**Pneumonia Detection**

**Project Type**: Image Classification – Binary (Pneumonia vs Normal)  
**Framework**: TensorFlow / Keras

**Objective**

To develop and evaluate deep learning models capable of detecting pneumonia from chest X-ray images using both custom CNN architectures and state-of-the-art transfer learning techniques, while applying data augmentation and fine-tuning strategies to enhance generalization and performance.

**Dataset**

* **Link**: https://www.kaggle.com/datasets/tolgadincer/labeled-chest-xray-images/data
* **Input**: Chest X-ray images
* **Classes**: Pneumonia, Normal
* **Preprocessing**:
  + Resizing to uniform dimensions (e.g., 128×128)
  + Normalization (rescale pixel values to [0, 1])
  + Data augmentation (for training set only)

**Models**

**Custom CNN Models (from scratch)**

* **Model 1**: 3 convolutional blocks with BatchNorm, ReLU, MaxPooling, Dropout
* **Model 2**: Slightly deeper architecture, additional Conv layers and more aggressive dropout

**Purpose**:

* Serve as baselines
* Demonstrate the effect of data augmentation and dropout on generalization

**Transfer Learning Models**

All models initialized with **ImageNet** weights and trained with custom dense heads for binary classification.

**ResNet50V2**

* Used with frozen base at first
* Fine-tuned top layers to reduce overfitting
* Lighter and faster compared to deeper models

**ResNet152V2**

* Deeper version for capturing complex patterns
* Regularized via dropout and early stopping
* Fine-tuned progressively

**EfficientNetB3**

* Pretrained with fine-tuning applied to all layers
* Balanced accuracy and model size
* Showed strong performance on validation/test sets

**Overfitting prevention techniques**:

* EarlyStopping
* ReduceLROnPlateau
* Dropout (0.2–0.5)
* Data augmentation
* Fine-tuning in stages

**Training Setup**

* **Epochs**: Typically 5–20 (based on early stopping)
* **Batch Size**: Defined by BATCH
* **Validation split**: 20% from training set
* **Callbacks**:
  + EarlyStopping (patience=5)
  + ReduceLROnPlateau (factor=0.2, patience=2)

**Metrics**

* **Loss Function**: Binary Crossentropy
* **Optimizer**: Adam (default learning rate unless tuned)
* **Evaluation**:
  + Accuracy
  + Precision
  + Recall
  + F1-Score
  + Confusion matrix
  + ROC-AUC