

**MRI segmentation in cancer of the GI tract**

24-1-R-21

**Instructor name:** Miri Weiss-Cohen

**Team members:**  
 Rotem Levi

Ofek Malka

[Project GitHub link](https://github.com/rotemlv/CapstoneProject)

**Table Of Content**

[Abstract 1](#_Toc177666443)

[Introduction 1](#_Toc177666444)

[Methodology 4](#_Toc177666445)

[Overview 4](#_Toc177666446)

[The BiFTransNet model 4](#_Toc177666447)

[Suggested improvements 6](#_Toc177666448)

[Implementation 7](#_Toc177666449)

[BiFTransNet 7](#_Toc177666450)

[Integrating BiFTransNet into the TransUNet codebase 10](#_Toc177666451)

[Implementing our concepts 11](#_Toc177666452)

[Challenges and solutions 13](#_Toc177666453)

[Training 13](#_Toc177666454)

[Results 14](#_Toc177666455)

[Hyperparameters 16](#_Toc177666456)

[Discussion 17](#_Toc177666457)

[Advancements in Architecture Design 17](#_Toc177666458)

[Hyperparameter Optimization 17](#_Toc177666459)

[Challenges and Future Directions 17](#_Toc177666460)

[Conclusion 18](#_Toc177666461)

[User Documentation 20](#_Toc177666462)

[User Manual 20](#_Toc177666463)

[Maintainer Manual 23](#_Toc177666464)

[Future considerations 24](#_Toc177666465)

[References 25](#_Toc177666466)

# Abstract

This research project focuses on advancing medical imaging analysis through the development of a novel deep learning architecture for enhanced MRI segmentation. Building upon the successful BiFTransNet framework previously utilized for gastrointestinal tract segmentation, our team has engineered a modified version incorporating residual multi-scale skip connections. The proposed architecture combines the strengths of Transformer-based models with the efficiency of U-Net structures, leveraging these advancements to tackle the complex challenges in medical image segmentation. By implementing a 2.5D segmentation strategy and refining hyperparameters, including an increased batch size, we improved the model's performance compared to its predecessor. Our enhanced segmentation model demonstrates superior accuracy in identifying anatomical structures critical for radiation therapy planning in GI cancers.

**Keywords:** GI-tract,U-Net, Semantic Segmentation, Transformer, Skip-connections

# Introduction

According to **[‎1, ‎2]**, cancers of the gastro-intestinal tract affect an estimated 5 million each year worldwide, representing more than a quarter of all cancers. Worryingly, the prevalence of such cancers is increasing, causing an estimated 3 million deaths per year.

Gastrointestinal (GI) tract cancers typically begin with genetic mutations in the cells lining the digestive tract. These mutations can lead to abnormal cell growth and eventually cancer. These cancers comprise a significant portion of global cancer incidences and mortalities, pose substantial challenges in medical imaging and treatment.

Gastro-intestinal cancers manifest as gastric carcinomas arise through successive changes and not de novo from normal epithelium. For the intestinal type of gastric cancer, this includes the transformation of normal mucosa into a mucosa that resembles intestinal epithelium. Treatment strategy mainly consists of surgical resection of the primary tumor and regional lymph nodes along with chemotherapy and radiotherapy **[‎3]**.

About half of patients are eligible for radiation therapy during treatment [‎**18**]. During such treatments, radiation oncologists try to deliver therapeutic doses of radiation, using X-ray beams pointed to tumors, while avoiding the healthy stomach and intestines.

With newer technology such as integrated magnetic resonance imaging (MRI) and linear accelerator systems (MR-Linacs), oncologists are able to visualize the daily position of the tumor and intestines, which can vary day to day. In these scans, radiation oncologists must manually segment (outline) the position of the stomach and intestines in order to adjust the direction of the x-ray beams to increase the delivered radiation dose to the tumor while avoiding the stomach and intestines (see **Fig 1** below).

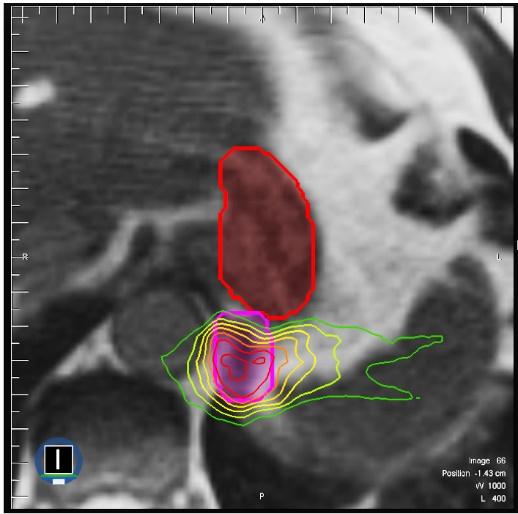


Figure 1: segmentation of the stomach and intestines on MRI scan. The tumor (pink) is close to the stomach (red). High doses of radiation are directed to the tumor while avoiding the stomach. Dose levels represented by the rainbow of outlines, with higher doses represented by red and lower doses represented by green [‎**2**].

However, manual segmentation of the tumor and the GI organs is a time-consuming and labor-intensive process that can prolong treatments from 15 minutes a day to an hour a day, which can be difficult for patients to tolerate. Automation of this process could shorten treatment duration, ease the burden on radiologists and could provide faster treatment and quicker recovery in radiation patients.

Deep learning methods in general and segmentation U-Nets can assist in reducing manual work and allowing more patients to receive appropriate treatment by automating the segmentation process. Indeed, deep neural network-based methods have been employed in recent years for the automated diagnosis of medical illnesses. It has outperformed traditional algorithms in terms of accuracy since the characteristics are learned from data using a general-purpose learning technique rather than being built by human engineers.

Our goal is to develop an enhanced GI tract segmentation method for MRI images that rivals current SOTA in accuracy, by incorporating Transformer based architectures in our model.

The field of medical segmentation networks (and segmentation in general), even with the recent advancements (incorporating transformers and similar attention mechanisms in different forms), has yet to obtain sufficiently accurate values (the problem of semantic segmentation, unlike classification, is not yet fully solved), but strides towards that goal are constantly being made.

Since segmentation of the GI tract has seen less attention (compared to brain or liver segmentation networks research), our aim is to compare and analyze the most recent architectures for GI segmentation and improve their results, thereby allowing future research an easier path towards solving the problem of automated medical segmentation.

Segmentation in the context of medical imaging, such as MRI scans, refers to the process of dividing an image into multiple segments or regions, each representing a different part of the body or anatomical structure. This technique is crucial for identifying and isolating specific areas of interest, such as tumors or organs, from the rest of the image. In the field of gastrointestinal (GI) tract segmentation, this process is particularly important for diagnosing and treating various conditions.

The significance of segmentation in GI tract imaging lies in its potential to revolutionize the diagnosis and treatment of gastrointestinal disorders. Traditional manual segmentation methods are labor-intensive and time-consuming, often extending treatment durations and potentially affecting patient recovery times. By automating this process, we can significantly reduce the burden on radiologists and improve patient outcomes.

# Methodology

## Overview

### The BiFTransNet model

As decided during the research process in the first stage of this project, we provide a U-Net model based on the BiFTransNet architecture, utilizing the transformer’s attention mechanism. This demanded a thorough analysis and understanding of the aforementioned model. Given the lack of a source code for replication purpose, we resolved to carefully examine the BiFTransNet paper, in order to build this base architecture properly and maintain a solid baseline for comparison.

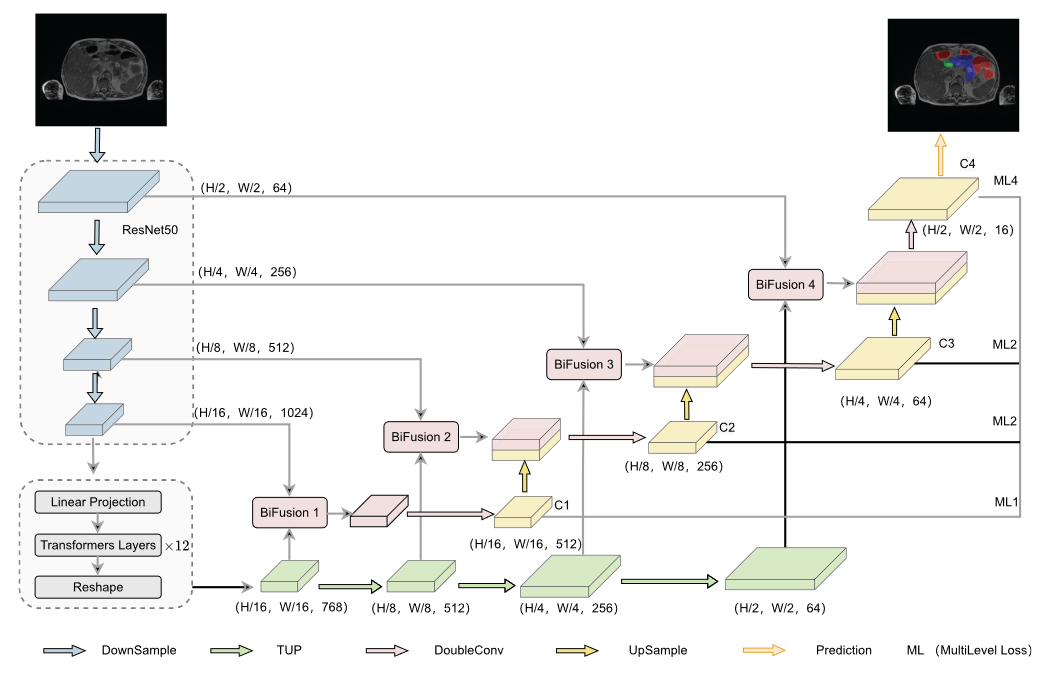


Figure 2 A block diagram of the BiFTransNet segmentation model.

