	# Hacettepe University Computer Engineering Department **BBM479 End of Term Development Report**

Project Details

Title	Pressure Ulcers
Supervisor	Cemil Zalluhoğlu

Group Members

	Full Name	**Student ID**
1	Mert Tazeoğlu	21946606
2	Asım Ateş	21827076
3	Yusuf Efe Kalaycı	21827517
4		

Current State (/ 40 Points)

Explain the current state of the project. By giving references to the plan proposed at the beginning of the project, explain what is achieved so far. Provide details of what has been done by explaining the technology and methodology used. Is the current state of the project inline with the plan? If you are behind the schedule, explain in detail the reasons.

Based on the scheduled activities outlined in the project plan, it is evident that approximately 70% of the project has been accomplished. Our initial semester objective was to achieve a substantial portion of the project before the mid-term break, sustain momentum during the break, and conclude nearly the entire project at the commencement of the second semester. A recap of the undertaken activities within the project's progression can be summarized as follows:

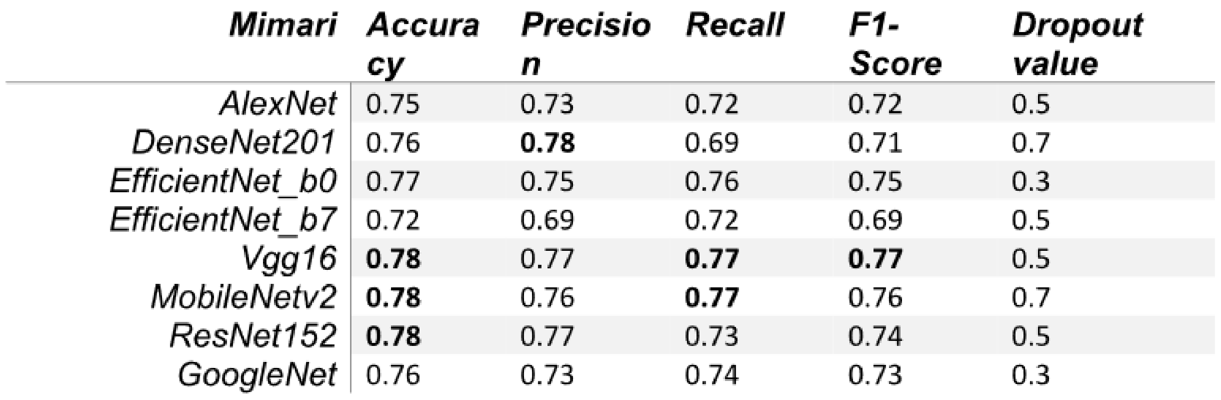
1. Creating Masks From JSON Files

Initially, in the context of medical image segmentation, we conducted image parsing along with JSON files containing coordinates of the target regions within the images. Subsequently, we processed the affected areas in the images, generating mask images. In this scenario, considering a segmentation function denoted as $y = f(x)$, the original image utilized to identify the target is represented by x, and the mask images crafted at this phase serve as the desired value, i.e., y.

These operations were specifically performed on the Ventura dataset, employing Python's image processing libraries. As part of an optimization technique, mask images were created using both fill and bounding box methods, which will be discussed in greater detail later in this report.

