Enhancing Medical Imaging with SVD: Noise Reduction, Compression, and Lung Segmentation Evaluation

Ritik Raut (rraut@uab.edu) | Amitesh Sahoo (sahooa@uab.edu) | Charani Sri Veerla (cveerla@uab.edu) | Asmita Pal (apal@uab.edu) | Aneri Soni (asoni2@uab.edu)

1. Overview

1.1 Problem Statement:

Medical imaging datasets such as X-rays are important for diagnosing a range of health conditions. Medical facilities generate and store thousands of X-ray images. And thus, they require a significant amount of storage space. Cloud storage is expensive and requires a robust infrastructure. These datasets need to be maintained for a long-term and thus the number of records just keeps on increasing. Sometimes, these need to be transmitted from one department to another over limited bandwidth networks. Due to the size and volume of images, this can be slow and inefficient. These images tend to be noisy due to various technical and environmental reasons and thus it compromises the image quality. Lower image quality can make it difficult for medical professionals to identify subtle features. Automated diagnostics might struggle with noisy data, providing inaccurate results. Image compression and Denoising help overcome these challenges. And making sure that these processes are performed well can further improve the diagnostics.

1.2 How SVD Solves It:

SVD is a decomposing method that separates the image into three different matrices, which helps in defining the singular values creating the significant features, or those with most strength, in the image.

- Imaging Denoising with SVD ensures the retention of only the most vital features that define the image. Larger singular values bear the more important information, while smaller singular values correspond to the presence of noise. Hence, SVD enables the enhancement in quality of images by reconstructing them into cleaner versions.
- SVD-based image compression entails that salient features are not lost in compression, as could have happened in a simple compression. The greater the number of singular values, the lesser the compression with higher quality and vice-versa. Effectively compressed images would reduce storage and enhance the speed.
- Using SVD as a preprocessing step for segmentation ensures that the model focuses only on the true structure, eliminating irrelevant background details. This improves efficiency and gives better segmentation results.
- The SVD is tunable through the number of singular values allowed, making finer tuning possible depending on model requirements. It can, therefore, be a key tool in the building of performance optimization and the realization of the required outcome.

2. Methodology

2.1 Method Explanation

- ***Literature Review:*** Extensive literature review was performed with an aim to find a suitable model architecture along with the datasets for lung segmentation. Initially, ResNet50, VGG16, and classical CNN architecture were considered, tested, but these models tended to be insufficient while segmenting medical images-can't handle every fine-grained detail in chest X-rays. So, we would prefer U-Net architecture because this works well in a medical image segmentation task.
- ***Data Preprocessing:*** Chest X-ray images and lung segmentation masks are resized to the size of 512 X 512 pixels to keep the dataset consistent. Further, images are normalized

such that their pixel values are scaled within a range of 0 and 1 to be compatible with the U-Net Segmentation model.

- ***Noise Simulation:*** To simulate real world imaging scenarios, Gaussian noise was added. The Gaussian noise is added by adding variations to the pixel values which represent or mimic the real-world scenarios to simulate the distortions due to transmission errors.
- ***Noise Reduction:*** SVD is implemented to remove the variations keeping the top K singular values (features) intact. SVD filters out the noise and less important features so that only the important structures of the image remain. Thus, improving the quality of the image for further analysis.
- ***Image Compression:*** SVD finds its application in image compression. It reduces the dimensionality of an image by reconstructing the image using only the top K singular values. By maintaining the significant features, the size of the images is reduced while the structures and key features remain preserved.
- ***Segmentation:*** The integrated U-Net segmentation model segments the lungs from noisy, denoised, and compressed images. It is trained on the difference between the lungs and the rest of the image, and tested on noisy, denoised, and compressed images. The purpose of this is to enhance the accuracy and robustness of the model through visualization based on segmentation.
- ***Evaluation:*** To evaluate the performance and accuracy of the model, various evaluation metrics are calculated - Mean Squared Error (MSE), Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), Dice coefficient and Jaccard Index.

3. Experiments

3.1 Setup

- **Dataset:** The dataset used here is the "Chest X-Ray Lungs Segmentation" dataset from Kaggle, specially prepared for medical imaging including chest X-ray images and their respective segmentation masks.
    - **Data Composition** :
        - **Image Folder:** This folder contains 704 X-ray images of chests in grayscale format.

