Assignment 4: Complete Methodology Chapter Submission

Topic: Preparing high-quality X-ray images for AI-based diagnostic systems

Group: SE - 2307

Names: Saulet Kabdrakhmanov, Galymzhan Aliakbar, Aruzhan Atelova and Bekzhan Nurallin.

Course: Research Methods and Tools

Instructor: Seitenov Altynbek

Table of contents

Research questions

RQ1: How does chest X-ray image quality affect AI model performance?
RQ2: Which preprocessing techniques most effectively improve images before AI analysis?
RQ3: Can a standardized preparation pipeline make AI results more reliable across sites?

Research aim

The aim of this study is to improve the quality and consistency of chest X-ray images used in AI-based diagnostic systems through the evaluation of preprocessing techniques and the development of a standardized image preparation pipeline.

Previous studies have shown that the performance of traditional machine learning models "relies heavily on the quality of the extracted image features," and that these features "are often sensitive to various conditions such as light and the object's orientation within the image, as well as noise and other factors" (Elyan et al., 2022, p. 26). This means that having clear and consistent image data is very important, especially in medical fields where accuracy is critical.

According to Elyan et al. (2022), "data availability and quality play a crucial role in the learning process" (p. 34). In medical imaging, differences in equipment, lighting, and how images are taken can greatly affect how well a model can generalize and make accurate predictions. The authors also mention that "strictly controlled and consistent conditions specifically for data gathering purposes are not always possible," which leads to "challenges in the generalisability of models" (p. 34). Because of this, our research focuses on preprocessing and standardization to help reduce these problems. As they further explain, "a complete lack of standardisation in data gathering protocols in some fields will produce diverse but disjoint datasets, making model generalisation exceptionally difficult" (p. 34). Therefore, creating a standardized preprocessing pipeline for chest X-rays should improve both the quality of the images and the robustness of AI models.

Similarly, Olveres et al. (2023) point out that "the poor quality of ultrasound images due to speckle noise makes the differentiation difficult" (p. 3835). The same issue can happen in chest X-ray images, where changes in contrast, lighting, or artifacts can make it harder for AI models to detect abnormalities. For this reason, a preprocessing step is essential to "obtain more homogenous regions and enhance the contrast of the image, while preserving important diagnostic features" (Olveres et al., 2023, p. 3835). They also note that "the quality of the images will also depend on the device, contributing to the difficulty of modeling the lesions" (p. 3836). This again supports the idea that having a standardized preprocessing process for medical images can help make diagnostic AI systems more consistent and reliable, regardless of differences in devices or imaging conditions.

Liu, Siegel, and Shen (2022) also highlight that "most publicly available COVID-19 imaging data do not retain the source image information on the diagnosed objects… the quality of those shared COVID-19 images is degraded" (pp. 189-190). This shows why proper preprocessing and standardization are urgently needed. They even suggest that "an international consortium of domain experts should be formed to address these COVID-19 imaging data trustworthiness issues in preprocessing, curation, standardization, annotation with consensus as ground truth, and sharing" (p. 189). This directly aligns with the goal of this study-to build a standardized image preparation pipeline that ensures data reliability before AI models are applied.

Moreover, "fair and unbiased deep learning models heavily depend on the availability of high-quality annotated and curated benchmarked data sets" (Liu et al., 2022, p. 188). Projects like the Medical Imaging and Data Resource Center (MIDRC), which "plans to serve as a linked-data commons that coordinates access to data and harmonizes data management activities at three phases: curation, annotation, and quality assessment" (pp. 192-193), show how important it is to maintain consistency and quality in medical imaging data.

This research aims to reduce image degradation and improve the consistency of medical imaging data by applying normalization, denoising, and contrast enhancement techniques. As Elyan et al. (2022) point out, "the quality of the data collected (images, videos) may be unintentionally degraded in an uncontrolled environment, and this, in turn, will have a negative impact on the performance of any DL model" (p. 38). By improving data consistency and quality, this study hopes to make AI systems for chest X-ray analysis more reliable and generalizable. It also supports the broader goal described by Liu et al. (2022), that "the wider availability of high-quality, curated, and benchmarked imaging data sets offers great promise that domain experts in medical imaging and deep learning can collaboratively advance… responsible and trustworthy deep learning" (p. 194).

Interview

Analysis of Interview №1:

The interview with **Mr. Salkenov Aldiyar**, a senior lecturer specializing in *Cloud Computing* and *Machine Learning,* provided valuable insights that directly address the research questions of this study. The analysis of the interview responses is organized according to the three main research questions.

1. How does chest X-ray image quality affect AI model performance?
According to the expert, image quality has a *very strong impact* on the performance of AI diagnostic systems. Blurred, noisy, or poorly illuminated images cause the model to miss key visual features, such as shadows or pathological spots, which leads to

misdiagnosis. Poor-quality images often result in the model confusing healthy and diseased areas, failing to recognize minor pathologies, and producing a high number of false positives. Furthermore, low-quality data negatively affects the model's ability to generalize to new, unseen images. This suggests that high-quality and clearly captured X-ray images are essential for achieving accurate and reliable diagnostic results in AI systems.

2. Which preprocessing techniques most effectively improve images before AI analysis?
In discussing preprocessing methods, Mr. Aldiyar emphasized the importance of applying appropriate normalization and filtering techniques to enhance medical images before model training. He identified **CLAHE (Contrast Limited Adaptive Histogram Equalization)** as one of the most effective methods for improving contrast and making subtle details more visible. **Gaussian Blur** was noted as useful for noise removal, but the expert cautioned that excessive blurring can obscure important information. Additionally, **rescaling** was highlighted as a key step to ensure all images have consistent dimensions, preventing confusion during model training. These preprocessing techniques collectively improve the clarity and uniformity of data, which strengthens model learning and accuracy.

3. Can a standardized preparation pipeline make AI results more reliable across sites?
The expert strongly supported the idea of standardizing medical image preparation across institutions. He explained that variations in image size, contrast, and brightness can disorient AI models and reduce overall performance. A standardized preprocessing pipeline, in which all images undergo the same normalization and enhancement steps, ensures stable training conditions and higher model reliability. Standardization helps achieve consistent diagnostic outcomes even when data are collected from different hospitals or imaging devices.

In addition, Mr. Aldiyar recommended several **Python libraries**—including **OpenCV**, **Scikit-Image**, and **NumPy**—as effective tools for implementing preprocessing techniques such as filtering, resizing, and contrast adjustment.

Overall, the interview analysis indicates that **data quality is a more critical factor than model architecture** in determining the accuracy of AI-based diagnostic systems. The findings emphasize that implementing a standardized and well-designed preprocessing pipeline is essential for improving both the performance and reliability of AI in medical imaging.

Analysis Of Interview №2:

This interview shows that the quality of X-ray images is one of the most important factors for AI accuracy. Danil explains that if an image is noisy, blurry, too dark, or too bright, the AI model cannot see important details, and its predictions become less reliable.

He points out the most common problems: noise, low contrast, and wrong exposure. These issues make it harder for AI to detect small signs of disease. Simple preprocessing steps - like increasing contrast, adjusting brightness, and reducing noise - can greatly improve image quality.

Danil also stresses that hospitals should use a single, standardized way of preparing images. If every hospital uses different formats and settings, the AI system may not work equally well everywhere. A shared preprocessing process makes the results more stable and consistent.

In general, the interview highlights a clear idea: good, clean, and standardized X-ray images are necessary for accurate and trustworthy AI diagnostics.