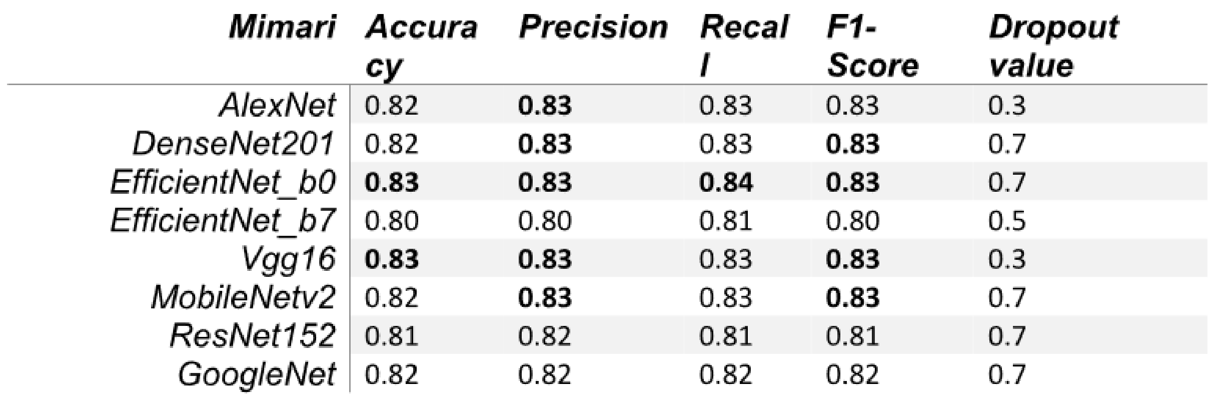
2. Classification

We conducted an experiment into deep learning methodologies to identify the most suitable approach. Subsequently, we experimented with various models, adjusting key parameters such as schedule, dropout, learning rate, and others. The outcomes of our tests are as follows:



Mimari	*Accuracy*	*Precision*	*Recall*	*F1-Score*	*Dropout value*
AlexNet	0.75	0.73	0.72	0.72	0.5
DenseNet201	0.76	**0.78**	0.69	0.71	0.7
EfficientNet_b0	0.77	0.75	0.76	0.75	0.3
EfficientNet_b7	0.72	0.69	0.72	0.69	0.5
Vgg16	**0.78**	0.77	**0.77**	**0.77**	0.5
MobileNetv2	**0.78**	0.76	**0.77**	0.76	0.7
ResNet152	**0.78**	0.77	0.73	0.74	0.5
GoogleNet	0.76	0.73	0.74	0.73	0.3

Figure-1: Best Classification Results For Ventura Dataset



Mimari	*Accura cy*	*Precision*	*Recal l*	*F1-Score*	*Dropout value*
AlexNet	0.82	**0.83**	0.83	0.83	0.3
DenseNet201	0.82	**0.83**	0.83	**0.83**	0.7
EfficientNet_b0	**0.83**	**0.83**	**0.84**	**0.83**	0.7
EfficientNet_b7	0.80	0.80	0.81	0.80	0.5
Vgg16	**0.83**	**0.83**	0.83	**0.83**	0.3
MobileNetv2	0.82	**0.83**	0.83	**0.83**	0.7
ResNet152	0.81	0.82	0.81	0.81	0.7
GoogleNet	0.82	0.82	0.82	0.82	0.7

Figure-2: Best Classification Results For Elazığ Dataset

Ultimately, based on our assessments, VGG16 emerged as the optimal deep learning model. Subsequently, we endeavored to enhance accuracy by optimizing the pretrained (best) VGG16 model, incorporating the use of RoI, a topic that will be elaborated on in part 5.

3. **Medical Image Segmentation**

3.1 - U-Net++

To select the CNN model for our project, we carried out a literature review encompassing various models. Following this, we conducted segmentation tests on models deemed suitable for our project using different datasets, including those for skin cancer and eye diseases. By comparing results using our metrics, we were able to eliminate some models. Subsequently, the remaining two models, DoubleUnet and Unet++, were tested on our dataset. Based on the outcomes, we concluded that the DoubleUnet model would be employed for the remainder of our project.

