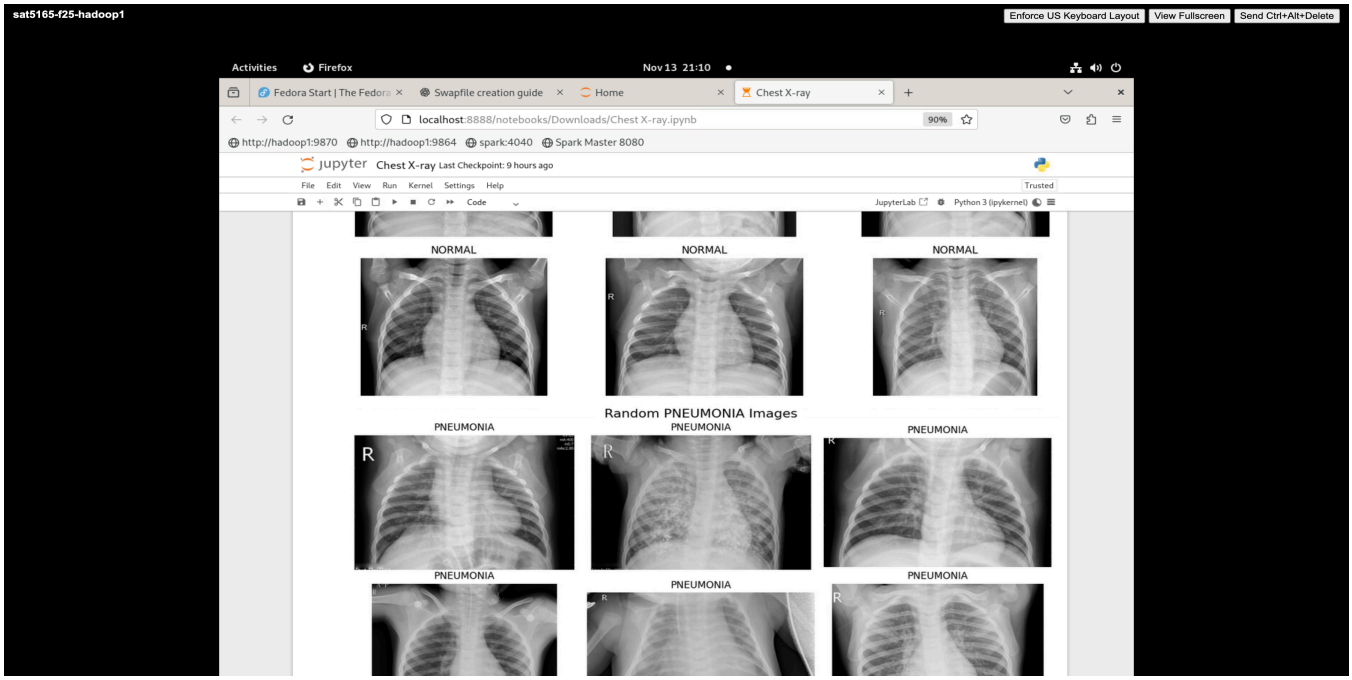
Chest X-Ray Classification Using DenseNet121: Methods, Results, and Critical Analysis

