This CVPR 2020 paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Iteratively-Refined Interactive 3D Medical Image Segmentation with Multi-Agent Reinforcement Learning

Xuan Liao[1], Wenhao Li*[2], Qisen Xu*[2], Xiangfeng Wang[2]✉,
Bo Jin[2]✉, Xiaoyun Zhang[1], Yanfeng Wang[1], and Ya Zhang[1]✉

[1] Cooperative Medianet Innovation Center, Shanghai Jiao Tong University
[2] Multi-agent Artificial Intelligence Laboratory, East China Normal University

{liaoxuan, xiaoyun.zhang, wangyanfeng, ya_zhang} @sjtu.edu.cn, {51164500105@stu,
51184501067@stu, xfwang@sei, bjin@cs} .ecnu.edu.cn

Abstract

Existing automatic 3D image segmentation methods usually fail to meet the clinic use. Many studies have explored an interactive strategy to improve the image segmentation performance by iteratively incorporating user hints. However, the dynamic process for successive interactions is largely ignored. We here propose to model the dynamic process of iterative interactive image segmentation as a Markov decision process (MDP) and solve it with reinforcement learning (RL). Unfortunately, it is intractable to use single-agent RL for voxel-wise prediction due to the large exploration space. To reduce the exploration space to a tractable size, we treat each voxel as an agent with a shared voxel-level behavior strategy so that it can be solved with multi-agent reinforcement learning. An additional advantage of this multi-agent model is to capture the dependency among voxels for segmentation task. Meanwhile, to enrich the information of previous segmentations, we reserve the prediction uncertainty in the state space of MDP and derive an adjustment action space leading to a more precise and finer segmentation. In addition, to improve the efficiency of exploration, we design a relative cross-entropy gain-based reward to update the policy in a constrained direction. Experimental results on various medical datasets have shown that our method significantly outperforms existing state-of-the-art methods, with the advantage of fewer interactions and a faster convergence.

1. Introduction

Medical image segmentation has been widely recognized as an essential procedure for subsequent medical image processes such as structural and functional analysis, diagnosis and treatment. The traditional dense manual annotation is extremely inefficient for 3D medical images and its performance highly depends on the physician's experience. With the development of convolutional neural networks (CNNs), automatic segmentation has greatly improved the efficiency of medical image process [13, 16, 23]. However, the accuracy and robustness for the current automatic methods still need to be improved for practical clinic use.

To get a better segmentation, interactive image segmentation [5, 7, 20] is introduced to integrate user hints (mostly in the form of points, scribbles and bounding boxes). This kind of interactive methods has become a popular research direction because it improves the performance of segmentation by adding new labeling constraints to the prediction model. Normally, a one-time interaction might not ensure the segmentation accuracy. Therefore, many existing methods are compatible with the iteratively-refined mode: the operator provides new hints according to the current result to refine the segmentation until it is satisfactory. Moreover, to reduce the number of interactions, the existing works introduce the idea of replacing the initial hints with an automatically-obtained coarse segmentation [5, 20]. Note that in this paper, we refer to such methods incorporating a coarse segmentation in the initial input as ***update methods*** and we will focus on this kind of methods in this paper.

Concerning the current update methods, there exist two main issues: 1) They usually ignore the dynamic process for successive interactions. Although the segmentation can be iteratively refined, the model always treat the segmentation of each refinement step in isolation, with an absence of the previous information. 2) Another problem is the loss of prediction uncertainty when using the binary segmentation result, instead of a segmentation probability for each voxel, as part of the model input. The rounding from dense segmentation probability to binary segmentation prediction may cause quantization error and accuracy loss.

To tackle the above two issues, this paper proposes a novel interactive medical image segmentation update method called *Iteratively-Refined interactive 3D medical image segmentation via Multi-agent Reinforcement Learning (IteR-MRL)*. We formulate the dynamic process of iterative interactive image segmentation as an MDP. Specifically, at each refinement step, the model needs to decide the labels of all voxels, according to the previous segmenta-

*Equal contribution

tions and supervision information from the interaction. After that, the model will get the feedback according to predefined measurement of segmentation, and the above process will be repeated until the maximum number of interactions is reached. We then adopt the RL methods to solve the MDP above, that is, to find the segmentation strategy to maximize the accumulated rewards received at each refinement step. However, it will be intractable to use single-agent RL for voxel-wise prediction due to the large exploration space. In addition, considering that the voxels in the segmentation task are interdependent, they can achieve better segmentation by a more comprehensive grasp of the surrounding information. To reduce the exploration space to a tractable size and explicitly model the dependencies between voxels, we introduce the multi-agent reinforcement learning (MARL) method. We treat each voxel as an agent which decides its own label. All agents share the same policy and collaborate with each other through convolutional kernels. Meanwhile, instead of considering the difference between the current prediction and the ground truth, we design a relative cross-entropy gain-based reward to prompt agents to explore more efficiently. Specifically, the algorithm gives a positive reward for an improvement and vice versa at each refinement step, so that the new prediction can be forced to outperform the previous one. Compared with supervised methods, such RL-based methods have the advantage of a faster refinement convergence. The problem of prediction uncertainty loss in existing works caused by segmentation map binarization can be settled by adopting segmentation probability rather than binary segmentation as part of the RL state. This augmented state space also derives an adjustment action space leading to a more precise and finer segmentation. Then the segmentation refinement procedure can be regarded as a series of actions to adjust the segmentation probability with a certain level. In this way, the prediction uncertainty is reserved and the algorithm explores in a finer granularity and a denser space.

