Final Report

Nitin Reddy Yarava
Arizona State University
Tempe, AZ
nyarava@asu.edu

Abstract

This report combines findings from a series of assignments and projects involving pretraining tasks, classification, segmentation, and localization applied to diverse datasets. It encompasses the development and refinement of models through pretraining, tailored to enhance subsequent tasks such as classification and localization, across various domains. The report also details nine preliminary warmup exercises aimed at boosting computational proficiency and analytical skills.

1. Pre-Training Introduction

The project's ultimate goal is to develop a unified framework capable of integrating different medical image datasets and performing classification, localization, & segmentation on it using one model. These tasks are essential for automatic analysis of medical images, aiding in the detection, segmentation, diagnosis, and monitoring of diseases. The datasets in focus spans a range of modalities and conditions in fundus photography.

Fundus photography, also known as retinal photography, is a specialized imaging technique used to capture detailed pictures of the interior back of the eye, known as the fundus. This area includes crucial structures like the:

*** Retina:** The light-sensitive layer responsible for converting light into electrical signals.

*** Macula:** The central region of the retina responsible for sharp central vision.

*** Optic disc:** The point where the optic nerve exits the eye, connecting it to the brain.

*** Blood vessels:** Supplying blood and oxygen to the eye's different parts.

1.1. Datasets

The datasets we will be using for fundus photography are:

1. EyePACS - Retinal images by EyePACS, a free platform for retinopathy screening.
2. DRIVE - The DRIVE database has been established to enable comparative studies on segmentation of blood vessels in retinal images.
3. DRISHTI-GS - Provides a comprehensive dataset of retinal images of both normal and glaucomatous eyes with manual segmentations from multiple human experts.
4. FIRE - It is a dataset comprised of retinal image pairs annotated with ground truth and an evaluation protocol for registration methods
5. STARE - This dataset concerns a system to automatically diagnose diseases of the human eye.
6. HRF - This database was established by a collaborative research group to support comparative studies on automatic segmentation algorithms on retinal fundus images.
7. DRIONS-DB - This is a public database for benchmarking optic nerve head segmentation from digital retinal images.
8. CHASE_DB1 - It is a dataset for retinal vessel segmentation.

1.2. Previous Work

The previous work's projects aim were to advance the state-of-the-art in medical image analysis by achieving high accuracy in:

*** Classification:** Distinguishing between different conditions and categorizing images accordingly.

*** Localization:** Identifying the exact location of abnormalities within an image.

*** Segmentation:** Precisely delineating the boundaries of specific regions, organs, or abnormalities.

1.3. Contributions and Accomplishments

- **Classification:** Utilized the *ChestX-ray14* dataset on a pretrained *DINOv2* model, and results were achieved and reported in terms of accuracy.

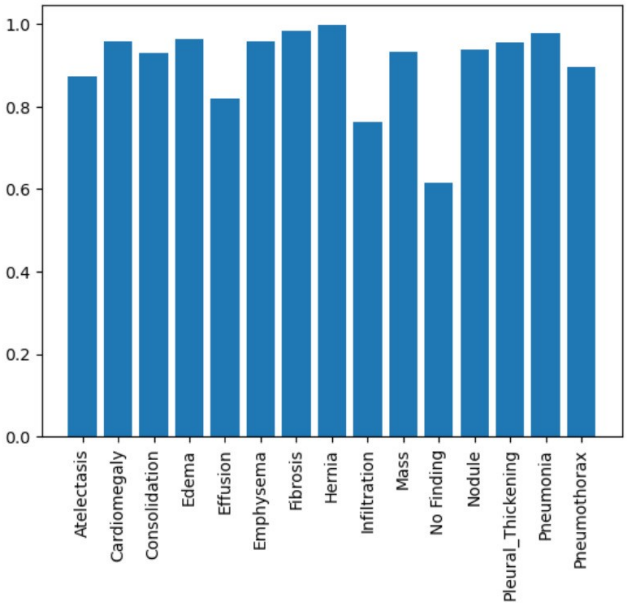


Figure 1. Results for each class in Accuracy

One part of improvement for this for this task is to report the scores in terms of the metrics used in the original paper (AUC) instead of ACC and also experiment more on hyperparameter tuning.

- **Localization:** Applied a *Mask R-CNN* model to the *TBX 11K* dataset for localizing tuberculosis-related abnormalities in chest X-rays.

- **Segmentation:** Applied a *U-Net* model to the *ChestX-Det* dataset for segmenting the chest x-rays.

  The results achieved were relatively significant as the model used was not as powerful and the model itself was downsized to have less channels in its architecture, as shown in the figure.

  Some of the areas to work on would be used a fuller version of U-Net with a Swin Backbone for the architecture and also perform classification and localization not just segmentation.

