DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Few-Shot Segmentation for Medical Volumetric scans

Shayan Siddiqui

DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Few-Shot Segmentation for Medical Volumetric scans

Few-Shot Segmentierung für Medizinische Volumetrische Scans

Author:	Shayan Siddiqui
Supervisor:	Prof. Dr. Nassir Navab
Advisor:	Abhijit Guha Roy, Prof. Dr. Christian Wachinger
Submission Date:	26 February 2019

I confirm that this master's thesis in informatics is my own work and I have documented all sources and material used.

Munich, 26 February 2019 Shayan Siddiqui

Acknowledgments

This work would not have been possible without the help and support of many people who were involved at various stages and levels. I would like to start by expressing a deep gratitude towards my thesis advisor Mr.Abhijit Guha Roy, who is currently working with the department of AI-Med at Ludwig Maximilian University and is also affiliated to the Department of Informatics at Technical University of Munich, for his constant encouragement, support and valuable suggestions throughout the project work. His previous published work on segmentation found a strong basis for this research. His expert opinion and grasp on the subject along with the leadership skills like project planning and time management led to a time bound execution with excellent results.

A big thanks to Sebastian Pölsterl, who is a Post doctoral researcher at AI-Med. He was kind enough to share his opinions in the inception of new ideas during various brainstorming session. Because of his long experience in the field of research his feedback and guidance helped in improving the overall shape of this thesis.

I would also like to thank Prof. Dr. Christian Wachinger, who is heading the laboratory for Artificial Intelligence at Ludwig Maximilian University. He has always been a source of help by providing critical and constructive suggestions. I am extremely thankful to him for providing all the necessary resources and facilities for carrying out all the experiments required for the completion of this project work successfully.

At last my sincere thanks to the respective Program Head of the Chair for Computer Aided Medical Procedures, Prof. Dr. Nassir Navab at the Fakultät für Informatik, Technische Universität München for allowing me to carry out this project work and providing his valuable comments throughout this thesis.

Abstract

Deep neural networks enable highly accurate image segmentation, but require large amounts of manually annotated data for supervised training. Few-shot learning aims to address this shortcoming by learning a new class from a few annotated support examples. We introduce, for the first time, a novel few-shot framework, for the segmentation of volumetric medical images with only a few annotated slices. Compared to other related works in computer vision, the major challenges are the absence of pre-trained networks and the volumetric nature of medical scans. We address these challenges by proposing a new architecture for few-shot segmentation that incorporates 'squeeze & excite' blocks. Our two-armed architecture consists of a conditioner arm, which processes the annotated support input and generates a task representation. This representation is passed on to the segmenter arm that uses this information to segment the new query image. To facilitate efficient interaction between the conditioner and the segmenter arm, we propose to use 'channel squeeze & spatial excitation' blocks – a light-weight computational module – that enables heavy interaction between the both arms with negligible increase in model complexity. This contribution allows us to perform image segmentation without relying on a pre-trained model, which generally is unavailable for medical scans. Furthermore, we propose an efficient strategy for volumetric segmentation by optimally pairing a few slices of the support volume to all the slices of query volume. We perform the experiments for organ segmentation on whole-body contrast-enhanced CT scans from Visceral Dataset. Our proposed model outperforms multiple baselines and existing approaches with respect to the segmentation accuracy by a significant margin.

Zusammenfassung

Tiefe neuronale Netze ermöglichen eine hoch präzise Bildsegmentierung, erfordern aber große Mengen an manuell annotierten Daten für das überwachte Training. Das Few-Shot Lernen zielt darauf ab, diesen Mangel zu beheben, indem es eine neue Klasse aus einigen kommentierten Unterstützungsbeispielen lernt. Wir stellen zum ersten Mal ein neuartiges few-shot Framework für die Segmentierung von volumetrischen medizinischen Bildern mit nur wenigen Annotationen vor. Im Vergleich zu anderen Arbeiten im Bereich Maschinelles Sehen, sind die größten Herausforderungen das Fehlen vortrainierter Netzwerke und der volumetrische Charakter von medizinischen Bildern. Wir stellen uns diesen Herausforderungen, indem wir eine neue Architektur für die Few-Shot Segmentierung vorschlagen, die 'squeeze & excite' Blöcke beinhaltet. Unsere zweiarmige Architektur besteht aus einem Konditionierer-Arm, der die annotierte Unterstützungseingabe verarbeitet und eine abstrakte Darstellung erzeugt. Diese Darstellung wird an den Segmentierungsarm weitergegeben, der diese Informationen verwendet, um das neue Eingabebild zu segmentieren. Um eine effiziente Interaktion zwischen dem Konditionierer und dem Segmentierungsarm zu ermöglichen, schlagen wir vor, 'Channel Squeeze & spatial excitation' Blöcke zu verwenden – ein leichtgewichtiges Berechnungsmodul – das eine starke Interaktion zwischen den beiden Armen mit einer geringfügigen Erhöhung der Modellkomplexität ermöglicht. Unser Beitrag ermöglicht es, Bildsegmentierung durchzuführen, ohne auf ein vorab trainiertes Modell zurückzugreifen, welches für medizinische Bilder in der Regel nicht verfügbar ist. Darüber hinaus schlagen wir eine effiziente Strategie für die volumetrische Segmentierung vor, indem wir einige wenige Segmente des Unterstützungsbildes optimal mit allen Segmenten des Eingabevolumens paaren. Wir führen Experimente zur Organsegmentierung an Ganzkörper-Kontrast-verstärkten CT-Scans aus dem Visceral-Datensatz durch. Unser vorgeschlagenes Modell übertrifft vergleichbare Ansätze in Bezug auf die Segmentierungsgenauigkeit deutlich.

Contents

1 Introduction

Fully convolutional neural networks (F-CNNs) have achieved state-of-the-art performance in semantic image segmentation for both natural [1, 2, 3, 4] and medical images [5, 6, 7, 8]. Despite their tremendous success in image segmentation, they are of limited use when only a few labeled images are available. F-CNNs are in general highly complex models with millions of trainable weight parameters that require thousands of densely annotated images for training to be effective. A better strategy could be to adapt an already trained F-CNN model to segment a new semantic class from a few labeled images. This strategy often works well in computer vision applications where a pre-trained model is used to provide a good initialization and is subsequently fine-tuned with the new data to tailor it to the new semantic class. However, fine-tuning an existing pre-trained network without risking over-fitting still requires a fair amount of annotated images (atleast in the order of hundreds). When dealing in an extremely low data regime, where only a single or a few annotated images of the new class are available, such fine-tuning based transfer learning often fails and may cause overfitting [9, 10].

Few-shot learning is a machine learning technique that aims to address situations where an existing model needs to generalize to an unknown semantic class with a few examples at a rapid pace [11, 12, 13]. The basic concept of few-shot learning is motivated by the learning process of humans, where learning new semantics is done rapidly with very few observations, leveraging strong prior knowledge acquired from past experience. While few-shot learning for image classification and object detection is a well studied topic, few-shot learning for semantic image segmentation with neural networks has only recently been proposed [9, 10]. It is an immensely challenging task to make dense pixel-level high-dimensional predictions in such an extremely low data regime. But at the same time, few-shot learning could have a wide impact on medical image analysis, where scarcity of annotated data is a norm, due to the dependence on medical experts for carrying out manual labeling. In this thesis, for the first time, we propose a few-shot segmentation framework designed exclusively for segmenting volumetric medical scans. A Key to making this possible is to integrate the recently proposed 'squeeze & excite' blocks within the design of our new few-shot architecture [14].

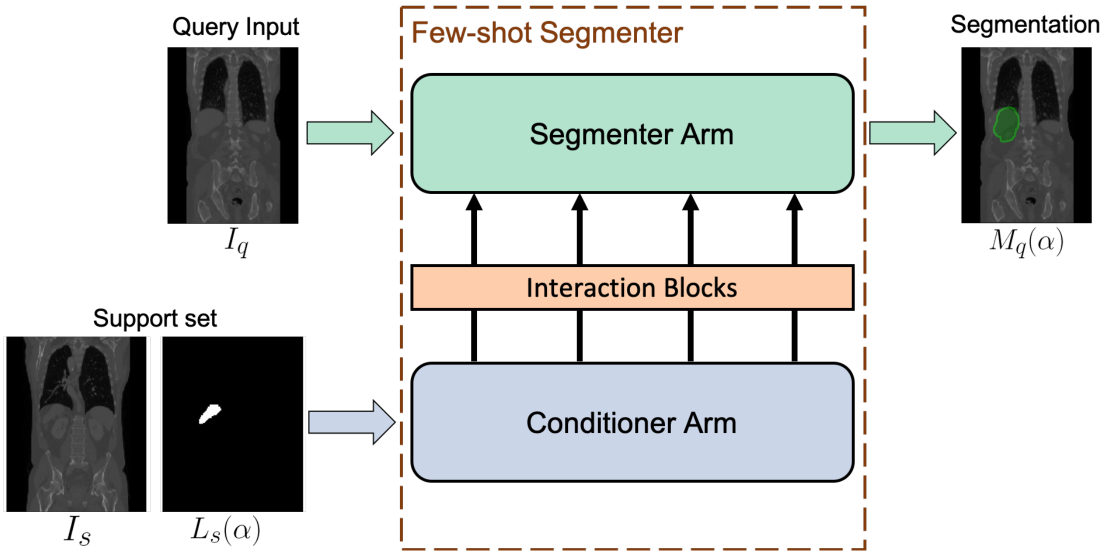


Figure 1.1: Overview of the few-shot segmentation framework. The support set consists of an image slice I_s and the corresponding annotation for the new semantic class $L_s(\alpha)$ (here α is the class liver). We pass the support set through the conditioner arm, whose information is conveyed to the segmenter arm via interaction blocks. The segmenter arm uses this information and segments a query input image I_q for the class α generating the label map $M_q(\alpha)$. Except for the support set, the few-shot segmenter has never seen annotations of a liver before.