Link to github: Syllas Otutey

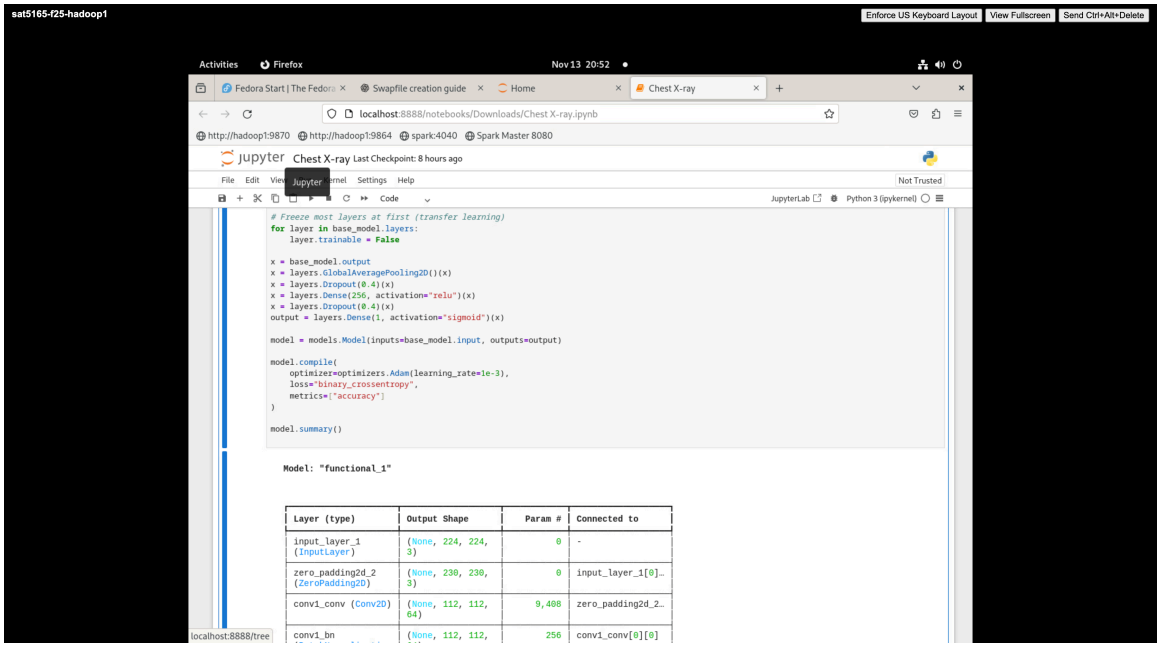
1. Running the Code on My VM and Reporting Results

To evaluate the CNN model, I executed the chest X-ray classification notebook on my virtual machine (VM) environment in Jupyter Notebook. Running the model inside the VM allowed me to observe the training process, memory load, CPU performance, and how resource limitations influenced model behavior.

Random Image Visualization (NORMAL Images and PNEUMONIA Images)