The experimental results indicate that the proposed IteR-MRL is robust to different initial segmentations and various medical datasets. Given the same initial segmentations, our proposed interactive algorithm surpasses the state-of-the-art update methods on different 3D medical image segmentation datasets including the images of brain tumor, heart and prostate. We summarize our contributions as follows:

- We formulate the interactive image segmentation task as an MDP and propose a novel voxel-wise interactive segmentation framework based on MARL for 3D medical images, enabling more effective utilization of user interaction.
- We propose to reserve the prediction uncertainty via the segmentation probability, which can enrich the information of previous segmentations and lead to a more precise and finer adjustment.
- Extensive experiments show that the segmentation is significantly improved over the iteration sequence with only a few interactions and a rapid convergence, by considering the relative gain between two successive steps.

2. Related work

Interactive image segmentation has been widely applied to both natural [4, 22] and medical images [15, 19, 20]. "Interactive" refers that the operator provides some hints to the segmentation model to achieve a better result. This section will briefly review the existing works.

2.1. Graph-based interactive image segmentation

Traditional methods make use of low-level features such as the histogram and similarities between pixels. GraphCut [4] and GrabCut [17] incorporate user hints into Max-Flow Min-Cut algorithm[3]. DenseCRF [10] considers pixel relations from neighbors to all pixel pairs. [8] proposes to use geodesic distance to calculate the distance between pixels, which is sensitive to contrast and suitable for medical images. [21] introduces a segmentation method for fetal MRI by learning from user annotations in only one slice.

2.2. CNN-based interactive image segmentation

Recently, using convolutional neural networks (CNNs) has become popular for computer vision problems. Many CNN-based methods have developed for interactive image segmentation tasks. [22] is the first one to use CNN in interactive image segmentation. [15] replaces the Gaussian mixture model in GrabCut with a CNN for MRI segmentation. Another work, 3D U-Net [7] learns to produce a complete segmentation from sparsely-annotated slices of one 3D medical image. In order to save the budget of the initial user hints, the following methods, known as update methods, choose to take an automatically-produced segmentation as part of the model initial input. [20] proposes a two-stage method called DeepIGeoS to refine the segmentation using the initial coarse segmentation in input. However, the refined segmentations after the first step cannot be efficiently used in this model. [5] extends DeepIGeoS to an iterative version: Inter-CNN, which iteratively refines the previous refined binary prediction in both training and testing stages. One of their problems is the ignorance of the dynamic process for successive interactions. Another problem is the accuracy loss caused by the quantization from probability to binary segmentation.

2.3. RL-based interactive image segmentation

There are also some methods using RL to explicitly model the dynamic process in interactive image segmentation tasks. SeedNet [18] uses an RL agent to simulate

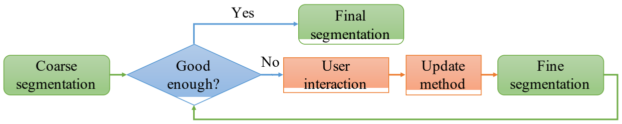

Figure 1. The flow chart of iteratively-refined interactive image segmentation approach. Given a coarse segmentation, the method iteratively refines it with user interaction until the fine segmentation is good enough.

the user behavior which gives the hints to the segmentation model. Since our method uses RL as the segmentation model to predict the segmentation, our method is orthogonal with it. Polygon-RNN [6] identifies the object segmentation as a polygon. Their model produces vertices sequentially until the polygon is closed. The user can contribute by adjusting the vertices. Based on this work, Polygon-RNN++ [1] develops a faster and more accurate algorithm by combining RL with graph neural network. However, these polygon-based methods cannot be applied to our tasks because of the incompatibility of the 3D images with polygon segmentation, and the extremely large action space even with the meshing strategy.

3. Methodology

In this section, we formulate the interactive image segmentation as an MDP and propose a novel MARL-based interactive medical image segmentation method to exploit the interaction information more efficiently.

3.1. Overview

In our work, we propose an iteratively-refined framework based on update methods, as shown in Fig. 1, which iteratively refines a coarse initial segmentation by integrating user interactions in order to get a more precise segmentation result. The initial segmentation can be obtained from any accessible segmentation methods.

As discussed in Sections 1 and 2, the main issue of the existing supervised learning-based algorithms is that they split the whole image refinement process into isolated steps. To address this problem, we adopt RL to explicitly capture the relation between successive predictions by designing the reward as the relative improvement. As the large state space and action space of voxel-wise prediction and the necessity of the collaborations between interdependent voxels, we use the idea of MARL: each voxel in a 3D image is regarded as an agent. The work PixelRL [9] also sees each pixel as an agent, but it focuses on general image processing tasks without human interaction. In contrast, the interactive image segmentation task is more suitable to adopt RL due to its intrinct sequentiality. Unlike dealing with non-interactive image processing tasks, we aim to better consider and effectively utilize external supervision signal from the user during the interaction.

Fig. 2 introduces the framework of our proposed method IteR-MRL. By utilizing the original 3D image, the previous segmentation probability and the interactive information as the state, the actor network in the middle gives an update to the segmentation probability and produces a new one. Note that the previous segmentation probability comes from the previous update iteration and the interactive information is the hint map transformed from user hints which will be introduced in Subsection 3.2. The actor network outputs actions of agents which adjust previous segmentation probability and generate current segmentation probability. Afterwards, there are two subsequent operations for the current segmentation probability. On the one side, it gives back a reward signal to the network, by computing the relative gain between previous and current cross entropy based on ground truth and successive segmentation probabilities, for parameter updating. On the other side, it is presented to the user and the user provides feedback, *i.e.* clicks on object or background for wrongly predicted areas. The click is represented as a red point on Fig. 2, which is enlarged for visualization. Generally, with a coarse segmentation probability produced by the initial method (any segmentation method) as its initial segmentation, IteR-MRL iteratively refines the segmentation probability until the segmentation is satisfactory. In addition, the actor network employed here is designed for MARL and it regards voxels on the 3D image as agents who collaborates with each other.

