Project Report

Ammar Saleem(I210327,AI-J)

Executive Summary Report

1. Introduction to Diabetic Retinopathy:

Diabetic Retinopathy (DR) is a severe complication of diabetes, affecting the retina and potentially leading to irreversible vision impairment or blindness. With approximately 463 million adults worldwide living with diabetes, the prevalence of DR has surged, underscoring its significance as a global public health concern. The disease progresses through stages, initially causing microvascular abnormalities and later advancing to proliferative stages with abnormal blood vessel growth.

The asymptomatic nature of early-stage DR makes regular eye screenings crucial for early detection and intervention. Machine learning models, as demonstrated in the provided code, offer a promising solution. By leveraging advancements in computer vision, these models enhance the efficiency and accuracy of DR diagnosis, contributing to early intervention and improved patient outcomes. This intersection of healthcare and technology plays a vital role in addressing the challenges posed by diabetic retinopathy on a global scale.

2. Project Overview:

- Problem: Detection and classification of diabetic retinopathy in retinal images.

- Dataset: Utilized a publicly available dataset from Kaggle, including train and test images with corresponding labels.

  - https://www.kaggle.com/competitions/aptos2019-blindness-detection/data

- Source Code:

  - [Original Code](https://www.kaggle.com/code/hazemahmedmurshedi/diabetic-retinopathy-detection-by-resnet18)

3**. Data Exploration and Feature Engineering:**

- Data Loading: Loaded the dataset using pandas and extracted relevant information.

- Image Processing: Utilized libraries like OpenCV for image preprocessing and manipulation.

- Feature Engineering: Applied data augmentation techniques, including resizing, color jittering, affine transformations, and horizontal flipping, to enhance model generalization.

- Custom Model Architecture: Implemented a custom neural network model with a GeM layer, leveraging transfer learning from a pre-trained ResNet18 model.

- Training Loop: Introduced dynamic adjustments to the model architecture during the forward pass.

- Optimization: Used the Adam optimizer and Smooth L1 Loss criterion for model training.

- Testing: Evaluated the model on the test dataset, computing accuracy and presenting predictions.

4. Key Standard Libraries and Functions:

- OpenCV: Applied for image preprocessing, resizing, and data augmentation.

  - `cv2.resize()`, `cv2.cvtColor()`, `cv2.flip()`

- NumPy: Utilized for numerical operations and array manipulations.

  - Array creation, slicing, and transformations.

- Pandas: Employed for data manipulation and exploration.

  - `pd.read_csv()`, DataFrame manipulation, and value extraction.

- PyTorch: Used for building and training the neural network model.

  - Defining custom models, loss functions, and optimization with `torch.nn` and `torch.optim` modules.

5. Contribution:

- Advanced Data Augmentation:

  - Implemented an extensive set of image augmentation techniques, including contrast adjustments, brightness variations, hue shifts, saturation changes, blur and sharpening effects, and random rotations, scales, shears, and shifts. This contributes to a more diverse and robust training dataset, enhancing the model's ability to handle real-world variations.

- Dynamic Model Architecture Adjustment:

  - Introduced a mechanism to dynamically set the input size of the first fully connected layer (`fc1`). This adaptive architecture adjustment allows flexibility in handling varying input sizes, ensuring compatibility with different image resolutions. It improves the model's scalability and adaptability to diverse datasets.

- Enhanced Training Loop:

  - Extended the training loop to include a flexible number of iterations (`iterations`). This improvement provides better control over the training process, allowing for more fine-grained adjustments and increased convergence opportunities. The iterative training approach contributes to improved model learning and adaptability.

- Improved Code Readability:

  - Enhanced code readability through structured comments and documentation. Clearly explained the purpose and functionality of critical code segments, making it easier for collaborators or future developers to understand and modify the code. This improves code maintainability and collaboration potential.

- Iterative Model Evaluation:

  - Implemented an iterative testing loop over ten iterations to compute and average the test predictions. This approach provides a more stable evaluation metric, reducing the impact of random variations in test predictions. The result is a more reliable and consistent accuracy assessment.

- Dynamic Learning Rate Scheduling:

  - Applied a learning rate scheduler (`optim.lr_scheduler.StepLR`) to adjust the learning rate during training. This adaptive learning rate strategy enhances model convergence and performance by dynamically altering the learning rate based on training progress. It contributes to more efficient optimization.

- Comprehensive Report Contribution:

  - Provided a detailed and clear executive summary report covering essential aspects of the project, including problem definition, dataset details, code source links, and a thorough overview of contributions. This report aims to facilitate better understanding and dissemination of the project's objectives and outcomes.

6. Conclusion:

The project successfully addresses the challenge of diabetic retinopathy detection using machine learning techniques. The combination of advanced data augmentation, custom model architecture, and dynamic adjustments contributes to an effective solution for real-time applications.

7. Future Work:

Future work may involve fine-tuning hyperparameters, exploring additional neural network architectures, and incorporating advanced optimization techniques to further enhance model performance.