THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

Improved Unet model for Medical Image Segmentation Tasks

DAVID SALDUBEHERE
SPRING 2025

A thesis
submitted in partial fulfillment
of the requirements
for a baccalaureate degree
in Computer Science
with honors in Computer Science

Reviewed and approved* by the following:

Huijuan Xu
Assistant Professor of Computer Science
Faculty Reader

Mohamed Almekkawy
Associate Professor of Computer Science
Thesis Honors Advisor and Thesis Supervisor

*Electronic approvals are on file.

Abstract

The Unet encoder decoder architecture has become a staple for computer vision tasks in the medical imaging field. In an effort to enhance the performance of these networks, many have tried to adapt other deep learning techniques for computer vision. In this vein, I propose a novel encoder decoder architecture that relies on graph based representations of images instead of the typical 2D pixel format. Images are decomposed into patches which are then efficiently organized to form a graph with patches as nodes. This representation allows adopting networks to manipulate data to conform to irregular structures present in medical images, a capability not possible with grid based methods. Additionally, I propose a novel graph refinement module to remove the residuals of predicted segmentation boundaries commonly present with complex objects. Previous research and my experiments illustrate the potential of this architecture and auxiliary module.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to thank Ahmed Al-Qurri for his supportive technical expertise and my advisors for taking the time to review this work.

Chapter 1
Purpose and Motivation

1.1 Deep Learning and Medical Imaging

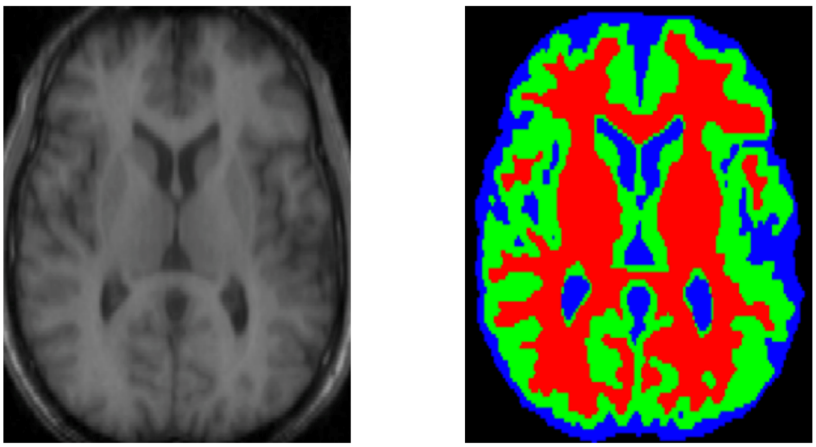
Deep learning has a rich history in the field of medical imaging. Since the success of AlexNet, a large deep learning model that achieved incredible performance in the ImageNet Large-Scale Visual Recognition Challenge in 2012, deep learning has grown in popularity for computer vision applications [1]. Applications of these models have even surpassed physicians in several tasks [8]. In particular, the deep learning framework known as the Unet architecture is the adopted standard for medical image semantic segmentation [21].

1.1.1 Semantic Segmentation in the Context of Computer Vision

There are several types of computer vision tasks that can be performed on medical images. Segmentation refers to the process of finding boundaries between objects in an image as opposed to classification, which is the process of assigning a label to an object wherever it may be in the image. Semantic segmentation achieves this by classifying each pixel in an image therefore providing the necessary information to construct object boundaries.

1.1.2 Specific Applications of Semantic Segmentation in Medicine

Semantic segmentation has been used in radiotherapy planning, brain tissue segmentation, and many other medical applications due to its efficiency and accuracy compared to physicians [20]. In some cases, deep learning is replacing traditional algorithmic methods of segmentation. FreeSurfer, a popular software for automated processing of structural brain MRIs, is one such example. FastSurfer is a clone of FreeSurfer, but its segmentation



(a) Axial slice (b) Tissue segmentation

Figure 1.1. An example of semantic segmentation on a MRI image with tissue boundaries identified and displayed

engine is driven by a deep learning model enabling much faster inference and higher accuracy [9, 10, 12, 13]. Semantic segmentation is not limited to MRI images. A variety of data modalities can be involved in medical image segmentation tasks X-ray, Positron Emission Tomography (PET), Computed Tomography (CT), Magnetic Resonance Imaging (MRI), and Ultrasound (US) [26]. Our research focuses specifically on MRI and US datasets, but there is significant evidence that methods capable of performing well on tasks will generalize well to other clinical tasks [2]. Therefore, my goal is to provide a flexible and robust model to enhance the medical field through its application with semantic segmentation.

Chapter 2
Background and Prior Work

2.1 Vision Transformers

2.1.1 Introduction

Prior Unet architectures utilize convolutional neural networks with a sequence of down-sampling layers for feature extraction and a sequence of upsampling layers to restore the spatial dimensions and classify or perform segmentation. With the introduction of the transformer architecture, new networks implement a vision transformer by creating a sequence of patches constructed from pixels to utilize the attention mechanisms while maintaining the encoder decoder structure for feature extraction. Due to the nature of typical convolutional neural networks, they have a small receptive field. The attention mechanism of a vision transformer provides a large receptive field allowing for networks to learn associations between components across the entire image [7].

2.1.2 Implementations of Vision Transformers

SwinNet and TransNet, two popular vision transformer networks, surpass the performance of standard Unet architectures. Both networks utilize the Unet shape and correspondingly describe their proposed encoder and decoder structures. The differentiator is their implementation and placement of the vision transformer [4,5]. They also both conveniently were trained and tested on the same dataset, the Synapse multi-organ CT dataset, which provides a useful comparison tool.