It should be noticed that instead of quantizing the segmentation probability to binary segmentation prediction like previous methods [5, 20], we directly use the segmentation probability as the previous segmentation information and feed it into the model. The segmentation probability is introduced to enrich the previous segmentation information and achieve more accurate results. With the segmentation probability, we can derive an adjustment action space leads to a more precise and finer segmentation comparing with the binary segmentation quantization. Specifically, we can adjust the segmentation probability at each step and choose the best adjustment magnitude from a set of various scales. The adjustment action for one agent in MARL model is based on both its own and neighbors' states, where neighbors are considered as near voxels.

3.2. Multi-agent RL framework for interactive image segmentation

In this subsection, we describe the MARL setting for interactive image segmentation. Let $\boldsymbol{x} = (x_1, \cdots, x_N)$ be one arbitrary image in the dataset and x_i is the i_{th} voxel of $\boldsymbol{x}$. We treat each x_i as an agent whose policy is defined as $\pi_i(a_i^{(t)}|s_i^{(t)})$. $s_i^{(t)}$ and $a_i^{(t)}$ are the state (image, previous segmentation probability, user interaction) and action (adjustment to previous probability) for x_i at the step t; $a_i^{(t)} \in \mathcal{A}$ and $\mathcal{A}$ is the action set; $s_i^{(t)} \in \mathcal{S}$ and $\mathcal{S}$ is the state set. By using convolutional kernels, one agent can access to

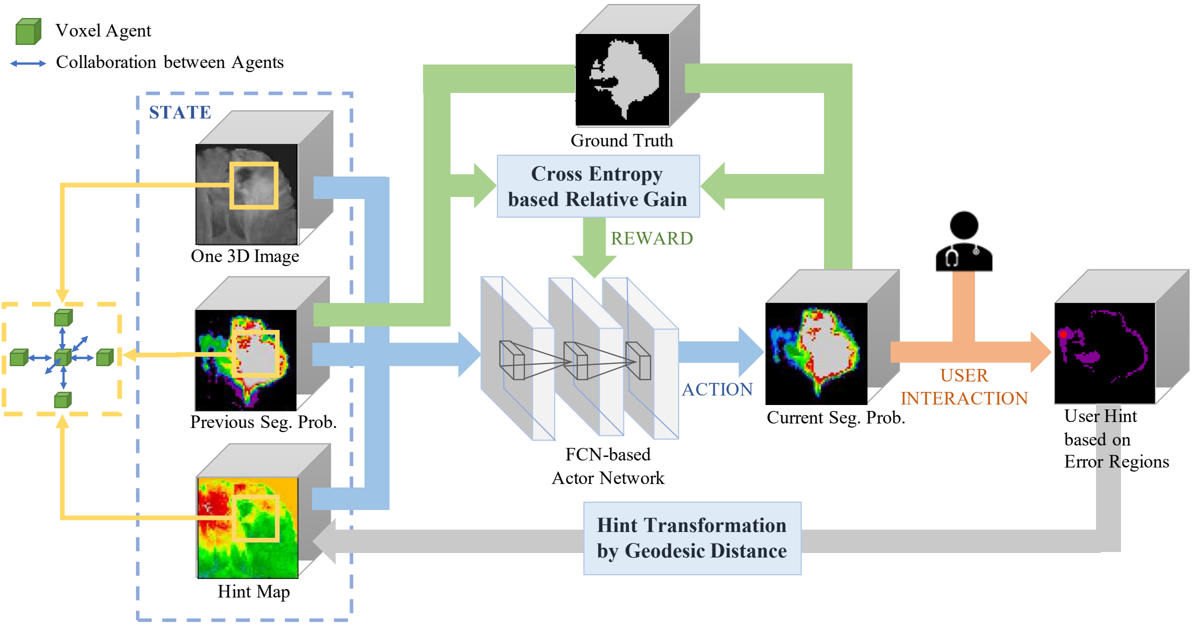

Figure 2. Overview of Iteratively-Refined interactive 3D medical image segmentation algorithm based on MARL (IteR-MRL). At each refinement step, the state containing image, previous segmentation probability and the hint map is feeded into the actor network, then the actor network produces current segmentation probability derived by its output actions. Next, the user gives back hint clicks (the red point) based on error regions and new hint map is generated by hint transformation. At every step, the reward is determined by the relative gain between previous and current segmentation cross entropy. Voxels are regarded as agents who collaborate with each other in our method.

its neighbors' states as well.

From the point of view of the whole image, the previous segmentation is refined to a new one. By taking the global action $\boldsymbol{a}^{(t)} = (a_1^{(t)}, \cdots, a_N^{(t)})$, the image agent transfers to the global state $\boldsymbol{s}^{(t+1)} = (s_1^{(t+1)}, \cdots, s_N^{(t+1)})$ and gets the global reward $\boldsymbol{r}^{(t)} = (r_1^{(t)}, \cdots, r_N^{(t)})$.

We now define the state, action and reward of a single agent x_i in IteR-MRL.

State. For our problem formulation, the state for voxel agent x_i at the step t is the concatenation of its voxel value b_i, its previous segmentation probability $p_i^{(t)}$ to be object label and its two values on hint maps $h_{+,i}^{(t)}$ and $h_{-,i}^{(t)}$: $s_i^{(t)} = [b_i, p_i^{(t)}, h_{+,i}^{(t)}, h_{-,i}^{(t)}]$ with $p_i^{(t)} \in [0,1]$. For the initial state $s_i^{(0)}$, the initial coarse segmentation probability denotes initial probability $p_i^{(0)}$.

