Objective:
The goal of this assignment is to create a pipeline that combines advanced OpenCV image preprocessing techniques with a deep learning model to classify images into predefined categories. You will explore both traditional image processing and modern deep learning to achieve a robust solution.

Assignment Description

Dataset Description:

This dataset is a curated collection of 3710 chest X-ray images, accompanied by comprehensive metadata, aimed at advancing research in medical image analysis, particularly for multi-label disease classification tasks. The dataset provides an opportunity to explore the application of machine learning in identifying and classifying multiple co-occurring chest conditions from X-ray imagery.

Dataset [Download Link]: https://www.kaggle.com/datasets/rishabhrp/chest-x-ray-dataset

Tasks:

Step 1: Data Augmentation and Preprocessing (OpenCV)
Perform the following preprocessing steps on the dataset using OpenCV:
Resize all images to 224x224 pixels.
Apply Gaussian noise to simulate real-world noise.
Implement advanced augmentations like rotation, perspective transformation, and histogram equalization.
Split the data into training, validation, and test sets.

Step 2: Feature Extraction Using OpenCV
Extract features from the images using advanced OpenCV techniques:
Use ORB or SIFT to identify keypoints and descriptors.
Save the descriptors for visualization and analysis.

Step 3: Deep Learning Pipeline
Use a pre-trained convolutional neural network (e.g., ResNet50, VGG16) for classification.
Fine-tune the model on the preprocessed images.
Experiment with different optimizers, learning rates, and loss functions.
Use transfer learning and freeze certain layers during training.

Step 4: Integrate OpenCV Features with Deep Learning
Combine OpenCV-derived features (e.g., keypoint descriptors) with deep learning predictions to create a hybrid model.
Evaluate whether the combination improves classification performance.

Step 5: Evaluation
Evaluate the model's performance on the test set using accuracy, precision, recall, and F1-score.
Visualize model performance using confusion matrices and ROC curves.

Compare the results of the deep learning model with and without OpenCV preprocessing.

Deliverables:

A Jupyter notebook (.ipynb) with:
Code implementation.
Visualization of preprocessing steps.
Model architecture summary.
Evaluation metrics and plots.
A report summarizing:
Methodology used.
Comparison of model performance.
Insights and conclusions.

Extra Credit:

Implement Grad-CAM to visualize what the CNN focuses on during classification.
Automate the entire pipeline using a Python script.

Evaluation Criteria:

Criteria	Weightage
Data Preprocessing & Augmentation	20%
OpenCV Feature Extraction	20%
Deep Learning Model Performance	30%
Integration of OpenCV & DL	25%
Report & Documentation	5%