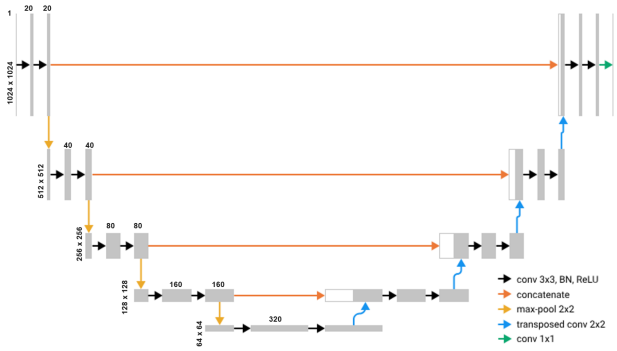


Figure 2. Downsized Architecture for Segmentation

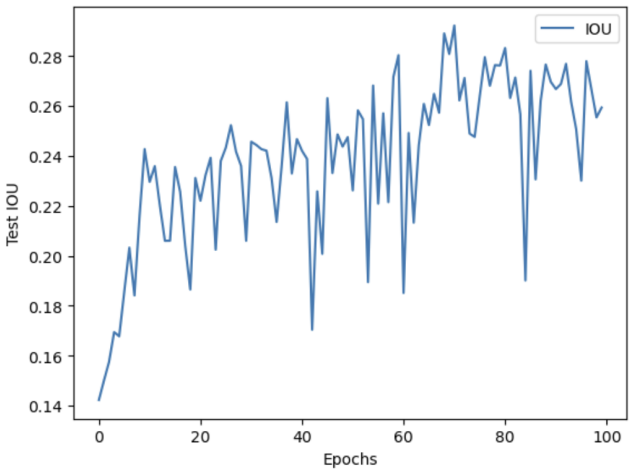


Figure 3. Max IOU: 29.22 — Baseline: 40.2

1.4. Future Work

- **Segmentation on Fundus Images:** Implement and fine-tune a segmentation model for the various fundus image datasets (e.g., EyePACS, DRIVE) to accurately segment retinal structures and lesions.

- **Integration Across Datasets:** Develop a unified model or framework that can adapt to the different tasks and datasets, possibly through transfer learning or multi-task learning approaches.

- **Evaluation and Optimization:** Extensively evaluate the models on unseen data from different datasets and modalities and optimize them for better performance and efficiency.

1.5. Challenges & Strategies

Note: These suggestions do not present explicit methods or solutions to the problem; rather, they aim to acknowledge and, to some extent, tackle the underlying issue by offering potential directions for overcoming challenges and

may not be specific to the project I am undertaking.

Integrating different datasets for training a single segmentation model in medical imaging poses several challenges, primarily due to the heterogeneity in data sources, annotation standards, image characteristics, and clinical contexts. Successfully overcoming these challenges can significantly enhance the model's robustness, generalization ability, and applicability across diverse medical conditions and imaging modalities not just in fundus photography but in other medical imaging tasks as well. Below are some of the key challenges and strategies to approach them:

1. **Variability in Image Acquisition Parameters.**

   Challenge: Medical images from fundus photography, can be different in terms of acquisition parameters including resolution, contrast, brightness, and field of view. These variations can arise from different imaging equipment, settings, and protocols used across datasets.

   Approach:

   *** Preprocessing and Standardization:** We need to implement robust image preprocessing steps to normalize the images. Techniques such as equalization can be used to standardize brightness and contrast across images. Resizing images from different datasets to a common resolution helps the model learn image characteristics regardless of the dataset.

   *** Domain Adaptation Techniques:** Utilizing domain adaptation methods to minimize the distribution gap between different datasets could also be helpful. Techniques such as employing feature-level adaptation techniques could also aid in domain adaptation.

2. **Inconsistencies in Annotations**

   Challenge: Different datasets usually come with different annotation standards, levels of detail, and quality. Annotations might be bounding boxes, or pixel-wise segmentations.

   Approach:

   *** Annotation Standardization:** Developing a unified annotation framework or schema to remove the differences in annotation formats and details. For example, converting all annotations to pixel-wise masks if segmentation is the goal *(which is in this project)*.

3. **Class Imbalance Across Datasets**

   Challenge: When we combining datasets, some classes or conditions may be more in some datasets and less in others, leading to class imbalance. This can make the model a biased one.

   Approach:

   *** Balanced Sampling:** Use balanced sampling techniques to ensure that each class is equally represented during training. For Example, we can oversample minority classes or undersample majority classes.

   *** Data Augmentation:** Specifically augmenting data from underrepresented classes to increase their size in the training set.

1.6. Plan

Starting with the dataset integration, I am expecting to mainly have problems mentioned in the previous section namely, variability in images, annotation inconsistencies, imbalanced datasets and probably more. I plan to deal with these by using the above mentioned techniques and also try to find better techniques in tackling those problems.

In case of the Segmentation model to be utilized, I will have to look more into the models between UPerNet, UNet/UNet++, Swin (Segmentation).

My training stategy for now will be to start simple, by trying to integrate two datasets and training the model and if successful, then move forward with integrating the other datasets.

For dealing with different data modalities, The most promising approech would be to use a shared embedding space to represent the data from different modalities on the same space and then use contrastive learning to train the model to find similar and dissimilar points.

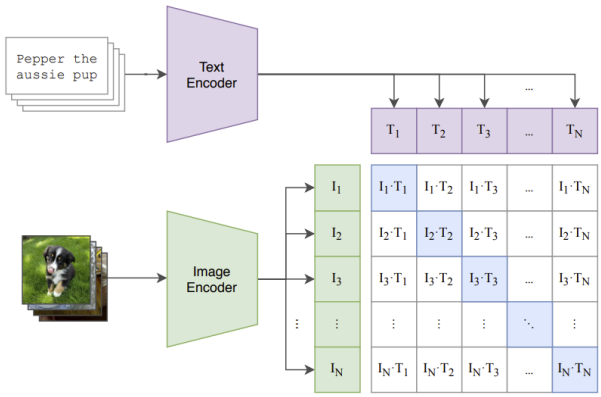


Figure 4. Example of how Contrastive Learning Framework can be used on the shared embedding space

2. Pre-Warmup Introduction

This report presents the outcomes of the exercises 1 to 9 and with their respective methodologies used. The datasets for the exercises are downloaded from Japanese Society of Radiological Technology [2] in a *.zip* format and unpacked during runtime using the command *!unzip filename*. Aditionally, for experiments 2-4 and 7, a pretrained *ResNet18* model, and for experiments 5-6 a *U-Net model* has been employed. In case of Exercises 8 and 9 unsupervised learning techniques were used for Anomaly detection and clustering.

