50.039 Theory and Practice of Deep Learning

Early Detection of Skin Lesions

Jon-Taylor Lim Ming Te	1005053
Foo Chuan Shao	1005549
Nigel Mun Yit Hung	1005031

GitHub Link: https://github.com/chuanshaof/50.039-DL

WandB Link: https://api.wandb.ai/links/jooz-cave/gcn8sysl

Original Dataset Link: https://challenge.isic-archive.com/data/#2018

Table of Contents

1. Introduction

Skin cancer is a prevalent and potentially life-threatening disease worldwide, with early detection playing a crucial role in improving survival rates. Most skin cancers are visible, making skin examinations, both self-conducted and by dermatologists, vital for early detection. This is critical because it can significantly improve the chances of a cure before the cancer becomes more dangerous or deadly. (Skin Cancer Foundation, 2023)

Dermoscopy, a non-invasive imaging technique, has become integral in diagnosing skin lesions. It enables the visualisation of diagnostic features not visible to the naked eye, such as pigment networks, vascular structures, and specific patterns associated with different types of skin cancers (NIH, 2024). However, accurately segmenting dermoscopic attributes remains challenging and time-consuming for dermatologists. Therefore, there is a pressing need for a robust deep-learning neural network capable of automatically segmenting the dermoscopic attribute on lesion images with high accuracy and efficiency. The integration of artificial intelligence (AI) and machine learning technologies in skin cancer detection shows promise in overcoming these challenges. Recent studies demonstrate the robust potential of AI-based techniques to enhance diagnostic accuracy and patient outcomes, particularly in the early identification of melanoma (Dermatology Times). Notably, advanced AI techniques involving Convolutional Neural Networks (CNNs) and hybrid models have shown high effectiveness in diagnosing skin lesions through dermoscopic images. Such AI-driven approaches offer the potential for more accurate, efficient, and less subjective analysis of dermoscopic images, ultimately leading to better patient outcomes through the early detection of skin lesions. With this, this is a semantic image segmentation task and we aim to produce a regional mask of the detected skin lesion.

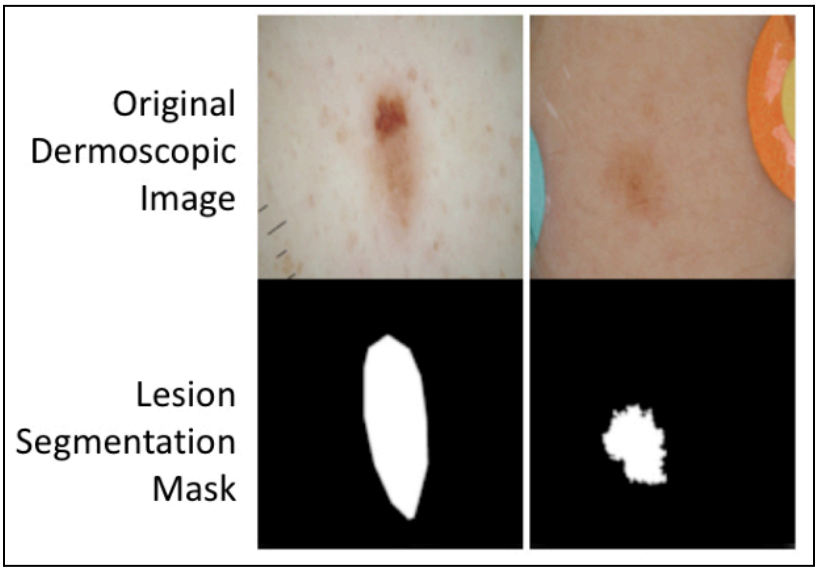

Figure 1: ISIC 2018 Task 1 Objective

2. Dataset

2.1. Overview

In our study, we leveraged a dataset of dermoscopic lesion images from the International Skin Imaging Collaboration (ISIC) 2018 Challenge, specifically Task 1: Lesion Boundary Segmentation. The dataset is publicly available and can be accessed through the ISIC archive (https://challenge.isic-archive.com/data/#2018). Each image within the dataset follows the naming convention ISIC_<image_id>.jpg, where <image_id> is a unique seven-digit identifier, facilitating easy reference and retrieval.

The lesion images were obtained from a historical sample of patients presented for skin cancer screening, from several different institutions. Every lesion image contains exactly one primary lesion; other fiducial markers, smaller secondary lesions, or other pigmented regions may be neglected.

The ISIC 2018 dataset inherently comes with a predefined split for training, validation, and testing. This division consists of 2,595 images for training, 100 images for validation, and 1,000 images for testing.

Dataset	Number of Images	Image Dimensions	Color Space
Training	2,595	Varies	RGB
Validation	100	Varies	RGB
Test	1000	Varies	RGB

Table 1: ISIC 2018 Task 1 Dataset

Each image in the dataset is provided in JPEG format, in RGB color space. Corresponding to each input image is a ground truth mask in grayscale, highlighting the precise boundary of the lesion. These masks match the dimensions of their respective images, serving as the standard against which the segmentation performance of our models is measured.

2.2. Image Augmentation

To enhance the detection accuracy of skin cancer from dermoscopic images, our project adopts on-the-fly augmentation within a data preprocessing pipeline. This approach enriches the dataset's diversity with each training iteration, exposing the model to a variety of augmented images. This method, as opposed to static preprocessing before training, simulates a more varied dataset without needing additional storage and improves the model's ability to generalise across the diverse manifestations of skin lesions.

We selected the Albumentations library for our preprocessing due to its efficiency and versatility in augmenting images. This choice provided us with a comprehensive set of tools for performing image transformations.

We have executed a series of transformations on our images, including resizing, rotating, flipping, and normalizing. Resizing ensures that all images fed into our neural network have uniform dimensions. Rotation and flipping (both horizontally and vertically) augment our dataset, introducing a variety of perspectives and orientations, helping our model become more robust and less prone to overfitting on the training data. Normalization adjusts the pixel values in our images, aiding the neural network to converge quickly during training by providing data within a scale it can more easily interpret.

3. Training Methodology

3.1. Overview

Through the many models that we have chosen to explore, we begin by exploring the hyperparameters and configurations. We iterate through various permutations with the help of the Weights & Biases (WandB) platform to search.

After finding out the best hyperparameters, we follow it up by conducting regular training. Throughout training, model checkpoints are saved after each epoch, allowing the model's performance to be validated. This validation is crucial for monitoring the model's ability to generalize to unseen data and for early stopping if necessary.

