Denoising Optical Coherence Tomography Images via Deep Learning

Armaan Lalani – Eric Liang – Predrag Muratovic – Young Seok Seo

Introduction

Breast cancer is a global problem; it is the most common cancer in more than 100 countries, including Canada, according to a study in 2020 by the World Health Organization [1]. With the help of advancing technology and medicine, patients with early-stage breast cancer can undergo breast conserving surgery (BCS) which selectively removes malignant tissue with a tumor-free margin. Our client, Perimeter Medical Imaging (PerimeterMed), uses artificial intelligence to help surgeons determine whether all of the cancer tissue was removed in the BCS. Following an operation, the surgeon collects thin tissue slices to be examined using Optical Coherence Tomography (OCT), which provides high-resolution images of subsurface tissue in real-time [2]. These images are analyzed using our client's Convolutional Neural Network (Image Assist AI) which detects any residual malignant tissue. Traditionally, the results of a BCS are validated using a postoperative histology tissue assessment, which can take up to 7 days; our client's technology allows surgeons to confirm the results of a BCS within minutes in the operating room to mitigate follow-up operations.

The primary problem presented by PerimeterMed is the timeframe required to conduct a margin assessment. The margin assessment is a process used to determine whether or not the surgeon was successful in removing the tumor during BCS utilizing OCT images. Conducting an accurate margin assessment requires the use of denoised OCT images, which is currency accomplished by using two signals to reduce noise, taking approximately two to three minutes. Noise can be defined as additional artifacts that appear as a grainy structure. Multiple scans are necessary for a margin assessment, which leads to a process timeframe of roughly 15 minutes (multiple scans and their associated denoising). This timeframe is problematic for two primary reasons: the first being a reduction of throughput in the number of patients a surgeon is able to see and the second being potentially keeping patients under unnecessary sedation during the margin assessment if the assessment reveals no malignant tissue. As a result, the solution to this problem involves the use of a deep learning model to reduce the levels of noise in individual signals, thus reducing the total amount of time needed to conduct a margin assessment.

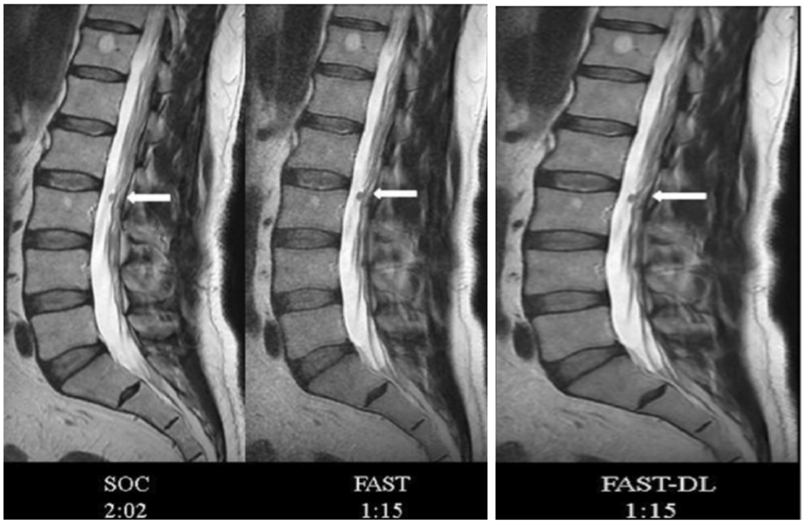


Figure 1: A representation of various methods of reducing noise. From left to right: 2 signal averages approach (current PerimeterMed implementation), 1 signal average approach, 1 signal deep learning approach (project goal)

At the onset of the project, there were a number of requirements that were discussed with PerimeterMed. Since our team was primarily focused on developing a model, our requirements were primarily system related, which can be further broken down into functional and nonfunctional requirements. Our primary functional requirement was a deep learning model that removes a certain level of noise from OCT scans. We initially performed research on two primary models to accomplish the denoising problem, UNets and Conditional Generative Adversarial Networks, both of which are discussed in detail in the Methods section. Our primary nonfunctional requirement was any improvement in the image quality through the reduction of noise. As a group, we were tasked with researching various metrics to quantitatively evaluate how well our model achieved this goal, therefore, there was no initial goal for the value we hoped to achieve.

Data

This section describes the following components relevant to data preparation: description, cleaning/preprocessing, analysis, and augmentation.

The primary data source for the project is PerimeterMed's internal dataset that contains 1692 grayscale OCT scans of breast tissue specimens in U16 (i.e. 16-bit) format. The total data size is 2.68 GB and was provided to the team in a storage bucket on Google Cloud Platform (GCP). To access the data, the team downloaded the data folder from GCP to the local environment in the Virtual Machine that PerimeterMed had provided us access to.

The data cleaning process consisted of two steps: converting the raw image to arrays of pixel values that can be read and manipulated accordingly and extracting Regions of Interest (ROI) from the images. For image conversion, the company provided the team with appropriate Python functions that can be utilized to read a grayscale U16 image as a 2D array of corresponding pixel values. Each image is of size 1024 by 2688 pixels after conversion. Furthermore, the raw images contain regions other than the breast tissue itself. Specifically, the top of the image is empty space between the camera and the glass surface used to contain the specimen, and the bottom of the image is a region with large light attenuation and contains mostly noise. The team pre-processed the images based on the physical parameters of the aforementioned two regions (i.e. 5mm and 27.6mm) and information about the number of pixels per millimeter (i.e. 15.2 px/mm) given by PerimeterMed. Figure 2 below shows a raw and preprocessed image.

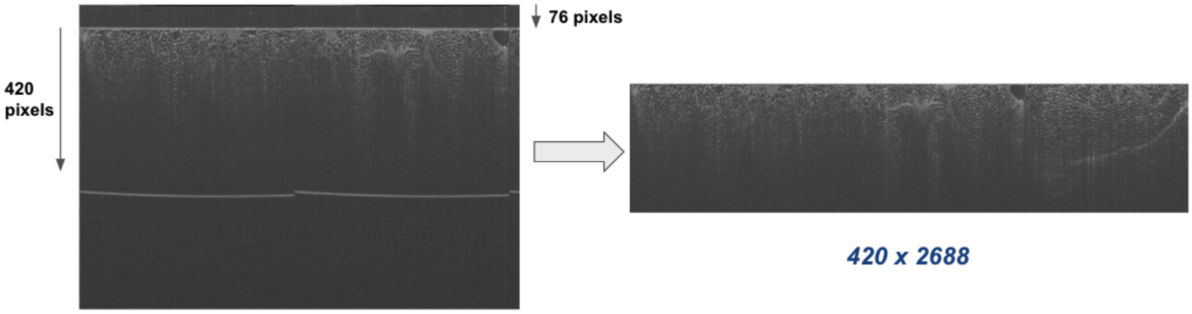


Figure 2: Image preprocessing visualization. Left to right: raw image (1024 by 2688), preprocessed image (420 by 2688)