2.1. Exercise 1

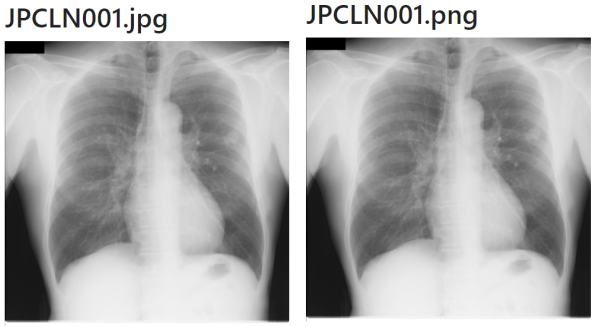
For the first exercise, I have unzipped two files namely *Practice_PNGandJPG.zip* & *Practice_DICOM.zip*. The former comprises images of the format *png* and *jpg* while the latter comprises images in the *dcm* format.

2.1.1 png, jpg

This task was addressed by using the Pillow library [1]. Reading images in both jpg and png formats was accomplished using the command *Image.open("filepath+filename")*, while the writing of images utilized Pillow's *.save()* method.

2.1.2 dcm

For the dicom format, the process involved importing the pydicom library [3]. Reading images was achieved through *pydicom.dcmread()* method, and for writing images, the *.save_as()* method from the same library was utilized.



(a) .jpg image (b) .png image

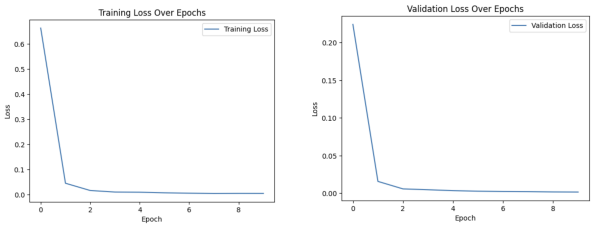
Figure 5. Visualization of Images

2.2. Exercise 2

The exercise involved the task of classifying orientations in the Up, Down, Left, or Right directions. The name of the dataset used is *Directions01.zip* which contained 2 folders namely train and test. A pretrained ResNet model was used to classify.

2.2.1 Observations & Results

The dataset was resized to align with the dimensions of the resnet model model initally. The training process utilized the Binary Cross Entropy Loss function was executed for a total of 10 epochs.



(a) Training Loss (b) Validation Loss

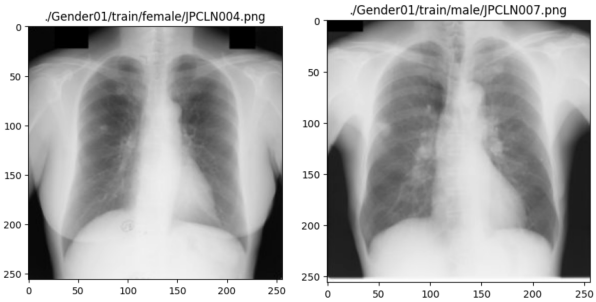
Figure 6. Visualization of Losses

As depicted in Figure 1, the losses exhibit a rapid decrease after the initial epoch, eventually reaching an error of zero. This was achieved with an 80:20 ratio split of training and validation datasets. A comparable trend persisted when experimenting with a 50:50 training and validation split, with convergence occurring one loop later.

The accuracy on the tested data was consistently 100% in both cases.

2.3. Exercise 3

This exercise involved the task of classifying genders (*male/female*). This dataset was also from [2] and the name of the file is Gender01 availble to download in a *.zip* format. In total (*train+test*) there were nearly 250 images.



(a) Female chest X-ray (b) Male chest X-ray

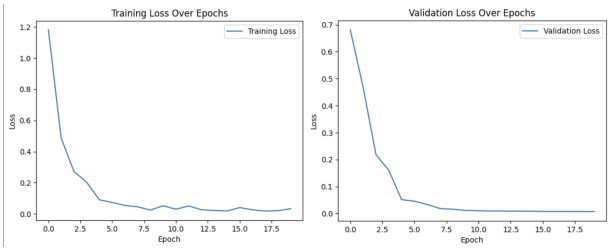
Figure 7. Visualization of chest X-rays from train data

Through experimentations, the final parameters I have chosen to get my best result are:
1. learning rate: 0.001 with a learning rate scheduler of gamma value=0.1 every 5 steps.
2. loss function: Binary Cross Entropy Loss
3. optimizer: Adam
4. epochs: 20

5. batch size: 16
6. dropout rate: 0.35

The training and validation split was 70:30.

2.3.1 Observations & Results



(a) Training Loss (b) Validation Loss

Figure 8. Visualization of Losses

The best results achieved on test data were: ACC: 91.4% and AUC: 91.32

2.4. Exercise 4

This exercise involved the task of predicting age between 16-89. This dataset was also from [2] and the name of the file is XPAge01_RGB available to download in a *.zip* format. In total (*train+test*) there were nearly 250 images.

