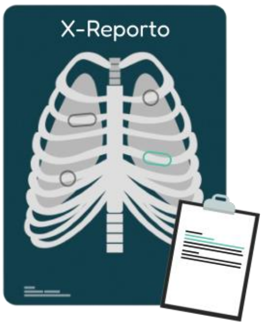
Cairo University
Faculty of Engineering
Department of Computer Engineering

X-Reporto



A Graduation Project Report Submitted

to

Faculty of Engineering, Cairo University

in Partial Fulfillment of the requirements of the degree

of

Bachelor of Science in Computer Engineering.

Presented by

Zeinab Moawad Fayez

Supervised by

Dr. Yahia Zakaria

July 2024

Abstract

This project addresses the increasing burden on radiologists due to the rising prevalence of chest diseases, which results in long queues of X-ray reports needing diagnosis. The objective of the project is to develop a semi-automated reporting system that supports radiologists by generating preliminary reports for each anatomical region and identifying diseases in those regions. Our approach involves enhancing chest X-ray images to correct defects caused by the X-ray devices, ensuring that critical diagnostic information is preserved and easily identifiable. The tool generates comprehensive, template-based reports tailored to the specific diseases identified, thereby streamlining the reporting process and reducing the time required for diagnosis.

The primary outputs of the project include the enhanced X-ray images and the detailed, disease-specific reports generated by the tool. Development and testing of the tool were conducted to ensure accuracy and reliability. Testing results demonstrate significant improvements in image clarity and diagnostic accuracy, as well as a reduction in the time required for report generation.

This project successfully developed and implemented the tool. The outcomes of this project not only alleviate the workload of radiologists but also contribute to better patient care by enabling faster and more precise medical interventions, ultimately increasing patient survival rates.

This project is sponsored by Voyance Health [1], a software development company in the medical field, which has provided invaluable support by offering server resources for training models on our large dataset. They also contributed the basic idea of the project and have been closely following our progress, offering guidance and feedback throughout the development process.

1

ACKNOWLEDGMENT

We extend our sincerest gratitude and appreciation to our esteemed supervisor, Dr. Yehia Zakaria, for his dedicated supervision, invaluable guidance, and unwavering support throughout our project. His mentorship has been instrumental in our journey.

We also wish to thank Voyance Health for their sponsorship and provision of servers, which were essential for training our models.

Special thanks go to Eng. Mohamed Shawky and Eng. Omar Samir, technical assistants at the Faculty of Engineering, Cairo University, for their continuous support and assistance.

Additionally, we express our heartfelt appreciation to our department for providing us with the necessary courses and resources to undertake this project. Their support and encouragement have been pivotal.

Thank you all for making this journey possible and for your unwavering belief in our capabilities.

Table of Contents

Contents

List of Figures

List of Tables

Contacts

Team Members

Name	Email	Phone Number
Zeinab Moawad	zeinab.hassan00@eng-st.cu.edu.eg	+2 01111125861

Supervisor

Name	Email	Number
Dr. Yahia Zakria	Yahiazakaria13@gmail.com	+2 0١11 729 7303

Chapter 1: Introduction

X-Reporto is a specialized tool designed to aid radiologists by managing an increasing patient load and optimizing the medical reporting process for chest x-ray images. It enhances images affected by device defects for clearer and more accurate diagnostics. X-Reporto automates detailed report generation using templates, streamlining the workflow. By analyzing images, it identifies potential diseases and their locations within the chest, saving time for radiologists and improving diagnostic accuracy, patient care, and treatment outcomes.

1.1. Project Outcomes

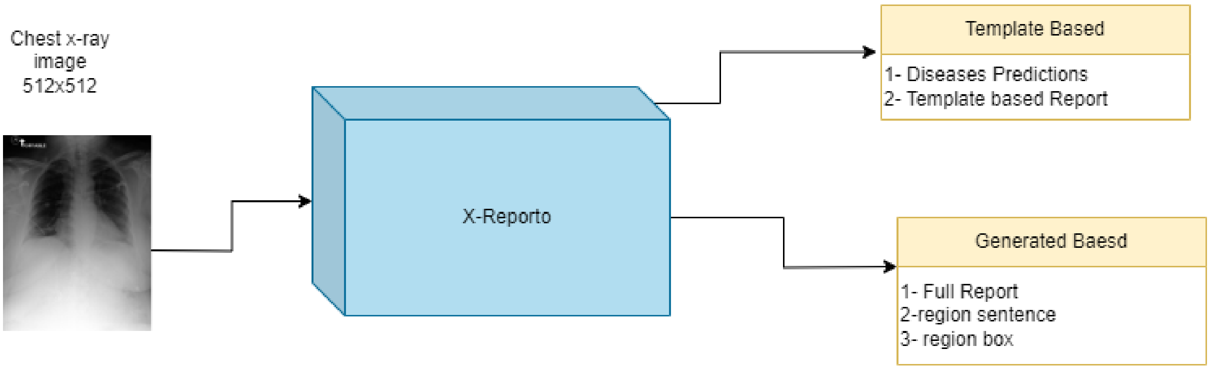


Figure ١ X-Reporto Outcomes

The outcomes of our project focus on developing a tool that significantly enhances the efficiency and accuracy of chest X-ray diagnostics. This tool generates comprehensive reports on chest X-ray images, detailing findings for each anatomical region and identifying specific diseases. It improves image quality by correcting defects from the X-ray device, ensuring clear and accurate diagnostic information. Additionally, it provides radiologists with template-based reports tailored to the identified diseases, streamlining the reporting process and reducing diagnosis time. These enhancements help radiologists manage their workload more effectively and contribute to better patient care by enabling faster and more precise medical interventions.

