Carried out by:

YASSER El KARKOURI

ID : **S450915**

Module: Artificial Intelligence

Option-2 : VinBigData Chest X-ray Abnormalities Detection

Module supervisor teacher:
Dr Salvatore Filippone

18 Mars 2024

Content table

Figures Table

I. Introduction:

The advancement of artificial intelligence (AI) is revolutionizing our world in all fields, including medicine. By introducing AI into healthcare, we're seeing big changes in how doctors diagnose and treat diseases. This shift is especially visible in projects like the VinBigData Chest X-ray Abnormalities Detection. Here, AI helps to quickly and accurately spot problems in chest X-rays, something that was traditionally done by radiologists.

This use of AI means diseases can be caught earlier and treated more effectively. In the VinBigData project, the goal is to identify 14 different types of chest issues without a doctor having to look at each X-ray. This is a huge step forward. It not only makes the diagnosis process faster but also more reliable. It's important because it can lead to better health outcomes for patients around the world. Moreover, using AI in this way can help make high-quality healthcare more accessible. AI can provide consistent, accurate assessments of chest X-rays. This levels the playing field, ensuring that everyone, regardless of their location, has access to the best healthcare possible.

Addressing these challenges, the objective of this project is to localize and to classify a range of thoracic abnormalities from chest X-rays with a good accuracy and efficiency. This project is not just about advancing the state of the art in medical imaging analysis but it's also about providing a robust tool that can alleviate the workload on radiologists, ensuring that patients in rural areas have access to quality diagnostics, and ultimately, paving the way for improved healthcare delivery worldwide. The potential impact of successfully achieving these objectives is immense, offering a beacon of hope for enhancing diagnostic processes and patient care through the integration of AI technologies in the medical field. **[1]**

II. Literature Review:

1. Introduction to CADe/CADx Systems :

Computer-Aided Detection (CADe) and Computer-Aided Diagnosis (CADx) systems represent a transformative leap in in the world of medical imaging, serving as pivotal tools that augment the capabilities of radiologists and other healthcare professionals. These innovative systems support the early detection and detailed analysis of abnormalities found in various imaging studies, including X-rays, MRIs, and CT scans. The main aim of CADe (Computer-Aided Detection) systems is to highlight areas of concern or irregularities within medical images, aiding doctors in identifying issues that could be easily overlooked. On the other hand, CADx (Computer-Aided Diagnosis) systems take this a step further. They not only detect these irregularities but also assess them, offering insights or probabilities about what these irregularities might signify.

In the context of chest X-ray analysis, the progression of CADe/CADx systems has been marked by the integration of increasingly sophisticated machine learning models, especially with the advent of deep learning. These models have shown remarkable ability to learn from large datasets of annotated images, leading to significant improvements in both the accuracy and reliability of anomaly detection and diagnosis. The shift towards deep learning has also facilitated the development of systems capable of not just detecting, but also localizing and classifying a wide range of thoracic abnormalities, addressing one of the long-standing challenges in the field. **[2]**

2. Challenges in Chest Radiograph Interpretation:

The interpretation of chest radiographs poses significant challenges due to the complex thoracic anatomy, subtlety of radiographic findings, high diagnostic workload, interobserver variability, and evolving diagnostic criteria. The dense overlap of critical structures within the chest, such as the heart, lungs, and bones, complicates the identification of pathologies, while the subtlety of certain conditions, like early-stage lung cancer, demands high precision and expertise from radiologists. Additionally, the sheer volume of images to be reviewed can lead to fatigue and potential oversight, and variability in diagnostic interpretations among clinicians underscores the need for more consistent and accurate diagnostic aids. These factors collectively highlight the critical need for advanced diagnostic tools, such as Computer-Aided Detection (CADe) and Computer-Aided Diagnosis (CADx) systems, which promise to enhance diagnostic accuracy, alleviate radiologist workload, and standardize interpretations across the board, thereby improving patient care through timely and precise diagnoses. **[3]**

3. Review of the Project Dataset:

The dataset we'll be working with during this project is the VinDr-CXR dataset crafted by the Vingroup Big Data Institute this dataset significantly advances the capabilities for automatic detection and diagnosis of thoracic abnormalities. It comprises over 18,000 meticulously annotated CXR scans in DICOM format, sourced from two major hospitals in Vietnam, and represents a monumental effort involving a panel of experienced radiologists for the presence of 14 critical radiographic findings, including aortic enlargement, atelectasis, calcification, cardiomegaly, consolidation, interstitial lung disease (ILD), infiltration, lung opacity, nodule/mass, other lesions, pleural effusion, pleural thickening, pneumothorax, pulmonary fibrosis, and an observation for "No finding" to capture the absence of all listed abnormalities ***(see figure 1).*** **[4]**

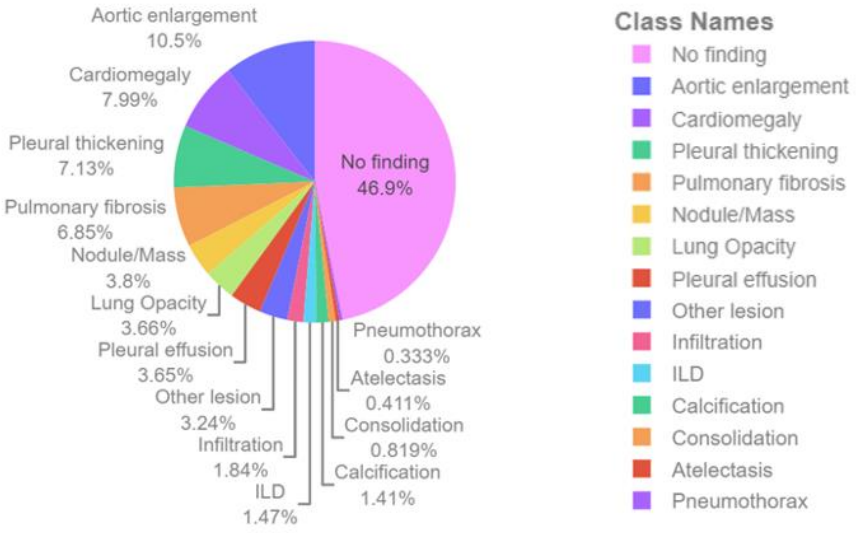


Figure 1 : Distribution of Thoracic Abnormalities in the VinDr-CXR Dataset

From the provided research paper, we understand that there are 14 distinct possible abnormalities and the "No Finding" class within the VinDr-CXR dataset. To visually comprehend the frequency of each annotated class, a bar chart is employed. This graphical representation will serve to illustrate the number of times each abnormality has been identified by the radiologists, thereby providing a clear indication of the annotation distribution across different thoracic conditions in the dataset ***(see figure 2).*** **[4]**

