

Lung Cancer Detection using Ensemble Learning  
with Convolutional Neural Networks

Group 5: Creator’s Garage

Team Members:-

Yogesh yadav

Yyadav8 - 1233640176

Najia Khan

Nkhan42 - 1221919582

Pranav Nadimpalli

Pknadimp - 1224691624

Michael Darnall

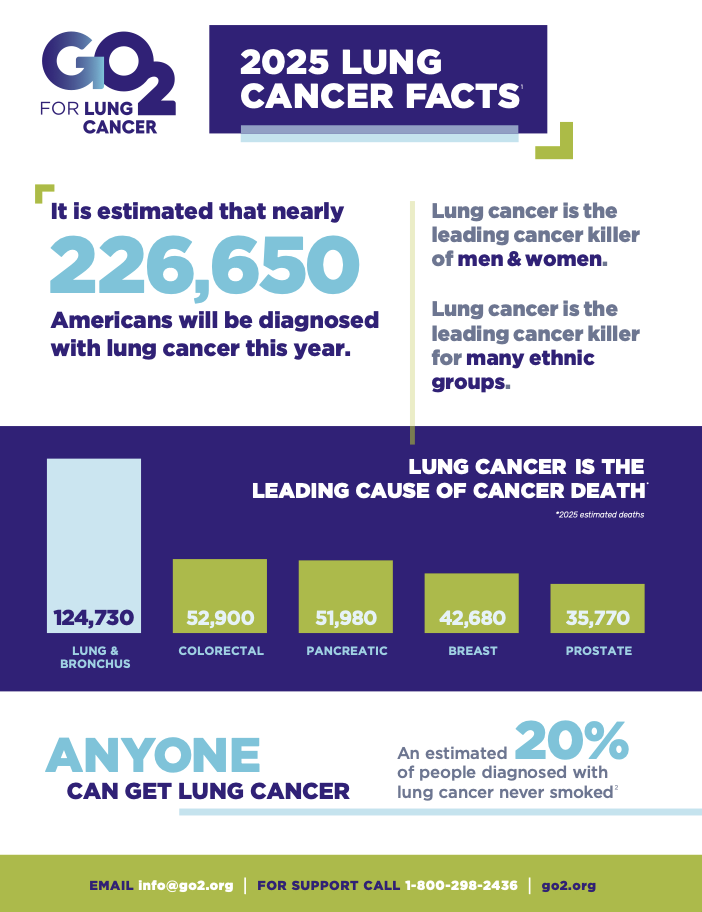
Mdarnal1 - 1221153323

Jorge Alejandro Cajiao Corro

Jcajiao - 1233462882

**Introduction**

Lung cancer remains the leading cause of cancer-related deaths in the United States. In 2022 alone, the disease claimed 131,888 lives nationwide. Despite advancements in treatment, the overall five-year survival rate for lung cancer stands at approximately 28.4%, largely due to late-stage diagnoses. Early detection significantly improves outcomes, with a 64% five-year survival rate when diagnosed at an early stage, compared to just 9% for late-stage diagnoses. ​



Computed tomography (CT) scans are the primary imaging modality for lung cancer screening and diagnosis, offering high-resolution images that are crucial for detecting early-stage tumours. However, interpreting these scans can be challenging, especially in the early stages where tumours may be small or ambiguous. Deep learning models, particularly convolutional neural networks (CNN), have shown promise in analysing medical images by learning patterns indicative of disease.​

In our project, we developed a CNN-based ensemble model designed to analyse CT scan images for early detection of lung cancer. By integrating multiple pre-trained models and employing a weighted average ensembling technique, our system mimics the collaborative approach of medical experts, enhancing diagnostic accuracy. This ensemble method has demonstrated a prediction accuracy of 99%, highlighting its potential as a robust tool for early lung cancer detection.​

**Theory**

The Convolution Neural Network (CNN) is a deep learning network architecture that learns from data by finding similar patterns in images, which recognises objects, classes, and categories. In our project, we have used five pre-trained CNN models - Resnet-50, Inception, Xception, DenseNet, and MobileNet to identify cancer patients with the help of CT-scan images.

1. Resnet50v2: This model is known as Residual Network, which is 50 layers deep with 34 residual layers. The receding gradient problem is solved by ResNet by reusing the residual blocks in the architecture. Residual blocks connect the beginning and end of a convolution block with skip connections or shortcut connections.
2. Inception V3: It is a 48-layer deep architecture incorporated with auxiliary classifiers, optimisers and Factorised 7x7 convolutions. In general, it consists of convolution filters, Pooling layers, and Relu. It can achieve high accuracy in extracting the features and classifying the images.
3. Xception: It is an adjunct of the Inception architecture, with modified depth-wise separable convolution, which is better than Inception. It is a 71-layer deep architecture.
4. DenseNet201: This model is 201 layers deep, requiring fewer parameters than the earlier CNN models. Each layer of DenseNet takes the feature maps of all the preceding layers as inputs, thereby reinforcing the feature propagation.
5. Mobilenetv2: This model was developed by Google, particularly to use in smartphones. Mobilenet revolves around the idea of using depthwise separable convolutions. This model introduces an idea of inverted residual structure and linear bottleneck, where the last convolution of a residual block has a linear output before it’s added to the initial activations.