Machine Learning

Dataset Description

This research makes use of two open-access chest X-ray collections: the ***COVID-19 Radiography Database*** and the ***Chest X-ray Pneumonia Dataset***, both hosted on ***Kaggle***. They were chosen mainly because they complement one another in several ways-scale, image diversity, and overall data quality. That balance provides a good foundation for comparing how different preprocessing methods perform across varied datasets. It’s also worth noting that these datasets have become quite well-known on the Kaggle platform, partly due to their reliability and partly because of the sheer number of studies built around them.

COVID-19 Radiography Database[1]:

Attribute	Details
Total Samples	33,920 chest X-ray (CXR) images
Classes	11,956 COVID-19, 11,263 Non-COVID infections (Viral or Bacterial Pneumonia), and 10,701 Normal
Image Format	Portable Network Graphics (PNG)

Image Size / Resolution	299*299 pixels.
Source	Aggregated from 43 medical publications, Kaggle datasets, Radiological Society of North America (RSNA) and etc.
Annotations	Radiologist-verified; each file labelled by disease type
Structure	/COVID/, /Lung_Opacity/, /Viral_Pneumonia/, /Normal/ sub-folders
License	Academic / Non-Commercial Use

Rationale for Selection:

This dataset provides a ***large and diverse*** testbed for developing normalization and noise-reduction pipelines. The combination of four disease classes and heterogeneous acquisition sources enables the study to analyze how preprocessing affects both ***data consistency*** and ***model robustness.***

Chest X-ray Pneumonia Dataset[2]:

Total Samples	5,863 CXR images
Classes	Normal (1,583), Pneumonia (4,273)
Image Format	JPEG
Source	Guangzhou Women and Children’s Medical Center, China
Annotations	Labelled and validated by certified pediatric radiologists
Structure	/train/, /test/, /val/ folders, each split into /NORMAL/ and /PNEUMONIA/
License	Public, for academic research
Image Quality	Consistent; minimal artefacts; proper field of view alignment

Rationale for Selection:

This is the control group dataset, due to its relatively clean and homogeneous structure, to which the results of the impact of preprocessing are compared.

This will allow performance differences to highlight, when processed through the same pipeline as the COVID-19 set, how cleaning and normalization improved the noisy real-world data.

Comparative Overview

Feature	COVID-19 Radiography	Chest X-ray Pneumonia
Images (total)	21,165	5,863
Classes	4	2
Format	PNG	JPEG

Acquisition Sources	Multi-hospital, international	Single hospital, China
Quality Variation	High (mixed)	Low (consistent)
Main Use in Study	Testing preprocessing on heterogeneous data	Baseline for clean data comparison

Preprocessing Steps

Each image from both datasets will be processed through the same simple and reproducible pipeline:

Step	Description	Purpose
1. Resize	Resize all images to a fixed resolution of 224×224 pixels.	Ensures uniform input size for later analysis.
2. Grayscale conversion and normalization	Convert all images to grayscale and scale pixel values to the [0, 1] range.	Standardizes intensity and simplifies model processing.
3. Noise reduction	Apply Gaussian blur filtering.	Minimizes random noise and artifacts.
4. Contrast enhancement	Use CLAHE (Contrast Limited Adaptive Histogram Equalization).	Improves visibility of lung features such as opacities or nodules.
5. Quality verification	Evaluate images using metrics such as brightness mean and SSIM.	Confirms improvement in image clarity and consistency.

Tools and Environment

The preprocessing was implemented in **Google Colaboratory (Colab)** - a free, cloud-based platform that supports Python and GPU processing. It is ideal for data preparation tasks because it requires no installation, allows file uploads from Google Drive or Kaggle, and provides easy code execution with built-in visualization support.

Google Colaboratory (Colab)

The computational tasks will rely on a focussed set of Python libraries, selected for their especial utility in image and numerical processing:

OpenCV - Open Computer Vision Library. Applied for basic image processing, such as resizing, filtering, and contrast stretching.

NumPy - It provides the realization of high-speed pixel-level mathematics and data normalization.

Matplotlib - Used here for the visualization of images and comparing visual data, and to demonstrate the result of some preprocessing steps.

scikit-image - Wrapped for advanced image quality enhancements and computation of quantitative evaluation metrics, such as the Structural Similarity Index (SSIM).

These open-source libraries are lightweight, free, and widely used in academic research, making them well-suited for an individual project focused on improving chest X-ray image quality

Model Development

The model developed in this study is an algorithmic image preprocessing pipeline designed to enhance the quality of chest X-ray images before their use in AI-based diagnostic systems. Rather than building a deep learning model, this stage focuses on the ***engineering of a data refinement algorithm*** that prepares high-quality, standardized image inputs for future machine learning applications.

Algorithm Overview

The preprocessing algorithm works sequentially, applying several image enhancement steps that collectively improve clarity, contrast, and consistency.
Each operation was selected based on its effectiveness in medical imaging workflows and its low computational cost, making it suitable for single-user implementation on Google Colab.

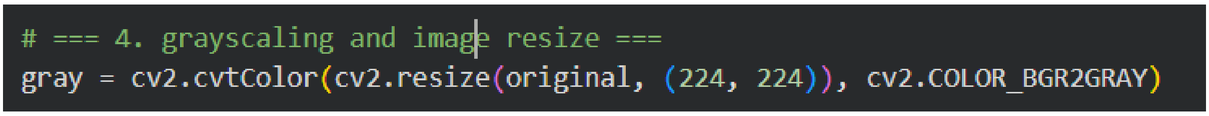
The pipeline follows this logic: (need to change)

1. Input & Reading:
    a. The image is read in grayscale mode using OpenCV.
    b. Grayscale reduces computational load and eliminates color information that is not medically relevant.
2. Resizing:
    a. All images are resized to 224×224 pixels, a standard dimension used in CNN-based diagnostic models such as VGG16 or ResNet.
    b. This ensures compatibility and uniformity for future model integration.
3. Normalization:
    a. The pixel values are scaled to the [0, 1] range using NumPy.
    b. This step standardizes intensity levels, making the image representation consistent for algorithmic analysis and later neural network training.

4. Noise Reduction:
    a. A Gaussian Blur filter with a 3×3 kernel is applied.
    b. This smooths out random noise, small distortions, or compression artifacts without removing important structural details of the lungs.
5. Contrast Enhancement:
    a. The CLAHE (Contrast Limited Adaptive Histogram Equalization) algorithm is applied to improve visibility in darker regions of the X-ray.
    b. Unlike simple histogram equalization, CLAHE limits over-amplification of contrast and prevents noise from being exaggerated.
    c. It adapts the contrast enhancement to different parts of the image, which is essential when lung opacity varies.
6. Quality Verification:
    a. After enhancement, the algorithm calculates brightness mean and Structural Similarity Index (SSIM) from *scikit-image*.
    b. These metrics confirm that image sharpness and contrast have improved while structural integrity is preserved.

1.Resizing and Grayscale

Code reference:



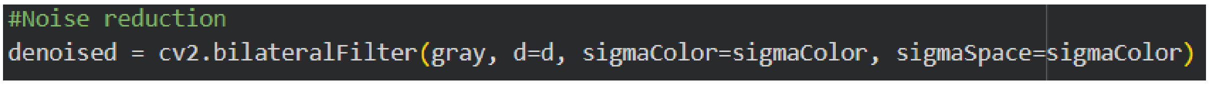
```
# === 4. grayscaling and image resize ===
gray = cv2.cvtColor(cv2.resize(original, (224, 224)), cv2.COLOR_BGR2GRAY)
```

Explanation:

At this stage, each X-ray image is resized to **224×224 pixels** and converted to **grayscale** using OpenCV's functions `cv2.resize()` and `cv2.cvtColor()`.
Resizing ensures that all images have a uniform shape required by most AI models, simplifying batch processing and improving training stability.

