CliniScan: Lung Abnormality Detection Using Chest X-Rays

Done by :-

Y.Venkata Lingeshwara Reddy

Table of Contents

Project Description

The project focuses on developing an AI-powered system capable of automatically detecting and localizing lung abnormalities from chest X-ray images. Using deep learning–based object detection models, the system identifies pathological findings such as cardiomegaly, pneumothorax, fibrosis, infiltration, pleural effusion, and other thoracic abnormalities. The main objective is to assist radiologists by providing fast, accurate, and interpretable predictions that highlight abnormal regions using bounding boxes.

This system processes raw DICOM images, improves them using medical-grade pre-processing techniques, converts annotations to YOLO format, and trains a robust neural network capable of detecting abnormalities with high accuracy.

Dataset Used

The dataset used is the VinBigData Chest X-ray Abnormalities Detection dataset, which consists of DICOM-format chest radiographs along with bounding box annotations provided by radiologists.

Environment Setup

For this project, the environment includes:

Programming Language

- Python 3.8 or above

Essential Libraries

- **pydicom** → for reading DICOM files
- **OpenCV** → image processing and enhancement
- **NumPy** → numerical computation
- **Pandas** → handling annotation CSV
- **Matplotlib** → visualization
- **tqdm** → progress bars
- **Ultralytics YOLOv8** → training and detection model

Hardware Requirements

- 6-core or higher CPU
- Optional GPU for training (NVIDIA recommended)
- At least 50–100GB free disk space for processing and saving images

Data Exploration

Basic data analysis is performed to understand dataset structure, file formats, number of images, annotation complexity, class imbalance, and image characteristics. Initial visualizations help identify preprocessing needs.

The Data exploration I did it for this project is:

1.Dataset Size

Count DICOM files in train & test folders.

Image Dimensions

Most images are high-resolution (≈2836×2336 pixels).

Pixel Format

12-bit grayscale images (0–4095 intensity range).
Many images use **MONOCHROME1**, meaning the colors are inverted (must be corrected).

Annotation Distribution

- Some images have multiple abnormalities
- Some have no findings
- Classes are imbalanced (some abnormalities appear very rarely.

Sample Visualization

Displaying raw DICOM images using pydicom helps identify:

- intensity issues
- noise
- embedded borders
- need for CLAHE & normalization

Data Pre-Processing

Data preprocessing is a crucial step in any medical imaging project, especially in chest X-ray analysis, where raw DICOM images contain high-resolution pixel data, noise, inconsistent contrast, unnecessary borders, and non-standard sizes. Before training a deep learning model, these images must be cleaned, enhanced, and standardized so the network can learn meaningful features effectively. This stage improves image quality, corrects intensity issues, removes irrelevant regions, and transforms the dataset into a uniform, model-ready format. Proper preprocessing directly impacts model accuracy, stability, and overall detection performance.

Modules Used:

Pydicom

`pydicom` is a Python library used to read and work with DICOM files — the standard format in medical imaging.
It allows extraction of pixel data, metadata (like photometric interpretation), and patient information.
It is essential because chest X-rays in this dataset are stored in DICOM format.

NumPy

NumPy provides efficient numerical operations on arrays.
It is used for:

- handling pixel matrices
- normalization
- intensity transformations
- resizing calculations
- cropping operations

Almost every preprocessing step uses NumPy internally.

OpenCV (cv2)

OpenCV is a powerful image-processing library written in C++ and optimized for speed.
It is used for:

- resizing
- grayscale conversion
- denoising

- CLAHE contrast enhancement
- cropping borders
- padding
- RGB conversion
- image saving

It forms the core of the preprocessing pipeline.

tqdm

`tqdm` is used for progress bars.
Since preprocessing thousands of large X-rays takes time, tqdm shows:

- how many images processed
- estimated time
- live progress

It improves visibility and workflow efficiency.

Matplotlib

(used earlier during exploration phase)

Matplotlib is used to display sample chest X-ray images for:

- visual validation
- understanding contrast
- verifying preprocessing output

It helps confirm visual correctness at each stage.

Pandas

Pandas is used for reading and analyzing `train.csv` which contains:

- bounding box coordinates
- abnormality labels
- radiologist IDs

This library helps structure label data before converting to YOLO format.

Ultralytics YOLOv8

(Used later during training phase)

YOLOv8 is a state-of-the-art object detection architecture used to detect abnormalities.
It requires:

- preprocessed images
- YOLO-format labels
- dataset.yaml

Description about Functionalities used in the pre-processing:

1. Reading DICOM Files

DICOM is the standard format used in medical imaging.
It contains:

- pixel data
- patient information
- acquisition details (window width, window center, photometric interpretation)

Using **pydicom**, the pixel array is extracted from each file.

2. Grayscale Conversion

Chest X-rays are grayscale images.
If any image has additional channels, it is converted to 1-channel grayscale to maintain uniform processing.