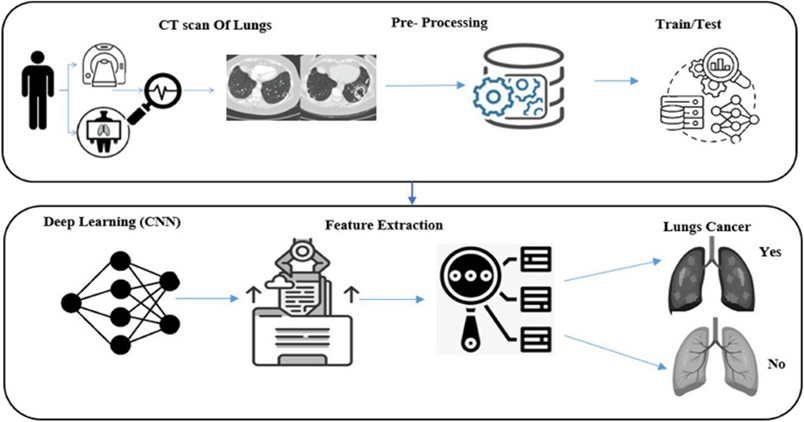
****

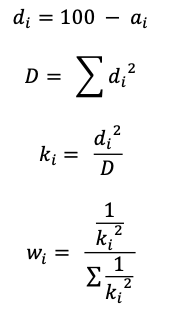
Figure 1. Steps taken in Convolution Neural Networks

**Weighted Ensembling:**

Ensemble learning methods are superior in prediction to an individual model and help in preventing overfitting. The ensembling concept is based on the simple philosophy that a collection of multiple models provides better performance compared to individual models. In our work, a weighted average of the output probabilities has been introduced as a method for ensembling. It is found to be better than the unweighted average.

In this method, if one model, say Xception, is performing better than the other four models i.e., i.e., having lower validation error, it is assigned a higher weight so that its contribution in deciding the class value is higher.

Assuming the accuracy percentage of the 𝑖 − 𝑡h model as 𝑎𝑖, the validation error is (100 – 𝑎𝑖). We define weight factor w𝑖 as below:



**Work**

In this project to train different CNN models to detect cancer, we first preprocessed the pulmonary CT scan images obtained from a dataset from Kaggle.

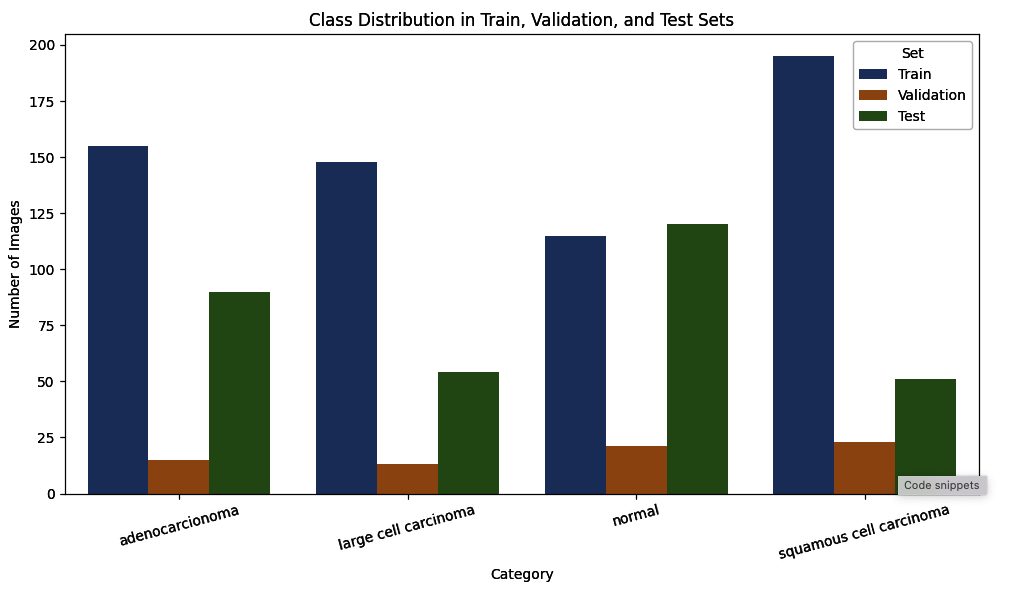
Dataset: <https://www.kaggle.com/datasets/mohamedhanyyy/chest-ctscan-images>.

Figure 2. Class distribution in train, validation, and test sets

The images in the dataset are in the PNG format. These images are divided into three portions: training, testing, and validation datasets. One small portion is retained as a validation set to test the efficacy of the trained model, while the remaining portion is divided into 5 folds. Each time, one separate fold is picked up as test data and the remaining folds are used as training data. As the first step of data preprocessing, we took each image and used the Opencv library to resize the image to a 224 by 224-pixel image. Then we reshaped each image and standardised it by dividing each pixel by 255. We converted each image into a NumPy array and created a NumPy array file for each training and test set by appending NumPy arrays of each image. These NumPy array files are used to train and test the CNN models.