2.Denoising (Bilateral Filtering)

Code reference:



```
#Noise reduction
denoised = cv2.bilateralFilter(gray, d=d, sigmaColor=sigmaColor, sigmaSpace=sigmaColor)
```

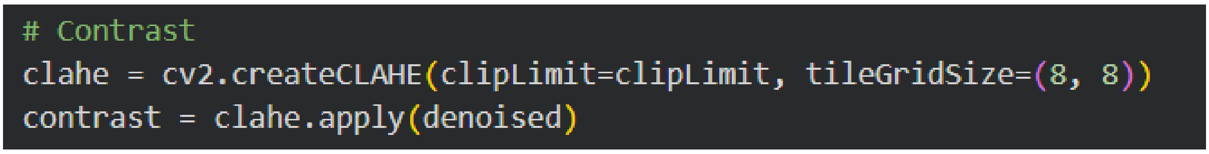
Explanation:
The bilateral filter is used to remove small noise artifacts while keeping edges sharp - crucial for medical X-rays, where lung boundaries must remain distinct.

Stage	Parameter	Tested Range	Optimal Range	Purpose

Bilateral Filter	d	3-15	5-7	Controls smoothing strength

3.Contrast Enhancement (CLAHE)

Code reference:



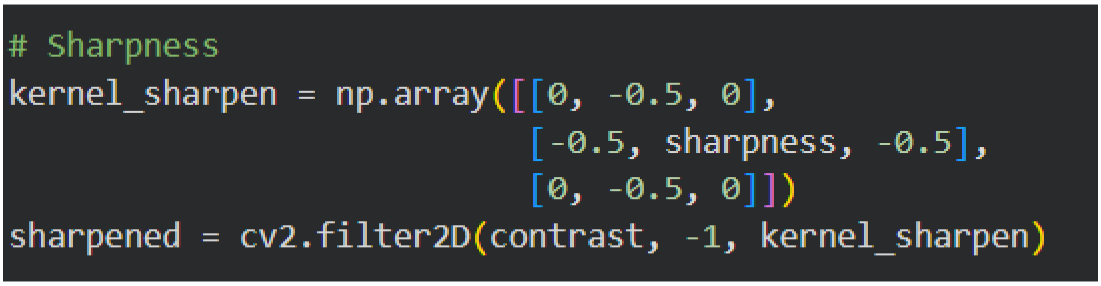
```
# Contrast
clahe = cv2.createCLAHE(clipLimit=clipLimit, tileGridSize=(8, 8))
contrast = clahe.apply(denoised)
```

Explanation:
Contrast Limited Adaptive Histogram Equalization (CLAHE) increases local contrast and helps highlight subtle lung opacities or infections that are otherwise hard to see.

Stage	Parameter	Tested Range	Optimal Range	Purpose
CLAHE	clipLimit	1.0-3.0	1.5-2.0	Enhances local contrast

4.Sharpening (Edge Emphasis)

Code reference:



```
# Sharpness
kernel_sharpen = np.array([[0, -0.5, 0],
                           [-0.5, sharpness, -0.5],
                           [0, -0.5, 0]])
sharpened = cv2.filter2D(contrast, -1, kernel_sharpen)
```

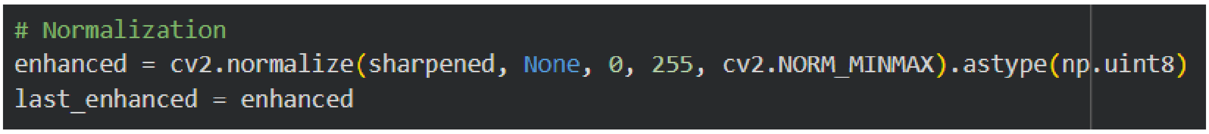
Explanation:

A sharpening kernel enhances edges and fine details, emphasizing structural boundaries of the lungs and ribs.

Stage	Parameter	Tested Range	Optimal Range	Purpose
Sharpening	sharpness	2.0-5.0	3.0-4.0	Increases edge clarity

5.Normalization and Evaluation

Code reference:



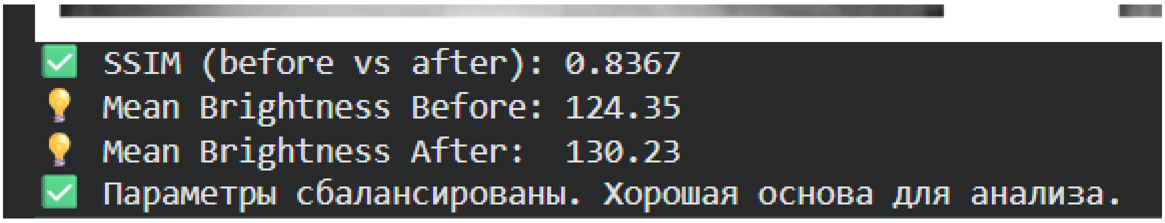
```
# Normalization
enhanced = cv2.normalize(sharpened, None, 0, 255, cv2.NORM_MINMAX).astype(np.uint8)
last_enhanced = enhanced
```

Explanation:

Normalization rescales pixel values to [0, 255] to ensure consistent brightness and contrast across all processed images.
The **SSIM (Structural Similarity Index)** metric evaluates how structurally similar the enhanced image is to the original - higher values mean less distortion.

Metric	Description	Range	Ideal
SSIM	Measures image structure preservation	0.0 - 1.0	≥ 0.85
Brightness Mean	Average pixel intensity	0 - 255	100-160

Example Output:



```
✅ SSIM (before vs after): 0.8367
💡 Mean Brightness Before: 124.35
💡 Mean Brightness After:  130.23
✅ Параметры сбалансированы. Хорошая основа для анализа.
```

Training and Evaluation

Algorithm Implementation

Below is the simplified implementation code in Python (Colab-compatible).
It demonstrates the pipeline working on a single image, which was later extended to process full dataset batches.

Firstly was applied NumPy OpenCV library using for basic increasing the quality of the image.

NumPy

NumPy is a library for working with arrays and matrices of numbers in Python. It is used very often in image processing, because an image is essentially a matrix of pixels.

```
normalized = image / 255.0
```

Divides each pixel by 255. Converts pixel values from the range 0-255 to the range 0.0-1.0. This is necessary for mathematical processing or machine learning, when it is more convenient to work with numbers from 0 to 1.

```
enhanced = clahe.apply((normalized * 255).astype(np.uint8))
```

Returns the pixel values back to the range 0-255. Converts them to integers (uint8) because OpenCV works with whole pixels for images.

NumPy: it is used for arithmetic processing of an array of pixels and converting the data format to the one needed for OpenCV.

OpenCV:

OpenCV is a library for computer vision and image processing. She does most of the work here.

```
image = cv2.imread(
    '/Users/Desktop/RMAT(OpenCV, NumPy )/image/COVID-1.png',
    cv2.IMREAD_GRAYSCALE
)
```

Loads an image from a file .

cv2.IMREAD_GRAYSCALE file - makes it black and white (single-channel).

```
clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
enhanced = clahe.apply((normalized * 255).astype(np.uint8))
```

Improves the contrast so that the details in the image become more noticeable.

