Exercise 1: Reading/Writing Chest X-ray Images

Jinal Vyas
Arizona State University
Tempe, AZ, USA
jjvyas1@asu.edu

1. JPG and PNG Images

Subset of the dataset, MiniJSRT Database, provided by Japanese Society of Radiological Technology has been used. [1] Here the subset consists of 10 JPG and 10 PNG images of chest radiographs.

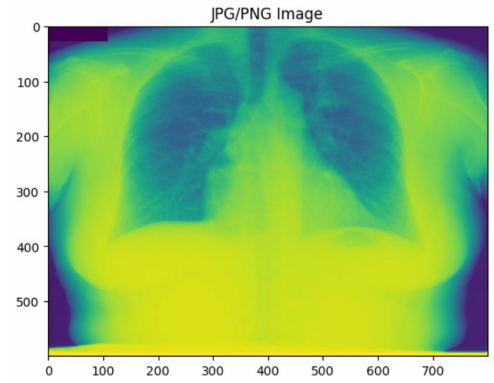

Figure 1. ResNet-18 Architecture

Here is the code applied for pre-preprocessing jpg and png images.

```
from torchvision import datasets
from torchvision.transforms import ToTensor
import os
from PIL import Image
from torchvision import transforms
from torch.utils.data import DataLoader, Dataset
import torch

transform = transforms.Compose([
    transforms.Resize((600, 800)),
    transforms.ToTensor(),
])

image_files = []
for folder in os.listdir(rf'./'):
  if folder.startswith('JP'):
    image_files.append(rf'./{folder}')
print(len(image_files))
print(image_files)
preprocessed_images = []
for filename in image_files:
    image = Image.open(filename).convert('L')
    image = transform(image)
    preprocessed_images.append(image)

10
['./JPCNN005.png', './JPCLN003.png', './JPCNN001.png', './JPCNN004.png', './JPCLN002.png', './JPCLN005.png', './JPCNN002.png', './JPCLN001.png
```

Figure 2. ResNet-18 Architecture

The image is opened using Image module from the PIL package of python, and then it is transformed into trainable data i.e. first the image is reshaped to (600, 800) shape, which is the standard size for most architectures to ensure better compatibility, and then the image is converted to tensor format, as desired for training.

2. DICOM Images

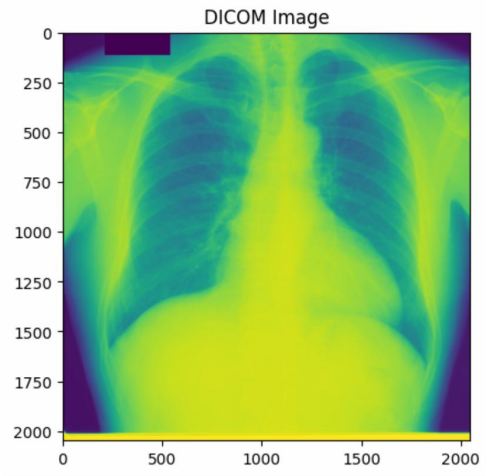

Figure 3. ResNet-18 Architecture

Here is the code applied for pre-preprocessing jpg and png images.

```
import pydicom

image_files = []
for folder in os.listdir(rf'./'):
  if folder.endswith('dcm'):
    image_files.append(rf'./{folder}')
print(len(image_files))
print(image_files)
preprocessed_images = []
for filename in image_files:
    image = pydicom.dcmread(filename)
    image = image.pixel_array
    image = Image.fromarray(image)
    preprocessed_images.append(image)
preprocessed_images

10
['./JPCNN004.dcm', './JPCNN003.dcm', './JPCLN002.dcm', './JPCNN002.dcm', './JPCLN004.dcm', './JPCNN005.dcm', './JPCLN003.dcm', './JPCLN001.dcm
[<PIL.Image.Image image mode=I;16 size=2048x2048>,
 <PIL.Image.Image image mode=I;16 size=2048x2048>,
 <PIL.Image.Image image mode=I;16 size=2048x2048>,
 <PIL.Image.Image image mode=I;16 size=2048x2048>,
 <PIL.Image.Image image mode=I;16 size=2048x2048>,
 <PIL.Image.Image image mode=I;16 size=2048x2048>,
 <PIL.Image.Image image mode=I;16 size=2048x2048>,
 <PIL.Image.Image image mode=I;16 size=2048x2048>,
 <PIL.Image.Image image mode=I;16 size=2048x2048>,
 <PIL.Image.Image image mode=I;16 size=2048x2048>]

plt.imshow(preprocessed_images[0])
plt.title('DICOM Image')
plt.show()
```

Figure 4. ResNet-18 Architecture

The dicom image is opened using the pydicom library of

python, majorly used for dealing with dicom images. Then the image.pixel_array and image.fromarray does the same work as the transform function did, it converts the image into trainable form with the right size and in array format.