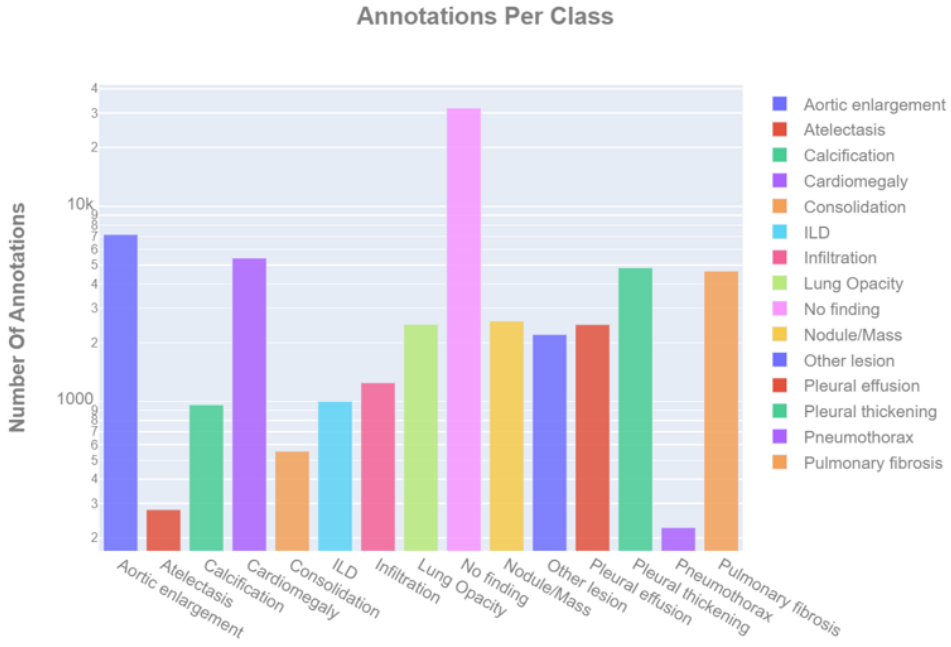


Figure 2:Frequency of Radiologic Annotations for Thoracic Abnormalities in the VinDr-CXR Dataset

We acknowledge the presence of 17 possible radiologists who have contributed to the annotation of the VinDr-CXR dataset. To discern the distribution and volume of annotations made by each radiologist, we will utilize a histogram. This visual tool will help us understand individual contributions and compare the annotation workload among the radiologists, offering insights into the collaborative effort involved in compiling this extensive dataset ***(see figure 3).***

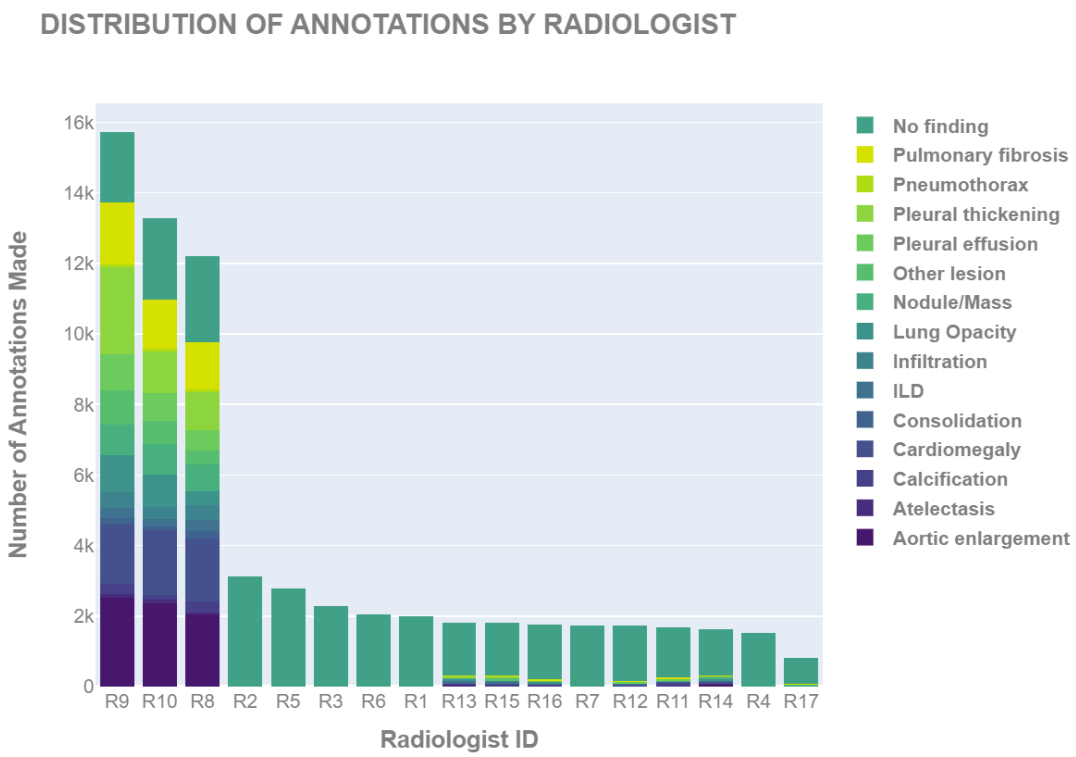


Figure 3:Comparative Distribution of Annotations by Each Radiologist in the VinDr-CXR Dataset

III. Methodology:

The VinDr-CXR dataset comprises 18,000 postero-anterior (PA) chest X-ray scans in DICOM format, each annotated by radiologists for the presence of various thoracic pathologies, as well as a "No finding" label to indicate the absence of abnormalities.

1. Software Framework and Tools:

In developing a sophisticated system for developing our model based on the VinDr-CXR dataset, a robust framework of software tools and programming languages was employed. This section outlines the general framework, programming languages, and specific libraries and tools utilized in the project.

1.1. Programming Language and Software:

Our project extensively uses **Python** for its versatility and support for data analysis and machine learning. For code development and debugging, **Visual Studio Code** serves as the chosen editor, providing robust features tailored for Python programming. The interactive development and data exploration are facilitated through **Jupyter Notebooks**, allowing for a dynamic coding environment that integrates code execution, documentation, and visualization seamlessly.

1.2. Machine Learning and Computer Vision Frameworks:

The core of our image processing and model training relies on **Detectron2**, a powerful library by Facebook AI Research designed for object detection and segmentation, built atop the **PyTorch** machine learning library. **PyTorch** provides the computational backbone, enabling flexible and efficient model design and deployment. These tools are instrumental in developing models that accurately localize and classify various thoracic abnormalities in chest X-rays.

1.3. Hardware and Environment:

Our computational environment is powered by **CUDA toolkit 12.1**, optimizing performance for GPU-accelerated applications, crucial for handling deep learning models and large datasets. The project benefits from the computational capabilities of an **NVIDIA GeForce RTX 4060 Laptop GPU**, ensuring fast model training and inference times, essential for processing the VinDr-CXR dataset effectively.

1.4. Libraries for Data Handling and Visualization:

Data handling and visualization are supported by a suite of **Python libraries**. **Matplotlib** and **Panda**s offer robust solutions for visualizing data and managing it in high-performance data structures, respectively. **NumPy** enhances numerical computations, particularly for operations on arrays and matrices common in image data. For direct image manipulation tasks, **OpenCV**

(cv2) is utilized, providing comprehensive functions for image processing required for preparing the dataset for analysis***(see figure 4).***