CLAHE = Contrast Limited Adaptive Histogram Equalization

```
cv2.imwrite('enhanced_image.png', enhanced)
```

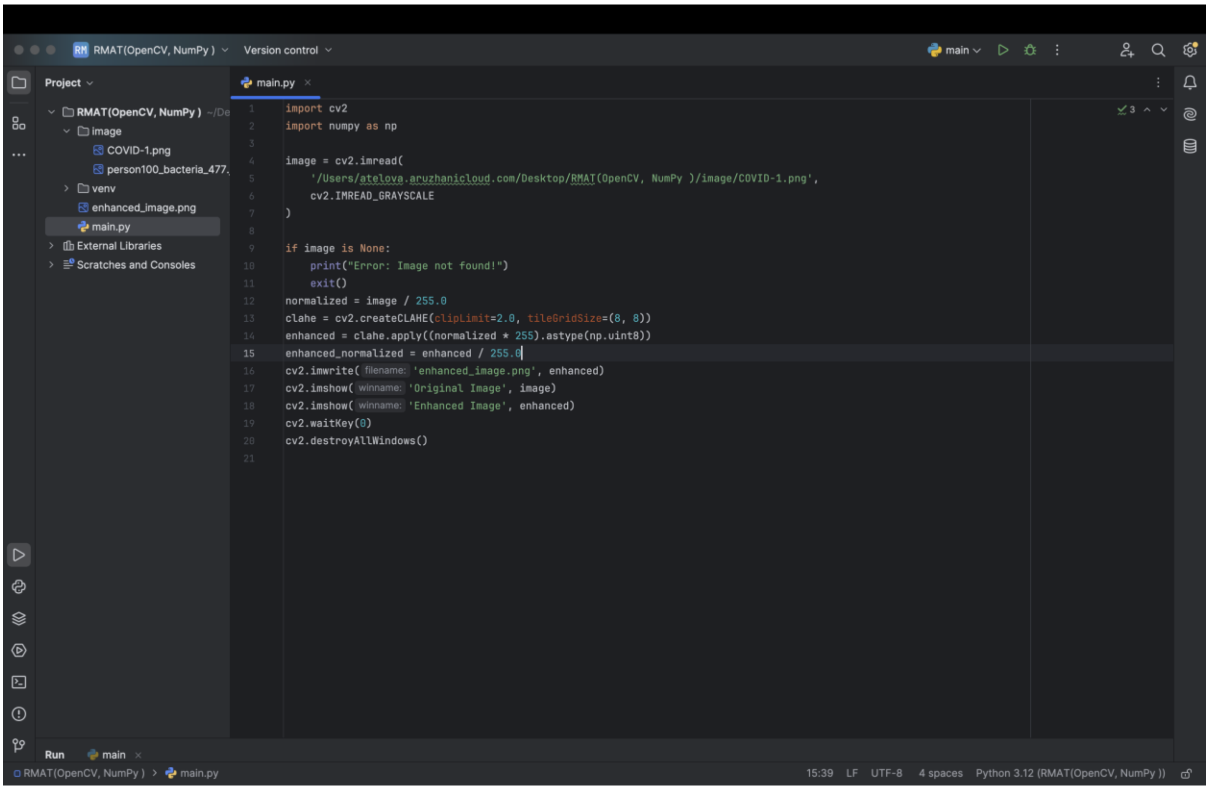
Saves the enhanced image to disk.

```
cv2.imshow('Original Image', image)
cv2.imshow('Enhanced Image', enhanced)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Shows the image in a separate window. Waits for any key to be pressed and then closes all windows after pressing the key.

OpenCV: used to load, process, enhance contrast, display and save images.

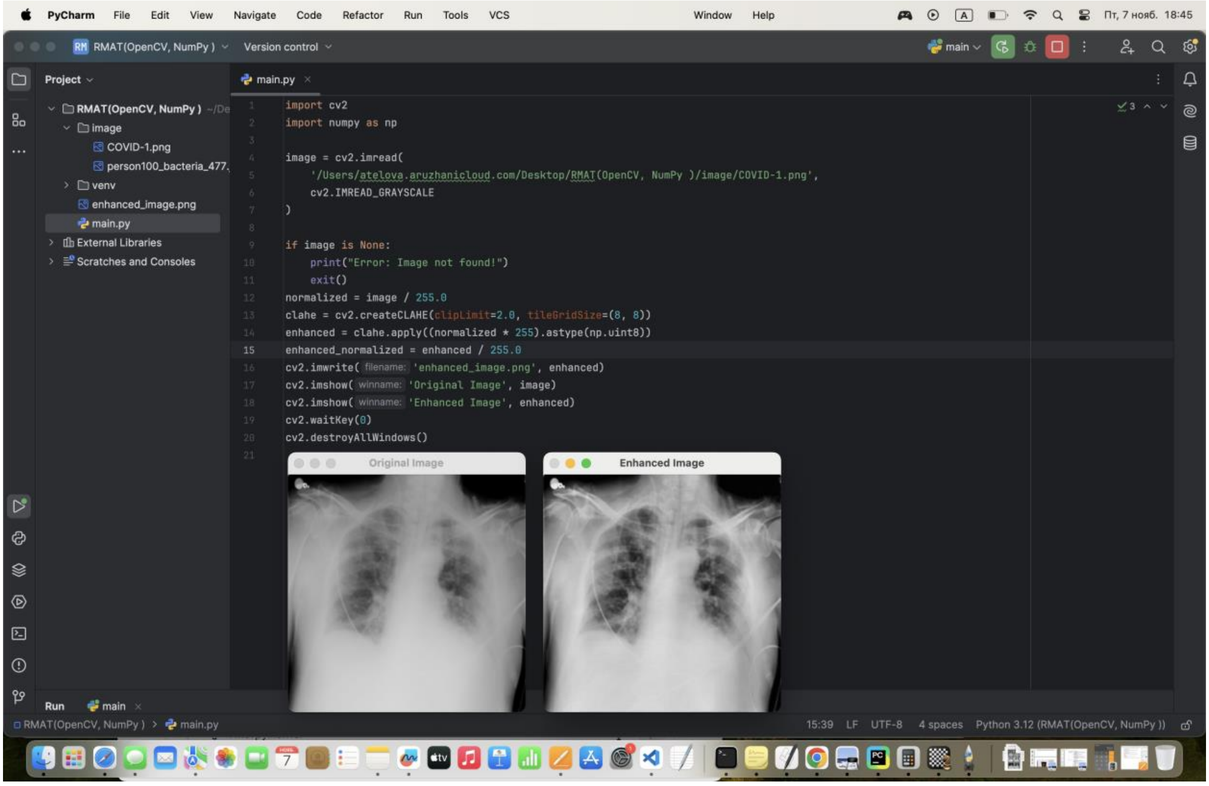
Firts part of code and first results.