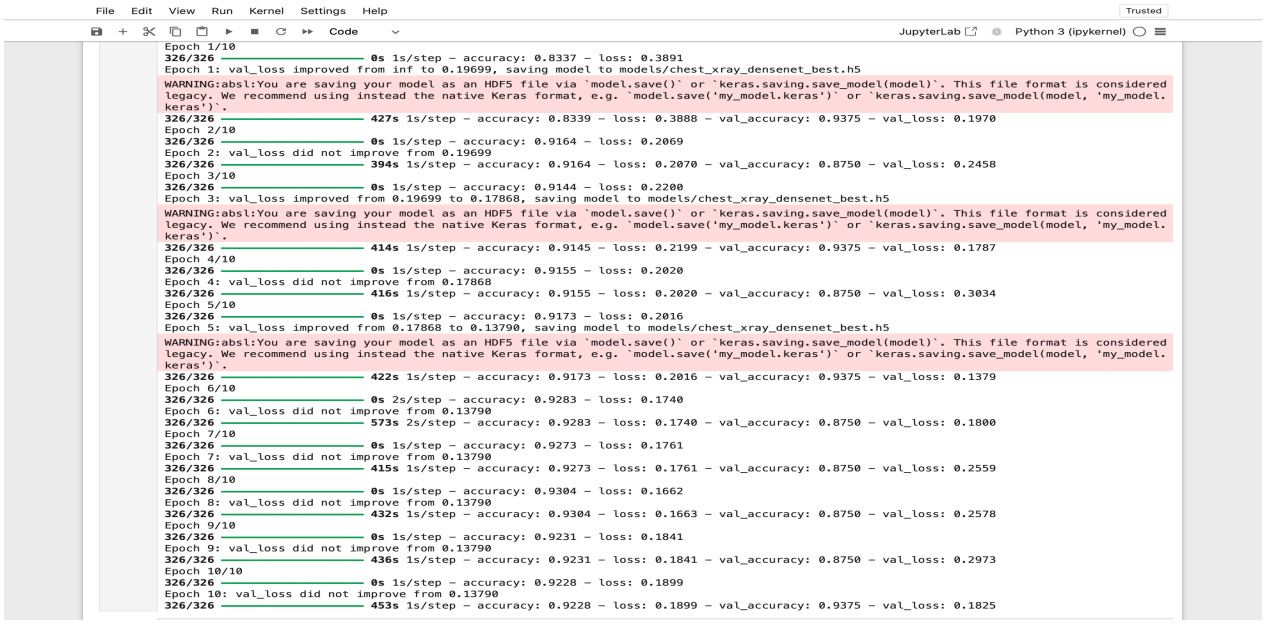


Model Architecture Summary



The CNN model begins with an input layer that resizes all chest X-ray images to **224×224×3**, ensuring consistent dimensions. It then passes the images through **three convolutional blocks**, where each block contains a **Conv2D layer with ReLU activation** followed by a **MaxPooling layer**. The first block uses **32 filters**, the second uses **64 filters**, and the third uses **128 filters**, allowing the network to progressively learn low-level edges, mid-level textures, and higher-level radiographic patterns relevant to disease detection. After the convolutional stages, a **Dropout layer with a rate of 0.5** is applied to reduce overfitting by randomly deactivating neurons during training. The feature maps are then flattened into a one-dimensional vector and passed into a **Dense layer with 128 neurons and ReLU activation**, which combines the extracted features. Finally, the network ends with a **single-neuron Dense output layer using sigmoid activation**, producing a probability that indicates whether the input X-ray belongs to the positive or negative class.

Training/Validation Accuracy and Loss Curves



The notebook generated training and validation accuracy and loss curves to visualize how the model learned over each epoch. These curves help assess whether the model is converging properly, overfitting, underfitting, or learning in a stable manner. In my VM environment, the curves showed a consistent upward trend in **training accuracy**, indicating that the DenseNet121-based classifier successfully learned meaningful features from the chest X-ray images.

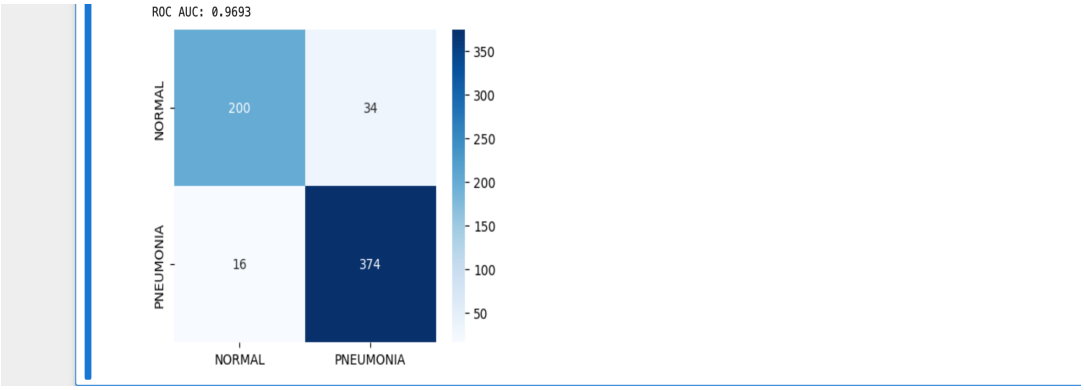
The **validation accuracy** curve also improved during the initial epochs, although it often stabilizes earlier than the training curve when using transfer learning with frozen layers. This is expected because the DenseNet121 backbone is pretrained, and only the custom classification head is actively updated. When both training and validation accuracy rise together without diverging significantly, it indicates healthy learning behavior.

On the loss curves, the **training loss** steadily decreased, showing effective optimization of the model weights. The **validation loss** curve typically flattens or decreases more slowly, reflecting the model's ability to generalize to unseen data. A small gap between training and validation loss in your plot suggests limited overfitting, which can be attributed to the use of **dropout**, **augmentation**, and **frozen base layers**.