- **Mask Folder**: The folder contains 704 corresponding black-and-white mask images for lung segmentation.
- **Metadata File** (Metadata.csv): Supplies additional context; the file contains six columns:
  - id: The unique identifier of each image.
  - Gender: A patient's gender.
  - age: Patient's age.
  - county: Geographic location of the patient.
  - ptb: Binary label of the lung as TB+ or TB-.
  - remarks: Additional notes or comments.
- **Dataset Statistics**:
  - Total size:3.59 GB
  - Total images: 704 chest X-rays, 704 masks
- **Source and Availability**:

  The dataset can be found on Kaggle at: Chest X-Ray Lungs Segmentation Dataset

Implementation Details: We have considered the implementation of the project in Jupyter Notebook using Python-based libraries for data preprocessing, model training, and related tasks of evaluation.

IDE: Jupyter Notebook

Programming language: Python

Libraries: The project will utilize Python's libraries to ease implementation at each of the following stages: processing and visualization of data, processing of images, and deep learning.

- File and Data Management:
  - ***os***: For handling file system operations.
  - ***numpy***: It is useful for efficient number computations.
  - ***pandas**:* It is a library used to manipulate and analyze structured data.

- Data Visualization:
    - ***matplotlib.pyplot*** *and* ***seaborn****:* These packages are supposed to provide valuable visualizations in order to analyze the dataset and metrics of evaluation.
- Image Processing and Analysis:
    - ***cv2 (OpenCV)****:* This will be used for the reading, resizing, and processing of medical images.
    - ***scipy.ndimage****:* This module provides functions useful in more advanced image processing, like labeling and finding objects.
    - ***skimage.metrics****:* To calculate the structural similarity between images.
- Machine Learning and Deep Learning:
    - ***tensorflow***: This will allow the construction and training of a deep learning model with the U-Net.
    - ***sklearn.model_selection****:* This is used to split a dataset into training, validation, and testing subsets.
- Utilities:
    - ***tqdm*****:** to display progress bars during computational loops.
    - ***warnings****:* This can be used to control and suppress warnings at runtime.
    - ***random:*** To generate random numbers useful for the reproducibility of experiments.
    - ***concurrent.futures****:* For parallel operations, if performance becomes an issue.
- Statistical Computations*:*
    - ***scipy.stats***: To perform statistical tests and calculations.

- **Metrics for Evaluation:** The performance metrics used for this project are: *MSE* (Mean Square Error), *SSIM* (Structural Similarity Index), *PSNR* (Peak Signal-to-Noise Ratio), *Jaccard Index* and *Dice Coefficient*.

a) **MSE:** MSE measures the average squared difference between the pixel values of the original image and the denoised image. Lower MSE indicates better denoising.

$$\textbf{MSE} = \mathbf{1/mn} \sum_{i=1}^{m} \sum_{j=1}^{n} [\boldsymbol{I}(\boldsymbol{i},\boldsymbol{j}) - \boldsymbol{I}'(\boldsymbol{i},\boldsymbol{j})]^{2}$$

where (I – original, clean image; I’ – denoised image)

b) **PSNR:** PSNR represents the ratio between the maximum possible pixel value and the distortion (noise). Higher PSNR values indicate better denoising quality.

$$\textbf{PSNR} = \mathbf{10}\,\mathbf{log_{10}} \left(\frac{\boldsymbol{MAX_I^2}}{\boldsymbol{MSE}}\right)$$

where (I – original, clean image; I’ – denoised image)

c) **SSIM:** SSIM measures the similarity between two images by considering luminance, contrast, and structure. The range is between -1 and 1, with 1 meaning perfect similarity.

$$\textbf{SSIM} = \frac{(2\mu_I\mu_{I'} + C_1)(2\sigma_{II'} + C_2)}{(\mu_I^2 + \mu_{I'}^2 + C_1)(\sigma_I^2 + \sigma_{I'}^2 + C_2)}$$

μ_I , $\mu_{I'}$ are the mean pixel values; σ_I^2 , $\sigma_{I'}^2$ are the variances of pixel values; $\sigma_{II'}$ is the covariance between I and I’ & C_1, C_2 are two small constants to stabilize the division.