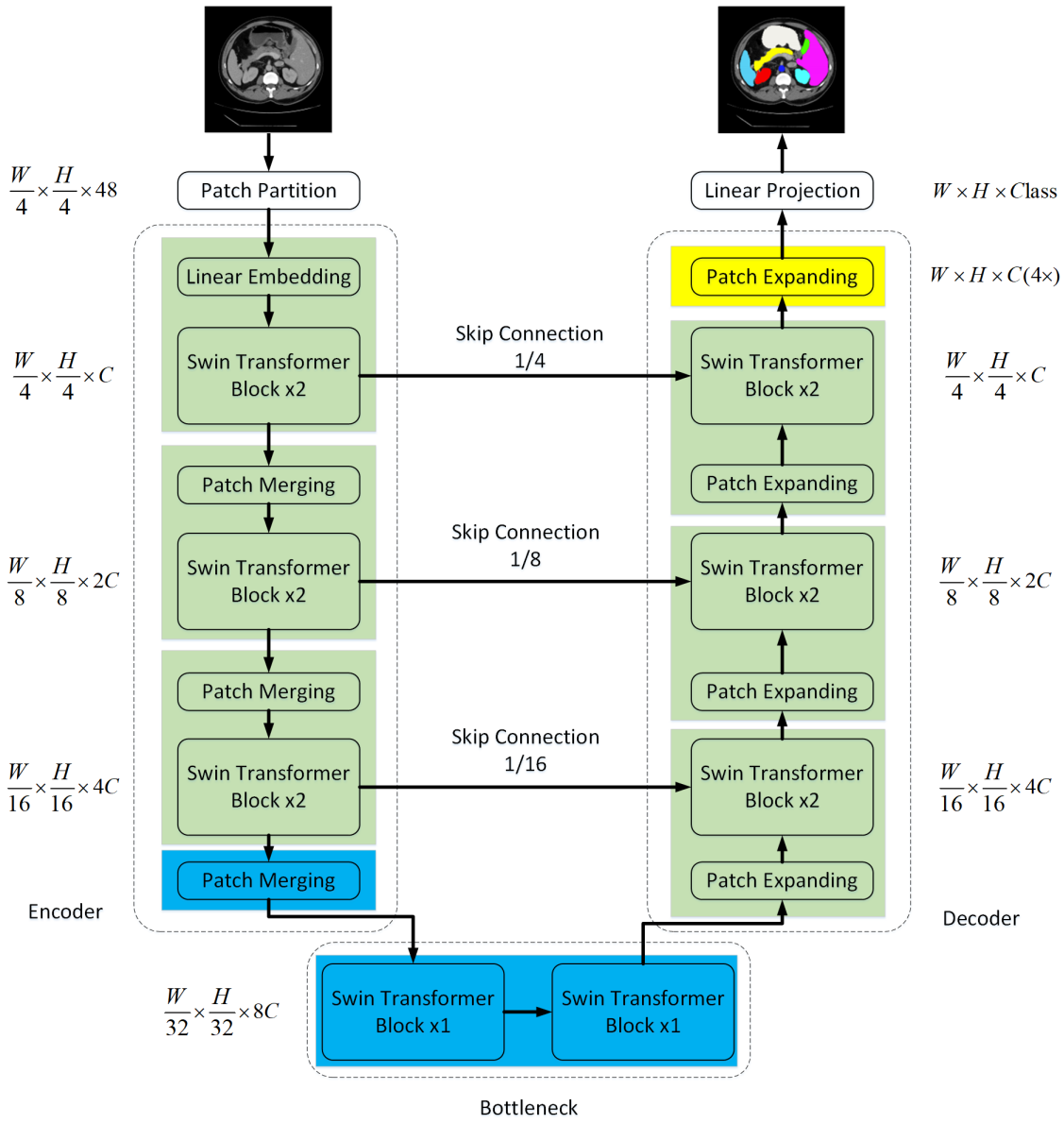


Figure 2.1. The network architecture of SwinUnet showcasing the encoder and decoder structure

2.2 Vision Graph Neural Networks

2.2.1 Graph Construction

In this same vein, Han repurposed graph networks typically used for point cloud segmentation and introduced a similar patch mechanism utilizing a K nearest neighbor algorithm to connect the image patches based on a similarity threshold. These patches then form a

graph where similar patches are connected. The degree of a patch is determined by the hyperparameter K [11].