The authors used the TransUNet model from the original paper [insert citation], with publicly available source code. TransUNet’s different components were used as-is, with the following additions:

* Transformer features up-sampler (TUP): This component is used to feed the output features of the transformer bridge in TransUNet to the different decoder blocks, as additional skip-connection data.
* Bi-Fusion block: This component receives the outputs of the aforementioned TUP block, along the encoder outputs (the traditional U-Net skip connections) and combines the two through three main operations – spatial attention, channel attention and multimodal fusion.
* Multi-leveled loss function: This component influences both the model and the training function. Multi-level loss is a strategy which aims to help guide the learning process by taking the loss of all four of the decoder outputs, these are up-sampled to fit the required output spatial dimensions and further convolved (1x1 convolution) to fit the channels dimension, and are consequently fed into the loss function.

The transformer up-sampler and Bi-Fusion block work together with the goal of enhancing the spatial information found in the traditional skip-connections with additional attention guided context that the transformer layers obtain. This information is sequentially up-sampled and fed into the decoder blocks alongside the skip-connections, where both are fed into a Bi-Fusion block, which uses a combination of attention mechanisms (Channel-Attention and Spatial-Attention) to best combine the two.

The Bi-Fusion block incorporates the attention mechanisms described in the papers[‎27, ‎28]. The papers describe two methods as follows:

*Squeeze-and-Excitation Blocks (Channel Attention)*

Squeeze-and-Excitation (SE) blocks enhance deep neural networks by modeling channel-wise dependencies and recalibrating feature responses. The process involves a "squeeze" operation using global average pooling to embed spatial information into channel descriptors, followed by an "excitation" operation that applies a gating mechanism to selectively emphasize or suppress channels. This adaptive recalibration enhances relevant features and introduces input-conditioned dynamics, effectively acting as a self-attention mechanism on channels.

*Spatial Attention*

The spatial attention component emphasizes important image regions by applying average-pooling and max-pooling along the channel axis. These operations capture complementary information and highlight informative areas. The resulting feature descriptors are concatenated and processed through a 7x7 convolution layer to produce a 2D spatial attention map. After applying a sigmoid activation, the resulting weights are multiplied element-wise with the input feature map to selectively emphasize or suppress different spatial regions.

The authors specify the use of convolutional layers instead of linear ones – this is done in order to improve the model’s learning capacity, via the ReLU activation function (introduction of additional non-linearity in the model).

### Suggested improvements

On top of this architecture, we explored the following improvements:

*Multi-scale context* – via a multi scale skip connections strategy. The idea of using additional skip connections in U-Net based models has a long successful track record. We saw such examples during our research stage, used with various degrees of success [‎24, ‎20]. We chose to implement the residual skip-connections strategy, as seen in TransAttUNet (and in the following image, figure 1), aiming to capture the improvement shown in said paper.

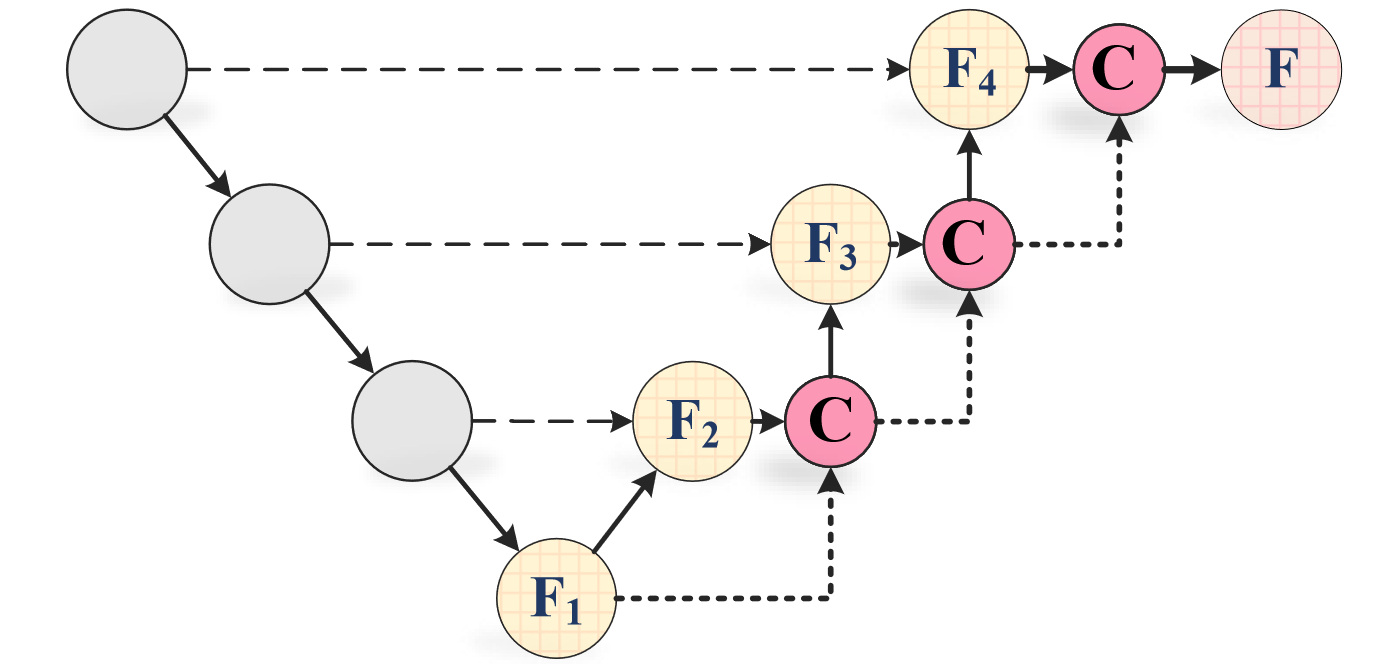


Figure 3 Residual multi-scale skip connections, from TransAttUNet

*Transformer decoder* – inspired by 3D-TransUNet’s coarse-to-fine attention refinement component. The included ablation study indeed showed a marked improvement when using this component in the aforementioned model. It is worth noting, however, that 3D-TransUNet was trained with a batch size of 2, due to the computational cost of the transformer decoder. Further research led us to forego the use of this component, as the implementation proved to be too memory intensive when trained using our desired batch sizes.

