Recurrent Saliency Transformation Network: 
Incorporating Multi-Stage Visual Cues for Small Organ Segmentation

Qihang Yu¹, Lingxi Xie²(✉), Yan Wang², Yuyin Zhou², Elliot K. Fishman³, Alan L. Yuille²
¹Peking University  ²The Johns Hopkins University  ³The Johns Hopkins Medical Institute
{yucornetto,198808xc,wyanny.9,zhouyuyiner,alan.l.yuille}@gmail.com  efishman@jhmi.edu

Abstract

We aim at segmenting small organs (e.g., the pancreas) from abdominal CT scans. As the target often occupies a relatively small region in the input image, deep neural networks can be easily confused by the complex and variable background. To alleviate this, researchers proposed a coarse-to-fine approach [46], which used prediction from the first (coarse) stage to indicate a smaller input region for the second (fine) stage. Despite its effectiveness, this algorithm dealt with two stages individually, which lacked optimizing a global energy function, and limited its ability to incorporate multi-stage visual cues. Missing contextual information led to unsatisfying convergence in iterations, and that the fine stage sometimes produced even lower segmentation accuracy than the coarse stage.

This paper presents a Recurrent Saliency Transformation Network. The key innovation is a saliency transformation module, which repeatedly converts the segmentation probability map from the previous iteration as spatial weights and applies these weights to the current iteration. This brings us two-fold benefits. In training, it allows joint optimization over the deep networks dealing with different input scales. In testing, it propagates multi-stage visual information throughout iterations to improve segmentation accuracy. Experiments in the NIH pancreas segmentation dataset [34] (best viewed in color). We highlight the pancreas in red seen from three different viewpoints. It is a relatively small organ with irregular shape and boundary.

Figure 1. A typical example from the NIH pancreas segmentation dataset [34] (best viewed in color). We highlight the pancreas in red seen from three different viewpoints. It is a relatively small organ with irregular shape and boundary.

1. Introduction

This paper focuses on small organ (e.g., the pancreas) segmentation from abdominal CT scans, which is an important prerequisite for enabling computers to assist human doctors for clinical purposes. This problem falls into the research area named medical imaging analysis. Recently, great progress has been brought to this field by the fast development of deep learning, especially convolutional neural networks [18][27]. Many conventional methods, such as the graph-based segmentation approaches [1] or those based on handcrafted local features [41], have been replaced by deep segmentation networks, which typically produce higher segmentation accuracy [33][34].

Segmenting a small organ from CT scans is often challenging. As the target often occupies a small part of input data (e.g., less than 1.5% in a 2D image, see Figure 1), deep segmentation networks such as FCN [27] and DeepLab [5] can be easily confused by the background region, which may contain complicated and variable contents. This motivates researchers to propose a coarse-to-fine approach [46] with two stages, in which the coarse stage provides a rough localization and the fine stage performs accurate segmentation. But, despite state-of-the-art performance achieved in pancreas segmentation, this method suffers from inconsistency between its training and testing flowcharts, which is to say, the training phase dealt with coarse and fine stages individually and did not minimize a global energy function, but the testing phase assumed that these two stages can
cooperate with each other in an iterative process. From another perspective, this also makes it difficult for multi-stage visual cues to be incorporated in segmentation, e.g., the previous segmentation mask which carries rich information is discarded except for the bounding box. As a part of its consequences, the fine stage consisting of a sequence of iterations cannot converge very well, and sometimes the fine stage produced even lower segmentation accuracy than the coarse stage (see Section 4).

Motivated to alleviate these shortcomings, we propose a Recurrent Saliency Transformation Network. The chief innovation is to relate the coarse and fine stages with a saliency transformation module, which repeatedly transforms the segmentation probability map from previous iterations as spatial priors in the current iteration. This brings us two-fold advantages over [46]. First, in the training phase, the coarse-scaled and fine-scaled networks are optimized jointly, so that the segmentation ability of each of them gets improved. Second, in the testing phase, the segmentation mask of each iteration is preserved and propagated throughout iterations, enabling multi-stage visual cues to be incorporated towards more accurate segmentation. To the best of our knowledge, this idea was not studied in the computer vision community, as it requires making use of some special properties of CT scans (see Section 3).

