CHAPTER 5

HYBRID ARCHITECTURE FOR CLASSIFYING DR STAGES

Medical image analysis has transformed modern healthcare, providing novel diagnostic tools that immensely improve clinical judgment and patient care. The increasing use of medical imaging methods such as X-rays, CT scans, Magnetic Resonance Imaging (MRI), and FP has led to massive data accumulation. An efficient and intelligent interpretation of this data requires automated methods to produce correct and easy-to-interpret estimates.

5.1 MEDICAL IMAGE ANALYSIS: TRANSFORMING HEALTHCARE WITH ML AND DL

Conventional ML methods have been crucial in this area, supporting various applications in medical image analysis [92,93]. Classical ML methods have been extensively employed for specific diagnostic tasks, e.g, tumour detection, retinal disease classification, etc. (such as SVM and k-Nearest Neighbours (k-NN) classifiers). These methods depend on a strong hand-crafted feature extraction module, in which domain experts must detect and select suitable features in the imaging data.

Although this method has exhibited effectiveness in different applications, it is inherently demanding in terms of time and human effort. Furthermore, the efficiency of these methods is generally subject to the knowledge of the person who performs the feature extraction, contributing to the bias and variability of the analysis. Such limitations highlight the demand for more stable and automatic procedures, which can provide reliable results broadly applicable to many different datasets and health settings.

The advent of DL has revolutionised medical image analysis by providing a whole new level of sophistication and effectiveness. DL-based models, typically CNNs, have shown unprecedented power in learning and automatically extracting the hierarchy of features directly from raw image data. Unlike classical ML methods, where features are designed manually for analysis, the CNNs have removed this manual feature designing dependency and have made better analysis easier, with improved accuracy and

reproducibility. This has made CNNs an invaluable asset for a broad spectrum of medical imaging applications [94].

CNNs for healthcare use cases are numerous and impactful. For example, CNN-based models have demonstrated remarkable success in tumour segmentation, for which accurate delineation of tumour boundaries is important for treatment planning. CNNs have also been used to detect pneumonia from chest X-rays, providing quicker and more precise diagnosis in ICUs.

However, despite remarkable advances through DL, many issues hinder its widespread clinical application. One of the main problems is overfitting, when the model does great with training data and fails with new data. This problem is even more severe in medical image analysis, as the size of annotated datasets is usually limited by the high cost and the expertise required for manual labelling. Furthermore, concerns about DL models' generalizability in both imaging modalities, patient demographics, and clinical settings continue. Differences in image quality, acquisition protocols, and patient cohorts strongly affect model performance, requiring the design of models with enhanced robustness and adaptability.

To tackle these challenges, researchers today have been extending their investigations on sophisticated methods like data augmentation, TL, and hybrid model architectures, among others, which offer a mix of classical ML and DL based methods. Closer to the tasks in this field, there have been efforts in generating large, standardised, and annotated datasets as a benchmark for training and assessing medical imaging models. Joint projects between the healthcare system, academia, and industry have been at the forefront of developing the discipline and evolving to next-generation diagnostics.

5.1.1 Advancing Medical Image Analysis: Transitioning from ML and DL to Hybrid Approaches

Over the years, the trend in medical image analysis techniques has moved from classical ML and DL models to modern hybrid and ensemble schemes. Although ML and DL have been powerful in guiding many breakthroughs, with the complexity and diversity of the datasets faced, the community is increasingly turning its enthusiasm towards hybrid strategies [95]. These state-of-the-art techniques are to solve the following problems:

robustness, generalisation, performance consistency, and improving the availability of the method for clinical application.

Hybrid methods are significant step forward in capturing different techniques to exploit their synergistic properties. Often, these models mix ML/DL techniques or a DL architecture with domain-specific approaches to deal with complex medical imaging problems. For example, hybrid models are particularly efficient for multimodal data analysis, where data from different imaging sources, such as CT and MRI, need to be integrated to gain a complete understanding. In addition, these models have proven successful in applications that require combining spatial and temporal information, including monitoring disease progression over time based on sequential imaging. Hybrid approaches are general enough to be unified to solve various medical image analysis tasks.

The basic reason for adopting hybrid techniques is the ability to provide more accuracy and robustness than the independent model can. In medical diagnostic work, where accuracy is critical, these methods have been reported to significantly enhance performance metrics like sensitivity, specificity, and predictive value. Hybrid models have been of great value whenever data features are heterogeneous and data need to be reconciled to ensure consistent results. They are characterised as an evolution from individual methods to hybrid methods in medical image analysis due to the increasing complexity of images and the high demands of the applications in the clinic. Clinical settings require models that are accurate but also interpretable, stable, and capable of handling real-world variability. Hybrid approaches meet these requirements more closely by providing stronger resistance to data non-regularity, more flexible support for various types of imaging data, and the embeddedness in diagnostic routines.

5.1.2 Hybrid Approaches in Medical Image Analysis

In the past few years, hybrid methods have attracted much attention in medical image processing, as they can alleviate the limitations of single model approaches [96]. Both works exploit several methodologies to improve the model and may enhance each other, but in different ways.

Hybrid models combine different techniques; they usually mix something like traditional ML with DL or combine different NN models. These models are also a

motivation for models that incorporate the complementary strengths of multiple weak models, allowing us to compromise on handling specific tasks for some instances. Hybrid models are particularly beneficial when heterogeneous datasets have to be combined or several analysis approaches are used [97].

A typical cross-model is the hybrid model in medical image analysis, such as CNN and traditional classifiers (SVM, for example), which are combined. In this approach, a CNN is applied to obtain features of interest from medical images, and a traditional classifier (e.g., SVM) is used to predict the ultimate decision [98-100]. This strategy takes advantage of the CNN's high capability in extracting spatial features and the SVM's suitability for dealing with high-dimensional feature space, consequently enabling more reliable and faster classification.

Besides integrating diverse ML paradigms, hybrid models also consist of several neural network architectures. For example, a mixed approach combines a CNN for spatial processing and an RNN for temporal or sequential processing. This is especially useful in medical imaging tasks associated with dynamic or time-based data series, like tracking disease progressions or treatment effects over time [101].

Hybrid models combine the complementary advantages of various methods and can handle broader tasks more efficiently. One of the main advantages of the learning methods under consideration is that hybrids can be tailored for specific challenges related to the domain in which they are employed: Hybrid Network, CNNs for spatial, and RNNs for temporal analysis. Fuse the sequential information into a Hierarchical loose-coupling sequencing information hierarchical model of the medical images: dynamic disease monitoring. This flexibility allows hybrid models to be designed for a wide range of specific tasks, such as multi-modal imaging data analysis, for which multiple types of imaging data must be combined to diagnose accurately [102]. Additionally, hybrid models can also enhance interpretability due to the integration of the complementary aspects of traditional ML approaches (traditionally being more interpretable) and the deep feature learning capabilities of DL models.

5.1.3 Enhancing DR Image Classification through Hybrid Approaches

Classification of DR is difficult and complex since the stages of the disease are not well defined, and in the image quest of clinicians, the separation between one stage and

the next is subtle in some cases. Hybrid methods have been widely applied to address these issues, and some advantages provide a basis for enhancing DR classification models, such as those related to accuracy, interpretability, and robustness [103,104].

- **Feature Enrichment**

Enriching the feature representations is one of the most important advantages of hybrid strategies. By integrating CNN with other technologies, hybrid models capture more comprehensive features from fundus images. CNNs are particularly good at extracting these local features, MAs and HEMs, in the details, which is important for the diagnosis in the early stages of DR. At the same time, DL models such as CNN can be combined with other methods such as other classic ML algorithms, or different neural network model structures, to form hybrid models that can learn both global features (e.g., health of the whole retina) and local features (e.g., some specific lesion or abnormality of the eye). This enriched feature extraction helps to capture the fine differences among those DR stages.

