Thickened 2D Networks for Efficient 3D Medical Image Segmentation

Qihang Yu\textsuperscript{1} Yingda Xia\textsuperscript{1} Lingxi Xie\textsuperscript{2} Elliot K. Fishman\textsuperscript{3} Alan L. Yuille\textsuperscript{1}

\textsuperscript{1} The Johns Hopkins University \quad \textsuperscript{2} Noah’s Ark Lab, Huawei Inc. \quad \textsuperscript{3} The Johns Hopkins Medical Institute

\{yucornetto, 198808xc, alan.l.yuille\}@gmail.com yxia25@jhu.edu efishman@jhmi.edu

Abstract

There has been a debate in 3D medical image segmentation on whether to use 2D or 3D networks, where both pipelines have advantages and disadvantages. 2D methods enjoy a low inference time and greater transfer-ability while 3D methods are superior in performance for hard targets requiring contextual information. This paper investigates efficient 3D segmentation from another perspective, which uses 2D networks to mimic 3D segmentation. To compensate the lack of contextual information in 2D manner, we propose to thicken the 2D network inputs by feeding multiple slices as multiple channels into 2D networks and thus 3D contextual information is incorporated. We also put forward to use early-stage multiplexing and slice sensitive attention to solve the confusion problem of information loss which occurs when 2D networks face thickened inputs. With this design, we achieve a higher performance while maintaining a lower inference latency on a few abdominal organs from CT scans, in particular when the organ has a peculiar 3D shape and thus strongly requires contextual information, demonstrating our method’s effectiveness and ability in capturing 3D information. We also point out that “thickened” 2D inputs pave a new method of 3D segmentation, and look forward to more efforts in this direction. Experiments on segmenting a few abdominal targets in particular blood vessels which require strong 3D contexts demonstrate the advantages of our approach.

1. Introduction

Medical image segmentation is an important prerequisite of computer-assisted diagnosis (CAD) which implies a wide range of clinical applications. In this paper, we focus on organ segmentation with 3D medical images, such as Computed Tomography (CT) and Magnetic Resonance Imaging (MRI). To deal with volumetric input, two main flowcharts exist. The first method borrows the idea from natural image segmentation, cutting the 3D volume into 2D slices, and training a 2D network which deals with each slice individually or sequentially. Another way is to crop patches from the volume, and train a 3D network to deal with volumetric patches directly. Afterwards, both methods test the original volume in a sliding window manner.

Both of these two methods have their own advantages and disadvantages. A 2D network, since it takes a full-size slice as input and thus only needs to slide along single axis, has a much lighter computation and inference speed, but suffers from the lack of information of relationship among slices (see Figure 1). For mainstream patch-based 3D approach, in opposite, has the perception of 3D contexts, but also suffers from several weaknesses: (i) patch-based method has a limited receptive field, and makes the model easier to be confused; (ii) the lack of pretrained models makes the training process unstable; and (iii) in the testing stage, patch-based method requires sliding along all three axes, resulting in a high inference latency.

This paper presents a novel framework which uses a 2D network to mimic 3D segmentation by thickening 2D inputs. Therefore the model enjoys both the light computa-
Figure 2: Toy examples of different methods dealing with a 3D volume, including (a) 3D network (input patch size $1 \times H' \times W' \times D'$), (b) 2D network (input single slice $1 \times H \times W$), (c) thickened 2D network (input $C$-slice $C \times H \times W$). At training, 2D, 3D, T2D network is trained with single full-size slices, cropped patches, and thickened full-size slices respectively. At testing stage, 2D and T2D slide along single axis and stack the slice-prediction into a volume prediction, while 3D slides along all three axes in a sliding window manner. (Best viewed in color.)

2. Related Work

Computer aided diagnosis (CAD) is a research area aiming at helping human doctors in clinics. Recently a lot of CAD approaches are based on medical imaging analysis to get accurate descriptions of the scanned organs, soft tissues, etc. One topic with great importance in this area is object segmentation, i.e., determining which voxels belong to the target in 3D data. In natural image segmentation, conventional methods based on graph [1, 25] or handcrafted local features [32] have been gradually replaced by techniques from deep learning, which could produce higher segmentation accuracy [5, 20, 38]. And it has been proved successful not only in the natural image area but also medical im-

The essence lies in information loss. Technically, for a 2D network, information from all input slices are mixed together into channel dimension, and, without a specifically designed scheme, it is difficult to discriminate them from each other. For a 2D network with thickened input, information from different slices are fused in the first convolution. Since background noises are not filtered, mixing slices together makes it hard to pick up useful information for distinguishing each slice. We argue that this early fusion (information from all slices are fused in the first convolution) is what leads to the information loss and the worse performance with thickened inputs.

Therefore, we propose two solutions based on a 2D segmentation backbone to address this problem. (1) Early-Stage Multiplexing (ESM): we postpone the stage that these information are put together. The backbone is divided into two parts, and the thickened input is divided into several mini-groups. We multiplex first part of the backbone to deal with each mini-group individually, while in the second part the information from different slices are fused. (2) Slice-Sensitive Attention (SSA): to improve the discriminative power of the fused feature maps, we introduce slice-sensitive attention between the pre-fusion stage and the decision stage. The options of fusing these two sources of information are also studied in an empirical manner.