Next, the team proceeded to analyze the preprocessed OCT images. The primary objective of the analysis was to explore the noise distribution in the images for augmentation purposes and develop a sensible quantitative metric that could be used to evaluate model performance in the later stages of the project. In medical images, the main source of noise is due to the physical characteristics of light, such as its corpuscular nature. Specifically, as photons arrive irregularly on the photosites, two adjacent pixels which are supposed to have similar pixel values can end up having different photon counts. As a result, we can quantitatively measure such noise by computing the standard deviation of pixel values. On the contrary, the signal is defined as the unique characteristics of an image without any sources of noise. It is measured as the mean pixel values due to the effect of light attenuation as depth increases. Hence, the more structurally visible an image appears, the stronger the signal is. Based on this analysis, a quantitative metric that measures the image quality is the Signal-To-Noise Ratio (SNR), which is simply the mean of the pixel values divided by the standard deviation. The larger the SNR value is, the higher the image quality [3]. However, in all the preprocessed OCT images, there is a clear distinction between the top and bottom half. Specifically, the top half contains distinguishable tissue structures and boundaries, while the bottom half is uniformly gray and full of noise due to light attenuation (i.e, the light can not reach that deep under the tissue). Figure 3 shows a sample comparison between the top and bottom half of an OCT image.

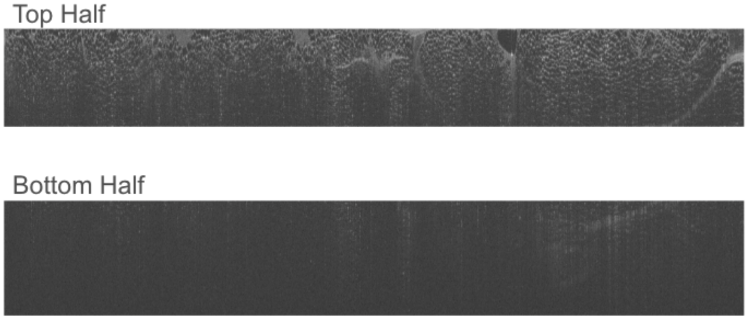


Figure 3: Top vs. Bottom half of an OCT image (i.e, signal vs noise)

In essence, to calculate SNR on the OCT images, the signal must be derived from a homogenous ROI with a high level of structural integrity in the top half while the noise can only be calculated from a noisy strip in the bottom half. This way, the signal strength is not dependent on the effect of light attenuation and the noise does not encapsulate the change in pixel values from the natural boundaries between tissue structures [Appendix C]. During the research phase, other image evaluation metrics such as Edge Preservation Index (EPI) and Mean Squared Error (MSE) were also explored. However, upon reviews with PerimeterMed, we discovered different constraints regarding these metrics (i.e, major constraint being the lack of ground truth / gold-standard image) and agreed on SNR being the most representative one [Appendix D].

Lastly, for the model to properly denoise the given OCT images, it first needs to learn the characteristics of the existing noise profiles. Thus, there needs to be additional noise added to these images and the resulting noisy images are the real input to the model while the added noise, also known as the residual image, is used to optimize the output. PerimterMed had previously conducted noise profile research and discovered that the predominant noise profile in the OCT images is speckle noise, a multiplicative granular noise texture as a result of interference among wavefronts in coherent imaging systems [4]. It is defined as the following:

$$Speckle\ Noise\ =\ Image\ Pixels\ \times\ Rand$$

where Rand is a normally distributed random matrix with mean 0 and small variance (0.05 in the context of the study) of the same shape as the image pixel matrix. However, due to the aforementioned non-uniform signal versus noise distribution in the OCT images, speckle noise needs to be scaled accordingly before being added to the original images. Specifically, there should be more noise added to the top of the image where there is mostly signal, and less added to the bottom where there is already a large amount of intrinsic noise. To accomplish this, the team utilized a linearly decreasing function as follows:

$$\boldsymbol{y\ =\ k\ \times (1 - \frac{1}{420} * d)}$$

- y → % of speckle noise
- k → slope of the linearly decreasing function (i.e, how quickly the % of speckle noise decreases as depth increases)
- d → depth of the image (i.e, bounded between 0 and 420)

By varying the slope of the function (i.e. k), the same original image can be augmented into multiple noisy and residual images used as inputs and outputs to the model. Figure 4 shows a comparison amongst original, noisy, and residual images.

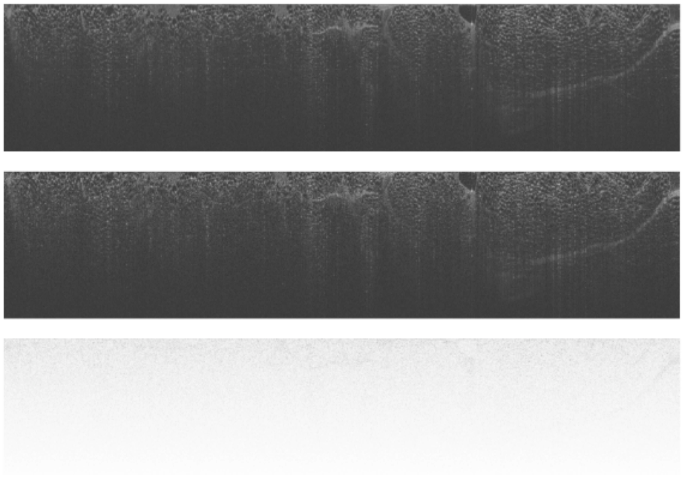


Figure 4: From top to bottom: original, noisy (speckle noise added), and residual image. For visualization purposes, the residual image is the complement of the absolute pixel value difference between the original and noisy image. The smooth noise boundary can be visualized in the residual image.

