Spatial Context-Aware Self-Attention Model For Multi-Organ Segmentation

Hao Tang, Xingwei Liu, Kun Han, Xiaohui Xie
University of California Irvine
{htang6, xingweil, khan7, xhx}@uci.edu

Xuming Chen, Huang Qian, Yong Liu
Shanghai Jiao Tong University
School of Medicine

Shanlin Sun, Narisu Bai
DeepVoxel Inc.

Abstract

Multi-organ segmentation is one of the most successful applications of deep learning in medical image analysis. Deep convolutional neural nets (CNNs) have shown great promise in achieving clinically applicable image segmentation performance on CT or MRI images. State-of-the-art CNN segmentation models apply either 2D or 3D convolutions on input images, with pros and cons associated with each method: 2D convolution is fast, less memory-intensive but inadequate for extracting 3D contextual information from volumetric images, while the opposite is true for 3D convolution. To fit a 3D CNN model on CT or MRI images on commodity GPUs, one usually has to either downsample input images or use cropped local regions as inputs, which limits the utility of 3D models for multi-organ segmentation. In this work, we propose a new framework for combining 3D and 2D models, in which the segmentation is realized through high-resolution 2D convolutions, but guided by spatial contextual information extracted from a low-resolution 3D model. We implement a self-attention mechanism to control which 3D features should be used to guide 2D segmentation. Our model is light on memory usage but fully equipped to take 3D contextual information into account. Experiments on multiple organ segmentation datasets demonstrate that by taking advantage of both 2D and 3D models, our method is consistently outperforms existing 2D and 3D models in organ segmentation accuracy, while being able to directly take raw whole-volume image data as inputs.

1. Introduction

Segmentation of organs or lesions from CT images has great clinical implications. It can be used in multiple clinical workflows, including diagnostic interventions, treatment planning and treatment delivery [10]. Organ segmentation is an importance procedure for computer-assisted diagnostic and biomarker measurement systems [38]. Organ-at-risk (OAR) segmentation and tumor segmentation are also crucial to the planning of radiation therapy [32]. Moreover, the segmentation-based models of anatomical structures can support surgical planning and delivery [13].

Organ segmentation is typically done manually by experienced doctors. However, manually segmenting CT image by doctors is often time consuming, tedious and prone to human error, as a typical CT scan can contain up to hundreds of 2D slices. Computational tools that automatically segment organs from CT images can greatly alleviate the doctors’ manual effort, given a certain amount of accuracy is achieved.

There is a vast volume of work on organ segmentation using CT or magnetic resonance (MR) image. Traditional segmentation methods are mostly atlas-based. These methods rely on a set of accurate image templates with manual segmentation, and then use image registration to align the new image to the templates. Because of the reliance on the pre-computed templates, these methods may not adequately account for the anatomical variance due to variations in organ shapes, removal of tissues, growth of tumor and differences in image acquisition [47]. Also, registration is computationally intensive and may take up to hours to complete [4, 7, 12, 16, 24, 31, 35, 36, 43, 41, 14, 9, 3, 40, 42, 6, 50].

Deep learning-based methods provide an alternative solutions with substantial accuracy improvement and speed-up. With recent advances in deep learning especially deep convolutional neural network, automatic segmentation using computer algorithm has shown great promise in achieving near human performance [1, 52, 17, 33], and various applications have been deployed in clinical practice.

Fully convolutional network [19] and U-Net [28] are two of the most widely used deep learning-based segmentation algorithms for this purpose. Many its variants have been proposed in recent years, including V-Net [21] and Attention U-Net [23]. These methods can use either 2D or 3D convolutions as its basic building component. 2D methods
usually operate on a slice by slice basis, while 3D methods often operate on a 3D block or a stack of multiple 2D slices [18, 53]. The whole volume prediction can be obtained by predicting each slice or block using a sliding window. Additionally, some may stack multiple 2D slices in the input channel and use 2D convolution as a way to include some 3D features, and this is often referred as 2.5D model.