3.2.Training Procedure

3.2.1. Hyperparameter Tuning

Our project employed a hyperparameter tuning process documented in the *sweep.ipynb* file, aiming to optimise our model for the precise task of skin lesion segmentation. Through the utilisation of Weights & Biases (WandB), we established a sweep configuration targeting essential hyperparameters tailored to our model's architecture. An aspect of our methodology was the decision to train each set of hyperparameters for 3 epochs. This duration, averaging 20 minutes per sweep, was chosen to explore a wide range of hyperparameter combinations without incurring large computational costs or time.

The choice of 3 epochs is also supported by research suggesting that significant improvements in model performance can be observed within the initial epochs of training, especially when the model architecture and dataset are well understood (Smith, L.N., 2017). Early iterations often have substantial learning, allowing us to gauge the efficacy of different hyperparameter sets quickly. This approach is advantageous in our project where computational resources are limited, and rapid iteration is necessary.

Using Bayesian optimisation, we navigated the hyperparameter space efficiently. Unlike grid or random search methods, Bayesian optimization constructs a probabilistic model mapping hyperparameters to the target metric — the Dice Score, in our case. This approach selects new hyperparameter combinations to evaluate, focusing on those most likely to improve upon the current best performance.

3.2.2. Model Refinement and Validation

After identifying the optimal hyperparameter set, we proceeded to the model refinement stage, training for 10 epochs in the train.ipynb file. This allows the model to further adjust its weights and biases towards minimising the loss function, enhancing its segmentation accuracy. The reason behind the choice of 10 epochs is because an increase in training duration can lead to

deeper model learning without significantly elevating the risk of overfitting, particularly when combined with effective regularisation techniques (Goodfellow, I., Bengio, Y., & Courville, A., 2016).

Model checkpoints are also saved based on the validation evaluation score, ensuring that only the model instances demonstrating good learning and generalization capability on unseen data were retained. This is crucial for mitigating overfitting, ensuring that the model's improved performance reflects advancements in learning the underlying patterns of the dataset, rather than memorising the training data.

3.3. Optimiser

The Adam optimiser is employed as the optimisation algorithm. This choice is driven by Adam's efficiency in computing adaptive learning rates for each parameter, making it particularly effective for problems with large datasets and high-dimensional spaces.

3.4. Loss Function

The Binary Cross-Entropy with Logits Loss (BCEWithLogitsLoss) is utilized as the loss function, which combines a sigmoid layer with the BCELoss in one single class. This is suitable for binary classification problems, such as segmenting a pixel into lesion and non-lesion areas.

3.5. Evaluation Metric

The Dice Score measures the similarity between the predicted segmentation and the ground truth. It ranges from 0 to 1, where a score of 1 indicates perfect agreement between the predicted and actual segmentations.

F1 Score and accuracy will not be used, because their utility in the context of semantic segmentation is limited due to several factors:

- Accuracy's Susceptibility to Imbalance: In medical image segmentation, the background (non-lesion) class often dominates the image. Accuracy can thus become skewed, offering an overly optimistic assessment when a model merely predicts the majority class.
- F1 Score's Non-Spatial Consideration: The F1 Score, although similar to the Dice Score in balancing precision and recall, does not account for the spatial distribution of pixels. In segmentation, where the precise delineation of boundaries is essentialt, this spatial consideration becomes crucial.
- Lack of Handling Complex Shapes: Accuracy and F1 do not account for the complex, irregular shapes typical of lesions in dermoscopic images. The Dice Score's focus on the overlap area makes it more sensitive to such shapes.

4. Model Iterations

4.1. U-Net

4.1.1. Overview of Model Architecture

The U-Net model, a convolutional neural network, serves as the foundational framework for our exploration of image segmentation tasks. Having a U-shaped architecture, U-Net is distinguished by a combination of a contracting path (encoder) and a symmetric expanding path (decoder). The encoder captures comprehensive contextual information, while the decoder ensures the precise localisation of segmentation features (Olaf R., Philipp F., Thomas B., 2015).

In our project, we have adopted the U-Net's encoder as a critical component for convolution operations. A significant aspect of our modifications to the U-Net structure involves its adaptation to accept various kernel sizes. This customization allows for more flexible and nuanced convolution operations, further amplifying our capability to tackle detailed segmentation tasks with greater accuracy and efficiency.

