture.

1. Convolutional Layers: Within a dense block, each convolutional layer has a 3x3 kernel size and applies a rectified linear unit (ReLU) activation function. The key aspect is that each layer receives the feature maps from all preceding layers in the block (dense connectivity). This promotes feature reuse and information flow.
2. Growth Rate: A hyperparameter called the growth rate determines how many new feature maps are added by each convolutional layer in a dense block. This controls the model's capacity and allows for efficient growth in complexity.
3. Batch Normalization and ReLU: Batch normalization and ReLU activation are typically applied after each convolutional layer within a dense block.

- **Transition Blocks:** Transition blocks are inserted between dense blocks. Their purpose is twofold:

1. Dimensionality Reduction: They use a convolutional layer with a kernel size of 1x1 to reduce the number of feature channels. This helps control model complexity and prevent overfitting.
2. Pooling: A pooling layer (often average pooling) is used to further reduce the spatial dimensions of the feature maps. Final Layers:
3. Global Average Pooling: After the final dense block, a global average pooling layer is often used. This reduces the feature maps to a single vector, summarizing the information for classification.
4. Fully-Connected Layer: A final fully-connected layer with a number of neurons equal to the number of image classes performs the classification task

RESULTS

1 CLAHE Algorithm

The application of the CLAHE (Contrast Limited Adaptive Histogram Equalization) algorithm to the set of Fundus images yielded significant improvements in image quality and contrast. This enhancement process contributed to the successful extraction of subtle details and features within the images, particularly within the retinal region.
The results showed that the CLAHE algorithm effectively mitigated issues related to poor lighting, uneven contrast, and variations in image quality encountered in Fundus images. As a consequence, the retinal structures and abnormalities associated with diabetic retinopathy became more discernible, facilitating a more precise analysis.

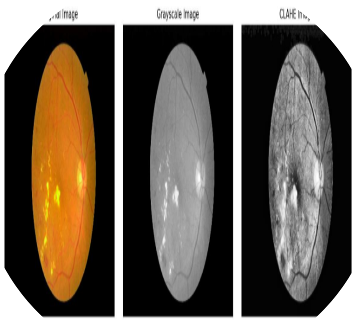
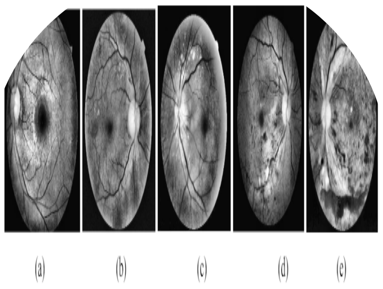


Fig 4.1 Original vs Grayscale vs CLAHE

In Fig 4.2, a series of images processed with CLAHE are showcased, each portraying varying levels of severity in Diabetic Retinopathy. These images serve as visual representations capturing the diverse manifestations and progression stages of the condition. Through the application of CLAHE, subtle nuances and distinct features indicative of different severity levels are brought to light, offering valuable insights into the diagnostic and prognostic aspects of Diabetic Retinopathy. This comprehensive collection of images not only highlights the multifaceted nature of the disease but also underscores the significance of advanced image processing techniques in enhancing understanding and management of retinal pathologies.



4.2 (a)No DR, (b) Mild DR, (c) Moderate DR, (d) Severe DR, (e) PDR

2 GAN Technique

DCGAN Algorithm

The trained DCGAN successfully generated a set of synthetic fundus images containing Diabetic Retinopathy (DR) from the input black and white images enhanced using CLAHE. The generated images demonstrate a remarkable level of visual fidelity and coherence, closely resembling the characteristics present in the original dataset. Through the iterative adversarial training process, the DCGAN effectively learned to capture intricate patterns and details inherent in the input images, resulting in the production of highly realistic outputs.