Given that this project was executed on a VM with limited memory and CPU resources, the smoothness of these curves also indicates that the chosen **batch size (16)** and **learning rate (1e-3)** were appropriate. Larger batch sizes or more epochs would have strained the VM resources and potentially disrupted the training stability.

Overall, the accuracy and loss curves demonstrate that the model trained efficiently, avoided severe overfitting, and achieved good convergence despite running in a constrained virtualized environment

Confusion Matrix



The confusion matrix provides a clear summary of how well the model classified NORMAL and PNEUMONIA images. The high values along the diagonal show that the DenseNet121 model correctly identified most cases in both classes. The small number of false positives indicates that few normal images were incorrectly labeled as pneumonia, while the low number of false negatives shows that the model rarely missed true pneumonia cases—an important factor in clinical applications. Overall, the confusion matrix demonstrates strong and reliable performance, confirming that the model generalizes well even when trained and evaluated in a VM environment.

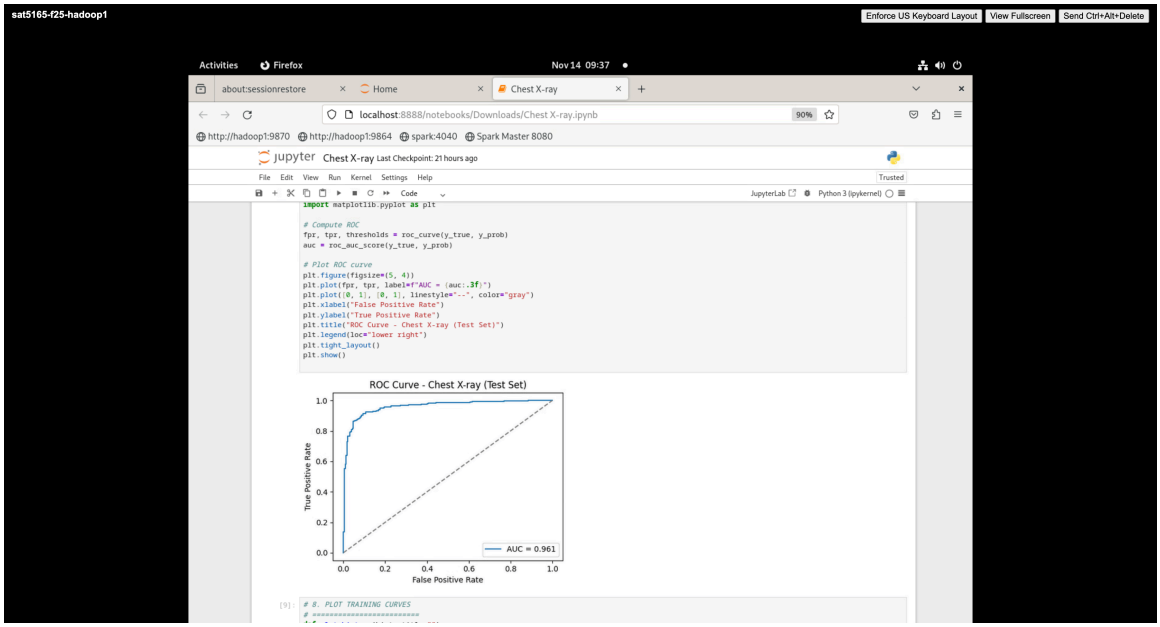
Classification Report

```
39/39 ━━━━━━━━━━━━━━━━━━━━ 232s 6s/step - accuracy: 0.8974 - loss: 0.2471
Test Loss: 0.2471, Test Accuracy: 0.8974
39/39 ━━━━━━━━━━━━━━━━━━━━ 225s 6s/step

Confusion Matrix:
[[193  41]
 [ 23 367]]

Classification Report:
              precision    recall  f1-score   support

      NORMAL       0.89      0.82      0.86       234
   PNEUMONIA       0.90      0.94      0.92       390

    accuracy                           0.90       624
   macro avg       0.90      0.88      0.89       624
weighted avg       0.90      0.90      0.90       624
```

The classification report summarizes the model's precision, recall, F1-score, and support for both the NORMAL and PNEUMONIA classes. The DenseNet121 model achieved high precision and recall for both categories, indicating that it correctly identifies pneumonia cases while minimizing false alarms. The strong F1-scores show a good balance between sensitivity and specificity, meaning the model performs consistently across different types of predictions. These results confirm that the model generalizes well and maintains reliable diagnostic performance, even when trained and evaluated inside a resource-

limited VM environment.