However, a CT image is inherently 3D. Cutting the images into slices or blocks often ignores the rich information and relation within the whole image volume. A big challenge in developing algorithm for consuming the whole image volume is the GPU memory limitation. Simply storing the tensors of the image features would require huge amount of GPU memory. One way is to adopt a coarse-to-fine strategy [57, 49, 45, 51, 44, 20, 48, 57, 49, 54, 2, 29], where in the first stage the organs of interest are roughly located, and in the second stage the segmentation masks are further refined by using a smaller input based on the localization. This usually requires training multiple CNNs for different stages and organs. Recently, several methods have been proposed to use the whole CT image volume as input, and achieve state-of-the-art accuracy and inference speed [33, 56, 57, 34, 11]. Despite their successes, there exists several disadvantages. First, to reduce the GPU memory consumption, these methods usually directly downsample the input in the very first convolution layer, which may lead to loss of local features. Moreover, they require carefully tailored image input size in order to fit the whole-volume image. However, they will still face GPU memory limitation if the image resolution becomes higher, because the memory requirement grows quickly with the size of the image volume. This makes previous whole-volume algorithm less effective when adapted to new dataset. Second, some of them make strong assumption on the organs/region they segment, thus lacking the ability to generalize well to other parts of the CT image.

We seek to incorporate 3D whole volume information into 2D model in a scalable way. We hypothesize that the benefits of using 3D convolution on the whole-volume image may come from its capability of modeling the shapes and relationships of the 3D anatomical structures. However, to model such shapes and relationships, we do not have to use very high-resolution image. 3D convolution on downsampled image volume may suffice to extract such information and can save a lot of computation and GPU memory. We can use 2D convolution on the original image slice to compensate for the loss of resolution. To fuse both 3D context features and 2D features, we implement a new module called multi-slice feature aggregation based on self-attention [39], which treats the 2D feature map as query and 3D context map as key, and uses self-attention to aggregate the rich 3D context information.

In this paper, we propose a new deep learning framework named Spatial Context-Aware Self-Attention Model (SCAA). Our main contributions are: i) a new framework for combining 3D and 2D models that takes the whole-volume CT image as input; ii) a self-attention mechanism to filter and aggregate 3D context features from the whole volume image to guide 2D segmentation; iii) the proposed method can scale to larger input volume without concerning the GPU memory limitation that common 3D methods face. Experiment results on a head and neck (HaN) dataset of 9 organs and an abdomen dataset of 11 organs show the proposed model consistently outperforms state-of-the-art methods in terms of organ segmentation accuracy, while being able to take the whole-volume CT image as input.

2. Method

Figure 1 shows the details of the proposed method. The proposed model consists of four parts: a 3D context feature encoder \( f^{3D} \), a 2D feature encoder \( f^{Enc}_2 \), a multi-slice feature aggregation (MSFA) module \( f^{MSFA} \), and at last a 2D decoder \( f^{Dec} \). The input to the model \( f : x \rightarrow f^{Dec}(f^{Enc}(f^{3D}(x))) \) is the whole CT image volume \( I \in \mathbb{R}^{D \times H \times W} \), and the outputs are \( D \) 2D segmentation masks for \( C \) classes \( m \in \mathbb{R}^{D \times H \times W \times C} \). \( D, H, W \) are the depth, height and width of the image volume.

2.1. 3D context feature encoder and 2D encoder

\( f^{3D} \) first downsamples the input 3D volume \( I \) to \( I' \in \mathbb{R}^{D_3 \times H_3 \times W_3} \). \( D_3, H_3, W_3 \) are the depth, height, and width of the downsampled 3D volume. We use a downsample factor of two in this work. Note we can also downsample the volume to other resolutions, e.g. isotropic 4mm resolution. It then applies 3D convolution blocks three times, where each convolution block consists of two residual blocks followed by one \( 2 \times 2 \times 2 \) max pooling, aiming at extracting 3D context features in the whole CT image.

The output of \( f^{3D} \) are four feature maps at different scales, denoted as \( F_i^{3D} \in \mathbb{R}^{C_i^{3D} \times D_i^{3D} \times H_i^{3D} \times W_i^{3D}} \), where \( i = 2, 3, 4, 5 \). This means the feature map \( F_i^{3D} \) is downsampled by a factor of 2\(^i\) compared to the original image. \( D_i^{3D}, H_i^{3D} \) and \( W_i^{3D} \) are the depth, height and width of the feature map at scale \( i \), the values of which depend on the size of input image. \( C_i^{3D} \) equals 24, 32, 64 and 64 for \( i = 2, 3, 4, 5 \) respectively. After \( F_5^{3D} \), we flatten the channel, depth, height and width dimension into a vector and regard it as a global descriptor \( F^{globe} \) for the 3D volume.