The parameters chosen are:
1. learning rate: 0.001 with a learning rate scheduler of gamma value=0.1 every 5 steps.
2. loss function: L1 Loss
3. optimizer: Adam
4. epochs: 40
5. batch size: 32
6. dropout rate: 0.2

The training and validation split was 80:20. I chose L1 loss function as I am performing regression to predict the value of age instead of classification.

2.4.1 Observations & Results

The *figure 5* represents the evaluation on the test set (*after each epoch* using the *eval* function in the code and the errors are collected in a list which is plotted. The figure represents the relationship between the update iteration and the test error.

The best MAE achieved was 7.9.

2.5. Exercise 5

This exercise involved the task of performing semantic lung segmentation. This dataset was from [2] and the name of the file is *Segmentation01_RGB* available to download in

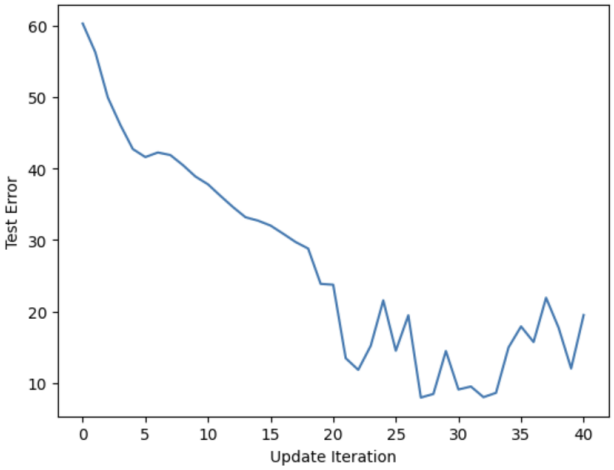
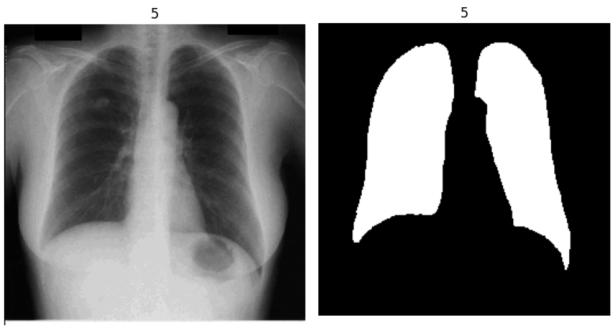


Figure 9. Training Loss

a *.zip* format. In total (*train+test*) there were 60 images.

The model used was a pre-trained U-Net.



(a) Sample Image from Segmentation01_RGB dataset (b) Target Image of (a) from Segmentation01 dataset

Figure 10. Visualization of Sample data

Figure 6 shows a sample image from the *Segmentation01_RGB* dataset along with the image's format, mode, and original size respectively. I have chosen the RGB version because the pretrained model required an input image which has 3 channels and the *Segmentation01* dataset had only 1 channel.

The parameters chosen are:
1. learning rate: *0.0001*.
2. loss function: *BCEWithLogitsLoss()* (chose this over BCE as the documentation said the former provides better numerical stability).
3. optimizer: *Adam*
4. epochs: 150
5. batch size: 32

2.5.1 Observations & Results

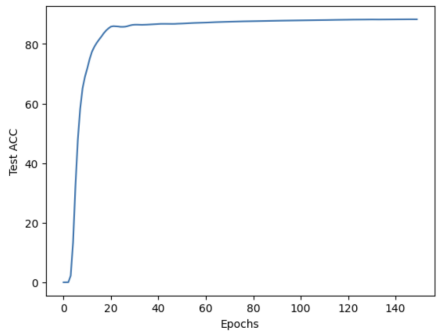


Figure 11. Test IOU of Exercise 5 over 150 epochs

The figure 7 represents the test accuracy over epochs (*after each epoch*) using the *eval* function in the code and the accuracies are collected in a list which is plotted. The figure represents the relationship between the update iteration and the test IOU.

The best test IOU score achieved was 86.27%.

Figure 12. Prediction for Sample 5 (Exercise 5) from figure 6.

2.6. Exercise 6

This exercise involved the task of performing Instance organ segmentation involving heart, right lung,left lung, chest area, and outside the body. This dataset was from [2] and the name of the file is *Segmentation02* available to download in a *.zip* format. In total (*train+test*) there were 247 images. The composition of the dataset is 85 images for the heart region, and 170 for the lung fields. The images are of the size *256x256*.

The model used here was a pre-trained U-Net but the number of classes are changed to 5.

(a) Sample Image from Segmentation02 dataset (b) Target Image of (a) from Segmentation02 dataset

Figure 13. Visualization of Sample data

Figure 9 shows a sample image from the *Segmentation02* dataset.

The parameters chosen are: *(same as for exercise 5*
1. learning rate: *0.0001*.
2. loss function: *CrossEntropyLoss*
3. optimizer: *Adam*
4. epochs: *150*
5. batch size: *32*

2.6.1 Observations & Results

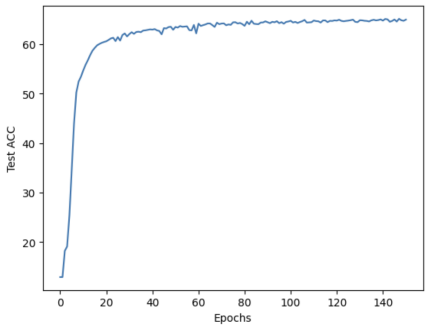


Figure 14. Test IOU of Exercise 6 over 150 epochs

The figure 10 represents the test IOU over epochs (*after each epoch*) using the *eval* function in the code and the accuracies are collected in a list which is plotted. The figure represents the relationship between the update iteration and the test accuracy.