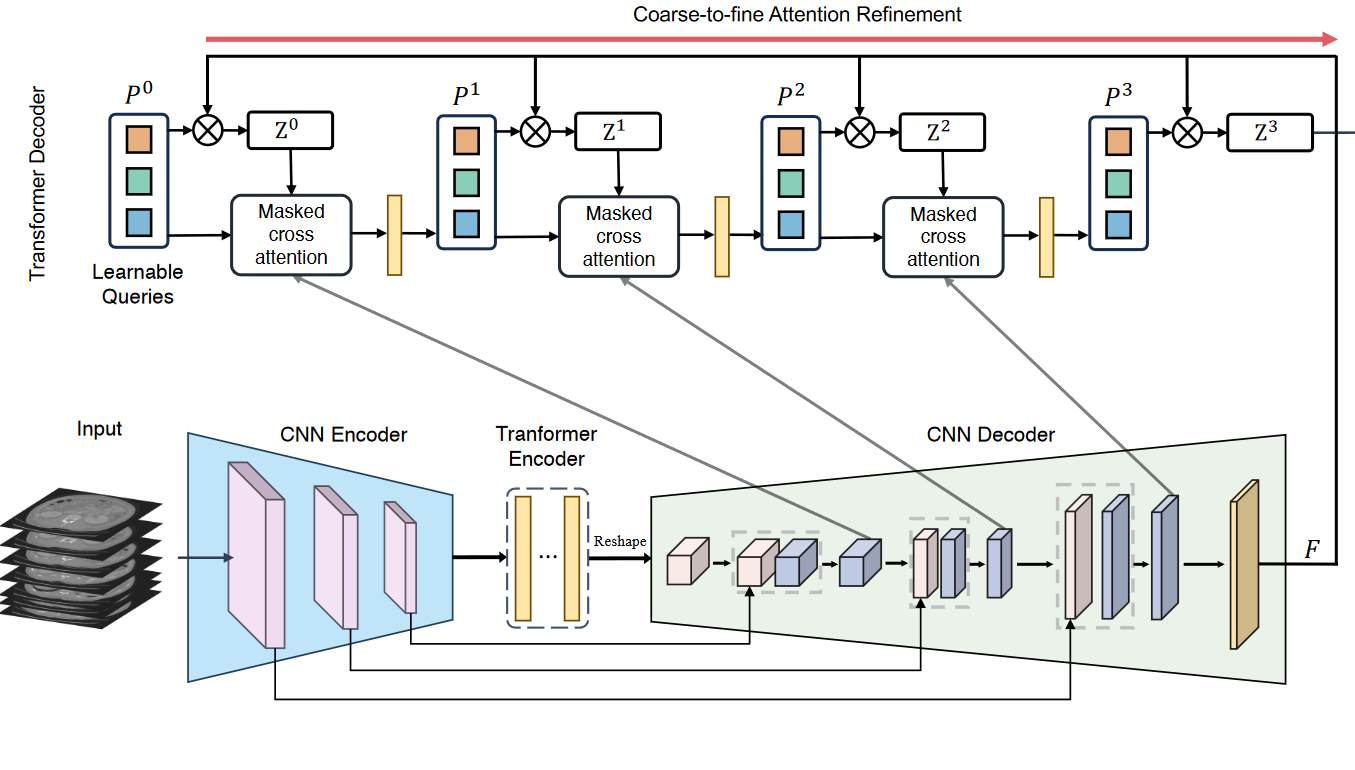


Figure 4 The 3D-TransUNet architecture, showing the transformer decoder component as Coarse-to-fine Attention Refinement

*Hyperparameters adjustments* – i.e. batch size, learning rate and epoch number. We found that current SOTA models used GPUs with limited memory capacity (up to 16GB of VRAM), hence, we mainly focused on increasing the batch size, using the A100 GPU available via Google Colab Pro+, thus decreasing the stochastic nature of the training process and providing a better sample size during the learning process. During the following section we will discuss the rest of our hyper-parameter choices. This decision allowed us to increase the batch size all the way to 64, providing a measurable improvement compared to current methods.

## Implementation

In this chapter we will discuss our model’s components in detail. This includes not only a description of the different components, but also the background and rationale behind our different implementation decisions, the limitations we faced and how we overcame those.

### BiFTransNet

We will begin by conducting a step by step review of our implementation of the original BiFTransNet model – which required some modifications in order to fit our requirements. This will be followed by a review of our proposed modifications and the insights we obtained during the coding and training process, ending with a comprehensive review of our finished model.

During our work building the BiFTransNet model we faced many different obstacles, mainly due to the lack of a shared implementation (source code) in the paper. BiFTransNet is built atop the TransUNet model, with a few modifications, which we designed according to the paper’s descriptions.

There are three modifications BiFTransNet introduces into the TransUNet architecture. The TransUNet model consists of a U-Net topology with a transformer bridge (between the encoder and the decoder). This modification allows the model to learn additional contextual information obtained by the transformer, and further improve its evaluation metrics.

*Transformer Up-sampler (TUP):* As previously stated, this component is used to up-sample the features obtained from the transformer parallel to the main U-Net decoder path. This is done in preparation for inserting said features into the Bi-Fusion block, with the goal of providing additional context to the traditional skip-connections in the U-Net architecture.

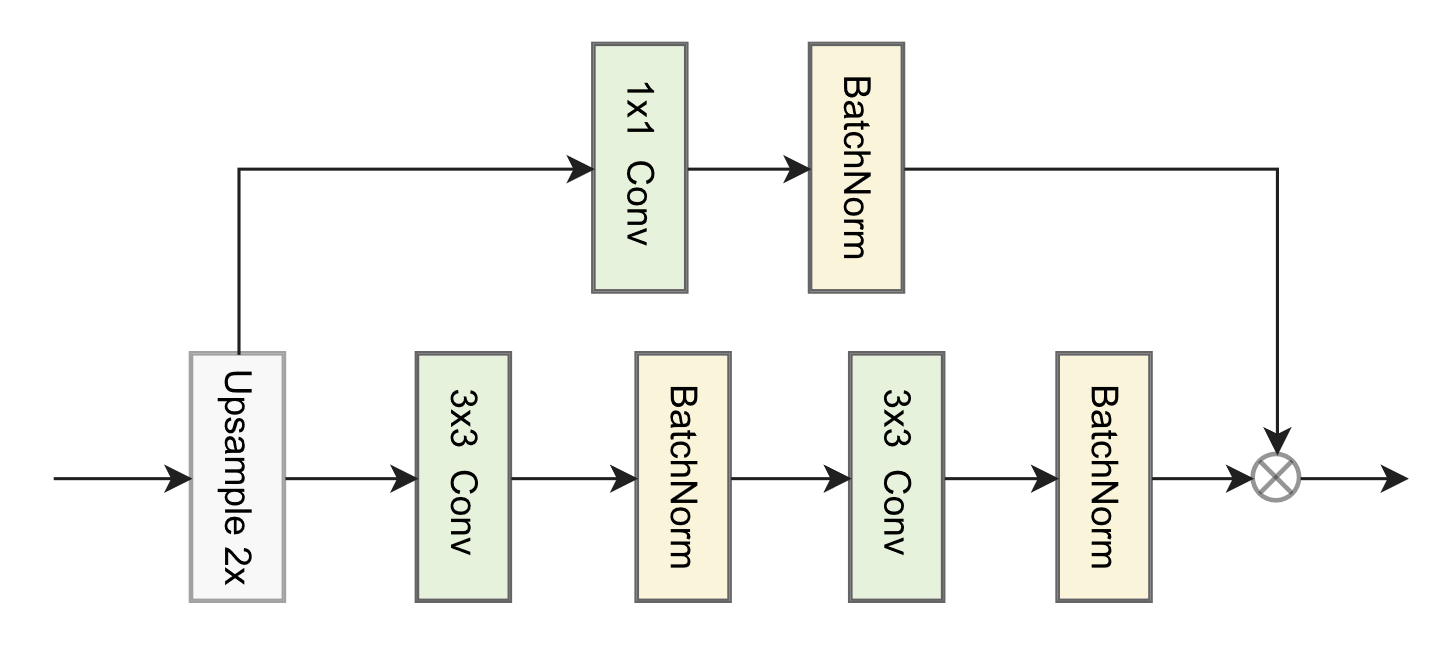


Figure 5 The TUP module

As seen in the figure, the TUP consists of two paths – a main path and a residual one. An activation function is not specified here, but we chose ReLU as a safe option. The X symbol is also left unexplained in the original paper, but, as seen in the Bi-Fusion figure below, it is specified as multiplication, which is the operation we used between the two paths in the TUP component. Also, the exact up-sampling algorithm is also unspecified, we chose to use a bilinear up-sampling – a simple yet robust up-sampling approach.

*Bi-Fusion block:* The Bi-Fusion block is used to combine skip connections with the up-sampled transformer features, this is done through three different paths incorporating attention mechanisms and a residual convolutional path, followed by a 1x1 convolution in order to fit the channels dimensions to the desired output shape.