Now we discuss the generation of a whole hint map. Concerning the user interaction at step t, the hint map $\boldsymbol{h}^{(t)}$ is transformed from the user's hints which are in the form of click points. By giving a hint point through a single click, the user indicates that the area around it is one error region. Intuitively, the closer one point is to the hint point, the more likely its label is mispredicted. Hence, the hint map is introduced to show the radiation area of the hint and spread the local interaction to the whole image. The number and positions of hints are chosen according to the user interaction rule. Actually, there are two channels of hint map both with the same size as the image: the object hint map $\boldsymbol{h}_+^{(t)}$ and the background hint map $\boldsymbol{h}_-^{(t)}$, respectively generated from the object hint set $hs_+^{(t)}$ (hints on object) and the background hint set $hs_-^{(t)}$ (hints on background). Hence, the user hint map is the concatenation of these two hint maps: $\boldsymbol{h}^{(t)} = [\boldsymbol{h}_+^{(t)}, \boldsymbol{h}_-^{(t)}]$. For one hint map $\boldsymbol{h}_\ell^{(t)}$, $\ell \in \{+,-\}$, we define that $\boldsymbol{h}_\ell^{(t)} = (h_{\ell,1}^{(t)}, \cdots, h_{\ell,N}^{(t)})$. The element $h_{\ell,i}^{(t)}$ on the hint map $\boldsymbol{h}_\ell^{(t)}$ is calculated by the minimum distance between x_i and the corresponding hint set $hs_\ell^{(t)}$:

$$h_{\ell,i}^{(t)} = \min_{\forall x_j \in hs_\ell^{(t)}} \mathcal{M}(x_i, x_j), \tag{1}$$

where $\mathcal{M}$ is a function to measure the distance between two voxels. Previous related works adopt several distance-measuring methods including geodesic [8], Gaussian and Euclidean distance. In the paper, we use the geodesic distance-based hint map to measure distances. The distance between two voxels is the minimum value of the sum of color gradients across all the paths connecting these two voxels. (See the hint map in Fig. 2.)

Action. While previous works [5, 20] output directly the segmentation probability from the network, we here predict the adjustment amount based on the previous probability as actions to make the result more stable without abrupt changes. The action $a_i^{(t)} \in \mathcal{A}$ for x_i at time step t is to adjust the previous segmentation probability $p_i^{(t)}$ by a certain amount $a_i^{(t)}$. Hence, the segmentation probability $p_i^{(t+1)}$ after taking the action $a_i^{(t)}$ is:

$$p_i^{(t+1)} \;=\; \mathcal{C}_0^1(p_i^{(t)} + a_i^{(t)}), \tag{2}$$

where $\mathcal{C}_a^b(x) = \min(\max(x, a), b)$ clips the value of x from a to b. $p_i^{(t+1)}$ is constrained to $[0, 1]$ for it represents a probability. The action set $\mathcal{A} = \{\mathcal{A}_k\}$ $(k = 1, 2, \cdots, K)$ contains K actions, allowing the agent to adjust the probability to various degrees under different situations. For example, it is reasonable to make a larger adjustment to a voxel when it is closer to a hint click. Additionally, one voxel tends to take one certain adjustment action when most of its neighbor voxels choose this action.

Reward. To improve the efficiency of exploration, we design a relative cross-entropy gain-based reward to update the model in a constrained direction. Specifically, the reward is designed as the relative improvement from the previous segmentation to the current one, which is the decreased amount of the cross entropy $\mathcal{X}_i$ between the ground truth y_i and the segmentation probability p_i:

$$r_i^{(t)} \;=\; \mathcal{X}_i^{(t-1)} - \mathcal{X}_i^{(t)}, \tag{3}$$

$$\mathcal{X}_i^{(t)} \;=\; -y_i \log(p_i^{(t)}) - (1 - y_i) \log(1 - p_i^{(t)}). \tag{4}$$

With (3), the agent gets a positive reward in the case its probability moves closer towards the true voxel label and vice versa. Instead of a distant goal, the relative gain provides the agent with a baseline to compare and surpass.

In general, the accumulated reward of one interactive sequence is

$$R_i \;=\; \textstyle\sum_{t=1}^{T} \gamma^{t-1} r_i^{(t)}, \tag{5}$$

where T is the total step number and the discount factor γ takes a value in $(0, 1]$.

3.3. Network and training

For fair comparison, the interactive network architecture of [20] named R-net is adopted as the backbone to our algorithm and all other baseline methods. We adapt the network to the one in Fig. 3 in order to fit the RL training algorithm: asynchronous advantage actor-critic (A3C) [14]. The network firstly uses three 3D convolutional blocks to extract low-level features. Then, the network is divided into policy head and value head, both have three 3D convolutional blocks to extract specific high-level features. The functionality of the policy head is to predict the distribution of action probabilities under a known state. In our case, given the image, hint maps and previous segmentation probability, the policy head predicts how likely it is to take each scale of adjustment to the previous segmentation probability. The functionality of the value head is to estimate the value of the current state. Specifically, the value head evaluates how good the current combination of the image, hint maps and the previous segmentation probability is.

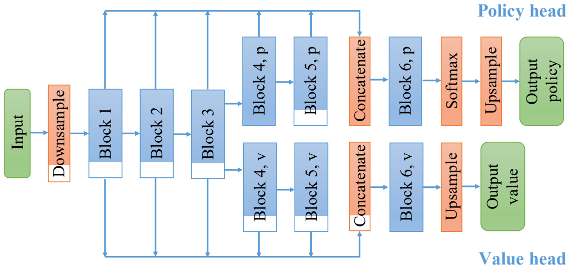

Figure 3. The network architecture for IteR-MRL. The policy and value heads share the low-level features and extract their own high-level features.

