**Detailed Report for Notebook V34 last**

**1. Introduction and Objectives**

The primary goal of this notebook is to develop a deep learning model using DenseNet201 to detect Pneumonia from chest X-ray images provided in the RSNA Pneumonia Detection Challenge. The key challenges addressed include:

* **Class Imbalance:** Pneumonia cases are the minority, requiring techniques like focal loss and class weighting.
* **Overfitting:** Employing progressive fine-tuning and regularization techniques to help the model generalize better.
* **Data Format Conversion:** Converting DICOM images (the original format) to PNG so that they can be used with Keras’ ImageDataGenerator.

**2. Dataset Description and Preprocessing**

**Data Source**

* **RSNA Pneumonia Detection Challenge Dataset:**  
  The dataset includes DICOM images and accompanying CSV files containing bounding box annotations. An image is labeled as Pneumonia if it contains one or more bounding boxes, and Normal otherwise.

**Data Preprocessing**

* **DICOM to PNG Conversion:**  
  The notebook converts DICOM images from the read-only Kaggle input directory (/kaggle/input/rsna-pneumonia-detection-challenge/stage\_2\_train\_images) into PNG format and saves them in a writable directory (e.g., /content/rsna-dataset/train\_png).
* **Label Generation:**  
  A CSV file is used to generate binary labels (0 for Normal, 1 for Pneumonia) by checking for the presence of bounding box values. The file paths are then updated to point to the newly converted PNG images.
* **Data Splitting and Augmentation:**  
  The dataset is split into training and validation sets using stratified sampling to maintain the class distribution. The training set is augmented (rotations, zoom, shear, shifts, etc.) to improve generalization, while the validation set is only rescaled.

**3. Model Architecture**

**Base Model: DenseNet201**

* **Pre-trained on ImageNet:**  
  The DenseNet201 model is loaded without its top classification layers.
* **Freezing Strategy:**  
  Initially, the DenseNet201 base is frozen to train only the custom classification head.

**Custom Classification Head**

* **Layers Added:**
  + A **SpatialDropout2D** layer (with dropout rate 0.5) to reduce overfitting by dropping entire feature maps.
  + **GlobalAveragePooling2D** to reduce spatial dimensions.
  + A **Dense layer (64 units)** with ReLU activation and L2 regularization to learn task-specific features.
  + A **Dropout layer** (rate 0.5) for further regularization.
  + A final **Dense layer (1 unit)** with sigmoid activation for binary classification.

**4. Loss Function and Class Imbalance**

**Custom Focal Loss Function**

* **Purpose:**  
  The focal loss is implemented to address class imbalance by down-weighting easy examples and focusing more on hard-to-classify cases.
* **Implementation:**  
  A custom focal loss function is defined and used during model compilation. This avoids dependency issues with TensorFlow Addons, ensuring compatibility and control over hyperparameters (gamma and alpha).

**Class Weights**

* **Calculated Using sklearn:**  
  Class weights are computed to further balance the loss contribution between the majority (Normal) and minority (Pneumonia) classes.

**5. Training Procedure**

**Training Strategy**

* **Initial Training (Custom Head Only):**  
  The model is first trained for several epochs with the DenseNet201 base frozen. This allows the new head to learn task-specific features without altering the pre-trained weights.
* **Progressive Fine-Tuning:**  
  After the initial phase, the last 50 layers of DenseNet201 are unfrozen and the model is recompiled with a lower learning rate (e.g., 1e-5) for fine-tuning.

**Callbacks**

* **EarlyStopping:** Monitors validation AUC and stops training if no improvement is seen.
* **ReduceLROnPlateau:** Reduces the learning rate when the validation AUC plateaus.
* **ModelCheckpoint:** Saves the best model using the .keras extension (as required by new Keras standards).

**6. Model Evaluation**

**Metrics and Evaluation**

* **Validation AUC:** The model achieves an AUC of approximately 0.86.
* **Confusion Matrix and Classification Report:**
  + When using a default threshold of 0.5, the model initially predicted all images as Normal.
  + After analyzing the probability distribution (Min ~0.066, Max ~0.495, Mean ~0.268), the optimal threshold was tuned to **0.34**.
  + With the threshold set at 0.34, the confusion matrix shows an improved balance, capturing a better trade-off between recall and precision for Pneumonia.
* **Additional Visualizations:**
  + **ROC Curve:** Shows the overall ranking performance of the model.
  + **Learning Curves:** Loss and AUC curves are plotted over epochs, indicating training progress and any signs of overfitting.

**7. Accuracy Curve Over Epochs**

**Adding Accuracy to the Model Metrics**

* **Initial Setup:**  
  The model was originally compiled with AUC only. To visualize accuracy over epochs, the compilation is updated to include the 'accuracy' metric.
* **Retraining:**  
  The model is retrained (or fine-tuned further) so that the history contains accuracy data.

**Plotting Accuracy**

A separate code cell is used to plot training and validation accuracy curves over epochs, which helps in tracking the performance evolution.

**8. Summary and Next Steps**

**Summary**

* **Data Handling:**  
  DICOM images are successfully converted to PNG, and labels are generated accordingly.
* **Model Performance:**  
  The fine-tuned DenseNet201 model achieves around 80% overall accuracy and an AUC of ~0.86.
* **Threshold Tuning:**  
  The optimal threshold was determined to be 0.34 to better capture Pneumonia cases.
* **Learning Dynamics:**  
  Learning curves indicate some overfitting (with training metrics being higher than validation), which can be addressed with further regularization or more aggressive data augmentation.

**Next Steps**

* **Threshold Optimization:**  
  Further experiment with threshold values based on precision-recall trade-offs.
* **Regularization:**  
  Increase dropout or L2 regularization if overfitting becomes more pronounced.
* **Model Interpretability:**  
  Consider implementing Grad-CAM or similar techniques to visualize which areas of the chest X-rays the model focuses on.
* **Deployment:**  
  Once the model meets performance expectations, explore deployment options for real-time inference.

This report encapsulates the end-to-end process and critical findings from Notebook V31. If you need additional details or modifications, feel free to ask!