In order to obtain more training data and reduce computation time, the team used 5 distinct k values {1,1.5,2,2.5,3} for every original image and split every noisy image of size 420 by 2688 into 7 square patches of size 420 by 420 followed by resizing to 224 by 224 pixel images. In total, the number of input & residual images is:

$$1692\ original\ images\ \times\ 5\ noisy\ images/original\ image\ \times\ 7\ patches/noisy\ image\ =\ 59,220.$$

Methods

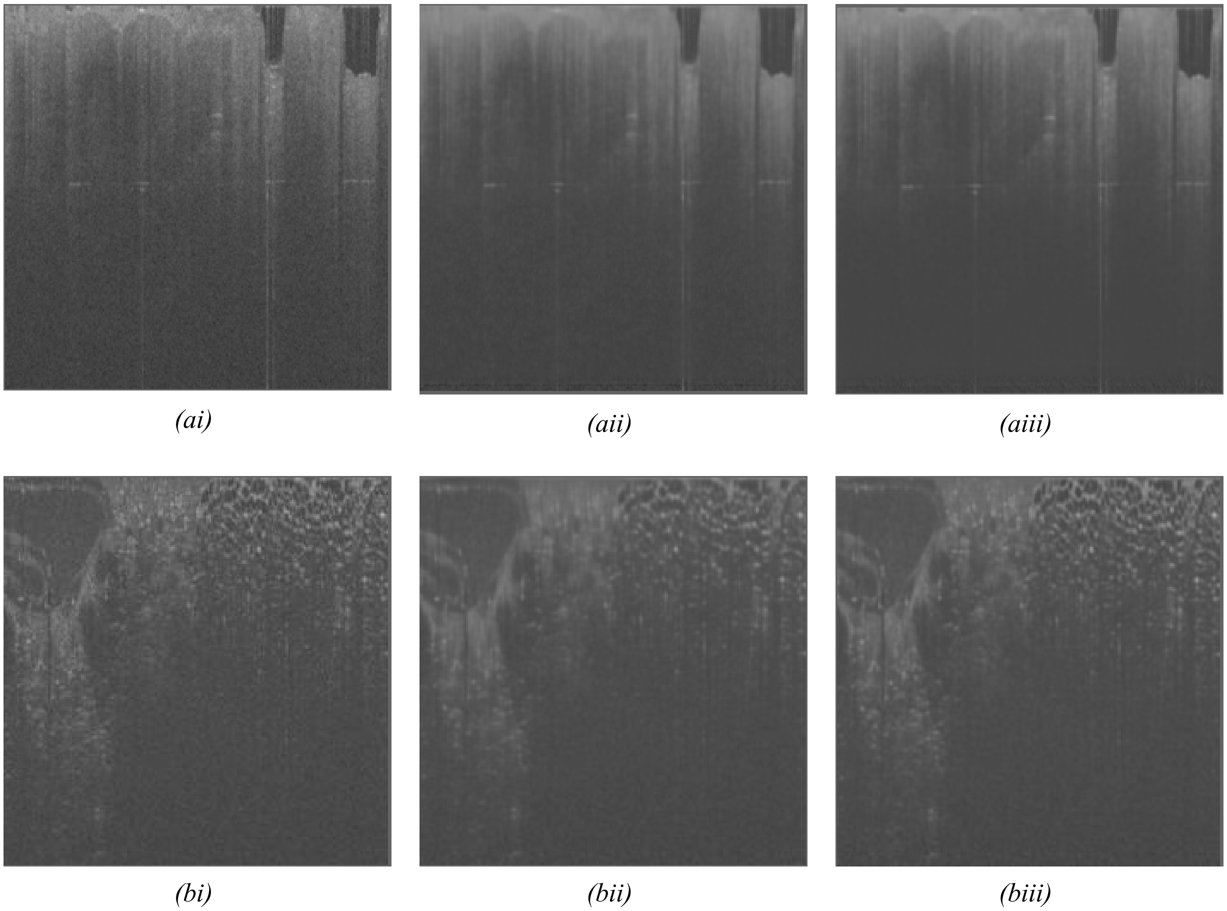
The team and client decided to approach the denoising problem with two different model architectures: a CNN architecture commonly used in denoising tasks, called DnCNN, and a variant of a Convolutional Neural Network (CNN), called UNet [Appendix E]. Both models were adapted from the github repository 'Image-Denoising-with-Deep-CNNs' [9] which provided the model architecture for both the DnCNN and UNet models. In order to appropriately utilize these models with OCT images, there were several minor alterations made to the code as well as redefining the PyTorch DataLoader class. All of the original files (224 by 224 pixel images) and their five associated noisy images (224 by 224 pixel images) were saved locally on the VM and their file paths were passed through the DataLoader class as inputs to the two primary models.

Note that the generic CNN and UNet models differ from CNNs used in categorical learning since the model output is an image instead of a categorical mapping; hence, the two models' goal was to learn the added speckle noise distribution and remove said noise. Models were to be trained by minimizing the difference between model output and the original scan. The primary motivation for using CNN-based architecture is because the input data was composed of 2D images. Based on previous research, CNNs perform well on 2D data with spatial relationships and are popular in image-based learning tasks due to their ability to detect important structural features. Therefore, the DnCNN is a simple, yet well-applicable architecture to perform denoising. The DnCNN is a CNN where the dimension of the latent space maintains the dimension of the input by not using pooling operations after layer activations [Appendix A1]. The benefit of this approach is that it is simple to implement and the depth of the model is capable of determining image patterns [5].

Although the DnCNN is a model capable of image denoising, it has been noted in literature that they have reached a bottleneck in denoising and cannot improve efficiency with better results [6]. The first half of forward propagation in UNet models begins similar to traditional CNNs, with convolutions/activations followed by a pooling layer (contractions); however, in the second half of the forward propagation, there are a series of 'up-convolutions' that up-sample the hidden layers via dilated convolution filters. The result of this up-convolution is combined (added) with the feature maps from the first half of the propagation when the dimension of the feature maps (from first half) are the same as the result [Appendix A2][7]. The addition of the previous feature maps is to ensure minimal structural loss because it adds patterns found during contractions. The UNet has similar benefits to the DnCNN; however, the UNet has less parameters to train and requires less images to train, effectively reducing training time and computational demand [7].

Results

Using the preprocessed and augmented data described in the *Data* section, the team was able to train and test the two models described in the *Methods* section, the DnCNN and UNet [Appendix B][Appendix F]. To illustrate the performance of the models, the team gathered a set of ten images (i.e. OCT scans) and denoised them via the aforementioned models: results are presented in Figure 5. Note, the examples in Figure 5 are test images and did not have added noise. Furthermore, given the time constraints and discussion with PerimeterMed, a sample size of ten was used because computing SNR values is quite time consuming because it requires manually searching for regions of interest to perform the metric.



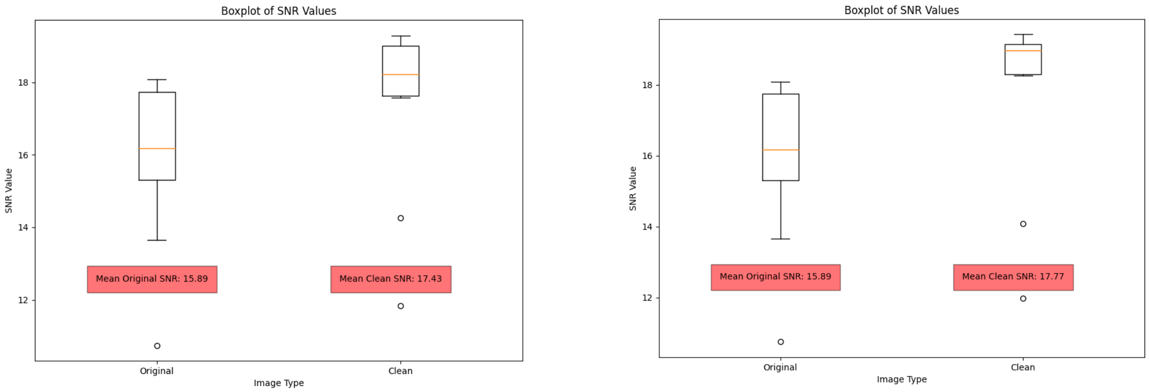
(ai) *(aii)* *(aiii)*

(bi) *(bii)* *(biii)*

Figure 5: Two examples ((a) and (b)) of scans used for denoising. (i) is the original scan with intrinsic noise; (ii) and (iii) are scans denoised by the DnCNN and UNet models, respectively.

