

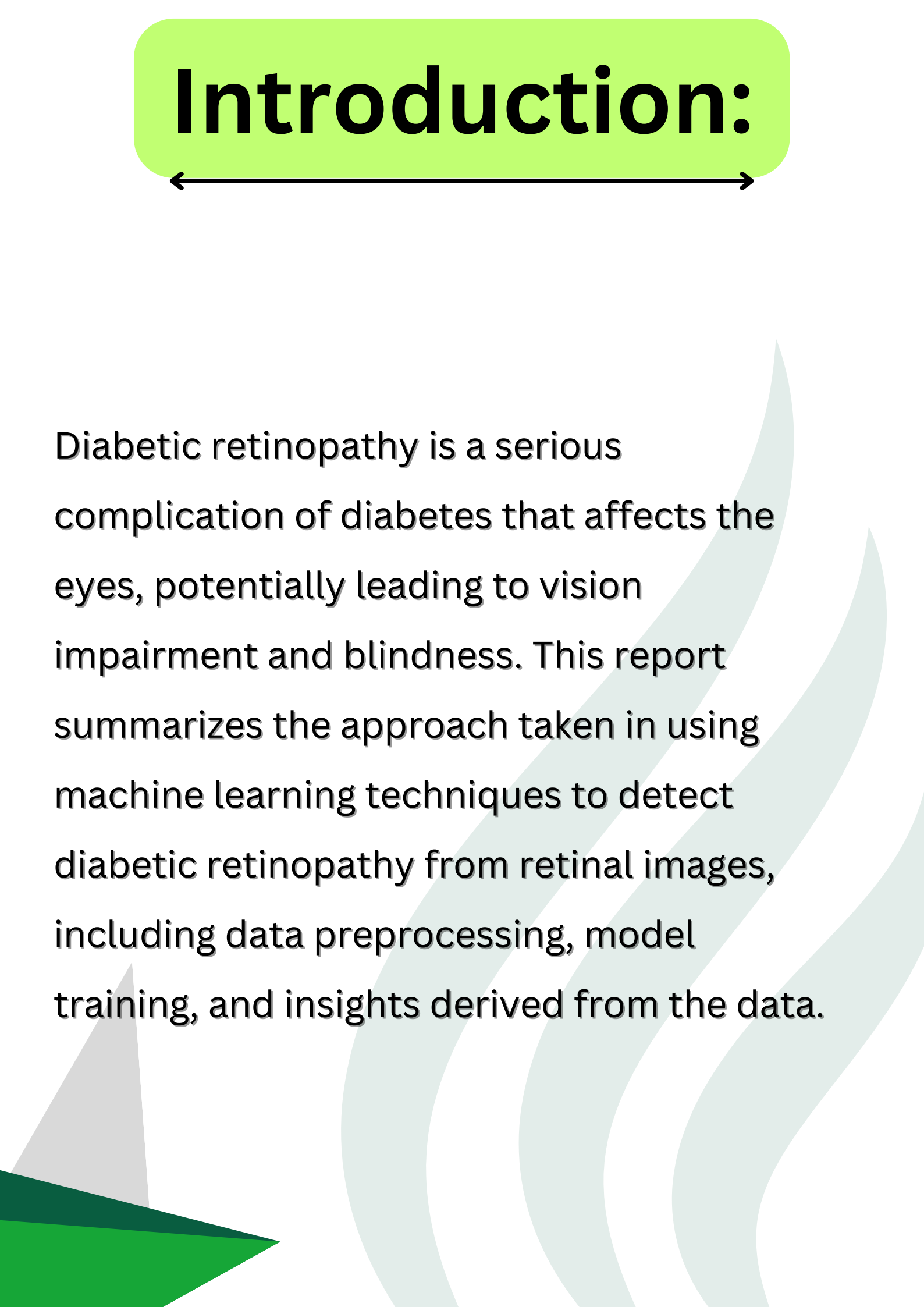
Two vertical lines, one dark green and one black, are positioned on the left side of the page.