1.1 Background on Few-Shot Segmentation

Few-shot learning algorithms try to generalize a model to a new, previously unseen class with only a few labeled examples by utilizing the previously acquired knowledge from differently labeled training data. Fig. 1.1 illustrates the overall setup, where we want to segment the liver in a new scan given the annotation of liver in only a single slice. A few-shot segmentation network architecture commonly consists of three parts: (i) a conditioner arm, (ii) a set of interaction blocks, and (iii) a segmentation arm. During inference, the model is provided with a support set $(I_s, L_s(\alpha))$, consisting of an image I_s with the new semantic class (or organ) α outlined as a mask $L_s(\alpha)$. In addition, a query image I_q is provided, where the new semantic class is to be segmented. The conditioner takes in the support set and performs a forward pass. This generates multiple feature maps of the support set in all the intermediate layers of the conditioner arm. This set of feature maps is referred to as *task representation* as they encode the information required to segment the new semantic class. The *task representation* is taken up by the interaction blocks, whose role is to pass the relevant information to the segmentation arm. The segmentation arm takes the query image as input, leverages the task information as provided by the interaction blocks and generates a segmentation mask M_q for the query input I_q. Thus, interaction blocks play a major role in passing the information from the conditioner to the segmenter, which forms the backbone for few-shot semantic image segmentation. Currently, weak interactions are used with a single connection at the last layer of the network [9, 10].

1.2 Challenges for Medical Few-Shot Segmentation

Existing work in computer vision on few-shot segmentation processes 2D RGB images and uses a pre-trained model for both segmenter and conditioner arm to aid the training. Pre-trained models provide a strong prior knowledge with effective features from the start of training. Hence, weak interaction between conditioner and segmenter is sufficient to train the model effectively. The direct extension to medical images is challenging due to the lack of pre-trained models. Instead, both the conditioner and the segmenter need to be trained from scratch. However, training the network in the absence of pre-trained models with weak interaction is prone to instability and mode collapse.

Instead of weak interaction, we propose a strong interaction at multiple locations between both the arms. The strong interaction facilitates effective gradient flow across the two arms, which eases the training of both the arms without the need for any pre-trained model. For effectuating the interaction, we propose our recently introduced 'channel squeeze & spatial excitation' (sSE) module [15, 14]. In our previous works, we used the sSE blocks for adaptive self re-calibration of feature maps to aid segmentation in a single segmentation network. Here, we use the sSE blocks to communicate between the two arms of the few-shot segmentation. The block takes as input the learned conditioner feature map and performs 'channel squeeze' to learn a spatial map. This is used to perform 'spatial excitation' on the segmenter feature map. We use sSE blocks between all the encoder, bottleneck and decoder blocks. SE blocks are well suited for effectuating the interaction between both the arms, as they are light-weight and therefore only marginally increase the model complexity. Despite which they can have a strong impact on the segmenter's features via re-calibration.

Existing work on few-shot segmentation focused on 2D images, while we are dealing with volumetric medical scans. Manually annotating organs on all slices in 3D images is time consuming. Following the idea of few-shot learning, the annotation should rather happen on a few sparsely selected slices. To this end, we propose a volumetric segmentation strategy by properly pairing a few annotated slices of the support volume with all the slices of the query volume, maintaining inter-slice consistency of the segmentation.

1.3 Contributions

In this work, we propose:

1. The first few-shot segmentation framework for volumetric medical scans.
2. Strong interactions at multiple locations between the conditioner and segmenter arms, instead of only one interaction at the final layer.
3. 'Squeeze & Excitation' modules for effectuating the interaction.
4. Stable training from scratch without requiring a pre-trained model.
5. A volumetric segmentation strategy that optimally pairs the slices of query and support volumes.

1.4 Overview

We discuss related work in Chapter. 2, present our few-shot segmentation algorithm in Chapter. 3, the experimental setup in Chapter. 4 and experimental results and discussion in Chapter. 5. We conclude with a summary of our contributions in Chapter. 6.

2 Prior Work

2.1 Few-Shot Learning

Methods for few-shot learning can be broadly divided into three groups. The first group of methods adapts a base classifier to the new class [16, 11, 17]. These approaches are often prone to overfitting as they attempt to fit a complex model on a few new samples. Methods in the second group aim to predict classifiers close to the base classifier to prevent overfitting. The basic idea is to use a two-branch network, where the first branch predicts a set of dynamic parameters, which are used by the second branch to generate a prediction [18, 19]. The third group contains algorithms that use metric learning. They try to map the data to an embedding space, where dissimilar samples are mapped far apart and similar samples are mapped close to each other, forming clusters. Standard approaches rely on Siamese architectures for this purpose [20, 21].

2.2 Few-Shot Segmentation using Deep Learning

Few-shot image segmentation with deep neural networks has been explored only recently. In one of the earliest work, *Caelles et al.* [22] leverage the idea of fine-tuning a pre-trained model with limited data. The authors perform video segmentation, given the annotation of the first frame. Although their model performed adequately in this application, such approaches are prone to overfitting and adapting a new class requires retraining, which hampers the speed of adaptation. *Shaban et al.* [9] use a 2-arm architecture, where the first arm looks at the new sample along with its label to regress the classification weights for the second arm, which takes in a query image and generates its segmentation. *Dong et al.* [23] extended this work to handle multiple unknown classes at the same time to perform multi-class segmentation. *Rakelly et al.* [10] took it to an extremely difficult situation where supervision of the support set is provided only at a few selected landmarks for foreground and background, instead of a densely annotated binary mask. Existing approaches for few-shot segmentation were evaluated on the PASCAL VOC computer vision benchmark [9, 10]. They reported low segmentation scores (mean intersection over union around 40%), confirming that few-shot segmentation is a very challenging task.

All of the above mentioned papers depend on pre-trained models to start the training process. Although access to pre-trained models is relatively easy for computer vision applications, no pre-trained models are available for medical imaging applications. Moreover, they use 2D RGB images, whereas we deal with 3D volumetric medical scans. This is more challenging because there is no established strategy to select and pair support slices with the query volume. This can lead to having situations where the query slice can be very different from the support slice or may not even contain the target class at all.

3 Methodology

In this section, we first introduce the problem setup, then detail the architecture of our network and the training strategy, and finally, describe the evaluation strategy for segmenting volumetric scans.