The best test IOU achieved was 65.12%.

2.7. Exercise 7

This exercise involved the task of performing localization of organs namely heart, right lung, and left lung.

Figure 15. Prediction for Sample 5 (Exercise 6) from figure 10.

This dataset was from [2] and the name of the file is *Segmentation0* available to download in a *.zip* format. In total (train+test) there were 247 images. The composition of the dataset is 85 images for the heart region, and 170 for the lung fields. The images are of the size 256x256.

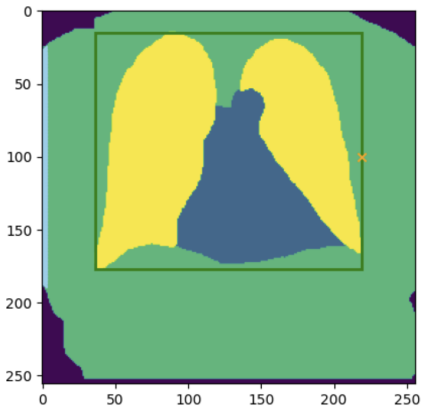


Figure 16. Sample image with bounding box

Disclaimer: This part of the assignment was done with the help of my teammates Sonny and Rishit

A resnet18 model was used with MSE Loss and Adam optimizer.

The parameters chosen are:
1. learning rate: 0.0001
2. loss function: MSE Loss
3. optimizer: Adam
4. epochs: 80
5. batch size: 8

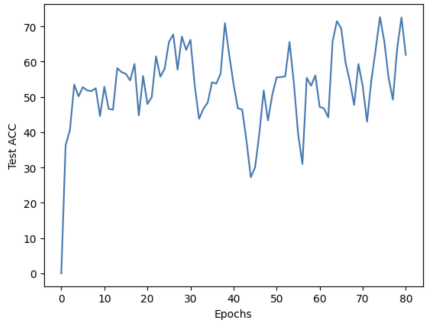


Figure 17. Test IOU of Exercise 7 over 80 epochs

2.7.1 Observations & Results

The figure 12 represents the test IOU over epochs (*after each epoch*) using the *eval* function in the code and the accuracies are collected in a list which is plotted. The figure represents the relationship between the update iteration and the test accuracy.

The best test IOU achieved was 72.65%.

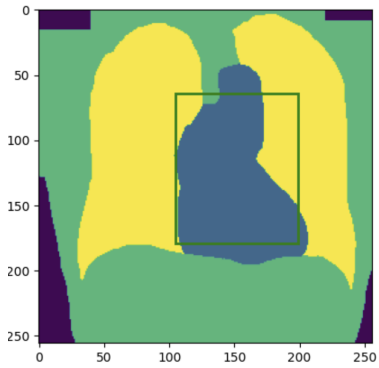


Figure 18. Prediction for Sample 5 (Exercise 7).

2.8. Exercise 8

This exercise involved the task of performing anomaly detection where the anomaly in this dataset would be a flipped chest x-ray image. This dataset was from [2] and the name of the file is *autoencoder_img* available to download in a *.zip* format. In total (normal+abnormal) there were 264 images. The composition of the dataset is 200+30 images for the normal images, and 34 for the flipped images (abnormal). The images are of the size 256x256.

By using Gaussian Mixture Models (GMM) within an unsupervised learning framework, we can seperate normal from abnormal chest X-rays.

A Gaussian Mixture Model is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters. GMMs are characterized by two key components:

1. Mixing Coefficient: The probability that a given data point belongs to the *k*th cluster.

2. Gaussian Components: Each component *k* in the mixture model has its own mean and covariance.

The probability density function of a GMM for a data point *x* is given by:

$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)$

2.9. Model Training

1. Initialization: The number of components K is set to 6, based on preliminary analysis suggesting this number effectively captures the diversity in the dataset. The GMM parameters are initialized.

2. Expectation-Maximization (EM) Algorithm: The EM algorithm is applied to iteratively update the parameters of the GMM to maximize the likelihood of the data given the model. The EM algorithm alternates between two steps:

Expectation (E-step): Calculate the responsibility that component k has for datapoint n:

$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(x_n|\mu_k,\Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n|\mu_j,\Sigma_j)}$

Maximization (M-step): Re-estimate the parameters using the current responsibilities:

* $\mu_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n$

* $\Sigma_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(x_n - \mu_k^{new})(x_n - \mu_k^{new})^T$

* $\pi_k^{new} = \frac{N_k}{N}$

2.9.1 Methodology and Results

The dataset already had the autoencoder file in a *.h5* format which has 128 embeddings in it. By running the normal images into the autoencoder, I took the mean and standard deviation of the same pixels in every image, stacked them and sent this distribution into the Gaussian Mixture model with components as 6.

I did the same for abnormal images as well and took the maximum score from the abnormal images and the minimum score from the normal images which would give me the epsilon value in the above formula or essentially the threshold value i.e., the threshold value is between these two.

The model was able to accurately identify 100% of normal images and 100% of the abnormal images.