d) **Jaccard Index:** The Jaccard Index, also known as the Intersection over Union (IoU), is a metric commonly used to evaluate the performance of segmentation models. It measures the overlap between the predicted segmentation mask and the ground truth mask. The formula for the Jaccard Index is:

$$\textbf{IoU} = \frac{|\boldsymbol{A} \cap \boldsymbol{B}|}{|\boldsymbol{A} \cup \boldsymbol{B}|}$$

where A is the set of pixels in the ground truth mask (y_true), B is the set of pixels in the predicted mask (y_pred), |A∩B| is the number of pixels where both the ground truth and the prediction are 1 (i.e., the intersection). |A∪B| is the total number of pixels where either the ground truth or the prediction is 1 (i.e., the union).

e) **Dice Coefficient:** The function calculates the Dice similarity coefficient (also known as the Dice score), which is a measure of overlap between two binary masks.

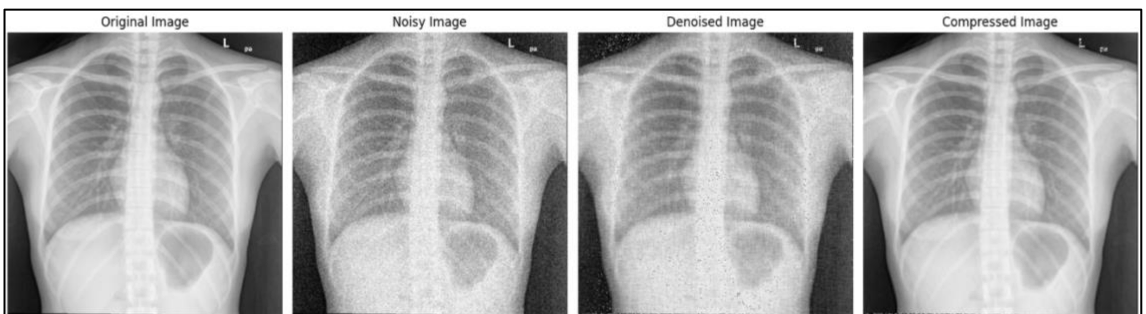
It gives a value between 0 and 1, where 1 indicates perfect overlap, and 0 indicates no overlap.

$$Dice\ Coefficient = \frac{2 \times Inter\sec tion + smooth}{Sum\ of\ True\ Labels + Sum\ of\ \Pr e dictedLabels + smooth}$$

3.2 Results

- **Performance Evaluation:** The model's performance was quantitatively evaluated using metrics such as Dice Coefficient, Jaccard Index, Accuracy, PSNR, SSIM, and MSE. The segmentation accuracy was 97.7% on compressed images, 76.43% on denoised images and 56.6% on noisy images on U-net.

  Another consideration is that noise intensity reduction would lead to further improved performance, as shown in the graph where for lower noise levels, better metrics are achieved for all categories of images. This really underlines the sensitivity of this model to noise presence and how important noise may be handled during pre-processing.