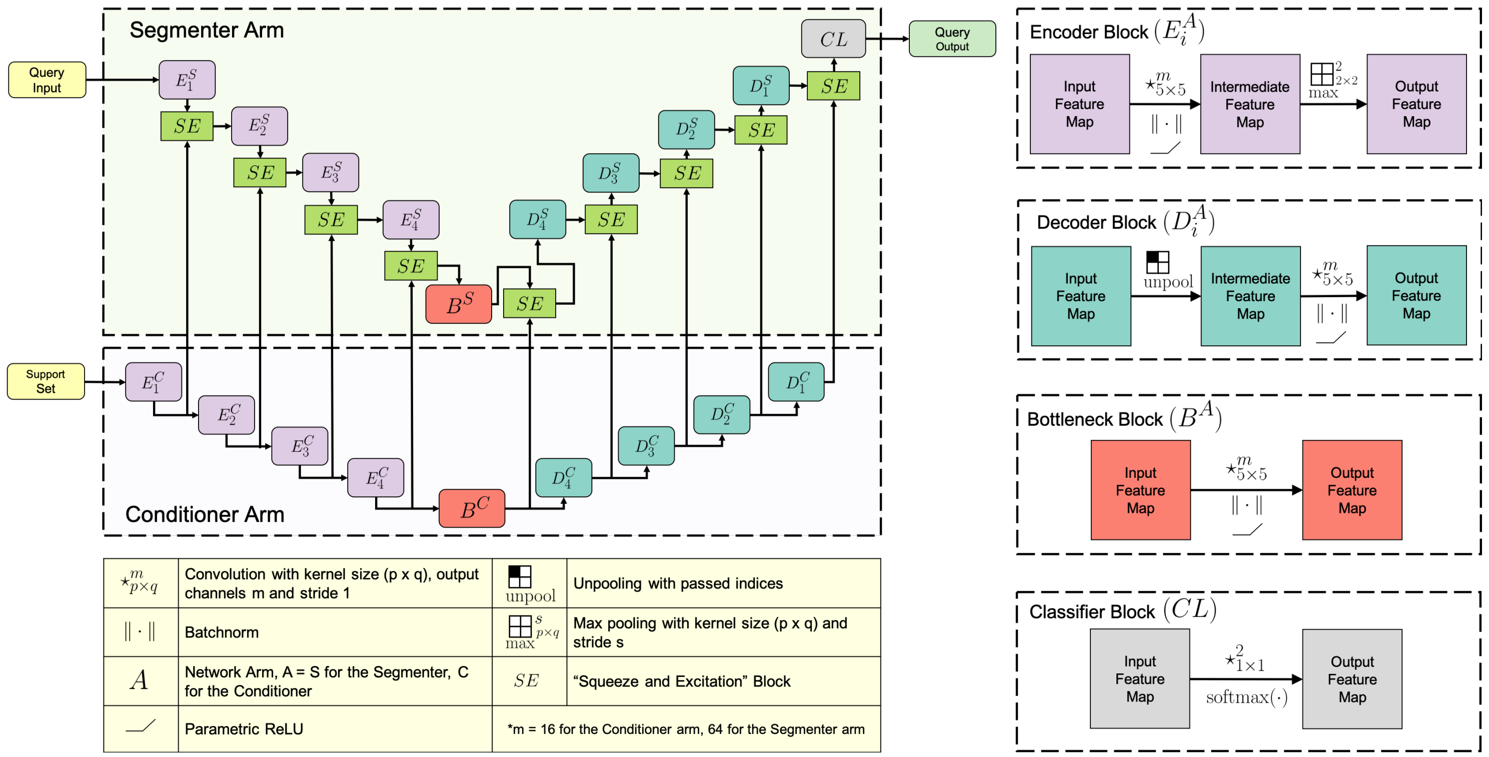
3.1 Problem Setup for Few-shot Segmentation

The training data for few-shot segmentation $\mathcal{D}_{\text{Train}} = \{(I_T^i, L_T^i(\alpha))\}_{i=1}^N$ comprises N pairs of input image I_T and its corresponding binary label map $L_T(\alpha)$ with respect to the semantic class (or organ) α. All the semantic classes α which are present in the label map $L_T^i \in \mathcal{D}_{\text{Train}}$ belong to the set $\mathcal{L}_{\text{Train}} = \{1, 2, \ldots, \kappa\}$, i.e., $\alpha \in \mathcal{L}_{\text{Train}}$. Here κ indicates the number of classes (organs) annotated in the training set. The objective is to learn a model $\mathcal{F}(\cdot)$ from $\mathcal{D}_{\text{Train}}$, such that given a support set $(I_s, L_s(\hat{\alpha})) \notin \mathcal{D}_{\text{Train}}$ for a new semantic class $\hat{\alpha} \in \mathcal{L}_{\text{Test}}$ and a query image I_q, the binary segmentation $M_q(\hat{\alpha})$ of the query is inferred. Fig. 1.1 illustrates the setup for the test class $\hat{\alpha}$ = liver for CT scans. The semantic classes for training and testing are mutually exclusive, i.e., $\mathcal{L}_{\text{Train}} \cap \mathcal{L}_{\text{Test}} = \emptyset$.

One fundamental difference of few-shot segmentation to few-shot classification or object detection is that test classes $\mathcal{L}_{\text{Test}}$ might already appear in the training data as the background class. For instance, the network has already seen the liver on many coronal CT slices as part of the background class, although liver was not a part of the training classes. This forms a prior knowledge that can be utilized during testing, when only a few examples are provided with the liver annotated.

3.2 Architectural Design

As mentioned earlier, our network architecture consists of three units: (i) a conditioner arm, (ii) interaction blocks with sSE modules, and (iii) a segmenter arm. The conditioner arm processes the support set to model how a new semantic class (organ) looks like in an image. It efficiently conveys the information to the segmenter arm through the interaction blocks. The segmenter arm segments the new semantic class in a new query image by utilizing the information provided by the interaction blocks. Figs. 3.1 and 3.2



$\star^m_{p\times q}$	Convolution with kernel size (p x q), output channels m and stride 1	unpool	Unpooling with passed indices
$\|\cdot\|$	Batchnorm	$\max^s_{p\times q}$	Max pooling with kernel size (p x q) and stride s
A	Network Arm, A = S for the Segmenter, C for the Conditioner	SE	"Squeeze and Excitation" Block
	Parametric ReLU	*m = 16 for the Conditioner arm, 64 for the Segmenter arm	

Figure 3.1: Illustration of the architecture of the few-shot segmenter. To the left, we show a block diagram with arrows illustrating the encoder-decoder based conditioner arm (bottom) and segmenter arm (top). Interaction between them is shown by SE blocks, which is detailed in Fig. 3.2. To the right, the operational details of the encoder block, decoder block, bottleneck block and the classifier block are provided.

illustrate the architecture in further detail, which is also described in the following sections.

In our framework, we choose the segmenter and conditioner to have a symmetric layout, i.e., both have four encoder and decoder blocks separated by a bottleneck block. The symmetric layout helps in having a strong interaction between matching blocks, as feature maps have the same spatial size. In existing approaches, conditioner and segmenter only interact via the final layer, before generating segmentation maps [9, 10]. Such weak interaction at a single location was sufficient for their application, because they were able to use a pre-trained model, which already provides reasonably good features. As we do not have a pre-trained network, we propose to establish a strong interaction by incorporating the sSE blocks at multiple locations. Such interactions facilitate training the model from scratch.

3.2.1 Conditioner Arm

The task of the conditioner arm is to process the support set by fusing the visual information of the support image I_s with the annotation L_s, and generate task-representative feature maps, capable of capturing what should be segmented in the query image I_q. We refer to the intermediate feature maps of the conditioner as *task representation*. We provide a 2-channel input to the conditioner arm by stacking I_s and binary map $L_s(\alpha)$. This is in contrast to *Shaban et al.* [9], where they multiplied I_s and $L_s(\alpha)$ to generate the input. Their motivation was to supress the background pixels so that the conditioner can focus on the patterns within the object (like eyes or nose patterns within a cat class). This does not hold true for our scans due to the limited visual patterns within an organ class. For example, voxel intensities within the liver are quite homogeneous with limited edges. Thus, we feed both parts of the support set to the network and let it learn the optimal fusion which provides the best possible segmentation of the query image.

The conditioner arm has an encoder-decoder based architecture consisting of four encoder blocks, four decoder blocks separated by a bottleneck layer, see Fig. 3.1. Both encoder and decoder blocks consist of a generic block constituting a convolutional layer with kernel size of 5×5, stride of 1 and 16 output feature maps, followed by a parametric ReLU activation function [24] and a batch normalization layer. In the encoder block, the generic block is followed by a max-pooling layer of 2×2 and stride 2, which reduces the spatial dimension by half. In the decoder block, the generic block is preceded by an unpooling layer [4]. The pooling indices during the max-pool operations are stored and used in the corresponding unpooling stage of decoder block for up-sampling the feature map. Not only is the unpooling operation parameter free which reduces the model complexity, but it also aids to preserve the spatial consistency for fine-grained segmentation. Furthermore, it must be noted that *no skip connections* are used between the encoder and decoder blocks unlike the standard U-net architecture [5]. The reason for this important design choice is discussed in Sec. 5.2.