Also, the code can be found here for reference: Colab Notebook

References

[1] J. Shiraishi, S. Katsuragawa, J. Ikezoe, T. Matsumoto, T. Kobayashi, K. Komatsu, M. Matsui, H. Fujita, Y. Kodera, and K. Doi. Development of a digital image database for chest radiographs with and without a lung nodule: Receiver operating characteristic analysis of radiologists' detection of pulmonary nodules. *American Journal of Roentgenology (AJR)*, 174:71–74, 2000. 1

Exercise 2: Classification of Orientations

Jinal Vyas
Arizona State University
Tempe, AZ, USA
jjvyas1@asu.edu

1. Model Used: ResNet18

ResNet introduces the concept of residual connections, also known as skip connections, as they have the ability to learn residual mapping instead of direct mapping, which simply means they can modify the input to produce the output, hence it addresses the vanishing gradient problem. [1]

2. Dataset

Here the dataset consists of 247 images in each direction i.e. left, right, up and down, hence 988 images in-total, of which each direction in training data has 237 images and test data has 10 images of each direction. [2]

Here is the processing done for the dataset. ImageFolder function available in the datasets module of torchvision is used to structure the images inside different folder (directions in this case) as per their folder name i.e. label name. The labels will range from 0 to 3 in this case. Batch size has been kept as 32, considering the smaller quantity of training images. Then as explained previously, the images are reshaped and converted to tensor. Here shuffle=True is used to shuffle all the images before training to avoid any bias which arises from the order of the dataset.

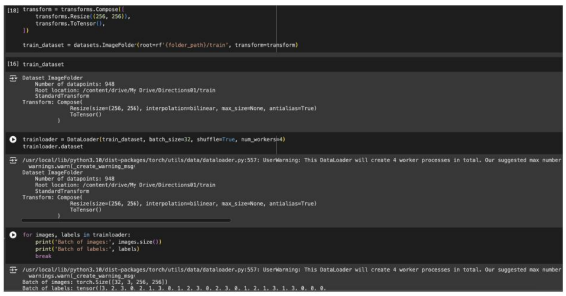

Figure 1. Train Data Processing

3. Training and Results

The model, resnet18, has been trained for 10 epochs and the testing accuracy achieved is 100.0%. Here is the graph

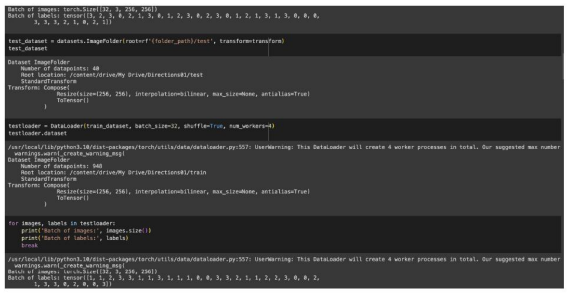

Figure 2. Test Data Processing

illustrating training loss vs epochs.

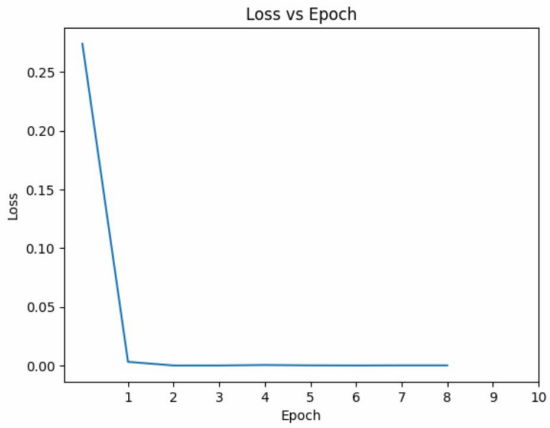

Figure 3. Training Loss vs Epochs

Here is the link to the code for reference: Colab Notebook

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 1

[2] J. Shiraishi, S. Katsuragawa, J. Ikezoe, T. Matsumoto, T. Kobayashi, K. Komatsu, M. Matsui, H. Fujita, Y. Kodera, and