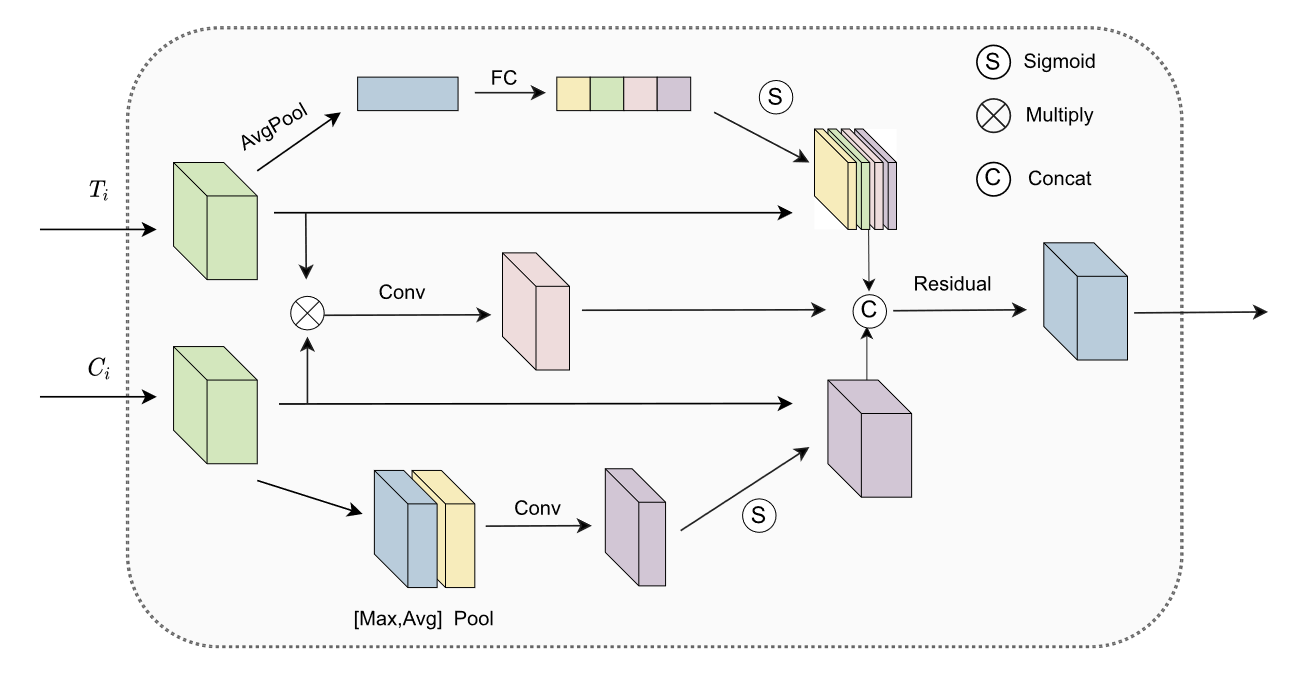


Figure 6 The Bi-Fusion block

The Bi-Fusion block consists of a few attention-guided and convolutional operations, and three different paths. The top path describes the Channel Attention operation (described in the “Squeeze & Excitation” paper), using an MLP to enhance the transformer features’ spatial information. The middle path’s is named “multi-modal fusion” – its goal is to merge both feature sets (skip and transformer) in a way which preserves the information contained in both, which is done through a multiplication operation followed by a convolutional layer. The bottom path describes the Spatial Attention operation, inspired by the “CBAM” paper. Here, the authors chose to forego the linear layers in favor of a convolutional approach, which is used to enhance the training process through the non-linearly introduced therein. Implementing this block proved quite difficult, as many operations shown in the above diagram are not explicitly described. We used a convolution + batch normalization + ReLU wherever a convolutional layer is described, and used instance normalization instead of the batch normalization described in the Spatial Attention part of the paper, due to the undefined behavior of the latter in the case described in the paper – where the channels dimension is 1.

*Multi-level loss function:* This part consists not only of an architectural change, but also required a modification of the training process. Seen in the BiFTransNet figure, the multi-level loss (marked ML) is essentially a collection of the four different decoder outputs of the network’s decoder path. Instead of taking the final decoder’s segmentation mask, BiFTransNet takes a list of them. The loss is then iteratively applied to each tensor in this list, and multiplied by a scalar value (a weight for this loss) – the scalars chosen in the paper are {0.1, 0.1, 0.1, 0.7}, for each decoder layer from the deepest to the shallowest.

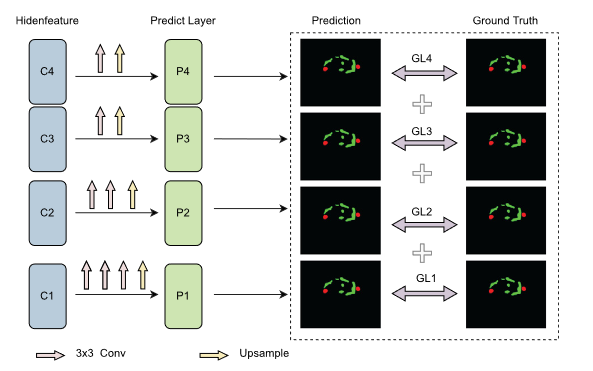


Figure 7 The multi-level loss calculation. represents the outputs of different levels of decoder

According to the authors, this strategy helps to better guide the learning process – by applying the loss to the deeper layers, the vanishing gradient problem becomes far less significant. During our tests, however, we discovered that this strategy did not provide additional accuracy.

Incorporating the multi-level loss is relatively straightforward – we added convolutional layers and up-sampling to the tail of each decoder block, and collected these outputs into a list. This list is then iterated upon during the training process, and each loss is weighted and summed.

During testing, we found that the multi-level loss strategy did not provide any measurable improvement, we also found that the memory footprint of training the model using this approach grew drastically, which led us to eventually remove it.

### Integrating BiFTransNet into the TransUNet codebase

The *TUP module* was inserted sequentially after the features obtained from the transformer have been re-shaped into a tensor (i.e. transformer features), these are fed into TUP modules and up-sampled to fit the size of the skip connections entering each block in the decoder.

The *Bi-Fusion* blocks are used right after their respective TUP module, where both sets of features – the skip connections and the transformer up-sampled features – are fed into it. Finally, the output of the Bi-Fusion block is fed into the decoder’s two convolutional layers.

The *multi-level loss* function is essentially an up-sampler component at the tail of each decoder block, where the outputs are reshaped to fit the final segmentation masks, and returned as a list of logits for the loss function to use. Additionally, the loss function is then modified to fit a list of logits instead of a single tensor – here, each layer’s loss is scaled according to the set of parameters mentioned in the original paper and summed, bringing us the total loss function.

Incorporating these three components into the baseline TransUNet source code proved to be quite challenging – the source code is written in an iterative manner (decoder blocks are created in loops), where a correct integration requires intricate knowledge of the dimensional changes that occur in each step of the model. Additionally, the lack of lower level details in the paper proved quite challenging to overcome, and we had to overcome difficulty in assessing the different components described therein.

### Implementing our concepts

On top of the BiFTransNet model, we incorporated the following changes:

**Multi-scale skip connections**

In order to provide our decoder with additional spatial context, we implemented a residual multi-scale connection between the different decoder blocks. This is done inside the decoder block module – here, we grab the prior decoder’s input and concatenate it to the current layer’s output. This operation proceeds a 1x1 convolutional layer used to fit the next block’s channel size requirements.

Implementing the multi-scale approach faced us with the issue of overfitting: unlike a more traditional U-Net topology, the BiFTransNet architecture already consists of three inputs that are all fed into the decoder block – the main path of the network, the skip connection and the transformer up-scaled features. Adding an additional input to this trio (the residual skip connection from the shallower layers) introduced overfitting during training, indicated by low training loss and a consistently higher validation loss. In order to solve this issue we employed two strategies – regularization (via dropout and weight decay) and feature reduction. We chose to perform a 1x1 convolution on the residual connection in order to decrease the channel count of the residual path, additionally limiting overfitting by reducing the extensive feature reuse in the model. This issue was further ameliorated by an increase of the regularization hyperparameter (weight decay), which helped us reach the best performance in this task, alongside the use of minor dropout in the transformer and the encoder part of our architecture.

**2.5D Training pipeline**

The 2.5D pipeline is a sophisticated training methodology that bridges the gap between traditional 2D and 3D image processing techniques. It combines the efficiency of 2D convolutional neural networks (CNNs) with the spatial awareness of 3D models, creating a hybrid approach that leverages the strengths of both paradigms. Due to the increased spatial content in the input, this approach has proven to be highly effective in GI tract segmentation tasks.

2.5D provides advantages in performance, efficiency (compared to 3D models), and we have found it more accurate when compared to the standard (2D) approach.

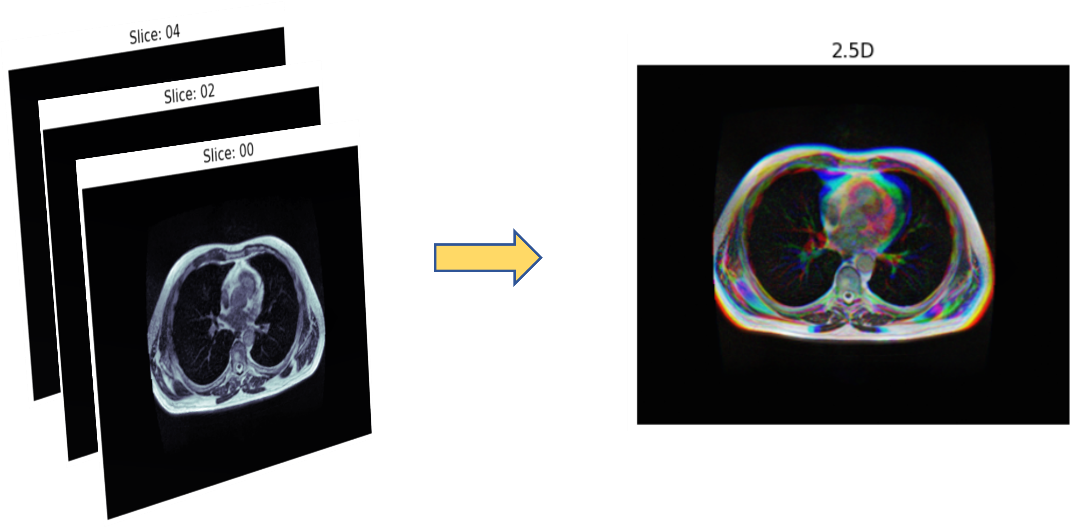


Figure 8 The 2.5D process - slides are concatenated over the RGB channels.