- **Improved Accuracy**

Hybrid methods also help increase classification accuracy for DR. By combining DL and classical methods or other NN architectures, hybrid models can take advantage of both. For instance, the discriminative extraction of features in CNNs is cascaded with the accuracy of SVM at the last step of classification. These three integrated network blocks allow the model to distinguish better the subtle changes present in fundus images of various stages of DR, and a more accurate and established model is established, with excellent diagnostic performance at a high level.

- **Enhanced Interpretability**

Interpretability is immensely valuable for the clinical adoption of ML models. Visualisation and explanation techniques like Gradient-weighted Class Activation Mapping (Grad-CAM) are commonly practised for any hybrid model. Grad-CAM locates the portions of the fundus images where the model focuses on making predictions, for example, in areas of HEM or retinal lesions. These visual explanations also increase trust and transparency, which are crucial for clinicians who know why a model made its predictions before making important decisions about patient care.

5.2 EXISTING HYBRID MODEL ARCHITECTURE FOR DR CLASSIFICATION

In the existing literature, DenseNet121, Xception, and EfficientNetB3 models have demonstrated high performance in various image classification tasks, particularly in medical imaging [105]. Building upon these architectures, a hybrid model is designed, trained, and explicitly tested on custom dataset where training and validation is done using 3,028 images from the Kaggle platform (2019) and tested using 339 images from Kathir Eye Hospital. The Existing hybrid architecture is shown in Figure 5.1.

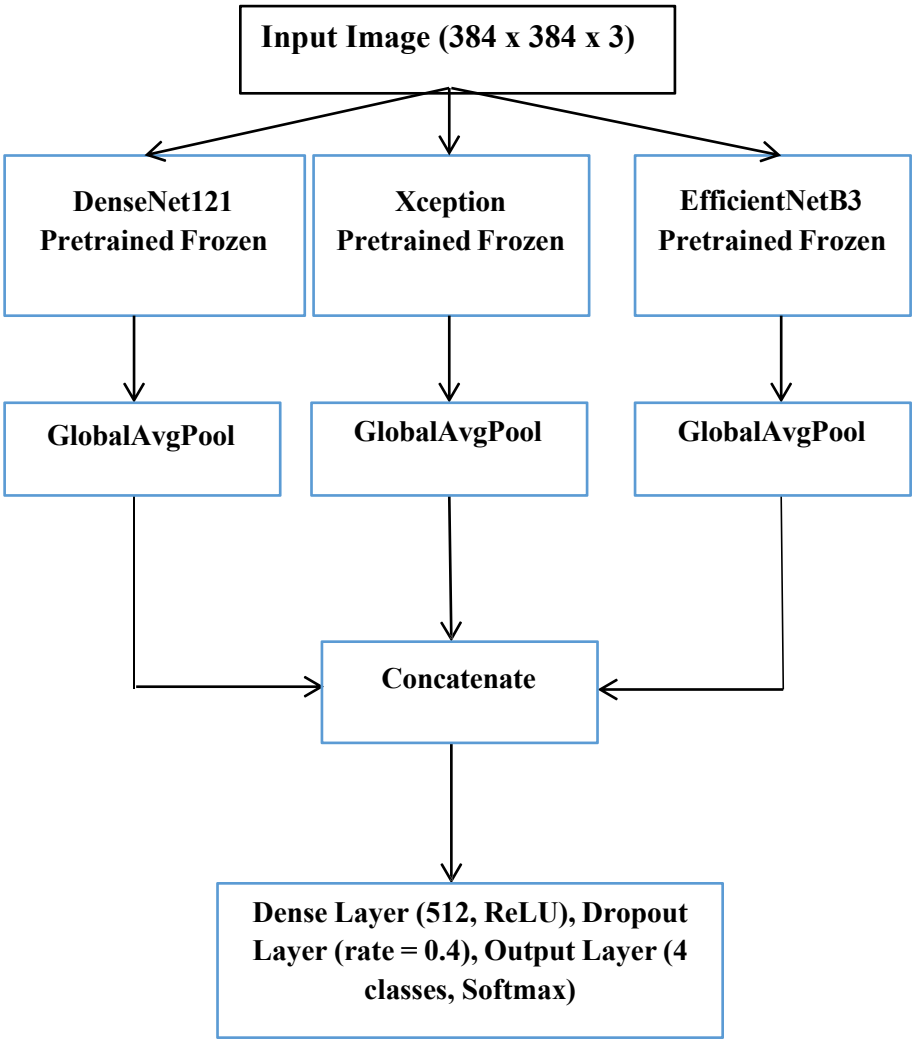

Figure 5.1 Existing Hybrid Model Architecture

Each backbone network (DenseNet121, Xception, and EfficientNetB3) is initialised with ImageNet pretrained weights and is employed as a feature extractor by

removing the classification head (include_top=False). During training, these base networks are frozen to retain their learned representations. Feature maps from each model are processed through GAP, and the resulting feature vectors are concatenated to form a unified feature embedding. This combined vector is then passed into a FC layer of 512 units with ReLU activation, and a dropout layer of rate 0.4 to avoid overfitting.

The last layer is a dense layer with four neurons of which four are output neurons that are activated with softmax indicating the four classes in the dataset. Adam optimiser and sparse categorical crossentropy loss function are employed for training the model, which is best suited for integer-encoded labels in multi-class classification. The RGB images are resized to 384×384 pixels and are taken as input images. This hybrid approach utilises the complementary strengths of the existing models, enhancing feature diversity and overall performance on the custom eye disease dataset.

5.3 DEVELOPMENT OF HYBRID MODEL ARCHITECTURE

DR is a significant trigger for blindness around the world, and early detection effectively treats and controls the disease. In recent years, DL approaches such as CNNs have achieved important advances in analysing medical images, particularly in detecting and classifying DR stages from fundus images.

This work introduces a hybrid DL model based on a pre-trained EfficientNetV2L model and custom convolutional layers. It utilises EfficientNetV2L's TL capability and the flexibility of the custom CNN layers to obtain an accurate and robust DR classification. EfficientNetV2L acts as the base learner due to its prior success in attaining high accuracy with computational efficiency.

5.3.1 Hybrid Model Architecture

In the proposed hybrid architecture, both TL and custom convolutional feature extraction techniques are combined to enhance classification performance. The model input is an RGB image of size 384 × 384 × 3, which is sufficient for spatial resolution of feature learning. The input image is provided in parallel to two branches: one on EfficientNetV2L, an extremely sophisticated pretrained backbone, and another constructed from a custom CNN. The aim is to learn both common characteristics from large data as well as task-specific characteristics pertinent to the target set.

As identified in chapter 3, the EfficientNetV2L model gave the better validation accuracy among all the other fine-tuned models. Hence it is chosen as the TL module for this hybrid approach. The EfficientNetV2L model is pretrained with ImageNet weights, and the classification head is stripped with include_top=False. For the prevention of overfitting and the preservation of the learned representations, all the base model's layers are frozen. The feature maps of the last convolutional layer are converted into a 1D feature vector using GAP without semantic information reduction but the spatial dimensions.

Meanwhile, a simple yet efficient Custom CNN-5 model is used, which in chapter 4 is found to be the most performing custom model. Custom CNN is made up of five separable convolutional layers (SeparableConv2D), each of which is followed by BN and MaxPooling2D. Separable convolutions reduce the number of parameters substantially without limiting the capacity of the model to learn spatial hierarchies. The CNN module is designed specifically to the characteristics of the dataset so that it is able to learn local and fine-grained features. The final convolutional output is sent through GAP, which also yields another 1D feature vector.