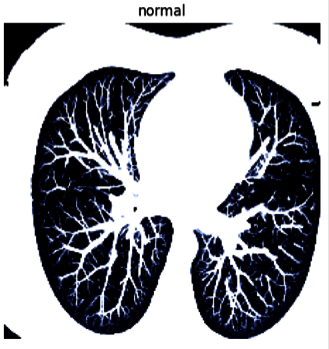
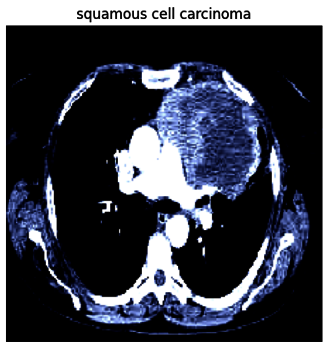
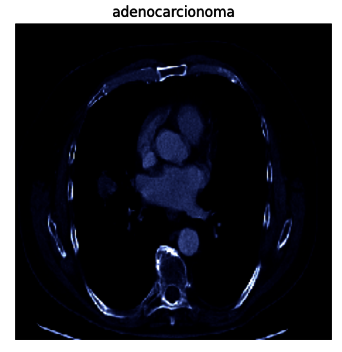


Figure 3. Examples of classified images from the dataset

**Models**

After preprocessing the image data, we used the (.npy format) data and trained all 5 models (resnet50\_v2, inceptionv3, xception, densenet201, and mobilenetv2) for 100 epochs and a batch size of 16. Below are the training loss and accuracy plots we obtained for all models.

For our data, Inception3 performed the least with a test accuracy of 92.14%, and Xception performed the best with a test accuracy of 99.80%. Based on the obtained test accuracy, we assigned weightage to each model using a mathematical formula as explained in the weighted ensembling theory above. The table below represents our accuracy stats and assigned ensembling weights for each model.

Table 1. Accuracy stats and assigned ensembling weights for each model

| **Model** | **Train Acc** | **Train Loss** | **Val Acc** | **Val Loss** | **Test Acc** | **Test Loss** |
| --- | --- | --- | --- | --- | --- | --- |
| mobilenet\_v2 | 95.16% | 0.1260 | 95.54% | 0.1234 | 99.26% | 0.0478 |
| xception | 99.80% | 0.0101 | 99.11% | 0.0341 | 99.26% | 0.0443 |
| resnet50\_v2 | 97.58% | 0.0637 | 98.21% | 0.0349 | 99.63% | 0.0219 |
| densenet201 | 97.98% | 0.0716 | 96.43% | 0.1022 | 99.26% | 0.0637 |
| inception\_v3 | 92.14% | 0.2054 | 96.43% | 0.1340 | 97.79% | 0.1204 |

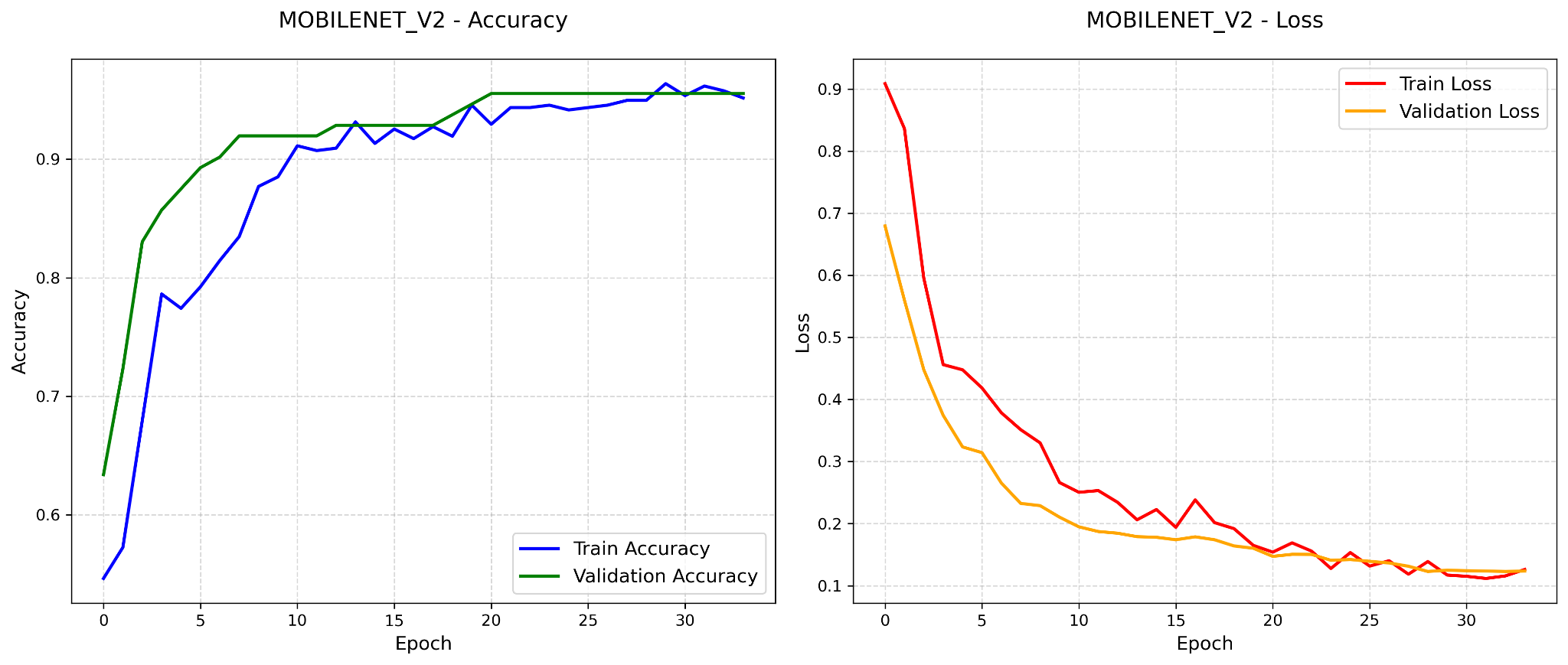


Figure 4 - MobileNet\_V2 Accuracy and Loss for Training and Validation

**MobileNetV2 (99.26% Test Acc)** – Lightweight design worked well, but shallower architecture limited peak accuracy compared to deeper models like Xception.