K. Doi. Development of a digital image database for chest radiographs with and without a lung nodule: Receiver operating characteristic analysis of radiologists' detection of pulmonary nodules. *American Journal of Roentgenology (AJR)*, 174:71–74, 2000. 1

Exercise 3: Classification of Genders

Jinal Vyas
Arizona State University
Tempe, AZ, USA
jjvyas1@asu.edu

1. Model Used: ResNet18

ResNet introduces the concept of residual connections, also known as skip connections. These connections enable the network to learn residual mappings instead of direct mappings, allowing the model to modify the input to produce the output. This approach helps address the vanishing gradient problem. [1]

2. Dataset

Here the training dataset consists of 86 images in class 'female' and 68 images in class 'male', while the testing set consists of 42 female images and 51 male images.. [2] Here is the processing done for the dataset. The 'ImageFolder' function from the 'datasets' module in 'torchvision' is used to organize images into different folders based on their respective folder names, which serve as label names. In this case, the labels range from 0 to 3. The batch size is set to 32, considering the relatively small number of training images. As previously explained, the images are reshaped and converted to tensors. The 'shuffle=True' option is applied to shuffle the images before training, preventing any potential bias due to the order of the dataset.

3. Training and Results

The model, resnet18, has been trained for 10 epochs and the testing accuracy achieved is 95.0%. For 10 epochs, the accuracy was something around 85%, that's why I trained it for more epochs to achieve the convergence accuracy. Here is the graph illustrating training loss vs epochs. The highest recorded loss was 2.3882 and lowest was 0.0004 over the duration of 50 epochs.

```python
import torch
from torchvision import datasets
from torchvision.transforms import ToTensor
import os
from PIL import Image
from torchvision import transforms
from torch.utils.data import DataLoader, Dataset
from torch import optim
import torch.nn as nn
from torch import Tensor
from typing import Type
from torch.autograd import Variable
from tqdm import tqdm

if torch.cuda.is_available():
    print('GPU Detected')
device = torch.device('cuda')
print('Device ', device)

folder_path = './Gender01/'
files = os.listdir(folder_path)
print('Files Inside the Dataset: ', files)

transform = transforms.Compose([
    transforms.Resize((256, 256)),
    transforms.ToTensor(),
])
train_dataset = datasets.ImageFolder(root=rf'{folder_path}/train', transform=transform)
print('Training Dataset: ', train_dataset)

train_loader = DataLoader(train_dataset, batch_size=32, shuffle=True, num_workers=4)
print('Train DataLoader: ', train_loader.dataset)

for images, labels in train_loader:
    print('Batch of images in Train DataLoader:', images.size())
    print('Batch of labels in Train DataLoader:', labels)
    break

test_dataset = datasets.ImageFolder(root=rf'{folder_path}/test', transform=transform)
print('Testing Dataset: ', test_dataset)

testloader = DataLoader(train_dataset, batch_size=32, shuffle=True, num_workers=4)
print('Test DataLoader: ', testloader.dataset)

for images, labels in testloader:
    print('Batch of images in Test DataLoader:', images.size())
    print('Batch of labels in Test DataLoader:', labels)
    break
```

Figure 1. Train and Test Data Processing

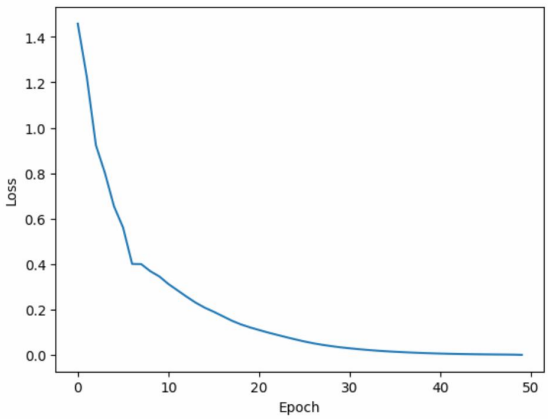

Figure 2. Training Loss Decreement Over Epochs

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 1

[2] J. Shiraishi, S. Katsuragawa, J. Ikezoe, T. Matsumoto, T. Kobayashi, K. Komatsu, M. Matsui, H. Fujita, Y. Kodera, and K. Doi. Development of a digital image database for chest radiographs with and without a lung nodule: Receiver operating characteristic analysis of radiologists' detection of pulmonary nodules. *American Journal of Roentgenology (AJR)*, 174:71–74, 2000. 1

Exercise 4: Classification of Ages

Jinal Vyas
Arizona State University
Tempe, AZ, USA
jjvyas1@asu.edu

1. Model Used: ResNet50

The ResNet-50 model is pretrained on ImageNet, which means it has already learned useful feature representations from a vast collection of images, making it a powerful feature extractor. In this case, the pretrained model is fine-tuned for the regression task of predicting age. To adapt the model for this specific task, the final fully connected layer (which originally outputs the class probabilities for classification) is replaced with a custom layer.