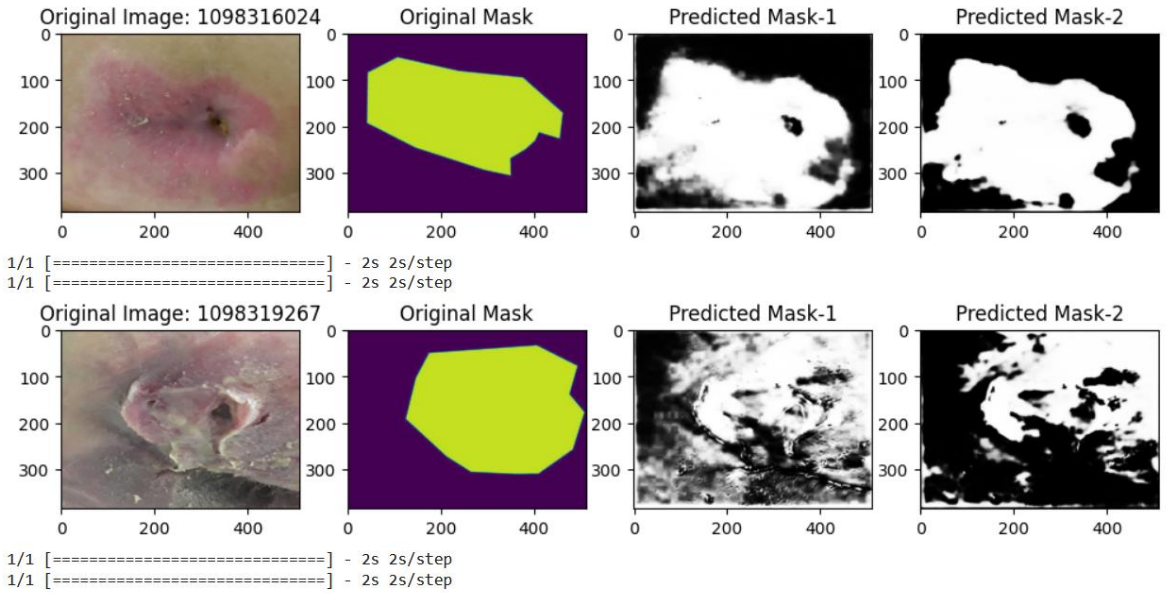
3.2 - Double U-Net

Many combinations of experiments have been carried out to find the parameters that will give the best performance for the Double U-Net model, and these can be summarized as follows:

- Ventura dataset is used
- For 2 different dataset types (boundboxing and fill images)
- For 2 different train/test splits (80/10/10 and 70/30)
- For 3 different learning rates (1E-02, 1E-03 and 0.0005)
- For 3 different batch sizes (4, 8, 16)

Based on the Dice_Coef and IOU metrics applied to the test dataset, the optimal configuration includes fill-type images, a learning rate of 1e-3, a batch size of 4, and a train/test split ratio of 70%-30%. It's important to highlight that certain experiments were repeated multiple times to ensure result consistency.

The experiment yielding the most favorable outcomes achieved peak performance during the 10th epoch of training, registering an IOU value of 0.6601 and a Dice coefficient value of 0.7929. Notably, the best results for the test dataset, which was not part of the model's training and thus unseen by the model, yielded an IOU of 0.4820 and a Dice coefficient of 0.6478. Despite identifying optimal parameters, it should be acknowledged that the visual results produced by the models did not consistently meet the desired level of success in every instance.



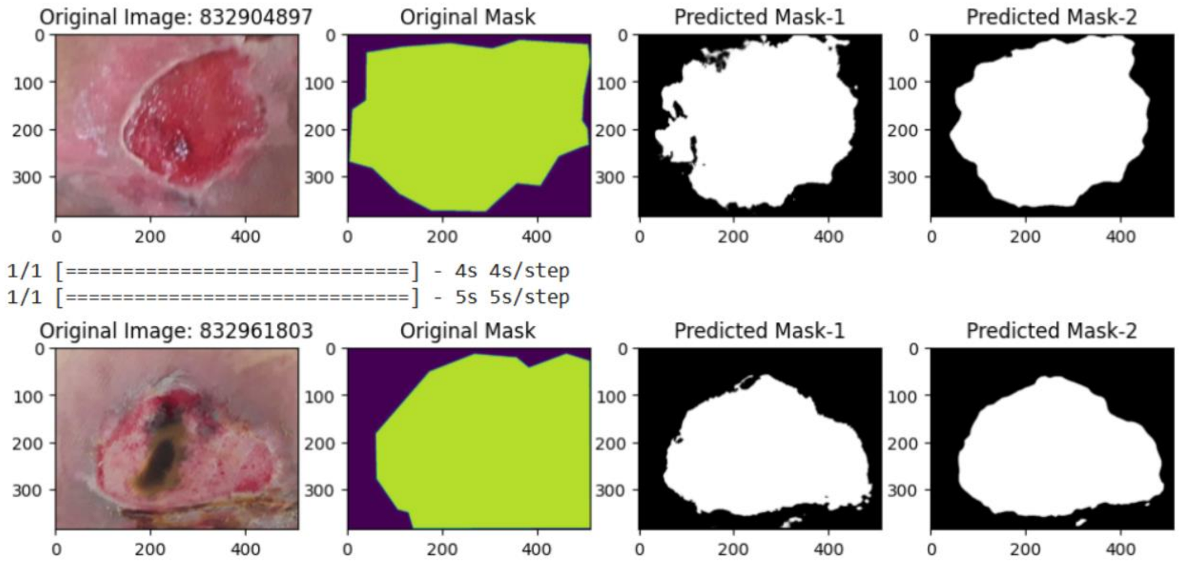
```
1/1 [==============================] - 2s 2s/step
1/1 [==============================] - 2s 2s/step
```

```
1/1 [==============================] - 2s 2s/step
1/1 [==============================] - 2s 2s/step
```