We respectively use θ_p and θ_v to denote the parameters of the policy and value heads. The input of the network is the state at time step t: $\boldsymbol{s}^{(t)}$. The value head outputs the estimated value of the current state $V(\boldsymbol{s}^{(t)})$. The gradient for θ_v is computed by:

$$d\theta_v \;=\; \nabla_{\theta_v} A(\boldsymbol{s}^{(t)}, \boldsymbol{a}^{(t)})^2, \tag{6}$$

$$A(\boldsymbol{s}^{(t)}, \boldsymbol{a}^{(t)}) \;=\; \textstyle\sum_{k=t}^{T} \gamma^{k-t} \overline{r}^{(k)} - V(\boldsymbol{s}^{(t)}), \tag{7}$$

where $\overline{r}^{(k)}$ is the mean reward of all voxels at time step k. $A(\boldsymbol{s}^{(t)}, \boldsymbol{a}^{(t)})$ is the advantage at time step t of taking $\boldsymbol{a}^{(t)}$ in condition of state $\boldsymbol{s}^{(t)}$, which indicates the actual accumulated reward without being affected by the state and reduces the variance of gradient. The policy head outputs the action policy $\pi(\boldsymbol{a}^{(t)}|\boldsymbol{s}^{(t)})$, which is the probabilities of taking each action $\boldsymbol{a}^{(t)}$. The gradient for θ_p is computed by:

$$d\theta_p \;=\; -\nabla_{\theta_p} \pi(\boldsymbol{a}^{(t)}|\boldsymbol{s}^{(t)}) A(\boldsymbol{a}^{(t)}, \boldsymbol{s}^{(t)}). \tag{8}$$

The two heads are jointly trained in an end-to-end manner.

4. Experiments

4.1. Datasets

In our paper, we do experiments on three 3D MRI datasets. Each image is cropped based on its non-zero region before used. For each dataset, we access all the image cases having ground truth and split them into several sets. The initial method is defined as the segmentation method of producing the initial coarse segmentations. If we use the images trained on initial method again in update method, the coarse segmentation probability (initial segmentation probability for update method) will be too perfect to be refined in update method because these images have already seen the ground truth in initial method. Therefore, we propose a new way for the dataset splitting: the dataset is split into three parts, two training sets with an equal amount of images and one testing set. In detail, we randomly selected N_{train} cases as the training set for initial method forming D_{train1} and randomly selected another N_{train} cases in the remaining dataset as the training set for update method forming D_{train2}. The remaining N_{test} cases are used as testing forming D_{test}. Note that the initial segmentation probabilities data for D_{train2} in update method are obtained by testing D_{train2} with initial method. The three datasets are as follows:

BraTS2015. Brain Tumor Segmentation Challenge 2015 (BraTS) [12] provides a dataset for brain tumor segmentation in magnetic resonance images. We use Fluid-attenuated Inversion Recovery (FLAIR) images which contain 274 cases and only segment the whole brain tumor. We set N_{train} as 117 and N_{test} as 40.

MM-WHS. Multi-Modality Whole Heart Segmentation (MM-WHS) [24] contains multi-modality whole heart images covering the whole heart substructures. We use the 20 MRI cases and segment the left atrium blood cavity. We set N_{train} as 8 and N_{test} as 4.

NCI-ISBI2013. NCI-ISBI 2013 Challenge [2] aims at automated segment prostate structures. It provides 80 prostate gland MRI data. We set N_{train} as 32 and N_{test} as 16.

4.2. Settings

Evaluation metrics. Normally, medical image segmentation is evaluated by the dice score:

$$Dice(S_p, S_g) = \frac{2|S_p \cap S_g|}{|S_p| + |S_g|}, \tag{9}$$

where S_p, S_g represent the prediction and the ground truth respectively. $|\cdot|$ is the number of voxels in the area.

As we study the interactive image segmentation task, we consider not only the dice score but also the user click number. Our goal is to get a high dice score with less user clicks.

User simulation. Since it would require large human resources to conduct the experiments with real physicians, we simulate user clicks like other works. While previous works usually give many clicks (≈ 40) for training but a few clicks for testing, our interaction policies for training and testing are consistent. Hence, the training setting is similar to that of testing in order to reduce the bias between training and testing. In one training/testing sequence of an image, we give N_{click} clicks each step. Specifically, the clicks are selected as the centers of the largest N_{click} error regions. In addition, a small disturbance ϵ_{noise} is added to each click point to force the model to be robust and also make it imitate the behavior of a real user.

Implementation details. For the preprocessing, all the images are normalized by the mean and the standard variation of the whole dataset $D = [D_{train1}, D_{train2}, D_{test}]$. Each image is cropped by the bounding box based on its non-zero region with an extension of $[0, 10]$ voxels and then resized to the size of $55 \times 55 \times 30$. Data augmentation involves flipping in three directions and random rotation with angle range $[-\pi/8, \pi/8]$ in three directions.

Initial / **Update**	BG	V-Net	HighRes3DNet	DeepIGeoS(P-Net)
Initial	0	77.15	75.39	82.16
Min-cut	27.46	80.69	77.05	84.08
DeepIGeoS(R-Net)	82.97	85.80	85.72	84.83
InterCNN	85.17	85.56	87.29	86.54
IteR-MRL	**86.14**	**88.53**	**87.43**	**87.50**

Table 1. Combination with different initial methods

Step	0	1	2	3	4	5
Clicks	0	5	10	15	20	25
Min-Cut	77.15	79.52 (+2.37)	79.97 (+0.45)	80.22 (+0.25)	80.46 (+0.24)	80.69 (+0.23)
DeepIGeoS(R-Net)	77.15	**85.62** (**+8.47**)	85.74 (+0.12)	85.73 (-0.01)	85.75 (+0.02)	85.80 (+0.05)
InterCNN	77.15	83.19 (+6.04)	84.39 (+1.20)	85.16 (+0.77)	85.52 (+0.36)	85.56 (+0.04)
IteR-MRL	77.15	84.35 (+7.20)	**86.78** (**+2.43**)	**87.61** (**+0.83**)	**88.18** (**+0.57**)	**88.53** (**+0.35**)