One can observe that both of the models are able to remove a substantial amount of noise from the original image. For instance, the ‘granular-ness’ of the original images (Figure 5.i) has been removed by both models (Figure 5.ii/iii) and present smoother images without loss of structure. Although both models remove noise without excess blurring, it is visibly noticeable that the UNet does a better job at removing noise while also minimizing blur. This can be seen best by observing the top third of example Figure 5b: The UNet preserves the honeycomb-like pattern present in the scan to a greater extent than the DnCNN model.

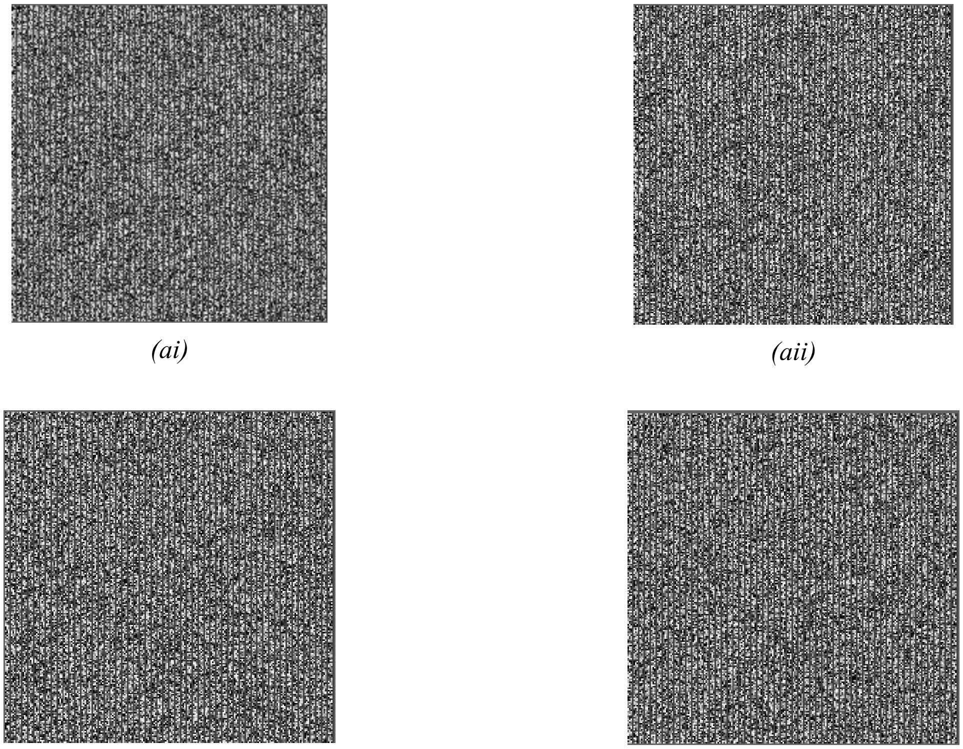
To better quantify the degree of noise removed and signal (tissue) kept, the team measured the SNR values on the ten samples [Appendix C] before and after denoising. Using this SNR data, the team generated boxplots to visualize the SNR distribution of the original scans and denoised scans (by both models) in Figure 6. Confirming visual observations presented by Figure 5, the boxplots show that the distribution of SNR values has been shifted in a positive direction. The median (orange tick) and mean of the SNR values present in denoised images by both the UNet and DnCNN are higher than the median and mean found in original images. The mean SNR value had a percent increase of 9.7% and 11.8% after denoising the images via the DnCNN and UNet models, respectively.



(a) *(b)*

Figure 6: Boxplots of Signal-to-Noise Ratio metric obtained from a sample of 10 original-cleaned image sets. (a) is for DnCNN and (b) is for UNet. Mean SNR values reported in red blocks.

Additionally, to determine if the denoising done by the models was removing signal, the team obtained residual images between the original and denoised scan (see Figure 7). The residual image will display excess signal removal when there is an evident pattern from the original image present in it. The residuals in Figure 7 were multiplied by 10 to better emphasize any underlying patterns that may be present.



(ai) *(aii)*

(bi) *(bii)*

Figure 7: 10X Residual (difference of clean and denoised) scans produced after denoising. Examples (a) and (b)

are the same examples ((a) & (b)) in Figure 5. (i) and (ii) represent residuals using the DnCNN and UNet models, respectively.

Discussion

The goal of this project as defined by PerimeterMed and the team was to reduce scan time for detecting malignant tissue during breast cancer surgery. More specifically, the initial goal was to develop a deep learning-based denoising algorithm that eliminates the need for two signals per OCT scan. Through careful analysis and comparison of various methods, the team was able to present PerimeterMed with two neural networks that are proficient in removing noise from tissue images while preserving important details.

The evaluation metrics and sample images shown in *Results* reveals that, of the two networks, the UNet model performs better on the dataset provided by the client. The SNR measurements on output images from UNet shows both higher mean value as well as narrower distribution (see Figure 6). Although the difference in mean SNR value from the two networks is not very large (~2%), a key requirement set out by PerimeterMed was that the denoising algorithm should not remove any of the signal in the images; upon examination of the model output with the client, it was noted that the UNet model more consistently preserves details like tissue striation in the region of interest. An additional requirement defined by PerimeterMed was that the introduction of a denoising process should not increase the total scan time. Both UNet and DnCNN have negligible inference time per image (less than 0.1 seconds), and provide a significant improvement over the current process. Since the UNet model is able to produce better results than DnCNN without adding any overhead, the team and the client have converged on UNet as the final model of choice.

Performance improvements shown by UNet may be due to reduced dimensionality compared to DnCNN. The noise profile displayed in OCT images is not overly complex; by first extracting features through contractions then adding information obtained at each layer back to the up-convolution layers, the UNet architecture is able to retain signal from the original image with fewer parameters to optimize. The presence of pooling layers to condense the image into feature maps may also provide benefits over DnCNN, where the dimensions do not change throughout the network. One aspect of the results we had to analyze was the apparent grid pattern shown in the residuals in Figure 7 due to potentially removing significant levels of signal. Based on our discussion with PerimeterMed, it was concluded that the grid-like structure appears on the original images most likely created from artifacts from the secondary glass line located 4mm below the scan line. The glass line is primarily responsible for the horizontal lines in the grid pattern while the vertical lines are most likely attributed to intrinsic image characteristics. This behaviour, however, is something that will have to be analyzed further to arrive at a conclusive answer. The model learns these patterns and removes them since they are a repetitive pattern.

