Using a Multi-Scale Patching With Vision Transformers for 3D Image Segmentation

Baris Imre
June 2023

***Abstract*— Automatic semantic segmentation of abdominal aortic aneurysms is a crucial medical task, given that manual segmentation, often required multiple times for various scans over time, can be a highly tedious job for professionals. Over the past decade, convolutional neural networks (CNNs), especially U-Net like models, have been the predominant research area in this field. Recently, vision transformer models, with segmentation-related modifications, have exhibited significant promise. However, these models almost invariably adopt the same patching mechanism that divides the input into equal-sized, non-overlapping sections. This method is not necessarily the most effective, but it has become a common practice since it was a remnant from the transition of transformers from text to images and was the first approach to successfully accomplish this. Consequently, it has been employed simply because it works. In this study, we introduce a tree-like patching method that utilizes a multi-scale perspective of the input with a vision transformer. By tokenizing multiple levels of the image with constant size patches, we aim to provide the transformer with more information, taking advantage of the long-range attention inherent in transformer networks. Furthermore, we propose an architecture that fundamentally incorporates this patching approach, in tandem with a fusion of U-Net-like structures and vision transformers. We demonstrate that, given the same architecture, the multi-scale patching outperforms its traditional counterpart in semantic segmentation of an abdominal aortic aneurysm dataset consisting of 90 CT scans. Our findings clearly show that there is a promising research direction in experimenting with more complex patching mechanisms.**

I. Introduction

Deep learning has revolutionized the field of medical imaging in recent years. Its rapid development, combined with the exponential increase in hardware capabilities, has enabled the accomplishment of many tasks that were previously considered unfeasible. With new processes and model architectures continually outperforming their predecessors it is becoming increasingly challenging for researchers to keep up with the pace of advancements. The exploration of higher performing and more efficient methods frequently sparks novel ideas, and concepts from diverse fields are often applied to medical imaging, occasionally yielding exceptional outcomes [1].

Semantic segmentation, aiming to assign a class label to each pixel (or voxel in 3D) in an input image, is an area of substantial research interest. Medical image segmentation using deep learning can serve as a valuable tool across various image modalities. It not only aids in diagnosis and can uncover findings that may be overlooked by other methods, but it can also facilitate early detection, reduce the need for highly trained personnel in poorer regions, and decrease reliance on costly measuring equipment by leveraging cheaper or older technologies. This is especially true for segmentation since manual segmentation of big scans is a laborious process that can take hours by a trained person. Moreover, the segmentation of medical images proves invaluable when applying other processes, such as classification, since segmented scans provide more information than raw ones [2].

U-Net [3] models and similar architectures have dominated the field of medical image segmentation for the past decade [4]. These models are intuitive, easy to understand, and effective, with a wealth of research focusing on improving U-Net-like models and their application to specific tasks. They process input images to leverage a multi-scale representation of the input through an convolutional encoder and decoder architecture.

Among the most prominent models in recent years, the transformer stands out due to its popularity and impressive performance. Initially designed for natural language processing tasks [5], this model has formed the backbone of numerous major deep learning architectures in recent years [6]. Moreover, through a few modifications to the images and the model architecture, it has recently been adapted for vision tasks in the form of the vision transformer [7]. Several instances of vision transformers outperforming other types of models in various tasks have been reported in the last two years [6]. In contrast to U-Net models, vision transformers harness the power of long-range attention to effectively capture relationships between every patch in a given input. By employing self-attention mechanisms, they can directly compute attention scores between any pair of patches, thereby modeling complex spatial interactions and global context, irrespective of their distance within the input image. In this paper, we explore a potential method of combining these two architectures into a single model designed to capitalize on multi-scale view of the U-Net and the long range attention of the transformer.

This study seeks to answer the research question: *to what extent does a multi-scale representation of input tokens achieved with an encoder network enhance the segmentation performance of a vision transformer in 3D CT images?* Throughout the course of this report, we will examine this question in depth, aiming to contribute valuable insights and advances to the field of medical imaging. We will start by providing some necessary background knowledge in section II, followed by

an introduction to the proposed model in section III and the dataset used in all experiments in section IV. The experimental setup and the benchmarks used for performance evaluation will be presented in section V. Then, we will introduce five variations of the architecture to understand and evaluate the components that make up this model in section VI. Finally, we will critically discuss the results and insights derived from these experiments in section VII.

II. Related Works

A. Segmentation and convolutional neural networks

Image segmentation is a well-established area of research. As early as 1979, N. Otsu presented a paper discussing a smart, yet straightforward, threshold selection for gray-level histograms [8]. This method categorizes a series of pixels by optimally deciding on a threshold value for their histograms. Numerous image segmentation algorithms have been developed over the years, spanning from simple thresholding calibrations to region growth, k-means clustering, conditional and Markov random fields, and many more [9].

In recent years, deep learning methods have continually improved. Convolution operations and convolutional neural networks (CNN) [10] have been extremely popular in deep learning and medical image segmentation [11]. One of the earliest tangible results of deep learning applied in medical image segmentation was accomplished by Ciresan et al. [12], where they automatically segmented biological brain membranes in electron microscopy images. Fully convolutional networks (FCNs) [13] are among the early architectures that utilize convolutions exclusively in image segmentation. Long et al. [13] showed that end-to-end convolutional networks can train and infer segmentation tasks with high performance.

B. U-Net

U-Net [3] is a popular deep learning architecture that has made a significant leap in segmentation with deep learning. The U-Net architecture is arguably still one of the most popular methods within segmentation, especially in medical fields [14]. As its name suggests, U-Net is a ”U” shaped network. An input image is gradually downsampled on the contracting side (left) of the architecture using max-pooling layers while passing through convolutional blocks [10]. The result is a set of feature maps that are downsampled and sets of feature maps from the various levels of the encoder with various levels of downsampling. This process is reversed on the expanding (right) side of the network, where the feature maps are upsampled using transposed convolutions and the multiple levels from the encoder are concatenated into the feature maps gradually. These skip connections throughout the network enable the architecture to use a multi-scale view of the input.

C. Transformers

Attention is all you need [5] is the original paper that proposed the transformer architecture, using scaled dot product self-attention. This paper more or less defined the landscape of transformers and the future direction of transformer research for the past 5 years. The idea of transformers came from the following motivational path. In many natural language processing (NLP) tasks, the input of a system is sequential. The reason for this is simple, text or speech in nature is a sequential construct. Analyzing text without word order loses a significant amount of information since it is the ordered combination of words that makes it meaningful. In the world of deep learning, before transformers, these tasks were handled with recurrent neural networks or their (advanced) variants like LSTMs [5]. These models processed data in a sequential fashion since usually the output of one iteration is needed for the next iteration as input. This makes parallelization and thus efficient training quite difficult and actually in most cases not possible. On top of this, sequential models represent order with connected inputs and outputs in consecutive timesteps, meaning the relation of tokens that are far apart is diminished or has the potential to diminish over iterations. So in the end, there was a need for a parallelizable model that could represent long and short-distanced sequence information. Hence the transformer.