Table 2. Performance improvement in one interactive sequence

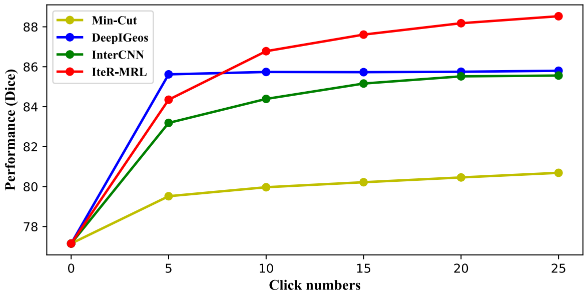

Figure 4. Performance improvement shown by curves

As the proposed IteR-MRL can be easily adapted to the interaction-free mode, we firstly train a pure segmentation model IteR-MRL0 as the pretrained model for IteR-MRL. IteR-MRL0 is trained for 1000 epochs and IteR-MRL finetunes on IteR-MRL0 for 500 epochs. The learning rate adopts the step decay schedule with an initial learning rate 10^{-4}. Parameter setting is as follows: $T = 5$, $N_{click} = 5$, $\gamma = 0.95$, $\epsilon_{noise} = [-3, 3]^3$. We use Adam algorithm for optimization with minibatch size 1.

The model training time with one Nvidia Titan X GPU varies from several hours to two days for different datasets. The average inference time for each update step is 894ms, which includes 424ms of the interaction simulation time.

4.3. Results

For fair comparison, we apply denseCRF to all the models compatible with CRF as the final refinement processing.

Comparisons with state-of-the-art methods. We compare IteR-MRL with three state-of-the-art methods: Min-Cut [10], DeepIGeoS(R-Net) [20] and InterCNN [5].

In Table 1, the update methods receive the coarse segmentations from four different initial segmentation methods: BG (set all voxel labels to background), V-Net [13], HighRes3DNet [11] and DeepIGeoS(P-Net) [20]. The experimental results show that IteR-MRL achieves better performances than baselines under each initial method, which

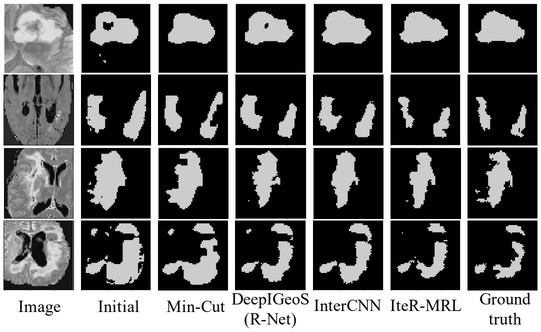

Figure 5. Visualization of different update methods

Dataset	BRATS2015	MM-WHS	NCI-ICBI2013
Initial	77.15	79.60	79.34
Min-Cut	80.69	83.21	79.92
DeepIGeoS(R-Net)	85.80	85.21	79.97
InterCNN	85.56	84.76	82.14
IteR-MRL	**88.53**	**86.92**	**82.71**

Table 3. Performances on different datasets

shows the robustness and generalization of our method.

To validate whether considering the relative gain between successive predictions can result in rapid improvements, we also analyze the performance improvement during one refinement sequence in Table 2. We use V-Net here as the initial method (77.15). For the first refinement step, all the update methods have significant improvements in performance (from $+2.37$ to $+8.47$). Starting from the second step, most performances have encountered stagnation (very little improvement) though with newly-added user hints. DeepIGeoS(R-Net) even has a degradation (-0.01) at the third step. While the other methods improve slowly at each refinement step, IteR-MRL has a relative high improvement, which proves the effectiveness of considering the relation gain between successive predictions. The large improvement at each refinement step also leads to a good result (88.53) in the end. In addition, we notice that IteR-MRL's performance at the second step has already surpassed the others' final performances, achieving a reduction of user click number. Fig. 4 provides a global view of performance improvement in one interactive sequence.

Fig. 5 gives the visualization of different update methods using V-Net as the initial method. Specifically, we visualize the refined segmentations after five refinement steps. It can be observed that while the other methods tend to produce a rather smooth boundary, IteR-MRL performs better in capturing edge details.

The above results are obtained from the experiments with the dataset BraTS2015. More experiments are also conducted on the other two datasets MM-WHS and NCI-ISBI2013 in Table 3 to verify the robustness, with the initial method V-Net. The results prove that IteR-MRL has stable performances on various types of datasets.

Actions				States		IteR-MRL
±0.1	±0.2	±0.4	±1.0	Probability	Binary	
			✓		✓	84.03
		✓		✓		84.29
	✓			✓		86.51
✓				✓		87.20
✓	✓			✓		87.88
✓	✓	✓		✓		**88.53**
✓	✓	✓	✓	✓		88.02

Table 4. Combination of different action and state settings

Step Clicks	0 0	1 5	2 10	3 15	4 20	5 25
Absolute reward	77.15	**85.55** (**+8.40**)	86.53 (+0.98)	87.59 (**+1.06**)	87.75 (+0.16)	87.65 (-0.10)
Relative reward	77.15	84.35 (+7.20)	**86.78** (**+2.43**)	**87.61** (+0.83)	**88.18** (**+0.57**)	**88.53** (**+0.35**)