Chapter 2: Literature Survey

2.1. Double Convolution

Convolutional Operation with Filters: The convolution operation involves sliding the filter over the input data and computing the element-wise product between the filter and the overlapping region of the input. The sum of these products forms a single element in the output feature map. The filter is then moved to the next position, and the process is repeated until the entire input is traversed.

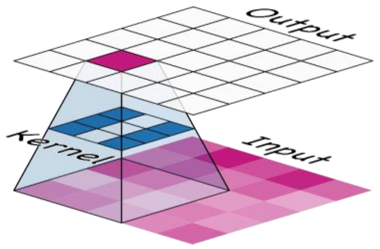


Figure 2 Convolution

After filtering, the feature maps pass through the activation function. The function introduces non-linearity to the network, allowing it to learn complex patterns and make better predictions. The rectifier function has a graph like this:

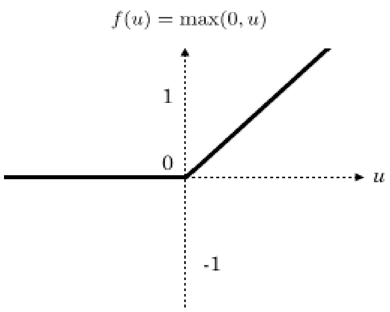


Figure 3 Relu Function

2.2. Up Sampling Bilinear:

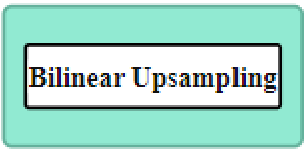


Figure 4 Up Sampling Bilinear Conv Arch

A more sophisticated approach that uses linear interpolation to compute the values of the new pixels, resulting in smoother transitions than nearest neighbor up sampling.

2.3. MAX Pooling:

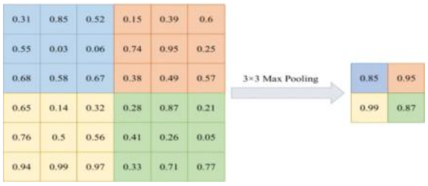


Figure 5 Max Pooling

Max pooling is a commonly used feature extraction operation, typically applied in convolutional neural networks. It reduces the spatial dimensions of features by selecting the maximum value within each small window or region.

2.4. CNN:

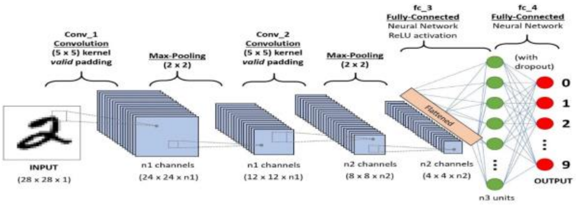


Figure 6 Conv Neural Networks

O'shea et all [1] Proposed the CNN network which is a network architecture for deep learning which learns directly from data. CNNs are particularly useful for finding patterns in images to recognize objects. They can also be quite effective for classifying non-image data such as audio, time series, and signal data.
Layers used to build CNN: Convolutional neural networks are distinguished from other neural networks by their superior performance with image, speech, or audio signal inputs.

They have three main types of layers, which are:

i. Convolutional Layer
ii. Pooling Layer
iii. Fully Connected (FC) Layer

2.5. VGG-19:

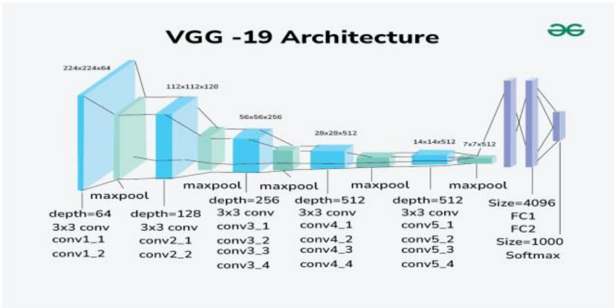


Figure 7 VGG Network Architecture

Simonyan, Karen et al [2] Proposed the basic idea of the VGG16 model, with the exception that it supports 19 layers. The numbers “16” and “19” refer to the model’s weight layers (convolutional layers). In comparison to VGG16, VGG19 contains three extra convolutional layers. In the final section of this essay, we’ll go into greater detail on the features of the VGG16 and VGG19 networks.

Very tiny convolutional filters are used in the construction of the VGG network. Thirteen convolutional layers and three fully connected layers make up the VGG-16.

2.6. U-Net:

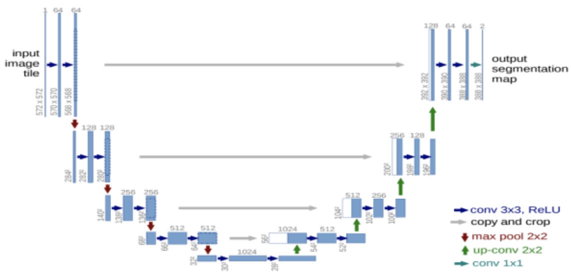


Figure 8 U-Net Architecture

The U-Net architecture, proposed by Ronneberger et al., is designed for semantic segmentation and comprises a contracting path and an expansive path. The contracting path includes repeated applications of two 3x3 convolutions, followed by a ReLU and a 2x2 max pooling operation for downsampling, with feature channels doubling at each step. The expansive path involves upsampling the feature map, applying a 2x2 "up-convolution" to halve the feature channels, concatenating with the cropped feature map from the contracting path, and performing two 3x3 convolutions followed by a ReLU. Cropping is necessary due to border pixel loss in convolutions. The final layer uses a 1x1 convolution to map each 64-component feature vector to the desired number of classes. The network has a total of 23 convolutional layers.