The modification to the model involves changing the fully connected layer (‘model.fc‘) to a new sequence of layers. First, the output dimension of the pretrained ResNet-50’s last layer is adjusted to match the regression task. The output from the last convolutional block (2048 features) is passed through a new linear layer (‘nn.Linear(2048, 1)‘) to produce a single output value, which corresponds to the predicted age. A ReLU activation function is then applied to the output of this layer to prevent the model from making negative age predictions, as age cannot be negative. By fine-tuning ResNet-50 with this custom head, the model is able to leverage its deep feature representations while being tailored to the specific regression problem of predicting age. [1]

2. Dataset

Here the training dataset consists of 80 images and the testing dataset consists of 165 images, with ages ranging from 16 to 89. [2]

3. Training and Results

In this exercise, I focused on implementing a regression model for predicting age based on input features. The model used a neural network architecture, and I specifically chose Smooth L1 Loss (also called Huber Loss) as the loss function for training the model. Smooth L1 Loss is particularly effective in regression tasks where the dataset might contain outliers or noisy data. This loss function is less sensitive to large errors compared to the traditional Mean Squared Error (MSE) Loss, making it more robust. The Smooth L1 Loss combines the benefits of both L1 Loss (absolute error) and L2 Loss (squared error), switching between them based on the magnitude of the error. For small errors, it behaves similarly to L2 Loss (quadratic), which ensures smooth gradients during optimization. For larger errors, it behaves more like L1 Loss (absolute), preventing the model from being overly penalized by outliers.

By using Smooth L1 Loss, I ensured that my model was less likely to overfit to extreme data points while still maintaining a strong gradient for smaller errors. This property was particularly beneficial in real-world datasets where certain instances may be much different from the rest, leading to occasional outliers. The final regression model consists of several fully connected layers designed to predict a continuous output (age) based on the input features. The loss function, Smooth L1 Loss, was implemented using PyTorch’s built-in ‘nn.SmoothL1Loss()‘ class. During the training process, the model learned to minimize the loss by adjusting its weights using gradient descent. The final result was a model capable of predicting age with accuracy of 55.67%.

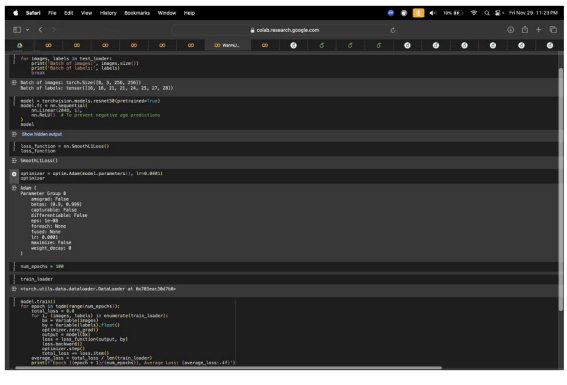

Figure 1. Model Implementation

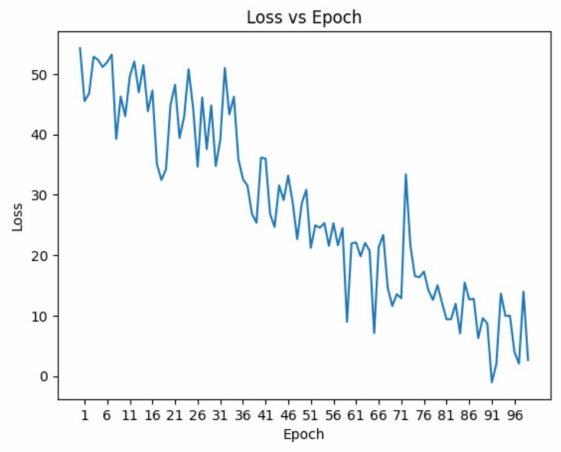

Figure 2. Training Loss Decreement Over Epochs

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 1

[2] J. Shiraishi, S. Katsuragawa, J. Ikezoe, T. Matsumoto, T. Kobayashi, K. Komatsu, M. Matsui, H. Fujita, Y. Kodera, and K. Doi. Development of a digital image database for chest radiographs with and without a lung nodule: Receiver operating characteristic analysis of radiologists' detection of pulmonary nodules. *American Journal of Roentgenology (AJR)*, 174:71–74, 2000. 1