2.10. Exercise 9

This exercise involved the task of clustering the images which are oriented left, right, up, down or flip. This exercise required a combination of two datasets from [2] and the name of the files are *autoencoder_img* and *Directions01* available to download in a *.zip* format. The *Directions01* file contained the images oriented in left, right, up, down and the flipped images from the dataset defined in the previous exercise.

2.10.1 Methodology

The methodology is the same as Exercise 8 for this exercise.

I have used the same autoencoder as the above exercise with the same number of components (6) in the Gaussian Mixture model and it worked.

2.10.2 Results

The result was 95.62% accuracy. Meaning that in 95% of the cases the datapoints(images) were assigned to the correct clusters.

I. CheXpert dataset + ConvNeXt model

The CheXpert dataset contains over 224,000 images of size varying from *1800 x 1800* to over *4000 x 4000*. The dataset requires you to signup first and then be able to download it. The dataset can be found at in a *.zip* format at [**?**], and is nearly 500 GB in size. The dataset also required *azcopy* to be installed to download it from the Azure platform on the */scratch* location on SOL.

The ConvNeXt model is a pure ConvNet model. It was built using the training techniques and some of the architectural designs of the SOTA hierarchical transformers. The original code for the model can be found here [**?**].

3. Introduction to Classification

Training ConvNeXt on CheXpert

One of the first things that I observed in my process of integrating the model and dataset was that the CheXpert dataset had images of varying sizes with no particualar pattern to group them. Another observation was that the ConNeXt model was built with images of size around *300 x 300* in mind and when this was replaced with images approximately the size of *2000 x 2000*, there was constant crashes on the local computer, either using RAM or through CUDA, with a message saying *CUDA out of memory* or just the application crashing when using RAM.

I later experimented with various batch sizes running on the local computer starting from a batch size of 16 down to a batch size of 2, which gave the same error. The model finally ran when the batch size was 1 on the RAM(16 GB) but still gave the same error when run on GPU(8GB).

I decided to use the batch size of 1 and chose not to resize or rescale the images because of the possibility of information loss. When this model was run on the a30 GPU, it gave the same error as on the local machine about not enough CUDA memory and crashed. The model worked fine when run on the a100 GPU. It has been 12 hours since and the model is still running and progress is about 8,000 images of the 224,000 images.

3.1. Fine-tune ConvNeXt with Pretrained weights

I used the pre-trained weights from the original github of the authors of the ConvNeXt paper and scheduled it on SOL on the a100 GPU but no resources have been allotted yet.

3.2. Comparision

Both of the models are currently running on SOL and do not have results to compare them and measure the performances.

3.3. Performance measures from Classmates

	Swin	InternImage	ConvNeXt
ChestX-ray14	0.8287	-	0.8141
ChestXpert	-	-	-
MIMIC	-	-	-

Table 1. Performances Achieved (AUC)

Rishit Ratna: Gave performance results on using *Swin-B* model on the *ChestX-ray14* dataset.

Tarun: Contributed the performance on the *ConvNeXt* model performed on the *NIH-ChestX-ray14* dataset.

The rest of the students assigned to the model and datset combinations in our group are still running it.

3.4. Competitions

Currently, there are no running contests for this dataset and when the model is done running, but I could compare it with the top solutions/approaches of the top performer of the past competition once the model is done training.

3.5. Idea for ConvNeXt + 3 datasets

Disclaimer: These are not specific methods or ways that provide a solution to the problem but instead address or atleast attempt to address the underlying problem and provide possible directions to overcome them.

1. Addressing the problem of Data Integration starting with standardizing Image sizes and having a unified data format helps achieve consistency in training the model.

2. Tackling the problem of data distributions is essential to make sure the imblances does not affect the model being biased.

3. MIMIC poses an additional challenge of providing information such as textual reports and finding ways to incorporate them into the model is also crucial.

4. Introduction to Classification + Localization

TBX 11K dataset + Mask R-CNN model

The TBX 11K dataset [**?**] contains around 11,200 images which is 17 times larger than the existing dataset. There are Four categories - healthy, active TB, latent TB, unhealthy but no TB. It was tested using the golden standard (diagnostic microbiology). It contains images of size 3000 x 3000 resolution.