2.7. GANs:

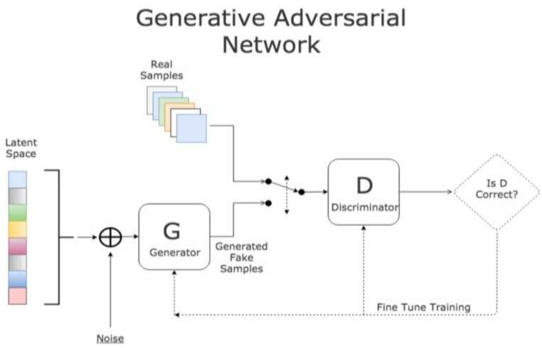


Figure 9 Generative Adversarial Network

The GANs proposed by [3] Ahmadi et al, represent a cutting-edge approach to generative modeling within deep learning, often leveraging architectures like convolutional neural networks. The goal of generative modeling is to autonomously identify patterns in input data, enabling the model to produce new examples that feasibly resemble the original dataset. Generative Adversarial Network (GAN) consists of two neural networks, namely the Generator and the Discriminator, which are trained simultaneously through adversarial training.

1. **Generator:** This network takes random noise as input and produces data (like images). Its goal is to generate data that's as close as possible to real data.
2. **Discriminator:** This network takes real data and the data generated by the Generator as input and attempts to distinguish between the two. It outputs the probability that the given data is real.

2.8. HOG (Histogram of Oriented Gradients)

HOG is a feature descriptor that counts occurrences of gradient orientation in localized portions of an image. It divides the image into small regions called cells, computes a histogram of gradient directions within each cell, and then normalizes local contrast in overlapping blocks.
The main steps for computing HOG features from an image are:

1. **Compute the gradient magnitude and angle.**
2. **Calculate Histogram of Gradient for cell.**
3. **Normalize the histogram for Blocks.**

2.9. Haar

Haar features are extracted from rectangular areas in an image. The feature’s value is based on the pixel intensities. Usually, it is calculated using a sliding window, and the area within the window is partitioned into two or more rectangular areas.
the key strength of Haar features lies in their ability to represent three patterns:

1. Edges: Either vertical or horizontal due to how we oriented the rectangular area. They are useful for identifying boundaries between different image regions.
2. Lines: The diagonal edges in an image. They are useful for identifying lines and contours in objects.

Center-surrounded features: This detects the changes in intensity between the center of a rectangular region and its surrounding area. This is useful to identify objects with a distinct shape or pattern. They are similar to convolution kernels taught in the Convolution Neural Networks course. We will apply these haar.

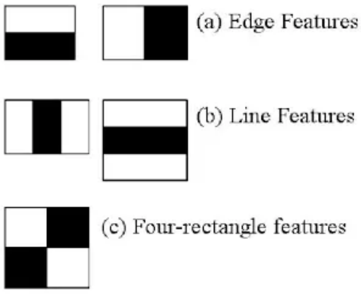


Figure 10 Feature Extraction Filters

2.10. Implemented Approach

2.10.1. Denoiser Approach

GANs have been applied successfully in medical imaging [5, 7] but have not previously been used with high-resolution imaging techniques. The challenge is that the high-resolution images produced include finely detailed features with high-frequency content.
Our GAN-based method adapts the U-Net network architecture to meet the specialized requirements of improving the quality of images generated.
We demonstrate that they can be trained with limited data, perform well with high-resolution datasets, and generate greatly improved images.

Chapter 4: Datasets Literature Survey

In this chapter, we describe our data preprocessing steps, including data cleaning and preparation, to facilitate effective learning from the dataset.

4.1. Data Preprocessing for Effective Learning

We followed the steps below for preparing our dataset for the training process.

1. We need to create a file for reading data for each module so we upload all the dataset in a subset in colab which we can create a file of data information on the server easily for training, evaluation and testing.

2. Apply statistics on data to set parameters in Modules and the result

Mean	0.474
STD	0.301
Range Bounding box numbers per image	24-29 Boxes
Region Classifiers positive weights	2.24
Abnormal Classifiers positive weights	6
Maximum Sentence Length	264
Average Number of sentences	16

Table 1 MIMC CXR Data Statistics

3. Apply Data preprocessing:
   1- preprocessing image: resize longest dimension to 512 and pad image, all operations consider bounding box transformation.
   2- preprocessing sentences: by tokens all box phrases, also padding attention mask, label ids, input ids to be maximum sequence length.

Chapter 5: System Design and Architecture

5.1. System Architecture

The architecture of the system is structured around a modular design that enhances scalability and maintainability. The overall system is represented as a block diagram that shows interactions between the different modules and their respective functions.

As we have 6 separated modules, each module has specific functionality and expects certain modules output and generates correct output for next modules. Follow of our modules is that

1. We first should remove any noise in the x-ray chest.
2. We detect anatomical regions in chest and generate visual features for them
3. We Select some regions that have findings in it.
4. From these visual features we generate corresponding findings in it including abnormality and diagnosis.
5. Finally, we generate predictions for possible diseases and its location in x-ray.