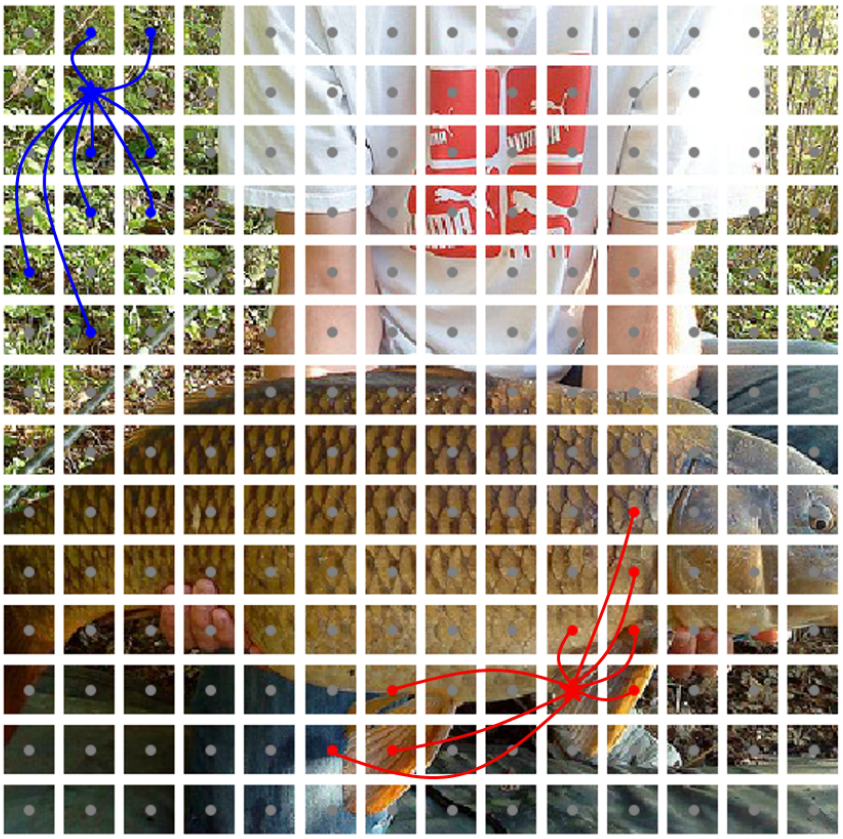


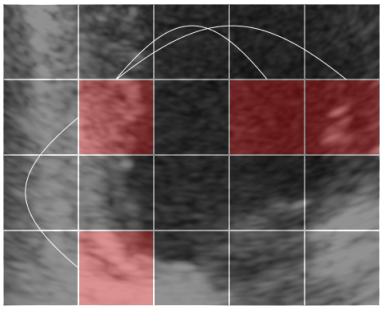
Figure 2.2. Example of two nodes in a graph constructed from an image patches with $K = 8$

2.2.2 Limitations of Original Vision Graph Neural Networks

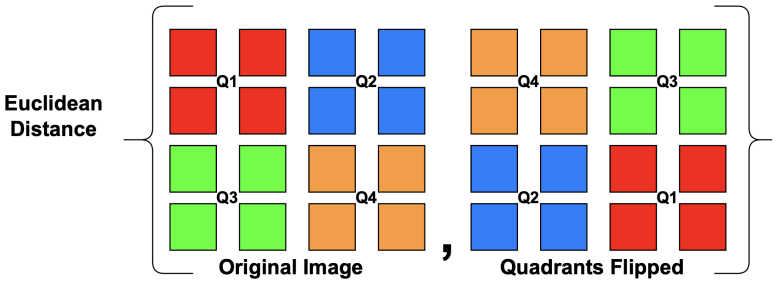
This implementation of a vision graph neural network has shown promise, but the graph construction serves as a performance bottleneck. Efforts to reduce this burden have focused on alternatives to the K nearest neighbor operation. Munir experimented with a static graph construction in the development of MobileViG and an efficient dynamic axial graph construction (DAGC) method in the updated GreedyViG model. Munir's proven ability to achieve superb performance with the fairly lightweight DAGC motivated me to select this method for graph construction in my proposed model. Static graph construction is fixed for all images and does not adapt to the training data. DAGC is dynamic and requires a new graph representation per data point via the same greedy K nearest neighbor algorithm. Despite this burden, DAGC only considers patches within

the same column or row as potential candidates for connections created via the K nearest neighbor algorithm effectively squeezing out additional performance gains from the cumbersome original VGNN model.

To determine the thresholds for connections Munir also implements an efficient algorithm that compares an image segmented into quadrants against the transpose of the quadrants [22, 23]. The full algorithm combines these two methods to create an efficient



(a) DAGC with $K = 3$



(b) Figure illustrating the base image transposition

and flexible graph construction method that I will be using as the core of my proposed model.

2.2.3 Full Picture

Munir then organized several DAGC operations alongside sequential graph convolutions to form the GreedyViG image segmentation network [23]. We expand upon this work by implementing these operations in a Unet architecture. Some efforts to introduce graphical neural networks (GNNs) components to Unet architectures have been successful, but there has not been any attempt to merge state-of-the-art Unet architectures with GNNs components. Jiang proposed a basic Unet structure using the original inefficient K nearest neighbor graph construction, but this barebones model lacked advanced modules found in many other modern Unet models [15].

Algorithm 1 DACG Construction Algorithm

Require:

K: Distance between connections
H, W: Image resolution
X: Input image
$X_{quadrants}$: Quadrants of the input flipped across the diagonals

$m \leftarrow 0$ ▷ Initialize roll distance counter
$norm \leftarrow \texttt{norm}(X, X_{quadrants})$ ▷ Matrix norm of tensors
$\mu \leftarrow \texttt{mean}(norm)$ ▷ Mean of norm values
$\sigma \leftarrow \texttt{std}(norm)$ ▷ Standard deviation of norm values
$X_{final} \leftarrow 0$ ▷ Initialize output tensor

/* Vertical Connection Processing */

while $mK < H$ **do**
  $X_{rolled} \leftarrow \texttt{roll\_down}(X, mK)$ ▷ Roll image down by mK pixels
  $dist \leftarrow \texttt{norm}(X, X_{rolled})$ ▷ Compute distance
  if $dist < \mu - \sigma$ **then** ▷ Generate mask based on distance threshold
    $mask \leftarrow 1$
  else
    $mask \leftarrow 0$
  end if
  $X_{down} \leftarrow mask \cdot (X_{rolled} - X)$ ▷ Extract relevant features
  $X_{final} \leftarrow \texttt{max}(X_{down}, X_{final})$ ▷ Keep maximum values
  $m \leftarrow m + 1$ ▷ Increment roll distance
end while

/* Horizontal Connection Processing */

$m \leftarrow 0$ ▷ Reset roll distance counter
while $mK < W$ **do**
  $X_{rolled} \leftarrow \texttt{roll\_right}(X, mK)$ ▷ Roll image right by mK pixels
  $dist \leftarrow \texttt{norm}(X, X_{rolled})$ ▷ Compute distance
  if $dist < \mu - \sigma$ **then** ▷ Generate mask based on distance threshold
    $mask \leftarrow 1$
  else
    $mask \leftarrow 0$
  end if
  $X_{right} \leftarrow mask \cdot (X_{rolled} - X)$ ▷ Extract relevant features
  $X_{final} \leftarrow \texttt{max}(X_{right}, X_{final})$ ▷ Keep maximum values
  $m \leftarrow m + 1$ ▷ Increment roll distance
end while

return $\texttt{Conv2d}(\texttt{Concat}(X, X_{final}))$ ▷ Return processed features

2.3 Residual Refinement Modules

In my proposed model, I include a novel residual refinement module. I took inspiration from BASNet, a boundary-aware salient object detection network, that utilizes a refinement module to focus on crisp and accurate boundaries for challenging image segmentation datasets. This refinement module, referred to as a Residual Refinement Module (RRM), outputs final pixel level classification as the simple result of tensor addition of the predicted residuals and the original classification head output:

$$S_{\text{refined}} = S_{\text{coarse}} + S_{\text{residual}} \tag{2.1}$$

Testing of BASNet showcased the model's ability to eliminate ambiguities at complex segmentation borders [24].