We perform experiments on two CT datasets for small organ segmentation. On the NIH pancreas segmentation dataset [34], our approach outperforms the state-of-the-art by an average of over 2%, measured by the average Dice-Sørensen coefficient (DSC). On another multi-organ dataset collected by the radiologists in our team, we also show the superiority of our approach over the baseline on a variety of small organs. In the testing phase, our approach enjoys better convergence properties, which guarantees its efficiency and reliability in real clinical applications.

The remainder of this paper is organized as follows. Section 2 briefly reviews related work, and Section 3 describes the proposed approach. After experiments are shown in Sections 4 and 5, we draw our conclusions in Section 6.

2. Related Work

Computer-aided diagnosis (CAD) is an important technique which can assist human doctors in many clinical scenarios. An important prerequisite of CAD is medical imaging analysis. As a popular and cheap way of medical imaging, contrast-enhanced computed tomography (CECT) produces detailed images of internal organs, bones, soft tissues and blood vessels. It is of great value to automatically segment organs and/or soft tissues from these CT volumes for further diagnosis [2] [40] [13] [45]. To capture specific properties of different organs, researchers often design individualized algorithms for each of them. Typical examples include the the liver [25] [15], the spleen [26], the kidneys [23] [1], the lungs [16], the pancreas [7] [31], etc. Small organs (e.g., the pancreas) are often more difficult to segment, partly due to their low contrast and large anatomical variability in size and (most often irregular) shape.

Compared to the papers cited above which used conventional approaches for segmentation, the progress of deep learning brought more powerful and efficient solutions. In particular, convolutional neural networks have been widely applied to a wide range of vision tasks, such as image classification [18] [37] [14], object detection [10] [32], and semantic segmentation [27] [5]. Recurrent neural networks, as a related class of networks, were first designed to process sequential data [11] [39], and later generalized to image classification [22] and scene labeling [31] tasks. In the area of medical imaging analysis, in particular organ segmentation, these techniques have been shown to significantly outperform conventional approaches, e.g., segmenting the liver [8], the lung [12], or the pancreas [35] [3] [36]. Note that medical images differ from natural images in that data appear in a volumetric form. To deal with these data, researchers either slice a 3D volume into 2D slices (as in this work), or train a 3D network directly [29] [30] [17] [43]. In the latter case, limited GPU memory often leads to patch-based training and testing strategies. The tradeoff between 2D and 3D approaches is discussed in [20].

By comparison to the entire CT volume, the organs considered in this paper often occupy a relatively small area. As deep segmentation networks such as FCN [27] are less accurate in depicting small targets, researchers proposed two types of ideas to improve detection and/or segmentation performance. The first type involved rescaling the image so that the target becomes comparable to the training samples [42], and the second one considered to focus on a subregion of the image for each target to obtain higher accuracy in detection [4] or segmentation [46]. The coarse-to-fine idea was also well studied in the computer vision area for saliency detection [19] or semantic segmentation [21] [24]. This paper is based on a recent coarse-to-fine framework [46], but we go one step further by incorporating multi-stage visual cues in optimization.

3. Our Approach

We investigate the problem of segmenting an organ from abdominal CT scans. Let a CT image be a 3D volume $X$ of size $W \times H \times L$ which is annotated with a binary ground-truth segmentation $Y$ where $y_i = 1$ indicates a foreground voxel. The goal of our work is to produce a binary output volume $Z$ of the same dimension. Denote $\mathcal{Y}$ and $\mathcal{Z}$ as the set of foreground voxels in the ground-truth and prediction, i.e., $\mathcal{Y} = \{i \mid y_i = 1\}$ and $\mathcal{Z} = \{i \mid z_i = 1\}$. The accuracy of segmentation is evaluated by the Dice-Sørensen coefficient (DSC): $\text{DSC}(\mathcal{Y}, \mathcal{Z}) = \frac{2|\mathcal{Y} \cap \mathcal{Z}|}{|\mathcal{Y}| + |\mathcal{Z}|}$. This metric falls in the range of $[0, 1]$ with 1 implying perfect segmentation.
the complicated and variable background contents, deep networks such as FCN from being confused by foreground pixels is often smaller than are relatively small. In each 2D slice, the fraction of the GNet is trained 2 each axis, an individual 2D-FCN on a slice segmentation (in the training phase so as to prevent the models from tation so as to prevent the models from