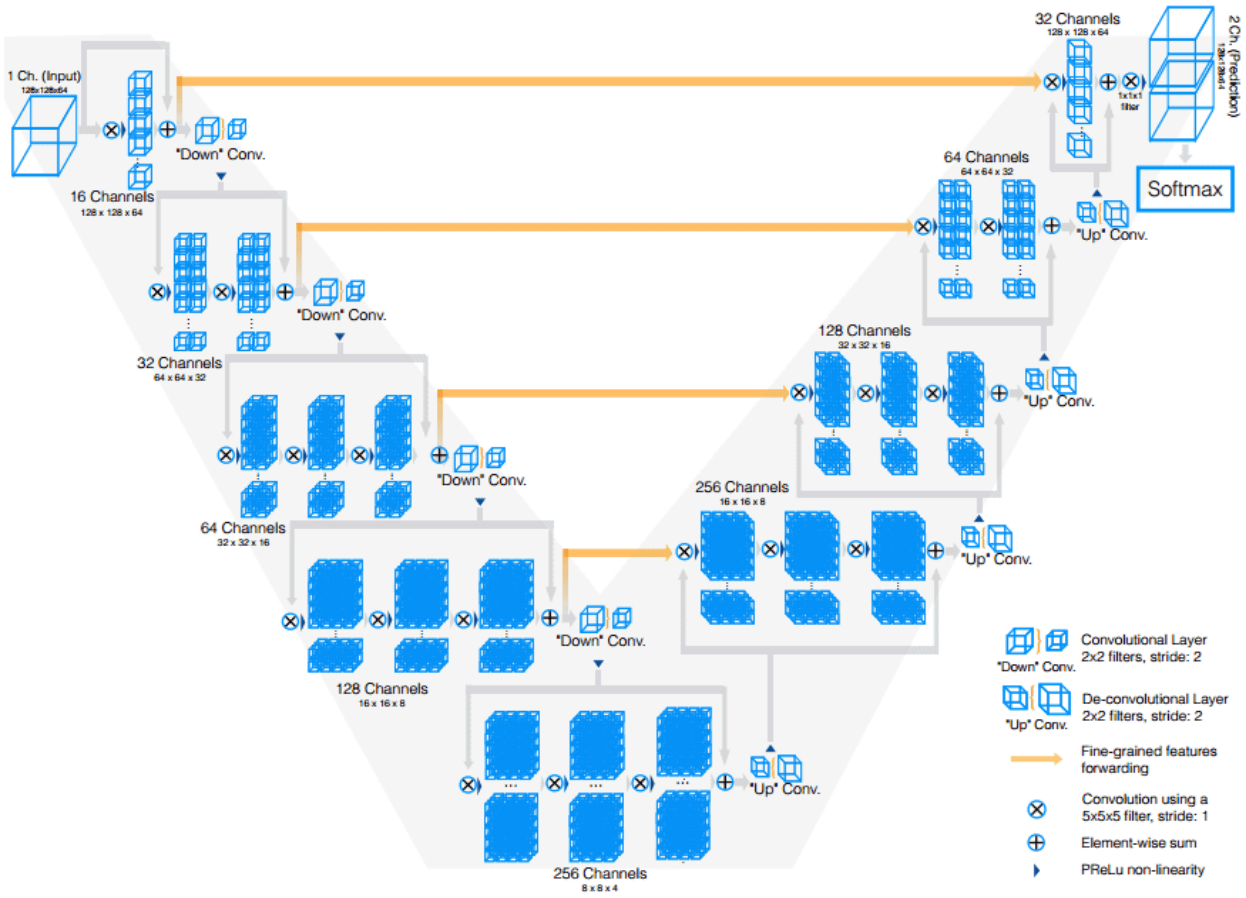

Figure 2: U-Net Architecture

4.1.2. Detailed Exploration of Model Components

Encoder:
We use four DoubleConv blocks, enhancing the channel depth progressively—64, 128, 256, and 512. Each block conducts convolutions with a variable kernel size, stride of 1, and variable padding, paired with batch normalization and ReLU activation, to extract and refine features while maintaining the feature map's spatial dimensions.

After each DoubleConv block, max-pooling with a kernel size of 2 and stride of 2 is applied, reducing the spatial dimensions of the feature maps by half, enabling the encoder to condense and emphasize relevant features across broader contexts efficiently.

Bottleneck:
Feature Processing: At the network's core, the bottleneck doubles the channel depth from 512 to 1024 through a DoubleConv block, serving as a node for the deepest feature abstraction.

Decoder:
Upsampling and Refinement: The decoder reverses the encoding process through transposed convolutions and DoubleConv blocks, sequentially restoring spatial dimensions and integrating high-level and low-level features via skip connections. This ensures the model reconstructs detailed and accurate segmentation maps.

Skip Connections: These merge feature maps from the encoder with those in the decoder, crucial for recovering spatial details lost during downsampling.

Final Convolution:
Segmentation Map Generation: After the last DoubleConv block in the decoder, a final 1x1 convolutional layer (with kernel_size=1) maps the 64-channel feature map to the desired number of output channels for segmentation. This layer acts as a classifier for each pixel, determining its class or belonging to a specific segment.

4.1.3. Hyperparameters Tuning (43 sweeps)

For U-Net, we completed a total of 43 sweeps. Given the number of permutations possible, we have found 43 sweeps to sufficiently cover different possible outcomes. The tuned hyperparameters include:

- Learning Rate: 0.0001, 0.001, 0.01
- Batch Size: 4, 8, 16, 32, 64
- Dropout Rate: 0, 0.1, 0.2, 0.3
- Kernel Size: 3, 5, 7

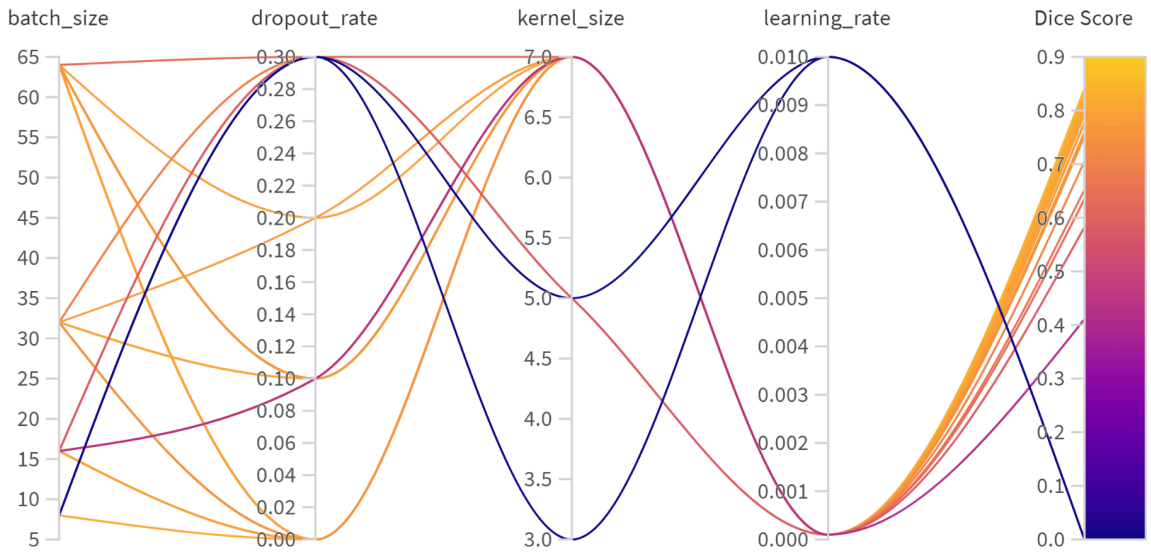

Figure 3: Combinations of hyperparameters with the associated Dice Score for U-Net

Out of the 45 sweeps, we have found that the best hyperparameters achieved a dice score of 0.8315. The hyperparameters are:

- batch_size: 16
- dropout_rate: 0
- learning_rate: 0.0001
- kernel_size: 7

The training loss, validation loss, and dice scores can be found below:

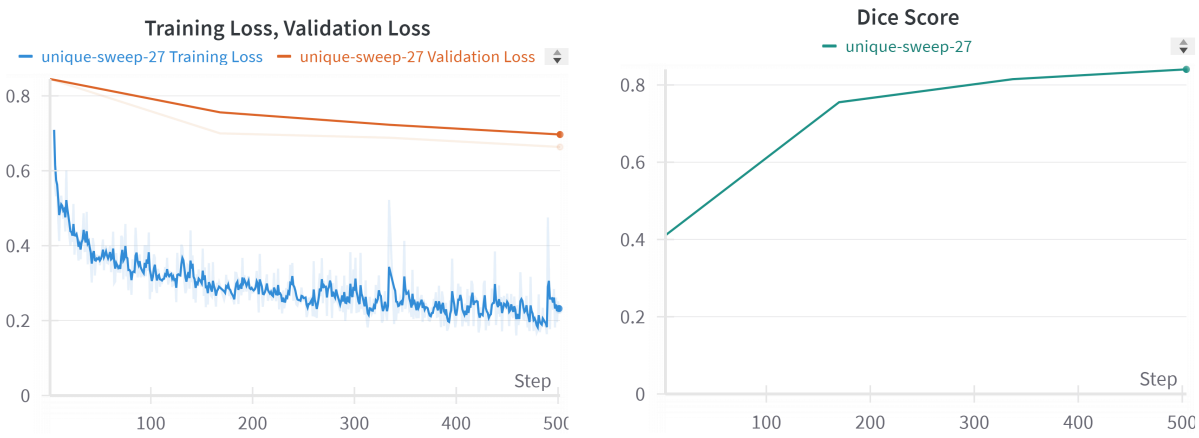

Figure 4: Training loss, Validation loss and Dice Score for each sweep step for U-Net

4.1.4. Further Training

We proceeded to an extended training phase, consisting of 10 epochs of training. Our model was evaluated at the end of each epoch, with the performance metric being the Dice Score. This approach allowed us to continuously monitor and capture the most effective model iteration throughout the training process.

With this, we have achieved the highest Dice Score of 0.8596 on the 8th epoch. This score signifies a high degree of agreement between the model's segmentation outputs and the ground truth. A visualisation of the model outputs and the ground truth can also be found below.

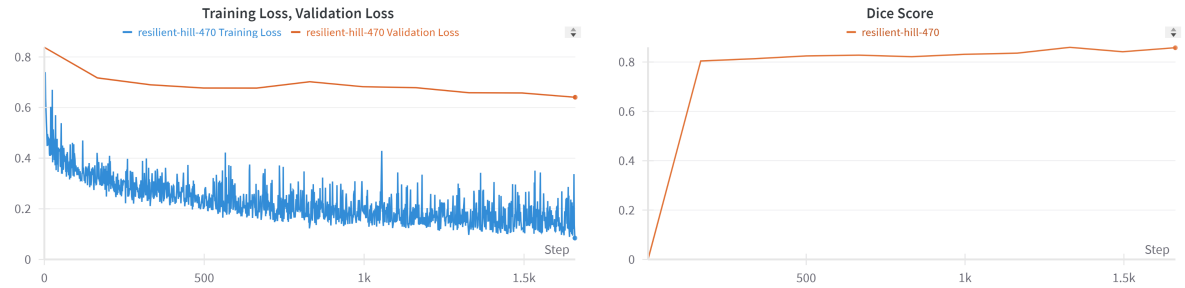

Figure 5: Training loss, Validation loss and Dice Score for each training step for U-Net

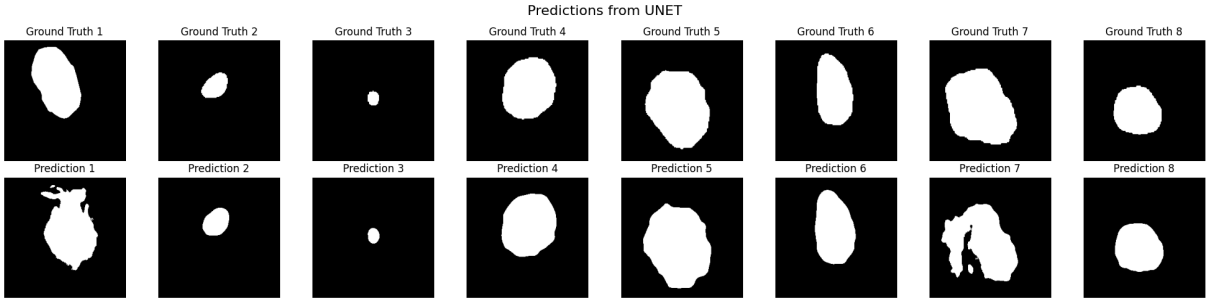

Figure 6: Predicted VS Ground truth for test set for U-Net

4.2. UNETR

4.2.1. Definition and Model Architecture

The UNETR model merges the transformer's global processing with the UNet's segmentation advantages. It excels in parsing complex medical images for nuanced segmentation, essential in diagnostics and treatment planning (Cornell University, 2021)

Our project adapts UNETR to meet specific goals by refining its transformer aspects, like embedding dimensions and attention head counts, enhancing detail capture and processing efficiency in varied medical imagery.

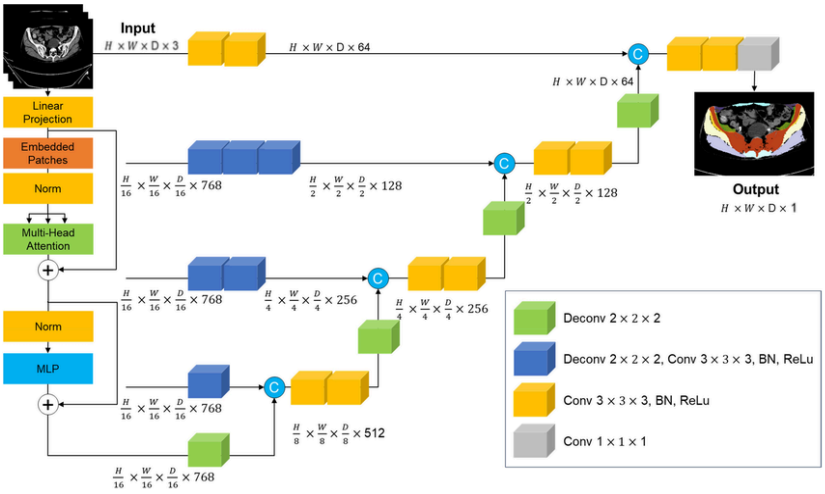

Figure 7: UNETR Architecture

4.2.2. Detailed Exploration of Model Components

Vision Transformer (ViT):
In the Patch Embedding process, the Vision Transformer (ViT) segments each input image into patches, subsequently embedding each patch into a d-dimensional vector space. This enables the model to analyse local features unique to each patch.

The Attention Mechanism of the ViT is built upon layers, each equipped with multi-attention heads. This configuration allows the model to effectively capture and interpret complex relationships between patches, significantly boosting its capacity to identify intricate patterns dispersed throughout the image.