\( f^{Enc}_2 \) is similar to U-Net encoder. It consumes one axial slice of the CT image \( S \in \mathbb{R}^{H \times W} \) and applies 2D convolution blocks five times, where each block consists of two convolutions followed by instance normalization and ReLU activation, and a max pooling at the end. The 2D feature encoder outputs five sets of feature maps at different scales, denoted as \( F_i^{2D} \in \mathbb{R}^{C_i^{2D} \times D_i^{2D} \times H_i^{2D} \times W_i^{2D}} \), where
Figure 1: Overview of spatial context-aware self-attention model (SCAA). SCAA consists of a 3D context feature encoder, a 2D encoder, a 2D decoder and a multi-slice feature aggregation (MSFA) module. SCAA starts with extracting 3D features from the downsampled CT image using 3D convolutions. Then the 2D encoder extracts 2D features and uses MSFA module to fuse 2D and 3D features hierarchically. Lastly, the 2D decoder decodes the fused 2D and 3D features and outputs 2D segmentation masks of each organ. The numbers of feature channels in $F_i$ are 64, 96, 128, 192, 256 for $i = 1, 2, 3, 4, 5$ respectively. The number of feature channels in $F_{2D,i}$ is 96, 128, 192, 256 for $i = 2, 3, 4, 5$ respectively. The numbers of feature channels in $F_{3D,i}$ are 24, 32, 64, 64 for $i = 2, 3, 4, 5$ respectively. $W_{q,i}$ is implemented by using $1 \times 1 \times 1$ 3D convolution with 2, 2, 4, and 4 feature channels for $i = 2, 3, 4, 5$ respectively. $W_{k,i}$ is implemented by using $1 \times 1$ 2D convolution with 2, 2, 4, and 4 feature channels for $i = 2, 3, 4, 5$ respectively. The $xy$ spatial resolution of the output of the adaptive pooling are $16 \times 16, 8 \times 8, 4 \times 4, 4 \times 4$ for scale $i = 2, 3, 4, 5$ respectively. The number of attention heads for scale $i = 2, 3, 4, 5$ is 2, 2, 4, and 4 respectively.

2.2. Multi-scale feature aggregation

Inspired by Transformer [39, 25], we implement a self-attention mechanism to filter and extract useful 3D context features from our 3D feature maps $F_{3D,i}$, and we name this module as multi-scale feature aggregation (MSFA). We regard 3D features as values, and generate queries from the 2D features and keys from the 3D features. Based on feature similarities of current 2D features and all slices in the 3D feature map (along $z$ dimension), the MSFA will generate an attention vector $a_i \in \mathbb{R}^{3D}$ the same length of the depth of the 3D feature map. This attention then is applied to the 3D feature map to generate a 2D feature map that is considered as the aggregated 3D context features.

We start by mapping our 2D feature map $F_{2D,i}$ and 3D feature map $F_{3D,i}$ ($i = 2, 3, 4, 5$), to one query and $D_{3D}$ keys, of the same embedding space. We use a weight metric $W_{q,i} (1 \times 1 \times 1$ 3D convolution) to generate our query $q \in \mathbb{R}^{c_{emb} \times H_i \times W_i}$. We use a weight metric $W_{k,i} (1 \times 1 \times 1$ 3D convolution) to generate our keys $\{k_j\}$ of size $C_{emb} \times H_i^{3D} \times W_i^{3D}$, where $j = 1, 2, ..., D_{3D}$. An adaptive pooling operation is used to reduce the spatial resolution of the query and keys to $H_i' \times W_i'$, followed by a flatten operation to make them one dimensional. As a result, the embed dimension of the query and keys is now of size $i = 1, 2, 3, 4, 5$. $C_{3D}$ equals 64, 96, 128, 192 and 256 for $i = 2, 3, 4, 5$ respectively.
\( C_{embed}, H'_i, W'_i \):

\[
(F_{3D}^{i})_{c,chw} = (F_{3D})_{c,chw}(W^k_i)_{c,c} \\
(F_{3D}^{i})_{c,chw} = (F_{3D})_{c,chw}(W^k_i)_{c,c} \\
k_i = \text{Flatten}((\text{AdaptivePool}_{D_i} H'_i W'_i (F_{3D}^{i}))) \\
q_i = \text{Flatten}((\text{AdaptivePool}_{H'_i W'_i} (F_{3D}^{i})))
\]

(\cdot)_j is the Einstein summation convention.