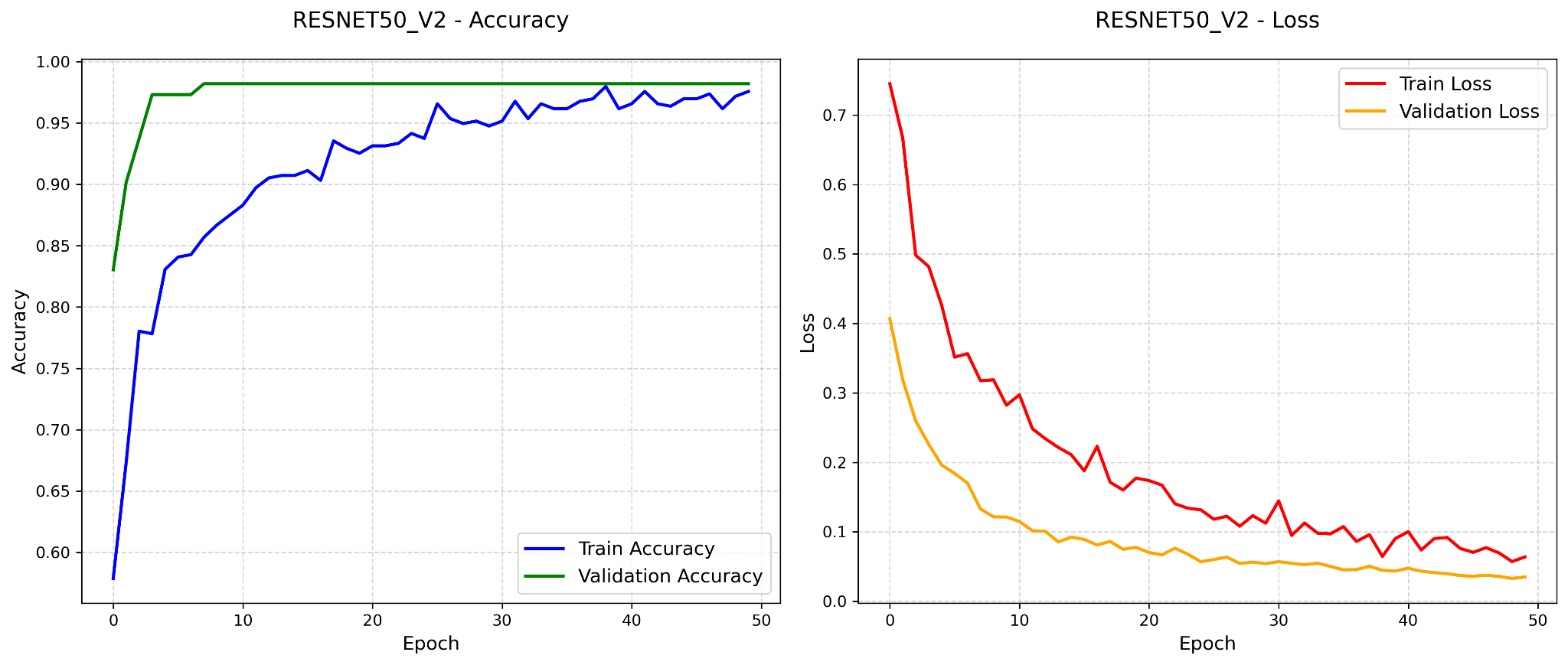


Figure 5- ResNet50\_V2 Accuracy and Loss for Training and Validation

**ResNet50\_v2 (99.63% Test Acc)** – Strong performance thanks to residual connections preventing gradient issues, making it stable for deep learning on medical images.