5.1.1. Block Diagram

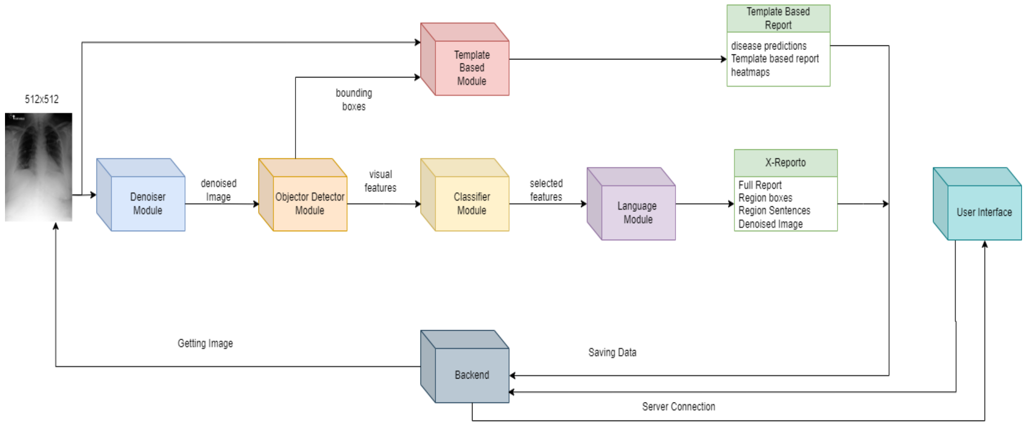


Figure 11 X-Reporto System Block Diagram

5.2. Denoiser Deep Learning Approach

5.2.1. Functional Description

X-ray images of medical information must be preserved for getting a report on it correctly .X-ray devices may have defects which affect medical information. Denoiser takes an image to enhance it and remove noise without effect on medical information of the image.

5.2.2. Modular Decomposition

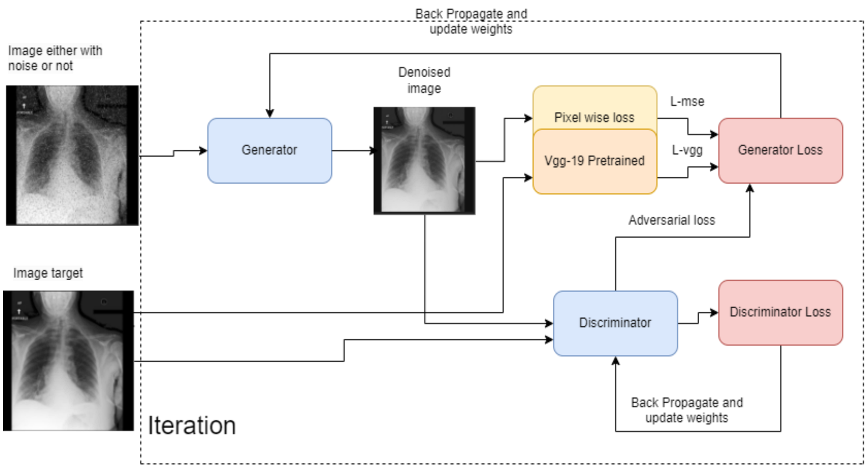


Figure 12 Denoiser Architecture

5.2.2.1 Generator Architecture

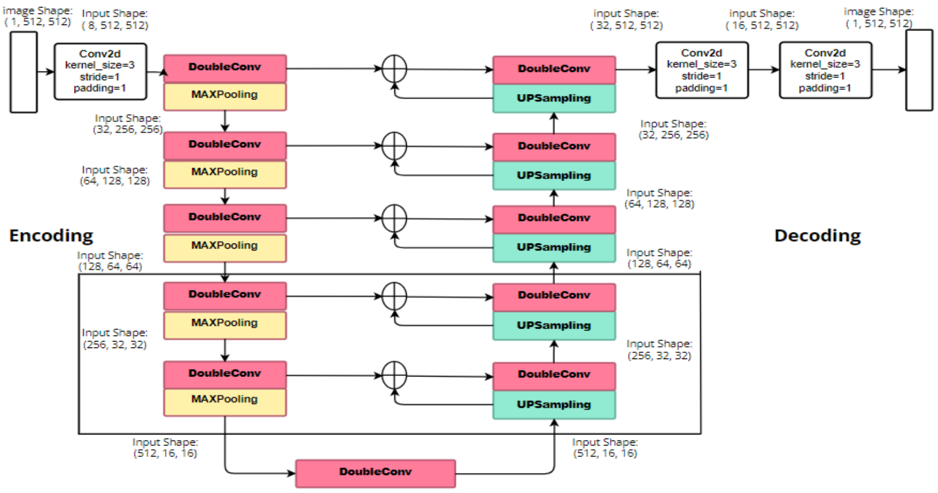


Figure 13 Generator Architecture

U-Net architecture proposed for biomedical image segmentation. It comprises a down-sampling network followed by an up-sampling network It adapts the U-Net architecture in three main ways:

- There are five (instead of four) down-sampling layers and up-sampling layers.
- All convolution layers keep the same image size.
- Eight 1 × 1 convolution kernels are applied to the input.

In the down-sampling process, three sets of two convolution kernels (the three boxes) extract feature maps. Then, followed by a pooling layer, the feature map projections are distilled to the most essential elements by using a signal maximizing process. Ultimately, the feature maps are 1/32 of the original size: 16x16.Successful training should result in 512 channels in this feature map, retaining important features.

In the up-sampling process, bi-linear interpolation is used to expand feature maps. At each layer, high-resolution features from the down-sampling path are concatenated to the up-sampled output from the layer below to form a large number of feature channels.

This structure allows the network to propagate context information to higher-resolution layers, so that the following convolution layer can learn to assemble a more precise output based on this information.