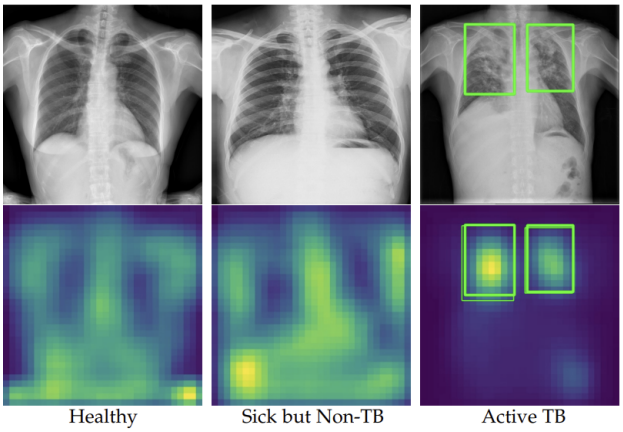


Figure 19. Sample of dataset from Original paper part 1

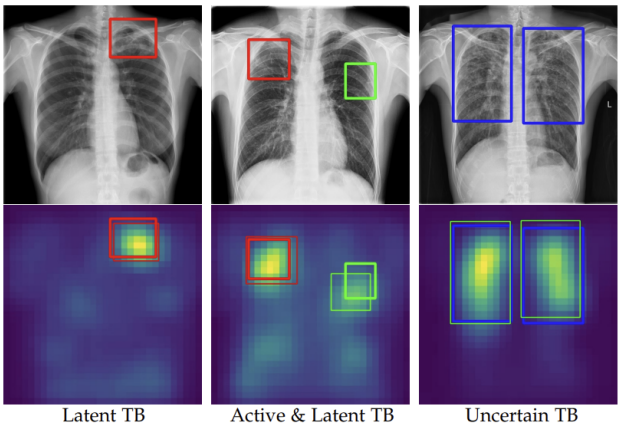


Figure 20. Sample of dataset from Original paper part 2

Mask R-CNN (Region-based Convolutional Neural Network) is a popular deep learning model architecture used for object instance segmentation. It is an extension of the Faster R-CNN framework, designed to handle both object detection and pixel-wise segmentation tasks simultaneously. The model was introduced by Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick in their 2017 paper titled "Mask R-CNN." [**?**]

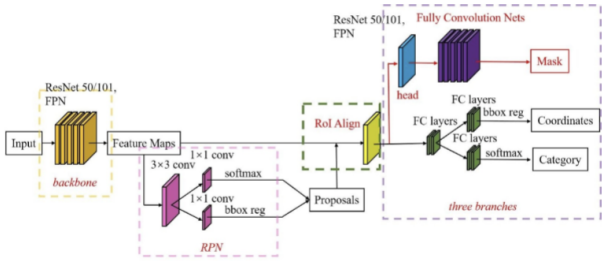


Figure 21. Mask R-CNN architecture

The main advantage of Mask R-CNN model is that it uses ROI Align instead of pooling meaning stride is not quantized.

4.1. Training Mask R-CNN on TBX 11K

The dataset was fairly small compared to others. It was nearly 3GB. The images on Kaggle are of the size 512x512 which is what I used. And in this dataset, there are actually five categories, i.e., Healthy, Sick but Non-TB, Active TB, Latent TB, and Uncertain TB.

The the image lists of the training, validation, and training+validation sets, namely 'TBX11K_train.txt', 'TBX11K_val.txt', and 'TBX11K_trainval.txt', respectively.

I have experimented with various batch sizes ranging from 8, 16, 32 and 64 running on the local computer, and found that the batch size of 16 was optimal.

I decided to use the batch size of 16 and chose to resize the images because of the possibility of information loss. This model has been tuned to accommodate the new dataset and is scheduled to run on SOL and has not been assigned a GPU yet.

4.2. Fine-tune Mask R-CNN with Pre-trained weights

I used the pre-trained weights from pytorch which has a ResNet50 architecture as its backbone and this model is from the original Mask R-CNN paper.

4.3. Comparision

Both models are currently operational on SOL, and there are no available results for performance comparison.

II. Performance measures from Classmates

	DINO	Mask R-CNN	Swin
NODE21	-	-	-
TBX11K	-	-	-
VinBigData CXR	-	-	-

Table 2. Performances Achieved (AUC)

All the students assigned to the model and dataset combinations in our group are still running it and yet to get the results.

4.4. Competitions

Presently, there are no ongoing competitions for this dataset. Upon completion of the model training, I plan to compare its performance with the top solutions and approaches employed by the leading participant in previous competitions.

4.5. Idea for Mask R-CNN + 3 datasets

Note: These suggestions do not present explicit methods or solutions to the problem; rather, they aim to acknowledge and, to some extent, tackle the underlying issue by offering potential directions for overcoming challenges.

1. Addressing the problem of Data Integration starting with standardizing Image sizes, preprocessing and having a unified data format helps achieve consistency and ensures compatibility in training the model.
2. Tackling the problem of data distributions is essential to make sure the imbalances does not affect the model being biased.
3. Multi-label Learning is important as this would then enable the model to classify, localize and also produce a bounding box on all three types of datasets.
4. Annotation Consistency also helps in accurate model training.

5. Introduction to Classification + Localization + Segmentation

5.1. Dataset

Pneumothorax, also recognized as a collapsed lung is a condition that involves the unusual accumulation of air in the pleural space, which is the area between the lung's outer surface (visceral pleura) and the inner chest wall (parietal pleura). It's a relatively common respiratory disorder that affects a diverse group of individuals across different settings.

Identifying a pneumothorax on a chest X-ray is generally straightforward for skilled physicians or radiologists.

Usually, radiologists diagnose the condition using chest radiography, but confirmation can be difficult in some cases. Using a deep learning algorithm that accurately detects pneumothorax can benefit the physicians.

The SIIM-ACR Pneumothorax Segmentation dataset [**?**] contains over 12,000 training images and nearly 3,500 testing images. The dataset has two CSV files: one for training and another for testing. The CSV file for training contains the IDs of the images (X-rays) along with their corresponding RLE (Run-Length Encoding) masks, whereas the CSV file for testing contains the IDs of the images (X-rays).

The dataset images of DICOM format paired with annotations that has image IDs and RLE masks. In these annotations, the presence of pneumothorax (also known as a collapsed lung) are marked with encoded binary masks. There are some images in the training set that are also annotated with multiple masks.

In cases where images do not display signs of pneumothorax, the mask value is set to -1, which indicates the absence of a mask.