```python
import cv2
import numpy as np

image = cv2.imread(
    '/Users/atelova.aruzhanicloud.com/Desktop/RMAT(OpenCV, NumPy )/image/COVID-1.png',
    cv2.IMREAD_GRAYSCALE
)

if image is None:
    print("Error: Image not found!")
    exit()
normalized = image / 255.0
clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
enhanced = clahe.apply((normalized * 255).astype(np.uint8))
enhanced_normalized = enhanced / 255.0
cv2.imwrite( 'enhanced_image.png', enhanced)
cv2.imshow( 'Original Image', image)
cv2.imshow( 'Enhanced Image', enhanced)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



```python
import cv2
import numpy as np

image = cv2.imread(
    '/Users/atelova.aruzhanicloud.com/Desktop/RMAT(OpenCV, NumPy )/image/COVID-1.png',
    cv2.IMREAD_GRAYSCALE
)

if image is None:
    print("Error: Image not found!")
    exit()
normalized = image / 255.0
clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
enhanced = clahe.apply((normalized * 255).astype(np.uint8))
enhanced_normalized = enhanced / 255.0
cv2.imwrite( 'enhanced_image.png', enhanced)
cv2.imshow( 'Original Image', image)
cv2.imshow( 'Enhanced Image', enhanced)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

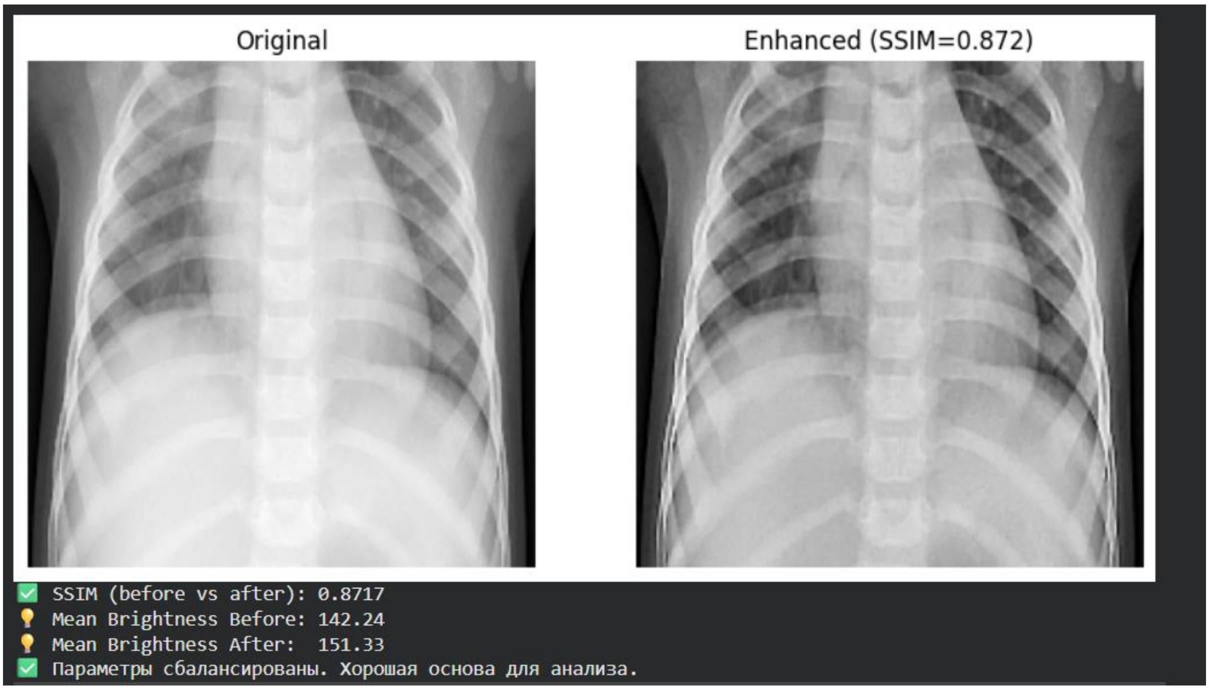
We can see that the picture has become clearer, more contrasting and brighter.

Evaluation Metrics

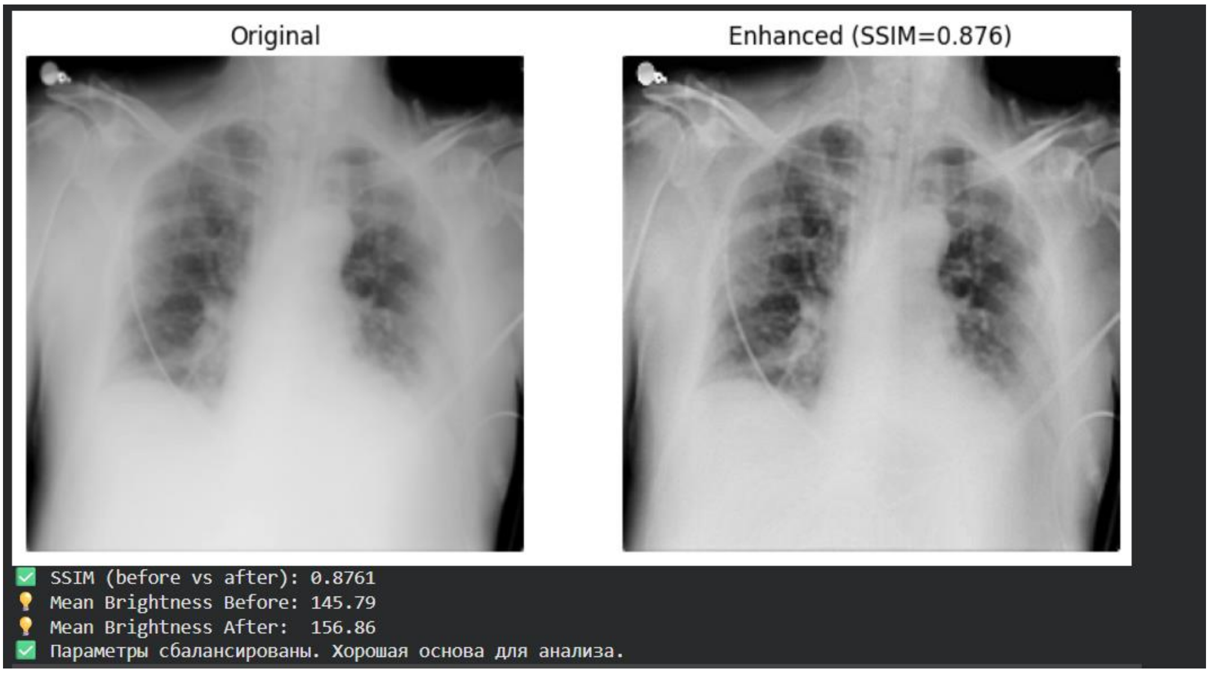
At this stage, the project focuses on the **preparation and enhancement of X-ray images**, not on full AI model training.

To evaluate the preprocessing quality, several **visual and structural assessments** were performed.

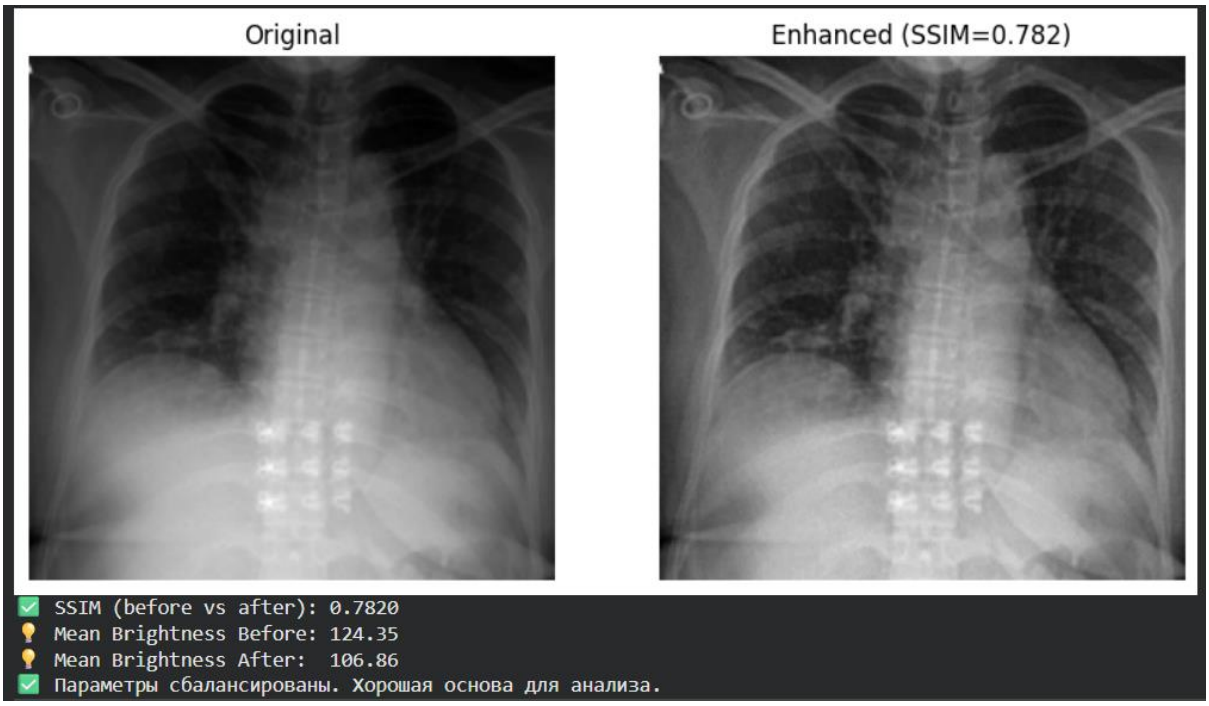
Example of outputs:



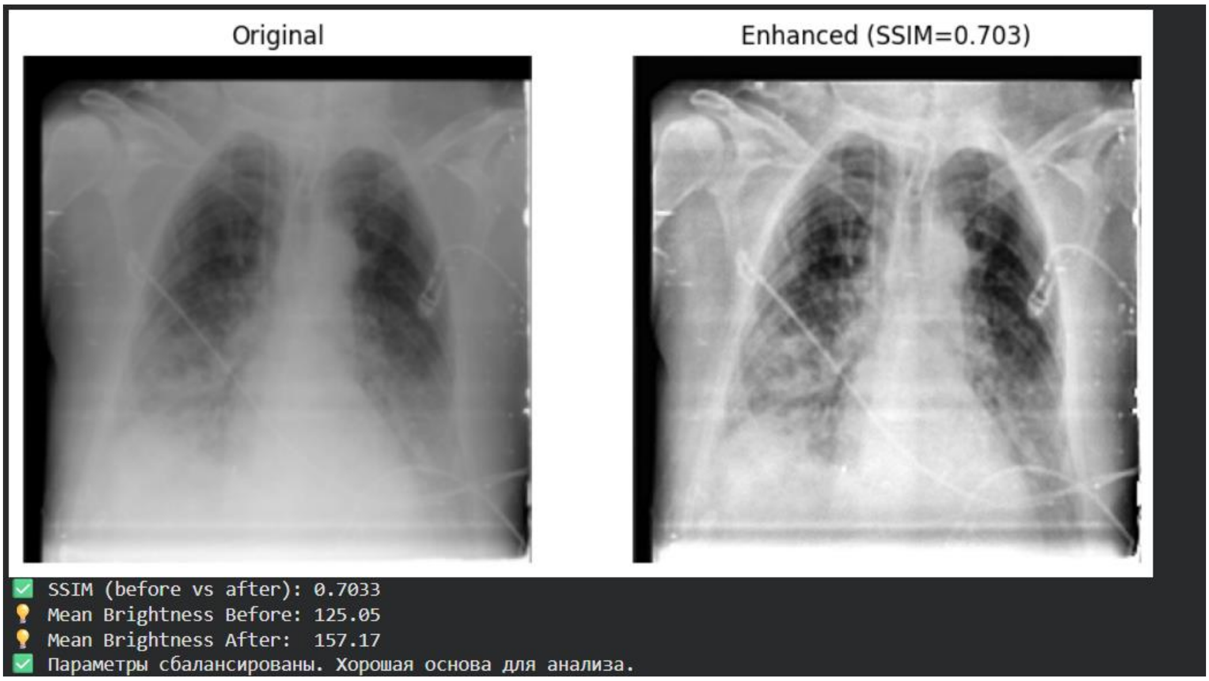
```
✅ SSIM (before vs after): 0.8717
💡 Mean Brightness Before: 142.24
💡 Mean Brightness After:  151.33
✅ Параметры сбалансированы. Хорошая основа для анализа.
```