Experiments are conducted on several abdominal organs individually, including two regular organs and three blood vessels in our own dataset, and the hepatic vessels in the Medical Segmentation Decathlon (MSD) [29] dataset. Our approach achieves a better performance compared with popular 3D models in terms of dice score and also has a lower inference latency. We also prove that our method improves the ability to distinguish neighboring slices with a designed metric.

The remainder of this paper is organized as follows. Section 2 briefly reviews related work, and Section 3 presents our approach. After experiments are shown in Section 4, we conclude this work in Section 5.
age area [22, 26], outperforming conventional approaches, e.g., when segmenting the brain [3, 15], the liver [9, 18], the lung [13, 14], or the pancreas [7, 28].

These deep learning methods in medical image segmentation can be classified into the following types according to their way to deal with 3D volumetric data:

Researchers taking 2D method cut each 3D volume into 2D slices, and train a 2D network to process each of them individually [24, 26]. Such methods often suffer from missing 3D contextual information, for which various techniques are adopted, such as using 2.5D data (stacking a few 2D images as different input channels) [27, 28], training deep networks from different viewpoints and fusing multi-view information at the final stage [34, 35], and applying a recurrent network to process sequential data [2, 4].

A 3D workflow, researchers directly train a 3D network to deal with the volumetric data [8, 22]. These approaches, while being able to see more contextual information, often require much larger memory consumption. So some existing methods work on small patches and fuse the outputs of all patches [11, 40], or down-sample the whole volume into a lower resolution [10, 22]. In addition, unlike 2D networks that can borrow pre-trained models from natural image datasets, 3D networks need to be trained from scratch, which means it could suffer from unstable convergence properties [30]. A discussion on 2D vs. 3D models for medical imaging segmentation is available in [17].

Some works also explore the way to find a way to combine 2D and 3D methods. [35] proposes to use a 3D network to fuse 2D predictions from different viewpoints. [23] tries to use a 2D network which predicts the distance from voxel to organ boundary, and afterwards, 3D reconstruction is applied to generate a final dense prediction. In [19], a hybrid segmentation network consisting of 2D encoder and 3D decoder is designed, thus the network benefits from 2D pre-trained weight and 3D contextual information.

In this paper, we provide an alternative way to incorporate 3D contextual information into 2D networks. Our method differs from what mentioned above by (1) the third dimension is incorporated into channel dimension, thus no 3D operation or module is introduced and (2) the model is still in 2D manner thus enjoy a light computation cost at inference.

3. Our Approach

3.1. Problem Statement and 2D/3D Baselines

3.1.1 Problem Statement

In this task, a CT scan is regarded as a 3D volume $X$ of size $H \times W \times D$, and each voxel $X_{h,w,d}$ indicates the intensity at the specified position measured by the Haunsfield unit (HU). It is annotated with a binary ground-truth segmentation $Y$ where $y_i = 1$ means a foreground voxel. Suppose our model predicts a volume $Z$, and $Y = \{ (h, w, d) \mid Y_{h,w,d} = 1 \}$ and $Z = \{ (h, w, d) \mid Z_{h,w,d} = 1 \}$ are the foreground voxels in the ground-truth and prediction, respectively. The segmentation accuracy is evaluated using the Dice-Sørensen coefficient (DSC): $DSC (Y, Z) = \frac{2|Y \cap Z|}{|Y| + |Z|}$, which has a range of $[0, 1]$ with 1 implying a perfect prediction.

3.1.2 2D and 3D Baselines

2D and 3D models are most common methods in nowadays medical image segmentation. 2D segmentation networks deal with 3D volume by cutting them into stacked 2D slices firstly. Usually, the slicing procedure is based on three axes, the coronal, sagittal, and axial. On each axis, an individual 2D deep network is trained. Final prediction is obtained by stacking 2D slices predictions, and the contextual information is added in a post-processing manner through fusing three-views results. In contrary, 3D segmentation networks deal with 3D volume directly. Due to limitation of GPU memory, the mainstream 3D methods usually crop the volume into smaller patches instead of taking the entire volume as input, especially when data is in high resolution. In the testing stage, a sliding window of the same size is moved regularly along three axes, and prediction at each voxel is averaged over all predictions it gets.

A problem for 2D segmentation networks is the lack of 3D contextual information. Although the final three-axes fusion makes compensation to some degree, it is still single-slice-based when training and testing. It is enough for some easy and large-scale organ like kidney, liver, and spleen, yet it works badly when facing more challenging tasks. 3D segmentation networks also suffer from several deficits: (i) need to be trained from scratch, which could bring unstable convergence properties issues; (ii) sliding window manner is time-consuming and patch-based method limits the receptive field the network can see, thus it cannot get enough global information for an individual slice and is easy to get confused.

The drawbacks lie in 2D and 3D methods motivate us to propose thickened 2D network, where we feed multiple slices as a multiple channels into 2D network. By incorporating contextual information into 2D network, we expect the method to be more powerful than pure 2D methods and more robust and faster than pure 3D methods.