5.2.2.2 Discriminator Architecture

Discriminator has six 2D 3×3 CNN layers and two fully connected layers. Each CNN layer is followed by a leaky rectified linear unit as the activation function. Following the same logic as in G, all convolutional layers in D have the same small 3×3 kernel size. Let CkSs-n denote a convolution layer with a kernel size of k × k, a stride of s × s, n output channels, and leaky Relu activation function. The discriminator network consists of C3S1-64, C3S2-128, C3S1-128, C3S2-128, C3S1-256, C3S2-4, one hidden fully connected layer with 64 neurons and leaky Relu activation, and an output layer with one neuron and linear activation. There is no sigmoid cross-entropy layer at the end of the discriminator.
This combination of strides helps in gradually reducing the spatial dimensions while increasing the number of channels (feature depth), which is a common practice in convolutional neural networks for tasks like discrimination.

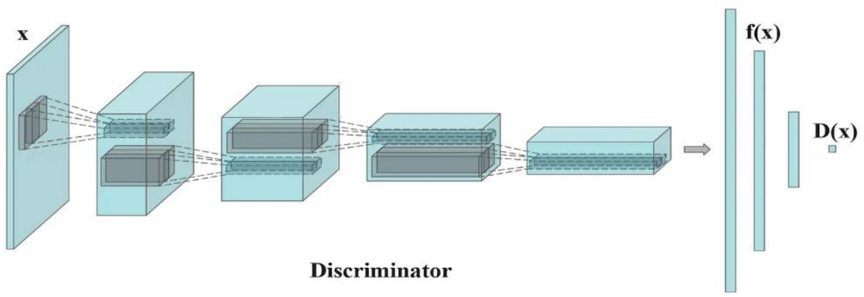


Figure 14 Discriminator Architecture

5.2.3. Design Constraints

- **Structure of data for model:** Changing data to make input image be fit in our project so making data preprocessing to be suitable in pipeline and generating multi types of noise on it is required.
- **Choosing the number of up sampling and down sampling layers is 5 in the generator:** change layers of U-Net to extract important features in image and recover noise.
- **All convolution layers keep the same image size:** in Convolution step images keep the same size does not change as want to filter noise To make down sampling and up sampling be able to extract important features in image and recover noise.
- **Choosing number of training generator and discriminator for each batch in epoch:** In GANs training needs to train generator and discriminator independently to be able to learn so after tuning the best result after train is to train generator 4 times and discriminator 3 times.

5.2.4. Methodologies

5.2.4.1. Noise Generation

The Types of noise handled:

- Block-Pixel noise: each pixel is set to zero with probability 0.05.
- Add Convolve noise: convolved with a Gaussian kernel k, and noise is added.
- Gaussian-Projection noise: add Gaussian noise.
- Salt and Pepper noise.

5.2.4.2. Data Preprocessing

- Resize the image by the longest dimension.
- Choose the type of noise (in the next point) to add noise by equal probability or choose to not add any noise by high probability.
- Pad image to be 512 x512 pixel
- Normalize to be in range 1-0

5.2.4.3. Experience

We tried to implement a denoiser model from Paper[6] but their approach did not work for our problem due to the section of data used and we want full image x-ray.
Try to change the data loader for the model to be suitable for our application and the results illustrated in figure 4.3.4.2.1. This is because model is not able to extract important features from the image so perceptual loss *lvgg* effect on trains to try to reduce it, so no learning and losses are still constant.

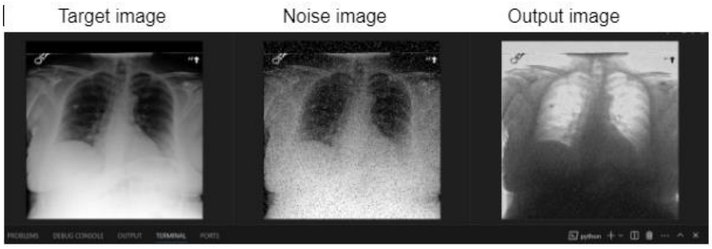


Figure 15 First Trial Denoiser Result

Increasing layer to five (instead of four) down-sampling layers and up-sampling layers and results improved as illustrated these results on 150 images of train and 50 images of test by ssim: 0.69 by 25 epoch.

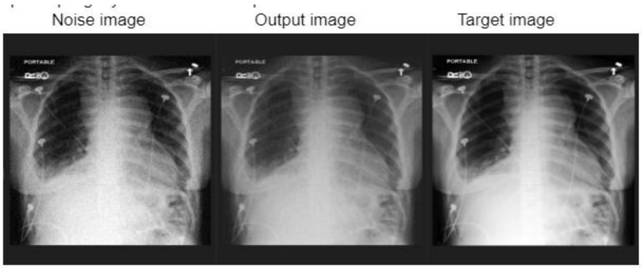


Figure 16 Second Successful Denoiser Result

5.2.4.4. Trainer:

In this section we show the loss functions used for the training procedure.

GAN loss: The discriminator's job remains unchanged which is *ladv*, but the generator is tasked with not only generating adversarial samples but also being near the ground truth output by pixel wise loss an *lmse*. Moreover, perceptual losses *lvgg* are also used to penalize any structure that differs between output and target. Thus, the generator loss is a weighted average of three losses *ltotal*.

- **Adversarial loss:** we compute the adversarial loss by using binary cross entropy loss.