We start with training 2D deep networks for 3D segmentation Each 3D volume X is sliced along three axes, the coronal, sagittal and axial views, and these 2D slices are denoted by \( X_{C,w} \) \( (w = 1, 2, \ldots, W) \), \( X_{S,h} \) \( (h = 1, 2, \ldots, H) \) and \( X_{A,l} \) \( (l = 1, 2, \ldots, L) \), where the subscripts C, S and A stand for coronal, sagittal and axial, respectively. On each axis, an individual 2D-FCN \([27]\) on a 16-layer VGGNet \([37]\) is trained. Three FCN models are denoted by \( M_{C} \), \( M_{S} \) and \( M_{A} \), respectively. We use the DSC loss \([30]\) in the training phase so as to prevent the models from being biased towards the background class. Both multislice segmentation (3 neighboring slices are combined as a basic unit in training and testing) and multi-axis fusion (majority voting over three axes) are performed to incorporate pseudo-3D information into segmentation.

The organs investigated in this paper (e.g., the pancreas) are relatively small. In each 2D slice, the fraction of the foreground pixels is often smaller than 1.5%. To prevent deep networks such as FCN \([27]\) from being confused by the complicated and variable background contents, \([40]\)

3.2. Recurrent Saliency Transformation Network

Following the baseline approach, we train an individual model for each of the three viewpoints. Without loss of generality, we consider a 2D slice along the axial view, denoted by \( X_{A,l} \). Our goal is to infer a binary segmentation mask \( Z_{A,l} \) of the same dimensionality. In the context of deep neural networks \([27]\) \([5]\), this is often achieved by first computing a probability map \( P_{A,l} = f(X_{A,l}; \theta) \), where \( f(\cdot; \theta) \) is a deep segmentation network (FCN throughout this paper) with \( \theta \) being network parameters, and then binarizing \( P_{A,l} \) into \( Z_{A,l} \) using a fixed threshold of 0.5, i.e., \( Z_{A,l} = \mathbb{1}[P_{A,l} \geq 0.5] \).

In order to assist segmentation with the probability map, we introduce \( P_{A,l} \) as a latent variable. We introduce a saliency transformation module, which takes the probability map to generate an updated input image, i.e., \( I_{A,l} = X_{A,l} \odot g(P_{A,l}; \eta) \), and uses the updated input \( I_{A,l} \) to replace \( X_{A,l} \). Here \( g(\cdot; \eta) \) is the transformation function with parameters \( \eta \), and \( \odot \) denotes element-wise product, i.e., the transformation function adds spatial weights to the original input image. Thus, the segmentation process becomes:

\[
P_{A,l} = f(X_{A,l} \odot g(P_{A,l}; \eta); \theta).
\]

This is a recurrent neural network. Note that the saliency transformation function \( g(\cdot, \eta) \) needs to be differentiable so that the entire recurrent network can be optimized in an end-to-end manner. As \( X_{A,l} \) and \( P_{A,l} \) share the same

---

1 Please see Section 4.3 for the comparison to 3D networks.

2 This is a simple segmentation baseline with a relatively shallow network. Deeper network structures such as ResNet \([14]\) and more complicated segmentation frameworks such as DeepLab \([5]\), while requiring a larger memory and preventing us from training two stages jointly (see Section 3.2), often result in lower segmentation accuracy as these models seem to over-fit in these CT datasets.
spatial dimensionality, we set \( g(\cdot, \eta) \) to be a size-preserved convolution, which allows the weight added to each pixel to be determined by the segmentation probabilities in a small neighborhood around it. As we will show in the experimental section (see Figure 5), the learned convolutional kernels are able to extract complementary information to help the next iteration.

To optimize Eqn \((1)\), we unfold the recurrent network into a plain form (see Figure 3). Given an input image \( X_{A,t} \) and an integer \( T \) which is the maximal number of iterations, we update \( I_{A,t} \) and \( P_{A,t} \), \( t = 0, 1, \ldots, T \):

\[
I_{A,t} = X_{A,t} \odot g\left(P_{A,t-1}; \eta\right), \quad (2)
\]

\[
P_{A,t} = f\left[I_{A,t}; \theta\right]. \quad (3)
\]

Note that the original input image \( X_{A,t} \) does not change, and the parameters \( \theta \) and \( \eta \) are shared by all iterations. At \( t = 0 \), we directly set \( I_{A,0} = X_{A,0} \).