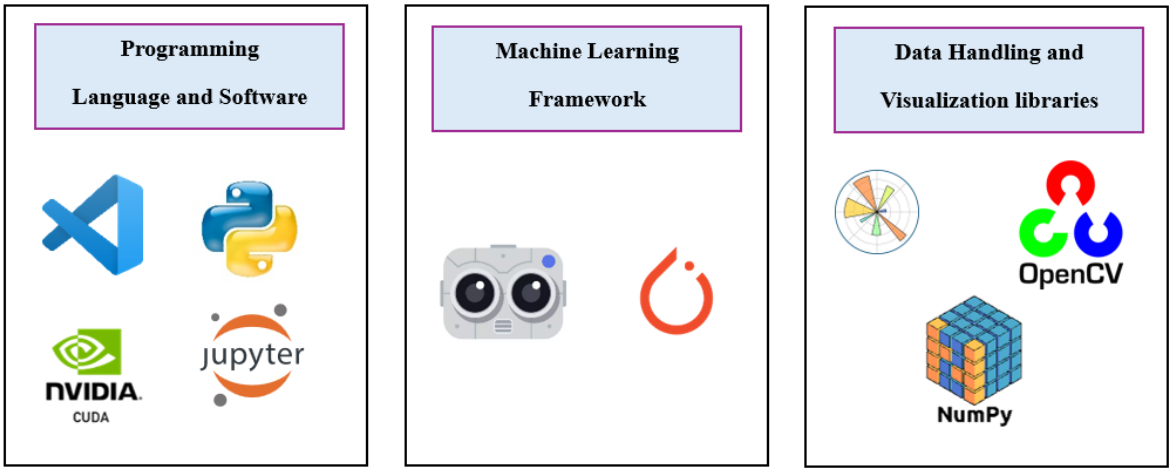


Figure 4:Overview of Computational Tools and Libraries

2. Data Preprocessing :

The preprocessing of this dataset is pivotal to ensuring its compatibility with machine learning models and easing the computational load during training and analysis. This stage encompasses several steps:

- **Conversion to PNG:**

For more straightforward manipulation and compatibility with a broader range of image processing tools, the X-ray images were converted from the original DICOM format to PNG format. This conversion aids in reducing file sizes (from ~200 GB to ~0.5 GB) and simplifies the implementation of image processing techniques.

- **Image Resizing:**

The images were resized to two resolutions: 256x256 and 512x512 pixels. Resizing images to smaller dimensions decreases the computational resources required for processing and model training. This is crucial in computer vision because it allows for a faster training process and reduces the risk of overfitting by limiting the number of learnable parameters in the model. Additionally, using standardized

image sizes is beneficial for batch processing as it ensures consistency in input data dimensions, which is a prerequisite for most deep learning frameworks ***(see figure 5).***

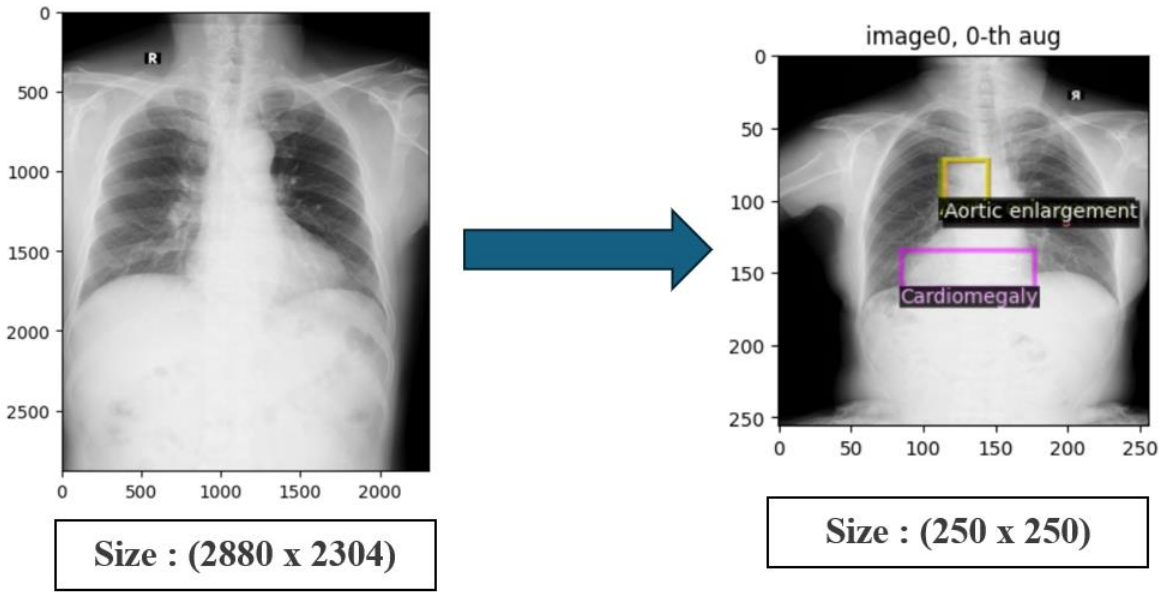


Figure 5:Image resize for pre- processing

- **Monochromatism Correction:**

Some images in the dataset exhibited monochromatism, which could lead to misinterpretation by computer vision models. A preprocessing step was implemented to correct these monochrome images to ensure consistency in visual data representation. This correction ensures that all images have similar chromatic properties, which is important for the model to learn features effectively without being confounded by variations in image intensity and contrast ***(see figure 6).***

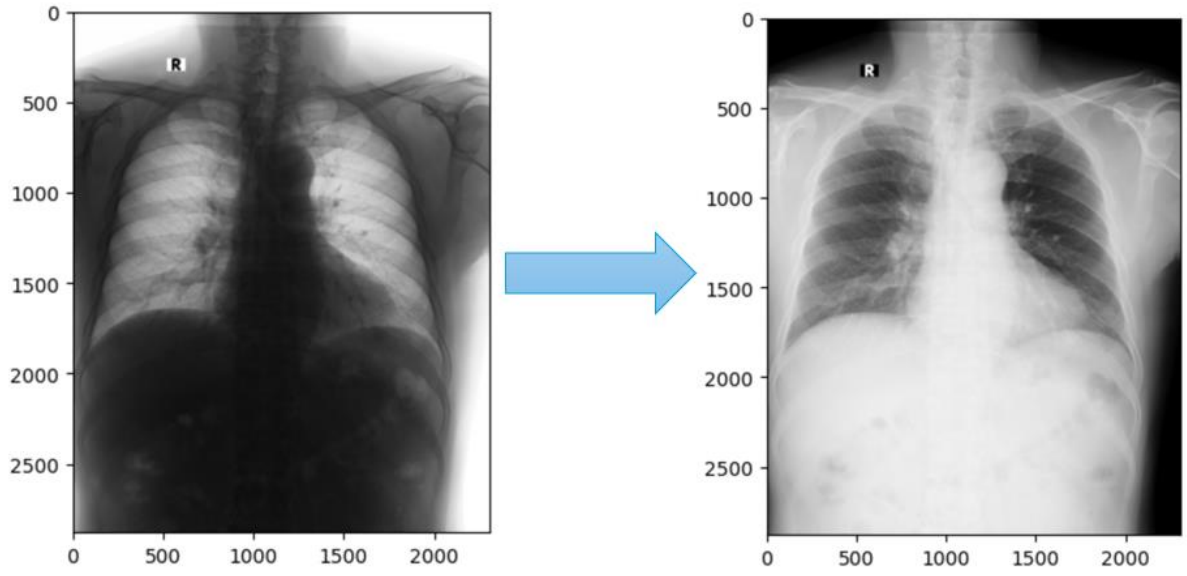


Figure 6:Fix Monochromatism of the Images

3. Training Phase :

Our project leverages the Detectron2 framework for its extensive suite of pre-trained models and its seamless model training and evaluation functionalities. Specifically, we've selected **the R50-FPN architecture**, a decision informed by its proven efficacy in object detection tasks across varied scales. This choice allows us to harness the power of pre-existing, sophisticated neural network architectures and their pre-trained weights, facilitating a more efficient fine-tuning process tailored to our specific dataset and competition requirements ***(see figure 7)*.**