Generator Loss:

- **Perceptual loss:** To allow the generator to retain a visually desirable feature representation, we also use the mean squared error (*MSE*) of features extracted by the pre-trained VGG network to represent a given image. We then define the perceptual loss as the Euclidean distance between the feature representations of a ground truth image and the corresponding denoised image.

$$lvgg = \sum_{i=1}^{W} \sum_{j=1}^{H} \left(vgg(I^{ND})_{i,j} - vgg(G(I^{LD}))_{i,j}\right)^2$$

- **Pixel-wise MSE**. The pixel-wise MSE loss is calculated as

$$lmse = \sum_{c=1}^{W}\sum_{r=1}^{H}\left(I^{ND}_{c,r} - G(I^{LD})_{c,r}\right)^2$$

$$l_{total} = \lambda_g\, ladv + \lambda_p\, lmse + \lambda_v\, lvgg$$

Where $\lambda_g = 20$, $\lambda_p = 1$, $\lambda_v = 100$ by tuning

5.3. Object Detector Classical Learning Approach

5.3.1. Functional Description

Object detector to detect 29 regions in the chest and extract the features by leveraging a combination of selective search, feature extraction, and machine learning models.

5.3.2. Modular Decomposition

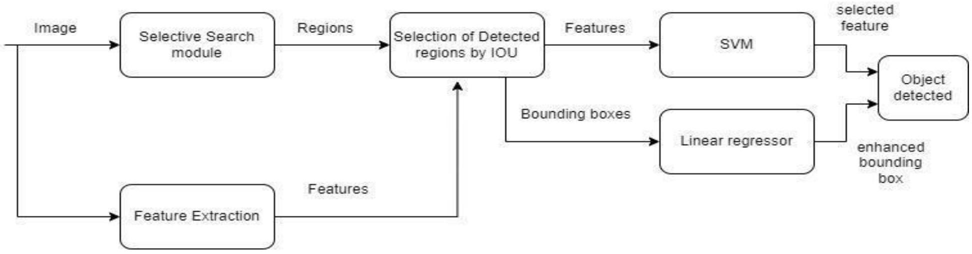


Figure 17 Object Detector ML Block Diagram

Region Proposals: Generate region proposals using selective search. Which implement from scratch.

Selective Search:

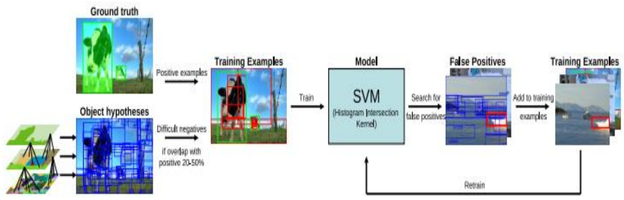


Figure 18 RCNN Selective Search

- **Initial Segmentation**: The algorithm starts by performing an initial segmentation of the image using a method like Felzenszwalb's segmentation algorithm. This divides the image into multiple small regions based on color, texture, and intensity information.

- **Feature Extraction**: For each region obtained from the initial segmentation, various features are extracted, including color histograms, texture descriptors (e.g., Local Binary Patterns), size, and shape features.
- **Region Similarity Calculation**: The algorithm computes the similarity between all pairs of regions.
- **Region Merging**: The most similar pairs of regions are iteratively merged. After each merger, the features of the new region are recalculated, and similarities with other regions are updated. This process continues until a stopping criterion is met, such as a fixed number of regions or a similarity threshold.
- **Bounding Box Generation**: The algorithm outputs the bounding boxes of the merged regions as candidate region proposals for object detection.

Feature Extraction: Extract features from each proposed region using Hog of Haar which implemented from from scratch to be suitable for medical images.

Classification: Classify each region using a trained SVM.
Bounding Box Regression: Refine the bounding box coordinates using a regression layer.

5.3.3. Design Constraints

Choosing Parameter for Felzenszwalb's segmentation algorithm: scale, sigma, Minimum area size to not detect unwanted region not related to chest.
Choosing feature Extraction: as images are related to the medical field so hog is more appropriate as the ability of HOG to robustly capture edge, shape and texture information makes it applicable to many medical imaging tasks involving detection. Its invariance to geometric transformations and lighting variations helps in analyzing the wide variability encountered in real-world medical images.

5.3.4. Experience

Our trail for training on small data by using but result is not good enough, is illustrated in figure 4.6.4.1 where the green boxes detected and red boxes is targets

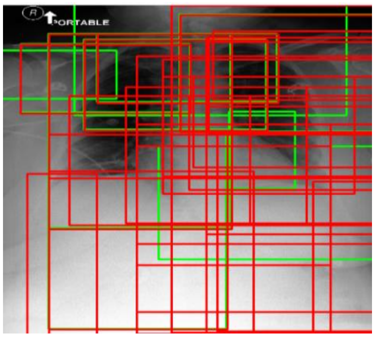


Figure 19 HOG Feature Extractor

Our use of Haar as feature extraction but not give us good results, is illustrated in figure 4.6.4.2 where the green boxes detected and red boxes is targets. After trials, using hog is good as the ability of HOG to robustly capture edge, shape and texture information makes it applicable to many medical imaging tasks involving detection. Its invariance to geometric transformations and lighting variations helps in analyzing the wide variability encountered in real-world medical images.

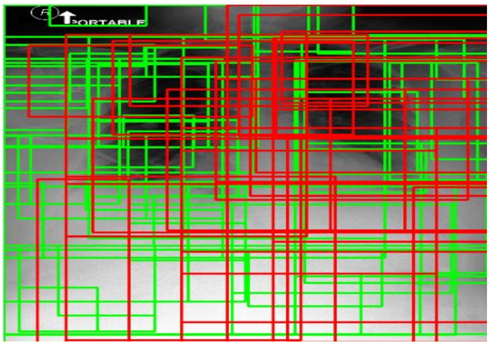


Figure 20 HAAR Feature Extraction

Chapter 6: System Testing and Verification

6.2. Testing Plan and Strategy

By Train Model in same data set and test it to make sure it's learned then train in all dataset by consider evaluate it by evaluation data and final test by testing data.