A scaled dot product is used to compute the response of the 2D feature map with the 3D feature map \( r_j = \frac{y_j}{\sqrt{C_{embed} H'_i W'_i}} \). A softmax is followed to generate our attention \( a_i = \text{softmax}(r) \). We then multiply the attention \( a_i \) over the depth dimension of \( F_{3D}^{i} \) and sum over the depth dimension to generate our aggregated context feature map \( F_{i}^{agg} \in \mathbb{R}^{C_{3D} \times H_i \times W_i} \):

\[
(F_{i}^{agg})_{chw} = (F_{3D})_{c,chw}(a_i)_{d}
\]

A multi-head attention mechanism is also used. We generate \( m_i \) such fused 2D feature map and then use a weight metric \( W^m_i (1 \times 1 \text{ 2D convolution}) \) to aggregate multiple self-attention output. \( m_i \) is 2, 2, 4 and 4 for \( i = 2, 3, 4, 5 \) respectively. This multi-head attention allows our model to focus on different parts of the 3D context volume to extract features required by different classes. Two 2D convolution blocks on the concatenated 2D feature map of \( F_{2D} \) and \( F_{3D}^{agg} \) are used to better combine the 2D and 3D context features. \( F_i \) denotes our final 2D feature map for scale \( i \). Note that \( F_1 \) is the same as \( F_{2D} \).

2.3. 2D decoder

\( f^{Dec} \) is similar to the U-Net decoder. Starting from \( F_5 \), a 2D upsample is used first to increase the spatial resolution by 2. Then we concatenate the upsampled features with the corresponding encoder feature map and apply one 2D convolution block. The last upsampled feature map is of the same resolution as our input image. We concatenate our 3D global descriptor \( F_{global} \) to each pixel’s feature vector and use a 1 \times 1 convolution to obtain the final axial segmentation mask for each class \( \{m^{2D}_c\} \), where \( c \in \mathbb{Z}^+_{\leq C} \) and \( C \) is the number of classes.

2.4. Loss function and implementation details

The loss function is defined as:

\[
L^{2D} = \sum_{c}^{N} 1 - \phi(m^c, g^c)
\]

\( g \) is the ground truth segmentation for the axial slice. \( \phi(m, g) \) computes a soft Dice score between the predicted mask \( m \) and the ground truth \( g \):

\[
\phi(m, g) = \frac{\sum_{i} m_i g_i + \alpha \sum_{i} m_i (1 - g_i) + \beta \sum_{i} (1 - m_i) g_i + \epsilon}{N + \alpha N + \beta N}
\]

\( N \) is the number of total pixels in the batch. \( \alpha \) and \( \beta \) are two hyper parameters controlling the penalty for false positive and true negative respectively, and we set them to both 0.5. \( \epsilon \) is used for numerical stability.

To facilitate the training of 3D context feature encoder, we add an auxiliary 3D segmentation loss. We first upsample \( F_{3D}^{i} \) by a factor of 16, so it has the same spatial resolution as the downsampled 3D image volume. A 1 \times 1 \times 1 3D convolution is used to obtain the 3D segmentation mask \( \{m^{3D}_c\} \). We use the same dice loss to get our 3D supervision loss \( L^{3D} \). The final loss is then \( L = L^{3D} + L^{2D} \).

We use one CT image for each batch during training. For each batch, we generate one 3D image volume and randomly sample 16 axial slices (batch size 16 for the 2D network). We only need to forward the 3D context encoder once per batch. We use Adam with initial learning rate \( 10^{-4} \) as optimizer for a total of 150 epochs. We applied elastic transformation and random jitter for data augmentation.

For testing, we only need to forward the 3D context feature encoder once, and we segment each 2D slice using the 2D decoder/encoder and MSFA.