DIABETIC RETINOPATHY CLASSIFICATION

PROJECT REPORT

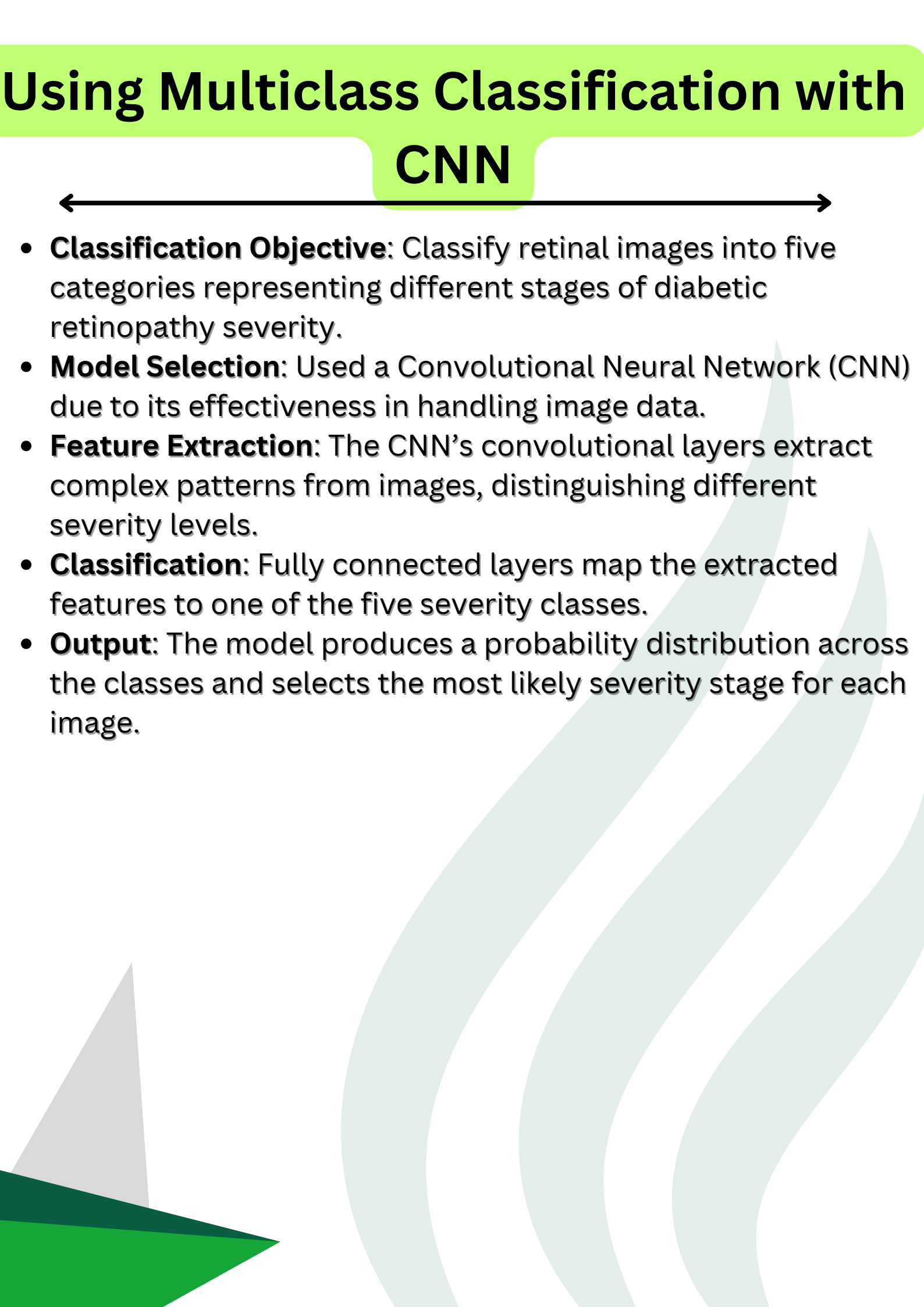
A horizontal line with a small black rectangular segment in the middle.Three large, light green, wavy lines that curve upwards from the bottom right towards the center of the page.A series of overlapping geometric shapes in green and black at the bottom left corner, including a triangle and a trapezoid.

Introduction:



Diabetic retinopathy is a serious complication of diabetes that affects the eyes, potentially leading to vision impairment and blindness. This report summarizes the approach taken in using machine learning techniques to detect diabetic retinopathy from retinal images, including data preprocessing, model training, and insights derived from the data.

Using Multiclass Classification with CNN

- 
- **Classification Objective:** Classify retinal images into five categories representing different stages of diabetic retinopathy severity.
 - **Model Selection:** Used a Convolutional Neural Network (CNN) due to its effectiveness in handling image data.
 - **Feature Extraction:** The CNN's convolutional layers extract complex patterns from images, distinguishing different severity levels.
 - **Classification:** Fully connected layers map the extracted features to one of the five severity classes.
 - **Output:** The model produces a probability distribution across the classes and selects the most likely severity stage for each image.

Custom CNN Architecture

This section introduces a custom Convolutional Neural Network (CNN) architecture designed to classify retinal images into multiple severity levels of diabetic retinopathy

The model includes the following components:

1. Convolutional Layers:

- Three convolutional layers progressively extract detailed features from input images, each followed by batch normalization and a ReLU activation function to stabilize training and introduce non-linearity.

2. Pooling Layers:

- Each convolutional layer is followed by a max-pooling layer, which reduces the spatial dimensions of feature maps, helping to retain important features while reducing computational load.

3. Fully Connected Layers:

- After flattening, the network has two fully connected layers. The first maps the features to a smaller dimension, and the second outputs a probability distribution across five classes, corresponding to the diabetic retinopathy severity levels.

4. Dropout:

- Dropout is applied after the first fully connected layer to reduce overfitting by randomly zeroing out a portion of the layer's activations during training.