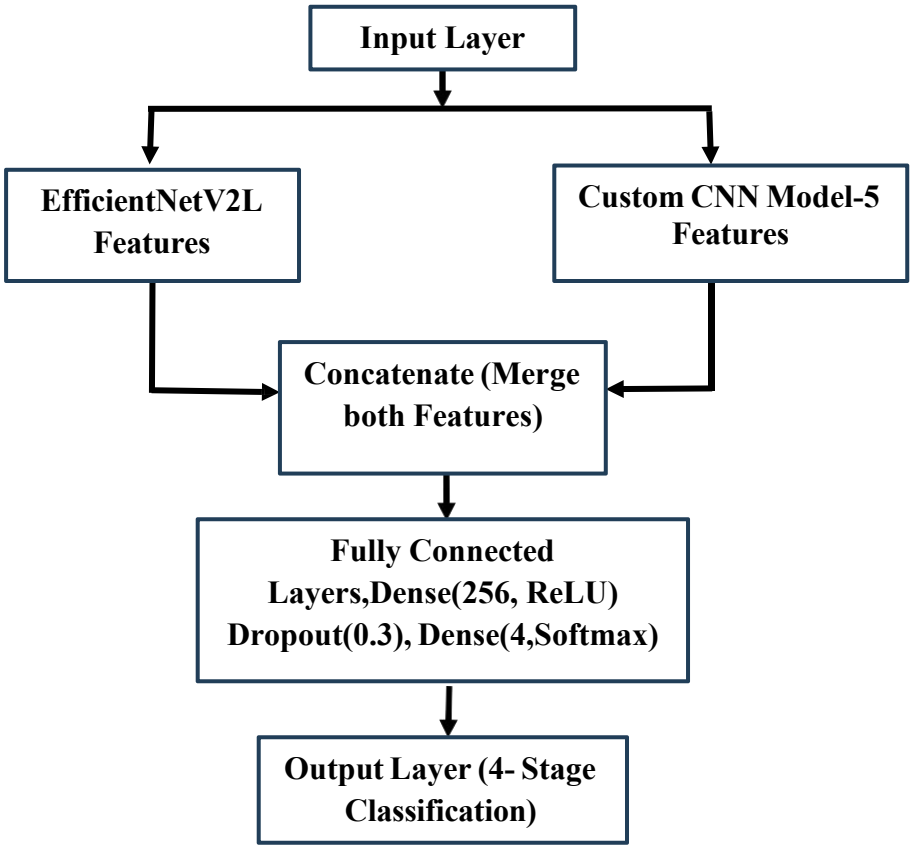

Figure 5.2 Flow chart of Hybrid Model Architecture

EfficientNetV2L and Custom CNN-5 branch outputs are concatenated through a Concatenate() layer to combine global and task-specific features into one representative. This combination draws on the benefits of both worlds: EfficientNetV2L's abstract, deep features and local pattern sensitivity of the custom CNN. The flow chart of hybrid model architecture is shown in the figure 5.2.

The concatenated features are then fed through FC (dense) layers for further fine-tuning and final classification:

The first dense layer with an activation function of 256 units is a ReLU. This layer aggregates the extracted features to learn higher-level data representations for the holistic perception of the input image.

A 30% dropout rate is used to avoid overfitting by making a random zero set of the neurons during training. This also aids in generalising the model so that it generalises unseen data well.

The last dense layer has 4 units representing the 4 DR stages: normal, mild, moderate, and severe. For the output layer, a softmax activation function interprets the output values as probability scores per class: the DR stage with the highest probability score is assigned as the predicted DR stage.

5.3.2 Training process and Hyperparameter Configuration

This hybrid model's training involves tuning important hyperparameters to achieve the performance level for the DR classification task.

Optimiser (Adam): The Adam optimiser efficiently updates weights during training. It adjusts the learning rate according to the gradient, aiding in faster convergence. The learning rate is set to 0.0001 to ensure stable updates, which helps prevent overfitting and promotes rapid model convergence.

Loss Function (Sparse Categorical Cross-entropy): Since the model performs multi-class classification, sparse categorical cross-entropy is used as the loss function. This function compares the predicted class probabilities with the proper labels, penalising incorrect predictions based on the actual class distribution. Sparse categorical cross-entropy is particularly useful for this task because it allows us to use integer labels (instead of one-hot encoding) for four-class classification.

Activation Function (Softmax - Output Layer): Softmax is applied to the output layer to convert the raw output into probabilities, guaranteeing that the sum of the output probabilities equals 1.

Dropout Rate (0.3): To prevent overfitting, a 30% dropout rate is applied after the first dense layer. Dropout is a regularisation technique that helps the model generalise unseen data better. By randomly setting a portion of the neurons to zero, dropout forces the network to learn more robust features, reducing the likelihood of overfitting the training set.

Batch Size (8): The model is trained using a batch size of 8, balancing computational efficiency with model performance. Smaller batch sizes allow faster updates but introduce noise into the gradient estimates. A batch size of 8 is commonly used for DL models as it provides a good trade-off between training speed and stability.

The Adam optimiser and the sparse categorical cross-entropy loss are used to compile the model. The model is trained on a DR dataset for some epochs, and the weights are updated to cause the loss function to take small values. The model is tested on the validation set at the end of each epoch, and training stops when the model reaches optimal performance on the validation data.

Considering the model's capacity to manage complex image data and correctly identify DR stages, it might be a beneficial tool for clinical purposes. The hybrid architecture achieves high accuracy and performs well on generalising many DR images by leveraging pre-trained models and tailored layers.

Therefore, it is evident that this hybrid DL model merges feature extraction properties of EfficientNetV2L with custom CNN layers to provide a hybrid solution that is powerful and efficient in terms of DR classification.

Given that the model is designed with a combination of TL and customised layers, the model can accommodate challenges in the DR classification and make accurate predictions efficiently. This hybrid network improves resilience and accuracy of the model, which monitors the progression of DR. This architecture with a blend of both worlds, Pre-trained DL models and custom network, opens up avenues for more work on automatic diagnosing of medical conditions and more advanced use of AI in medicine.

5.4 DATASET

The APTOS 2019 Blindness Detection dataset and a custom dataset are used for training, validation, and testing, respectively, in this study. A custom dataset is also established, in addition to the APTOS dataset, to evaluate the model stability and performance. The custom dataset contains images annotated similarly to the APTOS dataset. It is developed with real-time images at Kathir Eye Hospital in Tiruchengode after obtaining approval from the Institutional Human Ethics Committee. The dataset distribution across training, validation, and testing is shown in Table 5.1.

Table 5.1: Dataset split

Dataset	Total Images	Mild	Moderate	Normal	Severe
APTOS Training (70%)	2,356	259	699	1,263	135
APTOS Validation (20%)	672	74	199	361	38
Testing (10%)	339	37	101	181	20

The training and validation datasets retain the original partitioning from APTOS, assuring consistency in data distribution. The custom dataset used for testing follows the same proportionate split, allowing for a fair evaluation of the model's generalisation ability on unseen data.

Maintaining the same partitioning strategy, the model is trained on a well-established dataset and evaluated on a custom dataset to test its robustness and adaptability to real-world variations.

5.5 DATA PREPROCESSING AND TRAINING PIPELINE FOR MODEL EVALUATION

Image processing is a crucial preprocessing step in DL that enhances raw image data, making it more suitable for model training and analysis.

In DL applications, particularly CV tasks, images may be taken in diverse resolutions, lighting conditions, and orientations. Image processing methods like data

cleaning, resizing, augmentation, image enhancing methods, and normalisation ensure conformity and improve model accuracy.

Preprocessing: After dividing the dataset into training, validation, and testing sets, it is further preprocessed to ensure quality and balance. Corrupt images are also removed from the dataset using the norms standard in CNN-based models. All the bad and unreadable images are checked out and deleted to avoid possible noise during model training. This ensures the cleanness and reliability of the dataset for DL applications.

Resizing: After data preprocessing, all images from the APTOS and custom datasets are resized to a resolution of 384×384 pixels. This resizing is necessary as EfficientNetV2-L expects images of this particular size as input and performs best. Resizing images across all the datasets ensures consistency and avoids shape mismatch, helping the model extract richer features.

Augmentation: Data augmentation methods were used to further balance the training dataset and address the class imbalance. Data augmentations like rotation, flipping, and zooming artificially augment the number of images, making the model more representative of new data. The number of images per class is also oversampled to keep the class-to-class bias within the set limits. These pre-processing steps can result in better model performance, model robustness, and thus better classification. After performing the data augmentation, the balanced dataset is shown in Table 5.2.