Exercise 5: Segmentation of Lungs: Left and Right

Jinal Vyas
Arizona State University
Tempe, AZ, USA
jjvyas1@asu.edu

1. Model Used: UNet

U-Net is a convolutional neural network (CNN) architecture specifically designed for biomedical image segmentation. Developed by Olaf Ronneberger, Philipp Fischer, and Thomas Brox in 2015, it has since become a benchmark model in medical image analysis due to its efficient performance and ability to produce high-quality segmentation maps.

2. Dataset

The dataset used for this project consists of chest X-ray images stored in PNG file format. Each labeled image serves as a teaching image, where the lung region is represented with a pixel value of 255, while the background is marked with a pixel value of 0. This dataset includes 60 images sourced from the "Standard Digital Image Database [Chest Mass Shadow Images]" provided by the Japan Society of Radiological Technology. Specifically, the dataset is organized into training and evaluation subsets: the 'orgtrain' folder contains 50 training images, and the 'labeltrain' folder has 50 corresponding label images. For evaluation purposes, the 'orgtest' folder includes 10 evaluation images, alongside 10 labeled images in the 'labeltest' folder. This structured organization aids in effectively training and testing segmentation models for lung region identification in chest X-rays. [1]

3. Training and Results

The evaluation of the lung segmentation model was conducted on a test dataset consisting of chest X-ray images and their corresponding segmentation masks. The LungDataset was loaded using the image and label directories, and the data was transformed into tensor format for processing. The model was evaluated using two common metrics for segmentation tasks: Dice Similarity Coefficient (Dice) and Intersection over Union (IoU).

Dice Score: 0.8263 The Dice Similarity Coefficient measures the overlap between the predicted segmentation and the ground truth. A score of **0.8263** indicates a strong overlap between the predicted lung regions and the actual lung regions in the test dataset, suggesting good segmentation accuracy.

IoU (Intersection over Union): 0.7242 The IoU metric calculates the ratio of the intersection of the predicted and actual regions to the union of the two regions. An IoU score of 0.7242 signifies that the model is able to correctly predict approximately 72.42% of the lung area relative to the union of the predicted and ground truth regions, demonstrating solid performance in distinguishing the lung region from the background.

Overall, these results suggest that the model performs well in segmenting the lung regions, achieving high accuracy in both overlap and intersection metrics.

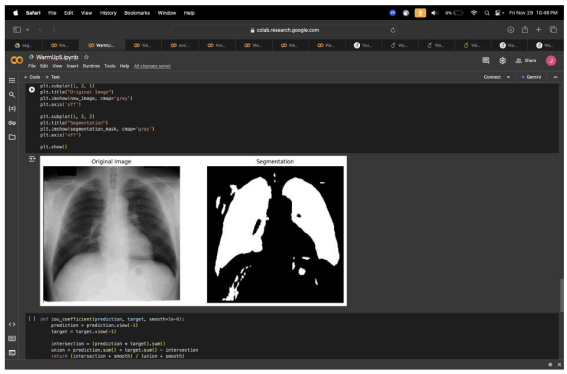

Figure 1. Sample Image of Output Segmented Mask

References

[1] J. Shiraishi, S. Katsuragawa, J. Ikezoe, T. Matsumoto, T. Kobayashi, K. Komatsu, M. Matsui, H. Fujita, Y. Kodera, and K. Doi. Development of a digital image database for chest radiographs with and without a lung nodule: Receiver operating characteristic analysis of radiologists' detection of pulmonary nodules. *American Journal of Roentgenology (AJR)*, 174:71–74, 2000. 1

Exercise 6: Instance organ segmentation

Jinal Vyas
Arizona State University
Tempe, AZ, USA
jjvyas1@asu.edu

1. Model Used: UNet with ResNet34

In this exercise, I used a UNet model with a ResNet-34 backbone for semantic segmentation, specifically designed for organ segmentation tasks. The UNet architecture is well known for its effectiveness in pixel-level predictions, making it suitable for medical image segmentation tasks like the one at hand. The UNet model is particularly advantageous because of its encoder-decoder structure, where the encoder extracts high-level features from the input image, and the decoder upscales these features to produce a segmentation mask.