When segmentation masks \( I_{A,t} \) (\( t = 0, 1, \ldots, T - 1 \)) are available for reference, deep networks benefit considerably from a shrunk input region especially when the target organ is very small. Thus, we define a cropping function \( \text{Crop} \left[ \cdot; P_{A,t} \right] \) which takes \( P_{A,t} \) as the reference map, binarizes it into \( Z_{A,t} = \left[ P_{A,t} \geq 0.5 \right] \), finds the minimal rectangle covering all the activated pixels, and adds a \( K \)-pixel-wide margin (padding) around it. We fix \( K \) to be 20; our algorithm is not sensitive to this parameter.

Finally note that \( I_{A,t} \), the original input (the entire 2D slice), is much larger than the cropped inputs \( I_{A,t} \) for \( t > 0 \). We train two FCN’s to deal with such a major difference in input data. The first one is named the coarse-scaled segmentation network, which is used only in the first iteration. The second one, the fine-scaled segmentation network, takes the charge of all the remaining iterations. We denote their parameters by \( \theta^C \) and \( \theta^F \), respectively. These two FCN’s are optimized jointly.

---

**Algorithm 1: The Testing Phase**

Input: input volume \( X; \) viewpoint \( V = \{C, S, A\}; \)
parameters \( \theta^C, \theta^F \) and \( \eta, \nu \in V; \)
max number of iterations \( T \); threshold \( \text{thr} \);

Output: segmentation volume \( Z; \)

1. \( t \leftarrow 0; I^{(0)}_{A,t} \leftarrow X, \nu \in V; \)
2. \( P^{(0)}_{V,t} \leftarrow f\left[I^{(0)}_{V,t}; \theta^C\right], \nu \in V, \forall t; \)
3. \( P^{(0)} = \frac{P^{(0)}_{C} + + P^{(0)}_{S} + P^{(0)}_{A}}{3}, Z^{(0)} = \left[ P^{(0)} \geq 0.5 \right]; \)
4. repeat
5. \( t \leftarrow t + 1; \)
6. \( I^{(t)}_{V,t} \leftarrow X_{V,t} \odot g\left(P^{(t-1)}_{V,t}; \eta\right), \nu \in V, \forall t; \)
7. \( P^{(t)}_{V,t} \leftarrow f\left[ \text{Crop} \left[ I^{(t)}_{V,t}; P^{(t-1)}_{V,t}; \theta^C\right], \nu \in V, \forall t; \)
8. \( P^{(t)} = \frac{P^{(t)}_{C} + + P^{(t)}_{S} + P^{(t)}_{A}}{3}, Z^{(t)} = \left[ P^{(t)} \geq 0.5 \right]; \)
9. until \( t = T \) or \( \text{DSC}\{Z^{(t-1)}, Z^{(t)}\} \geq \text{thr}; \)

Return: \( Z \leftarrow Z^{(t)} \).

We compute a DSC loss term on each probability map \( P_{A,t}, t = 0, 1, \ldots, T, \) and denote it by \( \mathcal{L}\{Y_{A,t}, P_{A,t}\} \).
Here, \( Y_{A,t} \) is the ground-truth segmentation mask, and \( \mathcal{L}\{Y, P\} = 1 - \frac{2 \times \sum_{i \in V} Y_{i} - P_{i}}{\sum_{i \in V} Y_{i} + P_{i}} \) is based on a soft version of DSC \[30\]. Our goal is to minimize the overall loss:

\[
\mathcal{L} = \sum_{t=0}^{T} \mathcal{L}\{Y^{(t)}_{A,t}, Z^{(t)}_{A,t}\}. \quad (4)
\]

This leads to joint optimization over all iterations, which involves network parameters \( \theta^C \), \( \theta^F \), and transformation parameters \( \eta \). \{\lambda_t\}_{t=0}^T \) controls the tradeoff among all loss terms. We set \( 2\lambda_0 = \lambda_1 = \ldots = \lambda_T = 2/(2T + 1) \) so as to encourage accurate fine-scaled segmentation.