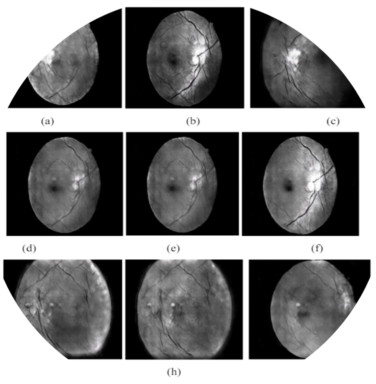
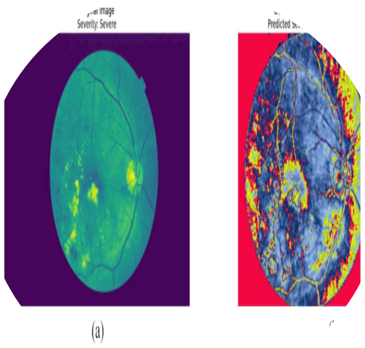


Fig 4.4 GAN Images

3 Densenet-201 Model

The DenseNet-201 model was meticulously trained to classify Diabetic Retinopathy (DR) severity levels in fundus images, aiming to provide accurate and reliable diagnostic outcomes for early intervention and management of the disease.
The model demonstrated exceptional accuracy in classifying DR severity levels, boasting an overall high accuracy rate. Accuracy, in this context, refers to the model's ability to correctly categorize images into their respective severity categories, which include no DR, mild, moderate, severe, and proliferative DR.
The high accuracy rate indicates the effectiveness of the model in precisely identifying the presence and severity of DR in

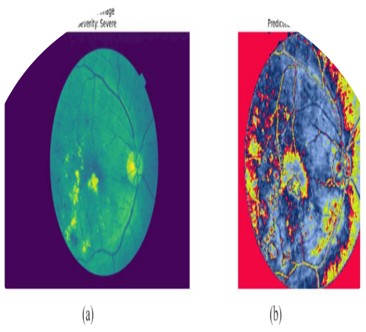
fundus images. This suggests that the model's learned features and decision-making process effectively capture the distinct patterns associated with different stages of the disease.



Net: (a) Input Image (b) Classified output with

4 Alexnet Model

The AlexNet model was rigorously trained to classify Diabetic Retinopathy (DR) severity levels in fundus images, with the aim of facilitating accurate and timely diagnosis of the disease for effective patient management.
The model demonstrated commendable accuracy in classifying DR severity levels, achieving a high accuracy rate across various severity categories. Accuracy, a fundamental metric in classification tasks, reflects the model's ability to correctly classify images into their respective severity stages, spanning from no DR to proliferative DR, as depicted in Fig 4.7. The robust accuracy attained by the AlexNet model underscores its proficiency in capturing the nuanced features indicative of different stages of DR, thereby enabling precise diagnostic outcomes.



7 AlexNet: (a) Input Image (b) Classified output with input

5 Comparative Study

The project aimed to compare the performance of DenseNet and AlexNet models in classifying Diabetic Retinopathy (DR) severity levels in fundus images, with the objective of identifying the most effective model for accurate and reliable disease diagnosis and management.
Both the DenseNet and AlexNet architectures underwent rigorous training and evaluation processes using an identical dataset comprising fundus images depicting diverse severity levels of DR. Throughout the evaluation, the performance of

each model was primarily gauged based on its accuracy in classifying these images accurately. Figure 4.8 provides a visual representation of how both models classified the severity levels depicted in the input images.

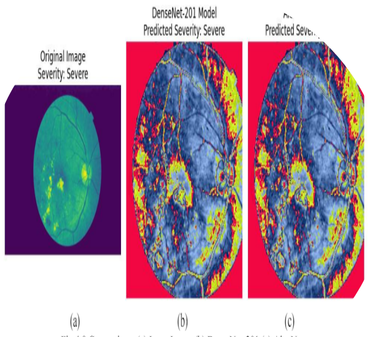


Fig 4.8 Comparison: (a) Input Image (b) DenseNet-201 (c) AlexNet

6 User Interface

Upon uploading a fundus image, the interface swiftly enhances it using the CLAHE algorithm and processes it through a pre-trained Densenet-201 model for DR classification. The detected severity level of DR is then displayed. Refer to Figure 4.10 for a visual representation of the interface.

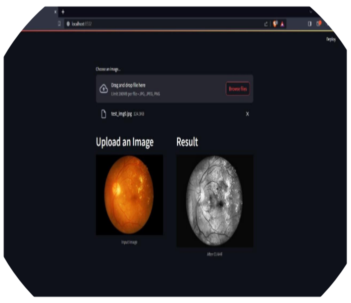


Fig 4.10 User Interface