3.2. Thickened 2D Inputs

3.2.1 Re-visiting 2D Segmentation

Suppose the input is a 3D volume $X$ of size $H \times W \times D$. When training single-slice based 2D deep networks for 3D segmentation, each 3D volume is sliced along three axes, the coronal, sagittal, and axial. We denote these 2D slices with $X_{C,h}$ ($h = 1, 2, ..., H$), $X_{S,w}$ ($w = 1, 2, ..., W$) and
Figure 3: The proposed architecture for processing thickened inputs. It divides the backbone into two parts. Information is propagated in the first part and fused in the second one. By multiplexing the first part of the backbone, this method only requires a small number of extra parameters. Different feature maps are denoted in different colors. (Best viewed in color.)

\[
X_{A,d} = [X_{A,d}, X_{A,d+1}, \ldots, X_{A,d+k-1}],
\]

where the subscripts C, S and A stand for coronal, sagittal, and axial, respectively. The models trained on each axis are denoted by \( M_C, M_S, \) and \( M_A \), respectively. We consider a 2D slice along the axial view, denoted by \( X_{A,d} \). Our goal is to infer a binary segmentation mask \( Z_{A,d} \) of the same shape. Usually, this is achieved by first computing a probability map \( P_{A,d} = f[X_{A,d}; \theta] \), where \( f[; \theta] \) is a deep segmentation network with \( \theta \) being network parameters, and afterwards, all predictions on single slice are concatenated into a 3D volume \( P_A = [P_{A,1}, P_{A,2}, \ldots, P_{A,D}] \). The final prediction are fused on three axes, \( P = F(P_C, P_S, P_A) \), where \( F \) is a fusion function, and \( P \) is binarized into \( Z \) using a fixed threshold, say 0.5, i.e., \( Z = \mathbb{1}[P \geq 0.5] \).

The prediction of 2D method is only based on single slice, which results in a lack of contextual information. Although 3D information is compensated in a post-processing manner through fusing predictions from three viewpoints, it is not enough to produce satisfying results on some hard target. Therefore, we put forward with incorporating contextual information into 2D network by thickening the 2D inputs.

Without loss of generality, the input is a k-slice group \( X_{A,d}^k = [X_{A,d}, X_{A,d+1}, \ldots, X_{A,d+k-1}] \), which are the slices starting from \( d \) to \( d + k - 1 \). The model also outputs a corresponding k-slice prediction \( P_{A,d}^k = f[X_{A,d}^k; \theta] \), where \( P_{A,d}^k = [P_{A,d}, P_{A,d+1}, \ldots, P_{A,d+k-1}] \). \( P_A \) is obtained with a re-group function \( G \), i.e., \( P_A = G[P_{A,1}^k, P_{A,2}^k, \ldots, P_{A,D-k+1}^k] \). In this paper, the re-group function \( G \) is assembling the groups and averaging on overlapped slices.

### 3.2.2 Information Loss

Despite that increasing slice thickness to a moderate degree boosts performance, we observe that increasing thickness to a larger number results in performance drop. We notice that the reason is the information loss brought by 2D convolution operation which fuses different slices. A 2D convolution can be regarded as weighted-sum of all input channels for each output channel, so there is no special connection between specific output channel and corresponding input channel, which leads to a confusion of intra-slice information and inter-slice information. Say \( R^C \) is a feature map with \( C \) channels, in a typical ResNet network, the feature map mapping is \( R^k \rightarrow R^{54} \rightarrow \cdots \rightarrow R^{2048} \rightarrow R^{256} \rightarrow R^k \). For the input \( R^k \) and output \( R^k \), their channels should have one-to-one relationship in a k-slice medical image segmentation. When \( k \) is small, say 1 or 3, it is easy for the network to figure out the mapping relations. But when \( k \) is large, say 12, the network can be confused after so many 2D convolutions, which result in a loss of slice-sensitive information, and this information loss leads to confusion and a worse result.

### 3.3. Adapting 2D Networks for Thickened Input

The key of alleviating such information loss with thickened 2D inputs is to postpone the stage that information from multiple slices is fused and introduce slice-sensitive features. For this purpose, we propose early-stage multiplexing (ESM) and slice-sensitive attention (SSA) to address information loss for thickened inputs.

#### 3.3.1 Early-Stage Multiplexing

Firstly, one way to address the information loss is to encode every slice into a same feature space before fusion, which reduces the variance among different slices and also filters out large amount of useless background information. Specifically, we use multiple small-thickness groups instead of one large-thickness group. By multiplexing part of the backbone for each mini-group, the fusion stage is postponed. Without loss of generality, we divide the original 2D segmentation backbone \( f[; \theta] \) into two parts, \( f_1[; \theta_1] \) and \( f_2[; \theta_2] \). In the first part, each mini-group forward propagates individually, therefore the intra-slice information is...
learned for each slice. Then in the remaining part, mini-groups get fused as one unity and inter-slice information is explored. With this modification, segmentation procedure on mini-groups with \( k \)-slice input becomes:

\[
\begin{align*}
X_{A,d}^k &= [X_{A,d}, X_{A,d+1}, \ldots, X_{A,d+k-1}], \\
O_{A,d} &= f_1[X_{A,d}; \theta_1], \\
O_{A,d}^k &= [O_{A,d}, O_{A,d+1}, \ldots, O_{A,d+k-1}], \quad (1) \\
Z_{A,d}^k &= f_2[O_{A,d}^k; \theta_2], \\
Z_A &= \mathcal{G}[Z_{A,1}, Z_{A,2}, \ldots, Z_{A,D}].
\end{align*}
\]

Here, \( O_{A,d} \) is an intermediate feature map for slice \( d \) at pre-fusion layer, and \( O_{A,d}^k \) is the normalisation function, which helps the network obtain distinguishable features for each slice before fusion, and leads to a better ability to figure out the corresponding relationship between input slices and output slices and a better result. The overall architecture with early-stage multiplexing is shown in Figure 3.