The transformer, is an innovative architecture primarily designed for sequence transduction, or translation tasks [5]. At its core, it uses a mechanism called ”attention” that weighs the influence of different input parts in the generation of each output part. Unlike traditional sequence-to-sequence models, the transformer does not require sequence-aligned inputs and outputs, and it avoids the need for recurrent computations, making it highly parallelizable and removes the problem of vanishing gradients. The architecture comprises two parts: an encoder that processes the input and a decoder that generates the output, both being stacks of self-attention and position-wise fully connected layers.

An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors [5]. This paper primarily focuses on a type of attention mechanism called ”scaled dot-product attention”. This mechanism involves a simple mathematical process depicted in equation 1. The queries (Q), keys (K), and values (V) are all vectors derived from the encoded input. The queries are matrix multiplied by the keys to form a compatibility function result. This result is then scaled by the dimension of the keys (d_k) because larger values could lead to very small gradients in the softmax function. The result is then multiplied by the values to generate an attention matrix that captures the relative attention of each token in a sequence to every other token.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \quad (1)$$

D. Vision Transformers

The idea of transformers dominated the area of NLP for some years before there were successful implementations of the architecture to images. The paper *An image is worth 16x16 words: transformers for image recognition at scale* [7] is one of the first successful implementations that adapted the transformer architecture to a visual task. The input of a

regular transformer is a sequence of tokens. Especially text is very straightforward to convert into a sequence of tokens. However, with images, some steps need to be taken before we can use the transformer architecture. In this paper, the following method is used to convert an image into a sequence of tokens. Starting with an image $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$ as input, the image is divided into square patches of size $(P \times P \times C)$. These patches are flattened and gathered into an array, making our input $\mathbf{x}_p \in \mathbb{R}^{N \times (P^2 \cdot C)}$ where N is the number of patches in the image. As transformers use a constant embedding size, these patches are projected to this constant size with a trainable linear projection (in later works sets of convolutions are used to project patches into tokens). From this point on, the input of the model is identical to the original transformer paper [5]. However, in this work only the encoder network of the transformer is necessary. After passing through the encoder stack, the output is put through a final fully connected layer to perform the actual task (called a task-specific head).

Swin UNETR [15] is a state-of-the-art deep learning architecture that combines swin transformers (a variant of the Vision transformer) and U-Net decoder. A swin (shifted window) transformer has two alteration to a vision transformer. Firstly, attention is computed within local windows, hence this computation is more local but much faster. Furthermore, these windows are shifted so that boundaries of attention are crossed between windows. Secondly, following every block within the swin transformer, patches (tokens) are merged into bigger groups with increasing embedding dimensions, and attention windows are also merged. This has an effective downsampling on the image. Swin UNETR uses a Swin transformer as an encoder, while sampling from every layer of the architecture for a multi-scale representation of the input. These representations are merged using a U-Net like decoder to achieve a state-of-the-art segmentation result. Due to this patch merging in the encoder and the structure of the decoder Swin UNETR uses a multi-scale view of the input to perform a segmentation task.

III. Methods

A. Multi-scale patches

As explained in section II, vision transformers patch inputs into tokens before processing them. Conventionally the size of the patches within a single architecture is constant [6]. Given an image of size $(H \times W \times C)$ and a square patching of size P, a conventional vision transformer would patch this input into $\frac{H}{P} \times \frac{W}{P} \times \frac{C}{P}$ patches all with the size $(P \times P \times P)$. In this study, we add on top of this by rescaling the input by downsampling it multiple times and applying this patching, still with a constant patch size, to all of these scales. In this way, we create patches that embed different scales within the same input. For example, given an input image of shape $128 \times 128 \times 128$ and a patch size of 16, a regular patching would result in 512 ($\frac{128}{16} = 8, 8 \times 8 \times 8 = 512$) patches all with size $(16 \times 16 \times 16)$. This holds true for our approach as well, however it is only the first scale. Then if this input was to be downsampled by a factor of 2, the size would be $(64 \times 64 \times 64)$. Applying the same patching would then result in 64 more patches, still all with size $(16 \times 16 \times 16)$. These new patches however, would actually represent a bigger part of the input image since we donwsampled by a factor of 2. In other words, one of these $(16 \times 16 \times 16)$ sized patches would actually represent a $(32 \times 32 \times 32)$ patch in the original input. Meaning this effectively bigger patch would embed the same part of the input as 8 smaller patches combined. Continuing this approach for multiple steps then results in an array of patches that represent different parts of the image in different scales all while having the same size. A 2D example of this patching is presented in figure 1. Another way of thinking about this is an octree (or quadtree in a two dimensional image) where the initial patches without the downsampling are the leaves of the tree and gradually we build this tree up with patches that represent bigger parts and hence combine multiple patches. Since all of these patches have the same size, they can be used as tokens in a vision transformer. This also means that the attention mechanism of the vision transformer can compute attention between a patch that represents a smaller part of the input and a patch that represents a big part of the input, potentially increasing its capabilities and understanding of the image.

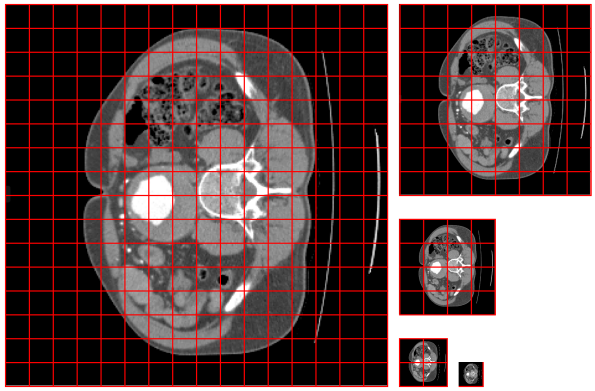

Fig. 1. A 2D example of multi-scale patching. The red squares represent one patch (token) each. Initially the image is not downsampled hence there are $\mathbf{16 \times 16}$ patches. Afterwards, the image is downsampled to half the size while the patch size is constant meaning there are $\mathbf{8 \times 8}$ patches. This approach continues until the entire image is one patch. This also shows that with the bigger image one patch embeds a smaller part of the image while with smaller images one patch embeds bigger parts of the image.