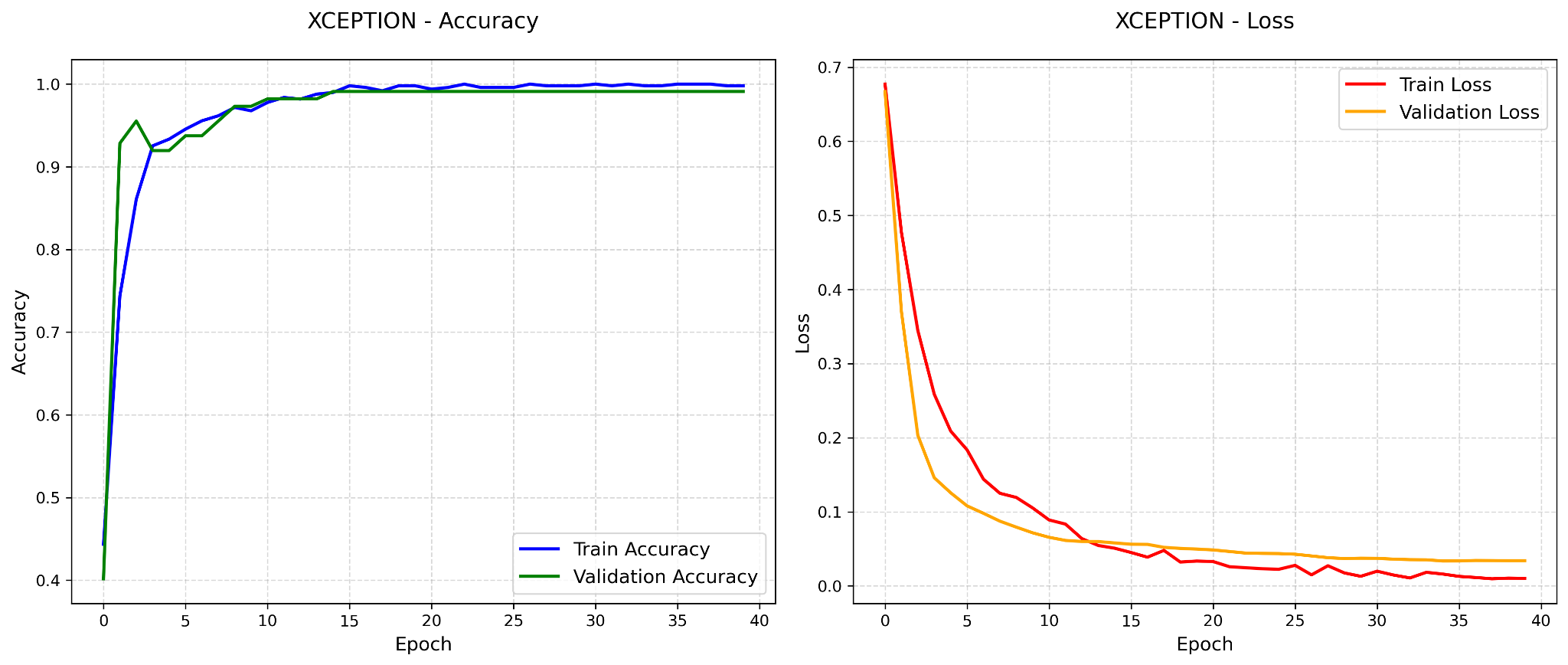


Figure 6 - Xception Accuracy and Loss for Training and Validation

**Xception (Best: 99.80% test accuracy) -** Outperformed others due to its depthwise separable convolutions that efficiently capture spatial hierarchies with minimal parameters, ideal for complex CT scan patterns.

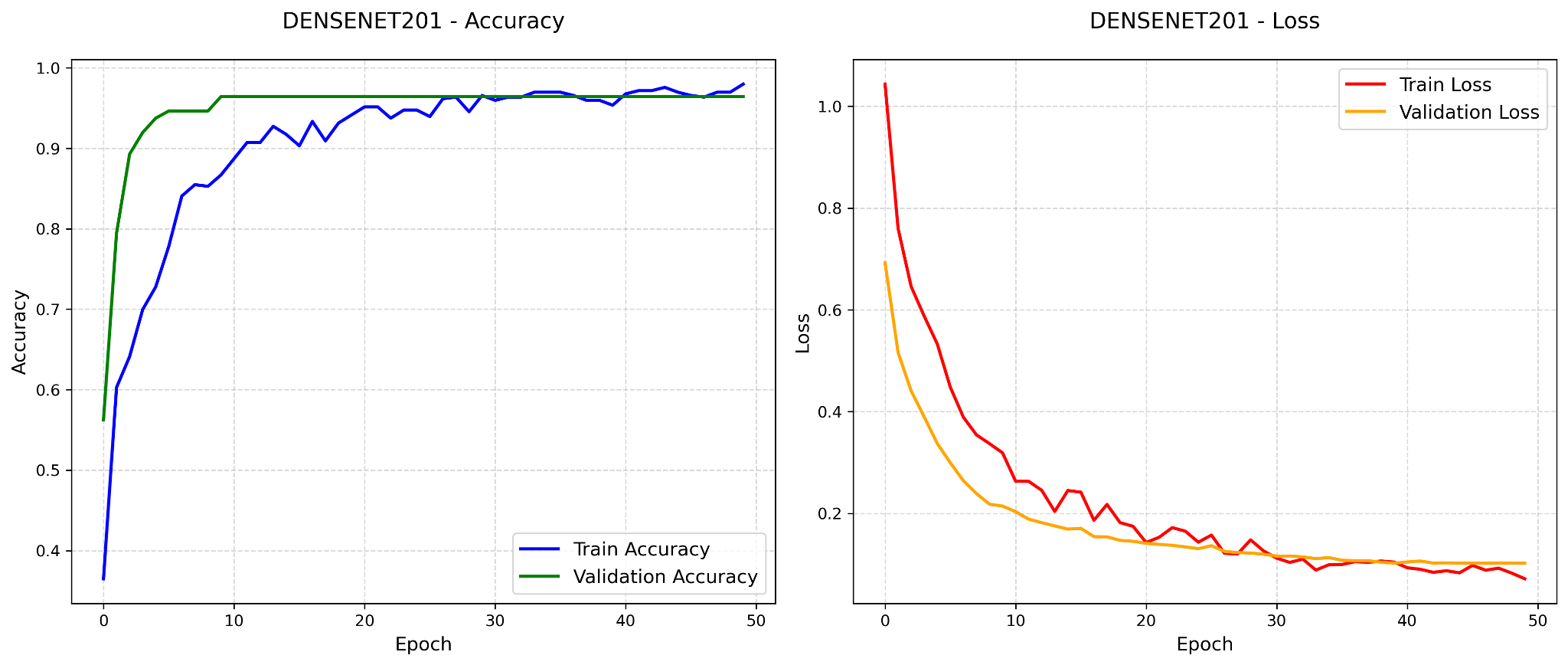


Figure 7 - DenseNet201 Accuracy and Loss for Training and Validation.