Table 5.2 Balanced dataset

Split	Total Images	Mild	Moderate	Normal	Severe	Total
Training	2,356 (70%)	1,263	1,263	1,263	1,263	5052
Validation	672 (20%)	74	199	361	38	672
Testing (Custom dataset)	339 (10%)	37	101	181	20	339
Total	3,367 (100%)	1374	1563	1,805	1321	6063

5.6 IMAGE ENHANCEMENT TECHNIQUES FOR THE HYBRID MODEL

The APTOS and custom datasets differ in multiple aspects, necessitating specific preprocessing steps to maintain consistency. Each preprocessing methods improve image quality prior to entering the data into a DL model.

During training, several preprocessing steps are performed to enhance image quality after applying oversampling techniques to balance the dataset. Brightness enhancement improves overall visibility, assuring that key features are not lost in poorly lit images. Contrast Limited Adaptive Histogram Equalisation (CLAHE) is utilised to enhance local contrast, making subtle details more distinguishable. Gaussian blurring is used to suppress unwanted noise while preserving important structures, and sharpening is applied to enhance edges, making critical features more prominent [106]. These preprocessing steps ensured the training data is optimised for effective learning, improving the model's generalisation capability across diverse samples.

For validation and testing, the same preprocessing steps are applied to maintain consistency with the training set, assuring that the model's evaluation is not affected by variations in image quality.

The detailed explanation of the preprocessing is as follows:

5.6.1 Brightness Enhancement

Brightness Enhancement is a preprocessing method that enhances the visibility of image details by amplifying the average image luminance. This is extremely useful for images that are too dark, which can confuse models due to insufficient feature extraction. The intensity values of pixels are multiplied to achieve the adjustment of brightness, which is given in Equation 5.1.

$$I' = I \times \alpha \tag{5.1}$$

where,

- I is the original pixel intensity.
- α is the brightness factor (here, 1.5).
- I′ is the new pixel intensity after enhancement.

This operation assures that darker images become more visible without losing significant information. Over-enhancing brightness causes loss of detail in highly illuminated regions. This is achieved using image enhancement libraries, such as PIL (Python Imaging Library), which applies a scaling factor to increase pixel intensity values. This technique assures that subtle features in the image are more distinguishable, contributing to better feature extraction during model training.

5.6.2 CLAHE

The CLAHE technique is used, which enhances local contrast while preventing over-amplification of noise. Unlike Global histogram equalisation, which applies contrast enhancement across the entire image, CLAHE divides the image into small tiles (regions) and applies histogram equalisation locally. This helps in preserving details in both bright and dark areas.

The formula is given in Equation 5.2:

$$I'(x,y) = \frac{CDF\ (I) - min(CDF)}{max(CDF) - min(CDF)} * 255 \qquad (5.2)$$

where,

- *I'(x,y)* is the new intensity of the pixel at position (x,y) after CLAHE.
- *CDF(I)* is the Cumulative Distribution Function (CDF) of pixel intensities within a local tile.
- Max(CDF) and min(CDF) are the tile's maximum and minimum CDF values.
- 255 is used to normalise the intensity values for an 8-bit image.

This formula ensures that the pixel intensities in each local tile are spread across the full intensity range, enhancing contrast while maintaining local details. However, to avoid over-enhancement, CLAHE uses a clip limit to restrict the contrast amplification. In the code clip, Limit=2.0 sets the threshold for limiting contrast enhancement, preventing excessive noise amplification, and tileGridSize=(8,8) divides the image into 8×8 small regions, ensuring localised contrast enhancement. The method is applied to the L (luminance) channel in the LAB colour space, preserving colour information while enhancing brightness and contrast.

5.6.3 Gaussian Blur

Gaussian blur is a widely used technique in image preprocessing that reduces image noise and smooths out variations in intensity. It is achieved by convolving the image with a Gaussian function, where pixel values are averaged based on a weighted distribution. The kernel size controls the effect, which determines the extent of blurring. Gaussian blur is used to reduce noise and smooth the image using a convolution operation, as given in Equation 5.3.

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{^2 + ^2}{2\sigma^2}} \quad (5.3)$$

where,

- $G(x,y)$ is the weight for each pixel in the convolution filter.
- σ is the standard deviation of the Gaussian distribution.

A higher σ results in stronger blurring. The filter size is usually chosen based on the noise level and required smoothness. Gaussian blur helps eliminate high-frequency noise, reduce sharp transitions in the image, and improve the performance of edge detection algorithms. This is particularly useful in DL, where excessive noise hinders the feature extraction.

5.6.4 Sharpening

Sharpening is a process to sharpen the edges and fine details of an image by introducing high-frequency components. This is typically achieved using a convolutional kernel, such as the Laplacian or unsharp mask, accentuating the contrast between adjacent pixels.

Sharpening is performed using a kernel that enhances edges by emphasising intensity differences, and it is given in Equation 5.4.

$$I' = I * K \quad (5.4)$$

where,

- I is the original image,
- K is the sharpening kernel given in Equation 5.5:

$$K = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad (5.5)$$

- * denotes a convolution operation.

This sharpening kernel increases the contrast of edges by subtracting blurred values from the original image and adding a weighted centre pixel value. This makes structures such as blood vessels and lesion borders more distinguishable.Sharpening is particularly useful in object detection and medical imaging applications, where fine structural details must be preserved. Figure 5.3 shows the sample fundus image after applying image enhancement techniques.

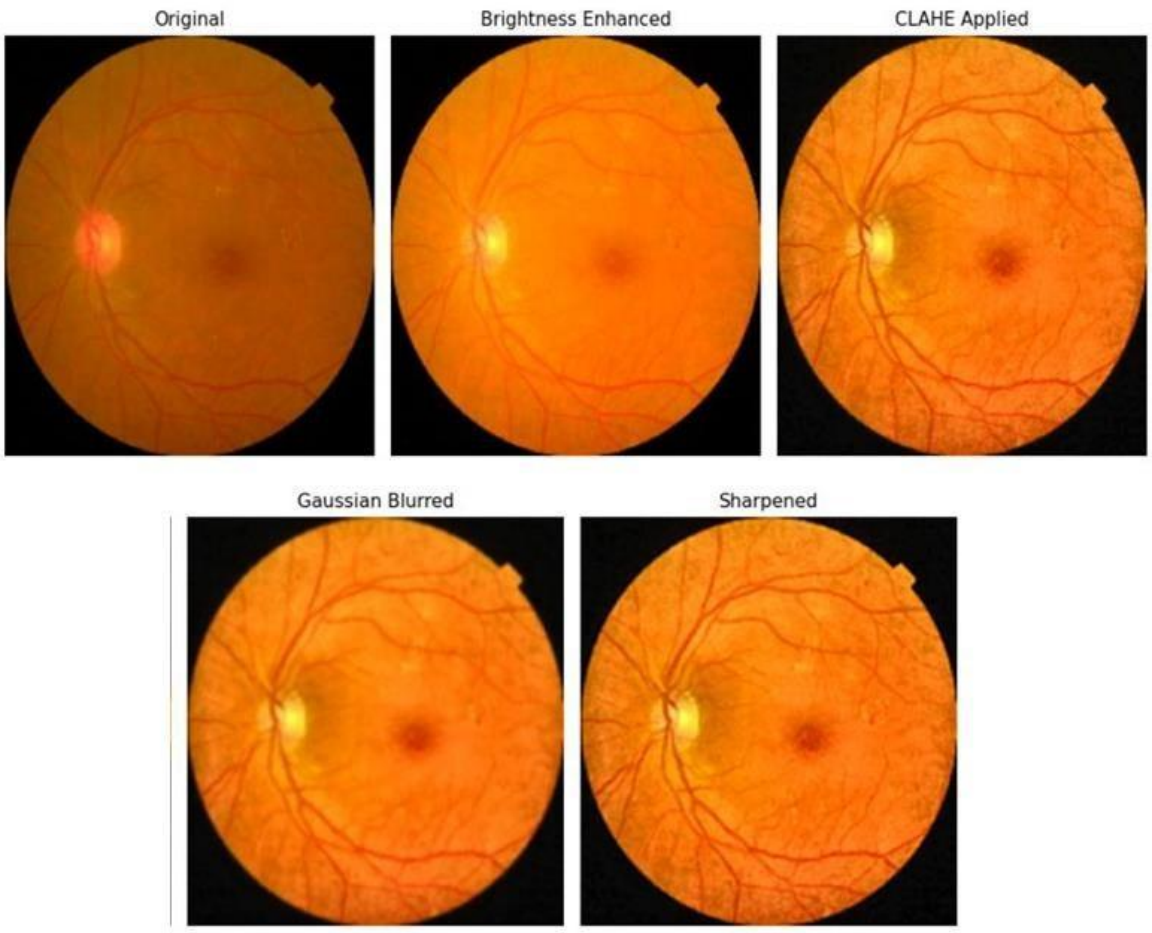

Figure 5.3 Sample Fundus Image after Preprocessing Techniques

By improving the clarity of edges, sharpening ensures that DL models extract more meaningful features, thereby enhancing overall classification accuracy.