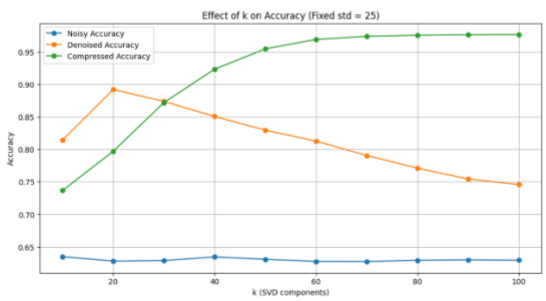


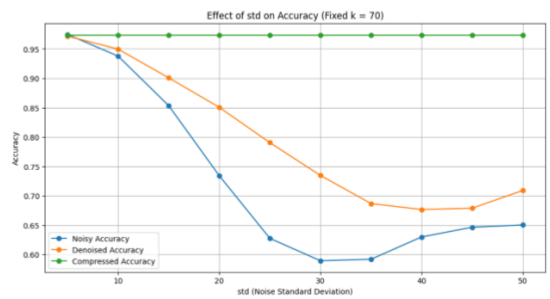

```
Processing images:   2%||           | 5/212 [00:05<04:05,  1.18s/it]
Testing completed!
Results:
Accuracy on Noisy Images: 0.5666
Accuracy on Denoised Images: 0.7643
Accuracy on Compressed Images: 0.9770
```

```
Epoch 62/64
123/123 ━━━━━━━━━━━━━━━━━━━━ 0s 144ms/step - accuracy: 0.9943 - dice_coefficient: 0.9846 - jaccard_index: 0.9697 - loss: 0.0125
Epoch 62: val_dice_coefficient improved from 0.95634 to 0.95675, saving model to best_model.keras
123/123 ━━━━━━━━━━━━━━━━━━━━ 27s 216ms/step - accuracy: 0.9943 - dice_coefficient: 0.9846 - jaccard_index: 0.9697 - loss: 0.0125 - val_accuracy: 0.9797 - val_dice_coefficient: 0.9567 - val_jaccar
d_index: 0.9175 - val_loss: 0.0993
Epoch 63/64
123/123 ━━━━━━━━━━━━━━━━━━━━ 0s 142ms/step - accuracy: 0.9947 - dice_coefficient: 0.9857 - jaccard_index: 0.9719 - loss: 0.0115
Epoch 63: val_dice_coefficient did not improve from 0.95675
123/123 ━━━━━━━━━━━━━━━━━━━━ 26s 209ms/step - accuracy: 0.9947 - dice_coefficient: 0.9857 - jaccard_index: 0.9719 - loss: 0.0115 - val_accuracy: 0.9791 - val_dice_coefficient: 0.9556 - val_jaccar
d_index: 0.9152 - val_loss: 0.1002
Epoch 64/64
123/123 ━━━━━━━━━━━━━━━━━━━━ 0s 139ms/step - accuracy: 0.9948 - dice_coefficient: 0.9862 - jaccard_index: 0.9727 - loss: 0.0113
Epoch 64: val_dice_coefficient did not improve from 0.95675
123/123 ━━━━━━━━━━━━━━━━━━━━ 26s 204ms/step - accuracy: 0.9948 - dice_coefficient: 0.9862 - jaccard_index: 0.9727 - loss: 0.0113 - val_accuracy: 0.9797 - val_dice_coefficient: 0.9565 - val_jaccar
d_index: 0.9170 - val_loss: 0.1017
```

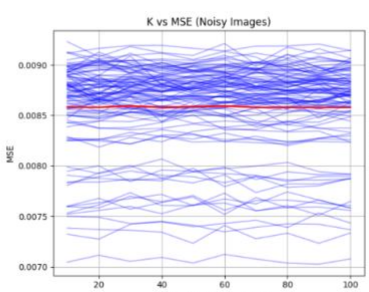
```
Final Training Metrics:
Training Accuracy: 0.9946
Training Dice Coefficient: 0.9859
Training Jaccard Index: 0.9723
Training Loss: 0.0117

Final Validation Metrics:
Validation Accuracy: 0.9797
Validation Dice Coefficient: 0.9565
Validation Jaccard Index: 0.9170
Validation Loss: 0.1017
```

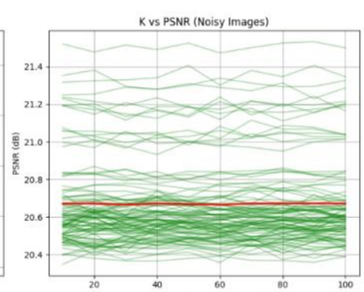
- **In-Depth Performance Analysis:** The segmentation model demonstrated consistent performance across various scenarios. It achieved higher metrics for clean and denoised images compared to noisy images, showcasing its reliability in standard conditions. However, its sensitivity to noise was evident, as the Dice coefficient and Jaccard Index decreased for noisy images. The metrics for compressed images via SVD showed a balance between maintaining image quality and compression effectiveness.