### 3.3.2 Slice-Sensitive Attention

We further address the problem of information loss by introducing slice-sensitive information. In second part of the network, the final feature map is a mixture of inter-slice information and intra-slice information, and the network predicts for different slices based on this same mixed feature map, which is hard if no specific-slice information is given. So we introduce slice-sensitive auxiliary cues from pre-fusion layer to the final feature map. Technically, we add highway connections with a slice-sensitive attention module from pre-fusion layer to prediction layer. And the slice-sensitive information from previous part of the backbone helps make a clearer prediction in attention mechanisms.

Specifically, we introduce slice-sensitive attention (SSA) module, which is illustrated in Figure 4. We follow [31, 33] to design an attention module in non-local manner. However, in our case the main problem lies in confusion in relationship between slices at channel-level. So the attention map should also focus on channel-wise relationship instead of pixel-wise one. Suppose the input feature map has a shape of \((C, H, W)\). Instead of computing the relationship between each location pair, we compute the correspondence between each pair of feature dimension (channel), formulated as \( y_p = \frac{1}{p \times q} \sum_{q} f(x_p, x_q)g(x_q) \), where \( p, q \in \{1, \ldots, C\} \), \( x_p, x_q \) represent \( H \times W \) dimensional vectors for the \( p \)-th and \( q \)-th channel and \( Z(x) \) is the normalisation coefficient. A pairwise function \( f \) computes a scalar between channel \( p \) and all channel \( q \). The unary function \( g \) computes a representation of the input \( x \) at the channel \( q \). This module intuitively tells where to look at for each dimension of high level feature, thus separate information for each slice from the mixed information during computation.

After equipped with slice-sensitive attention module, the segmentation procedure is updated with:

\[
\begin{align*}
U_{A,d}^k &= f_2[O_{A,d}^k; \theta_2], \\
Z_{A,d} &= \text{SSA}[U_{A,d}^k, O_{A,d}; \eta],
\end{align*}
\]

where SSA is a slice-sensitive attention module with parameters \( \eta \). \( U_{A,d}^k \) is the feature for all \( k \) slices, which is the output of second part of the backbone.

### 4. Experiments

#### 4.1. Datasets and Evaluation

To verify that our approach can be applied to various organs, we collect a large dataset which contains 200 CT scans. This corpus took 4 full-time radiologists around 3 months to annotate. We choose several blood vessels which require more contextual information and also other challenging organs like pancreas and duodenum. We randomly partition the dataset into two parts, one containing 150 cases for training, while another consisting of 50 cases for testing. Each organ is trained and tested individually. When a pixel is predicted as more than one organ, we choose the one with the largest confidence score.

We also conduct experiments on a public dataset from Medical Segmentation Decathlon which contains 303 cases of hepatic vessels, which are challenging if addressed without contextual information. We use 228 cases for training and 75 cases for testing.

The accuracy of segmentation is evaluated by the Dice-Sørensen coefficient (DSC): \( \text{DSC} (Y,Z) = \frac{2|Y \cap Z|}{|Y| + |Z|} \). This metric falls in the range of \([0, 1]\), with 1 implying perfect segmentation. We also compare the inference latency of different methods to demonstrate that our method is not only...
more robust from task to task but also has a much higher inference speed when it is tested in 2D manner.

4.2. Implementation Details

4.2.1 Network Architectures

We use DeepLabV3+ [6] based on ResNet50 [12] without ASPP module as our 2D backbone. The input slices are divided into mini-groups, 3 slices in each group. To obtain meaningful intra-slice information, each 3-slice group will go through the first convolution and the following layer1 part of the network. At the fusion stage, we concatenate the feature maps from different groups on channel dimension and use two convolution layers to compress it, one will compress the channel number to its half and another to 256.

After features go through the remaining part of the backbone as one unity, we apply a slice-sensitive attention module to extract the targeted features for each mini-group. Thus we obtain a prediction focusing on the targeted slices. The slice-sensitive attention module is designed in a channel-wise manner because the confusion comes from channel-level instead of pixel-level. Besides, we use an adaptive average pooling to 32 × 32 size followed by a 1 × 1 convolution to pre-process the input. The average pooling and convolution are meant to reduce the variance of different receptive field and also computation cost. We also add a switchable normalization [21] at the output to reinforce the module.