At last, normalisation is done, which scales pixel values, typically between 0 and 1, improving numerical stability during training. This step is fundamental in various applications, including medical imaging, object detection, and facial recognition, where precise and reliable image analysis is important.

5.7 RESULTS

The four models finetuned efficientV2L, Custom CNN Model_5, Existing hybrid model and proposed hybrid model are trained over 10 epochs, which ensures they have enough iterations to extract meaningful patterns from the dataset without overtraining and overfitting the dataset.

Figure 5.4 compares loss across the finetuned EfficientV2L, Custom CNN Model_5, the hybrid model in the literature, and the proposed hybrid model, which shows differences in training behaviour and performance efficiency.

For the finetuned EfficientNetV2L model, both gains in loss decrease linearly across the 10 epochs.

The beginning of the loss in this first epoch is around 1.0800, which shows a reasonable fit to the data. The model keeps improving as the number of epochs increases; the loss falls to 0.3654 by the tenth epoch.

This phenomenon indicates that the model has learned well and is optimising its parameters, resulting in a reduced loss and improved performance on the validation set.

The prepared model, Custom CNN Model_5, has a loss of 0.7304 in epoch one, showing that it is not as well optimised from the start but the model gets better after training.

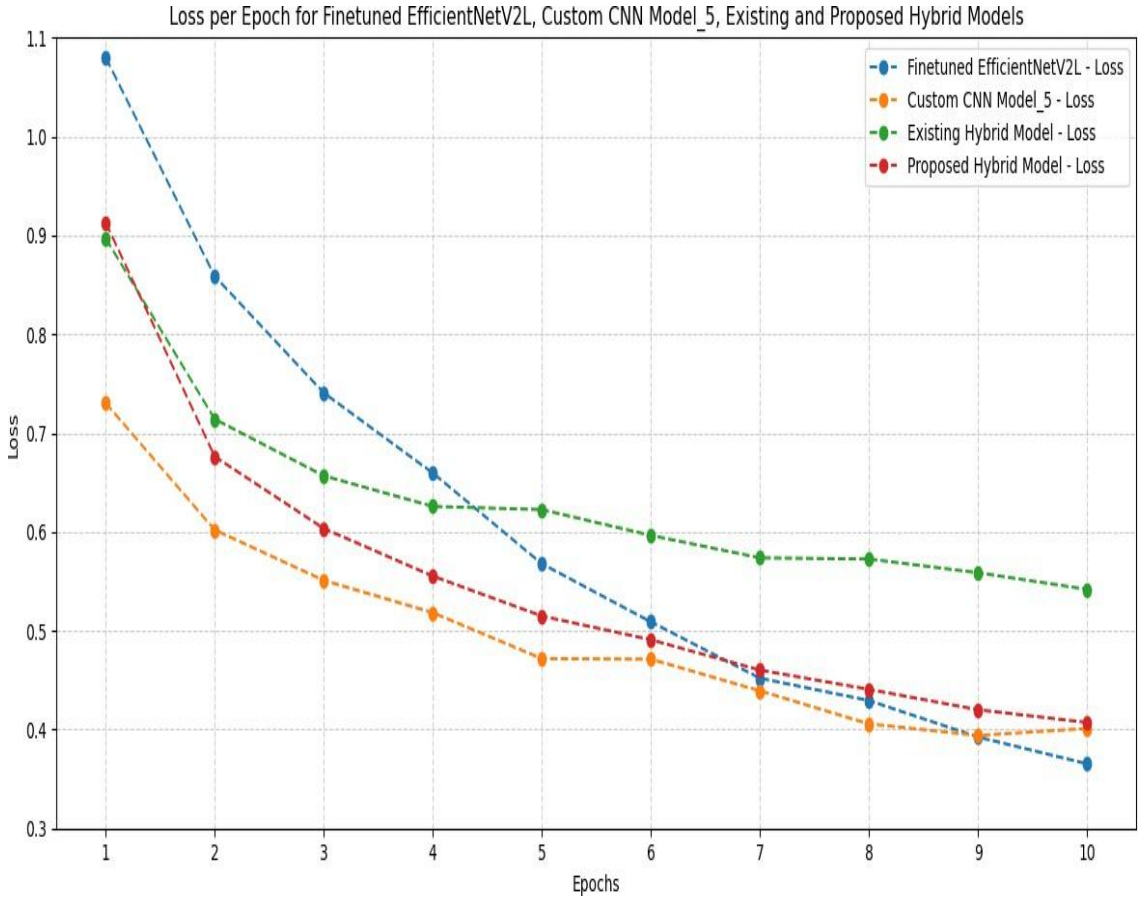

Figure 5.4 Training Loss of four models

Furthermore, the loss decreases rapidly in the first few epochs and caps out at 0.4011 in the last epoch. The drop in loss indicates that the Custom CNN model is learning, aligning, and tuning its weights, even if not to the same final loss as the Finetuned EfficientNetV2L model.

The Existing Hybrid Model's loss starts at 0.8970 from the first epoch, the greatest among the other models, as the Model shows its less-than-optimal initial performance. Nevertheless, the loss continuously goes down when trained.

After the 10th epoch, the loss is 0.5419. The decrease in loss with the passage of work underscores the fact that the hybrid model is improving. However, this hybrid model is still far behind the Finetuned EfficientNetV2L and Custom CNN models regarding loss minimisation.

Last but not least, at epoch 1, the Proposed Hybrid Model has a loss of 0.9131, which is also consistent with the Existing Hybrid Model. This model's loss decreases gradually in the training process and eventually reaches 0.4072 by the end of the 10th epoch.

The performance gain indicates that the hybrid model is learning to combine the predictions, but it cannot beat the basic Finetuned EfficientNetV2L and Custom CNN models for the final loss. However, the Drop in loss means the proposed hybrid model is also moving toward better data fitting.

Figure 5.5 shows validation loss for four models. When fine-tuning the model using Finetuned EfficientNetV2L, a steady loss reduction can be seen across the validation loss (0.9588 at epoch 1 and 0.3017 at epoch 10). This suggests that the model is learning to generalise and converge over time.