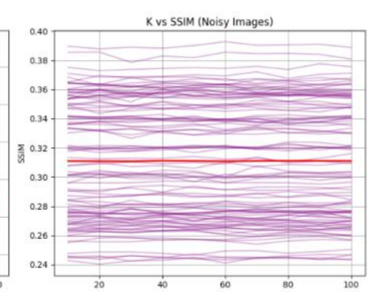


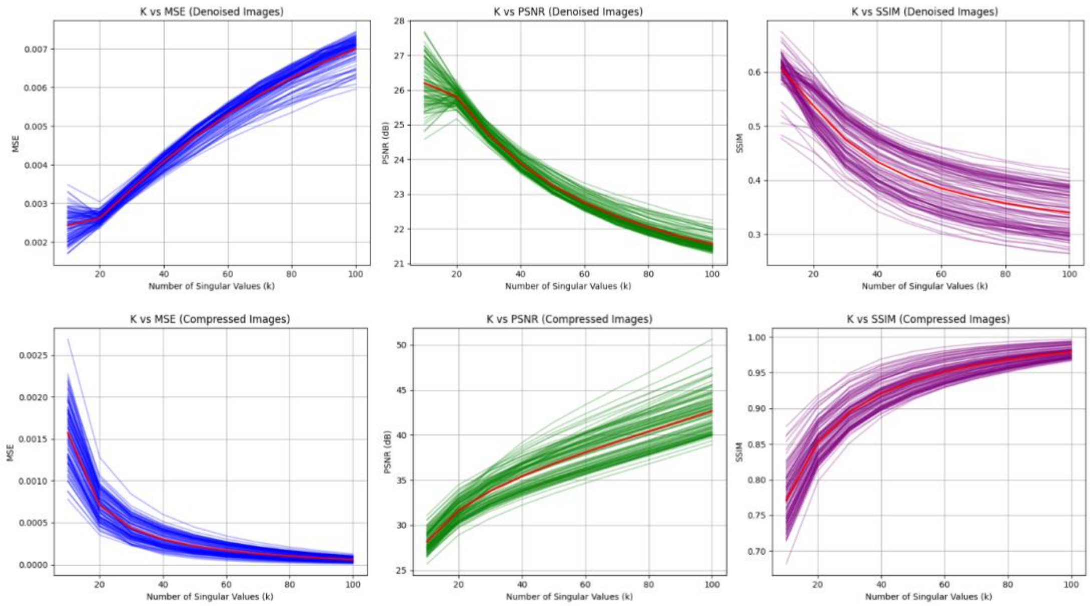


- **Parameter Study of k:** The parameter k, representing the number of singular values retained in SVD, significantly impacted the results. For compressed and denoised images, higher k values led to better performance metrics, indicating improved image reconstruction quality. The optimal k was found to balance accuracy and computational efficiency, with k=70. The below plots show metrics vs. k trend for 100 random images from the dataset.









- **Analysis of Noise Impact:** It is observed from the noisy image analysis that increasing the standard deviation significantly degraded the performance of segmentation. The metrics, Dice coefficient, and Jaccard index decreased with the increase in the amount of noise, depicting the sensitivity of the U-Net model for noisy inputs. Thus, application of the SVD-based denoising improved performance due to dampening the degrading effect caused by the noise.
  - **Why do the metrics degrade here?**

    Interestingly, the second graph for denoised images shows an opposite trend, where increasing the number of retained singular values (k) results in poorer performance. This is a phenomenon that can be described as follows:
    - **Impact of Noise on Higher Singular Values**: The Gaussian noise added (std=25) perturbs the entire singular value spectrum. However, the smaller singular values are disproportionately affected.
    - Consider retaining only a few singular values (k) which will effectively remove these noisy components from the image denoising process.
    - As *k* increases these noisy singular values are preserved and give rise to:
      - **Higher MSE**: The reconstruction is more away from the original image as the noise reintroduced at higher where k outweighs the benefit of capturing finer detail.