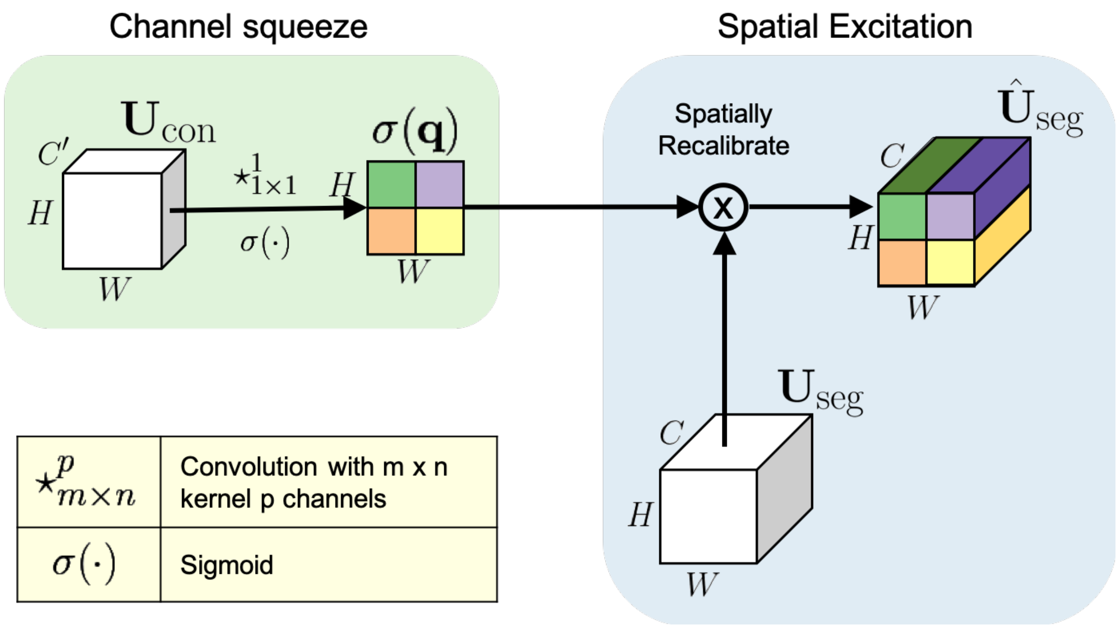


Figure 3.2: Illustration of the architecture of the 'channel squeeze & spatial excitation' (sSE) module, which is used as the interaction block within the few-shot segmenter. The block takes a conditioner feature map $\mathbf{U}_{\text{con}}$ and a segmenter feature map $\mathbf{U}_{\text{seg}}$ as inputs. 'Channel squeeze' is performed on $\mathbf{U}_{\text{con}}$ to generate a spatial map $\sigma(\mathbf{q})$, which is used for 'spatial excitation' of $\mathbf{U}_{\text{seg}}$, which promotes the interaction.

3.2.2 Interaction Block using 'Squeeze & Excitation' modules

The interaction blocks play a key role in the few-shot segmentation framework. These blocks take the *task representation* of the conditioner as input and convey them to the segmenter to steer segmentation of the query image. Ideally these blocks should: (i) be light-weight to only marginally increase the model complexity and computation time, and (ii) improve the trainability of network by improving the gradient flow.

We use the recently introduced 'Squeeze & Excitation' (SE) modules for this purpose. SE modules are computational units to achieve adaptive re-calibration of feature maps within any CNN [25]. SE blocks can boost the performance of CNNs, while increasing model complexity only marginally. For classification [25], the feature maps are spatially squeezed to learn a channel descriptor, which is used to excite (or re-calibrate) the feature map, emphasizing certain important channels. We refer to it as spatial squeeze and channel excitation block (cSE). In our recent work, we extended the idea to segmentation, where re-calibration was performed by squeezing channel-wise and exciting spatially (sSE), emphasizing relevant spatial locations [14, 15]. In both the cases, SE blocks are used for self re-calibration, i.e, the same feature map is used as input for squeezing and excitation operations. However, here we propose to use SE blocks for the interaction between the conditioner and the segmenter. The conditioner feature maps are taken as input for the squeezing operation and its outputs are used to excite the segmentation feature maps as detailed below.

Channel Squeeze & Spatial Excitation (sSE) The sSE block squeezes a conditioner feature map $\mathbf{U}_{\text{con}} \in \mathbb{R}^{H\times W\times C'}$ along the channels and excites the corresponding segmenter feature map $\mathbf{U}_{\text{seg}} \in \mathbb{R}^{H\times W\times C}$ spatially, conveying the information from the support set to aid the segmentation of query image. H, W are the height and width of feature maps, C' and C are the number of channels for the conditioner and the segmenter feature maps, respectively. Here, we consider a particular slicing strategy to represent the input tensor $\mathbf{U}_{\text{con}} = [\mathbf{u}_{\text{con}}^{1,1}, \mathbf{u}_{\text{con}}^{1,2} \dots, \mathbf{u}_{\text{con}}^{j,\imath}, \dots, \mathbf{u}_{\text{con}}^{H,W}]$, where $\mathbf{u}_{\text{con}}^{j,\imath} \in \mathbb{R}^{1\times 1\times C'}$ with $j \in \{1, 2, \dots, H\}$ and $\imath \in \{1, 2, \dots, W\}$. Similarly for segmenter feature map $\mathbf{U}_{\text{seg}} = [\mathbf{u}_{\text{seg}}^{1,1}, \mathbf{u}_{\text{seg}}^{1,2} \dots, \mathbf{u}_{\text{seg}}^{j,\imath}, \dots, \mathbf{u}_{\text{seg}}^{H,W}]$. The spatial squeeze operation is performed using a convolution $\mathbf{q} = \mathbf{W}_{sq} \star \mathbf{U}_{\text{con}}$ with $\mathbf{W}_{sq} \in \mathbb{R}^{1\times 1\times C'}$, generating a projection tensor $\mathbf{q} \in \mathbb{R}^{H\times W}$. This projection $\mathbf{q}$ is passed through a sigmoid gating layer $\sigma(\cdot)$ to rescale activations to $[0, 1]$, which is used to re-calibrate or excite $\mathbf{U}_{\text{seg}}$ spatially to generate

$$\hat{\mathbf{U}}_{\text{seg}} = [\sigma(q_{1,1})\mathbf{u}_{\text{seg}}^{1,1}, \dots, \sigma(q_{j,k})\mathbf{u}_{\text{seg}}^{j,\imath}, \dots, \sigma(q_{H,W})\mathbf{u}_{\text{seg}}^{H,W}]. \tag{3.1}$$

The architectural details of this module are presented in Fig. 3.2.

3.2.3 Segmenter Arm

The goal of the segmenter arm is to segment a given query image I_q with respect to a new, unknown class α, by using the information passed by the conditioner, which captures a high-level information about the previously unseen class α. The sSE modules in the interaction block compresses the *task representation* of the conditioner and adaptively re-calibrates the segmenter's feature maps by spatial excitation.

The encoder-decoder architecture of the segmenter is similar to the conditioner, with a few differences. Firstly, the convolutional layers of both the encoder and decoder blocks in the segmenter have 64 output feature maps, in contrast to 16 in the conditioner. This provides the segmenter arm with a higher model complexity than the conditioner arm. We will justify this choice in Sec. 5.3. Secondly, unlike the conditioner arm, the segmenter arm provides a segmentation map as output, see Fig. 3.1. Thus a classifier block is added, consisting of a 1×1 convolutional layer with 2 output feature maps (foreground, background), followed by a soft-max function for inferring the segmentation. Thirdly, in the segmenter, after every encoder, decoder and bottleneck block, the interaction block re-calibrates the feature maps, which is not the case in the conditioner arm.

3.3 Training Strategy

We use a similar training strategy to *Shaban et al.* [9]. We simulate the one-shot segmentation task with the training data $\mathcal{D}_{\text{Train}}$ in the following manner.

Batch Sampler In each iteration, we first randomly sample a label $\alpha \in \mathcal{L}_{\text{Train}}$. Next, we randomly select 2 image slices and their corresponding label maps, containing the semantic label α, from training data $\mathcal{D}_{\text{Train}}$. The label maps are binarized representing semantic class α as foreground and the rest as background. One pair constitutes the support set $(I_s, L_s(\alpha))$ and the other pair the query set $(I_q, L_q(\alpha))$, where $L_q(\alpha)$ serves as ground truth segmentation for computing the loss.

Training The support pair $(I_s, L_s(\alpha))$ is provided as input to the conditioner arm. The query image I_q is provided as input to the segmentation arm. The network predicts the segmentation $M_q(\alpha)$ for the query image I_q for label α with one forward pass. We use the Dice loss [6] as the cost function, which is computed between the prediction $M_q(\alpha)$ and the ground truth $L_q(\alpha)$. The learnable weight parameters of the network are optimized using stochastic gradient descent (SGD) with momentum.

3.4 Volumetric Segmentation Strategy

As mentioned in the previous section, the network is trained with 2D images as support set and query. But, during the testing phase, a 3D query volume needs to be segmented. Therefore, from the support volume, we need to select a sparse set of annotated slices that form the support set. A straight forward extension for segmenting the query volume is challenging as there is no established strategy to pair the above selected support slices to all of the slices of query volume, which would yield the best possible segmentation. In this section, we propose a strategy to tackle this problem.