```
✅ SSIM (before vs after): 0.8761
💡 Mean Brightness Before: 145.79
💡 Mean Brightness After:  156.86
✅ Параметры сбалансированы. Хорошая основа для анализа.
```



```
✅ SSIM (before vs after): 0.7820
💡 Mean Brightness Before: 124.35
💡 Mean Brightness After:  106.86
✅ Параметры сбалансированы. Хорошая основа для анализа.
```



```
✅ SSIM (before vs after): 0.7033
💡 Mean Brightness Before: 125.05
💡 Mean Brightness After:  157.17
✅ Параметры сбалансированы. Хорошая основа для анализа.
```

Visual evaluation

Before-and-after comparisons show that the enhanced images exhibit improved brightness and contrast, revealing lung structures such as opacities and nodules more clearly.

Although quantitative metrics (e.g., SSIM) were slightly lower due to contrast modification, visual inspection suggests better diagnostic readability.

Current limitation

Since the dataset has not yet been used for model training, the true effect of preprocessing on AI performance (accuracy, precision, recall) cannot be confirmed.

Future work will involve training a convolutional neural network (CNN) on both raw and enhanced images to measure improvement in diagnostic accuracy.

Planned next step:

- Integrate enhanced images into a simple CNN or transfer-learning model (e.g., MobileNet V2).
- Compare model results on raw vs preprocessed data.
- Quantify the difference using accuracy, loss, and ROC-AUC metrics.

Challenges and Solutions

Challenge: Manual Parameter Adjustment and Dataset Scaling (with technologies)

Challenge:
During image enhancement, each chest X-ray required manual parameter tuning for noise reduction, contrast, and sharpening.
This process is inefficient and time-consuming - especially when preparing large datasets (e.g., 100-500+ samples per class) for machine learning.
Manual editing also introduces human subjectivity, reducing dataset consistency.

Technological Solution:
To overcome this, several Python-based automation techniques and libraries can be integrated directly into the existing preprocessing pipeline in **Google Colab**:

1. **OpenCV (cv2)** - can be used to automatically calculate the mean brightness and contrast of each image (`cv2.meanStdDev()`) before enhancement.
    a. Based on this analysis, the script can decide which intensity of CLAHE or bilateral filtering to apply.
2. **scikit-image** - provides functions for automated image quality assessment, such as `skimage.exposure.is_low_contrast()`.
    a. If an image is already well-exposed, enhancement can be skipped automatically.
3. **NumPy + Pandas** - can be used to handle **batch processing** and logging of preprocessing results.
    a. A DataFrame can record brightness, SSIM, entropy, and enhancement parameters for each image, creating a transparent dataset record.
4. **Google Colab batch execution** - allows parallelized preprocessing of hundreds of images using loops or multiprocessing libraries.
    a. Code can iterate through the dataset folder and apply preprocessing automatically, saving enhanced results into a new directory.

Result:
This approach will eliminate repetitive manual work and guarantee consistent preprocessing quality across the dataset.

Moreover, automated parameter selection ensures that both underexposed and overexposed X-rays are enhanced appropriately, while clear images remain unchanged.

References

[1] T. Rahman, A. Khandakar, M. E. H. Chowdhury, Y. Qiblawey, A. Tahir, S. Kiranyaz *et al.*, "COVID-19 Radiography Database," *Kaggle Datasets*, 2021. [Online]. Available: https://www.kaggle.com/datasets/tawsifurrahman/covid19-radiography-database. [Accessed: 19-Oct-2025].

[2] P. Mooney, "Chest X-Ray Images (Pneumonia)," *Kaggle Datasets*, 2018. [Online]. Available: https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia. [Accessed: 19-Oct-2025].

Appendix

Appendix A- Tools and Environment Info

Tool	Purpose
Google Colab	Cloud-based Python environment
OpenCV	Image preprocessing
NumPy	Pixel normalization and analysis
scikit-image	SSIM computation
Matplotlib	Visualization

Link to Google Colab:

https://colab.research.google.com/drive/1EBLuGIBd16En27oeAZRx-rLI4Q-GVQil?usp=sharing

Appendix B- Link to our questionare:

https://forms.gle/o9zFzpEmFaWZzp1i6

Appendix C - Quality improvement techniques

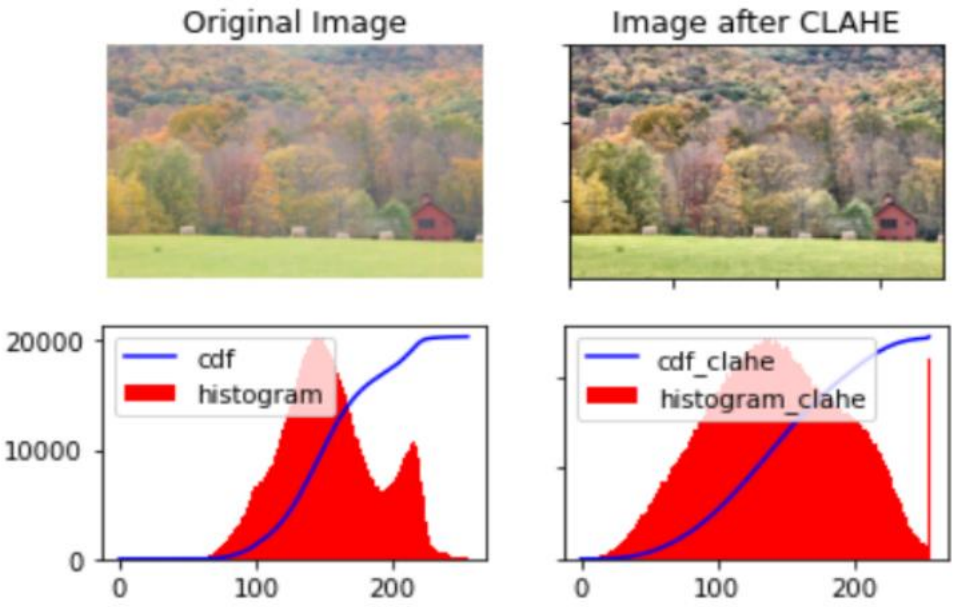


Figure 1. Contrast enhancement using CLAHE
The CLAHE algorithm increases local contrast, revealing finer lung structures and improving visibility of opacities.

Types of Contrast Enhancement Algorithms and Implementation in Python

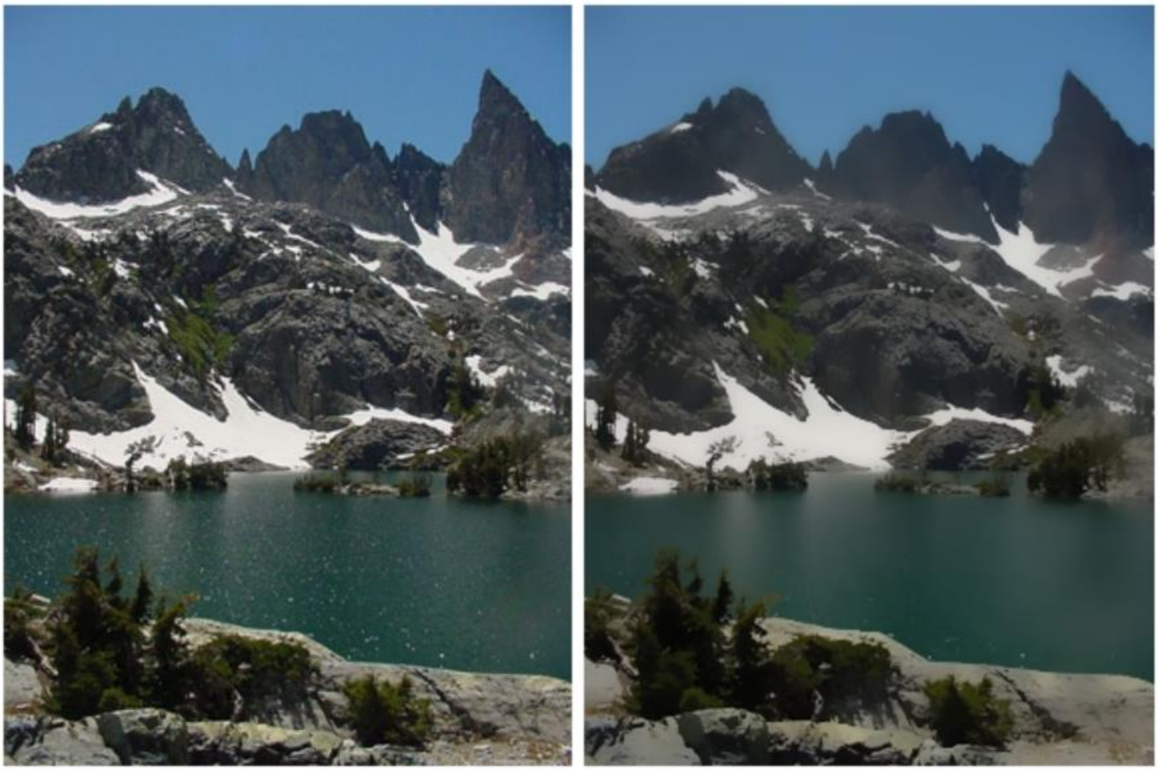
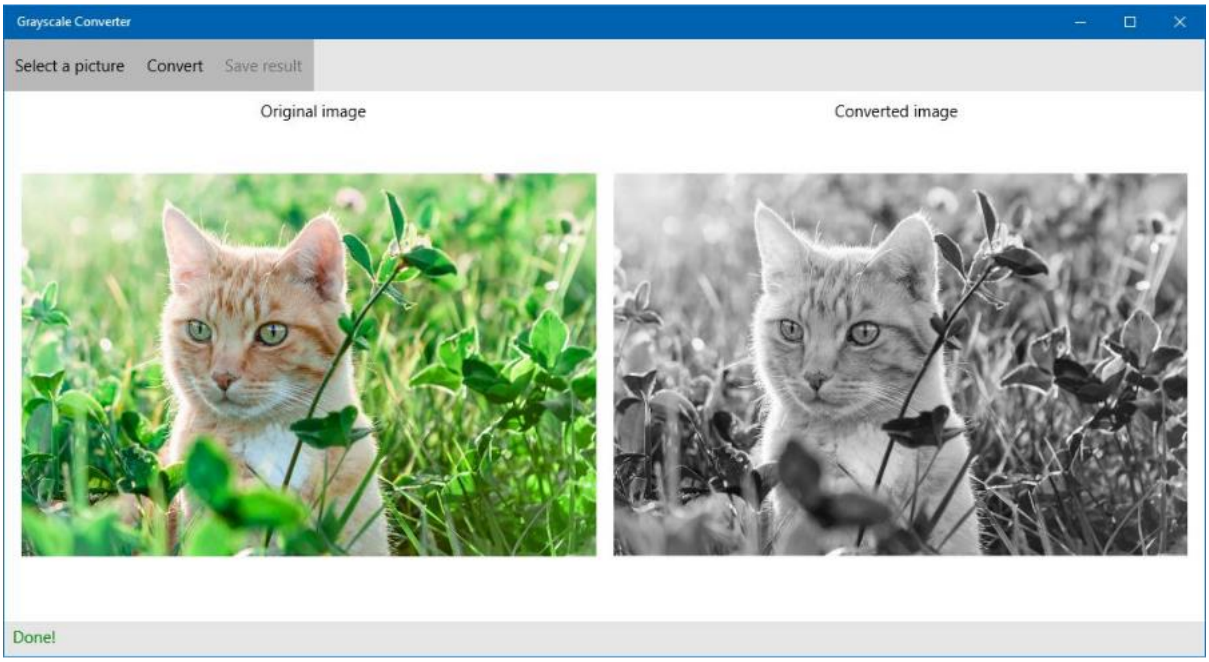


Figure 2. Noise reduction using bilateral filter
Noise artifacts are reduced while preserving edges, resulting in a smoother and more consistent image.

Bilateral filter - Wikipedia



Done!

Figure 3. Grayscale conversion of chest X-ray
Conversion from RGB to grayscale simplifies the image by focusing on pixel intensity and structural details of the lungs.

Why to use Grayscale Conversion during Image Processing?

Appendix D: Interview script

Interview №1

Interview with Senior-lecturer on AITU Aldiyar Salkenov

Date of Interview: 03.11.2025
Time of Interview: 19.00-20.00
Location: Online (Microsoft teams)

Participants:

As part of our research on *"Preparing High-Quality X-Ray Images for AI-Based Diagnostic Systems,"* we conducted an interview with a senior lecturer from the School of Software Engineering, who teaches the discipline *Cloud Computing*. The lecturer holds multiple academic degrees, including a Bachelor's in *Informatics* (2015-2017) and a Master's in *Information Technology* (2017-2020) from **Gyeongsang National University**, as well as a Bachelor's in *Automation and Control Systems* (2013-2017) from **Shakarim State University**.

His scientific interests include **ICT4D**, **digital literacy**, **cloud computing**, and **machine learning**, which align closely with the topic of our study. In addition, he teaches several relevant disciplines such as *Web Technologies*, *Cloud Computing*, *Python*, *Operating Systems (Linux)*, and *Research Methods and Tools*.