Convolution and Deconvolution Blocks:
The Conv2DBlock applies a 3x3 convolution, followed by batch normalization and ReLU activation. It is used for initial feature extraction and processing concatenated feature maps.

Each Deconv2DBlock doubles the spatial dimensions of its input feature maps. It utilises a 2x2 transposed convolution for up-sampling, followed by a 3x3 convolution, batch normalisation, and ReLU activation. This is important in reconstructing the spatial dimensions of the feature maps at various stages.

4.2.3. Hyperparameter Tuning (65 sweeps)

In UNETR, we conducted a series of 65 sweeps, given the extensive range of possible permutations. As the ratio of embedding dimensions needed to be proportional to the number of heads, we have chosen this set of tuples to iterate through. The hyperparameters adjusted during these sweeps are:

- Learning Rate: 0.0001, 0.001, 0.01
- Batch Size: 4, 8, 16, 32, 64
- Dropout Rate: 0, 0.1, 0.2, 0.3

- Embedding and Heads: (256, 8), (512, 8), (512, 16), (768, 12), (1024, 16), (1024, 32), (2048, 32)

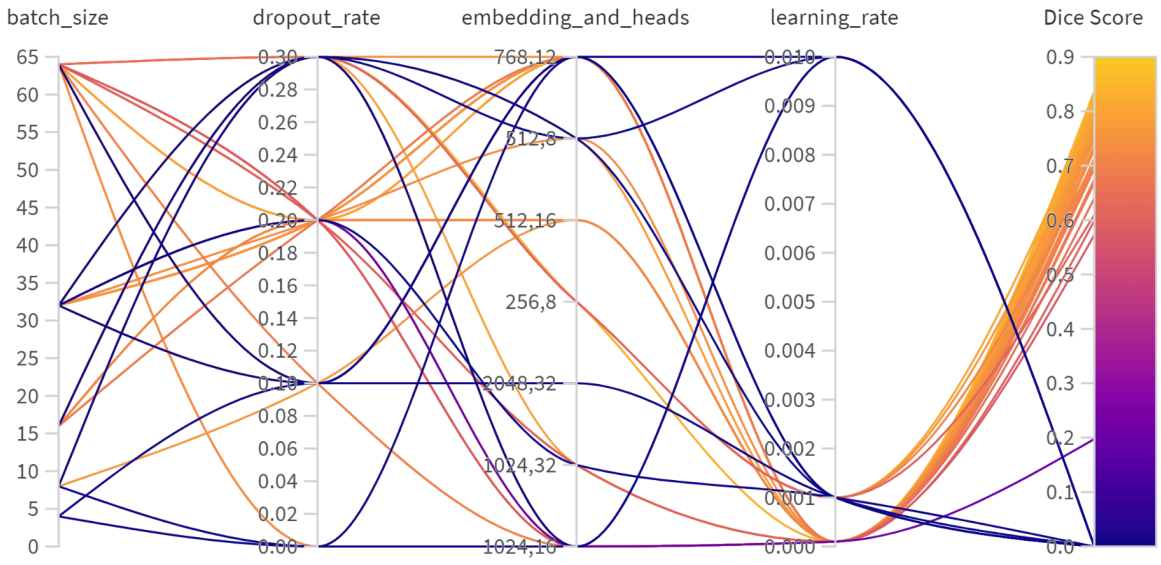

Figure 8: Combinations of hyperparameters with the associated Dice Score for UNETR

Out of the 65 sweeps, we have found that the best hyperparameters achieved a dice score of 0.8315. The hyperparameters are:

- batch_size: 16
- dropout_rate: 0.3
- learning_rate: 0.0001
- embedding_size: 256
- num_heads: 8

The training loss, validation loss, and dice scores can be found below:

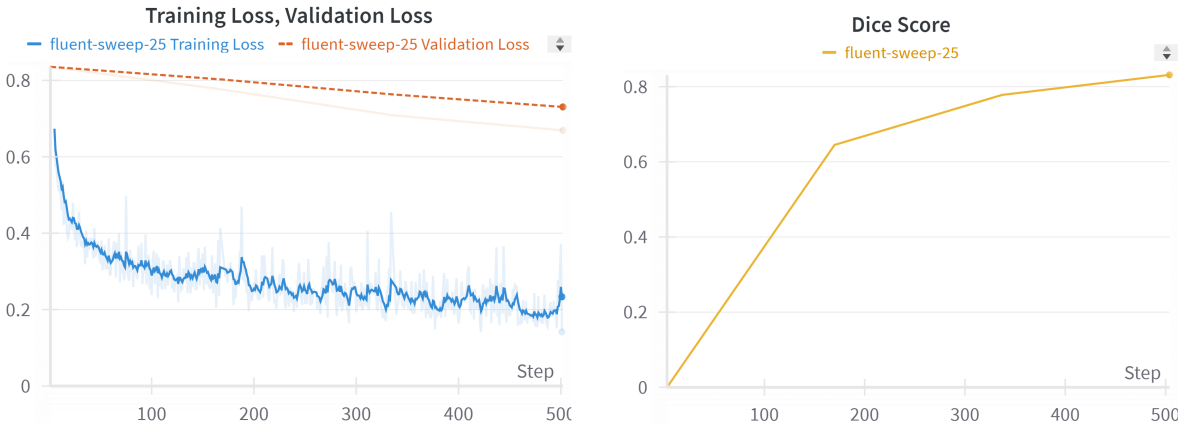

Figure 9: Training loss, Validation loss and Dice Score for each sweep step for UNETR

4.2.4. Further Training

UNETR achieved its highest Dice Score of 0.8434 at the 9th epoch. There was a decrease in performance during the 10th epoch, where the Dice Score dropped to 0.8075. Given this, we adopted the model weights from the 9th epoch for future applications, as they represented the peak of the model's performance. A visualisation of the model outputs and the ground truth can also be found below.