Figure-3: Some Test Results for the Best Double U-Net Model Developed in the First Stage

3.3 - Double U-Net Based On Pretrained Model

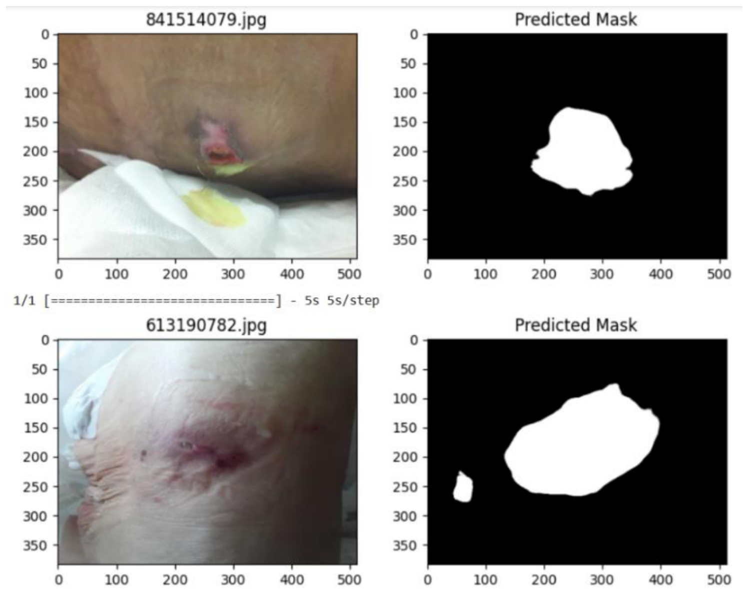
The provided image displays instances where the model did not achieve the desired level of success. Consequently, we opted for an alternative strategy. Rather than training the model from the ground up, we chose to train it based on a different model that had been previously trained for medical image segmentation and demonstrated highly successful outcomes. In this approach, we observed a generally improved performance, with the model even surpassing the original masks in terms of results for certain images:



```
1/1 [==============================] - 4s 4s/step
1/1 [==============================] - 5s 5s/step
```

Figure-4: Tests Where the Optimized Model Produces Better Results than the Original Masks

Initially, we had employed this model for cropped images. However, considering that the application we intend to develop will operate on uncropped images in real-world scenarios, we conducted tests using uncropped images. Surprisingly, we found that the model performed well without requiring any additional optimization.