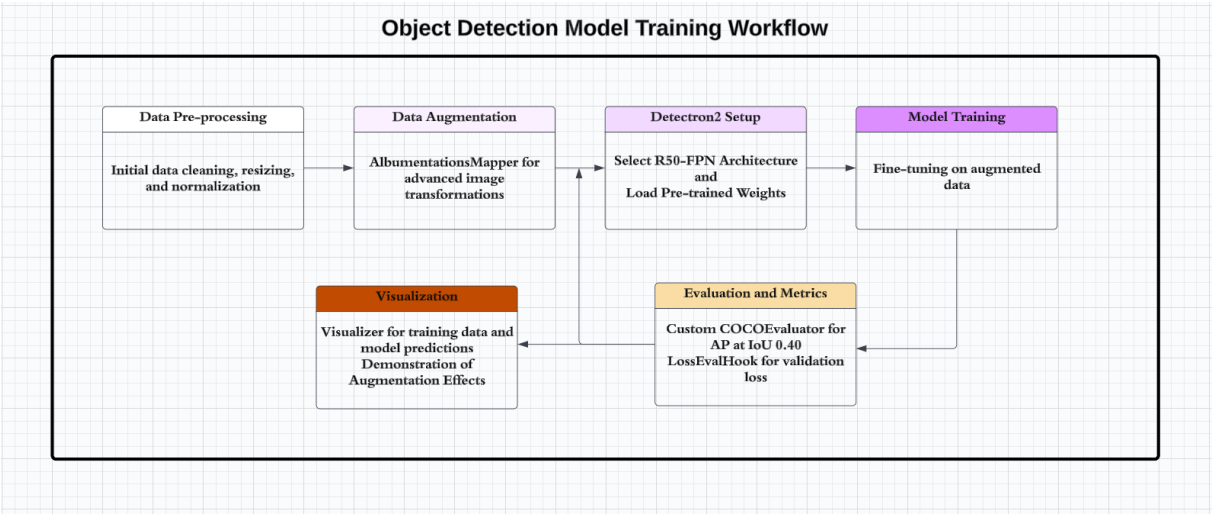


Figure 7:Model Training Pipeline

3.1. Data Augmentation:

A pivotal enhancement in our methodology is the incorporation of the **AlbumentationsMapper** for data augmentation. This custom mapper utilizes the **Albumentations** library to apply a diverse array of augmentations, significantly enriching our training dataset. Such augmentations not only introduce variability into the dataset but also mimic a wide range of real-world conditions, thereby bolstering the model's robustness and generalization capabilities.

The **Albumentations** library stands out for its extensive collection of transform techniques, enabling us to experiment with and implement a variety of augmentations, from basic geometric transformations to more complex pixel-level modifications. This strategic application of data augmentation is expected to markedly improve our model's performance, especially in handling challenging object detection scenarios.

3.2. Evaluation and Validation Loss Calculation:

To quantify our model's performance and ensure it aligns with the competition's metrics, we have customized the **COCOEvaluator** to calculate **Average Precision (AP)** at an **IoU** threshold of **0.40**, diverging from the standard range of **0.50** to **0.95**. This modification allows for a more relevant assessment of our model's accuracy in detecting objects, reflecting the specific requirements of our competition.

Moreover, understanding that the Evaluator does not inherently calculate validation loss, we've implemented a **LossEvalHook**. This hook facilitates the calculation of validation loss by engaging the model in training mode for the validation dataset, thereby enabling a comprehensive evaluation of model performance that includes loss metrics. These metrics are meticulously logged and reviewed to inform further refinements to our model.

3.3. Visualization and Augmentation Demonstration:

An essential component of our methodology is the visualization of training data, which provides intuitive insights into the model's detection capabilities and the effectiveness of applied augmentations. By leveraging Detectron2's Visualizer class, we can depict the training dataset complete with annotated bounding boxes, offering a clear view of how the model perceives and interprets different objects within images.

Furthermore, we will demonstrate the dynamic nature of our **AlbumentationsMapper** through visualizations that showcase the applied augmentations. This demonstration will highlight the mapper's role in diversifying the visual input to our model, ensuring that each image presents a unique learning opportunity, thereby enhancing the model's adaptability and accuracy.

IV. Experimental Results and Evaluation :

1. Pre-Processing :

Before training and testing our model, it's crucial to understand the variability and distribution of the annotated thoracic abnormalities within the chest X-ray images , that was the reason we generated a boxplot which provides a visual representation of the distribution of bounding box areas as a percentage of the source image area for each classified abnormality***(see figure 8)*.**

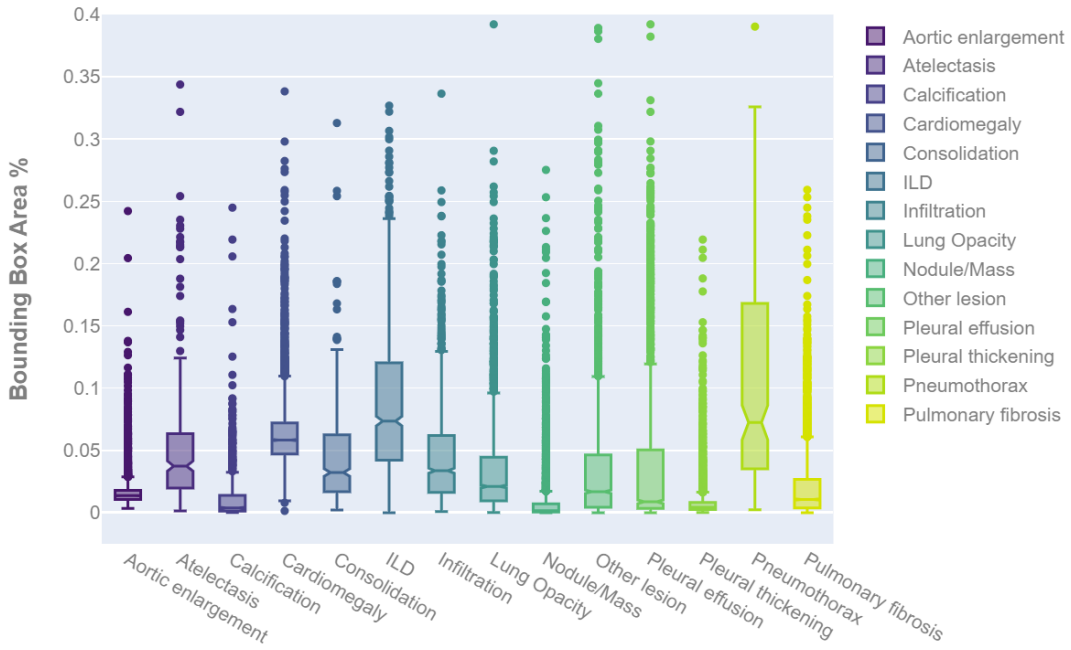


Figure 8:Distribution of Annotated Abnormality Sizes in Chest X-ray Imaging

The boxplot above illustrates the distribution of bounding box areas as a percentage of the source image area across different classes of abnormalities. It's evident that certain conditions, like cardiomegaly and pleural effusion, typically result in larger bounding boxes, reflecting the extensive area they cover in the lung fields. In contrast, conditions such as calcification and nodules/mass tend to have smaller bounding box sizes, indicative of the localized nature of these abnormalities.

Following our analysis of the bounding box size distribution, we move forward to discern any biases in the spatial distribution of annotations across the chest radiographs. This understanding is pivotal as it can reveal tendencies in the positioning of bounding boxes, which may indicate the prevalence of abnormalities in specific regions of the X-ray images. Such insights are crucial for informing potential data augmentation strategies and ensuring that our model does not inherit any positional biases that could affect its generalizability.