B. Abstract Architecture

To achieve the patching explained above, and to use a vision transformer (transformer encoder) for segmentation, we have designed a network architecture. Figure 2 shows this architecture with abstract building blocks. These blocks (modules) will be explained individually further. In this part, we will explain their purpose within the network and why they are necessary. Multi-scale patching has two parts. First, the input needs to be downsampled for different scales. For this purpose we use a U-Net like encoder, that adapts the contractive path of the U-Net architecture. The result of this network are skip connections that can be viewed as the multiple scales of the input. This is also a learned way of achieving the downscaling, we have

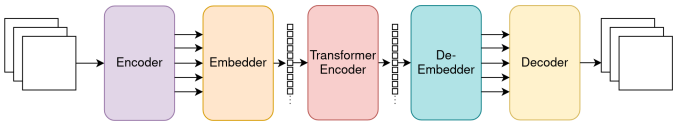

Fig. 2. The overall architecture of the proposed model. The input is encoded to a multi-scale representation by the encoder (represented by the multiple arrows), embedded into tokens by the embedder, processed by the vision transformer. Finally. reconstructed by the de-embedder and the decoder. Input is a 3D CT scan and the output is a segmentation map with one-hot encoding.

found that this works a bit better than non-learned ways of downscaling such as pooling methods. This part as a whole is called the encoder within the proposed network. The second part of the multi-scale patching is the extraction of the patches and their embeddings. Each skip connection of the encoder needs to be patched separately into tokens, and these tokens need to be embedded into the correct token size. We have also chosen to do this in a learnable way. This is handled by the embedder module. These tokens are processed in the vision transformer, which is just a transformer encoder. The output tokens of the vision transformer need to be de-tokenized. This is done by the de-embedder module that reverses the embedder module in a learnable way. This results in a multi-scale view of the input that has been processed by the vision transformer and shaped like smaller scales of the input. To combine these views, we use a decoder module that adapts the expanding path of a U-Net.

C. Building Blocks

In this subsection, we explain each building block of the presented architecture with technical details and design choices. The combination of these blocks form the network.

1) Convolutional Block: Every convolutional block in the proposed network is comprised of a convolutional layer followed by a parametric rectified linear unit (PReLU) activation function, dropout layer, and instance normalization layer sequence. The PReLu activation function introduces non-linearity to the network, enabling the learning of more complex relationships between input and output [16]. The dropout layer is a regularization technique that reduces overfitting and memorization by the network [17]. The instance normalization layer is used to stabilize and accelerate the training process by normalizing the activations within each instance of the input data [18]. This block can be comprised of regular 3D convolutions or 3D transposed convolutions to accommodate for needs of the specific part it is placed in. Strides, kernel sizes and padding are all configurable. Figure 3 visualizes the convolutional blocks.

2) Double residual convolution block: The main building block for the convolutional part of this network is a double convolutional residual block. This is inspired by ResNet [19] to provide more stability, higher learning ability, and mitigate vanishing gradients within all the convolutional parts of this architecture. Figure 4 shows this building block. The single convolutional block on the residual path is to match the channels correctly with the output of the main flow and uses a kernel size of 1.

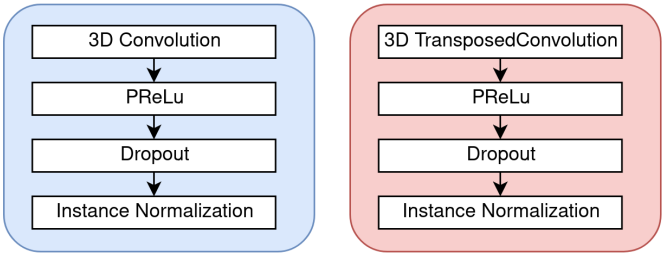

Fig. 3. Visualizations of the single convolutional block. The color codes of red and blue represent the transposed nature of these blocks (red means transposed convolution, blue means regular 3D convolution) and will be used in further diagrams for ease of visualization. The words up and down will be used to signify if the convolutions upscale or downscale the input.

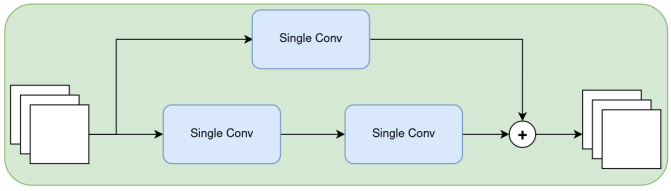

Fig. 4. Double residual convolutional block. The residual part has a single convolutional layer to match the amount of channels with the non residual part using a kernel size of 1. Again, the color code of green will be used for future diagrams to represent this block.

3) Encoder: The encoder module of our network is heavily based on a U-Net [3] encoder. It downsamples the given input to 4 lower resolutions using a combination of the single and the double blocks described above. There are residuals from each block of the encoder that will be used by the rest of the architecture. Figure 5 provides a visualization. We have chosen for 4 levels of downsampling since with an image size of 128 per side and patch size of 16 per side more than 4 downsamplings (each with a factor of 2) is not possible.

4) Embedder: The multi-scale representations of the input from the encoder need to be reshaped and embedded to the correct dimensions according to the patching the vision

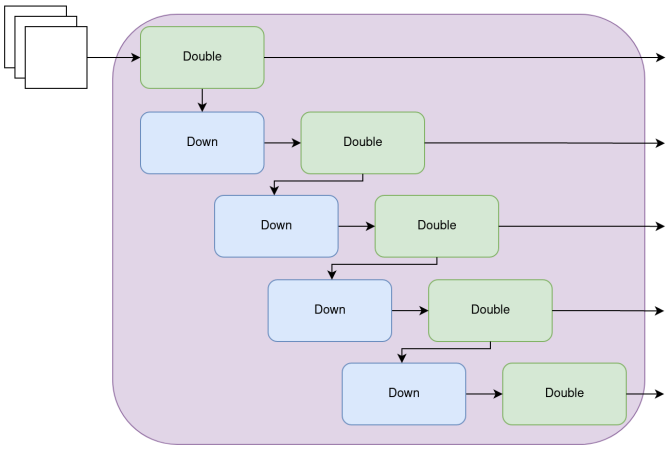

Fig. 5. The encoder module. U-Net like contracting path, using strided convolutions to downsample and double convolutional blocks for better learning. The skip connections (multi-scale view of the input) are represented by the arrows.

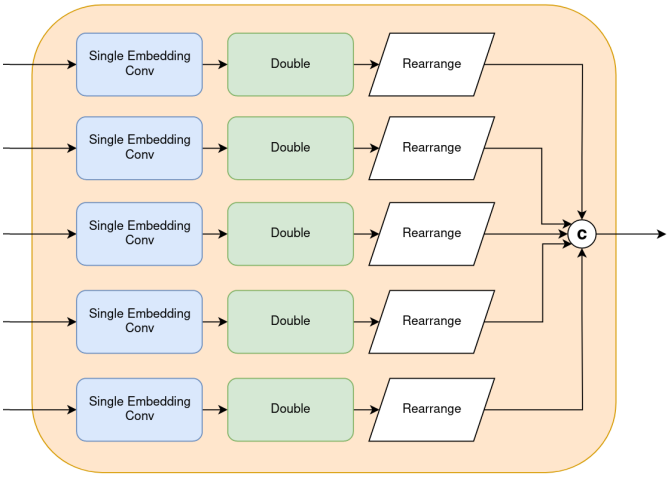

Fig. 6. The embedder module. Uses convolutions to take patches out of the input and create the appropriate amount of channels for the embeddings. Inputs are rearranged for the vision transformer. Every input level has its own set of modules. The double convolution blocks use a kernel size of 1.