4.2.2 2D Settings

We train the proposed method in a 2D setting. The training phase aims at minimizing DSC loss, which is denoted by $\mathcal{L}(Y^k_{A,d}, P^k_{A,d}) = 1 - \frac{2 \times \sum_i Y^k_{A,d,i} \cdot P^k_{A,d,i}}{\sum_i (Y^k_{A,d,i} + P^k_{A,d,i})}$, where $P^k_{A,d}$ is the prediction on the input $k$ slices. The model is initialized with ImageNet [16] pre-trained weight, and trained for 100k iterations with SGD optimizer, where the momentum is 0.9 and weight decay 0.0005. The batch size is 8, and learning rate is 0.005, which decays by a factor of 10 at iteration 70k and 90k. To align different slice sizes so that the model could be trained in parallel, images are cropped or zero padded to make sure that each slice size should be 512 × 512. Three models on each axis are trained separately. The whole training procedure takes 14 hours on 4 NVIDIA TITAN-Xp GPUs.

At testing phase, the volume is sliced into 15-slice groups where each group has 14 slices overlapped by each other. The final prediction is averaged for each slice. The output ranges in [0, 1] and we use a threshold of 0.5 to get final prediction. Predictions from three viewpoints are fused by majority voting.

4.2.3 3D Settings

All 3D models are trained with the same 3D setting described in this section, we also train our method in this setting to demonstrate its effectiveness in capturing contextual information. In 3D settings, we adopt SGD optimizer with polynomial learning rate scheduler starting from 0.01 with power of 0.9. The batch size is 128 × 128 × 64 and batch size is tuned to take full usage of the 12GB GPU memory. The training procedure lasts for 40k iterations.

At testing stage, we adopt a sliding-window manner with stride [32, 32, 16] for each axis respectively. The probability on each voxel will be averaged based on all probabilities it gets.

4.3. Results on Our Multi-organ Dataset

4.3.1 Validation Results

Results are summarized in Table 1. We have conducted experiments on several challenging blood vessels and organs, including superior m.a., celiac a.a., vein and pancreas, duodenum. On blood vessels, single slice based network usually cannot produce satisfying results, and inter-slice information plays an important role here. We also try to apply our algorithm to other small and challenging organs, proving that our method also works for other more general organs besides blood vessels. With our method, the inter-slice information and intra-slice information collaborate in a better way.

We report the results of our method in 3D setting, 2D single-view setting, and 2D three-views setting respectively. In 3D setting, our model is trained in the same 3D manner as baselines, which is described in 4.2. Our model shows a great ability to capture contextual information when trained in the same 3D setting as baselines.

In 2D setting, we report our results based on single-view and three-views fusion, respectively. Our single-view result is good enough and enjoy a lower inference latency. We also follow typical 2D methods to use a three-view fusion to add contextual information in a post-processing manner, and the results are boosted higher, yet still with a much lower inference latency compared with 3D baselines.

We compare our results with popular 3D models 3D U-Net, V-Net and a hybrid network AH-Net to verify that our model makes good use of 3D information while being efficient, and it turns out our results are consistently better than baselines. Due to the lack of intra-slice information, a 3D network suffers from serious false positive especially on small blood vessels which only take up thousands of voxels. And because of the lack of pre-trained weight, 3D methods are also unstable on different tasks using the same hyper-parameters. However, our method is more stable and much faster due to the thickened 2D design.

4.3.2 Diagnoses and Ablation Studies

We study the proposed Early-Stage Multiplexing (ESM) and Slice-Sensitive Attention (SSA) and different input
Table 1: Experiments on our multi-organ dataset, including superior m.a., celiac a.a., pancreas, duodenum, and vein, and on MSD dataset, including hepatic vessel. All models in 3D settings are trained with patch size $128 \times 128 \times 64$ and a customized batch size tuned to take full usage of the GPU memory. Inference time is evaluated with a $512 \times 512 \times 394$ case. Inference time of 2D networks with three-views fusion is the summation of three-view inference time. Single-view 2D models are trained and tested on coronal view. “Ours (3D)” means to run our method in the same 3D settings, so that the ability to capture contextual information is fairly compared between our method and other 3D models. It is noticeable that our method can be run in both 2D and 3D settings, yet 3D models are not capable to run in 2D settings.

| Methods       | Superior m.a. | Celiac a.a. | Duodenum | Pancreas | Vein | Hepatic Vessel | Latency (s) |
|---------------|---------------|-------------|----------|----------|------|----------------|-------------|
| 3D U-Net [8]  | 72.57%        | 56.31%      | 68.59%   | 85.89%   | 71.05%| 55.05%         | 446.64      |
| V-Net [22]    | 72.37%        | 55.87%      | 73.18%   | 85.98%   | 72.86%| 54.38%         | 833.42      |
| AH-Net [19]   | 73.03%        | 56.85%      | 70.16%   | 86.40%   | 71.97%| 60.61%         | 226.14      |
| Ours (3D)     | 73.82%        | 62.66%      | 70.89%   | 86.48%   | 72.36%| 59.12%         | 367.97      |
| Ours (2D) - SingleView | 73.06% | 60.61% | 70.16% | 86.53% | 75.45% | 58.33% | 110.75       |
| Ours (2D) - ThreeViews | 74.55% | 61.61% | 74.71% | 87.36% | 76.23% | 62.67% | 335.04       |

Table 2: Inter-slice similarity of different methods, along the axial view (smaller numbers are better). Our approach enjoys a better ability of discriminating different slices.