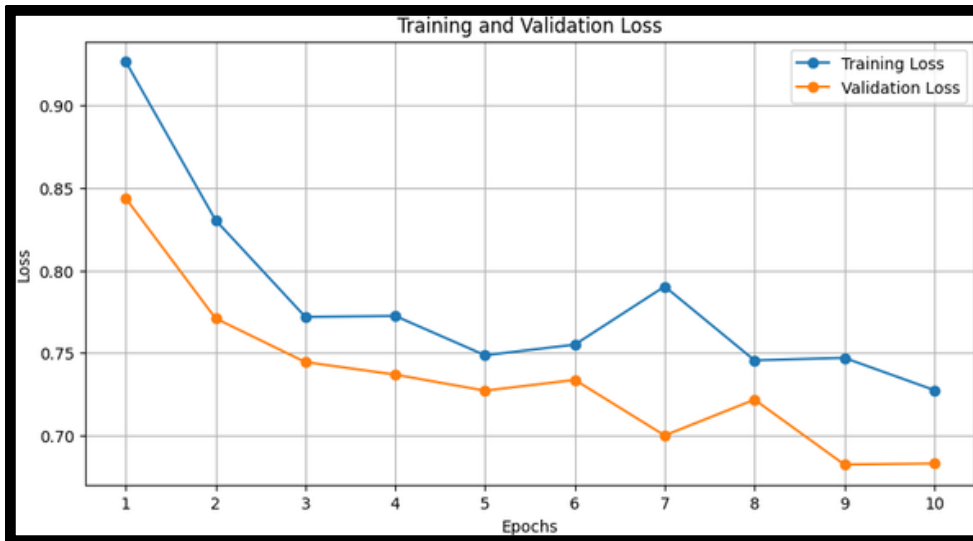
5. Device Setup:

- The model is configured to run on a GPU if available, optimizing performance during training and inference on large image datasets.

Model Training and Evaluation

- **Loss Function:** Cross-Entropy Loss, ideal for multiclass classification, is used to measure prediction accuracy.
- **Optimizer:** Adam Optimizer adjusts the learning rate adaptively to improve convergence.
- **Training Loop:** The model updates weights through backpropagation over multiple epochs, enhancing classification accuracy.
- **Validation Loop:** After each epoch, performance metrics (accuracy, precision, recall, F1-score) are calculated on the validation set to monitor generalization.
- **Performance Metrics:** Metrics provide insight into the model's effectiveness in classifying severity levels.
- **Loss Visualization:** A graph of training and validation loss helps track convergence and detect overfitting.
- **Progress Bars:** tqdm bars show real-time training and validation progress

Training and Validation Loss Graph



1. Loss Analysis

- Training Loss decreases sharply at first, then stabilizes with minor fluctuations, indicating initial learning followed by possible overfitting.
- Validation Loss steadily declines, reaching its lowest around epoch 9, showing good generalization.

2. Metrics Summary

- Accuracy reaches 76.73% by epoch 10, showing steady improvement.
- Precision and Recall improve gradually, with a balanced F1-score of 0.7347 at the end, indicating effective model performance.

3. Key Takeaways

- The model generalizes well, with validation loss and metrics stabilizing.
- Minor tuning could reduce training loss fluctuations and further improve performance stability.

AlexNet



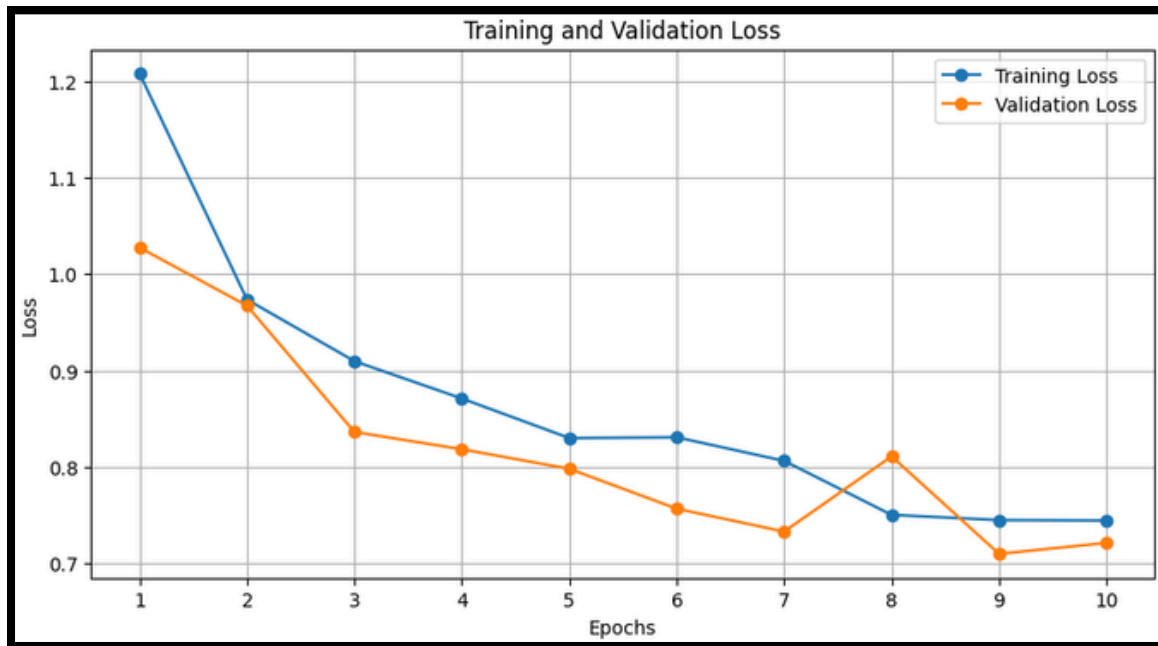
- This section describes the implementation of a custom AlexNet-based Convolutional Neural Network (CNN) model, tailored for diabetic retinopathy classification. The model consists of convolutional layers to extract spatial features, followed by fully connected layers for classification into five severity classes. It uses dropout regularization to mitigate overfitting and is structured to handle multiclass classification with efficient feature extraction and representation learning.