**Train Acc:** 97.98% | **Test Acc:** 99.26%

**Trade-off:** Dense connections improve feature reuse, but the **larger parameter count** (201 layers) caused a **small gap between train and test accuracy**, indicating mild overfitting.

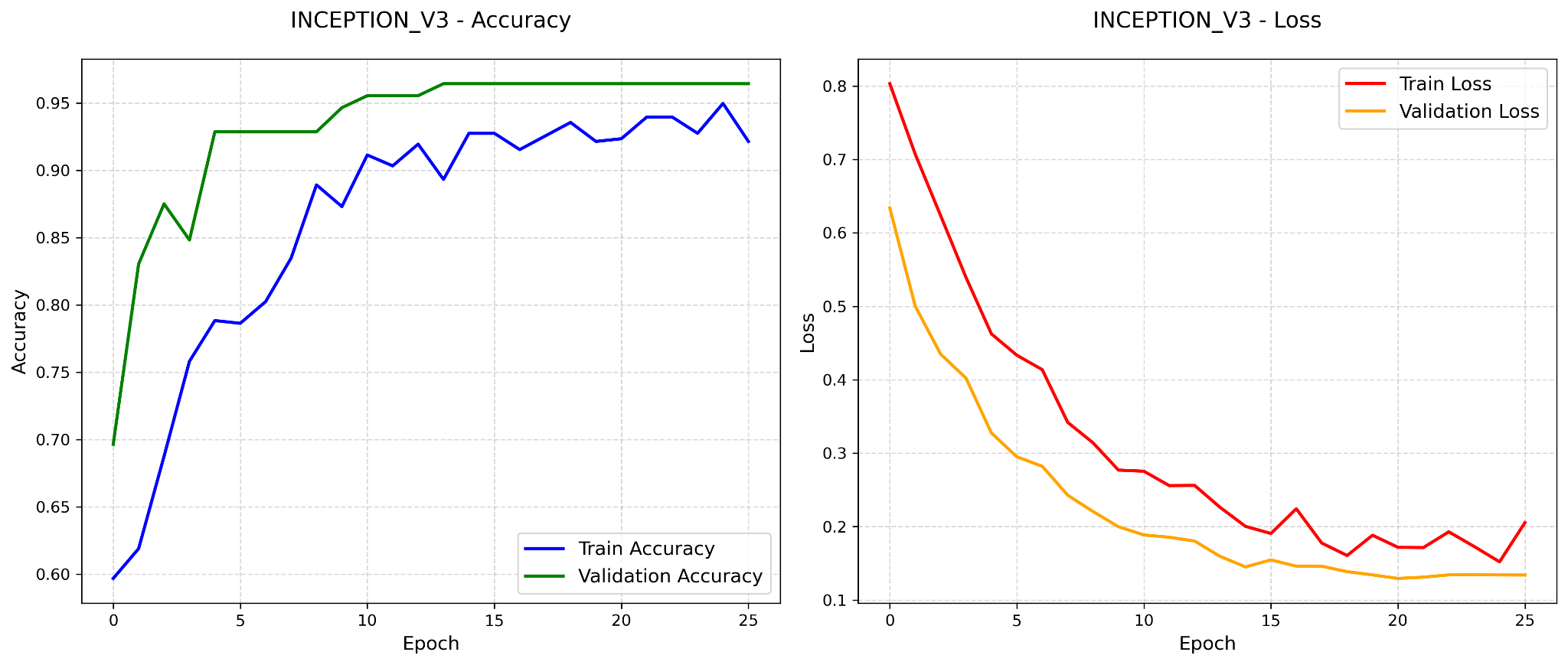


Figure 8 - Inception\_v3 Accuracy and Loss curves for Training and Validation data.

### **Inceptionv3 (Weakest Performance)**

**Train Acc:** 92.14% | **Test Acc:** 97.79%

**Why it struggled:** The **multi-branch architecture** may be **too complex** for this dataset, leading to **underfitting** (lowest train accuracy) and worse generalisation.