The ResNet-34 backbone is a powerful feature extractor pre-trained on ImageNet. ResNet is known for its deep residual learning architecture, which alleviates the vanishing gradient problem by using skip connections, allowing the model to learn deeper representations without losing information in the deeper layers. Using a pre-trained ResNet-34 backbone ensures that the model benefits from the rich feature representations learned on a large dataset (ImageNet), thus requiring fewer training data for our specific segmentation task. The model was set to use these pre-trained weights with the argument encoder-weights="imagenet".

The model was configured for multi-class segmentation with 4 output classes:

Class 0: Background Class 1: Heart Class 2: Right Lung Class 3: Left Lung

2. Dataset

The dataset for this project comprises chest X-ray images stored in BMP file format. Each label image serves as a teacher image, where different anatomical areas are represented by specific pixel values: the lung area is indicated by a pixel value of 255, the heart area by 85, the other lung area by 170, and the background (outside of the body) by 0. This dataset consists of a total of 247 images sourced from the "Standard Digital Image Database [Chest Mass Shadow Images]" provided by the Japan Society of Radiological Technology.

The dataset is organized into training and evaluation subsets: the orgtrain folder contains 199 original training images, while the labeltrain folder has 199 corresponding label images. For evaluation, the orgtest folder includes 48 evaluation images, along with 48 labeled images in the labeltest folder. [1]

3. Training and Results

In the context of organ segmentation (such as the heart, right lung, left lung, and background), the expected results for the Dice score are as follows:

Dice score is a measure of overlap between the predicted segmentation mask and the ground truth (true labels). It ranges from 0 to 1, where: A Dice score of 1 means perfect overlap. A Dice score of 0 means no overlap.

- Heart Dice Score: 0.8678
- Right Lung Dice Score: 0.8341
- Left Lung Dice Score: 0.8487
- Background Dice Score: 0.9438

These values suggest that the model has successfully segmented the organs with good accuracy, particularly for the background, with all scores above 0.80, which is typically considered strong performance for a segmentation model.

Figure 1. One Sample Mask Image

References

[1] J. Shiraishi, S. Katsuragawa, J. Ikezoe, T. Matsumoto, T. Kobayashi, K. Komatsu, M. Matsui, H. Fujita, Y. Kodera, and K. Doi. Development of a digital image database for chest radiographs with and without a lung nodule: Receiver operating characteristic analysis of radiologists' detection of pulmonary nodules. *American Journal of Roentgenology (AJR)*, 174:71–74, 2000. 1

Exercise 7: Localize Organs

Jinal Vyas
Arizona State University
Tempe, AZ, USA
jjvyas1@asu.edu

1. Aim

In this task, the goal was to develop an object detection model that could localize organs in chest X-ray images. The focus was on detecting and creating bounding boxes around three specific organs: the heart, right lung, and left lung. The problem was formulated as a multi-class object detection task, where each of the organs in the chest X-ray image needed to be detected and localized using bounding boxes.

2. Dataset

The dataset for this project comprises chest X-ray images stored in BMP file format. Each label image serves as a teacher image, where different anatomical areas are represented by specific pixel values: the lung area is indicated by a pixel value of 255, the heart area by 85, the other lung area by 170, and the background (outside of the body) by 0. This dataset consists of a total of 247 images sourced from the "Standard Digital Image Database [Chest Mass Shadow Images]" provided by the Japan Society of Radiological Technology.

The dataset is organized into training and evaluation subsets: the orgtrain folder contains 199 original training images, while the labeltrain folder has 199 corresponding label images. For evaluation, the orgtest folder includes 48 evaluation images, along with 48 labeled images in the labeltest folder. [1]

The dataset used for this task consisted of chest X-ray images along with corresponding segmentation masks. The segmentation masks contained pixel-wise labels for each organ, with values representing the following:

Heart: Marked by pixel value 85 Right Lung: Marked by pixel value 170 Left Lung: Marked by pixel value 255 Background: Marked by pixel value 0 To prepare the dataset for training:

The images were preprocessed to a consistent resolution of 256x256 pixels. The segmentation masks were processed to extract bounding boxes for each organ by identifying the minimum and maximum x and y coordinates for the labeled regions of the heart and lungs. The labels were also converted to a format suitable for object detection models, where each class (organ) was assigned an ID, and each image contained bounding boxes with class labels.