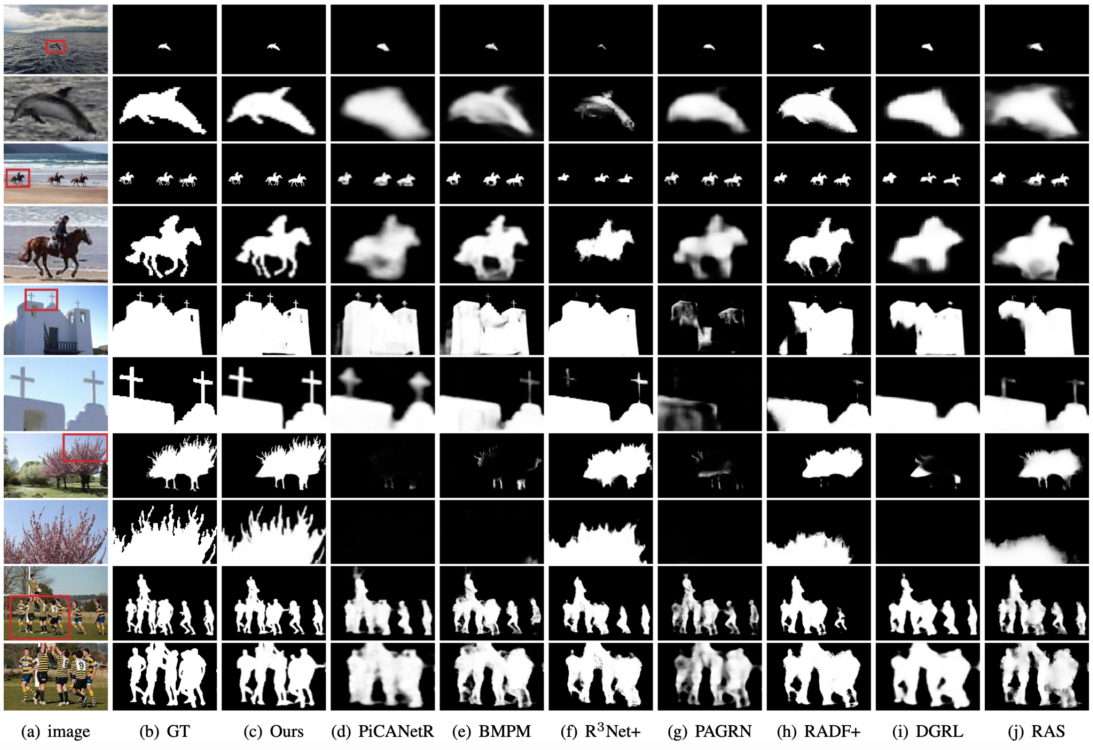


Figure 2.3. Collage of images showcasing the exceptional performance of BASNet

In summary, my proposed model introduces the following novel contributions:

1. A model architecture that utilizes graph neural networks for image operations in a state of the art encoder-decoder structure to utilize recent advancements made in both graph networks and Unet models.

2. A graphical residual refinement model to enhance coarse segmentation borders produced by previous layers.

Chapter 3
Network Design

3.1 Proposed Unet Overview

3.1.1 Encoder

The encoder portion of the model is 5 stages. The first stage creates the initial tensor from the input image to create 64 channels from the single input channel. The following stages consist of multiple graph construction blocks, graph convolution operations, and inverted residual connections. These hyperparameters can be changed to influence the behavior of the encoder and subsequently increase or decrease the amount of total model parameters. For the proposed model, I used 4 graph construction blocks and 4 graph convolution operations per encoder stage. The K parameter used in the K nearest neighbor operation is fixed per stage with values 8, 4, 2, 1 from first to last stage. Via ablation studies, Murri found that keeping the graph too sparse at lower resolutions can negatively impact accuracy [23]. Keeping K proportional to spatial dimension maintains the relative sparsity. Between each encoder stage there is a downsample to half the input spatial resolution and expand the channel dimension. This downsampling is done via 3 x 3 convolution; this could be done differently, but I chose the simplest method available.

3.1.2 Decoder

The decoder portion of the model encapsulates 4 stages. The first stage is the bridge connection that simply copies the final encoder output. Each subsequent stage uses bilinear upsampling to double to spatial dimension with a final output head to match the desired output image dimensions if needed. The channel dimension is handled slightly differently due to the nature of the skip connections which will be discussed in the next

section. This final output is then transmitted to the refinement module which will also be discussed in further detail.

3.1.3 Skip Connections

Another noteworthy aspect of my Unet model is the usage of full-scale skip connections introduced and tested by Huang in their Unet3+ model. Full scale skip connections were developed to address the limitations of existing skip connections and capture signals from varied scales in the hopes of providing the model with a greater ability to learn both the position and the boundary of objects that appear in various scales. This introduces an important invariance to scale allowing the model to correctly segment objects even if they appear in different sizes throughout the training data. Each decoder stage is connected to all other scales via a method of first scaling all previous encoders and decoder stages to a uniform size via maxpool or bilinear upsampling operations and then aggregation to produce a single tensor that is the final output for a specified decoder stage. This can be expressed concisely: let i index each stage of the encoder and N be the total number of stages. $C()$ denotes convolution and $H()$ denotes the feature aggregation. $D(), U()$ denote downsample and upsample operations respectively [14].