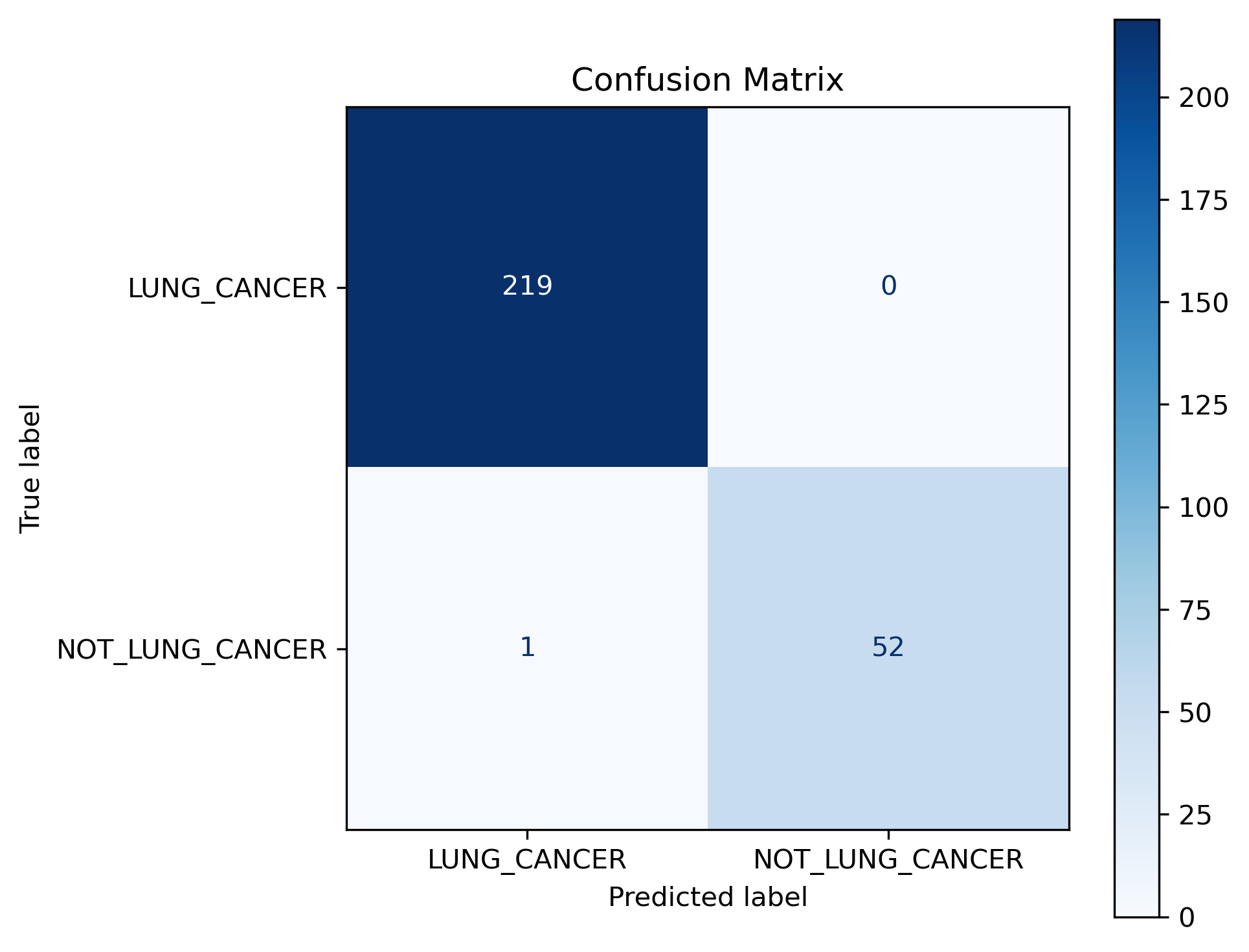


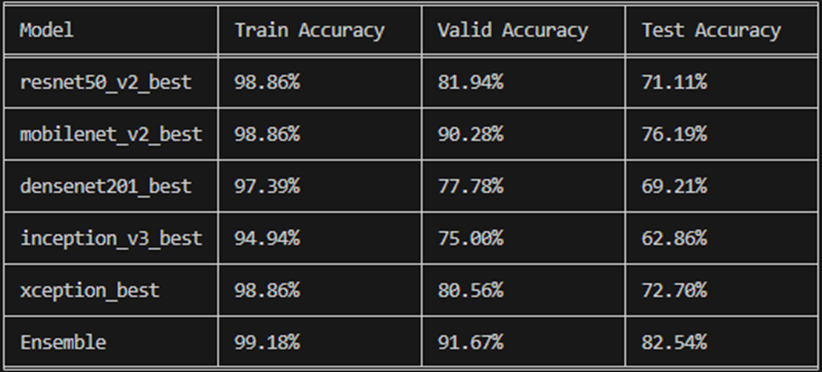
Figure 9 - Confusion Matrix for ensemble of 5 CNN models.

* The model is **near-perfect** for detecting cancer (0 false negatives) and makes **only 1 error** in non-cancer cases.
* **False Positive (1 case):** A non-cancerous scan was misclassified as cancerous. This might be due to benign abnormalities (e.g., granulomas) like tumours.

**Stretch Goal: Multi-Class Classification**

To make our model more useful, we pursued a stretch goal beyond the previous binary classification and trained our model to classify between four categories: adenocarcinoma, large cell carcinoma, squamous cell carcinoma, and normal tissue. These adjustments allow the model to support a more specific diagnosis and treatment plan, providing greater value in clinical settings.

Table 2 - Training, validation, and test accuracy for multi-class model



The ensemble model achieved a test accuracy of 82.54%, outperforming the individual CNN models, which ranged between 60-76% accuracy. MobileNet\_V2 achieved the highest accuracy (76.19%), while Inception\_v3 likely lagged due to its complex architecture and lower training generalization. The ensemble model effectively reduced variance and improved generalizability by using the strengths of each base model.

Further development is needed, particularly to improve performance on more ambiguous classes, such as squamous cell carcinoma, which exhibited a high number of misclassifications. With more refinement and more balance of a training set, this multi-class approach could serve as the foundation for a clinical diagnostic tool.