3. Resizing

The original X-rays are very large (≈3000×3000).
We first resize them to a manageable resolution (e.g., **1024×1024**) to:

- reduce processing time
- standardize image dimensions
- prepare for uniform preprocessing

4. Intensity Normalization

Since DICOM images are 12-bit (0–4095), normalization is needed to bring pixel values into the standard 8-bit range (0–255).

Two methods exist:

Min-Max Normalization (used in this project)

Z-score Normalization (optional)

This step ensures consistent brightness and stable model training.

5. Contrast Enhancement (CLAHE)

CLAHE (Contrast Limited Adaptive Histogram Equalization) improves visibility of lung fields by locally enhancing contrast.

Benefits:

- highlights soft-tissue structures
- improves abnormality boundaries
- useful for detecting opacity, fibrosis, effusions

CLAHE is widely used in radiology image enhancement.

6. Denoising

Chest X-rays often contain noise due to acquisition conditions.

We use:

- **Median Blur** (removes salt-and-pepper noise)
- **Gaussian Blur** (smooth noise + preserves edges)

This makes images cleaner and easier for the model to analyze.

7. Border Cropping

Some X-rays include black borders or unnecessary blank regions.
We detect non-zero regions and crop around actual lung area.

Benefits:

- reduces irrelevant pixels

- focuses model on lung region
- improves training accuracy

8. Padding for Uniform Shape

After cropping, images may not be square.
Deep learning models require consistent dimensions.

We pad the image to **640×640**, adding black padding equally on all sides to maintain aspect ratio.

9. Convert Grayscale to RGB

YOLO models expect **3-channel (RGB)** images.
We convert the grayscale image into 3-channel format by duplicating channels.

10. Saving as PNG

Processed images are saved as PNG format because:

- PNG retains high quality
- No compression artifacts
- Suitable for training medical models

Model Training and Augmentation

1. Model Training Overview (YOLOv8)

Model training is the stage where a deep learning model learns patterns from the annotated dataset.
In this project, the YOLOv8 object detection model is trained to identify lung abnormalities from preprocessed chest X-ray images.

YOLOv8 learns to:

- Predict bounding boxes around abnormal regions
- Classify abnormalities such as consolidation, effusion, pneumothorax, fibrosis, etc.
- Assign confidence scores to each prediction

The model improves over time through **forward propagation** (prediction) and **backpropagation** (error correction), gradually minimizing its loss values and improving detection accuracy.

2. Understanding Object Detection

Object detection involves two key tasks:

Classification

Determining *what* abnormality is present.

Localization

Determining *where* the abnormality is present in the image.

YOLO (You Only Look Once) performs both tasks in a single pass using a single neural network, making it extremely fast and efficient.
This is especially useful in medical imaging where quick and accurate interpretation is crucial.

3. Why YOLOv8 for Medical Imaging

YOLOv8 is suitable for medical imaging because:

- It provides **real-time performance** even on lower hardware.
- It offers **excellent accuracy**, especially for small abnormalities.

- It supports **custom datasets** and **custom image sizes**.
- It handles **high-resolution chest X-rays** effectively.
- It includes advanced features like anchors, auto-shape predictions, and augmentations.

In medical imaging, precision and reliability are essential — YOLOv8 offers a good balance of both.

4. Safe Medical Data Augmentations

Data augmentation artificially expands the dataset by modifying images in realistic ways. In medical imaging, augmentations must be *safe* so they do not distort anatomy or create misleading training samples.

The following augmentations are used:

Horizontal Flip (fliplr)

Simulates mirrored images while keeping anatomy realistic.

Brightness & Contrast Adjustments (hsv_v, hsv_s)

Simulates different exposure levels and helps the model deal with varying X-ray intensities.

Small Rotations (degrees)

Simulates slight variations in patient positioning.

Scaling / Zoom (scale)

Helps the model generalize to abnormalities of different sizes.

These augmentations help reduce overfitting and increase the robustness of the model.

5. Training Configuration Explanation

Your YOLOv8 training uses the following configuration:

- **imgsz = 640** → Final input image size
- **epochs = 20** → Number of training cycles
- **batch = 4** → Optimal for CPU training
- **degrees = 5** → Minor rotation
- **scale = 0.15** → Zoom-in/zoom-out
- **fliplr = 0.5** → Horizontal flip (50% chance)
- **hsv_h, hsv_s, hsv_v** → Hue, saturation, and brightness variations

- **patience = 10** → Early stopping
- **device = cpu** → Because no GPU available locally
- **workers = 2** → Number of data loader workers

These settings ensure stable training even without a GPU, while still applying medically-safe augmentations.

6. Model Evaluation Metrics

After training, the model is evaluated using:

mAP (Mean Average Precision)

Measures detection accuracy.
Higher mAP means the model is better at localizing and classifying abnormalities.

Precision

Out of all predicted abnormalities, how many were correct?
High precision = fewer false alarms.

Recall

Out of all actual abnormalities, how many were detected?
High recall = fewer misses.

Loss Values

During training, YOLO computes:

- **Box loss** (bounding box accuracy)
- **Class loss** (classification accuracy)
- **Object loss** (object confidence)