Code Analysis



- The CustomAlexNet class is designed for image classification with five output classes. The feature extraction part comprises multiple convolutional and pooling layers, which capture hierarchical features from input images. The classifier section uses fully connected layers with dropout, enhancing the model's ability to generalize by reducing overfitting. This architecture is implemented to run on a GPU if available, making it suitable for handling large datasets efficiently.

AlexNet Model Performance Summary



- The custom AlexNet model demonstrates significant improvement over 10 epochs, as shown in the loss and metric trends:
- Loss Trends: Training loss decreases sharply at first and stabilizes, while validation loss steadily declines, reaching a low point around epoch 9. This indicates effective learning and generalization.
- Accuracy and Metrics: Accuracy improves from 49.27% to 73.64%, showing consistent progress. Precision, recall, and F1-score also improve, with a final F1-score of 0.6878, suggesting balanced classification performance.
- Overall: The model generalizes well, though slight fluctuations in validation loss and metrics suggest that further tuning may enhance stability.

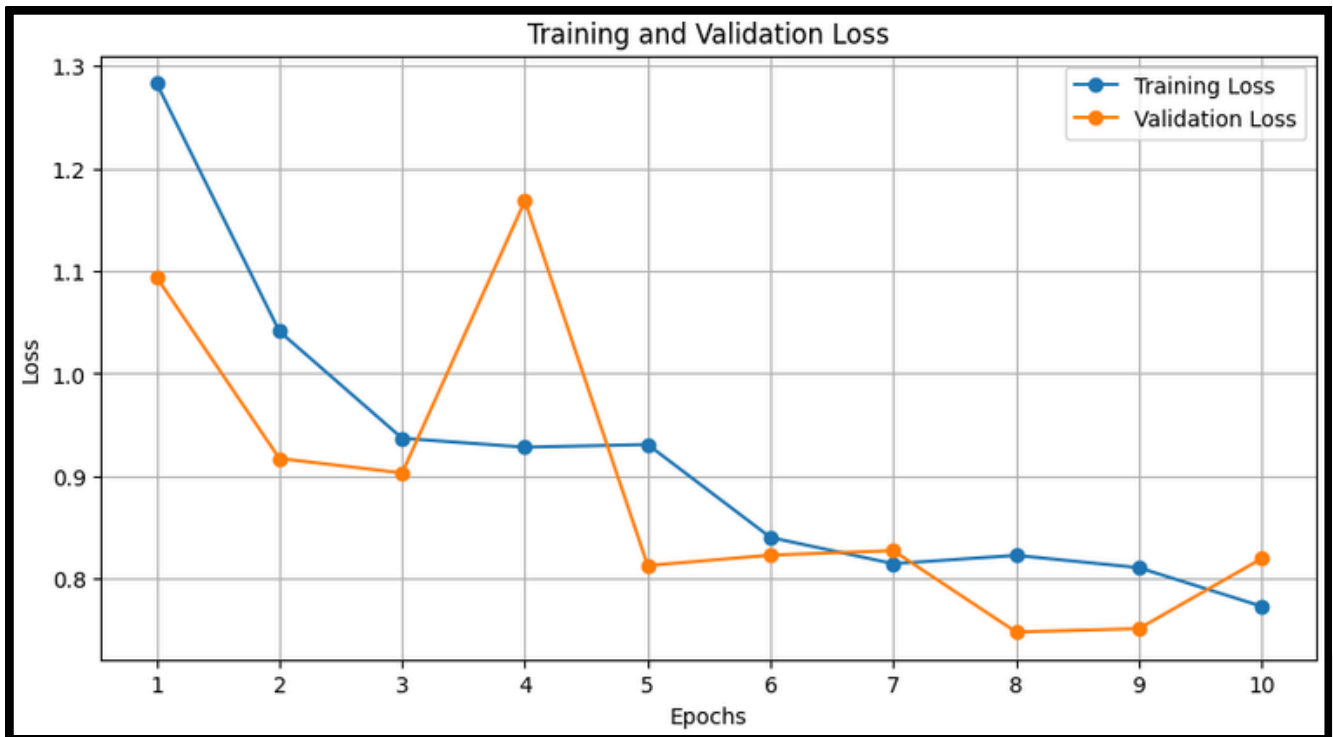
VGG Model

- This section covers the custom VGG-based Convolutional Neural Network (CNN) model designed for diabetic retinopathy classification. Like AlexNet, this model uses multiple convolutional layers but with a deeper and more complex architecture to capture detailed spatial features. It includes multiple blocks of convolutional and pooling layers, followed by fully connected layers in the classifier, allowing it to perform well on complex classification tasks.