Table 5. Influence of reward function on performance

Ablation study. We analyze the effect of different action sets to the algorithm performance in Table 4. Specially, when the action set only contains ± 1.0 (line 1), the segmentation probability becomes binary, because the segmentation probability can only take the values 0 and 1. The rest action sets are all designed for the states containing segmentation probability. The influence of the action value and the action number are both analyzed. For the influence of value, we fix the number of actions and let action values vary: we try ± 1.0, ± 0.4, ± 0.2 and ± 0.1 (line 1, 2, 3, 4). Comparing the states adopting segmentation probability to those of binary prediction, it can be found that the latter has a poor performance caused by the loss of prediction uncertainty. In addition, the results show that small action values have better performances than the larger ones. The reason is that a small action value allows IteR-MRL to make more detailed adjustments, but a large one may over-behave and never reach some specific states. For the influence of the action number, we gradually add new actions to the action set (line 4, 5, 6, 7). It can be observed that abundant actions lead to a better performance by providing IteR-MRL with various degrees of adjustment. In the case with a high confidence, IteR-MRL tends to take a large adjustment, which speeds up the refinement convergence. However, the addition of ± 1.0 relatively damages the performance because an adjustment of ± 1.0 is too extreme for most of the cases. In general, we learn that the combination of small and large actions except for ± 1.0 works best and finally choose $\mathcal{A} = \{\pm 0.1, \pm 0.2, \pm 0.4\}$ as the ideal action set in our model. Note that we have also thought about a continuous action space, but the experimental result shows that it is difficult to train and converge in a continuous action space. In addition, since the final prediction presented to the user is 0 or 1 for each voxel, we are not concerned about the inability to reach the optimal final state with a discrete action space.

In Table 5, we compare different designs for reward

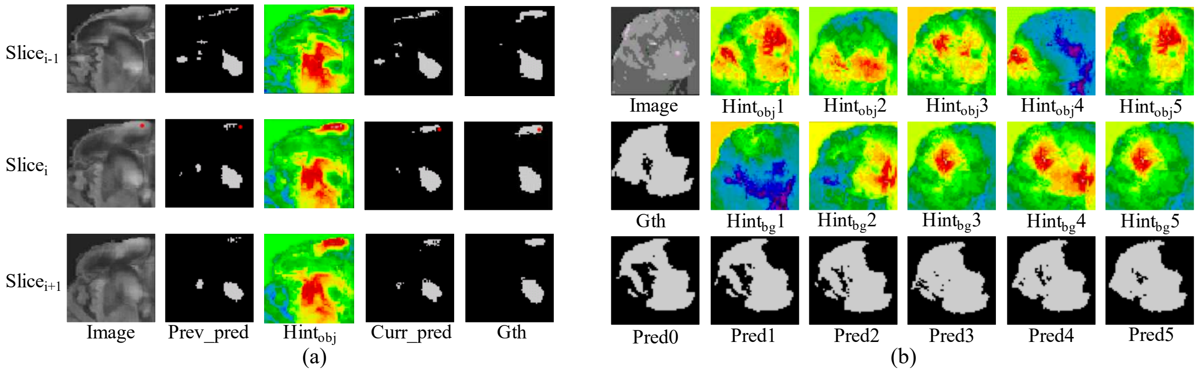

Figure 6. The visualization for the relation between predictions and hints. (a) The visualization of one click and its influence on prediction and hint maps. The slice with click and its two neighbor slices are shown. The user click is represented as red points. One row of five figures form a group, which corresponds to one slice [Image, Previous prediction, Object hint map, Current prediction, Ground truth]. (b) The visualization of prediction and hint map for each step. The figures in the first column are [Image, Ground truth, Initial prediction]. Afterwards, each column forms one step, which corresponds to [Object hint map, Background hint map, Prediction].

Step Clicks	0 0	1 5	2 10	3 15	4 20	5 25
Good Interaction	77.15	**84.35** **(+7.20)**	**86.78** **(+2.43)**	**87.61** **(+0.83)**	**88.18** **(+0.57)**	**88.53** **(+0.35)**
W/O Interaction	77.15	78.60 (+1.45)	79.53 (+0.93)	80.15 (+0.62)	80.56 (+0.41)	80.78 (+0.22)
Bad Interaction	77.15	76.86 (-0.29)	75.47 (-1.39)	74.84 (-0.63)	74.29 (-0.55)	72.76 (-1.53)

Table 6. Contribution of interactions to performance

Step Clicks	0 0	1 5	2 10	3 15	4 20	5 25
Immediate reward ($T = 1$)	77.15	**84.44** **(+7.29)**	81.26 (-3.18)	78.36 (-2.90)	76.07 (-2.29)	74.89 (-1.18)
Accumulated reward ($T = 5$)	77.15	84.35 (+7.20)	**86.78** **(+2.43)**	**87.61** **(+0.83)**	**88.18** **(+0.57)**	**88.53** **(+0.35)**

Table 7. Accumulated reward v.s. immediate reward

functions. An alternative to the relative reward mentioned before is the absolute reward, i.e. the difference between the current prediction and ground truth. The results show that the relative one performs better. A possible reason is that the relative gain can better reflect the agent's adjustment to segmentation probability.

As we know, the interaction and the model can both lead to the improvement of performance. We now analyze how much the interaction contributes to the performance by changing the interaction strategy. In addition to the good interaction used before, two more comparative experiments are done in Table 6. The one without interaction is to always fill the hint maps with random noise and the model will not receive any new interactive information. The other one with bad interaction is to randomly choose the user click points among all the voxels. In this case, the interaction may pass the wrong message to the model. As a result, we find that the meaningful interaction does help greatly improve the performance. It can also be observed that the one without interactions still has some gain of performance, which may come from the iterative model itself. Moreover, the degradation of the one with bad interactions indicates that ineffective interaction can damage the performance.

To verify the usefulness of accumulated reward over immediate reward, we analyze T, the total step number during one refinement sequence for training. Table 7 shows that accumulated reward outperforms immediate reward and the latter has a bad performance for multi-step interaction.

Fig. 6 presents the visualization for the relation between predictions and hints. Fig. 6(a) shows the influence of user interaction on prediction and hint maps. Since the data is 3D, we show the slice with click (the middle row) and its two neighbor slices (rows on both sides). The red parts on hint maps are the recommended object regions. We find that the proposed algorithm can successfully correct the local region around the user click (the red point). Besides, the corresponding regions on neighbor slices are also improved. In Fig. 6(b), we observe the change of predictions and hint maps in one interactive sequence which contains five steps. The user clicks are not shown because the slices with clicks vary at each step and we only focus on changes of one specific slice. With the indications of hint maps, IteR-MRL succeeds in refining the initial prediction step by step.