To facilitate this exploration, heatmaps have been generated for each class of thoracic abnormality, as well as an aggregate heatmap representing the average across all classes. These heatmaps provide a visual density distribution of where abnormalities are most frequently located within the images. ***(see figure 9)***

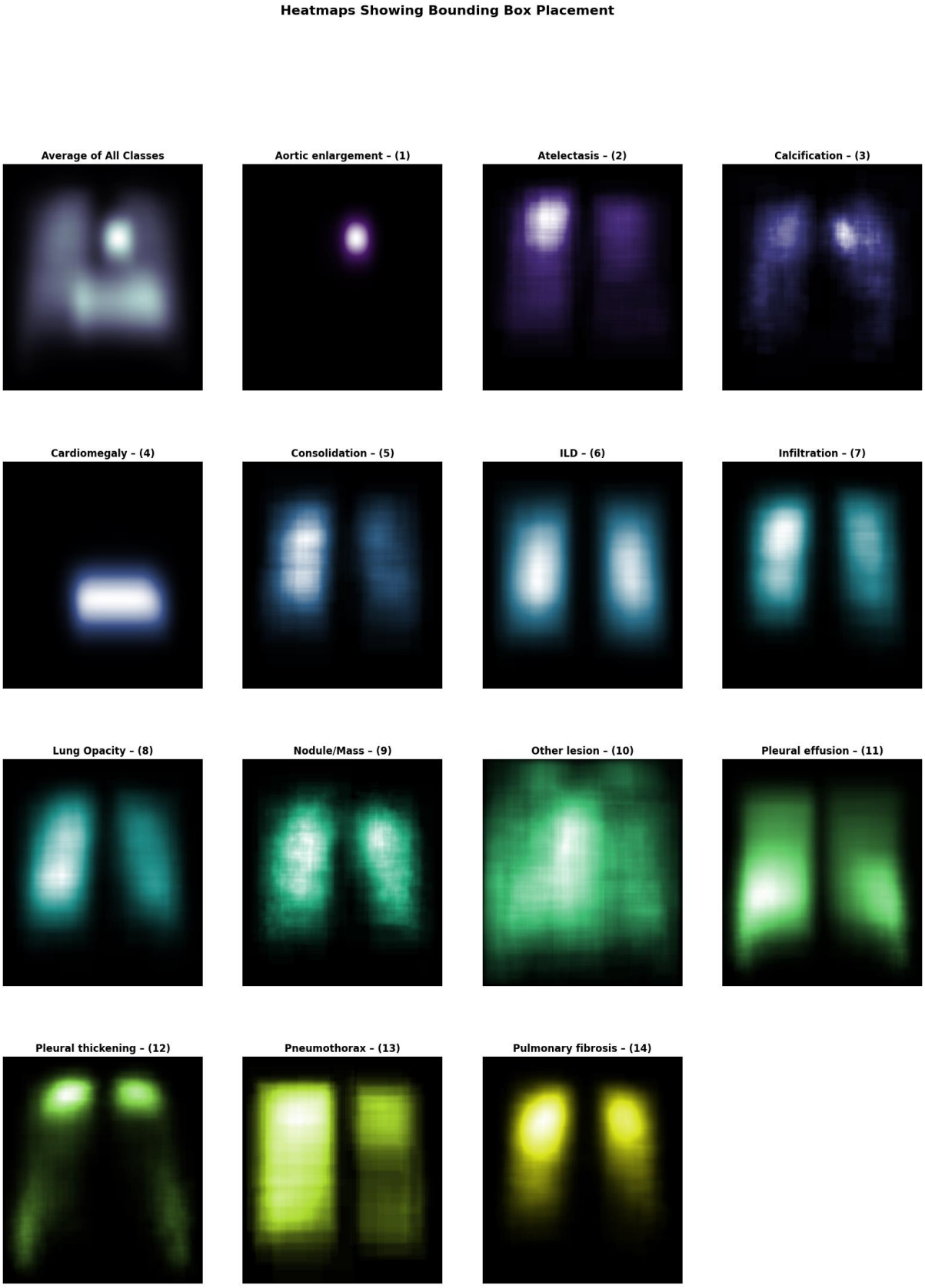


Figure 9:Spatial Distribution Heatmaps of Thoracic Abnormality Annotations in Chest X-Rays

2. Training Results and Evaluation:

The training of our AI model for thoracic abnormality detection was meticulously executed using Detectron2 framework, starting with the careful preparation of the dataset, including image resizing and annotation formatting. We integrated advanced data augmentation through the **Albumentations** library to enhance model robustness. Our training utilized a customized Faster **R-CNN** with **R50-FPN** architecture, with hyperparameters and learning rate schedules configured for optimal convergence. Training was performed using a specialized subclass of **DefaultTrainer**, which was adapted to include our augmentation strategies and a bespoke hook for regular validation loss evaluation. The **COCOEvaluator** provided interim performance metrics, guiding the fine-tuning of the model to achieve a balance between learning efficiency and generalization, as evidenced by the convergence of training and validation losses.

2.1. Loss Curve Visualization:

In order to understanding how well our machine learning model is learning over time it was pivotal to include a loss curve, as illustrated by the graph presented in the figure below ***(see figure 10)*.**

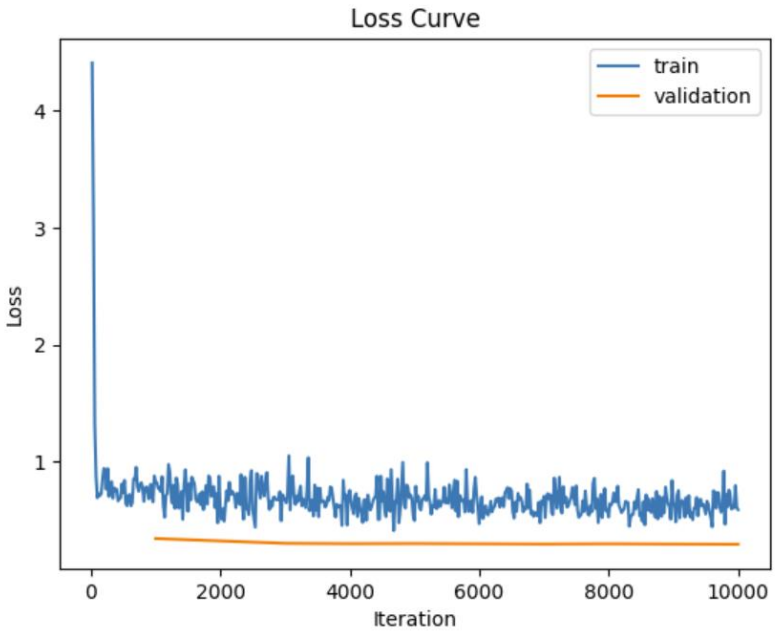


Figure 10:Comparing Training and Validation Loss Over Iterations

The close alignment of the training and validation losses suggests that our model is learning generalizable patterns rather than overfitting to the training data, which is crucial for robust performance on unseen data.

2.2. Performance Metric (AP40) Chart::

Subsequent to training, the model's predictive accuracy is quantified using the Average Precision at an Intersection over Union (IoU) threshold of 0.40, or AP40. This metric is particularly insightful for assessing detection performance ***(see figure 11)***.