A limitation of the current UNet architecture, however, is that the output dimension is smaller than that of the original image. Due to memory restrictions on the GCP instances provided by the client, the team decided to down-sample input images from 420x420 to 224x224. Simply up-scaling the output images back to the original dimensions would lead to a "blurring" effect, as shown in Figure 8. Although this issue does not relate to the model's performance, it needs to be addressed before the model can be directly incorporated into the client's pipeline.

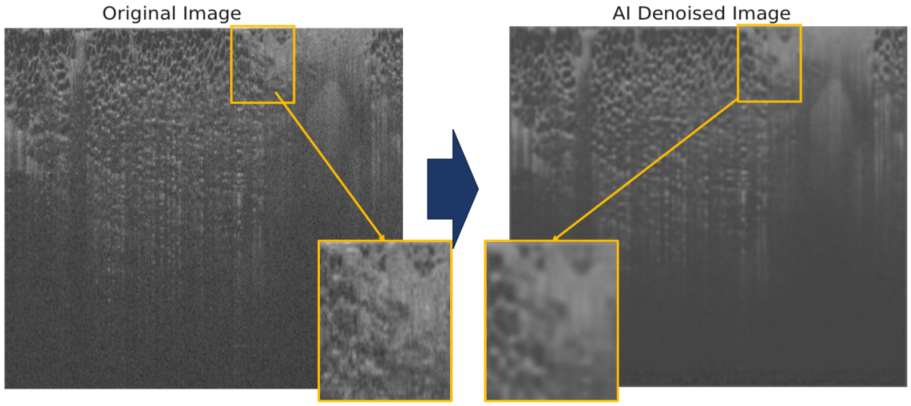


Figure 8: Original image vs. AI denoised image that was resized to match the original dimensions.

It is important to note that many of the requirements and constraints related to the project are especially emphasized because of the clinical nature of the data and applications. The team and client prioritized correctness and sensitivity to ensure that the implemented algorithm and the overall denoising process "do no harm". Much of the barriers faced in the data collection and processing phase were from a lack of readily available public data, since the client primarily works with real tissue images collected from surgery. The images provided to the team were stripped of any information or metadata to eliminate possible ethical or security concerns.

Implementation

The team developed and trained two models, a DnCNN and a UNet, in Python via the PyTorch library and also wrote several functions to preprocess and augment data. Using the UNet model, the client will need to implement denoising scans into their existing pipeline. The current plan would be to complete and gather OCT scans, denoise them using a final model, and run the client's 'Image Assist AI' that detects suspicious regions of interest in scans before presenting them to a physician.

Before the client can begin to use the model in production, there are some risks that must be addressed: There are two technical and one clinical risk present. Thus, the timeline for future implementation consists of mitigating these risks. First, there is a clinical risk in removing too much signal from original scans. The model must be extensively tested on positive (unhealthy tissue) and negative (healthy) scans. For instance, denoised positive scans must be checked to ensure suspicious regions remain easily detectable, and denoised negative scans must be checked to ensure the model is not adding new structures. Second, a technical risk that the client plans to mitigate is blurring in denoised scans. Recall that scans are initially segmented and downsampled before they can be used with the model, then upsampled to return to their initial size. Therefore, blurring arises from upsampling because of structure loss during downsampling; thus, upsampling cannot recover lost structure during downsampling. The client mentioned they would train a new version of the model with a larger input size to mitigate said risk. The other technical risk lies in excess computational time. The client would have to run both the denoising and image assist tasks on a single GPU, effectively increasing computation time. This risk may be mitigated by strengthening hardware available; for instance, using two or more GPUs with higher computational power.

Conclusion and Future Directions

In summary, the high-level goal of the project is to reduce PerimeterMed's 15-minute margin assessment time by developing a deep learning network that reduces the amount of noise in individual input OCT scans. The team has successfully designed and implemented a pipeline that takes in raw OCT images, denoises it with the aforementioned UNet model, and evaluates the image quality with the SNR metric. Furthermore, the denoised image has shown a 11.8% improvement in quality measured by SNR compared to the raw image. Specifically, the resulting images have shown to retain similar amounts of signal in homogenous regions with visible tissue structures, while at the same time having less noise in regions with large light attenuation and interference.

There are 2 major technical and clinical risks going forward that need to be addressed. First, as mentioned previously, there is blurring in the denoised image due to the downsampling process. In the future, it is recommended that a new model should be trained with a higher resolution matrix size (i.e. retain the 420x420 patches for the current dataset) to absolve the challenge. Second, due to the nature of the denoising network, it is inevitable that a small portion of the signal is reduced and even removed. Thus, the model needs to be tested extensively on positive and negative scans to ensure that there is a sufficient amount of signal left in the denoised image to maintain the initial label (unhealthy or healthy). This involves labeling the dataset and evaluating the developed model with metrics such as precision and recall.

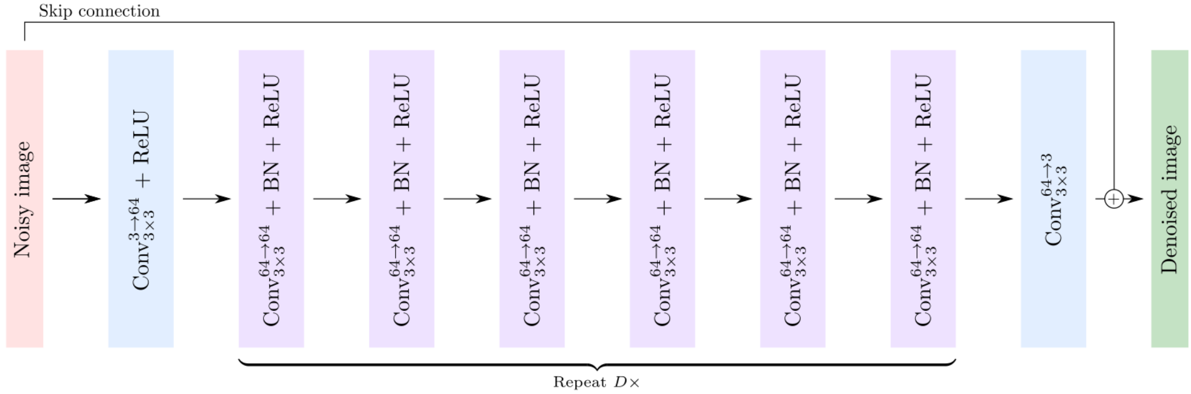
Attribution Table

Member	*Attribution*
Armaan Lalani	Data augmentation, model development, model training, obtaining residuals, test image evaluation (determining ROI and boxplots) Final Report: Introduction Final Presentation: Slides
Predrag Muratovic	Data augmentation, model development, model training, obtaining residuals. Final Report: Methods, Results, and Implementation.
Rui Liang	Data preprocessing, evaluation metric development, noise addition algorithm implementation, data augmentation Final Report: Data, Conclusion and Future Directions
Young Seok Seo	Data preprocessing and augmentation, evaluation metric research, DnCNN implementation and validation Final Report: Discussion