### 3.3. Training and Testing

The training phase is aimed at minimizing the loss function \( \mathcal{L} \), defined in Eqn \((4)\), which is differentiable with respect to all parameters. In the early training stages, the coarse-scaled network cannot generate reasonable probability maps. To prevent the fine-scaled network from being confused by inaccurate input regions, we use the ground-truth mask \( Y_{A,t} \) as the reference map. After a sufficient number of training, we resume using \( P^{(t)}_{A,t} \) instead of \( Y_{A,t} \).

In Section \[12\], we will see that this “fine-tuning” strategy improves segmentation accuracy considerably.

Due to the limitation in GPU memory, in each mini-batch containing one training sample, we set \( T \) to be the maximal integer (not larger than 5) so that we can fit the entire framework into the GPU memory. The overall framework is illustrated in Figure \[5\]. As a side note, we find that setting \( T \equiv 1 \) also produces high accuracy, suggesting that major improvement is brought by joint optimization.
3.4. Discussions

Coarse-to-fine recognition is an effective idea in medical imaging analysis. Examples include [45], our baseline, and [4] for metosis detection. Our approach can be applied to most of them towards higher recognition performance.

Attention-based or recurrent models are also widely used for natural image segmentation [6][21][42][24]. Our approach differs from them in making full use of the special properties of CT scans, e.g., each organ appears at a roughly fixed position, and has a fixed number of components. Our approach can be applied to detecting the lesion areas of an organ [17][55], or a specific type of vision problems such as hair segmentation in a face [28], or detecting the targets which are consistently small in the input images [38].

4. Pancreas Segmentation Experiments

4.1. Dataset and Evaluation

We evaluate our approach on the NIH pancreas segmentation dataset [34], which contains 82 contrast-enhanced abdominal CT volumes. The resolution of each scan is $512 \times 512 \times L$, where $L \in [181, 466]$ is the number of slices along the long axis of the body. The distance between neighboring voxels ranges from 0.5mm to 1.0mm.

Following the standard cross-validation strategy, we split the dataset into 4 fixed folds, each of which contains approximately the same number of samples. We apply cross validation, i.e., training the models on 3 out of 4 subsets and testing them on the remaining one. We measure the segmentation accuracy by computing the Dice-Sørensen coefficient (DSC) for each sample, and report the average and standard deviation over all 82 cases.

4.2. Different Settings

We use the FCN-8s model [27] pre-trained on PascalVOC [9]. We initialize the up-sampling layers with random weights, set the learning rate to be $10^{-4}$ and run 80,000 iterations. Different options are evaluated, including using different kernel sizes in saliency transformation, and whether to fine-tune the models using the predicted segmentations as reference maps (see the description in Section 3.3). Quantitative results are summarized in Table 1.

As the saliency transformation module is implemented
<table>
<thead>
<tr>
<th>Model</th>
<th>Average</th>
<th>Gain</th>
<th>Max</th>
<th>Min</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage-wise segmentation [46]</td>
<td>82.37 ± 5.68</td>
<td>−</td>
<td>90.85</td>
<td>62.43</td>
</tr>
<tr>
<td>Using 3 × 3 kernels in saliency transformation (basic model)</td>
<td>83.47 ± 5.78</td>
<td>+0.00</td>
<td>90.63</td>
<td>57.85</td>
</tr>
<tr>
<td>Using 1 × 1 kernels in saliency transformation</td>
<td>82.85 ± 6.68</td>
<td>−0.62</td>
<td>90.40</td>
<td>53.44</td>
</tr>
<tr>
<td>Using 5 × 5 kernels in saliency transformation</td>
<td>83.64 ± 5.29</td>
<td>+0.17</td>
<td>90.35</td>
<td>66.35</td>
</tr>
<tr>
<td>Two-layer saliency transformation (3 × 3 kernels)</td>
<td>83.93 ± 5.43</td>
<td>+0.46</td>
<td>90.52</td>
<td>64.78</td>
</tr>
<tr>
<td>Fine-tuning with noisy data (3 × 3 kernels)</td>
<td>83.99 ± 5.09</td>
<td>+0.52</td>
<td>90.57</td>
<td>65.05</td>
</tr>
</tbody>
</table>

Table 1. Accuracy (DSC, %) comparison of different settings of our approach. Please see the texts in Section 4.2 for detailed descriptions of these variants. For each variant, the “gain” is obtained by comparing its accuracy with the basic model.

by a size-preserved convolution (see Section 3.2), the size of convolutional kernels determines the range that a pixel can use to judge its saliency. In general, a larger kernel size improves segmentation accuracy (3 × 3 works significantly better than 1 × 1), but we observe the marginal effect: the improvement of 5 × 5 over 3 × 3 is limited. As we use 7 × 7 kernels, the segmentation accuracy is slightly lower than that of 5 × 5. This may be caused by the larger number of parameters introduced to this module. Another way of increasing the receptive field size is to use two convolutional layers with 3 × 3 kernels. This strategy, while containing a smaller number of parameters, works even better than using one 5 × 5 layer. But, we do not add more layers, as the performance saturates while computational costs increase.