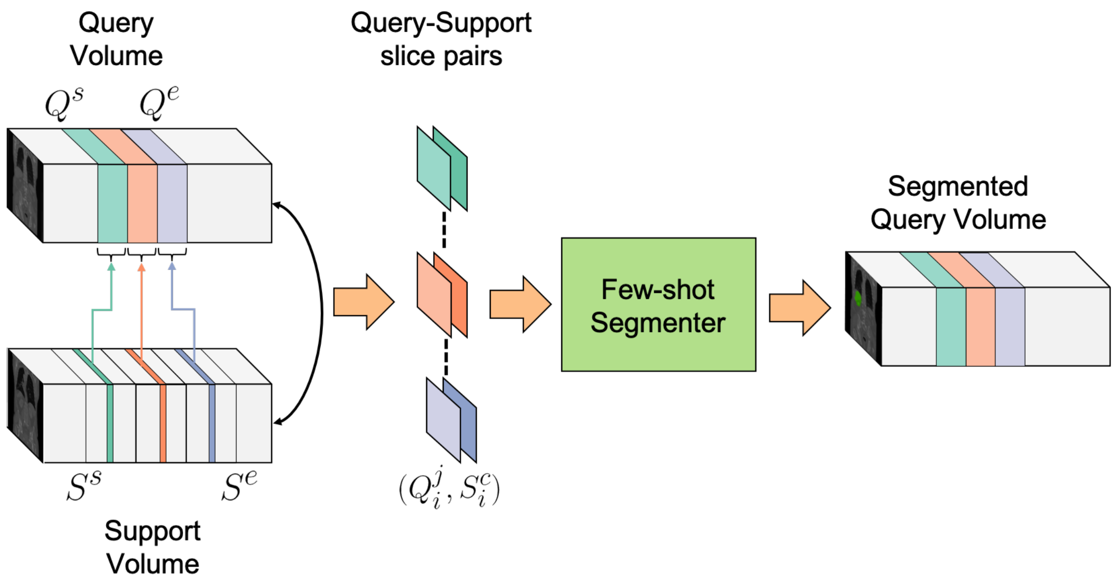


Figure 3.3: Illustration of the few-shot volumetric segmentation strategy for $k = 3$. We divide both the query volume and support volume into k group of slices. The annotated center slice of the i^{th} group in the support volume is paired with all the slices of i^{th} group of query volume to infer their segmentation. This is done for $i \in \{1, 2, 3\}$ and is passed to the few-shot segmenter for segmenting the whole volume.

Assume we have a budget of annotating only k slices in the support volume, a query volume is segmented with the following procedure, where the extent of the organs is specified in both volumes.

1. Given a semantic class, we first indicate the range of slices (along a fixed orientation) where the organ lies for both support and query volume. Let us assume the ranges are $[S^s, S^e]$ for the support and $[Q^s, Q^e]$ for the query volume. Here the superscript indicates the start s and end e slice indices.

2. Based on the budget k, both ranges $[S^s, S^e]$ and $[Q^s, Q^e]$ are divided into k equi-spaced group of slices. Let us indicate the groups by $[\{S_1^i\}, \ldots, \{S_k^i\}]$ and $[\{Q_1^i\}, \ldots, \{Q_k^i\}]$ respectively. Here the subscript indicates the group number.

3. In each of the k support volume groups, center slices $[S_1^c, \ldots, S_k^c]$ are annotated to serve as the support set.

4. We pair the annotated center slice S_j^c with all the slices of the group $\{Q_j^i\}$ for all $i \in \{1, \ldots, k\}$. This forms the input for the segmenter and the conditioner to generate the final volume segmentation.

The overall process of volumetric evaluation is illustrated in Fig. 3.3. In our experiments, we observed that if the support slice and query slice are similar, segmentation performance is better than if they were very dissimilar. This can be intuitively understood as the quality of support slice has a major impact on the segmenter's performance. In our evaluation strategy, for a fixed budget k, we made sure that the dissimilarity between the support slice and the corresponding query slice is minimal.

4 Dataset and Experimental Setup

4.1 Dataset Description

We choose the challenging task of organ segmentation from contrast-enhanced CT (ceCT) scans, for evaluating our few-shot volumetric segmentation framework. We use the Visceral dataset [26], which consists of two parts (i) silver corpus (with 65 scans) and (ii) gold corpus (20 scans). All the scans were resampled to a voxel resolution of 2mm^3.

4.2 Problem Formulation

As there is no existing benchmark for few-shot image segmentation on volumetric medical images, we formulate our own experimental setup for the evaluation. We use the silver corpus scans for training ($\mathcal{D}_{\text{Train}}$). For testing, we use the gold corpus dataset where one volume is used to create the support set (Volume ID: `10000132_1_CTce_ThAb`) and evaluation is done on the remaining 19 volumes.

We consider the following six organs as semantic classes in our experiments:

1. Liver
2. Spleen
3. Right Kidney
4. Left Kidney
5. Right Psoas Muscle
6. Left Psoas Muscle

We perform experiments with 4 Folds, such that each organ is considered as an unknown semantic class once per-fold. The training and testing labels for each of the folds are reported in Tab. 4.1.

Table 4.1: Semantic labels used for training and testing in all the experimental folds. Left and Right are abbreviated as L. and R. Psoas Muscle is abbreviated as P.M.

	Fold 1	Fold 2	Fold 3	Fold 4
Liver	*Test*	Train	Train	Train
Spleen	Train	*Test*	Train	Train
L./R. Kidney	Train	Train	*Test*	Train
L./R. P. M.	Train	Train	Train	*Test*

4.3 Hyperparameters for Training the Network

Due to the lack of pre-trained models, we could not use the setup from *Shaban et al.* [9] for training. Thus, we needed to define our own hyperparameter settings, listed in Table 4.2. Please note that the hyperparameters were estimated by manually trying out different combinations, rather than employing a hyperparameter optimization framework, which could lead to better results but is time-consuming at the same time.

Table 4.2: List of hyperparameters used for training the few-shot segmenter.

Hyperparameters	Value
Learning Rate	0.01
Weight decay constant	10^{-4}
Momentum	0.99
No. of epochs	10
Iterations per epoch	500

5 Experimental Results and Discussion

5.1 'Squeeze & Excitation' based Interaction

In this section, we investigate the optimal positions of the SE blocks for facilitating interaction. Furthermore, we compare the performance of cSE and sSE blocks. Here, we set the number of convolution kernels of the conditioner arm to 16 and the segmenter arm to 64. We use $k = 12$ support slices from the support volume. Since the aim of this experiment is to evaluate the position and the type of SE blocks, we keep the above parameters fixed, but evaluate them later. With four different possibilities of placing the SE blocks and two types cSE or sSE, we have a total of 8 different baseline configurations. The configuration of each of these baselines and their corresponding segmentation performance per fold is reported in Tab. 5.1.

Firstly, one observes that BL-1, 3, 5, 7 with sSE have a decent performance, whereas BL-2, 4, 6, 8 have a very poor performance (less than 0.1 Dice score). This demonstrates that sSE interaction modules are by far superior to cSE modules in this application of few-shot segmentation. One possible reason might be the flow of gradients from the segmenter to the conditioner. sSE connects the two arms through more units of the hidden layer in comparison to cSE, thus facilitating proper training of the conditioner by denser gradient flow.

Secondly, out of all the possible positions of the interaction block, BL-7, i.e., sSE blocks between all encoder, bottleneck and decoder blocks achieved the highest Dice score of 0.555. This performance is also consistent across all the folds. For Fold-1 (where liver is the test label), BL-1, BL-5 and BL-7 provided the same segmentation performance. Whereas, BL-7 outperformed the remaining baselines for Fold-2 (spleen), Fold-3 (L/R kidney) and Fold-4 (L/R psoas muscle) by a margin of 0.1 to 0.3 Dice points. This might be related to the relative difficulty associated with each organ. Due to the contrast and size, the liver is relatively easy to segment in comparison to spleen, kidney, and psoas muscles . From these results, we conclude that interaction blocks based on sSE are most effective and we use sSE-based interactions between all encoder, bottleneck and decoder blocks in subsequent experiments.

Table 5.1: The performance of our few-shot segmenter (per-fold and mean Dice score) by using either sSE or cSE module, at different locations (encoder, bottleneck and decoder) of the network. Left and Right are abbreviated as L. and R. Psoas Muscle is abbreviated as P.M.