**Extensive data augmentation methods**

Our comprehensive data augmentation strategy for the UW Madison GI tract dataset employs a sophisticated array of techniques to address the challenges posed by limited sample sizes in medical imaging research. Leveraging eight primary methods, we implemented a robust augmentation pipeline that significantly expands the dataset's diversity and generalization capabilities. These techniques include **flip and rotation** operations to simulate patient repositioning and variability in scanning protocols; **grid distortion** and **elastic transforms** to mimic organ deformation due to breathing or movement during scans; **coarse dropout** to simulate partial views or occlusions; **brightness and contrast adjustments** to reflect differences in imaging protocols and scanner calibrations; and **low resolution simulation** and **Gamma transformations** to model variations in image acquisition parameters and post-processing techniques. By implementing this sophisticated augmentation pipeline, we effectively expand our limited dataset, reduce overfitting risks, and enhance our model's capacity to handle real-world variability in GI tract imaging, ultimately leading to improved segmentation accuracy and robustness in clinical applications.

### Challenges and solutions

* Framework issues: Google Drive and Colab both caused us issues during training - by disconnecting the dataset from the model and by crashing during the final stage of training (or when downloading the trained model). We resolved the first issue by circumventing it – we import the data directly from Kaggle via API calls. This solution proved effective, but caused a small slowdown during the beginning of our training process. The other issues remained with us for the duration of this project, occasionally costing us time and additional cost for compute units, in cases where our model could not finish training due to a crash.
* Tversky loss: We faced a peculiar issue with NaN (Not a Number) values in our loss function. After investigating this issue via the torch.autograd tool, we isolated it to the built-in implementation of the Tversky loss function, alongside our dataset’s shape (some images are entirely black – filled with zeros). The issue we faced had to do with raising the mean loss (for a given batch) to the power of 1, which is defined as the default behavior within the Tversky loss implementation. Since the power is defined as a float, the derivative becomes , which caused issues when calculating the derivative of the mean operation itself due to floating point instability (the power can be slightly negative). We confirmed this as solved by increasing said power to 1.01.
* Lack of documentation: as previously stated, the lack of a complete explanation for the operations described in the BiFTransNet paper led us to research the papers used as sources, and choose the best techniques to apply when developing the base model. This problem required a detailed approach which relied on secondary sources, as we described during the previous chapter.
* Package versions: In using a pre-existing model as a baseline, and the Kaggle API as part of our training pipeline, we faced many version conflicts in our implementation process – TransUNet was developed back in 2021, and many portions of the code required modifications in order to work inside the current Google Colab environment. Also, the data-loader built for TransUNet was overhauled to fit the current version.

## Training

Besides the architectural decisions, we built a training-validation-testing pipeline. As discussed earlier, our implementation uses a 2.5D pipeline, which proved easy to implement (due to the existence of a ready-to-use 2.5D dataset in the Kaggle database) and provided better results. As decided during the first stage of our project, we have trained our model with different hyperparameter settings – for batch size, learning rate and weight-decay (regularization, in order to avoid overfitting). In the following chapter we will discuss the results we obtained and describe the best hyperparameter choices.

# Results

Our study on MRI segmentation in cancer of the GI tract yielded promising results. The enhanced segmentation model, incorporating the strengths of Transformer-based models, U-Net structures and multi-scale skip connections, demonstrated superior accuracy and efficiency. Here are the specific outcomes based on different models compared in our study:

|  |  |  |
| --- | --- | --- |
|  | Average | |
|  | Dice (%) | Iou (%) |
| Unet | 88.26 | 85.25 |
| FPN | 88.49 | 85.41 |
| Deeplabv3 | 88.57 | 85.37 |
| Deeplabv3+ | 88.61 | 85.48 |
| TransUnet | 88.30 | 85.59 |
| BifTransUnet | 89.51 | 86.54 |
| Ours | **90.71** | **87.85** |

These results underscore our model's improved performance over established architectures, notably enhancing segmentation precision critical for radiation therapy planning in GI cancers. The improvements are attributed to our strategic enhancements in the architecture and training phases, particularly the integration of residual multi-scale skip connections and a refined 2.5D segmentation strategy.

We also engaged in extensive data augmentation to address the challenges of limited sample sizes in medical imaging. This approach significantly enhanced our model's ability to generalize, reducing overfitting risks and improving robustness against variations in real-world clinical settings.

The comparison clearly illustrates the advancements our model offers, making it a viable tool for enhancing accuracy and efficiency in medical imaging segmentation, particularly for complex organ structures within the GI tract.

In the following sections, we present a series of graphs that illustrate the performance of our segmentation model throughout the training and validation phases. These visuals include the trends in training and validation loss, demonstrating the model's learning curve and stability over time. Additionally, graphs depicting the Dice coefficient and IoU scores are provided to assess the accuracy and precision of our model in segmenting GI tract images. These metrics are crucial for evaluating the effectiveness of our model, with higher values indicating better overlap between the predicted segmentations and the ground truth annotations.

Figure 9: Dice and IoU scores vs. epochs

Figure 10: Train and valid loss per epoch

## Hyperparameters

Our extensive experimentation led to the identification of superior hyperparameters for our Unet-based GI tract segmentation model. We set image dimensions to 224x224 pixels, balancing spatial resolution and computational efficiency. Batch size of 64 maximized GPU utilization while avoiding memory constraints. The learning rate started at 2e-3, decreasing via Cosine Annealing LR scheduling over 140 epochs to a minimum of 1e-6. L2 regularization with a coefficient of 2e-5 mitigated overfitting without hindering learning capacity. We employed the Adam optimizer with the aforementioned weight decay of 2e-5, with the goal of enhancing training stability and convergence speed. Our custom loss function combined BCE and Tversky losses, inspire by the choice of the BiFTransNet authors, is used to address class imbalance and promote a more precise boundary delineation.

This configuration yielded superior performance across validation metrics, demonstrating effectiveness in GI tract segmentation. The combination of these hyperparameters allowed our model to capture nuanced features in the images while maintaining computational efficiency. Further optimization may be achievable through fine-tuning based on specific dataset characteristics and available computational resources.

|  |  |
| --- | --- |
| **Ground Truth** | **Prediction** |
|  |  |
|  |  |
|  |  |
|  |  |

# Discussion

Our research project aimed at advancing medical imaging analysis through the development of a novel deep learning architecture for enhanced MRI segmentation of the gastrointestinal (GI) tract. By building upon and modifying the successful BiFTransNet framework, we have made significant strides in improving segmentation accuracy and efficiency.

## Advancements in Architecture Design

Our proposed architecture combines the strengths of Transformer-based models with the efficiency of U-Net structures, thereby tackling complex challenges in medical image segmentation [‎4]. The incorporation of residual multi-scale skip connections has proven particularly effective in capturing contextual information across different scales, leading to improved segmentation accuracy [‎20]. This approach aligns with recent trends in medical imaging segmentation, where hybrid architectures combining CNNs and Transformers have shown promising results [‎21].

However, our experience highlights an important consideration for future research. While Transformer decoders, such as those employed in 3D-TransUNet, have shown potential in improving segmentation accuracy [‎5], their implementation poses significant challenges in terms of memory efficiency. Our attempt to incorporate such a component was hindered by computational constraints, even when utilizing high-end GPUs like the A100 available via Google Colab Pro+. This limitation underscores the need for future work to focus on developing more memory-efficient implementations of Transformer-based components within U-Net architectures [22].

## Hyperparameter Optimization

Our research demonstrates the importance of hyperparameter tuning in deep learning models for medical image segmentation. By leveraging the computational resources available through Google Colab Pro+, we were able to significantly increase our batch size to 64, leading to improved model performance. This finding aligns with recent studies showing that larger batch sizes can lead to better generalization and faster convergence in deep neural networks [‎23].

## Challenges and Future Directions

While our modified architecture incorporating multi-scale skip connections has shown promise, we encountered overfitting issues that warrant further investigation. Systematic approaches to address overfitting, such as regularization techniques or data augmentation strategies, should be explored in future research. Additionally, given the increasing prevalence of GI tract cancers worldwide [‎1], there is a pressing need for continued advancements in automated segmentation methods to support radiation therapy planning and treatment outcomes.