As described in Section 3.3 we fine-tune these models with images cropped from the coarse-scaled segmentation mask. This is to adjust the models to the testing phase, in which the ground-truth mask is unknown, so that the fine-scaled segmentation needs to start with, and be able to revise the coarse-scaled segmentation mask. We use a smaller learning rate (10^-6) and run another 40,000 iterations. This strategy not only reports 0.52% overall accuracy gain, but also alleviates over-fitting (see Section 4.4.3).

In summary, all these variants produce higher accuracy than the state-of-the-art (82.37% by [46]), which verifies that the major contribution comes from our recurrent framework which enables joint optimization. In the later experiments, we inherit the best variant learned from this section, including in a large-scale multi-organ dataset (see Section 3). That is to say, we use two 3 × 3 convolutional layers for saliency transformation, and fine-tune the models with coarse-scaled segmentation. This setting produces an average accuracy of 84.50%, as shown in Table 2.

### 4.3. Comparison to the State-of-the-Art

We show that our approach works better than the baseline, i.e., the coarse-to-fine approach with two stages trained individually [46]. As shown in Table 2, the average improvement over 82 cases is 2.13 ± 2.67%, which is impressive given such a high baseline accuracy (82.37% is already the state-of-the-art). The standard deviations (5.68% of [46] and 4.97% of ours) are mainly caused by the difference in scanning and labeling qualities. The student’s t-test suggests statistical significance (p = 3.62 × 10^-8). A case-by-case study reveals that our approach reports higher accuracies on 67 out of 82 cases, with the largest advantage being +17.60% and the largest deficit being merely −3.85%. We analyze the sources of improvement in Section 4.4.

Another related work is [44] which stacks two FCN’s for segmentation. Our work differs from it by (i) our model is recurrent, which allows fine-scaled segmentation to be updated iteratively, and (ii) we crop the input image to focus on the salient region. Both strategies contribute significantly to segmentation accuracy. Quantitatively, [44] reported an average accuracy of 77.89%. Our approach achieves 78.23% in the coarse stage, 82.73% after only one iteration, and an entire testing phase reports 84.50%.

We briefly discuss the advantages and disadvantages of using 3D networks. 3D networks capture richer contextual information, but also require training more parameters. Our 2D approach makes use of 3D contexts more efficiently. At the end of each iteration, predictions from three views are fused, and thus the saliency transformation module carries these information to the next iteration. We implement VNet [30], and obtain an average accuracy of 83.18% with a 3D ground-truth bounding box provided for each case. Without the ground-truth, a sliding-window process is required which is really slow – an average of 5 minutes on a Titan-X Pascal GPU. In comparison, our approach needs 1.3 minutes, slower than the baseline [46] (0.9 minutes), but faster than other 2D approaches [34][35] (2–3 minutes).
contribute to visual recognition [47].

This reveals region. Similar phenomena happen in the models trained on different information. It is interesting to see that in saliency presentation worked even worse because the missing context.

In Figure 5, we visualize show how the recurrent network assists segmentation by incorporating multi-stage visual cues. This is a failure case by the baseline approach [46] (see Figure 2), in which fine-scaled segmentation worked even worse because the missing contextual information. It is interesting to see that in saliency transformation, different channels deliver complementary information, i.e., two of them focus on the target organ, and the remaining one adds most weights to the background region. Similar phenomena happen in the models trained in different viewpoints and different folds. This reveals that, except for foreground, background and boundary also contribute to visual recognition [47].