	Position of SE			Type of SE		Dice Score on Test set				
	Encoder	Bottleneck	Decoder	Spatial	Channel	Liver	Spleen	L/R kidney	L/R P.M.	Mean
BL-1	✓	×	×	✓	×	0.672	0.504	0.361	0.334	0.468
BL-2	✓	×	×	×	✓	0.068	0.018	0.077	0.016	0.045
BL-3	×	✓	×	✓	×	0.651	0.308	0.307	0.242	0.377
BL-4	×	✓	×	×	✓	0.062	0.016	0.093	0.033	0.051
BL-5	×	×	✓	✓	×	0.682	0.520	0.276	0.167	0.411
BL-6	×	×	✓	×	✓	0.051	0.014	0.013	0.003	0.020
BL-7	✓	✓	✓	✓	×	0.685	0.605	0.437	0.491	**0.555**
BL-8	✓	✓	✓	×	✓	0.026	0.003	0.002	0.002	0.008

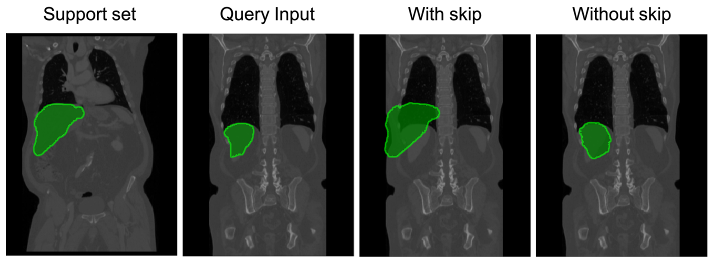
5.2 Effect of Skip Connections in the Architecture

Due to the success of the U-net architecture [5], using skip connections in F-CNN models has become a very common design choice. With skip connections, the output feature map of an encoder block is concatenated with the input of the decoder block with an identical spatial resolution. In general, this connectivity aids in achieving a superior segmentation performance as it provides a high contextual information in the decoding stage and facilitates the gradient flow. In our experiments, we intuitively started off with having skip connections in both the conditioner arm and the segmenter arm, but observed an unexpected behavior in the predicted query segmentation masks. By including skip connections, the network mostly copies the binary mask of the support set to the output. This is observed for all the folds both in train and test set. We refer to this phenomenon as the *copy over effect*. A set of qualitative examples of this is illustrated for each fold in Fig 5.1, where we see that, inspite of the support and the query images having different shapes, the prediction on the query image is almost identical to the support binary mask. We observed this for all the folds on both train and test data.

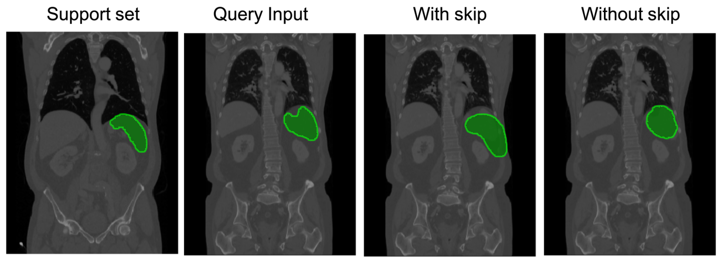
We also performed a quantitative analysis to observe the effect on Dice scores due to this *copy over effect*. Table 5.2 reports the performance with and without skip connections, where we observe a 7% decrease in Dice points due to the addition of skip connections. We also performed experiments by separately adding the skip connections in the conditioner and the segmenter arm. We observe that the inclusion of skip connections only in the conditioner arm reduced the performance by 5% Dice points, whereas adding them only in the segmenter arm made the training unstable. For this evaluation, the number of convolution kernels for conditioner and segmenter were fixed at 16 and 64, respectively, and the evaluation was conducted with $k = 12$ support slices.

5.3 Model Complexity of the Conditioner Arm

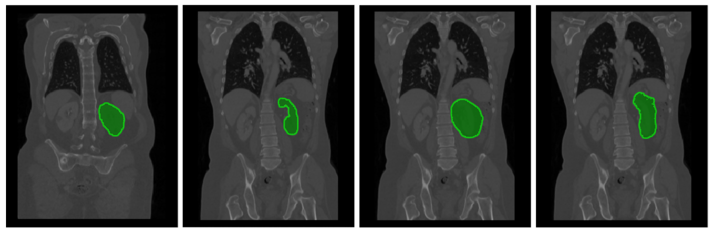
One important design choice is to decide the relative model complexity of the conditioner arm compared to the segmenter arm. As mentioned in Sec. 1.1, the conditioner takes in the support example and learns to generate *task representation*, which are passed to the segmenter arm through interaction blocks. This is utilized by the segmenter to segment the query image. We fix the number of kernels of the convolutional layers (for every encoder, bottleneck and decoder) for the segmenter arm to 64. We use this setting as this has proven to work good in our prior segmentation works across different datasets [8, 14]. Then, we vary the number of kernels of the conditioner arm to $\{8, 16, 32, 64\}$. The number of support slices was fixed to $k = 12$. We report the



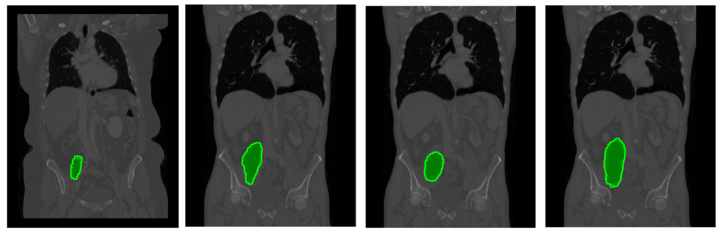
(a) Fold 1 – Liver



(b) Fold 2 – Spleen



(c) Fold 3 – Kidney



(d) Fold 4 – Psoas Muscle

Figure 5.1: Qualitative results of few-shot segmenter with and without skip connections to demonstrate the *copy over effect*. The sub-figures (a-d) refer to the examples from each of the folds namely liver, spleen, left kidney and right psoas muscles, respectively. For each sub-figure, the first column indicates the support image with the manual outline of the organ, the second column indicates the query image with manual annotation, the third column indicates the prediction of the query image with skip connection, and the fourth column indicates the prediction of the query image without skip connections (proposed approach). All annotations are shown in green. A clear *copy over effect* can be observed for all the folds when analyzing the mask of the support annotation and the prediction with skip connections.

Table 5.2: The segmentation performance (per-fold and mean Dice score) on test scans, with and without using skip connections within our few-shot segmenter. Left and Right are abbreviated as L. and R. Psoas Muscle is abbreviated as P.M.

Skip Connections		Dice Score on Test set				
Conditioner	Segmenter	Liver	Spleen	L/R kidney	L/R P.M.	Mean
×	×	0.685	0.605	0.437	0.491	**0.555**
✓	×	0.621	0.495	0.457	0.447	0.505
×	✓	0.101	0.025	0.026	0.020	0.043
✓	✓	0.487	0.570	0.452	0.498	0.486

Table 5.3: Effect of model complexity of the conditioner arm (Number of convolution kernels) on segmentation performance, provided a fixed model complexity (Number of convolution kernels fixed to 64) of the segmenter arm. Left and Right are abbreviated as L. and R. Psoas Muscle is abbreviated as P.M.

Channels in	Dice Score on Test set				
Conditioner Arm	Liver	Spleen	L/R kidney	L/R P.M.	Mean
8	0.623	0.251	0.390	0.273	0.384
16	0.685	0.605	0.437	0.491	**0.555**
32	0.618	0.505	0.352	0.279	0.439
64	0.650	0.343	0.368	0.238	0.400

segmentation results of these settings in Table 5.3. The best performance was observed for the conditioner with 16 convolution kernels. One possible explanation of this can be that, too low conditioner complexity (like 8) leads to a very weak *task representation*, thereby failing to provide a reliable support to the segmenter arm. Whereas, higher conditioner arm complexity like 32 and even 64 (same as segmenter complexity) kernels, might lead to improper training due to increased complexity under limited training data and interaction. We fix the number of conditioner convolution kernels to 16 in our following experiments.

5.4 Effect of the number of Support Slice Budget

In this section, we investigate the performance when changing the budget for the number of support slices k selected from the support volume for segmenting all the query volumes. Here k can be thought of as the 'number of shots' for volumetric segmentation. We vary k between $\{1, 3, 5, 7, 10, 12, 15, 17, 20\}$ and report the per-fold and overall mean segmentation performance in Table 5.4. The per-fold performance analysis revealed that the minimum number of slices needed for a decent accuracy varies with the size of the target organ to be segmented.

For Fold-1 (liver), one-shot volume segmentation ($k = 1$) yielded a Dice score of 0.666 which increased to 0.688 with $k = 20$. We observed a saturation in performance (Dice score of 0.68) with only 3 slices. The target organ liver is comparatively larger in volume and simple to segment, only $k = 3$ slices are needed for a decent segmentation. The segmentation performance only marginally increased with higher value of k. For Fold-2 (spleen), the segmentation performance initially increases with the increase in the value of k and then the performance saturates with $k \geq 10$ at a Dice score of 0.60. The spleen is more difficult to segment than liver, thus requires more support. For Fold-3 (right/ left kidney), we observe a similar behavior as Fold-2. The segmentation performance increases initially with increase in the value of k and then saturates at a Dice score of 0.44 (this is the mean between the two classes, left and the right kidney) at $k \geq 10$. Also for Fold-4 (right/ left psoas muscle), we see the Dice score saturates at 0.50 for $k = 10$. Overall mean Dice score across all the folds also saturates at 0.55 with $k = 10$.