Our study contributes to the growing body of research focused on improving medical imaging analysis through advanced deep learning architectures. As the field continues to evolve, we anticipate that hybrid models combining the strengths of CNNs, Transformers, and other architectural innovations will play a crucial role in advancing the accuracy and efficiency of medical image segmentation.

# Conclusion

Our research project has made significant strides in advancing medical imaging analysis through the development of a novel deep learning architecture for enhanced MRI segmentation of the gastrointestinal (GI) tract. By building upon and modifying the successful BiFTransNet framework, we've demonstrated the potential of hybrid architectures combining the strengths of Transformer-based models with the efficiency of U-Net structures. This approach has proven particularly effective in tackling complex challenges in medical image segmentation, especially in the context of GI tract cancers.

The incorporation of residual multi-scale skip connections has been a key factor in our improved segmentation accuracy. This design choice aligns with recent trends in medical imaging segmentation, where researchers have found that capturing contextual information across different scales leads to more precise delineation of anatomical structures [24]. Our results underscore the importance of considering multi-scale features in segmentation tasks, particularly for complex organs like those in the GI tract.

One of the most significant contributions of our work is the exploration of the interplay between architectural design and computational resources. While our attempts to incorporate Transformer decoders, inspired by models like 3D-TransUNet, showed promise in improving segmentation accuracy, they were ultimately hindered by memory constraints [‎5]. This experience highlights a crucial challenge in the field: balancing the desire for more powerful models with the practical limitations of available computing resources. Future research should prioritize developing more memory-efficient implementations of Transformer-based components within U-Net architectures.

Our research also emphasizes the critical role of hyperparameter optimization in deep learning models for medical image segmentation. By leveraging the computational resources available through Google Colab Pro+, we were able to significantly increase our batch size to 64, leading to improved model performance. This finding aligns with recent studies demonstrating that larger batch sizes can lead to better generalization and faster convergence in deep neural networks [‎25]. However, our experience also shows that increased computational power alone is not sufficient; careful tuning of hyperparameters remains essential for optimal performance.

Alongside the advances we've made, we also encountered challenges that warrant further investigation. The issue of overfitting, particularly pronounced in models with multiple input paths like BiFTransNet, proved to be a significant challenge. Our attempts to address this through regularization techniques and feature reduction strategies had mixed results, which led us to conclude that a more thorough regularization solution would provide additional improvements. Future research should focus on developing robust methods to prevent overfitting in complex segmentation models, possibly exploring novel regularization techniques or advanced data augmentation strategies.

The clinical implications of our work cannot be overstated. With GI tract cancers affecting millions worldwide and accounting for a substantial portion of global cancer-related deaths [‎1], the need for accurate and efficient segmentation tools in radiation therapy planning is pressing. Automated segmentation methods like ours have the potential to significantly reduce treatment durations, ease the burden on radiologists, and ultimately contribute to improved patient outcomes [‎26]. Furthermore, as integrated MRI and linear accelerator systems become more prevalent, the demand for reliable segmentation tools will only grow.

Looking forward, we anticipate that hybrid models combining the strengths of CNNs, Transformers, and other architectural innovations will play a crucial role in advancing the accuracy and efficiency of medical image segmentation. The integration of attention mechanisms, particularly in the form of Transformer-based components, shows great promise for capturing complex contextual relationships in medical images [‎4]. However, this integration must be balanced against practical considerations such as computational efficiency and interpretability.

In conclusion, our research demonstrates the potential of advanced deep learning architectures for improving GI tract segmentation in MRI images. While significant progress has been made, our findings highlight areas for future research, particularly in developing memory-efficient Transformer-based components and addressing overfitting issues in complex segmentation models. As medical imaging continues to play an increasingly critical role in cancer diagnosis and treatment, ongoing advancements in automated segmentation techniques will be essential for improving patient outcomes and streamlining clinical workflows. Our work contributes to this growing body of research, providing valuable insights for future developments in this rapidly evolving field.

# User Documentation

Our application allows the end user to segment images from the UW-Madison GI tract dataset using our model. Due to our focused approach in building and testing the model’s architecture, and refining the training process, we decided to provide a basic tool which only allows the segmentation of existing images. The tools to convert MRI slices into a numpy array also exist in our code, allowing a maintainer to adapt the current GUI application.

## User Manual

Our GUI can be described as follows: on startup – the model stationed in the my\_model.py file is loaded, with the weights stored in a dedicated binary (pre-trained weights) clicking the “start segmentation” button starts by segmenting the first image in the dataset by running it through the network and provides the user the predicted output. Due to the nature of this supervised dataset, the output includes the ground truth along the prediction, with an additional “difference” slice – providing an insight into how far our predictions are from the ideal output. After this, the used can click “segment next image” in order to advance along the dataset and view further segmentation results. The GUI allows the user to select how to view our segmented image: the default option shows the aforementioned four image (image, ground truth, prediction and the difference), but the user can choose to change view into an output of two copies of the image, with each showing the masks (prediction and ground truth) on top of the image, for an effective comparison between the two.

Additionally, our GUI system allows the user to upload or insert a different set of weights, by clicking the “load weights” button. This will prompt the user to select a file – the system will then attempt to load the weights into the model residing in the “my\_model.py” python file. A message will appear to indicate success or failure (if the file does not fit the current model, for instance), and subsequent the segmentation task will restart the order of the dataset – allowing the user to compare the results for the same slice across different sets of weights.