Distinguish-ability. We measure the ability of model to distinguish each slice by comparing the prediction of neighboring slices. In detail, we design a statistics, named inter-slice similarity, to measure the distinguish-ability of models. We compute the dice score between every pair of neighboring slices along axial axis. The closer this number is to inter-slice dice score computed based on ground truth, the better distinguish-ability the model has, which means the prediction has a more similar smoothness as ground truth. We use L2-distance to evaluate the similarity of inter-slice dice score of prediction and that of ground truth. The results are summarized in Table 2. The distinguish-ability decreased for DeepLab when facing thickened inputs. However, with the proposed method (ESM and SSA), the distinguish-ability increases and is even higher than the model dealing with normal input.

Ways to Introduce Slice-Sensitive Information. Based on 12-slice models and superior m.a., we have compared different ways to instantiate attention which combines the specific-slice feature and all-slice feature. As shown in Table 3, directly applying early-stage multiplexing already brings benefits to the model. But the improvement will be less if we introduce slice-sensitive information in an improper way, like concatenate or element-wise dot. With a slice-sensitive module, the feature maps will be re-weighted based on a channel-wise relationship, thus the performance is boosted higher.

Slice Thickness. We try different slice thickness to test our method capacity, from 3-slice to at most 24-slice with same batch size under 12G GPU memory limitation. The results are summarized in Table 4. We find that the results keep increasing until slice thickness reaches 15, while the axial model with over 18-slice model does not converge. The trend of performance increase is different for each viewpoint when slice thickness increases. For plane coronal, a major improvement happens when slice thickness increase from 3 to 6 (+2.12%). And models with slice thickness over 9 produce similar results. For plane sagittal, there happen two major improvements, one is from 3 to 6 (+0.85%) and another is from 12 to 15 (+1.30%).
Table 4: Dice score of our method (Early-Stage Multiplexing + Slice-Sensitive Attention) with different slice thickness on superior m.a.. We try different slice thickness from 3 to 24. It can be observed that our model performs well even for a much larger slice thickness. ‘-’ means the model does not converge, if some plane result is missing, the fusion will be the average on the remaining planes.

|     | 3-slice | 6-slice | 9-slice | 12-slice | 15-slice | 18-slice | 21-slice | 24-slice |
|-----|---------|---------|---------|----------|----------|----------|----------|----------|
| DSC_C | 70.37%  | 72.49%  | 73.07%  | 73.23%   | 73.06%   | 73.04%   | 72.73%   | 72.72%   |
| DSC_S | 70.49%  | 71.34%  | 71.65%  | 71.67%   | 72.97%   | 72.17%   | 73.17%   | 72.46%   |
| DSC_A | 70.70%  | 70.59%  | 72.76%  | 72.54%   | 72.64%   | -        | -        | -        |
| DSC_F | 72.61%  | 73.21%  | 74.05%  | 74.17%   | 74.55%   | 73.61%   | 74.17%   | 73.78%   |

Table 4: Dice score of our method (Early-Stage Multiplexing + Slice-Sensitive Attention) with different slice thickness on superior m.a.. We try different slice thickness from 3 to 24. It can be observed that our model performs well even for a much larger slice thickness. ‘-’ means the model does not converge, if some plane result is missing, the fusion will be the average on the remaining planes.

4.4. Results on the MSD Dataset

We further apply our approach on a public dataset – hepatic vessel segmentation in Medical Segmentation Decathlon [29]. Hepatic vessels have a tree-structure and are challenging if addressed by common 2D networks. Results are shown in Table 1. Our results outperform competitors, which illustrates the ability of our approach in capturing contextual information for such a tree-structure target.

We visualize two cases of hepatic vessel and one case of celiac a.a. and superior m.a. with ITK-SNAP [37] (see Fig 5). First two rows are for hepatic vessel, which is contiguous and like a multi-branch tree. Compared with baseline DeepLab, our method with thickened inputs does better in capturing this vessels’ contiguity. In the fourth row of superior m.a., DeepLab fails to predict at some points, resulting in a worse continuity of the prediction. Yet our method makes accurate prediction at thin bottlenecks of blood vessel with captured contextual information. We observe no obvious failure (a typical fail in baseline single axis result) and a better segmentation consistency in our approach.

5. Conclusions

This paper is motivated by the need of capturing 3D contexts, and aims at designing a network structure which works on thickened 2D inputs. The major obstacle is the information loss brought by fusion along the third dimension. We propose two keys to deal with this issue, i.e., postponing the stage of information fusion by early-stage multiplexing, and creating a shortcut connection based on slice-sensitive attention module between the pre-fusion stage and the final decision stage. Evaluated on a few medical image segmentation datasets, our approach reports higher segmentation accuracy and lower inference latency with thickened input data, demonstrating the effectiveness of capturing 3D contexts.

Our research sheds light on designing efficient 3D networks for segmenting volumetric data. The success of our approach provides a piece of side evidence that both 2D and 3D networks are not the optimal solution, as 2D methods benefit from natural image pre-training but inevitably lack contexts, meanwhile 3D methods usually waste time...
and memory for unnecessary computations. Absorbing advantages from both of them remains an open problem and deserves more efforts in future research.

References

[1] Asem M Ali, Aly A Farag, and Ayman S El-Baz. Graph cuts framework for kidney segmentation with prior shape constraints. In MICCAI, pages 384–392. Springer, 2007.