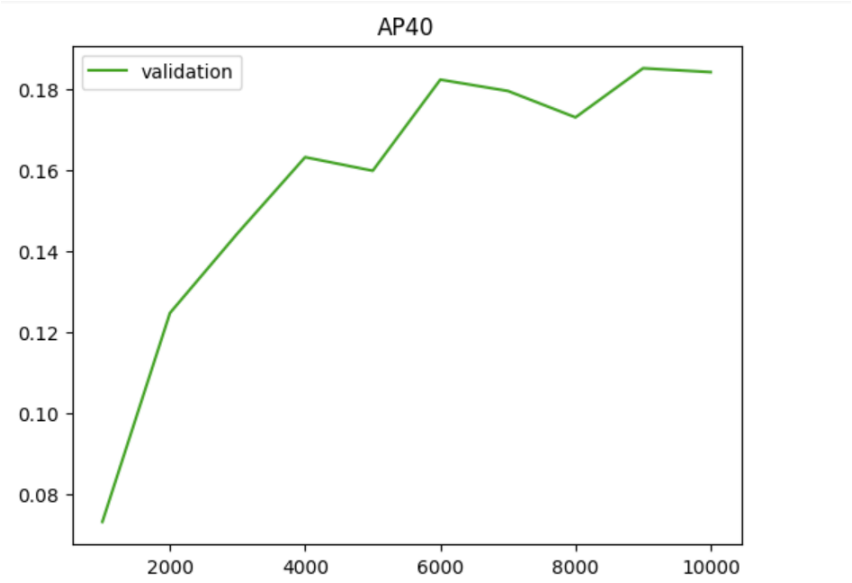
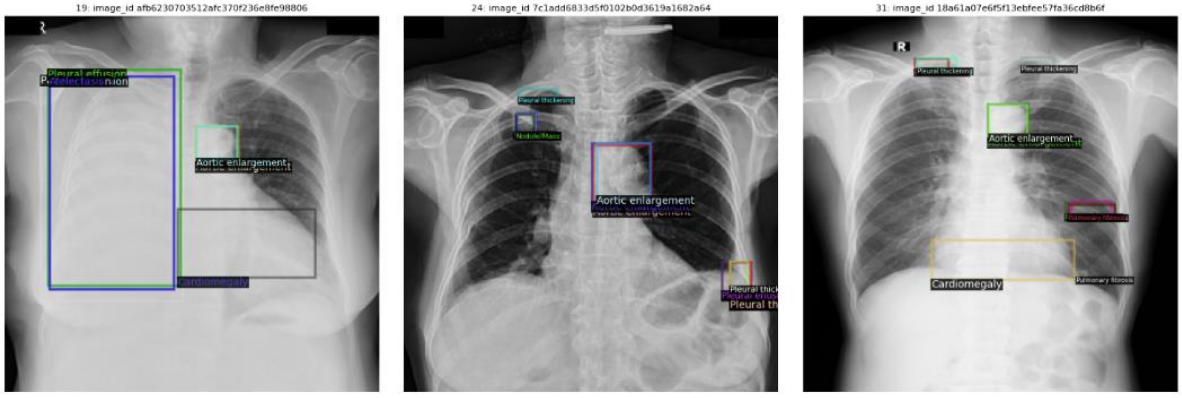


Figure 11:Progression of Model Accuracy

A closer analysis of the loss trends helps us to verify the model's stability and convergence, ensuring that our training regimen is effective and that the model is not overfitting to the training data.

2.3. Detection Example Visualization:

The following sample detections from our model illustrate the practical outcomes of the training. Each image showcases the model's ability to delineate the detected abnormalities within chest X-rays, with bounding boxes color-coded by class***(see figure 12)***.



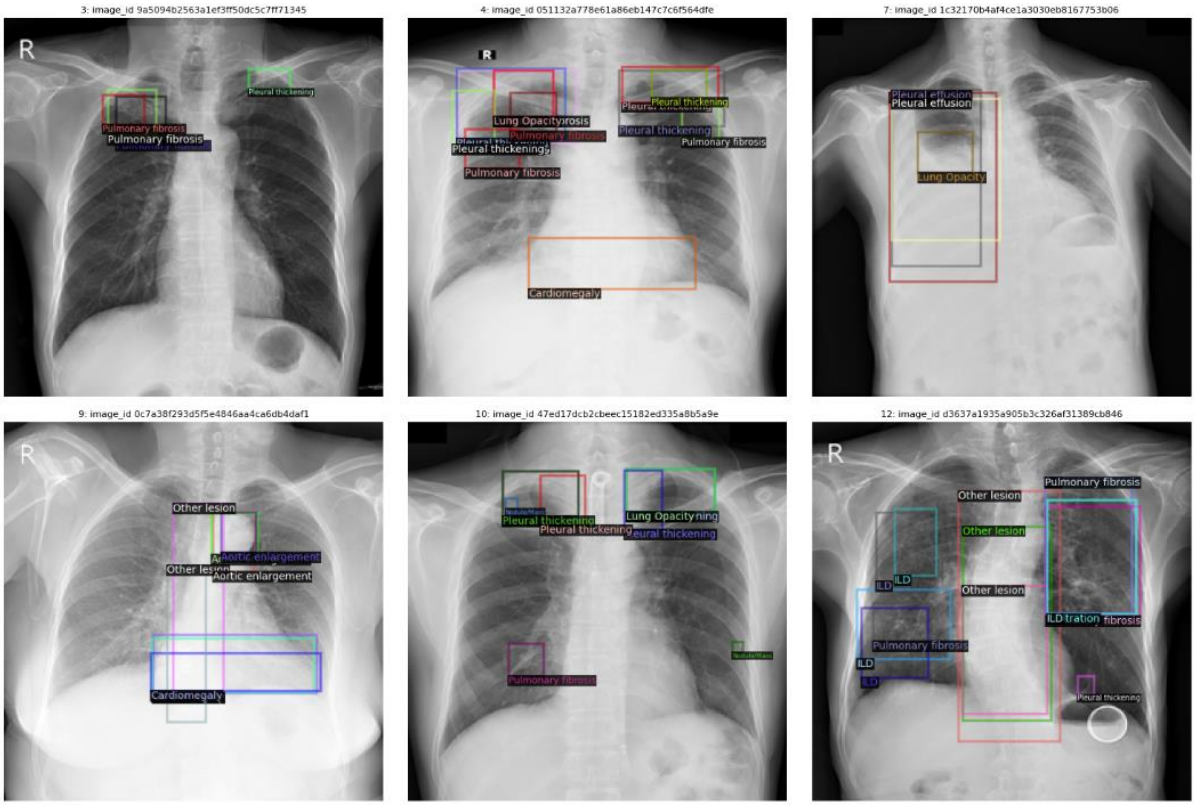
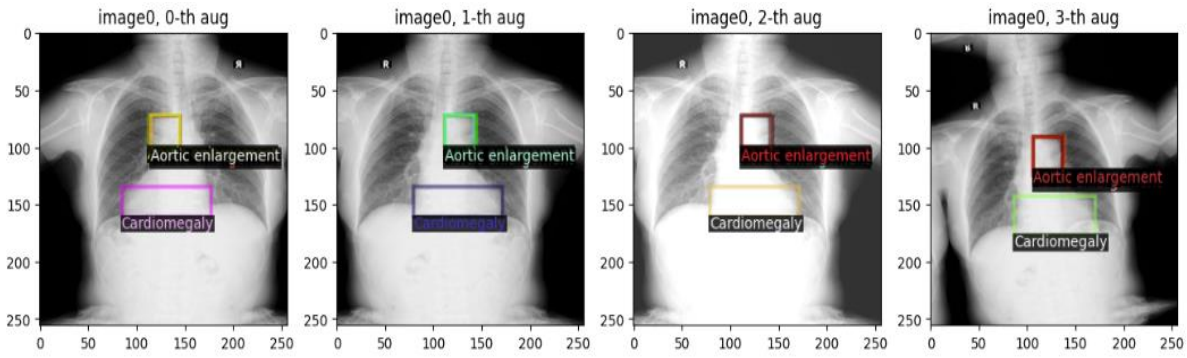


Figure 12: Localizing Thoracic Abnormalities in Chest X-ray Images

These detection examples underscore the model's proficiency in identifying diverse pathological features, thereby validating the quantitative metrics with tangible evidence of the model's operational performance.