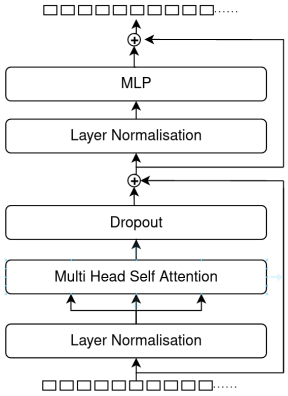

Fig. 7. One transformer block (layer). Tokens are normalized before attention is computed. Residual connections are used before and after the multi layer perceptron. Input and output shapes are the same.

transformer expects. If the patches are of size $(P \times P \times P)$, the flattened tokens would be of size P^3. However, vision transformers usually project these tokens into a bigger embedding dimension. To achieve this shape of data, we have implemented the embedder module that, per input scale, uses a single convolution with a kernel size and stride equal to the patch size. This extracts non-overlapping patches from the feature maps of the encoder. If input has the shape (B, C_1, H, W, D) where B is the batch size, C_1 is the amount of channels, and H, W, D are the spatial dimensions, the output from this single strided convolution will be of the shape $(B, C_2, \frac{H}{P}, \frac{W}{P}, \frac{D}{P})$ where P is the patch size and $C_1 \leq C_2$. We aim to make the channel dimension out embedding dimension, hence the increase in size from C_1 to C_2. Then a block of double convolution with kernel size of one scales the input to have the correct amount of channels, which is the embedding dimension of the network. The result of this operation will have the shape of $(B, E, \frac{H}{P}, \frac{W}{P}, \frac{D}{P})$ where E is the embedding dimension of the network and $C_2 \leq E$. Since the vision transformer expects flat tokens, these five dimensional tensors are reshaped to $(B, (\frac{H}{P} \cdot \frac{W}{P} \cdot \frac{D}{P}), E)$. These are the final tokens that will be the input of the vision transformer. The embeddings from all the scales are concatenated for the vision transformer to interpret as a singular series of tokens. We take the intermediate step of C_1 to C_2 to E for two reasons. Firstly, this just seems to learn and perform better. Secondly, doing this jump in channel count with the patching convolution would make the filter sizes within that operation extremely big and hence slow. We have empirically found out that this way works better than other methods. Figure 6 visualizes this module.

5) Vision Transformer: The vision transformer that we use is quite similar to the default transformer [5] encoder. The transformer is comprised of identical blocks stacked. We use learned positional embedding that are applied to every image level together. The idea behind this is to embed the levels together with the patches so the network can figure out which patches (tokens) are from which resolution level.

Transformer blocks first normalize the given input with a layer normalization. This step is different to the original transformer architecture as is discussed in detail by Xiong et al. [20]. This method has been widely accepted by all vision transformer architectures. Another variation from the original block is the use of flash attention [21]. This is a faster implementation of the multi head-attention mechanism (MHSA) that uses tiling and smart memory algorithms that reduces the complexity and increases the IO efficiency compared to regular MHSA. This enabled the use of smaller patch size in our network since regular attention is $O(N^2)$ with respect to input size and flash attention is almost linear. One layer of the vision transformer is visualized in figure 7.

6) De-Embedder: The de-embdeder of out network follows much of the same wisdom as the embedder. Use smaller filter counts for bigger kernels and chain more convolutions. The idea is exactly the opposite process of the embedder as well, where the input is first reshaped from the token shape to $(B, E, \frac{H}{P}, \frac{W}{P}, \frac{D}{P})$. Then using a transposed convolution block the spatial dimensions are restored to their original shapes of (B, C, H, W, D). Finally the channel counts are restored using a double convolutional block with the correct parameters. Figure 8 visualizes this module.

7) Decoder: The decoder part of this network is also heavily influenced by the U-Net architecture. Transposed convolutions are used to upsample the outputs of the layers below, while residuals are concatenated with these upsampled outputs. Again the basic building blocks of single and double convolutions are used. Figure 9 shows this architecture. The final output channel size is corrected with a double block to achieve the same shape as the labels. The final activation is a softmax layer instead of the PReLu compared to all other blocks.

IV. Dataset

In this section we present an overview of the target dataset. This dataset includes 90 CT scans of pre-operative patients collected from Amsterdam University Medical Center (AmsterdamUMC).

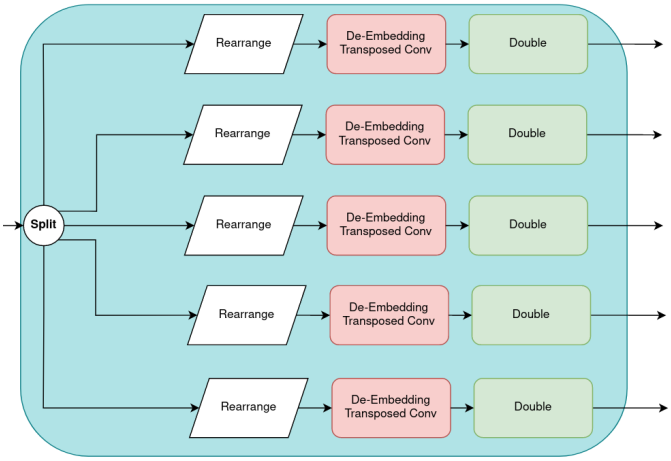

Fig. 8. The de-embedder module. Splits the tokens up into the corresponding levels of downsampling. Uses a rearrange operation with transposed convolutions to recover the shape of the skip connections. These skip connection (multi-scale view) are represented by the outgoing arrows.

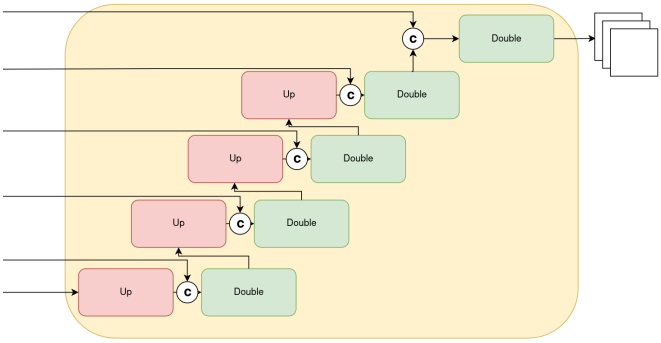

Fig. 9. The decoder module. Resembles the expanding path of a U-Net network. Skip connections (multi-scale views) are concatenate with the upscaled versions of the lover levels. The result is a one-hot encoded segmentation map.