Figure 10: Training loss, Validation loss and Dice Score for each training step for UNETR

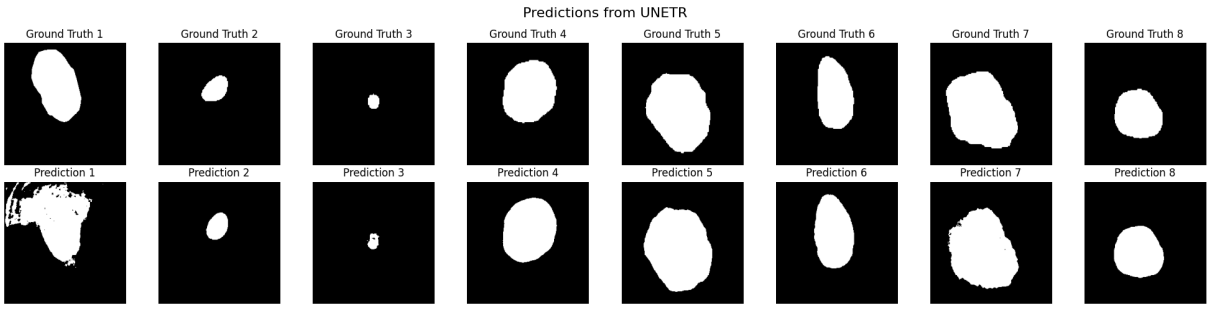

Figure 11: Predicted VS Ground truth for test set for UNETR

4.3. SegFormer

4.3.1. Definition and Model Architecture

The SegFormer model excels in semantic segmentation, combining efficiency with high performance. Its core, the Mix Transformer (MiT) backbone, captures detailed features across scales, improving its segmentation accuracy (Cornell University, 2021).

Inspired by SegFormer, we customised its architecture to enhance spatial information processing early on by adjusting the patch merging mechanism and transforming the Multilayer Perceptron (MLP) into a 1x1 convolution for better spatial feature handling. These changes aim further to improve the model's detail and texture recognition.

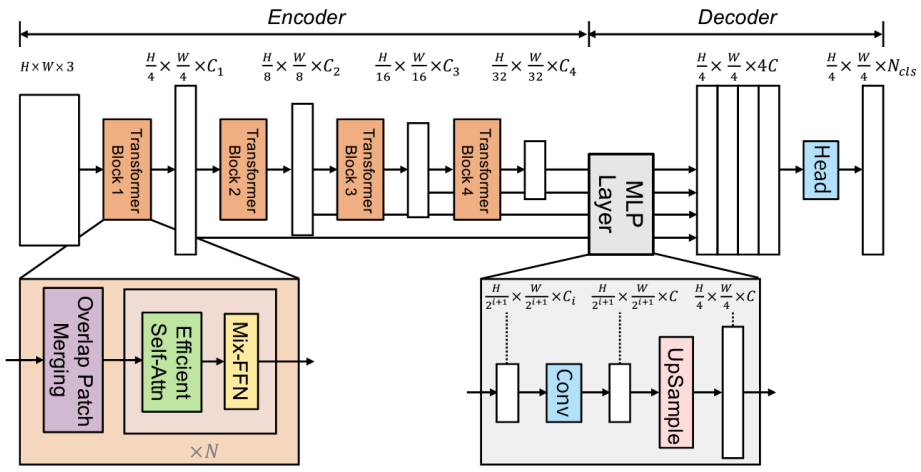

Figure 12: SegFormer Architecture

4.3.2. Detailed Exploration of Model Components

MiT Transformer Backbone:

Designed to process RGB images, employs an architecture that captures detailed features through three color channels and outputs a binary segmentation map. Image patches, sized at 16x16, are processed which enhances the model's capacity for in-depth feature analysis. This system uses 12 attention heads, spread across 12 layers, each adding significant depth to the feature extraction process. This multi-layered approach enables the capture of complex patterns essential for accurate image analysis. To maintain the model's robustness and counteract overfitting, a dropout rate of 0.1 is strategically used, ensuring an optimal balance between complexity and generalization.

Efficient Self-Attention:

Implements scaled dot-product attention, with scaling inversely related to the square root of key vectors' dimensionality, ensuring gradient stability. It uses einsum for efficient computation of attention scores, facilitating a balance between computational efficiency and feature-capturing capability.

Convolutional and Upsampling Layers:

Depthwise Separable Convolution (DsConv2d) is used as an efficient convolution operation that reduces computational cost while maintaining the model's ability to extract features, followed by normalising the features across the channel dimension, stabilizing the learning process. A Mixed Feed-Forward Network expands and then compresses the feature dimensionality, introducing non-linearity and additional capacity to the model.

Segmentation Head:

The outputs from different layers of the transformer are fused together, each upsampled to match the dimensions of the final feature map. A final convolution operation then maps the fused features to the desired output dimensions (number of segmentation classes), followed by an interpolation to match the input image size.

4.3.3. Hyperparameter Tuning (39 sweeps)

For SegFormer, we completed a total of 39 sweeps. This number of sweeps was chosen to efficiently explore the available permutations, which were fewer due to the streamlined selection of hyperparameters we focused on. The tuned hyperparameters include:

- Learning Rate: 0.0001, 0.001, 0.01
- Batch Size: 4, 8, 16, 32, 64
- Dropout Rate: 0, 0.1, 0.2, 0.3

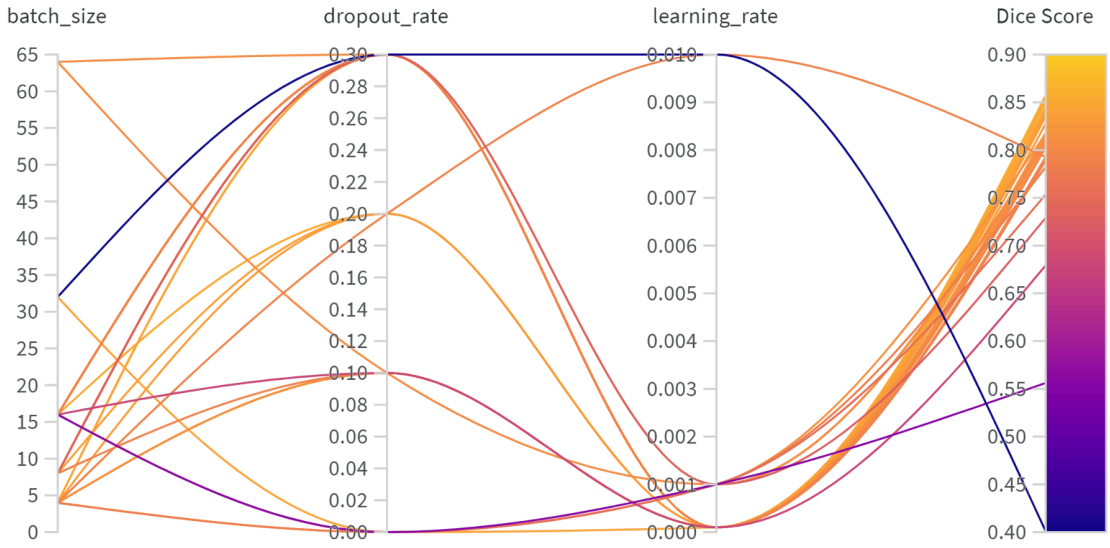

Figure 13: Combinations of hyperparameters with the associated Dice Score for SegFormer