3.1.4 Inverted Residual Connections and Depthwise Separable Convolutions

I also included the inverted residual connections and depthwise separable convolutions as proposed in the GreedyViG model. Due to the model's inherent high computational overhead due to the graph networks, I chose to use depthwise separable convolutions to reduce the number of parameters. Depthwise separable convolutions reduce the computational cost while maintaining most of the abilities of traditional convolutions. In other words, they are more efficient convolution operators [6]. Inverted residual connections were also included for efficiency concerns. These blocks are inter-block skip connections that provide the context enhancements of traditional skip connections while using fewer parameters due to its inverted structure that reduces the channel dimension

before applying the depthwise separable convolution [3].

$$X_{De}^{i} = \begin{cases} X_{En}^{i}, & i = N \\ \mathcal{H}\left(\left[\underbrace{C(\mathcal{D}(X_{En}^{k}))_{k=1}^{i-1}}_{Scales:1^{th}\sim i^{th}}, \underbrace{C(X_{En}^{i}), C(\mathcal{U}(X_{De}^{k}))_{k=i+1}^{N}}_{Scales:(i+1)^{th}\sim N^{th}}\right]\right), & i = 1, \cdots, N-1 \end{cases} \tag{3.1}$$

3.1.5 Deep Supervision

Another important feature of my model is support for deep supervision mechanisms. Deep supervision, also known as auxiliary supervision, is a technique that allows additional model outputs to be fed into a chosen loss function. My model contains 3 deep supervised outputs in addition to the final output of the refinement module. These outputs are supplied from the 2nd, 3rd, and 4th decoder stages. Deep supervision has been shown to act as a regularization agent for smaller datasets and can help reduce the depth of these Unet networks alleviating complexity and concerns of vanishing gradients [19].

3.2 Refinement Module

3.2.1 Overview

My novel refinement module learns a single output for each pixel. Originally, each pixel would be classified based on the highest ranking for each class, and the refinement module learns what offset must be applied to these rankings to produce the correct ranking matrix.

3.2.2 Structure

The flow of this module is simple: the final output of the Unet is passed directly into the refinement module. The refinement module stores this initial input, computes the offset, and then applies the offset to the initial input to produce the final output. To keep the component simple and lightweight, the channel dimension is maintained at a constant 64 channels similar to the BASNet refinement module. The refinement module itself forms a Unet structure with 5 stages in the decoder and encoder. The input dimension is halved in each stage of the decoder and upsampled with bilinear upsampling in the decoder. The core piece of my refinement module is the inclusion of

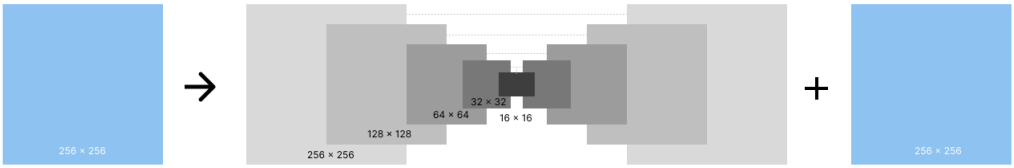


Figure 3.1. Refinement module architecture

graph construction and graph convolution blocks in the encoder as opposed to standard convolution operations. I utilized a K value of 4 for the initial stage and $K = 2$ for the remaining graph constructions, as the spatial resolution was just 16×16 at the bridge connection for a standard 256×256 input image. Ablation studies show that this component increased my model's capabilities across various metrics. Standard skip connections between corresponding encoder and decoder stages are also used.

Chapter 4
Training and Results

4.1 Datasets

4.1.1 CAMUS Dataset

The Cardiac Acquisitions for Multi-structure Ultrasound Segmentation (CAMUS) dataset was used for training and evaluation of this model. The dataset contains two and four-chamber acquisitions from 500 patients with reference measurements from one cardiologist on the full dataset and from three cardiologists on a fold of 50 patients acquired at the University Hospital of St Etienne. The full dataset was acquired from GE Vivid E95 ultrasound scanners, with a GE M5S probe. The dataset contains images of End-Diastole (ED) and End-Systole (ES) for each patient. The original image size of 512x512 was reduced to 256x256 for standardization purposes. No data augmentations were performed on the CAMUS dataset. This dataset contains 4 segmentation target classes [18]. NiBabel was utilized to load the dataset NIfTI files into 1 channel tensors.

4.2 Network Training

Training was performed using PyTorch 2.5.1, torchvision 0.20.1, and timm 1.0.15 in an Anaconda 24.11.1 environment. A single A6000 GPU was used for training and testing of all models. Training was done via the Adam optimizer with a learning rate of 1e-3 [16]. Standard Cross Entropy Loss was used for the training loss. Due to the VRAM requirements of my model, I kept the batch size low at 4. To foster reproducibility, I used a random seed of 42 whenever randomization was required.