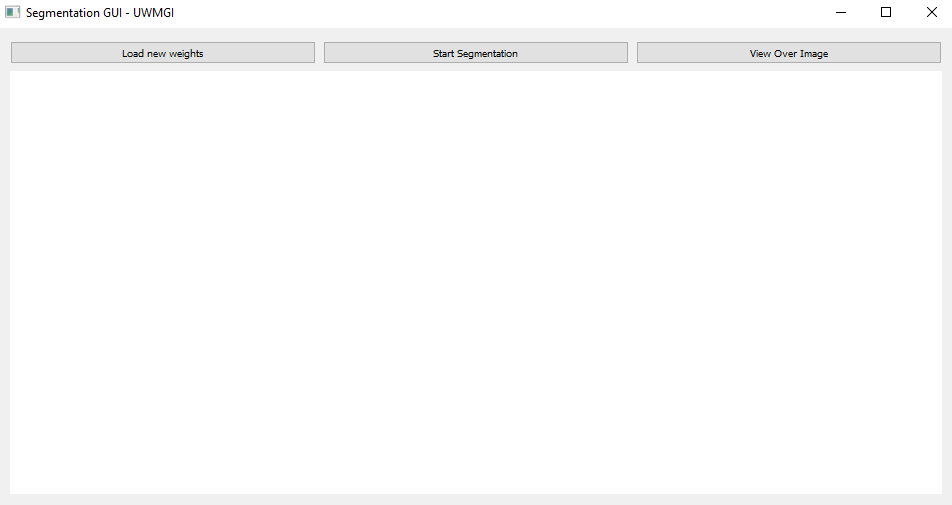
**GUI – a walkthrough example:**  


Figure GUI application on launch

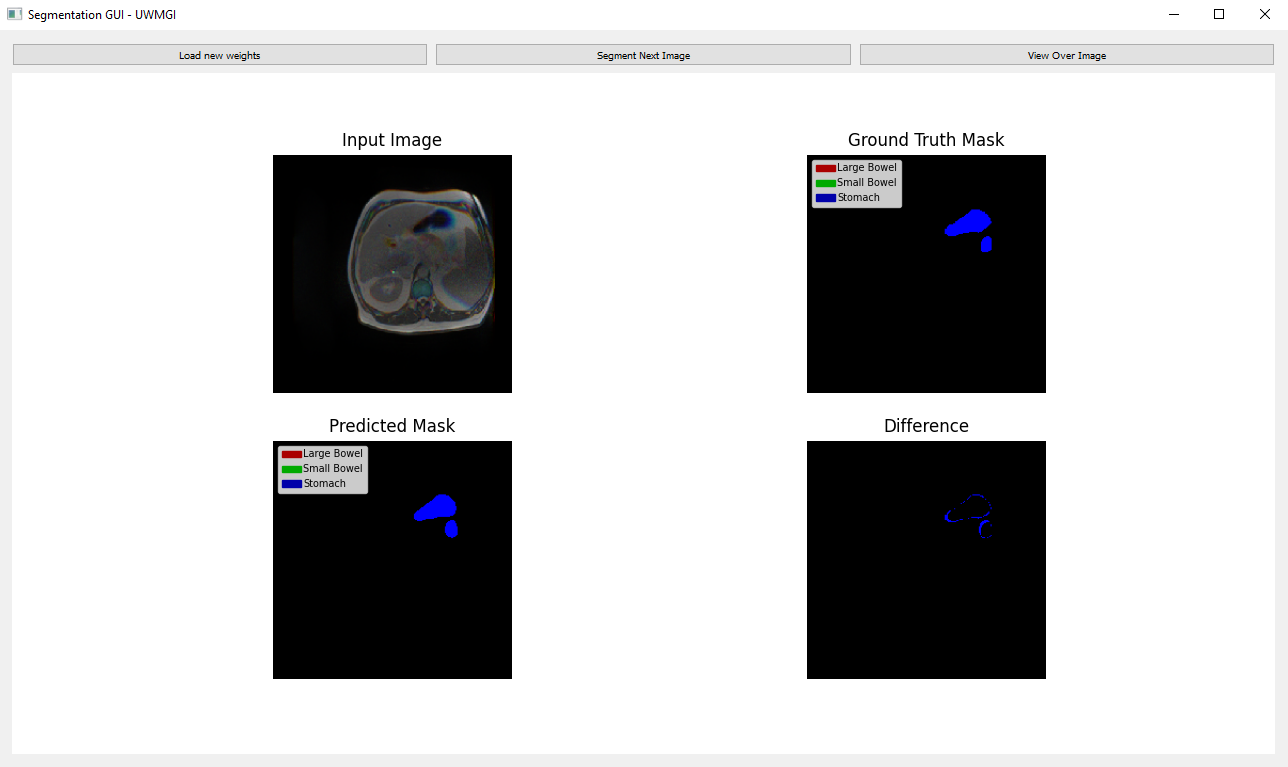


Figure GUI after clicking "Start Segmentation"

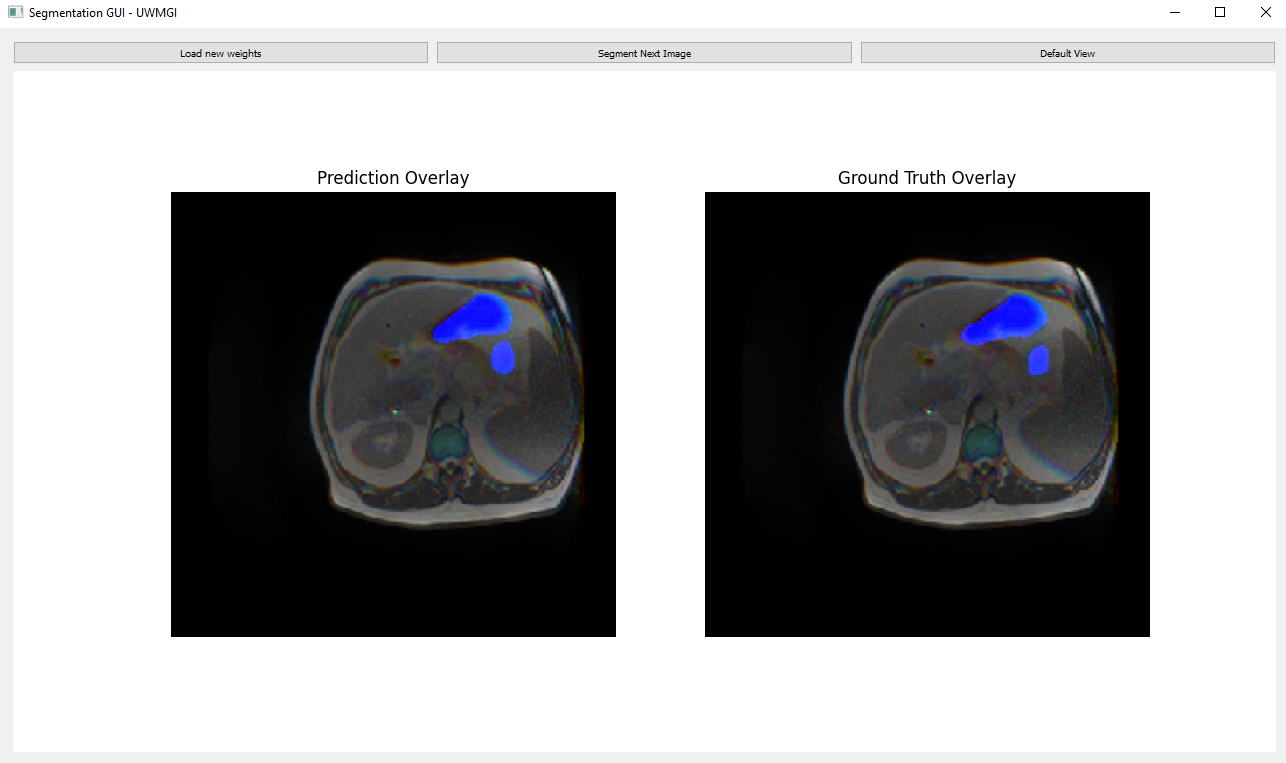


Figure Clicking "View Over Image" allows user to view the masks on top of the MRI slices

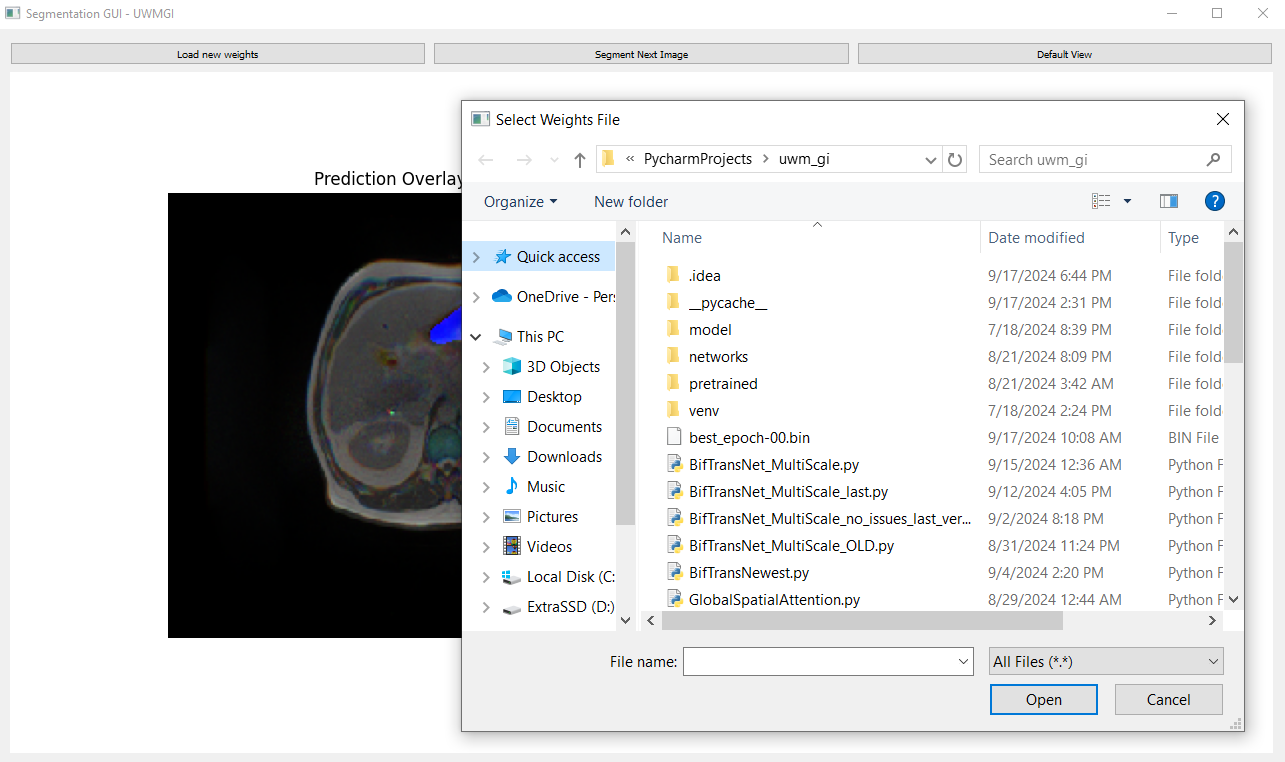


Figure Clicking "Load New Weights" allows the user to upload a different set of trained weights for the used model

## Maintainer Manual

The aforementioned codebase is very robust, allowing simple modifications for the model, the weights used and the dataset:

* **Replacing the trained weights:** the weights are stored as a binary file “weights.bin”, allowing the maintainer the option to upload a newly trained binary and replace the model without any code modifications. Unlike the above mentioned “load weights” GUI button, this task will modify the default loaded weights in the GUI system, which would be required if the model is changed. This task does not require any code modification – the maintainer needs only to replace the aforementioned file with a different one, using the same name.
* **Modifying the model:** Our model fits the currently placed weights, hence, any modification will cause the system to launch an error during initialization. This is intentional, since a different model necessitates a different set of weights. A maintainer can modify the code in the “my\_model.py”, import it to our training notebook, and train the model – then the new output weights can replace the old ones, as stated above.