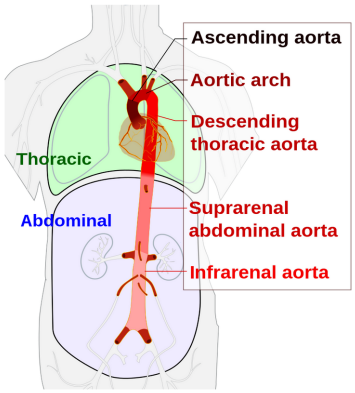

Fig. 10. Sections of the aorta, this dataset only includes annotations (labels) on the abdominal part of the aorta. [22]

TABLE I
LEARNING SETUP CONFIGURATION

Parameter	Used in the setup
Library	PyTorch, MONAI
Number of epochs	1500
Batch size	4
Optimizer	Adam
Loss function	Dice + Cross Entropy
Learning rate scheduling	Lower on plateau
Initial/final learning rates	1e-3 / 1e-5

A. Abdominal Aortic Aneurysms

Abdominal aortic aneurysms (AAA) are segmental, full-thickness dilations of the abdominal aorta exceeding the normal vessel diameter by 50 percent [23]. For the most part, they present no symptoms until the moment they rupture. AAA ruptures are often lethal [23]. Since they are mostly asymptomatic and lethal, they pose a significant challenge to the medical community, as without regular screening, there is no reliable way of identifying patients with aneurysms. In a CT scan of the human body, it is possible to identify aneurysms visually; however, it is more convenient to use segmentation algorithms to determine thicknesses and shapes, as doing this process manually is a tedious process especially if a single patient has many follow-up scans. The CT scans in this dataset are aimed to identify and process infrarenal aorta images. This meas that the aortas are only segmented in the abdomen. Figure 10 identifies where in the human body the infrarenal part of the aorta is located. The scans all have segmentation masks with three classes. The background, lumen (part of the aorta that the blood flows), and thrombus (clotted blood).

B. Labelling and Pre-Processing

The CT scans used in this study are obtained without annotations. In a separate study conducted by Alblas et al. [24], scans are initially manually annotated by human observers. Then, using implicit neural representations, the annotation are reconstructed. In other words, this process 'fills in' the gaps between manually annotated points using an implicit neural representation.

All CT scans in the dataset are resampled to have the same voxel size. We have calculated the target voxel size based on the median value of all scans per axis. These values are (1.2, 1.2, 0.9) with respect to the sagittal, coronal, and axial axes. Furthermore, the scans are cropped with respect to the foreground voxels, removing some of the background.

Figure 11 visualizes an example scan from the dataset that is labelled and pre-processed with the mentioned steps.

V. EXPERIMENTS

A. Setup

All experiments we present in this study are done using a single NVIDIA A40 GPU with 48GB VRAM. All models are trained on the same system with the same data files. The batch size is selected solely on the bases of available VRAM and kept constant in all experiments for comparison. The full configuartion of the learning setup is visible in table I.

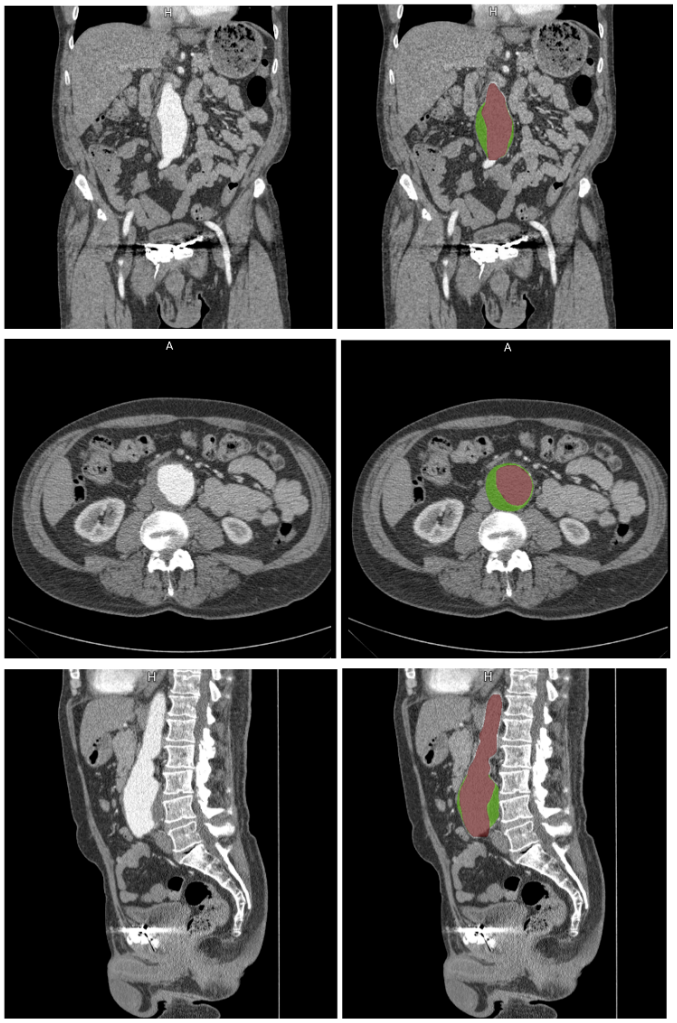

Fig. 11. An example visualization from the dataset. On the left are the original CT scans while on the right are segmentation labels. From top to bottom: coronal, axial, and sagittal views are given with the center voxel fixed in position. The red indicates the lumen, the green segmentation indicates the thrombus.

TABLE II
U-NET CONFIGURATION USED

Parameter	Value
Channels per layer	16, 32, 64, 128, 256
Residual units per layer	3
Strides per layer	2, 2, 2, 2
Normalization method	Instance Normalization
Dropout rate	0.1

B. Baseline models

We have selected two baseline models to compare with. These models are U-Net and Swin UNETR.

1) U-Net: The U-Net implementation used in this study is from MONAI. It uses the same crop size and pre-processing pipeline as the proposed model. The configuration is given in table II.

2) SWINUNETR: The Swin UNETR implementation used in this study is from MONAI. It uses the same crop size and pre-processing pipeline as the proposed model. The configuration is given in table III. As mentioned in the background section, Swin UNETR also takes advantage of a multi-scale representation using a swin transformer. However, this multi-scale representation is not explicitly made outside of the transformer, rather it is a product of it. We chose this model as a comparison model for this reason.

TABLE III
SWIN UNETR CONFIGURATION USED

Parameter	Value
Depths (per layer, MLP layers)	2, 2, 2, 2
Attention heads per layer	3, 6, 12, 24
Feature size	24
Normalization method	Instance Normalization
Dropout rate	0.1

TABLE IV
THE CONFIGURATION OF OUR MODEL

Parameter	Value
Channels per encoder/decoder layer	16, 16, 16, 16, 16
Strides per layer	2, 2, 2, 2
Normalization method	Instance Normalization
Dropout rate	0.1
Transformer embedding dimension	512
Transformer hidden size	2048
Number of attention heads	16
Number of transformer layers	12
Embedding patch size	$8 \times 8 \times 8$

C. Configuration of our model

We have made some design choices for the final configuration of our model based on the criteria of performance with memory as a limiting factor. We found that 16 feature maps per layer in the encoder module was enough for the vision transformer. The full configuration of our model can be found in table IV.

D. Data processing and augmentation

To increase the training capabilities of any model, the data processing (and augmentation) is crucial. Within this work, all models have been trained with exactly the same processing pipeline to ensure comparability. The following transformations are applied to each CT image in the training process. The list is in order of operations.

- Intensity scaling per CT window. All values that are lower than -300 HU are considered 0, all values higher than 300 are considered 1. The remainder is normalized between 0 and 1 linearly.
- Random 3D cropping with crop size of $(128 \times 128 \times 128)$. We have found that over-sampling foreground sections on a one to one basis with background results in better models. So, a crop is either centered on a background or foreground voxel with equal probabilities.
- Random flips in all three spatial axes. 10 percent chance per axis.
- Random intensity shifts per voxel. 10 percent change for a chance of 0.05.