3. Experiments

3.1. Datasets

Two datasets were used for evaluation: i) MICCAI 2015 head and neck (HaN)organ-at-risk (OAR) segmentation challenge dataset [26], containing a training of 33 CT images and a test of 15 CT images. The dataset contains manually labeled organ segmentation mask for 9 organs: brain stem, mandible, optic nerve left and right, optic chiasm, parotid left and right, submandibular gland (SMG) left and right; ii) an in-house abdomen multi-organ segmentation dataset\(^1\) (ABD-110) containing 110 contrast enhanced CT images and 11 organs (large bowel, duodenum, spinal cord, liver, spleen, small bowel, pancreas, left and right kidney, stomach and gallbladder). The 110 CT scans were collected from 110 patients who had radiotherapy during the past three years. The CT scans were manually delineated by one experienced doctor and then manifested by another. We use the official split of training set to train the model and test on the official test set on MICCAI 2015 challenge dataset, following the same protocol as previous work [33, 22, 26, 11]. All experiments on ABD-110 dataset was conducted using 4-fold cross validation.

We report the segmentation performance using dice similarity coefficient (DSC) in percentage and 95% hausdorff

\(^1\)Use of this dataset has been approved by an institutional review board (IRB).
measures the overlap between the predicted mask \( m \) and ground truth mask \( g \):

\[
DSC = \frac{2|m \cup g|}{|m \cap g|} \quad (5)
\]

3.2. Ablation study on ABD-110

To compare different ways of integrating 3D features and demonstrate the contribution of each of the add-on modules in the proposed model, we conducted ablation studies with the following different settings: 1) model with only a 3D global descriptor \( F_g \) concatenated to the last feature map (CA), to demonstrate the importance of including 3D features progressively during feature extraction. 2) model without MSFA module, but only uses the corresponding center slice feature from the 3D context feature maps (C-CA), to show the effectiveness and importance of using self-attention to aggregate 3D features. 3) SCAA model without concatenating \( F_g \) to the last feature map (SCAA*), to show whether \( F_g \) is crucial to improve the segmentation accuracy.

As we can see from Table 1, by adding the 3D global descriptor to the 2D U-Net, CA outperforms the 2D U-Net by 0.4% and lowers the 95%HD by 0.7 mm, showing the importance of integrating the 3D holistic information. Next, by progressively integrating 3D features to 2D feature extractor, C-CA outperforms CA by 1.4% in DSC and 0.2 mm in 95%HD, showing the importance of integrating 3D context features hierarchically. Finally, by adding the MSFA module based on self-attention, SCAA outperforms C-CA by 0.4%, demonstrating the effectiveness of the implemented self-attention mechanism to weigh the information from different adjacent slices. To find out which part of the 3D features contribute the most, we then compare the performance of SCAA and SCAA*. SCAA* achieves average DSC of 86.9% and average 95%HD 3.6 mm, while SCAA achieves 86.8% and 4.3 mm. They are very close in terms of DSC and SCAA* wins on 95%HD. This demonstrates that \( F_g \) has very little contribution, and progressively integrating 3D features with 2D encoder achieves the most performance gain.

For all following comparisons with other methods, we use the best performing configuration SCAA*.

3.3. Comparison with previous methods on ABD-110

To compare with previous methods of multi-organ segmentation on the ABD-110 dataset, we ran the following representative algorithms: U-Net [28], Attention U-Net [23], U*-Net [33], and nnU-Net [15]. U-Net is a well-established medical image segmentation baseline algorithm. Attention U-Net is a multi-organ segmentation framework that uses gated attention to filter out irrelevant response in the feature maps. U*-Net is a state-of-the-art end-to-end two-stage framework for multi-organ segmentation in the head and neck region. nnU-Net is a self-adaptive medical image semantic segmentation framework that wins the first in the Medical Segmentation Decathlon (MSD) challenge [30]. nnU-Net mainly consists of three main deep learning-based segmentation methods: a 2D U-Net (slice-wise), a 3D U-Net (patch-wise) and a coarse-to-fine cascade framework consisting of two 3D U-Nets. Its final model is an ensemble of the three methods. The above-mentioned works cover a wide range of algorithms for multi-organ segmentation and should provide a comprehensive and fair comparison to our proposed method on the in-house dataset. For 3D Attention U-Net, we followed the same preprocessing as in its original paper, to downsample the image to isotropic 2mm resolution due to GPU memory limitation. However, for all other methods, we feed the original CT image with its original image spacing.