Code Analysis

- The CustomVGG model uses a deep architecture with sequential convolutional and pooling layers for hierarchical feature extraction. Dropout is applied in the classifier section to reduce overfitting. The final layer classifies images into five severity levels of diabetic retinopathy. This structure, combined with GPU compatibility, makes it suitable for large-scale image classification.

VGG Model Performance Summary



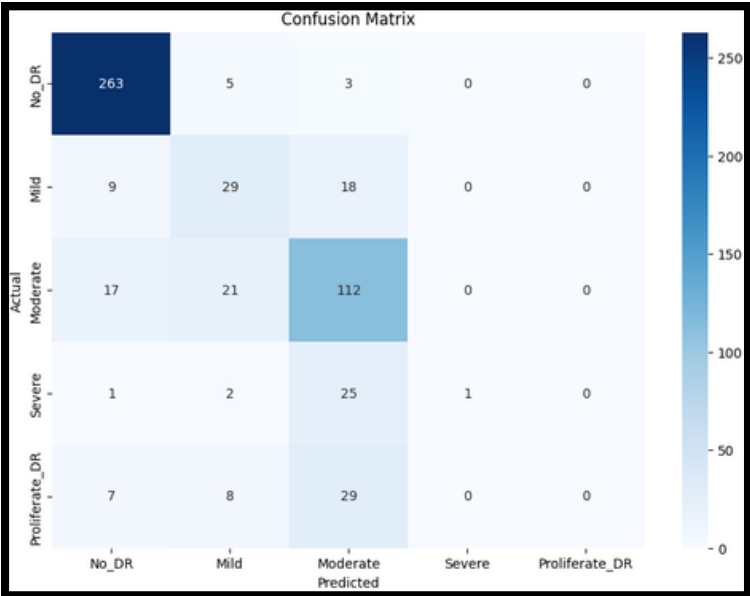
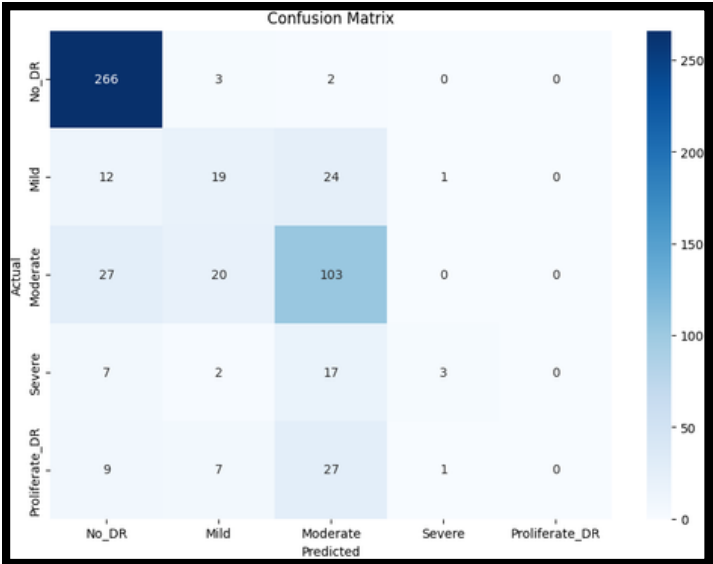
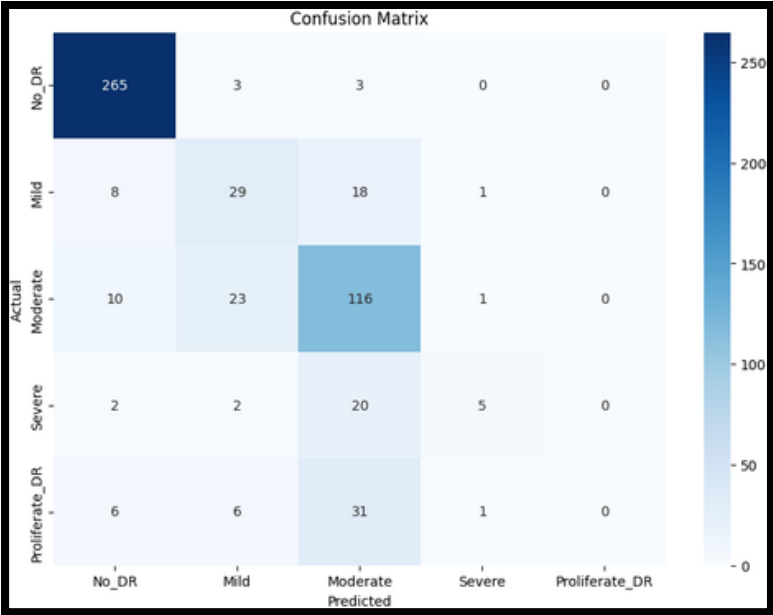
The VGG model shows consistent improvement over 10 epochs, reaching 72% accuracy and an F1 score of 0.6686. Key points:

- **Loss:** Training loss steadily decreases; lowest validation loss at epoch 8 indicates good generalization.
- **Metrics:** Accuracy, precision, recall, and F1 score improve gradually, showing effective learning.
- **Stability:** Consistent metric gains reflect stable performance.