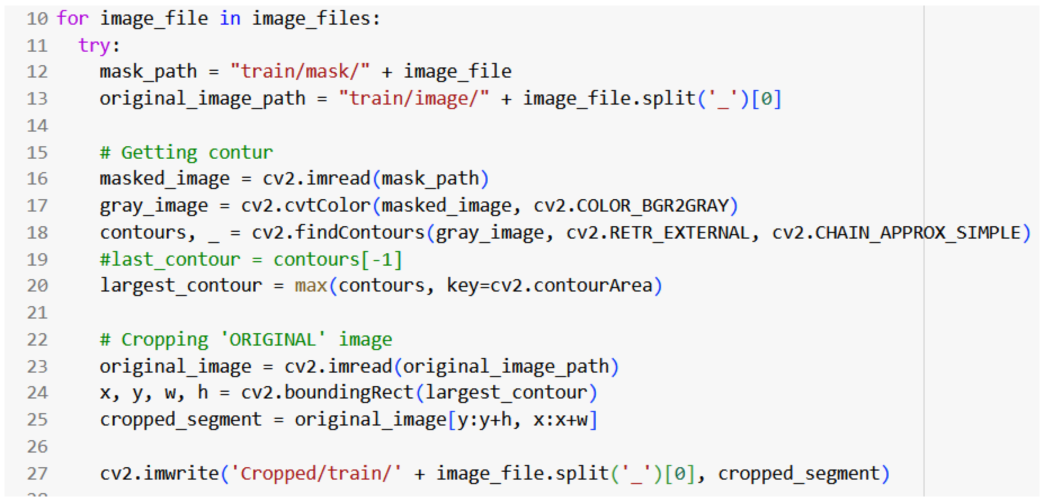
```
1/1 [==============================] - 5s 5s/step
```

Figure-5: Testing the Final Segmentation Model with Uncropped Images

4. Cropping Images and Producing .Txt Files

We initially operated on pre-existing cropped images for segmentation purposes. To render segmentation applicable in real-world scenarios and enhance the classification model through the Region of Interest (RoI) technique, we required code capable of detecting the affected portion within images and cropping that segment. To achieve this, we developed a function that:

- We segmented large (uncropped) images and identified the segmented section.
- We cropped the segmented section to generate the cropped image.
- Additionally, we stored the coordinates of the segmented section in a .txt file.



```python
for image_file in image_files:
  try:
    mask_path = "train/mask/" + image_file
    original_image_path = "train/image/" + image_file.split('_')[0]

    # Getting contur
    masked_image = cv2.imread(mask_path)
    gray_image = cv2.cvtColor(masked_image, cv2.COLOR_BGR2GRAY)
    contours, _ = cv2.findContours(gray_image, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    #last_contour = contours[-1]
    largest_contour = max(contours, key=cv2.contourArea)

    # Cropping 'ORIGINAL' image
    original_image = cv2.imread(original_image_path)
    x, y, w, h = cv2.boundingRect(largest_contour)
    cropped_segment = original_image[y:y+h, x:x+w]

    cv2.imwrite('Cropped/train/' + image_file.split('_')[0], cropped_segment)
```

Figure-6: A Code Block That Finds and Crops the Segmented Section

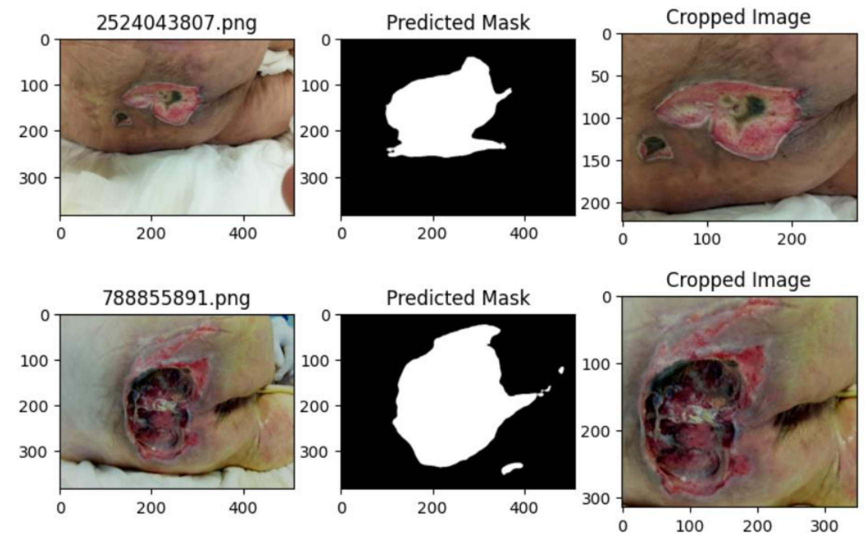


Figure-7: Producing Smaller Images With Cropping the Diseased Parts of Large Images