2.4. Data Augmentation Visualization:

To ensure our model's resilience to variations in imaging, data augmentation strategies such as random flips and rotations were employed. The images below display the range of augmented scenarios the model was exposed to during training ***(see figure 13)*.**



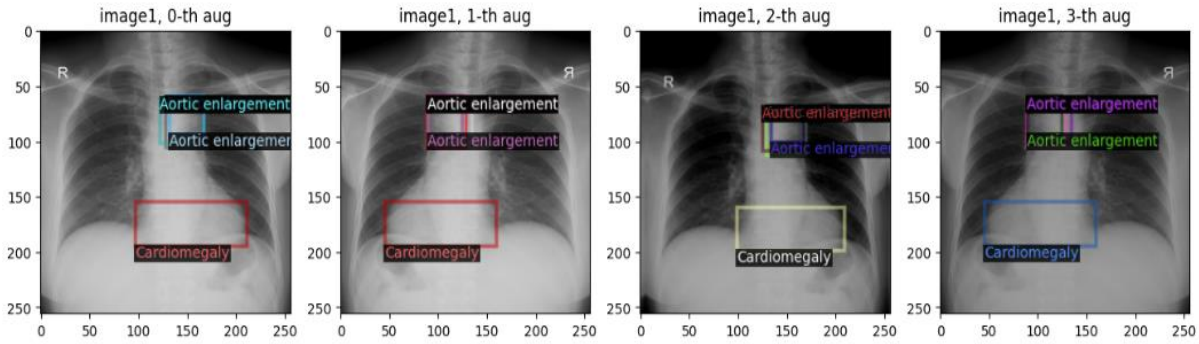


Figure 13:Data Augmentation Visualization

The variance introduced by these augmentations mitigates the risk of overfitting and prepares the model to recognize abnormalities across a spectrum of presentations, ensuring reliability in diverse clinical scenarios.

3. Prediction:

In the testing phase, our model was challenged with a subset of 3,000 unseen images to gauge its generalization capabilities and diagnostic accuracy. This crucial step tests the model's robustness by having it predict on data it has never encountered, thus providing a reliable measure of its performance in real-world scenarios.

The test images were processed using the DefaultPredictor from Detectron2, which produced a set of predictions for each image. These predictions consist of the detected abnormalities within the images, annotated with corresponding bounding boxes and confidence scores.

To manage the testing workflow, a dedicated pipeline was set up to orchestrate the loading of images, generation of predictions, and the formatting of these predictions into the competition's required submission format. Key functions such as format_pred and predict_batch were employed to transform the model's outputs into the structured submission.csv file. This file encapsulates all the predictions made by the model across the test dataset, presenting them in a standardized format that includes the class, confidence level, and bounding box coordinates for each anomaly detected.

Post-prediction, visualizations are created by superimposing the predicted bounding boxes onto the corresponding radiographic images. These visualizations serve as an immediate visual reference to evaluate the precision of the model's predictions and to identify any discrepancies or notable patterns.

(see figure 14).

image_id	PredictionString
8dec5497ec	0 0.7750 259.5887756347656 127.22438049316406 320.6328430175781 188.27528381347656 13 0.5030 366.079833984375 114.684684753417970 40
287422bed1	3 0.9863 146.17098999023438 232.75823974609375 408.9172058105469 328.5975341796875 0 0.7895 250.22015380859375 105.44329071044922 3
1d12b94b7a	0 0.8116 241.038330078125 169.54270935058594 294.41900634765625 215.4768524169922 3 0.7831 177.21856689453125 297.3041687011719 391
6b872791e2	11 0.3289 449.7713928222656 449.7750549316406 475.2036437988281 478.5747985839844 10 0.2717 13.282649040222168 446.64642333984375 3
d0d2addff91	0 0.8511 311.87353515625 149.5022735595703 374.15008544921875 205.97447204589844 3 0.7408 254.93898010253906 277.24734497070310 463.(
56beeb2009	3 0.9958 203.94021606445312 307.68533325195310 430.6386413574219 382.7339172363281 0 0.9853 294.0898742675781 206.44705200195312 350
645e178a29f	0 0.9939 260.8909912109375 97.35646057128906 322.6883544921875 167.67039489746094 11 0.2875 334.3214416503906 25.807382583618164 37
1c146ccd98	0 0.9857 260.3519592285156 158.59767150878906 314.8197937011719 209.1320343017578 3 0.8283 181.13278198242188 272.34808349609375 37
cd71f215993	13 0.8964 123.239013671875 156.8814239501953 165.56777954101562 215.45774841308594 11 0.7908 132.32595825195312 105.75555419921875
7bbef01d8c9	0 0.6018 270.2322082519531 117.18708801269531 319.2137756347656 162.5652313232422 2 0.2174 319.655517578125 187.6204071044922 356.2(
e06a136bef4	3 0.9982 179.80584716796875 236.53709411621094 432.25262451171875 333.5412292480469 0 0.4167 286.97491455078125 119.4190673828125 3
24d7e092a7	0 0.8638 271.55914306640625 148.0337677001953 338.94378662109375 209.16355895996094 3 0.7183 212.17880249023438 280.2935791015625 4
e1681b6ced	0 0.9452 249.54701232910156 105.25276184082031 306.60888671875 159.8262176513672 11 0.4586 291.4340515136719 35.36178970336914 350.(
17520d7b23	11 0.0710 120.62934875488281 30.484189987182617 172.03404235839844 66.16984558105469 11 0.0655 133.908935546875 27.887237548828125
93239daad5	0 0.9136 232.2572021484375 132.65728759765625 286.8580322265625 181.45059204101562 3 0.4090 173.52247619628906 278.8565673828125 37
703cd1b366	0 0.6961 260.5320739746094 173.9116973876953 318.2926940917969 228.54063415527344 3 0.2038 171.7884521484375 336.74127197265625 386
d73ac8ac75	9 0.4444 432.447998046875 141.05474853515625 445.5784912109375 154.63287353515625 11 0.2039 185.14730834960938 41.81868362426758 23
81fd659ccf1	3 0.7922 162.58377075195312 283.84210205078125 402.3016052246094 362.03729248046875 11 0.7425 293.5771179199219 44.98918914794922 3
3457c18599	0 0.9028 258.7891845703125 137.14654541015625 324.049560546875 200.55361938476562 3 0.3706 169.82688903808594 310.0987548828125 401
6100bc75e0	3 0.9465 170.61212158203125 227.32337951660156 393.89434814453125 292.9677429199219 0 0.8888 259.2715759277344 122.75959777832031 3
285becd123	10 0.7542 277.0570068359375 25.492416381835938 476.128662109375 382.3497314453125 7 0.2769 309.8046875 62.57678985595703 440.762817
8bfad13f756	3 0.9868 164.44387817382812 280.48907470703125 364.5272521972656 349.30718994140625 0 0.9665 236.46307373046875 155.33294677734375

Figure 14:Model Test Predctions

The output of this process, detailed in the submission.csv file, is a testament to the model's predictive capabilities. It not only represents the culmination of the model's training and validation efforts but also provides a clear and actionable output that can be subjected to further analysis to refine and enhance model performance.