Confusion Matrix Analysis for Model Evaluation

The confusion matrices for the three models (AlexNet, VGG, and the custom model) provide insights into their classification performance across diabetic retinopathy severity stages. Key observations:

- **Class-Specific Accuracy:** The confusion matrices highlight which severity stages each model predicts well and where misclassifications occur, identifying potential areas for improvement.
- **Comparison Across Models:** By comparing matrices, we can evaluate which model handles each class best and where they struggle, assisting in model selection.
- **General Performance Insight:** The distribution of correct vs. incorrect predictions reflects each model's robustness and ability to generalize across severity stages.



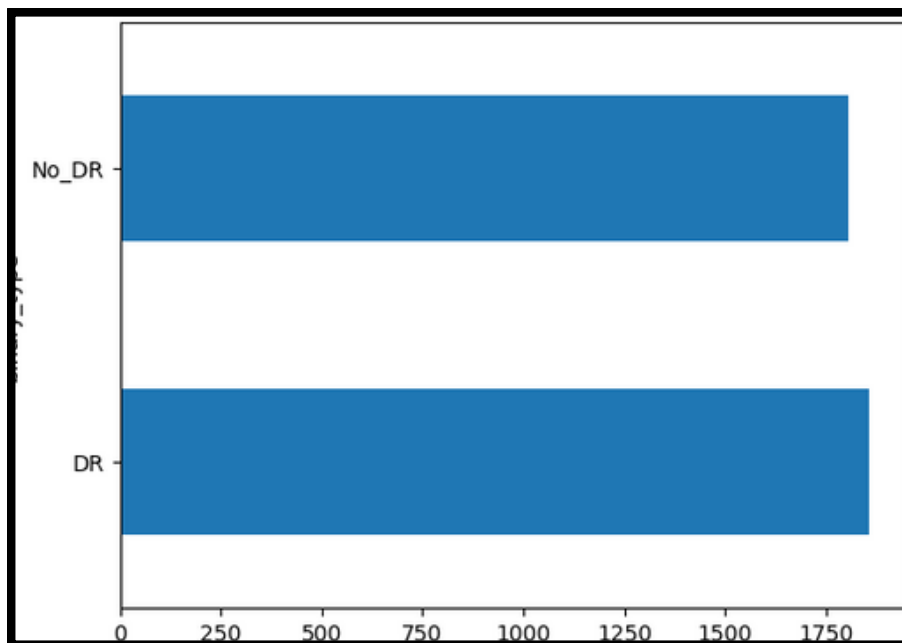
Summary of Findings (Binary Classification):

The project involved using machine learning models to classify images by diabetic retinopathy severity.

> Data Preprocessing: Images were resized and standardized.

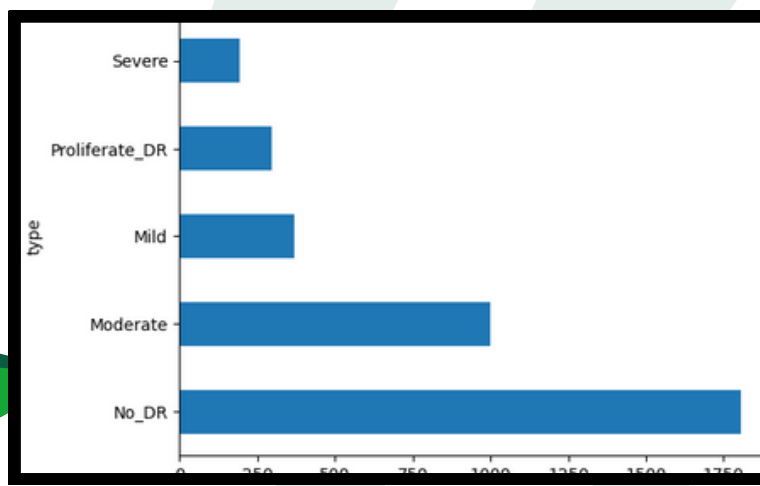
Augmentation methods, like rotation and brightness adjustments, were used to diversify data and handle class imbalance.

- **Binary Labels:** Maps diagnosis labels (0-4) to 'No_DR' or 'DR' for a simplified binary classification.



- **Multi-Class Labels:** Maps labels to specific stages ('No_DR', 'Mild', 'Moderate', 'Severe', 'Proliferate_DR').

Two new columns, `binary_type` and `type`, are added for binary and multi-class labels, allowing both classification approaches in model training.



Model Training and Evaluation:

The training involved optimizing the model with the Adam optimizer and CrossEntropyLoss as the loss function, with the model trained over 10 epochs.

Each epoch consists of two main steps:

1. Training Phase:

- Each epoch consists of a training loop where the model learns from the training dataset.
- The training loss is computed after each batch, indicating how well the model is fitting the data in real-time.
- Early in training (e.g., Epoch 1), loss is relatively high, but it decreases significantly by the later epochs, showing improved performance and convergence.