        - **Lower PSNR/SSIM**: The structural similarity and signal-to-noise ratio degrade because of interference from retained noise in the quality of the image.
      - **Noise vs. Signal Trade-off**:
        - The introduced noise (std=25) dominates over finer structural details encoded in the higher singular values.
        - Retaining more singular values (k) amplifies the effect of noise, which worsens the general quality of an image and correspondingly reduces segmentation metrics.
      - **Dominance of Low-Rank Components**: The largest singular values capture the main structures of an image, corresponding to the low-rank components. Smaller singular values mainly represent minor details and noise. For purposes of denoising, it is enough to retain only the largest singular values (lower *k*) Yield a better performance since the noisy components are effectively discarded.
- **Summary of Results:** Below are the average PSNR, average SSIM, and average MSE for three categories of images-Noisy, Denoised, and Compressed-predicted with a fixed k=70 and std=25, for 100 test images.
  - Key Observations
      - **PSNR : Peak Signal-to-Noise Ratio**

        The order of PSNR values is as follows:

        Noisy Images < Denoised Images < Compressed Images

        Compressed images recorded the highest PSNR, which in turn supported their reconstructive quality with less noise.

      - **SSIM (Structural Similarity Index):**

        This gives the order of SSIM values as follows:

        Noisy Images < Denoised Images < Compressed Images

Compressed images showed the highest structural similarity with original images, resisting the capability to maintain both fine details and global structure.

- **Mean Squared Error (MSE):**

  The order of MSE values is given by

  Compressed Images < Denoised Images < Noisy Images

  Accordingly, compressed images had the least error; thus they are seen to be very precise in reconstructing the original image.

- **Insights**

  Compressed images have practically always outperformed noisy and denoised images regarding PSNR and SSIM while keeping the lowest MSE. In the results, much work of improvement was done on the noisy images to the denoised ones, especially in error reduction (MSE) and perceptual quality improvement (PSNR and SSIM). These results further establish the efficacy of SVD-based methods for denoising as well as compression in medical imaging, therefore requiring an optimum k value.

```
Processing 100 images for k=70 and std=25: 100%|██████████| 100/100 [00:33<00:00,  2.98it/s]
Average PSNR, SSIM, and MSE for k=70 and std=25 for 100 images
     Category  Average PSNR (dB)  Average SSIM  Average MSE
0       Noisy          20.675116      0.217086     0.008580
1    Denoised          24.362778      0.349268     0.003665
2  Compressed          37.875662      0.925630     0.000179
```

4. Individual Contribution

- **Ritik Raut:** Conducted literature review, implemented SVD, enhanced the codebase, and analyzed all metrics in relation to k (number of singular value components).
- **Amitesh Sahoo:** Designed and evaluated the model architecture, contributed to the literature review, implemented SVD, enhanced the codebase, and visualized evaluation metrics.
- **Charani Sri Veerla:** Performed analysis of metrics vs. k, contributed to the literature review, enhanced the codebase and implemented SVD.
- **Asmita Pal:** Conducted literature review, compiled a list of data sources, enhanced the codebase, implemented SVD, and documented the project
- **Aneri Soni:** Contributed to the literature review, compiled a list of data sources, implemented SVD, enhanced the codebase, and conducted project documentation and analysis

5. References

1. Tapendu, I. (n.d.). *Lungs Segmentation using U-Net Architecture*. Retrieved November 24, 2024, from https://www.kaggle.com/code/iamtapendu/lungs-segmentation-using-u-net-architecture/notebook
2. Li, X. (n.d.). *Lecture 4: Singular Value Decomposition*. University of Alabama at Birmingham. Retrieved November 24, 2024, from https://uab.instructure.com/courses/1637340/files/folder/lecture?preview=76461947
3. Sahu, P. K., & Bhowmick, M. (2013). *Medical Image Segmentation: A Review*. Retrieved November 24, 2024, from https://d1wqtxts1xzle7.cloudfront.net.
4. Ronneberger, O., Fischer, P., & Brox, T. (2015). *U-Net: Convolutional Networks for Biomedical Image Segmentation*. arXiv preprint arXiv:1505.04597. Retrieved November 24, 2024, from https://arxiv.org/pdf/1505.04597.
5. DMICZ. (n.d.). *SVD Image Compression*. Retrieved November 24, 2024, from https://dmicz.github.io/machine-learning/svd-image-compression/.

6. J. Bertels *et al.*, "Optimizing the Dice Score and Jaccard Index for Medical Image Segmentation: Theory & Practice," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2019*, Lecture Notes in Computer Science, vol. 11765, D. Shen *et al.*, Eds. Cham: Springer, 2019, pp. 92–100, doi: 10.1007/978-3-030-32251-9_11