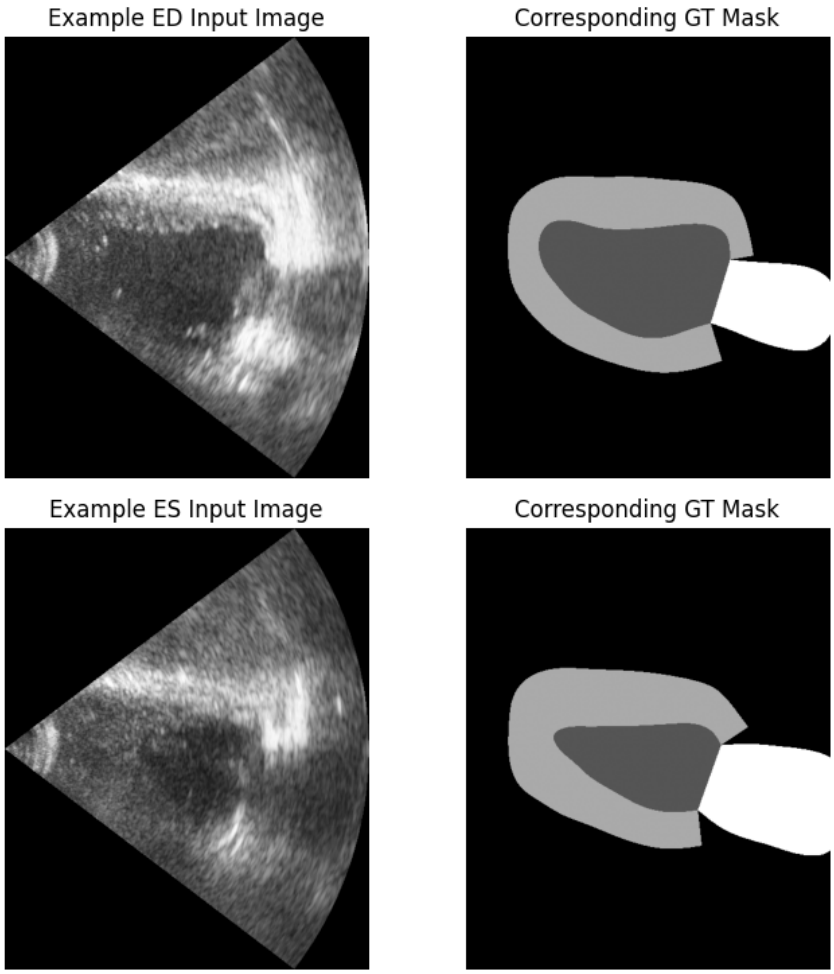


Figure 4.1. Example figure of input image and GT mask

4.3 Experiment Results

Testing was performed by averaging the DICE and IOU metrics across each of the 5 folds produced via 5-fold cross validation.

$$\text{DICE} = \frac{2|X \cap Y|}{|X| + |Y|} \tag{4.1}$$

$$\text{IOU} = \frac{|X \cap Y|}{|X \cup Y|} \tag{4.2}$$

for sets of pixels X and Y.

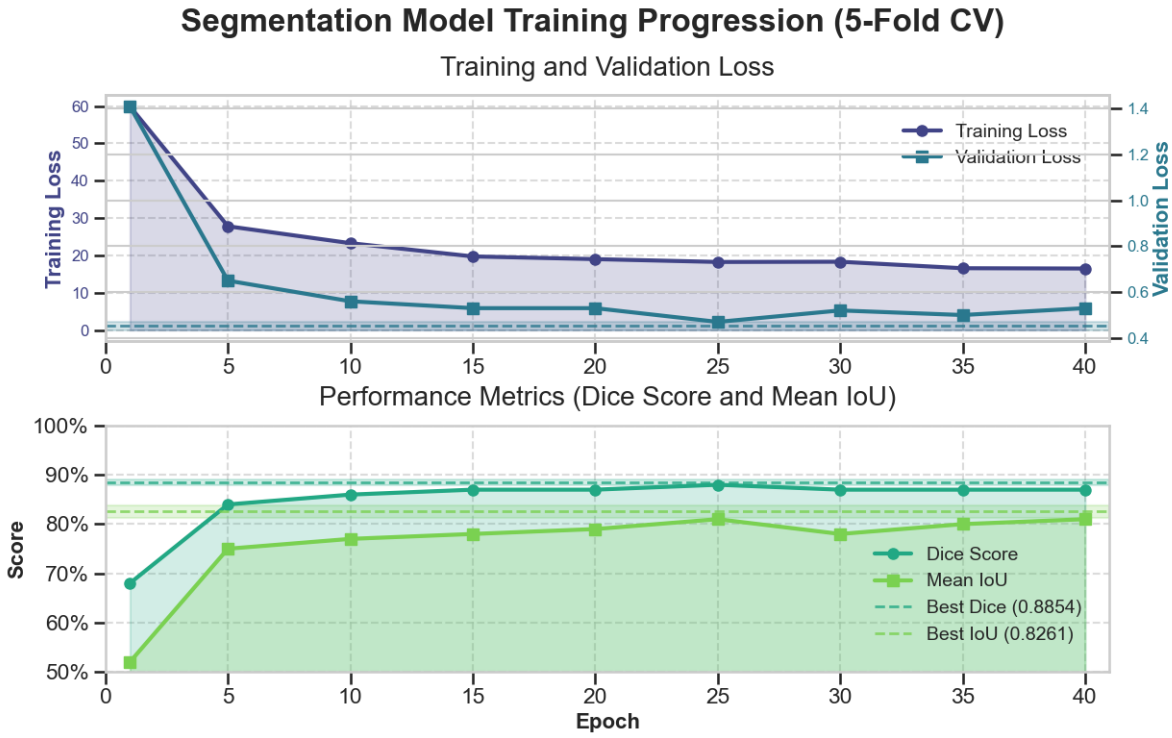


Figure 4.2. Proposed model training progression for the CAMUS dataset

4.3.1 CAMUS Results

Note: I **did not** include the background class in my averages. This substantially lowered metric scores, but provides a more accurate representation of each model's capabilities. The background class is typically extremely easy to classify due to class imbalance in datasets.

4.3.2 Models and Weights

All models were trained via the same script with 5-fold cross validation. The original Unet was constructed according to the published implementation with one level of depth removed to increase performance, as the original depth required a higher image resolution to function properly [25]. The SwinUnet[1], TransUnet[2], Unet++[3], and Unet3+[4] models utilized exact copies of published implementations. All code and weights used for these experiments are available at `https://github.com/davidsaldubehere/ViGR-Unet`.

[1] `https://github.com/HuCaoFighting/Swin-Unet/tree/main`
[2] `https://github.com/Beckschen/TransUNet`
[3] `https://github.com/4uiiurz1/pytorch-nested-unet`
[4] `https://github.com/dmMaze/UNet3Plus-pytorch/tree/master`

Table 4.1. CAMUS experiment results

	CAMUS ED Dataset		**CAMUS ES Dataset**	
Model	*DICE (%)*	*IOU (%)*	*DICE (%)*	*IOU (%)*
Proposed Model	**88.54**	**82.61**	**87.52**	**82.51**
Proposed Model w/o refinement module	88.23	81.80	87.48	81.72
Unet	75.72	63.96	75.41	65.99
SwinUnet	72.69	60.89	67.92	57.77
TransUnet	87.12	79.72	86.63	80.78
Unet++	87.66	81.39	86.22	80.12
Unet3+	87.54	80.71	86.97	81.44

4.3.3 Ablation Experiment

I conducted an ablation experiment by running a copy of my proposed model without adding the predicted residuals. All other hyperparameters were kept the same, and deep supervision was also utilized. As is evident from the results, the refinement module provides significant performance increases.