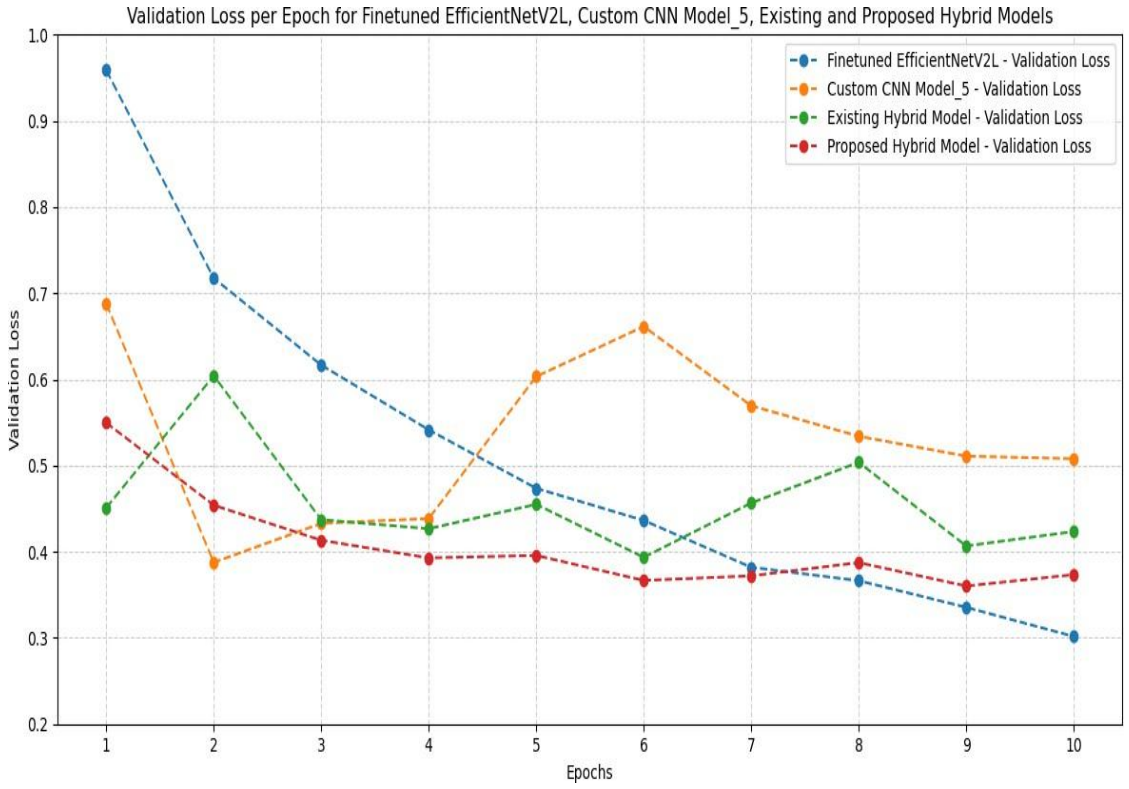

Figure 5.5 Validation Loss of four models

The Custom CNN Model_5 opens with a validation loss of 0.6880 and fluctuates slightly across the epochs before settling down to 0.5080 by the 10th epoch. Although it

improves with training, the model never reaches the same low validation loss as the Finetuned EfficientNetV2L model, suggesting its performance is less efficient.

The Existing Hybrid Model starts with a low validation loss value of 0.4510 after the first epoch. Performance is incrementally improved over the epochs, with the last validation loss of 0.4234. This is a sign that the hybrid also works well, but does not obtain the same optimisation level as the Finetuned EfficientNetV2L model.

The Proposed Hybrid Model continued with a validation loss of 0.5496 in the first epoch and has progressed. At the 10th epoch, the validation loss is 0.3734. While the proposed hybrid model shows a steady decrease in loss, it lags behind the Finetuned EfficientNetV2L model regarding final validation performance.

Figure 5.6 shows the analysis of training accuracy for four models. The Finetuned EfficientNetV2L model shows a steady improvement in accuracy, starting at 0.6040 in the first epoch and reaching 0.9209 by the 10th epoch. This consistent increase in accuracy demonstrates that the model is effectively learning and adapting to the data over time, achieving high performance by the final epoch.

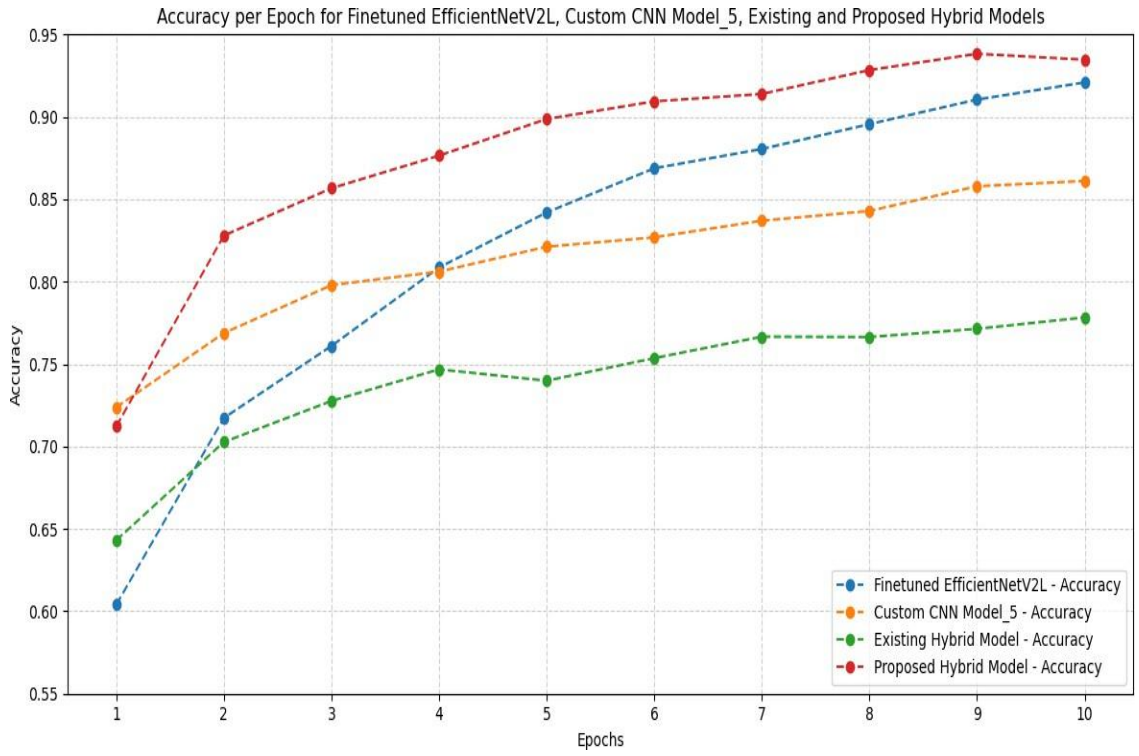

Figure 5.6 Training accuracy of four models

The Custom CNN Model_5 starts with an accuracy of 0.7238 in the first epoch, nearly identical to the Finetuned EfficientNetV2L model. However, its accuracy increases slowly, reaching 0.8611 by the 10th epoch. Despite this slower improvement, the Custom CNN model consistently demonstrates good learning capabilities throughout training.

The Existing Hybrid Model starts with an accuracy of 0.6431, the lowest among the four models, and shows slower progress compared to the other models. The peak accuracy for the hybrid model is 0.7783 (at 10 epochs), suggesting a learning progression over time, but it is lower than the Finetuned EfficientNetV2L and Custom CNN models.

The Proposed Hybrid Model has a starting point accuracy of 0.7122 and the highest progression from epochs. For epoch 10, the accuracy achieved 0.9347, beating the Finetuned EfficientNetV2-L on the task. The hybrid model can generalise and optimise more efficiently during the training period than the other models.

Figure 5.7 plots the validation accuracy of four of these networks. The Finetuned EfficientNetV2L model started with an accuracy of 0.6372 in the first epoch, but has continuously increased its validation accuracy from this epoch. The validation accuracy is 0.9289 at the 10th epoch. This progress suggests that the Finetuned EfficientNetV2L model's culture is learning and generalising well, performing well on validation.