V. Ethical Concerns :

In the pursuit of technological advancements, particularly in the field of artificial intelligence applied to medical imaging, ethical considerations must be at the forefront of the discussion. This project, while contributing significantly to the domain of thoracic disease detection using the VinBigData Chest X-ray Abnormalities Detection dataset, is not exempt from these crucial considerations.

1. Patient Privacy and Data Security:

Regarding patient privacy, there are serious issues when using X-ray pictures of patients to train AI models. Even if the dataset was anonymised, it is crucial to make sure that all patient data is kept safe and that the models don't unintentionally learn how to reconstruct personal information.

2. Error and Accountability:

Liability in the event that AI makes a diagnostic error is still a complicated matter. Ethical and legal analysis is necessary to decide who is accountable: the AI itself, the radiologists utilizing the system, or the software creators.

3. Transparency and Trust:

While AI can significantly aid in diagnosing diseases, it's imperative that the AI's decision-making process is transparent, and the healthcare professionals trust its judgments. Ensuring that the system's workings are interpretable to radiologists and other clinicians is essential for its acceptance and ethical deployment.

VI. Conclusion :

The journey of the VinBigData Chest X-ray Abnormalities Detection project stands as a testament to the transformative potential of artificial intelligence in enhancing diagnostic accuracy and patient care.The investigation into chest X-ray interpretation using AI technologies—including data preprocessing, deep learning, and performance assessment—offered a rich opportunity for innovation and the development of expertise.

The intricate task of detecting and classifying anomalies in the thoracic region refined the project's technical abilities and showcased the potential of AI to enhance—or even surpass—human precision in diagnostic processes. This project has not only been about improving model accuracy; it has also created opportunities for possible integration into clinical settings, which might lessen the strain on healthcare systems and guarantee that all patients have equal access to professional diagnosis.

Drawing inspiration from the progressive strides in computing, such as those mentioned by Kupiainen, this journey underscores the significance of nurturing technological developments that align with sustainability in healthcare. It advocates for the investment in not only the infrastructure but also the human element that interprets and leverages AI's insights responsibly.

In summary, this project surpassed the confines of academia, transforming into a deep exploration of the ethical and societal implications of implementing AI in healthcare. It fosters a duty to advance with technologically sophisticated, environmentally conscious, and ethically sound approaches, poised to address the significant healthcare challenges of our time. As we gaze toward the future, the unwavering commitment to responsible AI remains a guiding beacon, holding the promise of revolutionizing global health and individual well-being

VII. References :

[1] VinBigData Chest X-ray Abnormalities Detection;

https://www.kaggle.com/competitions/vinbigdata-chest-xray-abnormalities-detection/overview

[2] Enhancing Diagnostic Accuracy with CADe and CADx Systems in Breast Cancer Detection;
https://www.thieme-connect.de/products/ejournals/abstract/

[3] Computer-assisted diagnosis for an early identification of lung cancer in chest X rays;
https://www.nature.com/articles/s41598-023-34835-z

[4] VinDr-CXR: an open dataset of chest X-rays with radiologist's annotations;

https://www.nature.com/articles/s41597-022-01498-w

VIII. Appendix:

```
import torch
import detectron2
from detectron2.utils.logger import setup_logger
from tqdm.notebook import tqdm

from IPython.display import display, HTML
import subprocess

# --- Visualisation Imports  ---
import plotly.offline as py
py.init_notebook_mode(connected=True)
import plotly.io as pio
pio.templates.default = "plotly_dark"

!nvidia-smi
!nvcc --version

!python -m pip install pyyaml==5.1
import sys, os, distutils.core

dist = distutils.core.run_setup("./detectron2/setup.py")
required_packages = dist.install_requires
print(required_packages)
# Function to install packages using pip
def install_packages(packages):
    subprocess.check_call([sys.executable, "-m", "pip", "install"] +
packages)

# Call the function with the list of required packages
install_packages(required_packages)
sys.path.insert(0, os.path.abspath('./detectron2'))


!nvcc --version
TORCH_VERSION = ".".join(torch.__version__.split(".")[:2])
CUDA_VERSION = torch.__version__.split("+")[-1]
print("torch: ", TORCH_VERSION, "; cuda: ", CUDA_VERSION)
```

```
print(f"CUDA Available: {torch.cuda.is_available()}")
print("detectron2:", detectron2.__version__)


if torch.cuda.is_available():
    print(f"CUDA Version: {torch.version.cuda}")
    print(f"Device Name: {torch.cuda.get_device_name(0)}")


import os
import pickle
import numpy as np
import pandas as pd
import cv2
import yaml
import torch
from pathlib import Path
from typing import Any, Dict, List, Union
from tqdm import tqdm
from math import ceil
from detectron2.config import get_cfg
from detectron2 import model_zoo
from detectron2.engine import DefaultPredictor
from detectron2.utils.visualizer import Visualizer, ColorMode
from detectron2.data import DatasetCatalog, MetadataCatalog
from dataclasses import dataclass, field

setup_logger()

#Constants
THING_CLASSES = [
    "Aortic enlargement", "Atelectasis", "Calcification",
"Cardiomegaly",
    "Consolidation", "ILD", "Infiltration", "Lung Opacity",
"Nodule/Mass",
    "Other lesion", "Pleural effusion", "Pleural thickening",
"Pneumothorax",
    "Pulmonary fibrosis"
]
CATEGORY_NAME_TO_ID = {class_name: index for index, class_name in
enumerate(THING_CLASSES)}
```

```
            print(f"Loading from cache {cache_path}")
            with open(cache_path, "rb") as f:
                dataset_dicts = pickle.load(f)

        return dataset_dicts

    # Prediction and formatting
    def format_pred(labels: np.ndarray, boxes: np.ndarray, scores:
    np.ndarray) -> str:
        """Format the prediction strings for submission."""
        pred_strings = [f"{label} {score} {xmin} {ymin} {xmax} {ymax}"
    for label, score, (xmin, ymin, xmax, ymax) in zip(labels, scores,
    boxes.astype(np.int64))]
        return " ".join(pred_strings)

    def predict_batch(predictor: DefaultPredictor, im_list:
    List[np.ndarray]) -> List:
        """Run model prediction on a batch of images."""
        inputs_list = []
        for original_image in im_list:
            if predictor.input_format == "RGB":
                original_image = original_image[:, :, ::-1]

            image =
    torch.as_tensor(original_image.astype("float32").transpose(2, 0, 1))
            inputs = {"image": image, "height": original_image.shape[0],
    "width": original_image.shape[1]}
            inputs_list.append(inputs)

        with torch.no_grad():
            predictions = predictor.model(inputs_list)
        return predictions
    @dataclass

    class Flags:
        """Configuration flags."""
        # Providing default values for all fields to allow no-argument
    instantiation
        imgdir_name: str = "proc_data_512"
        lr_scheduler_name: str = "WarmupCosineLR"
        outdir: str = "results/test_512x512"
        split_mode: str = "all_train"
```

```
    train_data_type: str = "original"
    base_lr: float = 0.001
    debug: bool = False
    eval_period: int = 2000
    ims_per_batch: int = 2
    iter: int = 30000
    num_workers: int = 4
    roi_batch_size_per_image: int = 512
    seed: int = 111
    use_class14: bool = False
    aug_kwargs: Dict = field(default_factory=lambda: {})

    def update(self, param_dict: Dict) -> "Flags":
        """Updates the Flags object with values from a given
dictionary."""
        for key, value in param_dict.items():
            setattr(self, key, value)
        return self
```