4.4. Diagnosis

4.4.1 Joint Optimization and Mutli-Stage Cues

Our approach enables joint training, which improves both the coarse and fine stages individually. We denote the two networks trained in [46] by $\mathbb{I}^C$ and $\mathbb{I}^F$, and similarly, those trained in our approach by $\mathbb{I}^{cC}$ and $\mathbb{I}^{CF}$, respectively. In the coarse stage, $\mathbb{I}^C$ reports 75.74% and $\mathbb{I}^{cC}$ reports 78.23%. In the fine stage, applying $\mathbb{I}^F$ on top of the output of $\mathbb{I}^C$ gets 83.80%, which is considerably higher than 82.37% ($\mathbb{I}^F$ on top of $\mathbb{I}^C$) but lower than 84.50% ($\mathbb{I}^F$ on top of $\mathbb{I}^{cC}$). Therefore, we conclude that both the coarse-scaled and fine-scaled networks benefit from joint optimization. A stronger coarse stage provides a better starting point, and a stronger fine stage improves the upper-bound.

In Figure 5, we visualize show how the recurrent network assists segmentation by incorporating multi-stage visual cues. This is a failure case by the baseline approach [46] (see Figure 2), in which fine-scaled segmentation worked even worse because the missing contextual information. It is interesting to see that in saliency transformation, different channels deliver complementary information, i.e., two of them focus on the target organ, and the remaining one adds most weights to the background region. Similar phenomena happen in the models trained in different viewpoints and different folds. This reveals that, except for foreground, background and boundary also contribute to visual recognition [47].

4.4.2 Convergence

We study convergence, which is a very important criterion to judge the reliability of our approach. We choose the best model reporting an average accuracy of 84.50%, and record the inter-iteration DSC throughout the testing process:

$$d^{(t)} = \text{DSC}\{Z^{(t-1)}, Z^{(t)}\} = \frac{2 \times \sum_{i} Z_{i}^{(t)} Z_{i}^{(t-1)} + \sum_{i} Z_{i}^{(t)} + \sum_{i} Z_{i}^{(t-1)} + \sum_{i} Z_{i}^{(t)} Z_{i}^{(t-1)}}{\sum_{i} Z_{i}^{(t)} + \sum_{i} Z_{i}^{(t-1)} + \sum_{i} Z_{i}^{(t)} + \sum_{i} Z_{i}^{(t)} Z_{i}^{(t-1)}}$$.

After 1, 2, 3, 5 and 10 iterations, these numbers are 0.9037, 0.9677, 0.9814, 0.9908 and 0.9964 for our approach, and 0.8286, 0.9477, 0.9661, 0.9743 and 0.9774 for [46], respectively. Each number reported by our approach is considerably higher than that by the baseline. The better convergence property provides us with the opportunity to set a more strict terminating condition, e.g., using \( \text{thr} = 0.99 \) rather than \( \text{thr} = 0.95 \).

We note that [46] also tried to increase the threshold from 0.95 to 0.99, but only 3 out of 82 cases converged after 10 iterations, and the average accuracy went down from 82.37% to 82.28%. In contrary, when the threshold is increased from 0.95 to 0.99 in our approach, 80 out of 82 cases converge (in an average of 5.22 iterations), and the average accuracy is improved from 83.93% to 84.50%. In addition, the average number of iterations needed to achieve \( \text{thr} = 0.95 \) is also reduced from 2.89 in [46] to 2.02 in our approach. On a Titan-X Pascal GPU, one iteration takes 0.2 minutes, so using \( \text{thr} = 0.99 \) requires an average of 1.3 minutes in each testing case. In comparison, [46] needs an average of 0.9 minutes and [35] needs 2-3 minutes.
4.4.3 The Over-Fitting Issue

Finally, we investigate the over-fitting issue by making use of oracle information in the testing process. We follow [46] to use the ground-truth bounding box on each slice, which is used to crop the input region in every iteration. Note that annotating a bounding box in each slice is expensive and thus not applicable in real-world clinical applications. This experiment is aimed at exploring the upper-bound of our segmentation networks under perfect localization.

With oracle information provided, our best model reports 86.37%, which is considerably higher than the number (84.50%) without using oracle information. If we do not fine-tune the networks using coarse-scaled segmentation (see Table 1), the above numbers are 86.26% and 83.68%, respectively. This is to say, fine-tuning prevents our model from relying on the ground-truth mask. It not only improves the average accuracy, but also alleviates over-fitting (the disadvantage of our model against that with oracle information is decreased by 0.67%).