TABLE V
RESULTS

Model	Lumen		Thrombus		Parameters
	Dice	HD (mm)	Dice	HD (mm)	
Ours	0.893	38.7	0.689	73.4	95M
U-Net [3]	0.881	50.7	0.672	53.6	7M
Swin UNETR [15]	**0.916**	**19.3**	**0.796**	**44.2**	16M

E. Inference for testing

In order to test the full capabilities of the models, it is important to segment entire images. We have used sliding window inference with 70 percent overlap. We have also tested a "keep the largest component" method, however, most likely due to the big overlap of the inference, this made almost no difference. Hence it is not used. Inference was performed on the testing set which include 10 scans that are excluded from the training set.

F. Metrics

In this section we outline the metrics that will be used to evaluate the proposed model, the baseline models, and further on the ablation study.

1) Dice score: Dice score [25], formally Sørensen–Dice coefficient, is one of the most used metrics for image segmentation as well as being a loss function with some modifications. It is very closely related to intersection over union, and in fact can be rewritten as such. When comparing two volumes with A as the predicted volume and B as the label, the Dice score can be formalized as:

$$DSC(A,B) = \frac{2|A \cap B|}{|A| + |B|} \tag{2}$$

2) Hausdorff distance: Hausdorff distance is a measure of distance between two sets of points. Given some distance measurement function d, two sets A and B, and points a from A and b from B. $d(a,b)$ measures the distance between two points. Then a smaller definition of Hausdorff can be written as:

$$h(A,B) = \max_{a \in A}\{\min_{b \in B}\{d(a,b)\}\} \tag{3}$$

However, this is a directed measurement since $h(A,B)$ does not have to equal $h(B,A)$. A more general definition is derived as the following:

$$H(A,B) = \max\{h(A,B), h(B,A)\} \tag{4}$$

where the maximum of both distances is taken.

G. Results

The models described in detail above are trained using the same setup as our model with the dataset. Table V shows the metrics calculated on the test set with the trained models. Dice scores (Dice) and Hausdorff distances (HD) are presented per model. The results are separated between the foreground classes of lumen and thrombus.

As we can see from the results, our model does not perform as well as Swin UNETR however it manages to produce slightly better segmentation results than the U-Net with only the Hausdorff distance of thrombus being worse than the U-Net. Considering the configurations of the U-Net, many channels and resnet blocks, this is a nice result to see. It as least means that this method "works". More of the discussion for the results of this experiment is given in the main discussion section.

VI. Ablation Study

In this section the five models that make up the ablation study will be introduced and compared. All of these models have been created to test the strength and the contribution of certain aspects of the proposed architecture. The naming of these models can get confusing so they have been coded by letters to make referencing easier. Figure VI visualizes all these ablation models with the letter codes also present.

A. Models

1) Half-half (A): This model is a variation on the proposed architecture in an attempt to improve its performance. Half of the channels in the skip connections are fed into the vision transformer while the other half are not altered by the vision transformer but concatenated to the outputs of it instead.

2) U-net like (B): This model aims to evaluate the actual contribution of the vision transformer by removing it. The resulting model is a U-Net like encoder/decoder model.

3) Autoencoder (C): This model removes the skip connection between the encoder and the decoder of model B.

4) With embeddings (D): This model only removes the vision transformer while keeping the embedder and de-embedder modules in place. The goal of this model is to measure the impact of the embedding/de-embedding. We will compare the proposed architecture, model B, and this model to evaluate the leraned mebedding approach.

5) Single-scale ViT (E): This model does not utilize the lower scales of the input. It aims to demonstrate the difference the multi-scale view makes. For the embedding of the single level, the same embedding / de-embedding module is used.

B. Results

Table VI shows the results of all the ablation models with identical training setups. Our original model performs better than all the ablation models with the half-half (A) and U-Net like (B) performing similar. The rest of the models lack the ability to segment the images to a degree that our model does. Finally, for ablation models C, D, and E, using a "keep the largest component" function is used in 3D. These models produce outputs that have big components far away from the target area. For the remaining ablation models, this function again made no difference in scores. A discussion of these results is given in the discussion section of this report. Figure 13 shows the segmentation of one of the test cases with all the ablation models over 3 different slices.

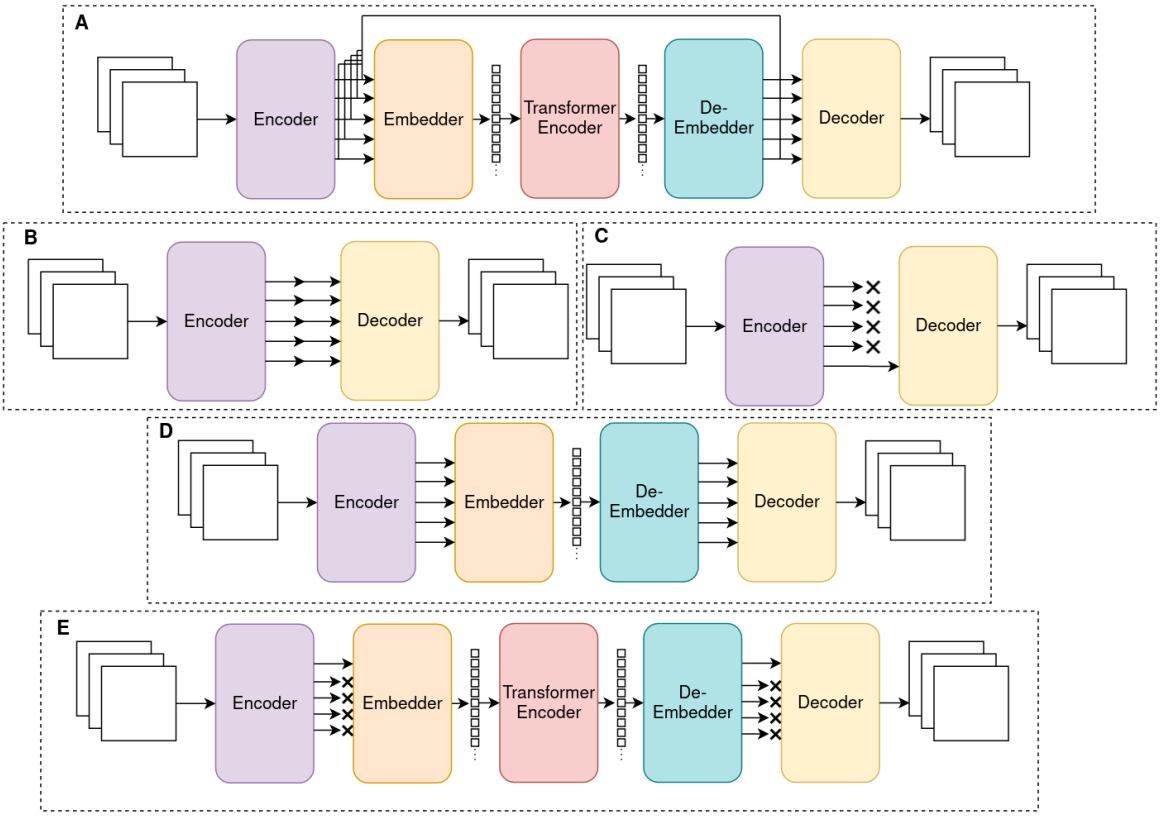

Fig. 12. All the ablation models created for this study. A: "half-half", is variation that uses half of the encoder channels in the vision transformer path and the other half is concatenated after the vision transformer path. B: "U-Net like" basically connects the encoder and the decoder immediately. C: "autoencoder". D: "with embeddings" adds the embedder/de-embedder couple to the U-Net like model. Aims to test the effect of the embeddings on the network. E: "single-scale ViT" presents almost the same architecture, however, only the non downsampled version of the input is patched and used in the rest of the network.