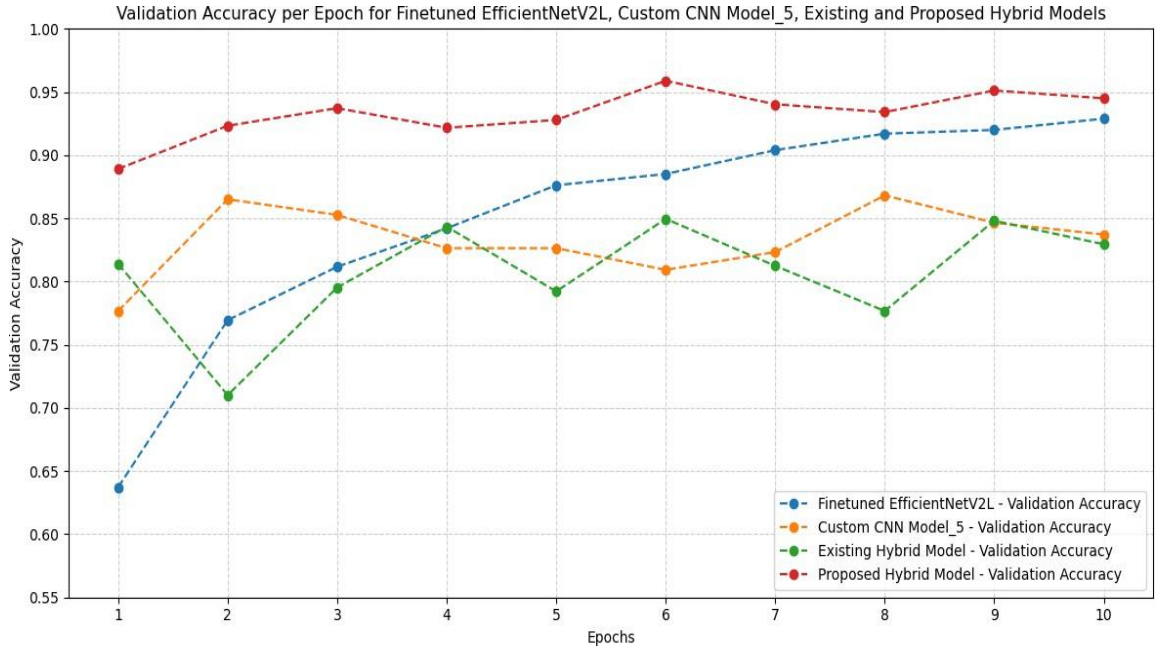

Figure 5.7 Validation accuracy of four models

The Custom CNN Model_5 starts at a validation accuracy of 0.7767, slightly higher than the Finetuned EfficientNetV2L model in the first epoch. The validation performance increases with a positive rotation score, reaching 0.8372 after 10 epochs. Although the model performs some learning during the epochs, its validation accuracy fails to match the EfficientNetV2L model, meaning there is a slower update.

The existing hybrid model's initial validation accuracy is 0.8140, which exceeds both finetuned_efficientnetv2l and Custom CNN in the first epoch. However, its validation accuracy varies across epochs, and a 0.8295 level is attained at epoch 10. Although the existing hybrid model starts with a high accuracy, its further performance is not as steady as the others, which may cause difficulties for generalisation.

The proposed Hybrid Model starts with a validation accuracy of 0.8891, having the highest initial performance among the models. This model shows substantial improvement across the epochs, with a peak performance of 0.9450 at the 10th. It is observed that the proposed hybrid model achieves the best validation accuracy, proving that it has the best learning performance and training tuning capability among all the competitors.

Table 5.3 summarises the performances of the tested models with different classification metrics and computational factors. Accuracy, precision, recall, and F1 score are presented in percentages to represent the classification performance of models. These measures offer a balanced estimation of each model's sensitivity and specificity to detect positive and negative cases.

In addition to performance metrics, the table also denotes the number of trainable parameters of each model, which indicates their computational complexity. Models that use fewer parameters usually have fewer computational requirements, but they may not be as accurate. Execution time is recorded (in seconds) to represent how fast each model trains and infers for the dataset.

The table demonstrates the trade-off between accuracy and efficiency by contrasting these models in the performance/efficiency space. This overview helps identify the best trade-offs of models when deployed in real-world applications, taking into account demanding hardware constraints and response time requirements.

The performance evaluation measures, such as accuracy, precision, recall, and F1-score, demonstrate the efficiency of the various DL models. The EfficientNetV2L + Custom CNN Hybrid model has the best scores: 94% for accuracy, 94% for precision, 93% for recall, and 94% for F1-score. The results indicate that merging the strong feature extraction, as in EfficientNetV2L, with the flexibility of custom CNN layers improves classification, which is important when dealing with complex medical images. The relatively high recall and F1-score indicate that this hybrid model is accurate and highly reliable in classifying the various types of categories in the dataset.

Table 5.3 Performance metrics for entire models

Models	Accuracy (%)	Precision (%)	Recall (%)	F1score (%)	Number of Trainable Parameters	Execution Time (S)
Finetuned VGG16 Model	85	84	85	85	7,145,604	2.7s
Finetuned ResNet-50 Model	87	86	87	87	1,318,532	1.4s
Finetuned InceptionV3 Model	89	88	89	89	656,516	1.2s
Finetuned Densenet121 Model	90	89	90	90	461,764	1.1s
Finetuned EfficientnetV2L Model	92	91	92	92	7,349,924	4.1s
Custom CNN Model - 1	45	41	45	42	15,073,504	3.5s

Models	Accuracy (%)	Precision (%)	Recall (%)	F1score (%)	Number of Trainable Parameters	Execution Time (S)
Custom CNN Model - 2	51	46	48	47	1,903,064	2.1s
Custom CNN Model - 3	60	49	57	53	1,17,834	1.3s
Custom CNN Model - 4	84	81	80	79	26,082,052	4.0s
Custom CNN Model - 5	88	86	84	84	1,13,407	1.2s
Hybrid Model – (DenseNet121, Xception, and EfficientNetB3)	75	73	75	71	2,361,860	3s
Hybrid Model (EfficientV2L and Custom CNN)	94	94	93	94	4,41,087	1.6s

Considering a general trend, the hybrid model has significant advantages over other models. For example, the well-known model DenseNet121 gives an accuracy of 90%, which is good but not as good as the hybrid model. Other models, such as ResNet-50 (87% classification), Inception V3 (89% classification), and VGG16 (85% classification), suffer from relatively poor classification.

Even CNN Model 4, which produces an accuracy of 84%, fails to match the precision and recall of the hybrid model. These results demonstrate the effectiveness of EfficientNetV2L's compact feature extraction and custom CNN layers, thus providing a more robust and generalised classification model.

Computational complexity is another factor since trainable parameters directly influence memory demand and training time. The EfficientNetV2L + Custom CNN model has 441,087 trainable parameters, balancing performance and computational feasibility.

In contrast, CNN Model 4 has over 26 million, making it significantly more resource-intensive. While the hybrid model requires more parameters than some custom CNN models (e.g., CNN Model 3 with 117,834 parameters), it is considerably more efficient than larger architectures while achieving higher accuracy. This trade-off assures that the model is deployed in real-world applications where both high performance and computational efficiency are important, making it an optimal choice for DL-based classification tasks.

Compared to Finetuned EfficientNetV2L, the hybrid shows an improvement of 2% in accuracy, 3% in precision, 1% in recall, and 2% in F1-score, while also utilising significantly fewer trainable parameters (7,349,924 vs. 4,41,087). When compared to Custom CNN Model - 5, the hybrid shows an even greater performance gain of 6% in accuracy, 8% in precision, 9% in recall, and 10% in F1-score, with only about four times more trainable parameters than Model - 5 (4,41,087 vs. 1,13,407). The results indicate that the hybrid architecture maintains an ideal balance between performance and parameter efficiency, positioning it as the most effective model among the evaluated approaches.

The hybrid model with DenseNet121, Xception, and EfficientNetB3 achieved 75% accuracy with 2,361,860 parameters and ~3s inference time. The EfficientNetV2L + Custom CNN model achieved 94% accuracy with only 441,087 parameters and ~1.6s inference time, showing superior performance and efficiency.

Confusion matrix is a key measure used to quantify the performance of classification models. It proves especially valuable in multiclass classification tasks, such as DR detection, where the objective involves classifying images into several categories (e.g., normal, mild, moderate, and severe stages) [107].

The matrix offers a detailed perspective on the model's performance by contrasting the predicted classes with the actual courses. It comprises rows and columns corresponding to the actual and expected labels, as illustrated in Figure 5.8.