The results are shown in section 3.3. First, by comparing 2D and 3D methods, we can see that the performance of 2D methods is on par with 3D methods on kidneys, spinal cord and liver, which is likely because those organs are usually large and have regular shapes. However, for organs like stomach, small and large bowels, 3D methods generally perform better. This may be because those organs often have more anatomical variance, and a 3D holistic understanding of the context is beneficial. Next, U*-Net was 1.4% lower than 3D U-Net and 3.5% lower than SCAA. This may be because U*-Net was designed mainly for the head and neck region where organs are relatively small and do not overlap too much with each other. The abdomen region, on the other hand, is more complicated as the bounding boxes of some organs overlap a lot with each other (e.g. large bowel and small bowel), which makes U*-Net less effective. Finally, comparing SCAA to nnU-Net, we find SCAA outperform nnU-Net by 1.2%. The best configuration of nnU-Net on ABD-110 was ensemble of a 2D U-Net (slice by slice) and a 3D U-Net (patch-wise). Both nnU-Net and SCAA consider the fusion of 2D model and 3D model, but they implement it in different ways - nnU-Net uses ensemble to combine 2D and 3D models while SCAA integrates 3D model into 2D model in an end-to-end fashion and jointly optimizes both models. This improvement then is likely due to the soft attention mechanism that allows SCAA to filter and extract relevant features from the large 3D context and better fuse the 2D and 3D models. Altogether, we demonstrated the effectiveness of the proposed method, which achieves an average DSC of 86.9% on the in-house dataset.

3.4. Performance on MICCAI2015

A second multi-organ segmentation dataset from MICCAI 2015 organ-at-risk (OAR) segmentation challenge[26]
1.4%. U-net focuses on a smaller region. This keeps SCAA away from a model that takes the whole-volume CT as input, SCAA was used for evaluation. First, as we can see from Table 3, SCAA outperforms [22] by 4.2%. [22] used a combination of 3D and 2D convolution on 21 stacked slices for OAR segmentation. This shows the use of larger context information is beneficial for a good segmentation accuracy. Next, by comparing SCAA to AnatomyNet [55] which is a 3D model that takes the whole-volume CT as input, SCAA was 2.4% higher. This is likely due to the attention mechanism that helps the model to filter irrelevant features from the entire volume. Also, SCAA outperforms Uₐ-Net [33] by 1.4%. Uₐ-Net is an end-to-end two-stage model that first detects bounding box of OARs and then segments organs within the bounding box. SCAA performed better, partly because SCAA did not enforce a ‘hard’ attention (bounding box) but rather use ‘soft’ attention to enable the model focus on a smaller region. This keeps SCAA away from potential bounding box regression error and missing detection. Finally, SCAA outperforms previous state-of-the-art method [11] in 5 out of 9 organs and achieves an average DSC of 82.6%, 0.2% higher than the state-of-the-art method. Also note that [11] is a two-stage segmentation framework, which consists of two DCNNs. Our proposed method (SCAA), however, is a one-stage end-to-end solution for multi-organ segmentation, requiring less training time and computation, as well as fewer parameters.

### 3.5 Memory consumption

One advantage of the proposed method is that it significantly reduces the GPU memory while at the same time preserves the large 3D context features. To demonstrate GPU memory consumption when using whole-volume as input, we estimated and measured the actual GPU memory cost (using PyTorch as framework) for different 3D models during training in Table 4. We made several assumptions: i) the input image volume is of size $256 \times 256 \times 256$. This is the size used for whole-volume input with original image sp-