Based on these results, we conclude that $k = 10$ is the maximum number of support slices required for our application, used in the next experiments.

5.5 Dependence on Support set

In all our previous experiments, one volume (`10000132_1_CTce_ThAb`) was used as a support volume and the remaining 19 as query volumes for evaluation purposes. In this section, we investigate the sensitivity of segmentation performance on the selection of the support volume. In this experiment, we randomly choose 5 volumes as support set. We select one at a time and evaluate on the remaining 15 volumes and report the per-fold and global Dice scores in Table 5.5.

We observe that changing the support volume does have an effect on the segmentation performance. In Fold-1 (liver), the performance varies by 6% Dice points across all the 5 selected support volume. This change is 5%, 8% and 5% Dice points for Fold-2 (spleen), Fold-3 (R/L kidney), Fold-4 (R/L psoas muscle), respectively. The overall mean Dice

Table 5.4: The segmentation performance (per-fold and mean Dice score) on test scans, by varying the number of annotated slice (k) as support in the support volume. Left and Right are abbreviated as L. and R. Psoas Muscle is abbreviated as P.M.

No. of support	Dice Score on Test set				
slices (k)	Liver	Spleen	L/R kidney	L/R P.M.	Mean
1	0.666	0.490	0.362	0.387	0.476
3	0.679	0.437	0.373	0.452	0.485
5	0.678	0.534	0.405	0.491	0.528
7	0.678	0.548	0.421	0.499	0.537
10	0.680	0.597	0.441	0.500	0.554
12	0.685	0.605	0.437	0.491	0.555
15	0.685	0.602	0.438	0.490	0.553
17	0.689	0.604	0.434	0.489	0.554
20	0.688	0.606	0.438	0.490	0.556

scores vary by 4% points. We conclude that it is important to select an appropriate support volume that is representative of the whole query set. Yet, a good strategy for making the selection remains as a future work. Nevertheless, our framework shows some robustness to the selection.

5.6 Comparison with existing approaches

In this section, we compare our proposed framework against the other existing few-shot segmentation approaches. It must be noted that all of the existing methods were proposed for computer vision applications and thus cannot directly be compared against our approach as explained in Sec. 1.2. Hence, we modified each of the existing approaches to suit our application. The results are summarized in Table 5.6.

First, we try to compare against *Shaban et al.* [9]. Their main contribution was that the conditioner arm regresses the convolutional weights, which are used by the classifier block of the segmenter to infer the segmentation of the query image. As we do not have any pre-trained models for our application unlike [9], we use the same architecture as our proposed method for the segmenter and conditioner arms. No intermediate interactions were used other than the final classifier weight regression. We attempted to train the network on our dataset with a wide range of hyperparameters, but all the

Table 5.5: The segmentation performance (per-fold and mean Dice score) on test scans, by using different volumes (Volume ID indicated in the first column) as the support volume. Left and Right are abbreviated as L. and R. Psoas Muscle is abbreviated as P.M.

Support	Dice Score on 15 Test set				
Volume ID	Liver	Spleen	L/R kidney	L/R P.M.	Mean
`10000100_1_CTce_ThAb`	0.748	0.550	0.445	0.454	0.550
`10000106_1_CTce_ThAb`	0.690	0.514	0.444	0.464	0.528
`10000108_1_CTce_ThAb`	0.718	0.560	0.406	0.465	0.537
`10000113_1_CTce_ThAb`	0.689	0.505	0.392	0.453	0.510
`10000132_1_CTce_ThAb`	0.694	0.533	0.369	0.501	0.524

settings led to instability while training. It must be noted that one possible source of instability might be that we do not use a pre-trained model, unlike the original method. This also substantiates our claim that, when training from scratch, spatial SE based interaction at multiple locations between the two networks are essential.

Next, we compare our approach against *Rakelly et al.* [10]. Again, this approach is not directly comparable to our approach due to the lack of a pre-trained model. One of the main contributions of their approach was the interaction strategy between the segmenter and the conditioner using a technique called *feature fusion*. They tiled the feature maps of the conditioner and concatenated them with the segmenter feature maps. We modified our model by introducing the concatenation based feature fusion instead of sSE modules at multiple locations between the conditioner and segmenter arms. As we have a symmetric architecture no tiling was needed. Similar to our proposed approach, we introduced this feature fusion based interaction at every encoder, bottleneck and decoder block. So, in this experiment, we are comparing our spatial SE based interaction approach to the concatenation based feature fusion approach. The results are reported in Table 5.6. We observe 19% higher Dice points for our approach.

Next, we attempted to create hybrid baselines by combining the feature fusion approach [10] with classifier weight regression approach [9]. We observe that by doing so the performance increased by 6% Dice points. Still, it had a much lower Dice score in comparison to our proposed approach.

As a final baseline, we compare our proposed framework against the fine-tuning strategy similar to [22]. For a fair comparison, we only use the silver corpus scans ($\mathcal{D}_{\text{Train}}$) and 10 annotated slices from the support volume (`10000132_1_CTce_ThAb`) for

Table 5.6: Comparison of our proposed few-shot segmenter against the existing methods. For each method, per-fold and mean Dice score is reported for the test set. Left and Right are abbreviated as L. and R. Psoas Muscle is abbreviated as P.M. *Classifier Regression [9] training resulted in mode-collapse, hence no Dice score is reported. Feature Fusion is abbreviated to F.F. and Classifier Regression to C.R.

	Dice Score on Test set				
Method	Liver	Spleen	L/R kidney	L/R P.M.	Mean
Proposed	0.680	0.597	0.441	0.500	**0.554**
C.R.* [9]	–	–	–	–	–
F.F. [10]	0.434	0.354	0.357	0.307	0.363
F.F. [10] + C.R. [9]	0.484	0.320	0.443	0.450	0.424
Fine-Tuning [22]	0.300	0.012	0.004	0.012	0.080

training. As an architectural choice, we use our segmenter arm without the SE blocks. We pre-train the model using $\mathcal{D}_{\text{Train}}$ to segment the classes of $\mathcal{L}_{\text{Train}}$. After pre-training, we use the learnt weights of this model for initialization of all the layers, except for the classifier block. Then we fine-tune it using the 10 annotated slices of the support volume having a new class from $\mathcal{L}_{\text{Test}}$. We present the segmentation performance in Table 4.1. Fine-tuning was carefully performed with a low learning rate of 10^{-3} for 10 epochs. Also, the 10 selected slices were augmented during the training process. Still, except fold-1 (liver, Dice score 0.30) all the other folds had a Dice score lesser than 0.01. Overall, this experiment substantiated the fact that fine-tuning under such a low-data regime is ineffective, whereas our few-shot segmemtation technique is much more effective.

5.7 Qualitative Results

We present a set of qualitative segmentation results in Fig. 5.2(a-d) for folds 1-4, respectively. In Fig. 5.2(a), we show the segmentation of liver. From left to right, we present the support set with manual annotation, query input with its manual annotation, and prediction of the query input. We observe an acceptable segmentation despite the differences in the shape and size of the liver in the support and the query slices. Note that the only information the network has about the organ is from a single support slice. In Fig. 5.2(b), we show a similar result for spleen. This is a challenging case where the shape of spleen is very different in the support and query slices. Also, there is a difference in image contrast between the support and query slices. There is a slight undersegmentation of the spleen, but considering the weak support the segmentation is surprisingly good. In Fig. 5.2(c), we present the results of left kidney. Here we also observe a huge difference in the size of kidney in support and query slices. The kidney appears as a small dot in the support, making it a very difficult case. In Fig. 5.2(d), we show the segmentation for right psoas muscle. In this case, the support and query slices are pretty similar to each other visually. The prediction from our framework shows a bit of over-inclusion in the psoas muscle boundary but a decent localization and shape. Overall, the qualitative results visually present the effectiveness of our framework both under simple and very challenging conditions.