5. Multi-Organ Segmentation Experiments

To verify that our approach can be applied to other organs, we collect a large dataset which contains 200 CT scans, 11 abdominal organs and 5 blood vessels. This corpus took 4 full-time radiologists around 3 months to annotate. To the best of our knowledge, this dataset is larger and contains more organs than any public datasets. We choose 5 most challenging targets including the pancreas and a blood vessel, as well as two kidneys which are relatively easier. Other easy organs such as the liver are ignored. To the best of our knowledge, some of these organs were never investigated before, but they are important in diagnosing pancreatic diseases and detecting the pancreatic cancer at an early stage. We randomly partition the dataset into 4 folds for cross validation. Each organ is trained and tested individually. When a pixel is predicted as more than one organs, we choose the one with the largest confidence score.

Table 3. Comparison of coarse-scaled (C) and fine-scaled (F) segmentation by [46] and our approach on our own dataset. A fine-scaled accuracy is indicated by ‡ if it is lower than the coarse-scaled one. The pancreas segmentation accuracies are higher than those in Table 2 due to the increased number of training samples and the higher resolution in CT scans.

<table>
<thead>
<tr>
<th>Organ</th>
<th>[46]-C</th>
<th>[46]-F</th>
<th>Ours-C</th>
<th>Ours-F</th>
</tr>
</thead>
<tbody>
<tr>
<td>adrenal g.</td>
<td>57.38</td>
<td>61.65</td>
<td>60.70</td>
<td>63.76</td>
</tr>
<tr>
<td>duodenum</td>
<td>67.42</td>
<td>69.39</td>
<td>71.40</td>
<td>73.42</td>
</tr>
<tr>
<td>gallbladder</td>
<td>82.57</td>
<td>82.12</td>
<td>87.08</td>
<td>87.10</td>
</tr>
<tr>
<td>inferior v.c.</td>
<td>71.77</td>
<td>71.15</td>
<td>79.12</td>
<td>79.69</td>
</tr>
<tr>
<td>kidney l.</td>
<td>92.56</td>
<td>92.78</td>
<td>96.08</td>
<td>96.21</td>
</tr>
<tr>
<td>kidney r.</td>
<td>94.98</td>
<td>95.39</td>
<td>95.80</td>
<td>95.97</td>
</tr>
<tr>
<td>pancreas</td>
<td>83.68</td>
<td>85.79</td>
<td>86.09</td>
<td>87.60</td>
</tr>
</tbody>
</table>

Results are summarized in Table 3. We first note that [46] sometimes produced a lower accuracy in the fine stage than in the coarse stage. Apparently this is caused by the unsatisfying convergence property in iterations, but essentially, it is the loss of contextual information and the lack of globally optimized energy function. Our approach solves this problem and reports a 4.29% average improvement over 5 challenging organs (the kidneys excluded). For some organs, e.g., the gallbladder, we do not observe significant accuracy gain by iterations. But we emphasize that in these scenarios, our coarse stage already provides much higher accuracy than the fine stage of [46], and the our fine stage preserves such high accuracy through iterations, demonstrating stability. An example is displayed in Figure 6.

6. Conclusions

This work is motivated by the difficulty of small organ segmentation. As the target is often small, it is required to focus on a local input region, but sometimes the network is confused due to the lack of contextual information. We present the Recurrent Saliency Transformation Network, which enjoys three advantages. (i) Benefited by a (re)global energy function, it is easier to generalize our models from training data to testing data. (ii) With joint optimization over two networks, both of them get improved individually. (iii) By incorporating multi-stage visual cues, more accurate segmentation results are obtained. As the fine stage is less likely to be confused by the lack of contexts, we also observe better convergence during iterations.

Our approach is applied to two datasets for pancreas segmentation and multi-organ segmentation, and outperforms the baseline (the state-of-the-art) significantly. Confirmed by the radiologists in our team, these segmentation results are helpful to computer-assisted clinical diagnoses.

Acknowledgements: This paper was supported by the Lustgarten Foundation for Pancreatic Cancer Research. We thank Wei Shen, Seyoun Park, Weichao Qiu, Song Bai, Zhuotun Zhu, Chenxi Liu, Yan Wang, Siyuan Qiao, Yingda Xia and Fengze Liu for discussions.
References