<table>
<thead>
<tr>
<th>Anatomy</th>
<th>U-Net</th>
<th>CA</th>
<th>C-CA</th>
<th>SCAA</th>
<th>SCAA*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large Bowel</td>
<td>80.5 ± 9.4</td>
<td>80.3 ± 9.3</td>
<td>80.2 ± 8.7</td>
<td>80.7 ± 9.7</td>
<td>77.1 ± 10.4</td>
</tr>
<tr>
<td>Duodenum</td>
<td>63.4 ± 18.6</td>
<td>64.5 ± 18.1</td>
<td>67.1 ± 16.0</td>
<td>70.2 ± 16.3</td>
<td>62.6 ± 17.7</td>
</tr>
<tr>
<td>Spinal Cord</td>
<td>90.3 ± 3.8</td>
<td>90.9 ± 4.0</td>
<td>89.8 ± 3.9</td>
<td>88.4 ± 4.6</td>
<td>91.6 ± 3.5</td>
</tr>
<tr>
<td>Liver</td>
<td>95.5 ± 1.9</td>
<td>95.7 ± 1.5</td>
<td>96.0 ± 1.1</td>
<td>95.9 ± 1.2</td>
<td>94.7 ± 1.9</td>
</tr>
<tr>
<td>Spleen</td>
<td>94.6 ± 3.1</td>
<td>95.1 ± 2.8</td>
<td>95.0 ± 1.9</td>
<td>92.0 ± 13.2</td>
<td>94.7 ± 2.1</td>
</tr>
<tr>
<td>Small Bowel</td>
<td>72.2 ± 16.2</td>
<td>73.7 ± 16.6</td>
<td>75.0 ± 13.8</td>
<td>73.7 ± 15.6</td>
<td>75.0 ± 13.8</td>
</tr>
<tr>
<td>Pancreas</td>
<td>79.8 ± 9.1</td>
<td>80.4 ± 9.8</td>
<td>79.0 ± 10.0</td>
<td>80.2 ± 12.1</td>
<td>76.3 ± 11.6</td>
</tr>
<tr>
<td>Kidney L</td>
<td>95.7 ± 1.2</td>
<td>95.6 ± 1.6</td>
<td>94.5 ± 9.0</td>
<td>94.2 ± 9.0</td>
<td>95.2 ± 1.5</td>
</tr>
<tr>
<td>Kidney R</td>
<td>95.3 ± 3.0</td>
<td>95.5 ± 2.9</td>
<td>94.9 ± 3.1</td>
<td>94.2 ± 9.2</td>
<td>94.8 ± 2.5</td>
</tr>
<tr>
<td>Stomach</td>
<td>84.2 ± 16.7</td>
<td>84.0 ± 15.2</td>
<td>85.4 ± 15.6</td>
<td>87.3 ± 14.5</td>
<td>83.1 ± 14.1</td>
</tr>
<tr>
<td>Gallbladder</td>
<td>78.6 ± 19.5</td>
<td>77.1 ± 20.5</td>
<td>78.2 ± 19.1</td>
<td>74.2 ± 29.6</td>
<td>72.6 ± 26.8</td>
</tr>
</tbody>
</table>

| Average | 84.6 | 85.0 | 86.4 | 86.8 | 86.9 |

1 2D U-Net/Attention U-Net with adjacent 3 slices stacked into channel as input (2.5D).
2 3D U-Net with patch-wise input.

Table 2: Dice similarity coefficient (DSC) comparison on ABD-110. Bold number means the best DSC performance. SCAA (proposed) achieves the highest DSC compared to other methods.
are the higher the better (best in bold).

Table 3: Comparison of DSC with previous methods on the MICCAI 2015 9 organs segmentation challenge. Numbers are the higher the better (best in bold).