2. Evaluation Phase:

- After each epoch, the model is evaluated on the validation dataset to track its performance on unseen data.

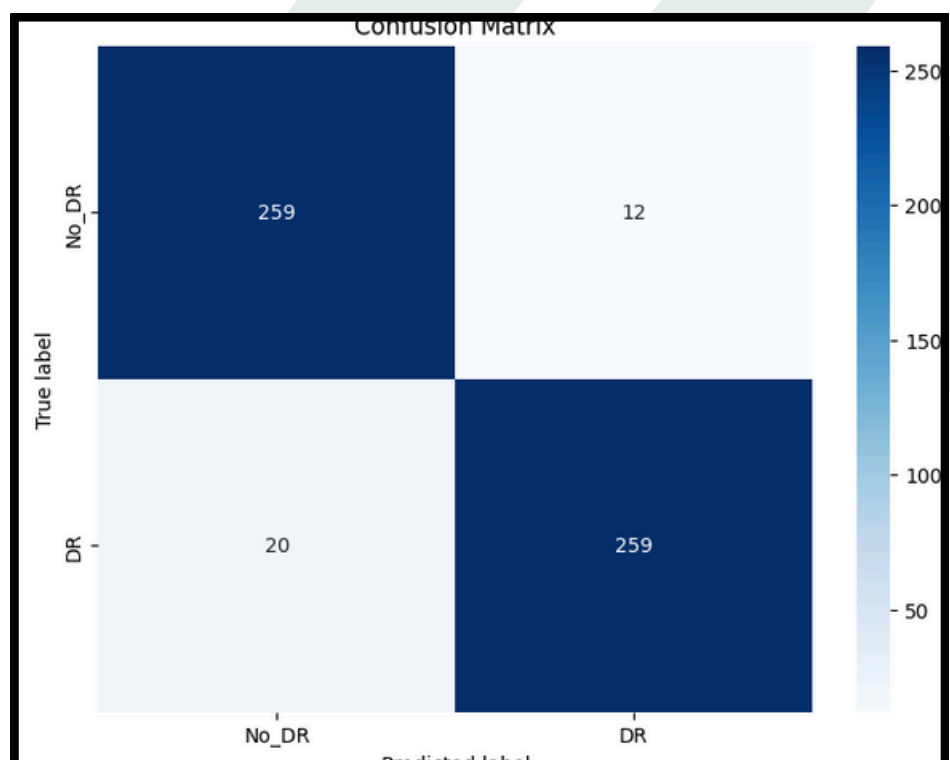
Key metrics are calculated:

- Epoch 1: Accuracy of 66.55%, precision of 77.86%, recall of 66.55%, F1 score of 62.51%.
- Epoch 2: Sharp improvement with 92.55% accuracy, 92.56% precision, 92.55% recall, and F1 of 92.55%.
- Epochs 4–10: Metrics stabilize around 95% accuracy, precision, and recall, indicating consistent performance and minimal overfitting.

The results demonstrate a clear progression, with metrics improving significantly in the initial epochs and stabilizing as the model converges. This approach helps verify that the model generalizes well to unseen data.

Confusion matrix and Visualization:

- Confusion Matrix Plotting (`plot_confusion_matrix`):
- This function generates a confusion matrix for the model's predictions versus the true labels.
- It uses Seaborn's heatmap function to display the matrix with annotated cell values, making it easy to see classification accuracy across different classes.
- The x-axis shows the predicted labels, and the y-axis shows the true labels, with blue color intensity representing the frequency of each outcome.
- Prediction Visualization (`visualize_predictions`):
- This function displays a specified number of images from the data loader along with the model's predictions and the true labels.
- It runs the model in evaluation mode and predicts labels for a batch of images without gradient tracking.
- Each image is plotted with a title showing the predicted and true class labels, enabling easy visual inspection of the model's classification accuracy on sample images.



Hyperparameter tuning:

Hyperparameter tuning process for a convolutional neural network (CNN) model by testing various combinations of learning rates, batch sizes, epochs, and dropout rates. The primary objective is to identify the best set of parameters that maximizes validation accuracy on a binary classification task.

Key Components

1. Hyperparameter Options:

- Learning Rates: [1e-4, 1e-3]
- Batch Sizes: [32, 64]
- Number of Epochs: [10, 15]
- Dropout Rates: [0.1, 0.2]

This grid search tests 16 combinations of the above parameters to identify the best configuration.

2. Training with Different Hyperparameters:

- The `train_with_hyperparameters` function initializes a new CNN model with the specified dropout rate, which is then moved to the appropriate device.
- For each parameter configuration, a custom `DataLoader` is created to load the training and validation data with the specified batch size.
- The model is trained using the Adam optimizer with the specified learning rate, and the loss is calculated using `CrossEntropyLoss`.
- Training proceeds in mini-batches over the defined number of epochs.

3. Evaluation Process:

- After each configuration completes training, the model is evaluated on the validation set.
- The accuracy score is calculated based on the validation set predictions and the actual labels.
- The accuracy score for each parameter configuration is printed, enabling comparison across configurations.