Out of the 39 sweeps, we have found that the best hyperparameters achieved a dice score of 0.8572. The hyperparameters are:

- batch_size: 16
- dropout_rate: 0.3
- learning_rate: 0.0001

The training loss, validation loss, and dice scores can be found below:

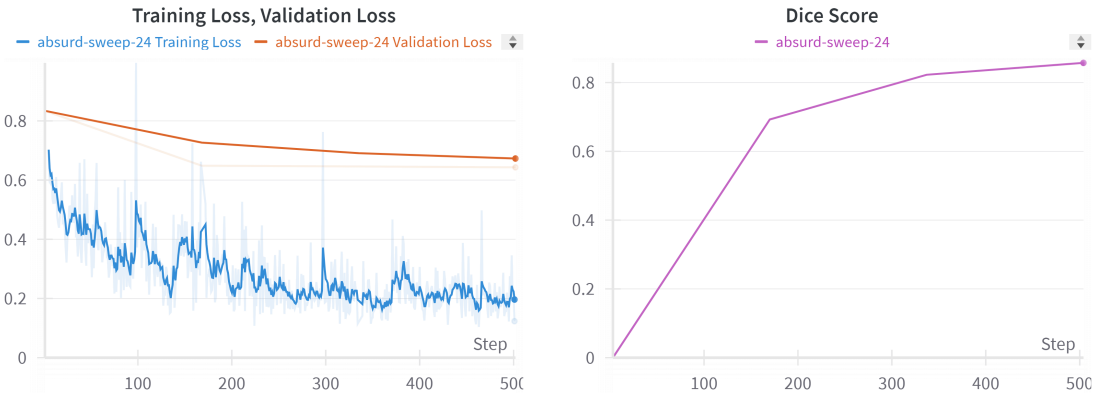

Figure 14: Training loss, Validation loss and Dice Score for each sweep step for SegFormer

4.3.4. Further Training

We proceeded with the extended training phase over 10 epochs, and evaluated the model's performance using the Dice Score at the end of each epoch to ensure precise monitoring and documentation of the model's segmentation abilities.

The peak performance of the SegFormer was observed at the 8th epoch, where it achieved a Dice Score of 0.8769. The scores began to decline in subsequent epochs, which highlighted the 8th epoch as the optimal point in our training regime. We have adopted the model weights from the 8th epoch as they represented the peak of the model's performance. A visualisation of the model outputs and the ground truth can also be found below.

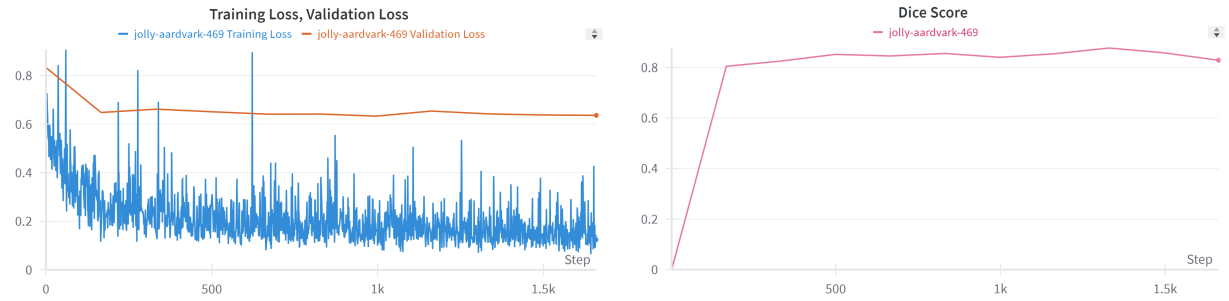

Figure 15: Training loss, Validation loss and Dice Score for each training step for SegFormer

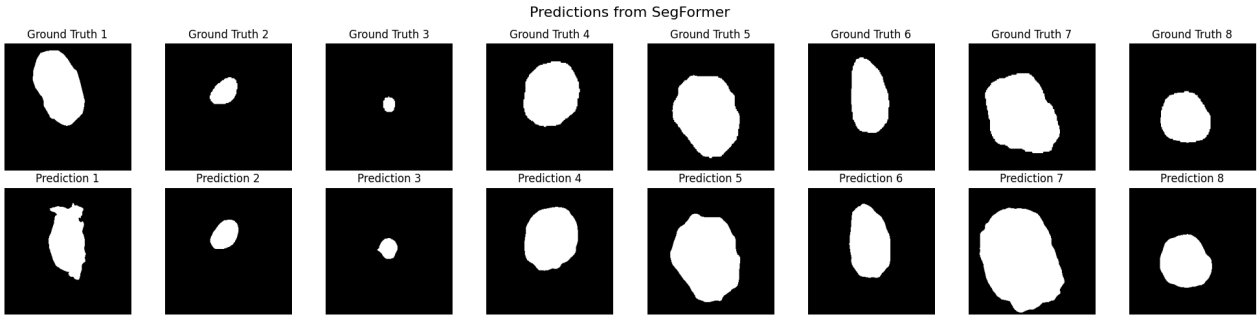

Figure 16: Predicted VS Ground truth for test set for SegFormer

5. Results

5.1.Test evaluation

In our study, we conducted a thorough comparison of our three trained models—U-Net, UNETR, and SegFormer—against well-established, state-of-the-art models, TransFuse. This comparison was essential to gauge the efficacy of our models with current industry standards.