6.2.1. Module Testing

6.2.1.1 Denoiser Deep Learning Approach Testing

Metrics used:

- The structural similarity index measure (SSIM) is used for measuring the similarity between two images. The Structural Similarity Index (SSIM) metric extracts 3 key features from an image: Luminance, Contrast, Structure.
- Peak signal-to-noise ratio is a widely used metric that measures the quality of a processed image by comparing it to the original, assuming both images share the same resolution. PSNR represents the ratio between the maximum possible power of a signal, which is the original image, and the power of the noise, which is based on the discrepancy between the original and processed images.

$$PSNR = 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right) \qquad MSE = \frac{1}{m\,n}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2.$$

Results After 2 Epoches:

Size	1000	20000
Data Split	Validation	Testing
SSIM	71.5	78.48
PSNR	25.4	28.55

Table 2 Denoiser Metrics Results

Output Noised Target

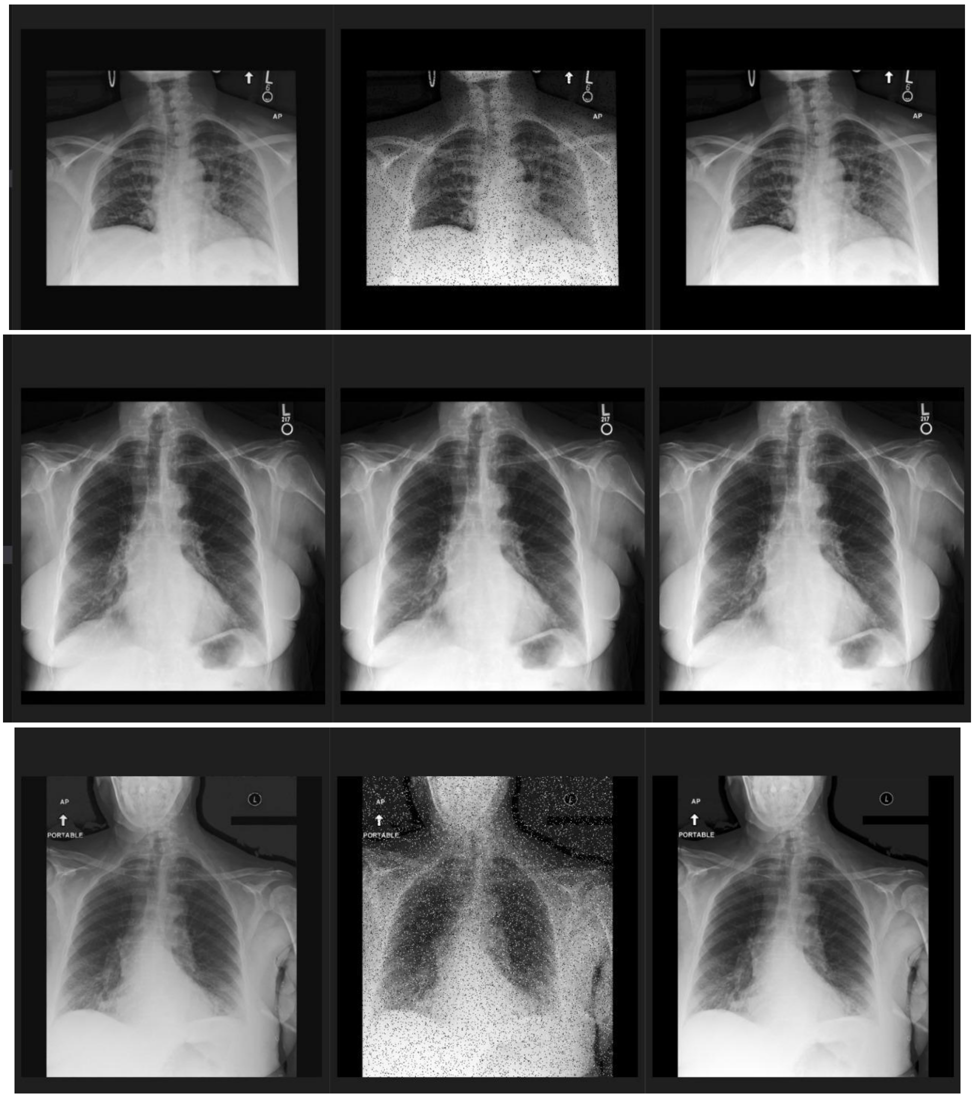
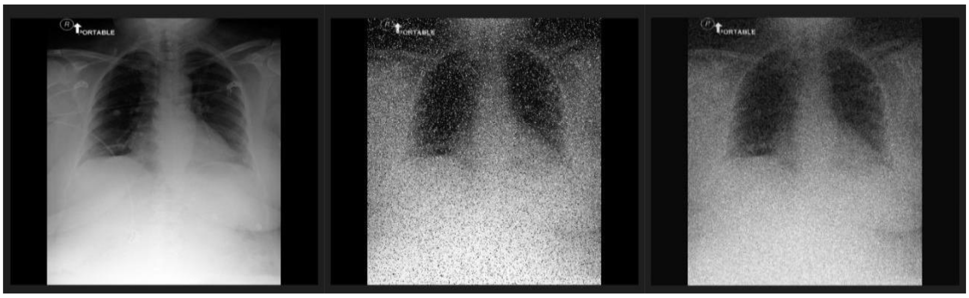