[2] Jinzheng Cai, Le Lu, Zizhao Zhang, Fuyong Xing, Lin Yang, and Qian Yin. Pancreas segmentation in mri using graph-based decision fusion on convolutional neural networks. In MICCAI, pages 442–450. Springer, 2016.

[3] Hao Chen, Qi Dou, Lequan Yu, Jing Qin, and Pheng-Ann Heng. Voxresnet: Deep voxelwise residual networks for brain segmentation from 3d mr images. NeuroImage, 170:446–455, 2018.

[4] Jianxu Chen, Lin Yang, Yizhe Zhang, Mark Alber, and Danny Z Chen. Combining fully convolutional and recurrent neural networks for 3d biomedical image segmentation. In NIPS, pages 3036–3044, 2016.

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. PAMI, 40(4):834–848, 2018.

[6] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In ECCV, pages 801–818, 2018.

[7] Chengwen Chu, Masahiro Oda, Takayuki Kitasaka, Kazu-nari Misawa, Michitaka Fujiwara, Yuichiro Hayashi, Yuki-taka Nimura, Daniel Rueckert, and Kensaku Mori. Multi-organ segmentation based on spatially-divided probabilistic atlas from 3d abdominal ct images. In MICCAI, pages 165–172. Springer, 2013.

[8] Özgün Çicek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3D u-net: Learning dense volumetric segmentation from sparse annotation. In MICCAI, 2016.

[9] Qi Dou, Lequan Yu, Hao Chen, Yueming Jin, Xin Yang, Jing Qin, and Pheng-Ann Heng. 3D deep supervised network for automated segmentation of volumetric medical images. Medical Image Analysis, 41:40–54, 2017.

[10] Eli Gibson, Francesco Giganti, Yipeng Hu, Estor Bonmati, Steve Bandula, Kurinchi Gurusamy, Brian Davidson, Stephen P Pereira, Matthew J Clarkson, and Dean C Barratt. Automatic multi-organ segmentation on abdominal ct with dense v-networks. TMI, 37(8):1822–1834, 2018.

[11] Mohammad Havaei, Axel Davy, David Warde-Farley, Antoine Biard, Aaron Courville, Yoshua Bengio, Chris Pal, Pierre-Marc Jodoin, and Hugo Larochelle. Brain tumor segmentation with deep neural networks. Medical image analysis, 35:18–31, 2017.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In CVPR, pages 770–778, 2016.

[13] Shiyeng Hu, Eric A Hoffman, and Joseph M Reinhardt. Automatic lung segmentation for accurate quantitation of volumetric x-ray ct images. TMI, 20(6):490–498, 2001.

[14] Dakai Jin, Ziyue Xu, Youbao Tang, Adam P Harrison, and Daniel J Mollura. Ct-realistic lung nodule simulation from 3d conditional generative adversarial networks for robust lung segmentation. In MICCAI, pages 732–740. Springer, 2018.

[15] Konstantinos Kamnitas, Christian Ledig, Virginia FJ Newcombe, Joanna P Simpson, Andrew D Kane, David K Menon, Daniel Rueckert, and Ben Glover. Efficient multiscale 3d cnn with fully connected crf for accurate brain lesion segmentation. MIA, 36:61–78, 2017.

[16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In NIPS, pages 1097–1105, 2012.

[17] Matthew Lai. Deep learning for medical image segmentation. arXiv preprint arXiv:1505.02000, 2015.

[18] Xiaomei Li, Hao Chen, Xiaojuan Qi, Qi Dou, Chi-Wing Fu, and Pheng Ann Heng. H-denseunet: Hybrid densely connected unet for liver and liver tumor segmentation from ct volumes. TMI, 2017.

[19] Siqi Liu, Daguang Xu, S Kevin Zhou, Olivier Pauly, Sasa Grbic, Thomas Mertelmeier, Julia Wicklein, Anna Jerebko, Weidong Cai, and Dorin Comaniciu. 3d anisotropic hybrid network: Transferring convolutional features from 2d images to 3d anisotropic volumes. In MICCAI, pages 851–858. Springer, 2018.

[20] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In CVPR, pages 3431–3440, 2015.

[21] Ping Luo, Jiamin Ren, and Zhanglin Peng. Differentiable learning-to-normalize via switchable normalization. ICLR, 2019.

[22] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In JV, pages 565–571. IEE, 2016.

[23] Tianwei Ni, Lingxi Xie, Huangjie Zheng, Elliot K Fishman, and Alan L Yuille. Elastic boundary projection for 3d medical image segmentation. CVPR, 2019.

[24] Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazu-nari Misawa, Kensaku Mori, Steven McDonagh, Nils Y Hammerla, Bernhard Kainz, et al. Attention u-net: Learning where to look for the pancreas. MIDL, 2018.

[25] Jeremie Papon, Alexey Abramov, Markus Schoeler, and Florian Grbic. 3d anisotropic hybrid segmentation. In MIA, 36:61–78, 2017.

[26] Menon, Daniel Rueckert, and Ben Glocker. Efficient multi-organ segmentation based on spatially-divided probabilistic atlas from 3d abdominal ct images. In MICCAI, pages 165–172. Springer, 2013.