The table below illustrates the comparison between all the models' highest dice scores:

	Ours			State of the art
(Dice score)	U-Net	UNETR	SegFormer	TransFuse
Validation	0.8596	0.8434	0.8769	-
Test	0.8405	0.8068	0.8427	0.8595

Table 2: Comparison of Validation and Test Dice Scores for all models

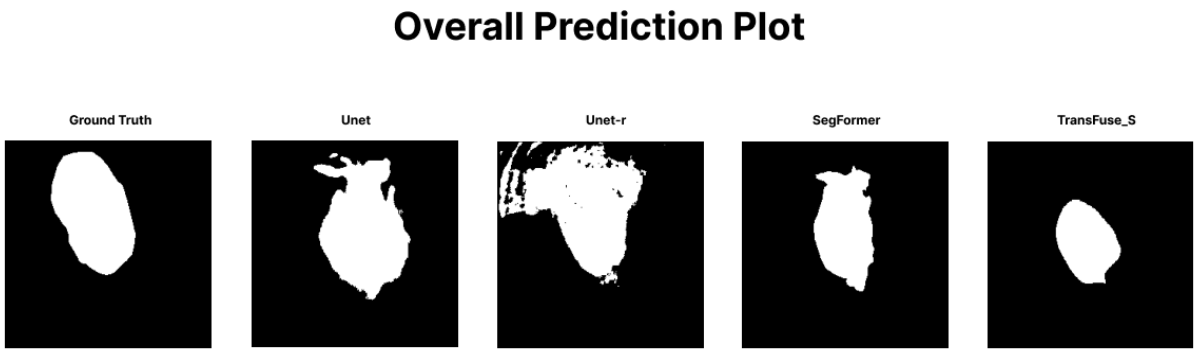

Figure 17: Predicted VS Ground truth for test set for SegFormer

For our models, SegFormer performed the best achieving a test score of 0.8427, closely approaching the performance of advanced models.

The results indicate that our SegFormer model competes closely with TransFuse, highlighting its potential as a high-performing tool in medical image segmentation. Although it did not surpass TransFuse, the narrow difference shows the advancements in our model development processes and suggests that with further tuning and development, our models could reach or even exceed the benchmarks set by existing state-of-the-art solutions. This evaluation not only validates the effectiveness of our models but also underscores the competitive landscape of segmentation technologies in the healthcare sector.

5.2. Limitations

Despite achieving promising results, our study encounters several limitations that need to be acknowledged. Firstly, the slight underperformance of our models compared to the state-of-the-art TransFuse model points to potential areas of improvement in network architecture and training methodologies.

Additionally, the models were trained and tested on a very specific dataset, which may limit their generalisability to other medical imaging datasets with different characteristics.

The computational demand is also noteworthy. Advanced models like SegFormer and UNETR require significant computational resources for training and inference, which might not be feasible in all clinical settings, particularly in resource-limited environments.

In the field of medical image segmentation, training with higher-resolution images would also be more realistic. However, we have trained our models only with 256x256 images due to computational constraints.

5.3. Future Work

To address these limitations and further enhance the performance of our segmentation models, several avenues for future work are proposed:

Model Enhancement: Future efforts could focus on exploring more recent neural network architectures like Vision Mamba (Zhu et al., 2024) which uses State Space Models that might offer improved accuracy and efficiency. Incorporating mechanisms such as attention gates or advanced feature aggregation techniques could potentially boost the segmentation accuracy.

Metric Diversification: Expanding the evaluation framework to include additional metrics like the Jaccard index or clinically relevant performance indicators could provide a more comprehensive assessment of the models' capabilities.

Data Augmentation and Diversity: Implementing more diverse and sophisticated data augmentation techniques could help improve the robustness and generalizability of the models. Additionally, training and validating the models across a broader range of medical imaging datasets would enhance their applicability to various medical conditions.

Computational Optimization: Research into more efficient training algorithms and model pruning techniques could make the deployment of these advanced models more viable in everyday clinical practice. Efforts to optimize models for faster inference could also be explored, potentially enabling real-time segmentation tasks.

Clinical Integration: Collaborating with clinical experts to refine the models based on practical feedback and integrating the models into diagnostic workflows to assess their real-world efficacy would be crucial steps towards their clinical adoption.

6. Conclusion

In our study, we evaluated the performance of three advanced models—U-Net, UNETR, and Segformer—on various semantic segmentation tasks, focusing particularly on skin lesion segmentation. Our results reveal that the Segformer model, which merges transformer and convolutional architectures, has the highest segmentation accuracy, achieving a Dice score of 0.8427. This performance closely approached that of the state-of-the-art TransFuse model, which scored 0.8595, demonstrating the efficacy of our model adjustments and training approach.

Our exploration confirms the value of hybrid deep learning architectures in efficiently tackling complex segmentation challenges. The study also highlighted the benefits of incorporating layer normalization and efficient self-attention mechanisms, which significantly contributed to the improved performance of our models.

Looking ahead, we believe that further enhancements can be achieved by adopting pretraining strategies on more extensive datasets and employing more sophisticated data augmentation techniques. Such advancements could refine our models' capabilities, paving the way for greater accuracy and applicability in medical imaging segmentation. This research not only underscores the potential of advanced model architectures but also sets the stage for future innovations in the field of semantic segmentation.

7. References

Early detection. The Skin Cancer Foundation. (2023, February 27). http://www.skincancer.org/early-detection

Massone, C., Hofman-Wellenhof, R., Chiodi, S., & Sola, S. (2021, August 23). Dermoscopic criteria, histopathological correlates and genetic findings of thin melanoma on non-volar skin. Genes. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8391530/

Renata Block, M. (2023, December 4). Skin cancer detection with AI: How intelligent is it?. Dermatology Times. https://www.dermatologytimes.com/view/skin-cancer-detection-with-ai-how-intelligent-is-it-

Smith, L. N. (2017, May 15). Cyclical learning rates for Training Neural Networks | IEEE Conference publication | IEEE Xplore. Cyclical Learning Rates for Training Neural Networks. https://ieeexplore.ieee.org/document/7926641/

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. http://alvarestech.com/temp/deep/Deep%20Learning%20by%20Ian%20Goodfellow,%20Yoshua%20Bengio,%20Aaron%20Courville%20(z-lib.org).pdf

Ronneberger, O., Fischer, P., & Brox, T. (2015, May 18). U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv.org. https://arxiv.org/abs/1505.04597

Hatamizadeh, A., Tang, Y., Nath, V., Yang, D., Myronenko, A., Landman, B., Roth, H., & Xu, D. (2021, October 9). UNETR: Transformers for 3D Medical Image segmentation. arXiv.org. https://arxiv.org/abs/2103.10504

Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., & Luo, P. (2021, October 28). Segformer: Simple and efficient design for semantic segmentation with Transformers. arXiv.org. https://arxiv.org/abs/2105.15203

Zhu, L., Liao, B., Zhang, Q., Wang, X., Liu, W., & Wang, X. (2024, February 10). *Vision mamba: Efficient Visual Representation Learning with bidirectional state space model*. arXiv.org. https://arxiv.org/abs/2401.09417