5. Boosting Classification With RoI
A region of interest (ROI), often abbreviated as ROI, refers to a specific subset within a dataset identified for a particular purpose. This concept finds widespread application in various fields. For instance, in medical imaging, one might delineate the boundaries of a tumor on an image or within a volume to measure its size accurately. In the realm of computer vision, an ROI serves to define the borders of an object under scrutiny. Various methods exist to determine these boundaries, but for our project, which aims to generate rectangular images, we opt to identify four corners to encompass the entire wound.

Upon scrutinizing the photographs in our dataset, we observed areas unrelated to the wound. In the final iteration, the uploaded photos on our mobile application may include regions devoid of scars. Consequently, we deemed the utilization of ROI techniques essential for achieving optimal results. As we approach the conclusion of our project timeline, efforts have been invested in incorporating ROI information into the code, and our experiments are ongoing.

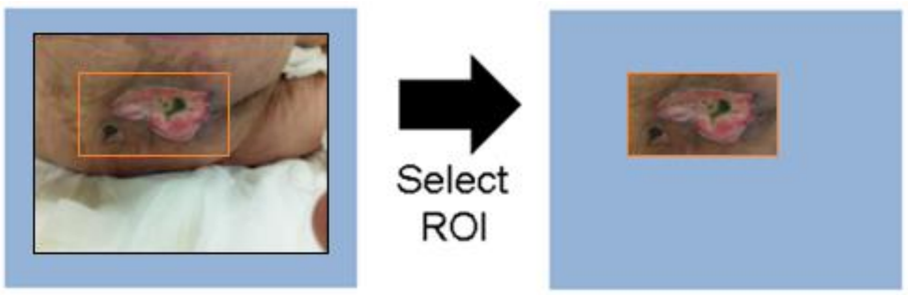


Figure-8: Working Principle of Region of Interest (RoI) in Ventura Dataset

Continuous Learning (/ 25 Points)

You have been working on the problem for almost 3 months. In these 3 months, what did you learn about this problem that you didn't know at the beginning? Did this new knowledge change your perspective on the problem? What else do you need to learn in the future?

Over the course of three months, we acquired both theoretical and practical expertise in the field of segmentation, an area previously unexplored in our machine learning and artificial intelligence pursuits. This period allowed us to apply image processing techniques, of which we had a foundational understanding, to real-life scenarios. Beyond mastering terminological intricacies like the dice coefficient and IoU, we delved into extensive practice in model optimization. Collaborative efforts also provided us with hands-on experience in optimizing someone else's code.

A most critical lesson learned in the realm of segmentation is the potential for metrics such as IoU and dice coefficient to be misleading. We came to recognize the importance of relying on direct images as the foundation for the most accurate interpretation.

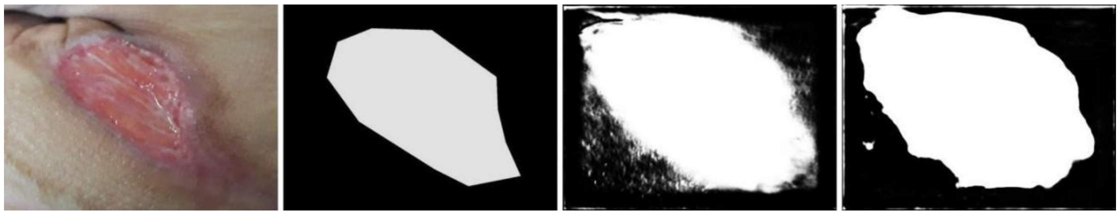


Figure-9: Best Model According To Metrics (In First Experiments)