Chapter 5
Conclusion and Further Work

5.1 Performance Factors

I noticed that my model suffered from slower inference times and higher VRAM memory requirements due to the nature of graph networks. This limits the batch size, accessibility, and tasks that my model can be used for. Additionally, it was not trivial to procure a GPU satisfying this requirement. I hope to see further research into the optimization of vision graph neural networks to resolve these drawbacks.

5.2 Overfitting

During development, I was often plagued by overfitting issues. Although I am unsure of if this is a common issue among graph-based networks, I know that similarly sized models of different backbones (by parameter count) did not suffer nearly as many overfitting issues as my model did. I was able to mitigate this by including drop out layers and drop paths [17]. Further work is needed to properly diagnose and resolve this issue.

5.3 Structural Optimization

The development process also revealed that my model is not invariant to small structural changes. The encoder portion uses the same hyperparameters for graph components and their corresponding construction algorithm parameters as was proposed in the original GreedyViG model B. I attempted to optimize some of these hyperparameters via grid search algorithms, but I was unsuccessful to find reproducible candidates that improved performance. Nueral architecture search (NAS) is a promising avenue for future work to

better optimize the model's structure and hyperparameters.

5.4 Refinement Module Considerations

Although I utilized ablation experiments to showcase the importance of my novel refinement module, further work is needed to fully prove its efficacy. Direct comparisons with other existing refinement modules could show that graph-based refinement modules are superior.

5.5 The Synapse Multi-Organ Segmentation Dataset

I am currently running experiments with was the Synapse Multi-Organ Segmentation Dataset[1]. The dataset contained 3779 axial abdominal Computed Tomography (CT) images. Each CT volume was made up of 85 to 198 slices, and each slice had a resolution of 512×512 px. For my testing, the images were resized to 256×256 px. Each slice is annotated with 8 organ classes: aorta, gallbladder, spleen, left kidney, right kidney, liver, pancreas, and stomach segmentation. To simplify comparisons with existing research, I am only reporting results for these 8 classes instead of the full 13 annotated classes that the dataset provides. Training will still use the 14 classes across all models tested (considering the additional background class). The differing shapes and locations of these organs provides the perfect testbed for my model. One important note: the published dataset actually provides 50 patient samples, but the authors only provided the annotations for 30 of them because the original dataset was published in a challenge setting that required a hidden test set for accurate benchmarking purposes. Results will be reported on the project github when complete.

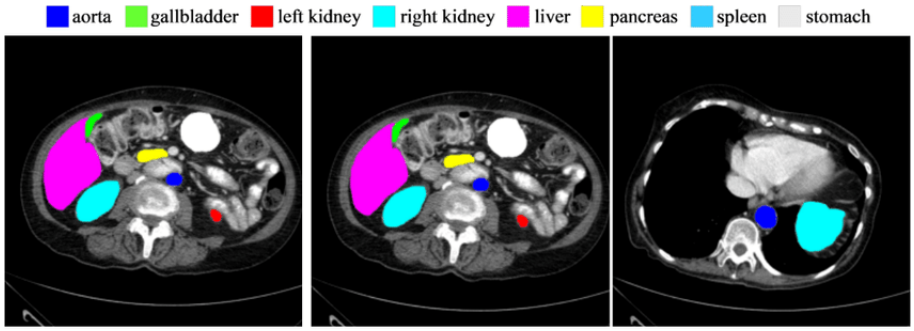


Figure 5.1. Example figure of input image and GT mask from the Synapse dataset

[1] `https://www.synapse.org/#!Synapse:syn3193805/wiki/217789`

5.6 Concluding Remarks

In conclusion, my proposed model leverages cutting-edge vision graph neural networks to define a novel Unet architecture that outperforms existing models on the CAMUS dataset. My introduction of a graph-based refinement module also provides a promising new avenue for future work in the field of medical image segmentation. I hope this work will enable new deep-learning applications to enhance the medical field.

Bibliography

[1] Md Zahangir Alom, Tarek M. Taha, Christopher Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Brian C Van Esesn, Abdul A S. Awwal, and Vijayan K. Asari. The history began from alexnet: A comprehensive survey on deep learning approaches, 2018.

[2] Michela Antonelli, Annika Reinke, Spyridon Bakas, Keyvan Farahani, Annette Kopp-Schneider, Bennett A. Landman, Geert Litjens, Bjoern Menze, Olaf Ronneberger, Ronald M. Summers, Bram van Ginneken, Michel Bilello, Patrick Bilic, Patrick F. Christ, Richard K. G. Do, Marc J. Gollub, Stephan H. Heckers, Henkjan Huisman, William R. Jarnagin, Maureen K. McHugo, Sandy Napel, Jennifer S. Golia Pernicka, Kawal Rhode, Catalina Tobon-Gomez, Eugene Vorontsov, James A. Meakin, Sebastien Ourselin, Manuel Wiesenfarth, Pablo Arbeláez, Byeonguk Bae, Sihong Chen, Laura Daza, Jianjiang Feng, Baochun He, Fabian Isensee, Yuanfeng Ji, Fucang Jia, Ildoo Kim, Klaus Maier-Hein, Dorit Merhof, Akshay Pai, Beomhee Park, Mathias Perslev, Ramin Rezaiifar, Oliver Rippel, Ignacio Sarasua, Wei Shen, Jaemin Son, Christian Wachinger, Liansheng Wang, Yan Wang, Yingda Xia, Daguang Xu, Zhanwei Xu, Yefeng Zheng, Amber L. Simpson, Lena Maier-Hein, and M. Jorge Cardoso. The medical segmentation decathlon. *Nature Communications*, 13(1):4128, 2022.