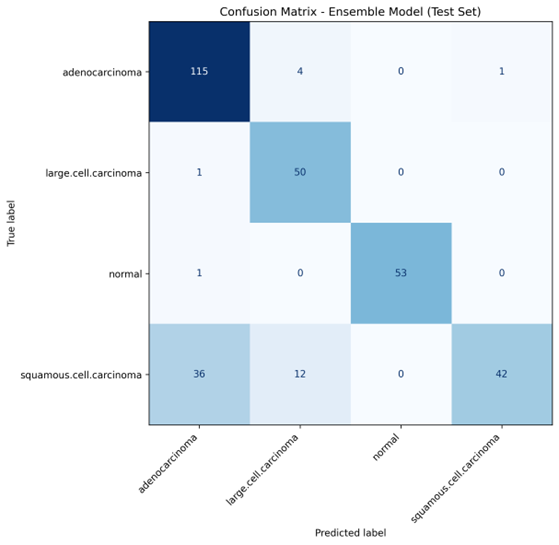
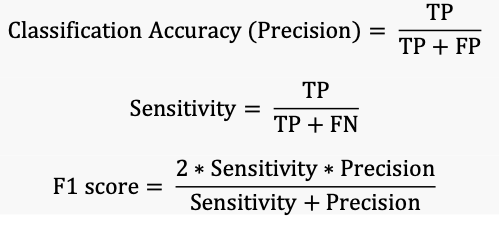


Figure 10 - Confusion Matrix for Multi-Class Ensemble of 5 CNN models

**Results**

To evaluate the performance of our approach, the metrics we adopted are classification accuracy, sensitivity, and F1-score, measured as follows:



where TP stands for True Positive, FP for False Positive, FN for False Negative and TN for True Negative.

In our confusion matrix, the Lung Cancer cases that are positive and are correctly classified by the model are termed as True Positive, and those incorrectly classified as Not Lung Cancer by the model are termed as False Negative. Similarly, for negative lung cancer cases, if classified correctly by the model are termed as True Negative, and incorrectly classified cases as Lung Cancer are termed as False Positive.

▪ We achieved a classification accuracy of 99% on our dataset with 100% sensitivity (recall) for positive Lung Cancer cases.

▪ We obtained 0 FP, i.e. no cancer patient was misinformed that they don’t have cancer.  
▪ We obtained 3 FN cases, i.e., 3 Normal CT scans were mispredicted as Lung Cancer cases.

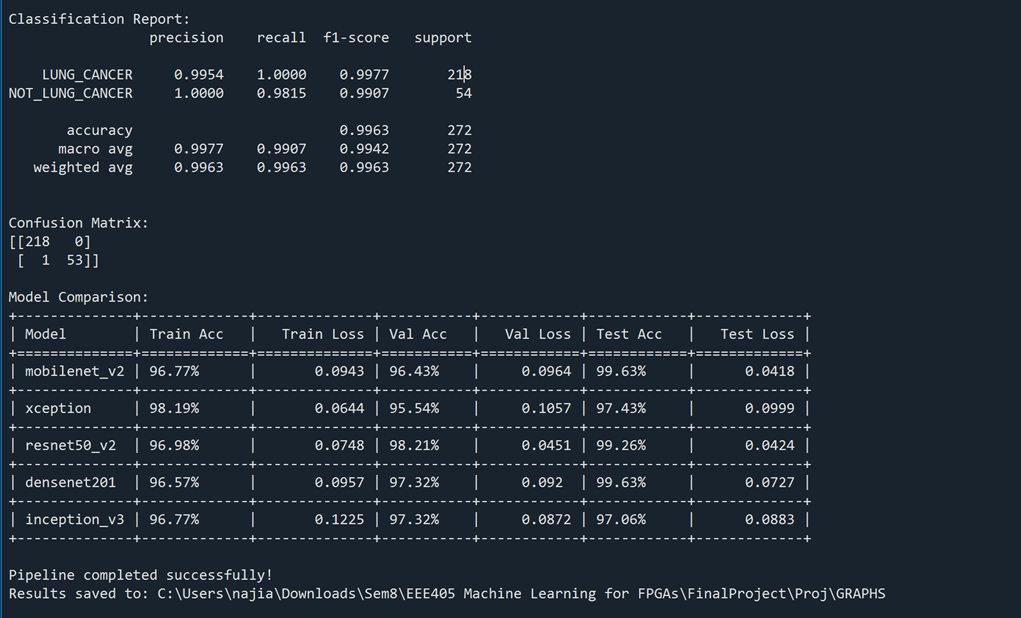


Figure 11 - Classification Report for ensemble of 5 CNN models.

**Conclusion**

We explored the training behavior of five CNN models. Our results demonstrate the effectiveness of ensembling multiple CNN architectures to improve diagnostic accuracy for lung cancer detection. The ensemble model achieved a 99% classification accuracy with 100% sensitivity for cancer cases, ensuring that no cancer-positive patients were misclassified. The confusion matrix confirmed zero false negatives for lung cancer. Although there was one false positive, this is a safer trade-off than missing cancer detection. This makes the model a reliable and efficient tool for early screening. As a stretch goal, we expanded the model to classify between four cases and reached an accuracy of 82.54%. This ensemble approach offers a clinically valuable solution for early lung cancer detection.

**Future Scope**

1. We further want to provide our model with the information on the type of cancer and severity of the case. The newly obtained model can be trained to perform type and severity prediction of the case, so that proper treatment can be provided.
2. We further plan to increase our dataset from various open sources to make our model more generalised & robust
3. Also, we can strengthen our dataset by using appropriate image augmentation techniques to further improve the model and make it more generalised & robust

**References**

1. https://www.kaggle.com/datasets/mohamedhanyyy/chest-ctscan-images.  
2. https://keras.io/api/applications/  
3. https://towardsdatascience.com/ensembling-convnets-using-keras-237d429157eb