TABLE VI
FINAL PERFORMANCES OF THE ABLATION MODELS

	Lumen		Thrombus		
Model	DSC	HD	DSC	HD	Parameters
Proposed	**0.893**	**38.7**	0.689	73.4	95M
Half half (A)	0.869	42.6	0.670	61.8	55M
U-Net like (B)	0.790	62.2	**0.759**	**56.6**	185K
Autoencoder (C)	0.769	70.6	0.714	58.9	89K
With embeddings (D)	0.420	230	0.399	255	42M
Single-scale ViT (E)	0.560	125	0.518	144	47M

VII. DISCUSSION

Throughout this study, we have explored the integration of vision transformers with U-Net like architectures for medical image segmentation. Our work has been made with the hypothesis that a multi-scale representation of the input can enhance the performance of vision transformers in such a task. We found that a multi-scale representation indeed improves the segmentation performance of a vision transformer. In this section we will discuss the various aspects of our study and the results achieved.

A. Results compared to baselines

As indicated in Table V, our proposed model may not have surpassed the state-of-the-art performance in the given segmentation task, yet it has still delivered a higher quality result than a contemporary U-Net. Swin Transformers have proven to be quite efficient in imaging tasks, and the internal multi-scale representation that Swin UNETR creates appears to offer a more suitable multi-scale approach for vision transformers. It's important to note that the primary objective of this study was not to outperform Swin UNETR, but to assess the effectiveness of multi-scale representation with vision transformers. However, considering the parameter count of Swin UNETR (approximately 15 percent of our model), Swin Transformers seem to be the superior backbone network available.

B. Comparison of the ablation models

As our ablation study demonstrates, none of the variations to our model outperformed the original. This is an encouraging discovery as it supports the hypothesis of using multi-scaled input for vision transformer. Beginning with the U-Net like model (B), it can be inferred that the inclusion of the embedder/de-embedder and the vision transformer together seems to contribute to a more successful model. In other

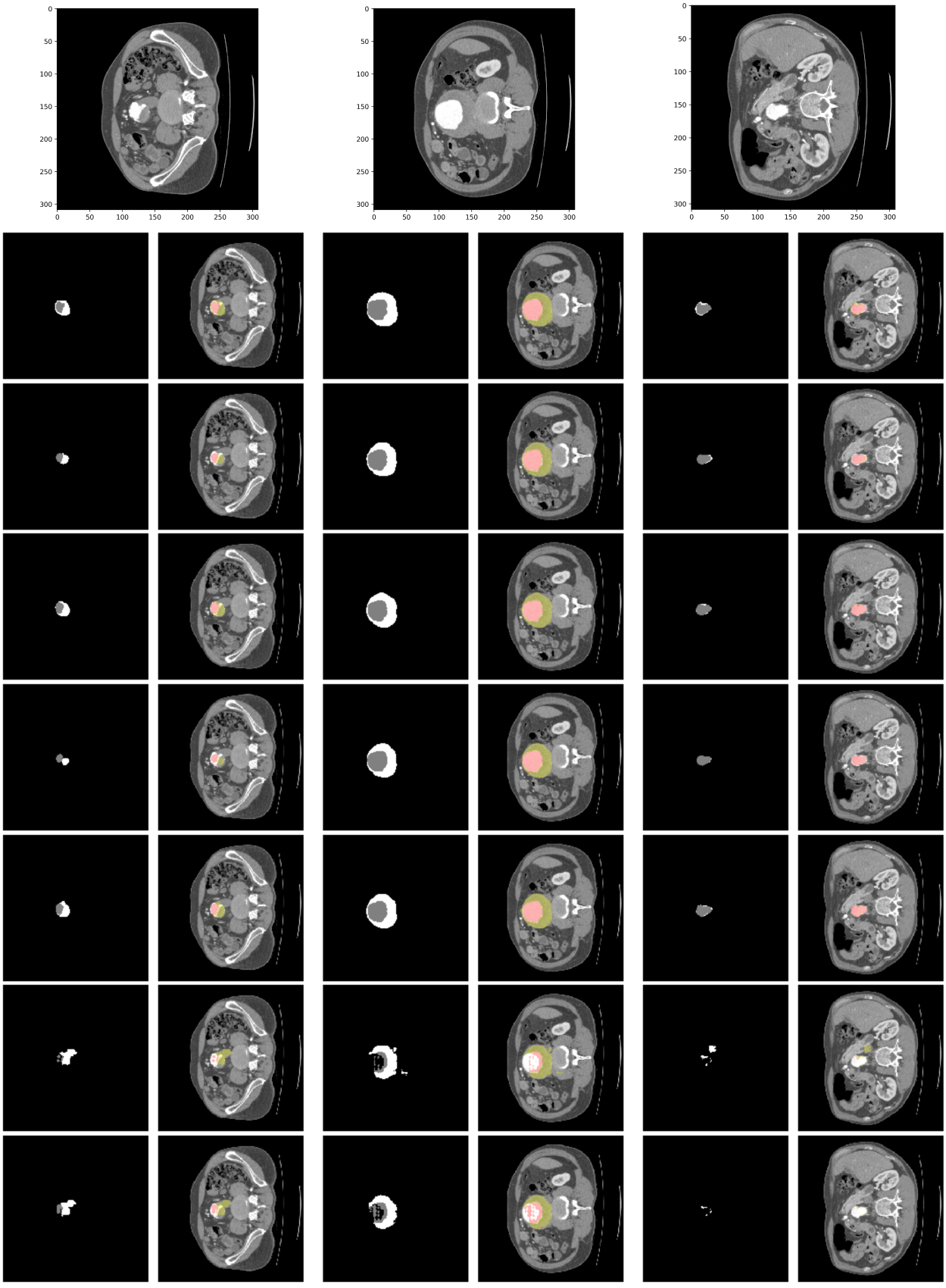

Fig. 13. An example segmentation from the test set with all the models mentioned in this section and the label. 3 different slices from the same scan are shown. On top the original slice without any annotations. From top to bottom: label, proposed, half half (A), U-Net like (B), autoencoder (C), with embeddings (D), single-scale ViT (E).

words, the surrounding structure of the transformer can't take full credit for the final result. Examining the single scale variant (E), we notice a significant decrease in performance compared to our model, suggesting that the transformer alone also can't take full credit. It's the combination of these elements that has produced the best performing model we've created. This also strongly supports the multi-scale approach to tokenization.