Selecting the Best Parameters:

A comparison of validation accuracy across all configurations determines the best-performing set of parameters.

If the accuracy for the current configuration exceeds the best accuracy observed so far, the script updates `best_acc` and `best_params`.

Results

The results of each tested configuration, including accuracy scores, are summarized as follows:

- Top Performing Configuration: The highest accuracy of 95.09% was achieved with the following parameter settings:
 - Learning Rate: 0.0001
 - Batch Size: 64
 - Number of Epochs: 15
 - Dropout Rate: 0.2
- Other Observations:
 - Generally, a lower learning rate (0.0001) and larger batch size (64) tend to yield higher accuracy.
 - Models with a dropout rate of 0.2 generally performed better than those with 0.1, suggesting that increased regularization improved generalization.
 - Increasing the number of epochs from 10 to 15 often improved accuracy, but only when paired with a suitable learning rate.

Final Model

1. Model Initialization:

- A new CustomCNN model is initialized with the best dropout rate (0.2) and transferred to the specified device (e.g., GPU or CPU).
- Data loaders for the training and validation datasets are created with the optimal batch size (64) and other selected hyperparameters.

2. Training Process:

- The model is trained for the optimal number of epochs (15) with an Adam optimizer using the best learning rate (0.0001).
- During each epoch, the model goes through mini-batch training, calculating loss using CrossEntropyLoss. Gradients are computed and updated via backpropagation.

3. Evaluation and Visualization:

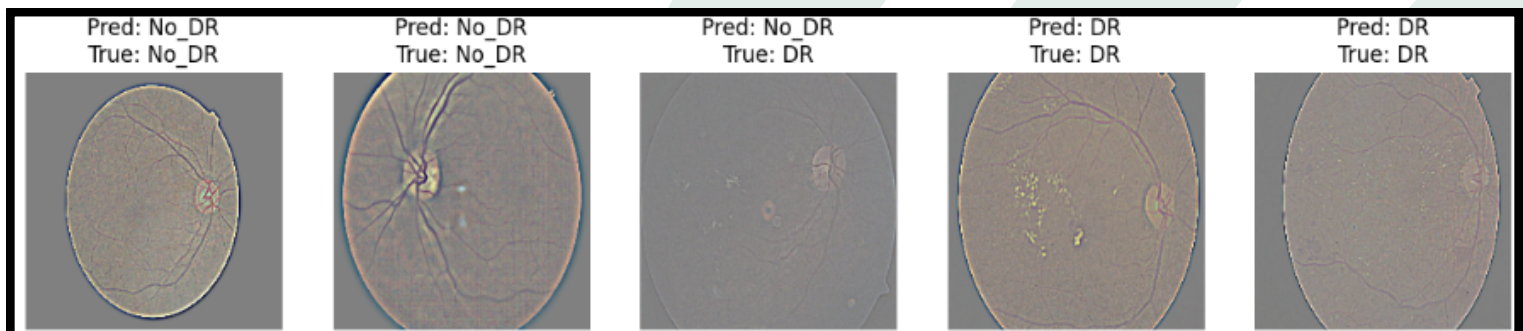
- After training, the model's performance is evaluated on the validation set. The predictions are compared against true labels to compute the final validation accuracy.
- A confusion matrix is generated and displayed, showing classification results for each category: "No_DR" (No Diabetic Retinopathy) and "DR" (Diabetic Retinopathy).
- The final validation accuracy achieved is 94.18%.

4. Model Saving:

- The trained model's parameters are saved to a file named diabetic_retinopathy_model.pth, allowing for future use or deployment.

5. Prediction Visualization:

- The visualize_predictions function is then used to display sample test images with both the model's predicted and true labels, allowing a quick qualitative assessment of the model's performance.



Specific Insights on Diabetic Retinopathy :

The model's classification of diabetic retinopathy images uncovered the following patterns, which provide valuable insights for clinical use and further research:

- **Early Stage Identification:** In early stages, the model successfully detected microaneurysms—small areas of ballooning blood vessels—which are among the first signs of retinopathy.
- **Progression Markers:** As retinopathy advances, the model could identify more severe indicators, such as hemorrhages and exudates, and detect neovascularization (formation of new, fragile blood vessels). These are critical for assessing progression and tailoring treatments.
- **Importance of Timely Intervention:** The model's high sensitivity in identifying early signs suggests potential for aiding in early intervention strategies, thereby improving patient outcomes and preventing further deterioration.

Conclusion:



The use of CNNs to detect and classify diabetic retinopathy from retinal images has demonstrated promising results, with high accuracy and reliable identification of disease markers. This approach offers a valuable tool for healthcare providers, enabling faster and potentially more accessible screening for diabetic retinopathy. Future improvements could focus on integrating more diverse datasets to improve generalizability, especially across different demographics and imaging conditions.

TEAM 29



- **Prasangeet Dongre**
(B23CH1033)
 - **Rajas Hitendra Kadu**
(B23CH1039)
 - **Pujari Mayuri Rajesh**
(B23ES1026)
 - **Prakhar Chauhan**
(B23BB1032)
- 
- 