To replace our model with a different one, first replace the import, here:

# model  
from my\_model import VisionTransformer

Then, it is required to update the configuration defined when instantiating the model, here:

def build\_model():  
 config\_vit = CONFIGS['R50-ViT-B\_16']  
 config\_vit.n\_classes = CFG.num\_classes  
 config\_vit.n\_skip = 3  
 config\_vit.patches.grid = (int(224 / 16), int(224 / 16))  
 model = VisionTransformer(config\_vit, img\_size=224, num\_classes=CFG.num\_classes)  
 model.to(CFG.device)  
 return model

## Future considerations

When using this model in clinical environment, consideration must be made to the nature of its topology. The 2.5D architecture requires the input to be one of 3 consecutive slices, placed at each of the RGB channels. This should not pose a clinical issue, as superior MRI scans tend to hold irrelevant data at the first and last slices (before and after reaching the organs of importance), but must be considered when feeding the data into our model – our training notebook provides a good example as to how this process is performed, added here:

def convert\_slices\_to\_npy(slice\_paths, output\_shape=(224, 224)):  
 # Read and resize each slice  
 slices = []  
 for slice\_path in slice\_paths:  
 slice\_path = DATASET\_PATH + slice\_path  
 img = cv2.imread(slice\_path, cv2.IMREAD\_GRAYSCALE)  
 resized\_img = cv2.resize(img, output\_shape)  
 slices.append(resized\_img)  
  
 # Stack the slices along the channel dimension  
 stacked\_slices = np.stack(slices, axis=-1)  
 # Normalize the pixel values to [0, 1]  
 normalized\_slices = stacked\_slices.astype(np.float32) / 255.0  
  
 return normalized\_slices

In order to use this function for inference, a transformation of the stacked slices is required before passing the slices into the model – a torch model expects an input of the shape [C, H, W], unlike the default image [H, W, C]. An example is provided in code:

# Prepare input tensor  
input\_tensor = torch.from\_numpy(np.transpose(normalized\_slices, (2, 0, 1))).unsqueeze(0).to(device)  
  
# Perform inference  
with torch.no\_grad():  
 outputs = model(input\_tensor)

# References

1. Jardim, S.R.; de Souza, L.M.P.; de Souza, H.S.P. “The Rise of Gastrointestinal Cancers as a Global Phenomenon: Unhealthy Behavior or Progress?” Int. J. Environ. Res. Public Health 2023, 20, 3640. <https://doi.org/10.3390/ijerph20043640>
2. happyharrycn, Maggie, Phil Culliton, Poonam Yadav, Sangjune Laurence Lee. (2022). UW-Madison GI Tract Image Segmentation. Kaggle. <https://kaggle.com/competitions/uw-madison-gi-tract-image-segmentation>
3. Vincenzo Catalano, Roberto Labianca et al. ELSEVIER Critical Reviews in Oncology/Hematology Volume 71, Issue 2, August 2009, Pages 127-164. <https://www.sciencedirect.com/science/article/abs/pii/S1040842809000043>
4. Jieneng Chen, Yongyi Lu, Qihang Yu et al. “Transformers Make Strong Encoders for Medical Image Segmentation” arXiv:2102.04306v1 [cs.CV] 8 Feb 2021. <https://arxiv.org/pdf/2102.04306>
5. Jieneng Chen, Yongyi Lu, Qihang Yu et al. “3D TransUNet: Advancing Medical Image Segmentation through Vision Transformers” arXiv:2310.07781v1 [cs.CV] 11 Oct 2023. <https://arxiv.org/pdf/2310.07781>
6. Neha Sharma, Sheifali Gupta et al. “U-Net Model with Transfer Learning Model as a Backbone for Segmentation of Gastrointestinal Tract” Bioengineering 2023. <https://www.mdpi.com/2306-5354/10c>
7. Xin Jiang, Yizhou Ding et al. “unified and simultaneous segmentation network for gastrointestinal images of CT & MRI”. ELSEVIER Computers in Biology and Medicine Volume 165, October 2023, 107326. <https://doi.org/10.1016/j.compbiomed.2023.107326>
8. Olaf Ronneberger, Philipp Fischer, Thomas Brox. “U-Net: Convolutional Networks for Biomedical

Image Segmentation”. arXiv:1505.04597v1 [cs.CV] 18 May 2015. <https://arxiv.org/pdf/1505.04597>

1. Reza Azad , Ehsan Khodapanah Aghdam , Amelie Rauland, Yiwei Jia et al. “Medical Image Segmentation Review: The Success of U-Net”. arXiv:2211.14830v1 [eess.IV] 27 Nov 2022. <https://arxiv.org/pdf/2211.14830>
2. Krithika Devi, K. Suganthi. “Review of Medical Image Synthesis using GAN Techniques”. ITM Web Conf. Volume 37, 2021. <https://www.itm-conferences.org/articles/itmconf/abs/2021/02/itmconf_icitsd2021_01005/itmconf_icitsd2021_01005.html>
3. Ashish Vaswani, Noam Shazeer et al. “Attention Is All You Need”. arXiv:1706.03762v7 [cs.CL] 2 Aug 2023. <https://arxiv.org/pdf/1706.03762>
4. Soufyane Ayanouz, Anouar abdelhakim Boudhir, Mohamed Ben Ahmed. “Unleashing the Power of NLP and Transformers: A Game-Changer in Medical Research and Clinical Practice and a revolution of Medical Text Analysis.: Case Study: Cancer report classification by priority”. NISS 2023. <https://dl.acm.org/doi/abs/10.1145/3607720.3607787>
5. Firuz Kamalov, David Santandreu Calonge, Ikhlaas Gurrib. “New Era of Artificial Intelligence in Education: Towards a Sustainable Multifaceted Revolution”. arXiv:2305.18303v2 [cs.CY] 25 Oct 2023. <https://arxiv.org/pdf/2305.18303>
6. Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov et al. “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. arXiv:2010.11929v2 [cs.CV] 3 Jun 2021. <https://arxiv.org/pdf/2010.11929>
7. Ozan Oktay, Jo Schlemper, Loic Le Folgoc et al. “Attention U-Net: Learning Where to Look for the Pancreas”. arXiv:1804.03999v3 [cs.CV] 20 May 2018. <https://arxiv.org/pdf/1804.03999>
8. Ahmed AL Qurri, Mohamed Almekkawy. “Improved UNet with Attention for Medical Image Segmentation”. MDPI. <https://www.mdpi.com/1424-8220/23/20/8589>
9. Salman Khan, Muzammal Naseer et al. “Transformers in Vision: A Survey”. arXiv:2101.01169v5 [cs.CV] 19 Jan 2022. <https://arxiv.org/pdf/2101.01169>
10. Noel E. Donlon, Robert Power et al. “Radiation and Immunotherapy in Upper Gastrointestinal Cancers: The Current State of Play”. MPDI. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7865314>
11. Grace W. Lindsay et al. “Attention in Psychology, Neuroscience, and Machine Learning”. PubMed. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7177153/>
12. Wang et al., "TransAttUNet: Multi-level Attention U-Net for Medical Image Segmentation," IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 10, pp. 4325–4336, Oct. 2021. <https://ieeexplore.ieee.org/document/9506750>
13. Chen et al., "Swin-Unet: Unet-like Pure Transformer for Medical Image Segmentation," arXiv preprint arXiv:2105.05537, 2021. <https://arxiv.org/abs/2105.05537>
14. Chen et al., "Memory-Efficient Transformers," arXiv preprint arXiv:2203.15556, 2022. <https://arxiv.org/abs/2203.15556>
15. Smith et al., "Don't Decay the Learning Rate, Increase the Batch Size," arXiv preprint arXiv:1711.00489, 2017. <https://arxiv.org/abs/1711.00489>
16. Chen et al., "U-Net++: Redesigning Skip Connections to Exploit Multiscale Features in Image Segmentation" (2018) <https://arxiv.org/abs/1807.10165>
17. Keskar et al., "On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima" (2017) <https://arxiv.org/abs/1609.04836>
18. Sharp et al., "Vision 20/20: Perspectives on Automated Image Segmentation for Radiotherapy" (2014) <https://onlinelibrary.wiley.com/doi/full/10.1002/acm2.12004>
19. Hu et al., "Squeeze-and-Excitation Networks," CVPR 2018 <https://arxiv.org/abs/1709.01507>
20. Woo et al., "CBAM: Convolutional Block Attention Module," ECCV 2018 [https://arxiv.org/abs/1807.06521](https://arxiv.org/pdf/1807.06521)