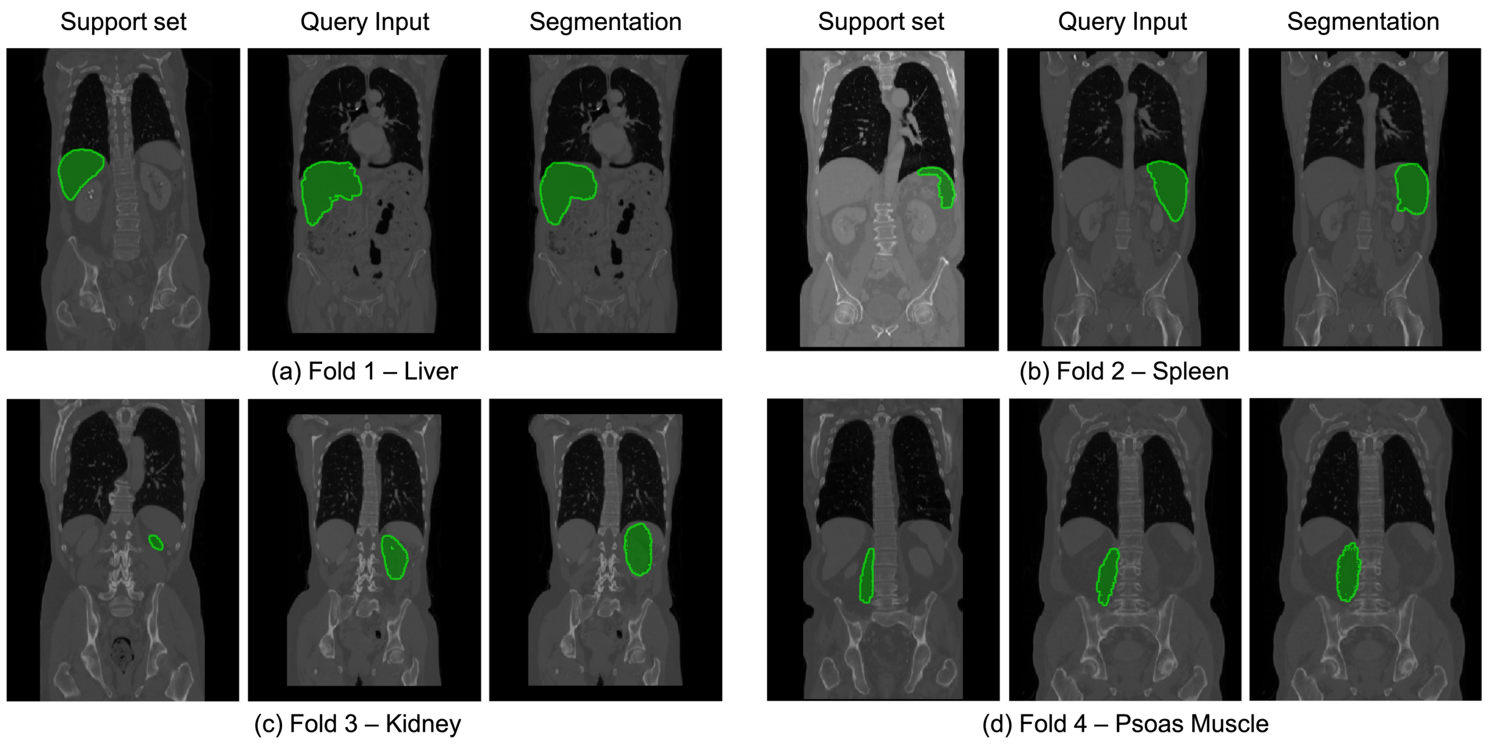


Figure 5.2: Qualitative results of our few-shot segmenter. The sub-figures (a-d) refer to examples from each of folds with liver, spleen, left kidney and right psoas muscles, respectively. For each of sub-figure, the first column indicates the support image with the manual outline of the organ, the second column indicates the query image with manual annotation, and the third column indicates the predicted segmentation for the query image. All the annotations are shown in green.

6 Conclusion

In this thesis, we introduced a few-shot segmentation framework for volumetric medical scans. The main challenges for extending the few-shot learning were the absence of pre-trained models to start from, and the volumetric nature of the scans. We proposed to use 'channel squeeze and spatial excitation' blocks for aiding proper training of our framework from scratch. Also, we proposed a volumetric segmentation strategy for segmenting a query volume scan with a support volume scan by strategic pairing of 2D slices. We conducted experiments on contrast-enhanced CT scans from the Visceral dataset for evaluating our framework. We compared our sSE based model to the existing approaches based on feature fusion [10], classifier regression [9] and their combination. Our framework outperformed all previous approaches by a large margin.

Besides comparing with the existing methods, we also provided detailed experiments for architectural choices regarding the SE blocks, model complexity, and skip connections. We also investigated the effect on the performance of our few-shot segmentation by changing the support volume and the number of budget slices from a support volume.

The exposition of our proposed approach is very generic and can easily be extended to other few-shot segmentation applications. Our approach is independent of pre-trained model, which makes it very useful for non computer vision applications.

List of Figures

List of Tables

Bibliography

[1] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio. “The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation”. In: *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*. IEEE. 2017, pp. 1175–1183.

[2] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. “Pyramid scene parsing network”. In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 2881–2890.

[3] 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.

[4] H. Noh, S. Hong, and B. Han. “Learning deconvolution network for semantic segmentation”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1520–1528.

[5] O. Ronneberger, P. Fischer, and T. Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.

[6] F. Milletari, N. Navab, and S.-A. Ahmadi. “V-net: Fully convolutional neural networks for volumetric medical image segmentation”. In: *3D Vision (3DV), 2016 Fourth International Conference on*. IEEE. 2016, pp. 565–571.

[7] A. G. Roy, S. Conjeti, D. Sheet, A. Katouzian, N. Navab, and C. Wachinger. “Error corrective boosting for learning fully convolutional networks with limited data”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2017, pp. 231–239.

[8] A. G. Roy, S. Conjeti, N. Navab, and C. Wachinger. “QuickNAT: A fully convolutional network for quick and accurate segmentation of neuroanatomy”. In: *NeuroImage* 186 (2019), pp. 713–727.

[9] A. Shaban, S. Bansal, Z. Liu, I. Essa, and B. Boots. “One-shot learning for semantic segmentation”. In: *arXiv preprint arXiv:1709.03410* (2017).

[10] K. Rakelly, E. Shelhamer, T. Darrell, A. A. Efros, and S. Levine. “Few-Shot Segmentation Propagation with Guided Networks”. In: *arXiv preprint arXiv:1806.07373* (2018).

[11] L. Fei-Fei, R. Fergus, and P. Perona. "One-shot learning of object categories". In: *IEEE transactions on pattern analysis and machine intelligence* 28.4 (2006), pp. 594–611.

[12] E. G. Miller, N. E. Matsakis, and P. A. Viola. "Learning from one example through shared densities on transforms". In: *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*. Vol. 1. IEEE. 2000, pp. 464–471.

[13] L. Fei-Fei. "Knowledge transfer in learning to recognize visual objects classes". In: *Proceedings of the International Conference on Development and Learning (ICDL)*. 2006, p. 11.

[14] A. G. Roy, N. Navab, and C. Wachinger. "Recalibrating Fully Convolutional Networks with Spatial and Channel 'Squeeze & Excitation' Blocks". In: *IEEE Transactions on Medical Imaging* (2018).

[15] A. G. Roy, N. Navab, and C. Wachinger. "Concurrent Spatial and Channel Squeeze & Excitation in Fully Convolutional Networks". In: *arXiv preprint arXiv:1803.02579* (2018).

[16] E. Bart and S. Ullman. "Cross-generalization: Learning novel classes from a single example by feature replacement". In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 1. IEEE. 2005, pp. 672–679.

[17] B. Hariharan and R. Girshick. "Low-shot visual recognition by shrinking and hallucinating features". In: *Proc. of IEEE Int. Conf. on Computer Vision (ICCV), Venice, Italy*. 2017.

[18] L. Bertinetto, J. F. Henriques, J. Valmadre, P. Torr, and A. Vedaldi. "Learning feed-forward one-shot learners". In: *Advances in Neural Information Processing Systems*. 2016, pp. 523–531.

[19] Y.-X. Wang and M. Hebert. "Learning to learn: Model regression networks for easy small sample learning". In: *European Conference on Computer Vision*. Springer. 2016, pp. 616–634.

[20] G. Koch, R. Zemel, and R. Salakhutdinov. "Siamese neural networks for one-shot image recognition". In: *ICML Deep Learning Workshop*. Vol. 2. 2015.

[21] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al. "Matching networks for one shot learning". In: *Advances in neural information processing systems*. 2016, pp. 3630–3638.

[22] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool. "One-shot video object segmentation". In: *CVPR 2017*. IEEE. 2017.

[23] N. Dong and E. P. Xing. "Few-shot semantic segmentation with prototype learning". In: *BMVC*. Vol. 3. 2018, p. 4.

[24] 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.

[25] J. Hu, L. Shen, and G. Sun. “Squeeze-and-excitation networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 7132–7141.

[26] O. Jimenez-del-Toro, H. Müller, M. Krenn, K. Gruenberg, A. A. Taha, M. Winterstein, I. Eggel, A. Foncubierta-Rodríguez, O. Goksel, A. Jakab, et al. “Cloud-based evaluation of anatomical structure segmentation and landmark detection algorithms: VISCERAL anatomy benchmarks”. In: *IEEE transactions on medical imaging* 35.11 (2016), pp. 2459–2475.