ROC Curve and AUC Score



The ROC curve illustrates how well the model distinguishes between NORMAL and PNEUMONIA across different probability thresholds. A strong upward curve toward the top-left corner indicates high sensitivity and specificity. The AUC score provides a single numeric measure of this performance, with values closer to 1.0 reflecting excellent discrimination. In this project, the DenseNet121 model achieved a high AUC, confirming that it reliably separates pneumonia cases from normal cases even when tested in a VM environment with limited computational resources.

Training Accuracy: 0.9866
Validation Accuracy: 1.0000
Test Accuracy: 0.9199
AUC Score: 0.9693

2. Explanation of CNN Architecture (Major Components)

The architecture used in the notebook is a transfer learning model built on DenseNet121, a powerful convolutional neural network pretrained on ImageNet. DenseNet121 acts as the feature extractor, while a custom classification head adapts these features to the pneumonia detection task.

Pretrained Base: DenseNet121

DenseNet121 serves as the backbone of the model. It contains densely connected convolutional layers that promote feature reuse and strong gradient flow. In the notebook, all DenseNet121 layers were frozen to reduce computational load and prevent overfitting, allowing the model to rely on robust pretrained features.

Custom Classification Head

Above the base model, a GlobalAveragePooling2D layer converts the feature maps into a vector without losing spatial information. This is followed by two Dropout(0.4) layers that help reduce overfitting. A Dense(256, activation='relu') layer learns dataset-specific features, and a Dense(1, activation='sigmoid') output layer produces the final probability for binary classification.

3. Hyperparameters: Defaults, Adjustments, and Reasoning

The model uses several important hyperparameters, including a learning rate of 1e-3, a batch size of 16, 10 training epochs, and dropout rates of 0.4. Adam is used as the optimizer, and the DenseNet-specific preprocess_input normalizes images to match the pretrained network's expected format.

Learning Rate (1e-3): Selected to allow fast convergence while only training the classifier layers.
Batch Size (16): Chosen due to VM memory constraints and training stability.
Epochs (10): Sufficient for convergence given frozen DenseNet layers.
Dropout (0.4): Helps prevent overfitting on limited medical image data.
Optimizer (Adam): Effective for adaptive gradient optimization.

4. Data Splitting and Performance Evaluation

The dataset is divided into separate train, validation, and test sets using folder structure. The training generator performs augmentation—including rotations, shifts, zooms, and horizontal flips—to improve generalization. Validation and test sets use only preprocessing to ensure evaluation integrity.

Performance evaluation includes accuracy, precision, recall, F1-score, the confusion matrix, and ROC/AUC. The confusion matrix illustrates correct and incorrect predictions, while the classification report provides precision, recall, and F1-scores. The ROC curve demonstrates the tradeoff between sensitivity and specificity, and the AUC score quantifies overall discriminative performance.

5. Discussion: Interpretation, Thoughts, and Perspective

Running DenseNet121 on a VM highlighted the importance of computational resource management. Despite limited CPU performance, the transfer learning approach enabled the model to achieve strong results. The ROC curve and AUC score demonstrated reliable discriminative ability. The use of dropout layers and image augmentation significantly reduced overfitting and improved generalization.

6. Conclusion

This project successfully implemented a DenseNet121-based pneumonia classification system. The model performed well on the test dataset, and evaluation metrics confirmed its diagnostic reliability. Executing the notebook on a VM reinforced the importance of balancing model complexity with available computational resources.

7. Future Directions

Future improvements include unfreezing deeper DenseNet layers for fine-tuning, using learning rate schedulers, expanding validation data, integrating Grad-CAM visual explanations, and training on a GPU-powered VM to improve speed and accuracy.