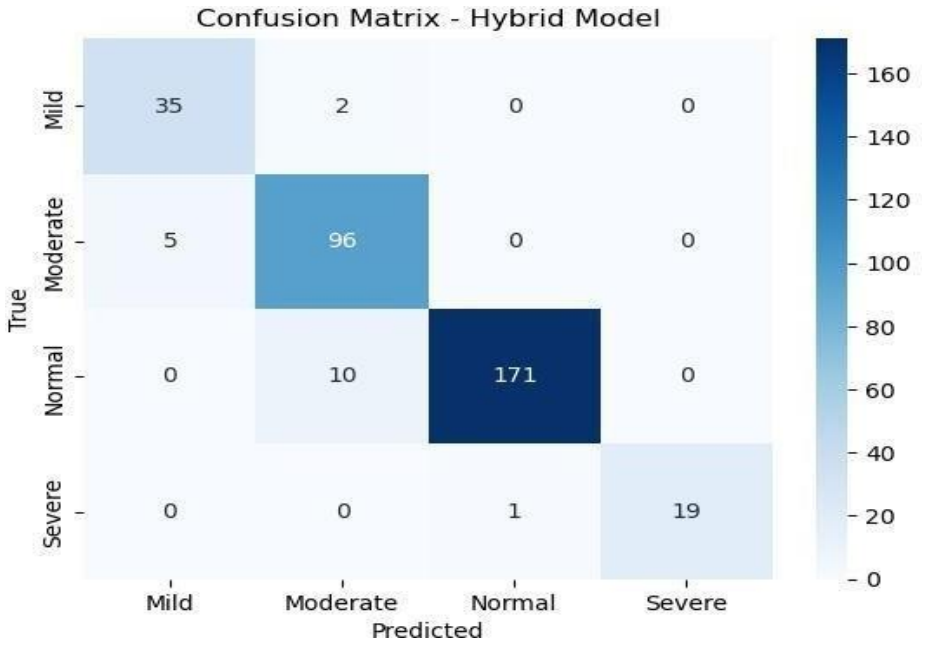

Figure 5.8 Confusion matrix for hybrid model

The diagonal entries are the numbers of correctly classified samples of each class, and off-diagonal entries are the misclassifications. In the confusion matrix of this hybrid model, the high diagonal values demonstrate that the model makes accurate predictions across all four classes, with only a few misclassifications.

From the confusion matrix, the model has classified most samples correctly across all severity levels of DR. The relatively high numbers along the diagonal, 35 for Mild, 96 for Moderate, 171 for Normal, and 19 for Severe, indicate that the model maintains consistent performance across all classes. Specifically, there are a few misclassifications and only a few cases in which a non-exemplar sample is assigned to another category. For instance, 10 Normal cases are treated as Moderate cases, but no severe misclassifications, which means the model doesn't confuse normal and severe cases. This is a balanced classification, meaning the model has no strong bias towards any class.

The consistency of the diagonal elements suggests that the hybrid model EfficientNetV2L with custom CNN layers helps discriminate various stages of DR. The model has good generalisation on the dataset and no bias toward any specific class. This evidence further confirms that the trained model is strong and is used as a DR classifier.

The confusion matrix also confirms that the proposed hybrid model works efficiently and accurately on all four DR severity levels; hence, it is effective as an automation tool for diagnosis.

The prediction accuracy measures the capability of the model to generalise to new data and new training examples. The spliced model produces the following class-wise predictions:

Normal Stage: The regular stage model performed well in predicting "normal" stage images with high accuracy, as shown in Figure 5.9. This suggests that the model effectively identifies images without signs of DR, which is important for distinguishing healthy retinal images from those showing signs of the disease.

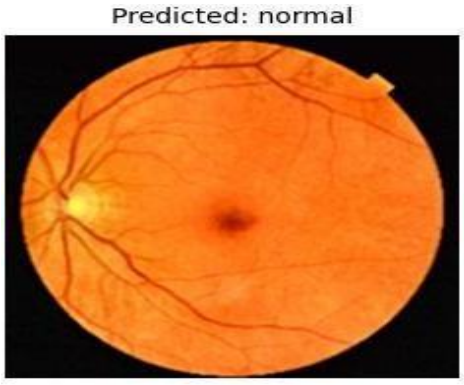

Figure 5.9 Predicted Normal DR Stage from Test Set

Mild Stage: The hybrid model performs well in classifying images with mild DR, as shown in Figure 5.10. Since the differences between the mild and normal stages are subtle, the model's success in accurately classifying these images highlights its sensitivity to early indicators of DR.

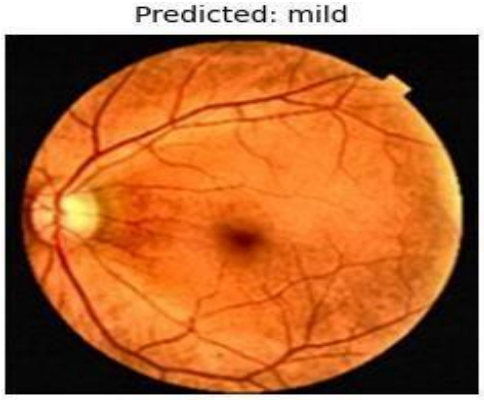

Figure 5.10 Predicted Mild DR Stage from Test Set

Moderate Stage: For the "moderate" stage, the hybrid model can identify images showing moderate levels of DR, where the retinal changes are more evident, as shown in Figure 5.11. The results for this class indicate that the model handles more complex features associated with DR progression.

Figure 5.11 Predicted Moderate DR Stage from Test Set

Severe Stage: The hybrid model also achieves strong results in classifying images of the "severe" stage of DR, as shown in Figure 5.12. At this level, retinal changes are typically severe, and the model recognises specific images that indicate severe DR.

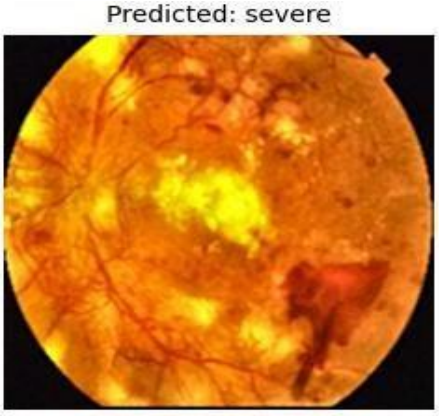

Figure 5.12 Predicted Severe DR Stage from Test Set

5.8 DISCUSSION

In the scenario of the DR classification, the hybrid model, comprising of EfficientNetV2L and custom CNN Model, is evaluated on DR grade images: normal, mild, moderate, and severe. During testing, the predicted class labels are assessed against the actual image data labels, quantifying the model's performance for classifying new

images. Once this is done, the model is fed with a batch of images from the test set, predicting each image.

The predicted images from the test set show that the hybrid model accurately classifies the different stages of DR and spots various features related to these stages. This means that it is a suitable tool for DR detection.

Thus, the hybrid model, which adopts the EfficientNetV2L architecture and the custom CNN Model which utilises advanced elements that significantly improve DR classification performance. With the advantages of EfficientNetV2L and custom convolutional layers, the feature extraction process is strengthened, the generalisation ability and adaptability are considerably enhanced, which enables it to be applied to complex image classification tasks.

One of the strengths of the hybrid model is that it can be used flexibly and adapted to different datasets. The proposed direction is that the architecture can be adapted to the model's specific requirements with custom convolutional layers, which is especially convenient for special tasks such as DR classification. The structure of the model would be adjusted as layers are added, filters increased, or proper regularisations are enforced, which is ideal for the data while obtaining superior performances in the specific application. Such flexibility makes the hybrid model versatile and reusable for other medical image classification problems or more general image processing tasks.

Accordingly, the proposed hybrid model outperforms the state-of-the-art hybrid model in achieving higher accuracy and drastically reducing the computational cost. By optimising the architecture and incorporating lightweight building blocks, the proposed model achieves better DR classification accuracy and reduces the trainable parameters, which is beneficial to computational efficiency and makes the model more applicable to a low-resource environment.