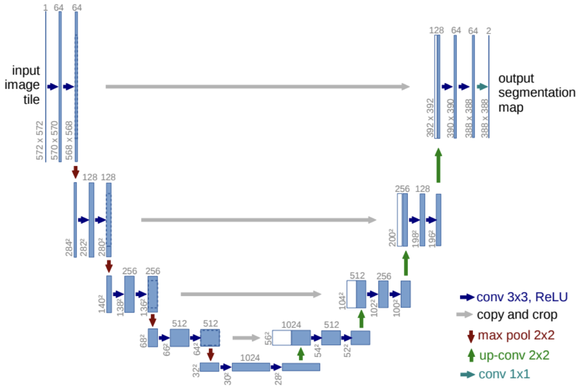
Appendix

Appendix A - Visual Depiction of Model Architectures

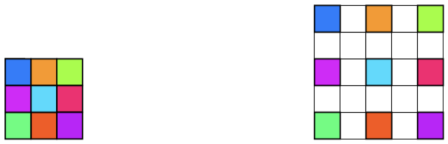
A1 - Visual Representation of DnCNN


Skip connection
Noisy image
$\mathrm{Conv}_{3\times3}^{3\to64}$ + ReLU
$\mathrm{Conv}_{3\times3}^{64\to64}$ + BN + ReLU
$\mathrm{Conv}_{3\times3}^{64\to64}$ + BN + ReLU
$\mathrm{Conv}_{3\times3}^{64\to64}$ + BN + ReLU
$\mathrm{Conv}_{3\times3}^{64\to64}$ + BN + ReLU
$\mathrm{Conv}_{3\times3}^{64\to64}$ + BN + ReLU
$\mathrm{Conv}_{3\times3}^{64\to64}$ + BN + ReLU
$\mathrm{Conv}_{3\times3}^{64\to3}$
Denoised image
Repeat $D\times$

A2 - Visual Representation of UNet



(i) Forward Propagation of UNet Model [7]



A standard 3 × 3 convolutution kernel (`dilation=1`).

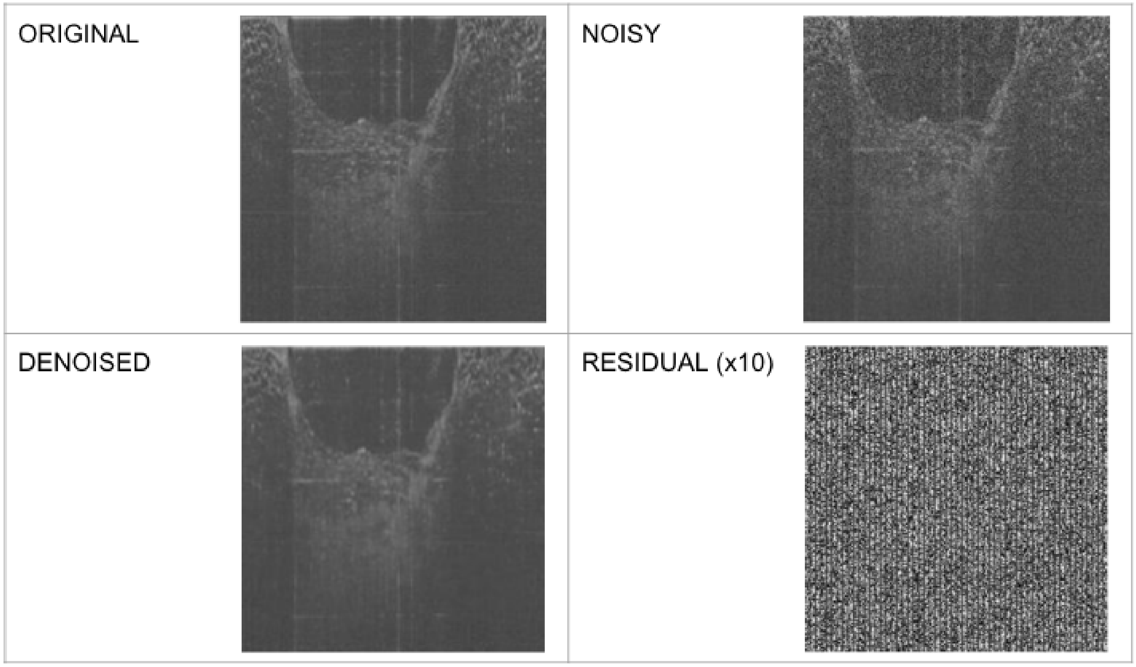
A 3 × 3 dilated kernel with `dilation=2`. In fact, a 5 × 5 kernel with holes of size 1.

(ii) Dilated Up-Convolution

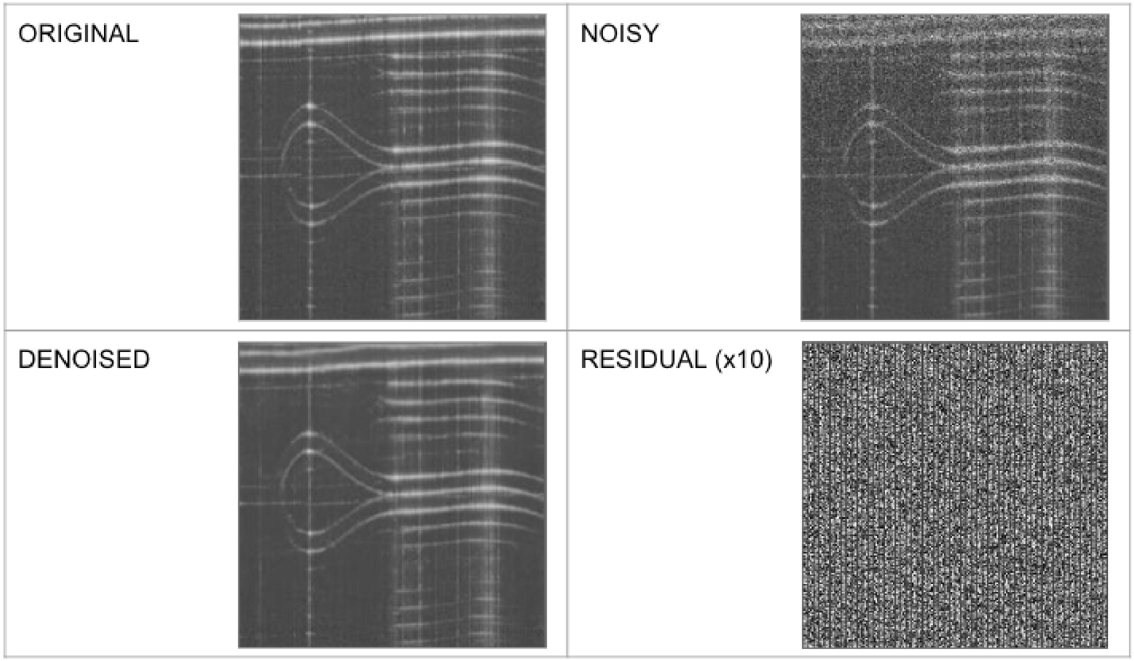
Appendix B - Sample Images from Training Pipeline