5. Conclusion

In this paper, we propose a novel iteratively-refined interactive segmentation method for 3D medical images using multi-agent reinforcement learning. The method explicitly models the dynamic process of interactive image segmentation task in order to get a rapid segmentation improvement at each iteration. The experimental results show that it performs better than the state-of-the-art methods and it is robust to various initial segmentations and datasets.

Acknowledgements This work is supported by NSFC (11871279, U1509219, 61702188, 61771306, 61521062), 111 Plan (B07022), SHEITC(2018-RGZN-02046) and STCSM (19ZR1414200, 18DZ2270700).

References

[1] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Efficient interactive annotation of segmentation datasets with Polygon-RNN++. In *CVPR*, pages 859–868, 2018.

[2] N Bloch, A Madabhushi, H Huisman, J Freymann, J Kirby, M Grauer, A Enquobahrie, C Jaffe, L Clarke, and K Farahani. NCI-ISBI 2013 challenge: automated segmentation of prostate structures. *The Cancer Imaging Archive*, 370, 2015.

[3] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of Min-Cut/Max-Flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (9):1124–1137, 2004.

[4] Yuri Y Boykov and M-P Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *ICCV*, pages 105–112, 2001.

[5] Gustav Bredell, Christine Tanner, and Ender Konukoglu. Iterative interaction training for segmentation editing networks. In *MLMI International Workshop*, pages 363–370, 2018.

[6] Lluis Castrejon, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler. Annotating object instances with a Polygon-RNN. In *CVPR*, pages 5230–5238, 2017.

[7] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3D U-Net: learning dense volumetric segmentation from sparse annotation. In *MICCAI*, pages 424–432, 2016.

[8] Antonio Criminisi, Toby Sharp, and Andrew Blake. Geos: Geodesic image segmentation. In *ECCV*, pages 99–112, 2008.

[9] Ryosuke Furuta, Naoto Inoue, and Toshihiko Yamasaki. Fully convolutional network with multi-step reinforcement learning for image processing. In *AAAI*, pages 3598–3605, 2019.

[10] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected CRFs with gaussian edge potentials. In *NeurIPS*, pages 109–117, 2011.

[11] Wenqi Li, Guotai Wang, Lucas Fidon, Sebastien Ourselin, M Jorge Cardoso, and Tom Vercauteren. On the compactness, efficiency, and representation of 3D convolutional networks: brain parcellation as a pretext task. In *IPMI*, pages 348–360, 2017.

[12] Bjoern Menze, Andras Jakab, Stefan Bauer, Jayashree Kalpathy-Cramer, Keyvan Farahani, Justin Kirby, Yuliya Burren, Nicole Porz, Johannes Slotboom, Roland Wiest, Levente Lanczi, Elisabeth Gerstner, et al. The multimodal brain tumor image segmentation benchmark (BraTS). *IEEE Transactions on Medical Imaging*, page 33, 2014.

[13] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-Net: Fully convolutional neural networks for volumetric medical image segmentation. In *3DV*, pages 565–571, 2016.

[14] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *ICML*, pages 1928–1937, 2016.

[15] Martin Rajchl, Matthew CH Lee, Ozan Oktay, Konstantinos Kamnitsas, Jonathan Passerat-Palmbach, Wenjia Bai, Mellisa Damodaram, Mary A Rutherford, Joseph V Hajnal, Bernhard Kainz, et al. Deepcut: Object segmentation from bounding box annotations using convolutional neural networks. *IEEE Transactions on Medical Imaging*, 36(2):674–683, 2016.

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

[17] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. GrabCut: Interactive foreground extraction using iterated graph cuts. In *ACM Transactions on Graphics*, volume 23, pages 309–314, 2004.

[18] Gwangmo Song, Heesoo Myeong, and Kyoung Mu Lee. Seednet: Automatic seed generation with deep reinforcement learning for robust interactive segmentation. In *CVPR*, pages 1760–1768, 2018.

[19] Guotai Wang, Wenqi Li, Maria A Zuluaga, Rosalind Pratt, Premal A Patel, Michael Aertsen, Tom Doel, Anna L David, Jan Deprest, Sébastien Ourselin, et al. Interactive medical image segmentation using deep learning with image-specific fine tuning. *IEEE Transactions on Medical Imaging*, 37(7):1562–1573, 2018.

[20] Guotai Wang, Maria A Zuluaga, Wenqi Li, Rosalind Pratt, Premal A Patel, Michael Aertsen, Tom Doel, Anna L David, Jan Deprest, Sébastien Ourselin, et al. DeepIGeoS: a deep interactive geodesic framework for medical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(7):1559–1572, 2018.

[21] Guotai Wang, Maria A Zuluaga, Rosalind Pratt, Michael Aertsen, Tom Doel, Maria Klusmann, Anna L David, Jan Deprest, Tom Vercauteren, and Sébastien Ourselin. Slic-seg: A minimally interactive segmentation of the placenta from sparse and motion-corrupted fetal MRI in multiple views. *Medical Image Analysis*, 34:137–147, 2016.

[22] Ning Xu, Brian Price, Scott Cohen, Jimei Yang, and Thomas S Huang. Deep interactive object selection. In *CVPR*, pages 373–381, 2016.

[23] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. UNet++: A nested U-Net architecture for medical image segmentation. In *DLMIA*, pages 3–11. 2018.

[24] Xiahai Zhuang and Juan Shen. Multi-scale patch and multi-modality atlases for whole heart segmentation of MRI. *Medical Image Analysis*, 31:77–87, 2016.