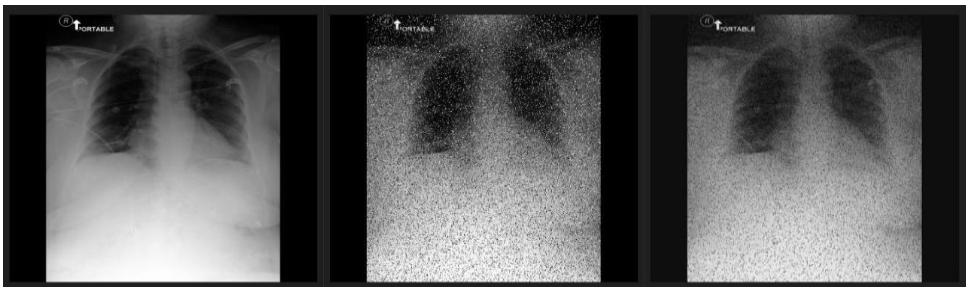
Table 3 Denoiser Image Results

Target Noised Output



Noise type: convolve noise, salt and pepper noise, gaussian projection noise
PSNR: 21.79

Target Noised Output



Noise type:block pixel noise, salt and pepper noise, convolve noise
PSNR: 21.39

Table 4 Denoiser Results On Images with Multiple Noise Types

6.2.1.2 Object detector Machine Learning Approach Testing

After Training SVM and linear regression by using hog feature extraction we get results:

Hog Feature Extraction:

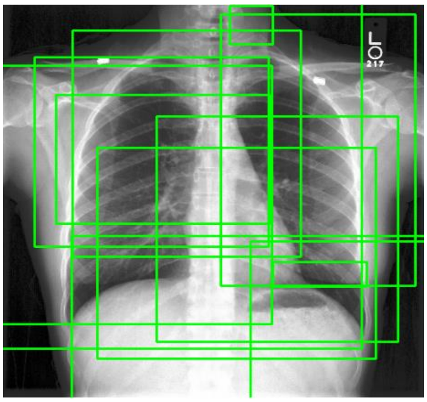


Figure 21 Bounding Boxes example (1) results from HOG

IOU Region(2)	0.44526825922402635
IOU Region(3)	0.5779246233969871,
IOU Region(4)	0.524560651521646
IOU Region(5)	0.5720338983050848,
IOU Region(6)	0.6641872261495803
IOU Region(7)	0.5597651335104152

IOU Region(8)	0.5094604484864482

Table 5 IOU Some Results for example (1)

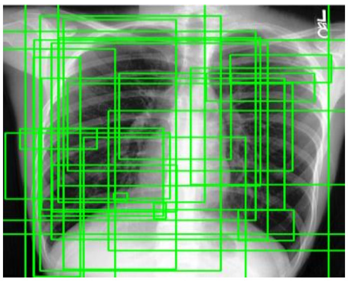


Figure 22 Bounding Boxes example (2) Same results from HOG

IOU Region(1)	0.5106138621442107
IOU Region(2)	0.44819189721272407
IOU Region(3)	0.5439073016042923,
IOU Region(10)	0.44059054516700064
IOU Region(12)	0.7573651698740793
IOU Region(13)	0.4669576059850374
IOU Region(14)	0.49874109010957834

Table 6 IOU Results for example (2)

6.3. Testing Schedule

we start testing the modules of X-reporto in February and try to enhance them until April

Chapter 7: Conclusions and Future Work

7.1. Faced Challenges

7.1.1 Denoiser Deep Learning

Training Generative Adversarial Networks (GANs) is particularly challenging because even minor changes to the parameters can significantly affect the model's performance. When working with medical images, such as chest X-rays, it is crucial to meticulously handle the data to ensure it is well-suited for this architecture. The primary goal is to preserve the critical medical features and diagnostic information within the images.

7.2. Conclusions

The project aimed to enhance automated medical report generation using advanced machine learning models and robust backend infrastructure. It focused on improving the efficiency, accuracy, and consistency of radiology reports, essential for medical diagnostics and patient care.

Key components of the system included image denoising, object detection, binary classifiers, and a language model for report generation. These models processed medical images, extracted features, and generated detailed reports. A heatmap module and disease classification component improved the system's ability to focus on abnormal regions and provide precise diagnostic information.

7.3. Future Work

7.3.1 Denoiser Deep Learning:

- Try to Mix types of noise and train modules on this data to be able handle many types of defects.
- Use Resnet on Perceptual Loss instead of Vgg for improving results.

References

[1] O'shea, Keiron, and Ryan Nash. "An introduction to convolutional neural networks." arXiv preprint

[2] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556. 2014 Sep 4.

[3] Ronneberger O, Fischer P, Brox T. U-net: Convolutional networks for biomedical image segmentation. InMedical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18 2015 (pp. 234-241). Springer International Publishing.

[4] Ahmadi, Mehdi, et al. "Reproducing AmbientGAN: Generative models from lossy measurements." arXiv preprint arXiv:1810.10108 (2018).

[5] Menon, Sumeet, et al. "Generating realistic covid-19 x-rays with a mean teacher+ transfer learning gan." 2020 IEEE International Conference on Big Data (Big Data). IEEE, 2020.

[6] Liu Z, Bicer T, Kettimuthu R, Gursoy D, De Carlo F, Foster I. TomoGAN: low-dose synchrotron x-ray tomography with generative adversarial networks: discussion. JOSA A. 2020 Mar 1;37(3):422-34.

[7] Q. Yang, P. Yan, Y. Zhang, H. Yu, Y. Shi, X. Mou, M. K. Kalra, Y. Zhang, L. Sun, and G. Wang, “Low-dose CT image denoising using a generative adversarial network with Wasserstein distance and perceptual loss,” IEEE Transactions on Medical Imaging, vol. 37, no. 6, pp. 1348–1357, June 2018