B1 - Images from DnCNN Training

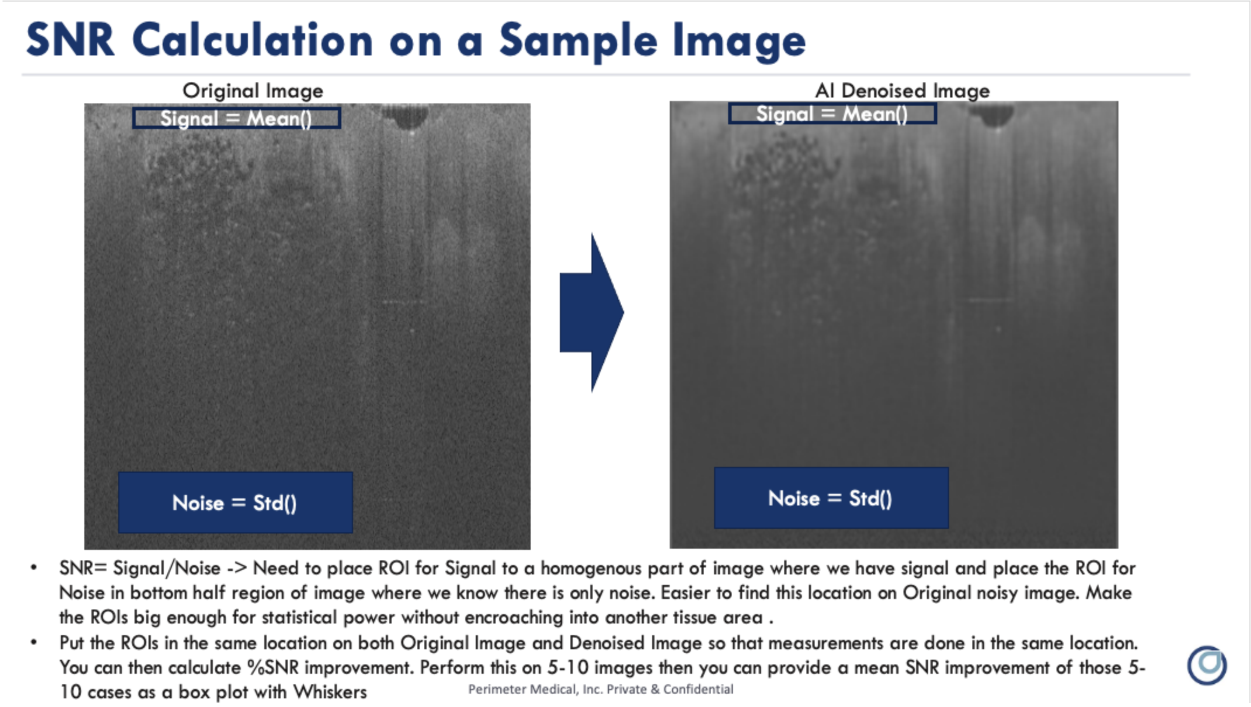
(from top left to bottom right: the original image segmented from the OCT scan, the image after the gradient speckle noise was added, the denoised image or model output, the residual which was calculated by taking the absolute difference of the denoised image and the original image and multiplying by 10 for visual clarity)



B2 - Images from UNet Training



Appendix C - SNR Calculation on a Sample Image



SNR Calculation on a Sample Image

- SNR= Signal/Noise -> Need to place ROI for Signal to a homogenous part of image where we have signal and place the ROI for Noise in bottom half region of image where we know there is only noise. Easier to find this location on Original noisy image. Make the ROIs big enough for statistical power without encroaching into another tissue area .
- Put the ROIs in the same location on both Original Image and Denoised Image so that measurements are done in the same location. You can then calculate %SNR improvement. Perform this on 5-10 images then you can provide a mean SNR improvement of those 5-10 cases as a box plot with Whiskers

Perimeter Medical, Inc. Private & Confidential

Appendix D - Other Image Quality Evaluation Metrics

The following metrics require a reference / denoised image

- **Edge Preservation Index (EPI)**: evaluates performance of denoising algorithm by comparing edges in reference and denoised image
- **Mean Squared Error (MSE)**: measures difference between reference and denoised image
- **Peak Signal-to-Noise Ratio (PSNR)**: measures quality of image reconstruction by comparing maximum power of signal with MSE of noise between reference and denoised image