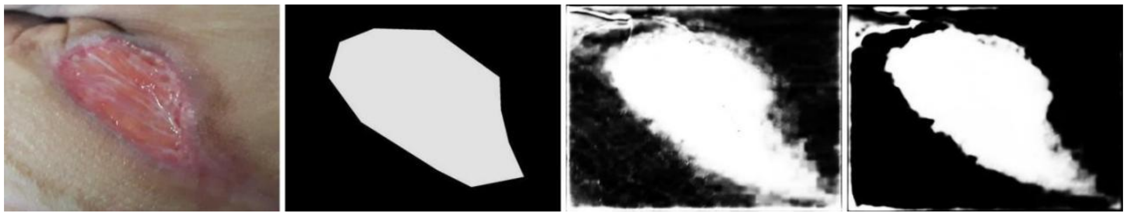


Figure-10: Best Model According To Images (In First Experiments)

A crucial insight gained in the realm of image processing is the distinct outcomes provided by boundboxing and fill images in segmentation. Notably, fill images, our preference due to their superior results, entail complete coverage of the object's area, accurately representing the object's shape and size. Consequently, this comprehensive filling enhances the perception of details within the object, facilitating a more precise determination of its location. Such precision proves instrumental in enhancing the performance of object classification and detection.

We aim to enhance the success of your second-term classification through various methodologies. As outlined in our plan, the optimization process comprises three stages:

1. Initial machine learning and parameter optimization (it was completed at the beginning and middle of the semester).
2. Optimization of results using Region of Interest (RoI) techniques (it will be completed during the mid-term break).
3. Optimization of results employing an alternative methodology (will be handled in the second semester).

Moving forward, there is a necessity to incorporate new techniques, particularly in model optimization, requiring the development of corresponding code. Mobile programming, especially with React Native, demands the utilization of numerous technologies, with a particular emphasis on JavaScript.

Risk Assessment and Management (/ 20 Points)

Try to identify the potential risks in the rest of the project. Were you able to identify these risks at the beginning of the project, or did you recently recognize them? What are your proposed solutions to each risk item? Are there any risks that will require a significant change in the project? If so, explain how this will affect the end results (also, outline your proposed revisions in the next section).

We faced several challenges and took corresponding measures to address them such as:
- Encounter of meaningless errors during mask image production: Some images lacked defined coordinates in the JSON files. We resolved this by skipping 5-10 relevant images.
- Insufficient success with the U-Net++ model: Transitioning to the Double U-Net model proved successful, overcoming the initial challenges.
- Overfitting issues despite good metric results with Double U-Net: Training our model on a pre-existing, medically trained model for image segmentation resolved this concern.
- Model training difficulties due to high RAM requirements: Acquisition of Colab-Pro, offering high RAM, addressed the issue.
- Lengthy training times for Double U-Net models: Each experiment took approximately 1 day, prompting strategic parameter adjustments for risk management.
- Code development for efficient parameter tracking: To facilitate analysis of classification results, we created code to record model parameters at the beginning of the .txt files displaying experimental results.
- Time constraints in RoI studies due to shared project computer usage: Consideration of regulated usage hours could alleviate this challenge.
- Extra effort due to discrepancies in coding libraries from external sources: We navigated this by aligning the installed libraries on our computer with those referenced in external solutions.

With the resolution of these challenges and substantial progress in the project, we anticipate a smooth continuation. As we approach the 2nd term, we believe the challenging aspects are behind us, and we are well-prepared to address any future issues effectively. We anticipate beginning the 2nd term with the project 80% complete, confident in our ability to handle any potential challenges.

Revisions (/ 15 Points)

If you feel like you need to revise your earlier plan, suggest your changes here. These changes may include changes in outcomes of the project, changes in milestones, changes in calendar, changes in workload distribution, etc. If you do not present any revisions here, at the end of the project you will be responsible for all the proposed outcomes in the Project Proposal.

We have no intentions of modifying the existing plan crafted at the outset of this period, which we have largely adhered to thus far. Our goal is to conclude it early in the second semester, avoiding unnecessary delays before the workload intensifies. To achieve this, we aim to make substantial progress on the Android-based mobile application by leveraging and integrating the functions, codes, and models developed during the break. Essentially, the Python-written backend will be linked to the mobile interface.