[27] Matthew Lai. Deep learning for medical image segmentation. arXiv preprint arXiv:1505.02000, 2015.

[28] Xiaomei Li, Hao Chen, Xiaojuan Qi, Qi Dou, Chi-Wing Fu, and Pheng Ann Heng. H-denseunet: Hybrid densely connected unet for liver and liver tumor segmentation from ct volumes. TMI, 2017.

[29] Siqi Liu, Daguang Xu, S Kevin Zhou, Olivier Pauly, Sasa Grbic, Thomas Mertelmeier, Julia Wicklein, Anna Jerebko, Weidong Cai, and Dorin Comaniciu. 3d anisotropic hybrid network: Transferring convolutional features from 2d images to 3d anisotropic volumes. In MICCAI, pages 851–858. Springer, 2018.

[30] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In CVPR, pages 3431–3440, 2015.

[31] Ping Luo, Jiamin Ren, and Zhanglin Peng. Differentiable learning-to-normalize via switchable normalization. ICLR, 2019.

[32] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In JV, pages 565–571. IEE, 2016.

[33] Tianwei Ni, Lingxi Xie, Huangjie Zheng, Elliot K Fishman, and Alan L Yuille. Elastic boundary projection for 3d medical image segmentation. CVPR, 2019.

[34] Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazu-nari Misawa, Kensaku Mori, Steven McDonagh, Nils Y Hammerla, Bernhard Kainz, et al. Attention u-net: Learning where to look for the pancreas. MIDL, 2018.

[35] Jeremie Papon, Alexey Abramov, Markus Schoeler, and Florentin Worgotter. Voxel cloud connectivity segmentation-supervoxels for point clouds. In CVPR, pages 2027–2034, 2013.

[36] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In MICCAI, pages 234–241. Springer, 2015.

[37] Holger R Roth, Le Lu, Amal Farag, Hoo-Chang Shin, Jiamin Liu, Evrim B Turkbey, and Ronald M Summers. Deepogan: Multi-level deep convolutional networks for automated pancreas segmentation. In MICCAI, 2015.
[28] Holger R Roth, Le Lu, Amal Farag, Andrew Sohn, and Ronald M Summers. Spatial aggregation of holistically-nested networks for automated pancreas segmentation. In *MICCAI*, 2016. 3

[29] Amber L Simpson, Michela Antonelli, Spyridon Bakas, Michel Bilello, Keyvan Farahani, Bram van Ginneken, Annette Kopp-Schneider, Bennett A Landman, Geert Litjens, Bjoern Menze, et al. A large annotated medical image dataset for the development and evaluation of segmentation algorithms. *arXiv preprint arXiv:1902.09063*, 2019. 2, 8

[30] Nima Tajbakhsh, Jae Y Shin, Suryakanth R Gurudu, R Todd Hurst, Christopher B Kendall, Michael B Gotway, and Jianming Liang. Convolutional neural networks for medical image analysis: Full training or fine tuning? *TMI*, 35(5):1299–1312, 2016. 3

[31] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, June 2018. 5

[32] Zehan Wang, Kanwal K Bhatia, Ben Glocker, Antonio Marvao, Tim Dawes, Kazunari Misawa, Kensaku Mori, and Daniel Rueckert. Geodesic patch-based segmentation. In *MICCAI*, pages 666–673. Springer, 2014. 2

[33] Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krähenbühl, and Ross Girshick. Long-term feature banks for detailed video understanding. *CVPR*, 2019. 5

[34] Yingda Xia, Fengze Liu, Dong Yang, Jizheng Cai, Lequan Yu, Zhuotun Zhu, Daguang Xu, Alan Yuille, and Holger Roth. 3d semi-supervised learning with uncertainty-aware multi-view co-training. *WACV*, 2020. 3

[35] Yingda Xia, Lingxi Xie, Fengze Liu, Zhuotun Zhu, Elliot K Fishman, and Alan L Yuille. Bridging the gap between 2d and 3d organ segmentation with volumetric fusion net. In *MICCAI*, pages 445–453. Springer, 2018. 3

[36] Qihang Yu, Lingxi Xie, Yan Wang, Yuyin Zhou, Elliot K. Fishman, and Alan L. Yuille. Recurrent saliency transformation network: Incorporating multi-stage visual cues for small organ segmentation. In *CVPR*, June 2018. 2

[37] Paul A. Yushkevich, Joseph Piven, Heather Cody Hazlett, Rachel Gimpel Smith, Sean Ho, James C. Gee, and Guido Gerig. User-guided 3D active contour segmentation of anatomical structures: Significantly improved efficiency and reliability. *Neuroimage*, 31(3):1116–1128, 2006. 8

[38] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, pages 2881–2890, 2017. 2

[39] Yuyin Zhou, Lingxi Xie, Wei Shen, Yan Wang, Elliot K Fishman, and Alan L Yuille. A fixed-point model for pancreas segmentation in abdominal ct scans. In *MICCAI*, pages 693–701. Springer, 2017. 2

[40] Zhuotun Zhu, Yingda Xia, Wei Shen, Elliot K Fishman, and Alan L Yuille. A 3d coarse-to-fine framework for automatic pancreas segmentation. *3DV*, 2018. 3