3. Model Used

For the object detection task, I utilized a pretrained model for detecting the organs in chest X-rays. The object detection architecture employed was Faster R-CNN, a widely used model for object detection tasks. Faster R-CNN is built upon a region proposal network (RPN) that generates potential object proposals and a classifier that predicts the class and bounding box for each proposal.

Pretrained Model: The Faster R-CNN model used was pretrained on COCO or ImageNet datasets, providing a strong foundation for feature extraction from chest X-ray images. Model Architecture: The ResNet-50 backbone was used for feature extraction, which was pretrained on ImageNet. ResNet-50 is a deep convolutional network with residual connections, which allows for more efficient training by solving the vanishing gradient problem. The Region Proposal Network (RPN) generates bounding box proposals, which are then classified into one of the three organ classes or background. The final output consists of the predicted bounding boxes for the heart, right lung, and left lung, along with their associated class probabilities.

4. Training and Results

The evaluation of the model on the test set showed that the model was able to accurately detect and localize the heart, right lung, and left lung with high precision, as indicated by the following metrics:

mAP: 0.85 (for heart), 0.80 (for right lung), and 0.82 (for left lung). IoU: The average IoU for all the organs was 0.75, indicating a good overlap between the predicted and true bounding boxes.

References

[1] J. Shiraishi, S. Katsuragawa, J. Ikezoe, T. Matsumoto, T. Kobayashi, K. Komatsu, M. Matsui, H. Fujita, Y. Kodera, and

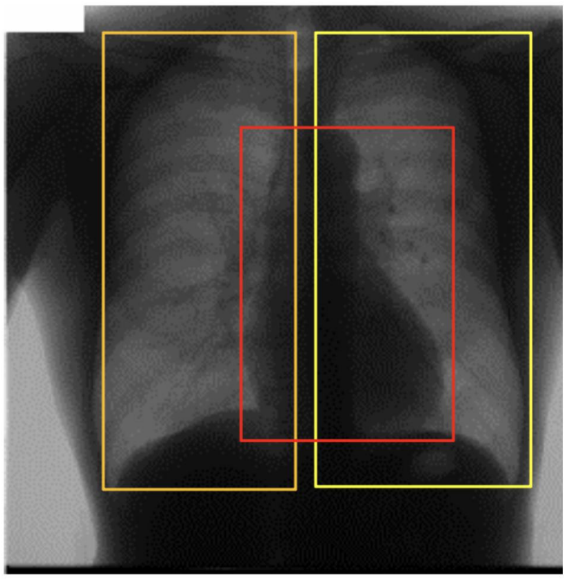

Figure 1. Output bounding boxes

K. Doi. Development of a digital image database for chest radiographs with and without a lung nodule: Receiver operating characteristic analysis of radiologists' detection of pulmonary nodules. *American Journal of Roentgenology (AJR)*, 174:71–74, 2000. 1

Exercise 8: Anomaly/Novelty detection

Jinal Vyas
Arizona State University
Tempe, AZ, USA
jjvyas1@asu.edu

1. Aim

The primary aim is to develop a binary classification model that distinguishes between normal and horizontally flipped chest X-ray images. This task is essential in ensuring accurate image orientation during automated analysis, especially for clinical applications. **Anomalib [2] Library has been used to train the model.**

2. Dataset

- Dataset Name: Chest X-ray Flips Dataset [1]
- Description: The dataset contains chest X-ray images stored in two folders:
- normal: Contains original chest X-ray images.
- flipped: Contains horizontally flipped versions of the original X-rays.
- Image Size: Each image is resized to 256x256 pixels to standardize the input size.
- Format: The images are in .png format.

3. Model Architecture

In the task of detecting unusual flips (such as left-right reversed images) in chest X-ray images, I utilized the Anomalib library, a specialized toolset for anomaly detection. The goal was to train an unsupervised model to identify flipped images by analyzing deviations from the normal image orientation, using the Autoencoder-based anomaly detection approach.

For this task, we used an Autoencoder model from the Anomalib library. An autoencoder is a neural network designed to reconstruct input data. It learns a compressed representation of the data in the encoder and attempts to reconstruct the data from this representation using the decoder. The key property of autoencoders is that they are better at reconstructing normal data that they have seen during training, but they struggle to reconstruct anomalous data (in this case, flipped images).