The notably poor result with the addition of the embedder/de-embedder without the vision transformer, as shown by ablation model D, points to the potential weakness of this model. Its performance is remarkably lower than all the other ablation models. Furthermore, the embedder/de-embedder duo accounts for 50 million parameters, over half of the network. This can be interpreted as follows: With the current state of the model, more than half of the learnable parameters are used merely to reshape the input to suit the vision transformer. This is certainly not ideal and should be addressed in future research.

C. Dataset Limitations

All scans included in the dataset are from pre-operative patients. The images feature the abdominal section of the aorta (see Figure 10), meaning that the dataset does not capture or segment the vessel's entire length. The datasets objective is to represent a specific type of AAA that occurs only in the infrarenal region (situated or occurring below the kidneys). However, a limitation of this dataset is the sharp borders in the segmentation maps where the labeling ends, even though the aorta visibly continues. A clear discontinuity in the segmentation map is visible in Figure 11. Another limitation of this dataset is number of samples. 90 scans is enough to train deep learning models, however, more is always appreciated. For example, in this dataset there are only a few patients without a thrombus (see figure 11), while most healthy humans do not present one. This makes many models biased towards segmenting a thrombus, even if there isn't one.

D. Future Work

In this paper we presented a novel architecture using a vision transformer as a backbone. However, there are still quite a lot of improvements that can be applied to this network to potentially achieve better results.

1) Attention Filtering: This model has been tested on a dataset where the foreground (mask labels that are not the background) occupy relatively little space in the whole image. This means that there are a lot of patches that the vision transformer needs to correctly label as all background. This issue could possible be fixed by a filtering system, where the attention of each token, possibly cross-attention with a bigger token (from a higher level of the tree), is leveraged. A successful implementation of this strategy would have two benefits. Firstly, all the parameters of the transformer would be deployed to more accurately segment foreground tokens. Secondly, since background tokens are filtered out, these would be immediately segmented as fully background, creating cleaner segmentations.

2) Separate Positional Encoding: In this work we have only used learned positional encoding. The assumption is that the model should learn which parts of the input and the multiple levels of input is where by itself. However experimenting with more explicit ways of positional encoding where the level information is already given to the transformer is a possible research path.

VIII. Conclusion

Throughout this study, we have explored the integration of vision transformers with U-Net like architectures for medical image segmentation. Our work has been made with the hypothesis that a multi-scale representation of the input can enhance the performance of vision transformers in such a task. We have shown a strong argument for this hypothesis with this work and believe answered it in a positive way.

Our proposed model, while not surpassing the state-of-the-art performance, still showed promising results, performing better than a contemporary U-Net. The ablation study provided valuable insights into the importance of different components of our architecture. Neither the U-Net like components nor the vision transformer could independently take full credit for the model's performance. Instead, it was their combination that led to the most efficient model we created.

However, we also identified potential areas of improvement. The embedder/de-embedder duo, responsible for reshaping the input to suit the vision transformer, accounted for more than half of the learnable parameters. Future research should focus on addressing this issue to further optimize the model.

In conclusion, this study contributes to the ongoing research in medical image segmentation, affirming the potential of vision transformers when used in conjunction with multi-scale representations. Our findings are a step forward in the pursuit of more efficient and effective models for this critical task, paving the way for future explorations in this direction.

References

[1] K. Suzuki, "Overview of deep learning in medical imaging," *Radiological Physics and Technology*, vol. 10, no. 3, pp. 257–273, Jul. 2017. [Online]. Available: https://doi.org/10.1007/s12194-017-0406-5

[2] M. P. Starmans, S. R. van der Voort, J. M. Castillo Tovar, J. F. Veenland, S. Klein, and W. J. Niessen, "Chapter 18 - radiomics: Data mining using quantitative medical image features," in *Handbook of Medical Image Computing and Computer Assisted Intervention*, ser. The Elsevier and MICCAI Society Book Series, S. K. Zhou, D. Rueckert, and G. Fichtinger, Eds. Academic Press, 2020, pp. 429–456. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128161760000235

[3] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015. [Online]. Available: https://arxiv.org/abs/1505.04597

[4] R. Wang, T. Lei, R. Cui, B. Zhang, H. Meng, and A. K. Nandi, "Medical image segmentation using deep learning: A survey," *IET Image Processing*, vol. 16, no. 5, pp. 1243–1267, jan 2022. [Online]. Available: https://doi.org/10.1049%2Fipr2.12419

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017. [Online]. Available: https://arxiv.org/abs/1706.03762

[6] S. H. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in vision: A survey," *CoRR*, vol. abs/2101.01169, 2021. [Online]. Available: https://arxiv.org/abs/2101.01169

[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 2020. [Online]. Available: https://arxiv.org/abs/2010.11929

[8] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[9] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *CoRR*, vol. abs/2001.05566, 2020. [Online]. Available: https://arxiv.org/abs/2001.05566

[10] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[11] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *CoRR*, vol. abs/1901.06032, 2019. [Online]. Available: http://arxiv.org/abs/1901.06032

[12] D. Ciresan, A. Giusti, L. Gambardella, and J. Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," *Advances in neural information processing systems*, vol. 25, 2012.

[13] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[14] N. Siddique, P. Sidike, C. Elkin, and V. Devabhaktuni, "U-net and its variants for medical image segmentation: A review of theory and applications," *IEEE Access*, vol. PP, pp. 1–1, 06 2021.

[15] A. Hatamizadeh, V. Nath, Y. Tang, D. Yang, H. Roth, and D. Xu, "Swin unetr: Swin transformers for semantic segmentation of brain tumors in mri images," 2022.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[17] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[18] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 1501–1510.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[20] R. Xiong, Y. Yang, D. He, K. Zheng, S. Zheng, C. Xing, H. Zhang, Y. Lan, L. Wang, and T.-Y. Liu, "On layer normalization in the transformer architecture," 2020.

[21] T. Dao, D. Y. Fu, S. Ermon, A. Rudra, and C. Ré, "FlashAttention: Fast and memory-efficient exact attention with IO-awareness," in *Advances in Neural Information Processing Systems*, 2022.

[22] Wikipedia, "Aorta — Wikipedia, the free encyclopedia," http://en.wikipedia.org/w/index.php?title=Aorta&oldid=1137563110, 2023.

[23] K. C. Kent, "Abdominal aortic aneurysms," *New England Journal of Medicine*, vol. 371, no. 22, pp. 2101–2108, 2014, pMID: 25427112.

[24] D. Alblas, C. Brune, K. K. Yeung, and J. M. Wolterink, "Going off-grid: Continuous implicit neural representations for 3d vascular modeling," 2022.

[25] T. Sørenson, *A Method of Establishing Groups of Equal Amplitude in Plant Sociology Based on Similarity of Species Content and Its Application to Analyses of the Vegetation on Danish Commons*, ser. Biologiske skrifter. I kommission hos E. Munksgaard, 1948. [Online]. Available: https://books.google.nl/books?id=rpS8GAAACAAJ