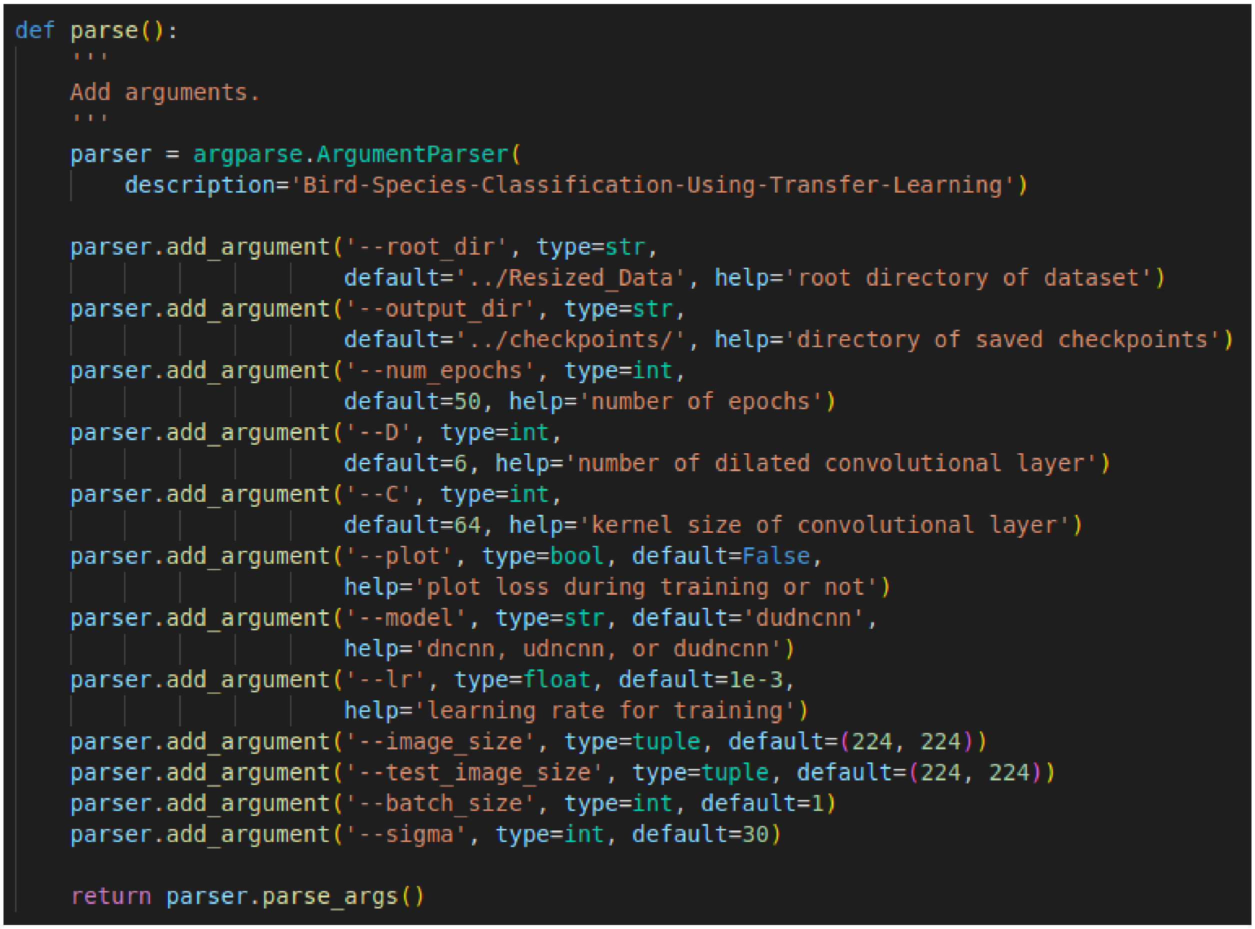
Appendix E - Model Architecture Change

Initially, alongside the UNet model, the team and client considered developing a conditional Generative Adversarial Network (cGAN) for denoising. However, because the team and client realized that the cGAN had an immense number of parameters to train effectively, about 166 million, the approach was abandoned. For example, to train the cGAN model for the problem at hand, one epoch during training took forty-five minutes while only using a tenth of the dataset; thus, training the cGAN was deemed to be impractical due to the team's and client's time constraints for the project. Furthermore, because a pre-trained model was available, the team considered freezing[1] certain layers of the model to optimize training time, but upon discussion with the client, the freezing was not advised. Instead, the team and client decided to

[1] Freezing layers in deep learning refers to maintaining the values for learned parameters in certain layers ('freezing') while training/optimizing parameters in the remaining layers. [8]

use a generic CNN, denoted as DnCNN, as a baseline for the UNet model to compare performances.

Appendix F - Model Parameters



```
def parse():
    '''
    Add arguments.
    '''
    parser = argparse.ArgumentParser(
        description='Bird-Species-Classification-Using-Transfer-Learning')

    parser.add_argument('--root_dir', type=str,
                        default='../Resized_Data', help='root directory of dataset')
    parser.add_argument('--output_dir', type=str,
                        default='../checkpoints/', help='directory of saved checkpoints')
    parser.add_argument('--num_epochs', type=int,
                        default=50, help='number of epochs')
    parser.add_argument('--D', type=int,
                        default=6, help='number of dilated convolutional layer')
    parser.add_argument('--C', type=int,
                        default=64, help='kernel size of convolutional layer')
    parser.add_argument('--plot', type=bool, default=False,
                        help='plot loss during training or not')
    parser.add_argument('--model', type=str, default='dudncnn',
                        help='dncnn, udncnn, or dudncnn')
    parser.add_argument('--lr', type=float, default=1e-3,
                        help='learning rate for training')
    parser.add_argument('--image_size', type=tuple, default=(224, 224))
    parser.add_argument('--test_image_size', type=tuple, default=(224, 224))
    parser.add_argument('--batch_size', type=int, default=1)
    parser.add_argument('--sigma', type=int, default=30)

    return parser.parse_args()
```

References

[1] "Cancer Today," *Global Cancer Observatory*. [Online]. Available: https://gco.iarc.fr/today/online-analysis-map?mode=cancer. [Accessed: 04-Oct-2022].

[2] F. T. Nguyen, A. M. Zysk, E. J. Chaney, J. G. Kotynek, U. J. Oliphant, F. J. Bellafiore, K. M. Rowland, P. A. Johnson, and S. A. Boppart, "Intraoperative evaluation of breast tumor margins with optical coherence tomography," *Cancer Research*, vol. 69, no. 22, pp. 8790–8796, 2009.

[3] Y. Fan, Y. Li, T. Gao, and X. Tang, "Noise reduction of OCT images based on the external patch prior guided internal clustering and morphological analysis," *MDPI*, 03-Aug-2022. [Online]. Available: https://www.mdpi.com/2304-6732/9/8/543/htm. [Accessed: 30-Nov-2022].

[4] K. Bansal and A. Kaur, "A review on Speckle noise reduction techniques ," *Research Gate*, Jan-2014. [Online]. Available: https://www.researchgate.net/publication/274546647_A_Review_on_Speckle_Noise_Reduction_Techniques. [Accessed: 30-Nov-2022].

[5] Zhang, Kai, et al. "Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising." IEEE Transactions on Image Processing, vol. 26, no. 7, July 2017, pp. 3142–3155, 10.1109/tip.2017.2662206.

[6] F. Jia, W. H. Wong, and T. Zeng, "DDUNet: Dense Dense U-Net with Applications in Image Denoising," IEEE Xplore, Oct. 01, 2021. https://ieeexplore.ieee.org/document/9607593 (accessed Nov. 30, 2022).

[7] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," arXiv.org, May 18, 2015. https://arxiv.org/abs/1505.04597 (accessed Nov. 2022).

[8] X. Xiao, T. Bamunu Mudiyanselage, C. Ji, J. Hu and Y. Pan, "Fast Deep Learning Training through Intelligently Freezing Layers," 2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), 2019, pp. 1225-1232, doi: 10.1109/iThings/GreenCom/CPSCom/SmartData.2019.00205.

[9] Cheng (2019) Image-Denoising-with-Deep-CNNs [Source code]. https://github.com/lychengrex/Image-Denoising-with-Deep-CNNs