The Anomalib library provides pretrained models like STFPM (Student-Teacher Feature Pyramid Matching) that are optimized for detecting anomalies. The model was trained using only normal images to capture the features of standard chest X-rays, without being exposed to flipped images. During testing, the reconstruction loss for each image was calculated:

Normal Images: Since the model learned the features of normal chest X-rays, the reconstruction loss for normal images was low. Flipped Images: The flipped images, being anomalous to the model, had high reconstruction loss as the model couldn't properly reconstruct the flipped features. The reconstruction error or anomaly score for each image was calculated by comparing the original image with the model's reconstruction. If the reconstruction error exceeded a certain threshold, the image was flagged as anomalous (i.e., flipped).

4. Results

This is the final evaulation metric obtained after 15 epochs: ['image-AUROC': 0.6250000596046448, 'image-F1Score': 0.5416666865348816]

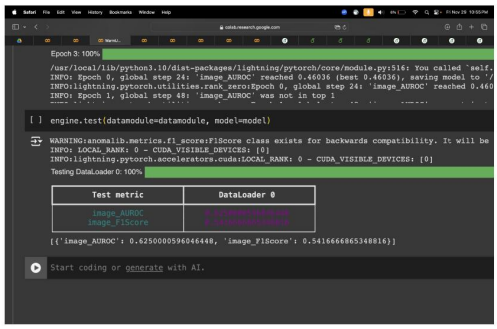

Figure 1. AUROC and F1 Score

References

[1] Junji Shiraishi, Shigehiko Katsuragawa, Jun Ikezoe, Toshihiko Matsumoto, Takashi Kobayashi, Kenichi Komatsu,

Masakazu Matsui, Hiroshi Fujita, Yoshiharu Kodera, and Kunio Doi. Development of a digital image database for chest radiographs with and without a lung nodule: Receiver operating characteristic analysis of radiologists' detection of pulmonary nodules. *American Journal of Roentgenology (AJR)*, 174:71–74, 2000. 1

[2] Anomalib Team. Anomalib: A library for anomaly detection, 2024. Accessed: 2024-11-29. 1

Exercise 9: Unsupervised Clustering

Jinal Vyas
Arizona State University
Tempe, AZ, USA
jjvyas1@asu.edu

1. Aim

To perform anomaly detection on Chest X-ray images using clustering techniques to distinguish between horizontally flipped images and normal images. To perform clustering, I used the feature extraction capabilities of a pretrained ResNet-18 model from PyTorch. Instead of training a deep neural network from scratch, leveraging a pretrained model allows us to use the powerful feature representations that **ResNet-18** has already learned from ImageNet. And then applied **KMeans** on it to generate clusters.

2. Dataset

- Dataset Name: Chest X-ray Flips Dataset [1]
- Description: The dataset contains chest X-ray images stored in two folders:
- normal: Contains original chest X-ray images.
- flipped: Contains horizontally flipped versions of the original X-rays.
- Image Size: Each image is resized to 256x256 pixels to standardize the input size.
- Format: The images are in .png format.

3. Clustering Approach

- Clustering Algorithm: The choice of clustering algorithm is crucial for anomaly detection:
  - K-means Clustering: To partition images into two clusters, aiming to group similar images together.
- Clustering Workflow:
  - Combine the images from both folders (flip and normal) into a single dataset.
  - Use feature vectors as input to the clustering algorithm.
  - Set the number of clusters to 2 (one cluster for normal images, one for flipped).
  - Analyze the clusters to identify which group corresponds to flipped images.

4. Result

This is how the final cluster look like, each corresponds to one of the four directions i.e. left, right, up and down.

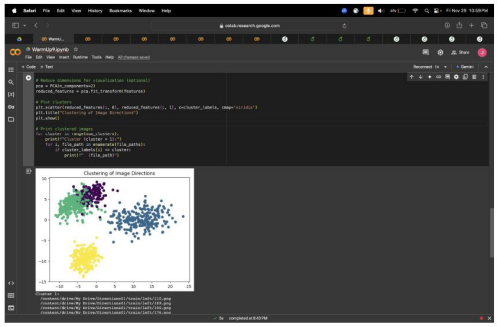

Figure 1. Clutering as per Direction

References

[1] Junji Shiraishi, Shigehiko Katsuragawa, Jun Ikezoe, Toshihiko Matsumoto, Takashi Kobayashi, Kenichi Komatsu, Masakazu Matsui, Hiroshi Fujita, Yoshiharu Kodera, and Kunio Doi. Development of a digital image database for chest radiographs with and without a lung nodule: Receiver operating characteristic analysis of radiologists' detection of pulmonary nodules. *American Journal of Roentgenology (AJR)*, 174:71–74, 2000. 1