[3] William Avery, Mustafa Munir, and Radu Marculescu. Scaling graph convolutions for mobile vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 5857–5865, June 2024.

[4] Hu Cao, Yueyue Wang, Joy Chen, Dongsheng Jiang, Xiaopeng Zhang, Qi Tian, and Manning Wang. Swin-unet: Unet-like pure transformer for medical image segmentation. In Leonid Karlinsky, Tomer Michaeli, and Ko Nishino, editors, *Computer Vision – ECCV 2022 Workshops*, pages 205–218, Cham, 2023. Springer Nature Switzerland.

[5] Jieneng Chen, Yongyi Lu, Qihang Yu, Xiangde Luo, Ehsan Adeli, Yan Wang, Le Lu, Alan L. Yuille, and Yuyin Zhou. Transunet: Transformers make strong encoders for medical image segmentation, 2021.

[6] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1800–1807, 2017.

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

[8] Andre Esteva, Katherine Chou, Serena Yeung, Nikhil Naik, Ali Madani, Ali Mottaghi, Yun Liu, Eric Topol, Jeff Dean, and Richard Socher. Deep learning-enabled medical computer vision. *npj Digital Medicine*, 4(1):5, 2021.

[9] Santiago Estrada, David Kügler, Emad Bahrami, Peng Xu, Dilshad Mousa, Monique M.B. Breteler, N. Ahmad Aziz, and Martin Reuter. Fastsurfer-hypvinn: Automated sub-segmentation of the hypothalamus and adjacent structures on high-resolutional brain mri. *Imaging Neuroscience*, 1:1–32, 11 2023.

[10] Jennifer Faber, David Kügler, Emad Bahrami, Lea-Sophie Heinz, Dagmar Timmann, Thomas M. Ernst, Katerina Deike-Hofmann, Thomas Klockgether, Bart van de Warrenburg, Judith van Gaalen, Kathrin Reetz, Sandro Romanzetti, Gulin Oz, James M. Joers, Jorn Diedrichsen, Paola Giunti, Hector Garcia-Moreno, Heike Jacobi, Johann Jende, Jeroen de Vries, Michal Povazan, Peter B. Barker, Katherina Marie Steiner, Janna Krahe, and Martin Reuter. Cerebnet: A fast and reliable deep-learning pipeline for detailed cerebellum sub-segmentation. *NeuroImage*, 264:119703, 2022.

[11] Kai Han, Yunhe Wang, Jianyuan Guo, Yehui Tang, and Enhua Wu. Vision gnn: an image is worth graph of nodes. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc.

[12] Leonie Henschel, Sailesh Conjeti, Santiago Estrada, Kersten Diers, Bruce Fischl, and Martin Reuter. Fastsurfer - a fast and accurate deep learning based neuroimaging pipeline. *NeuroImage*, 219:117012, 2020.

[13] Leonie Henschel, David Kügler, and Martin Reuter. Fastsurfervinn: Building resolution-independence into deep learning segmentation methods—a solution for highres brain mri. *NeuroImage*, 251:118933, 2022.

[14] Huimin Huang, Lanfen Lin, Ruofeng Tong, Hongjie Hu, Qiaowei Zhang, Yutaro Iwamoto, Xianhua Han, Yen-Wei Chen, and Jian Wu. Unet 3+: A full-scale connected unet for medical image segmentation, 2020.

[15] Juntao Jiang, Xiyu Chen, Guanzhong Tian, and Yong Liu. Vig-unet: Vision graph neural networks for medical image segmentation. In *2023 IEEE 20th International Symposium on Biomedical Imaging (ISBI)*, pages 1–5, 2023.

[16] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[17] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals, 2017.

[18] Sarah Leclerc, Erik Smistad, João Pedrosa, Andreas Østvik, Frederic Cervenansky, Florian Espinosa, Torvald Espeland, Erik Andreas Rye Berg, Pierre-Marc Jodoin, Thomas Grenier, Carole Lartizien, Jan D'hooge, Lasse Lovstakken, and Olivier Bernard. Deep learning for segmentation using an open large-scale dataset in 2d echocardiography. *IEEE Transactions on Medical Imaging*, 38(9):2198–2210, 2019.

[19] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-Supervised Nets. In Guy Lebanon and S. V. N. Vishwanathan, editors, *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, volume 38 of *Proceedings of Machine Learning Research*, pages 562–570, San Diego, California, USA, 09–12 May 2015. PMLR.

[20] Shun Liang, Fei Tang, Xiaoli Huang, Kai Yang, Tian Zhong, Rong Hu, Shuang Liu, Xingtai Yuan, and Yong Zhang. Deep-learning-based detection and segmentation of organs at risk in nasopharyngeal carcinoma computed tomographic images for radiotherapy planning. *European Radiology*, 29(4):1961–1967, April 2019. Epub 2018 Oct 9.

[21] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, 2015.

[22] Mustafa Munir, William Avery, and Radu Marculescu. Mobilevig: Graph-based sparse attention for mobile vision applications. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2211–2219, 2023.

[23] Mustafa Munir, William Avery, Md Mostafijur Rahman, and Radu Marculescu. Greedyvig: Dynamic axial graph construction for efficient vision gnns. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6118–6127, 2024.

[24] Xuebin Qin, Zichen Zhang, Chenyang Huang, Chao Gao, Masood Dehghan, and Martin Jagersand. Basnet: Boundary-aware salient object detection. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7471–7481, 2019.

[25] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[26] Eric Wu, Kevin Wu, Roxana Daneshjou, David Ouyang, Daniel E. Ho, and James Zou. How medical ai devices are evaluated: limitations and recommendations from an analysis of fda approvals. *Nature Medicine*, 27(4):582–584, 2021.