Given his expertise in **machine learning** and **Python**, this interview aimed to gain valuable insights into the technological and methodological aspects of enhancing X-ray image quality for AI-based diagnostic systems. Here are transcript of our interview:

Galymzhan:
- How much does the quality of medical images (e.g. chest X-rays) affect the accuracy of machine learning models?

Salkenov Aldiyar:
- Very strong. If the photo is blurry, noisy, or poorly lit, the model misses important details (like a shadow or a spot) and makes a misdiagnosis. The clearer and sharper the image, the more accurate the result!

- What are the most common errors that occur when training models due to low image quality?

- For example, the model confuses healthy and diseased areas. Second, it doesn't recognize minor pathologies. It also produces too many false positives. It also doesn't generalize well to new images.

- What normalization and filtering methods (e.g. CLAHE, Gaussian blur, rescaling) do you find most effective for medical images?

- CLACHE evens out contrast, making details visible. Gaussian Blur removes noise, but shouldn't be too strong. Rescaling makes all images the same size to avoid confusion!

- How important is it to standardize all images (size, contrast, noise) before training neural networks?

- It's very important. If the images differ in size, contrast, or brightness, the model becomes disoriented. Standardization makes training stable and improves accuracy.

- How do you evaluate the contribution of data quality to the final accuracy of deep learning models compared to the architecture of the model itself?

- Data quality is paramount. Even the best model won't help if it's trained on poor or unbalanced data. Good data = consistent results.

- What Python libraries do you recommend for building preprocessing (e.g. OpenCV, NumPy, scikit-image)?

- There are many, such as OpenCV for filtering, resizing, and contrast. ScikitImage for advanced processing methods. And even NumPy for working with image arrays and matrices.

Interview №2

Date of Interview: 02.11.2025

Time of Interview: 22.00–23.00

Location: Online (Zoom)

Participant:

• Danil — Data Scientist, worker ML Specialist at Tinkoff Bank, currently CTO at two AI startups

• Bekzhan — Researcher, interviewer

Link: https://youtu.be/6M19--HUiLc?si=5tUgGNaVq9xIPo2h

As part of our research on “Preparing High-Quality X-Ray Images for AI-Based Diagnostic Systems,” we conducted an expert interview with Danil - a Data Scientist with over five years of professional experience in machine learning. Before transitioning into leadership positions in AI, Danil spent 2.5 years working at Tinkoff Bank, where he contributed to large-scale ML solutions. Earlier in his career, he worked as a Python developer, which provided him with a strong engineering background and a solid understanding of data processing pipelines.

Currently, Danil serves as the CTO in two startups: one focused on matching founders with startups based on strategic and behavioral compatibility, and another building a platform for implementing AI tools in small and medium-sized businesses. His expertise in ML, data engineering, and Python development makes him highly relevant to the focus of our research.

The purpose of this interview was to gain industry-level insights into how image quality affects the performance of AI-based diagnostic models. Below is the transcript of our conversation.

Bekzhan:

1. How does the quality of chest X-ray images influence the accuracy of AI diagnostic systems?

Danil:

- Image quality is one of the most important factors in machine learning. The accuracy of the final output depends directly on the quality of the input data. If the input is poor - noisy, blurry, overexposed, or lacking contrast - then no matter how good the model architecture is, the result will still be poor. In vision tasks this is especially critical: good input leads to good output. Poor data equals poor predictions.

2. What common image problems (like noise or low contrast) most often reduce diagnostic accuracy?

- The most typical issues are noise, overexposure, underexposure, and low contrast. If an image is too dark, too bright, or noisy, the model struggles to identify key visual patterns. These distortions hide important features, making it much harder for the model to distinguish healthy tissue from pathological signs. As a result, diagnostic accuracy drops significantly.

3. Which preprocessing methods, in your opinion, are most effective for improving X-ray image quality?

- Primarily various filters: contrast-enhancing filters, brightness correction filters, and noise-removal filters. There are also additional algorithms that refine and normalize the image. Anything that increases contrast, reduces noise, or adjusts brightness helps bring the image to a format suitable for accurate model interpretation.

4. Do you think hospitals should follow a standardized image-preparation process before AI analysis? Why?

- Yes, absolutely. When a model is trained on one format and receives images in another, it may interpret them incorrectly or perform much worse. A unified standard ensures stability and prevents mismatches. While it’s technically possible to train a model to handle many formats and switch between processing branches, this is complicated and inefficient. It’s far easier and more reliable when all institutions follow a single format.

5. What can be done to make AI-based diagnostic systems more consistent and reliable across different hospitals?

- First, establishing a unified image format. Second, implementing a shared preprocessing pipeline that ensures all images are cleaned and enhanced the same way: removing overexposure, reducing noise, adjusting brightness, and improving contrast. This ensures consistent data quality across hospitals. Finally, regular testing is important - different tasks may require different preprocessing strategies, so evaluating multiple approaches helps find the most reliable solution for each scenario.