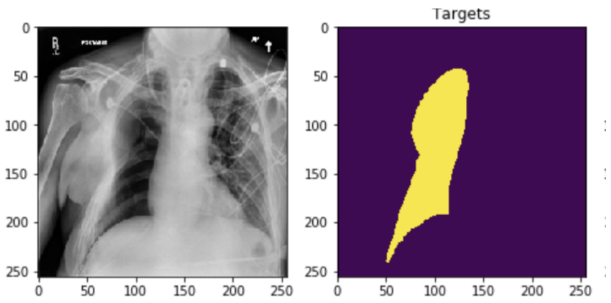


Figure 22. Sample of dataset from competition website part 1

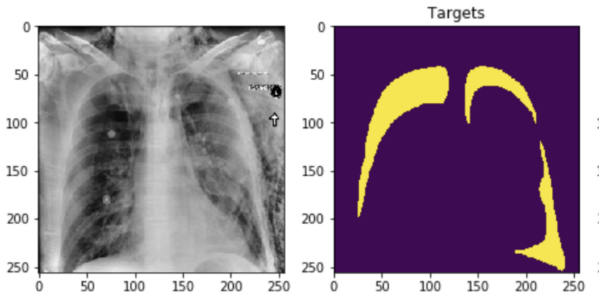


Figure 23. Sample of dataset from competition website part 2

5.2. Pneumothorax dataset + U-Net Model

The U-Net model is a convolutional neural network (CNN) that was developed primarily for biomedical image segmentation tasks. [**?**]

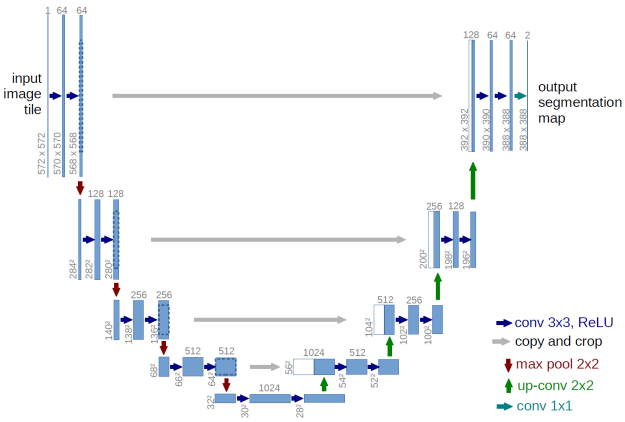


Figure 24. U-Net architecture

5.3. Training U-Net on Pneumothorax Dataset

The dataset was fairly small compared to others. It was nearly 0.5GB. The images on Kaggle are of the size 1024x1024 which I what I used.

I've conducted trials using different batch sizes—8, 16, 32, and 64—on my local computer. The batch size of 16 provided the best results.

I proceeded to use the batch size of 16 and chose to not resize the images because of the possibility of information loss and also the masks will have to be resized to the same size if I resize the input images. The model was modified to make a pipeline that produced results on the local computer and has been scheduled to run on SOL but does not have a GPU assigned to it yet.

5.4. Fine-tune U-Net with Pretrained weights

I used the pre-trained weights from pytorch and used the command *torch.load('../input/unetwithpretrainedresnet50-encoderpytorch/best_model.pth')* which has a ResNet50 encoder as its backbone.

5.5. Comparision

Both models are currently operational on SOL, and there are no available results for performance comparison.

5.6. Performance measures from Classmates

All the students assigned to the model and dataset combinations in our group are still running it and are yet to get the results.

	U-Net/UNet++	UPerNet	Swin
ChestX-Det	-	-	-
Pneumothorax	-	-	-
CheXmask	-	-	-

Table 3. Performances Achieved (AUC)

5.7. Competitions

Presently, there are no ongoing competitions for this dataset. Upon completion of the model training, I plan to compare its performance with the top solutions and approaches employed by the leading participant in previous competitions available on Kaggle and the competition's original website.

5.8. Idea for U-Net + 3 datasets

Note: These suggestions do not present explicit methods or solutions to the problem; rather, they aim to acknowledge and, to some extent, tackle the underlying issue by offering potential directions for overcoming challenges.

1. Addressing the problem of Data Integration starting with standardizing Image sizes, preprocessing and having a unified data format helps achieve consistency and ensures compatibility in training the model.

2. Tackling the problem of data distributions is essential to make sure the imblances does not affect the model being biased.

3. Multilabel Learning is important as this would then enable the model to classify, localize and also produce a bounding box on all three types of datasets.

4. Annotation Consistency also helps in accurate model training.

6. Conclusion

The projects taken in pretraining, classification, segmentation, and localization tasks have significantly helped me learn. The foundational pretraining tasks have substantially augmented the learning efficacy of models, facilitating more precise outcomes in both classification and localization tasks. Addressing challenges through innovative approaches such as enhanced transfer learning and augmented data techniques have been proposed to tackle them in some areas. The work completed lays a groundwork for further research, aiming to leverage deep learning advancements to enhace Medical Imaging.

References

[1] J.A.C. (Alex) and contributors. Available at: https://pillow.readthedocs.io/en/stable/reference/image.html (accessed: 20 january 2024). *Image module, Pillow (PIL Fork)*, 2010. 4

[2] G. Coppini, S. Diciotti, M. Falchini, N. Villari, and G. Valli. Neural networks for computer-aided diagnosis: detection of lung nodules in chest radiograms. *IEEE Transactions on Information Technology in Biomedicine*, 7(4):344–357, 2003. 4, 5, 6, 7, 8

[3] Darcy Mason and pydicom contributors. Available at: https://pydicom.github.io/pydicom/stable/auto_examples/index.html (accessed: 20 january 2024). *General examples*, 2008. 4