<table>
<thead>
<tr>
<th>Study</th>
<th>Brain Stem</th>
<th>Mandible</th>
<th>Optic Chiasm</th>
<th>Optic Nerve</th>
<th>Parotid</th>
<th>SMG</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raudashl et al. [26]</td>
<td>88.0</td>
<td>93.0</td>
<td>55.0</td>
<td>62.0</td>
<td>62.0</td>
<td>84.0</td>
<td>84.0</td>
</tr>
<tr>
<td>Fritscher et al. [8]</td>
<td>49.0 ± 9.0</td>
<td></td>
<td></td>
<td>81.0 ± 4.0</td>
<td>81.0 ± 4.0</td>
<td>65.0 ± 8.0</td>
<td>65.0 ± 8.0</td>
</tr>
<tr>
<td>Ren et al. [27]</td>
<td>58.0 ± 17.0</td>
<td>72.0 ± 8.0</td>
<td>70.0 ± 9.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wang et al. [46]</td>
<td>90.0 ± 4.0</td>
<td>94.0 ± 1.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zhu et al. [55]</td>
<td>86.7 ± 2.0</td>
<td>92.5 ± 2.0</td>
<td>53.2 ± 15.0</td>
<td>72.1 ± 6.0</td>
<td>70.6 ± 10.0</td>
<td>88.1 ± 2.0</td>
<td>87.3 ± 4.0</td>
</tr>
<tr>
<td>Tong et al. [37]</td>
<td>87.0 ± 3.0</td>
<td>93.7 ± 1.2</td>
<td>58.4 ± 10.3</td>
<td>65.3 ± 5.8</td>
<td>68.9 ± 4.7</td>
<td>83.5 ± 2.3</td>
<td>83.2 ± 1.4</td>
</tr>
<tr>
<td>Nikolov et al. [22]</td>
<td>79.5 ± 7.8</td>
<td>94.0 ± 2.0</td>
<td></td>
<td>71.6 ± 5.8</td>
<td>69.7 ± 7.1</td>
<td>86.7 ± 2.8</td>
<td>85.3 ± 6.2</td>
</tr>
<tr>
<td>Tang et al. [33]</td>
<td>87.5 ± 2.5</td>
<td>95.0 ± 0.8</td>
<td>61.5 ± 10.2</td>
<td>74.8 ± 7.1</td>
<td>72.3 ± 5.9</td>
<td>88.7 ± 1.9</td>
<td>87.5 ± 5.0</td>
</tr>
<tr>
<td>Guo et al. [11]</td>
<td>87.6 ± 2.8</td>
<td>95.1 ± 1.1</td>
<td>64.5 ± 8.8</td>
<td>75.3 ± 7.1</td>
<td>74.6 ± 5.2</td>
<td>88.2 ± 3.2</td>
<td>88.2 ± 5.2</td>
</tr>
<tr>
<td>SCAA (proposed)</td>
<td>89.2 ± 2.6</td>
<td>95.2 ± 1.3</td>
<td>62.0 ± 16.9</td>
<td>78.4 ± 6.1</td>
<td>76.0 ± 7.5</td>
<td>89.3 ± 1.5</td>
<td>89.2 ± 2.3</td>
</tr>
</tbody>
</table>

Table 4: GPU memory consumption comparison using whole-volume image as input of, and number of parameters for different methods on ABD-110. We used PyTorch as the deep learning framework to measure the actual GPU memory cost.

<table>
<thead>
<tr>
<th>Method</th>
<th>Batch size</th>
<th>Estimate (GB)</th>
<th>Actual (GB)</th>
<th># of parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D U-Net [28]</td>
<td>4</td>
<td>2.86</td>
<td>3.35</td>
<td>34.51 M</td>
</tr>
<tr>
<td>3D U-Net [23]</td>
<td>1</td>
<td>27.96</td>
<td>out of memory</td>
<td>5.88 M</td>
</tr>
<tr>
<td>3D Attention U-Net [23]</td>
<td>1</td>
<td>17.31</td>
<td>out of memory</td>
<td>6.40 M</td>
</tr>
<tr>
<td>SCAA (3D encoder)</td>
<td>1</td>
<td>3.22</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SCAA (2D U-Net &amp; MSFA)</td>
<td>4</td>
<td>2.13</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SCAA (total)</td>
<td>5.35</td>
<td>6.44</td>
<td>7.82 M</td>
<td></td>
</tr>
</tbody>
</table>

We visualize the attention vector $a_i$ learnt for $F_i^{3D}$ Figure 2 and the prediction of a random CT image from the

![Figure 2: Attention vector learnt by the proposed method. Y-axis is the slice number of the CT image, and the X-axis is the slice number (depth) of the 3D feature map $F_i^{2D}$.](image)

ABD-110 dataset Figure 3. As seen from Figure 2, the 3D slice features that are useful when segmenting each 2D slice are mostly its adjacent slices. This accords with the intuition that the most prominent and useful 3D information should be mostly from its neighbouring slices. But it is also important to incorporate full 3D context information progressively. We have demonstrated the effectiveness of the self-attention mechanism in Section 3.2 by comparing to C-CA that only integrates the corresponding center slice.
4. Conclusion

In this paper, we propose a Spatial Context-aware Self-Attention model for multi-organ segmentation. The proposed model uses a self-attention mechanism to filter useful 3D contextual information from the large 3D whole-volume CT image to guide the segmentation of 2D slice. It addresses the GPU memory concerns that common whole volume-based 3D methods confront. Experiments on two multi-organ segmentation datasets demonstrate the state-of-the-art performance of the proposed model.
References


[22] Stanislav Nikolov, Sam Blackwell, Ruheena Mendes, Jeffrey De Fauw, Clemens Meyer, Cian Hughes, Harry Askham,


