SeedNet: Automatic Seed Generation with Deep Reinforcement Learning for Robust Interactive Segmentation

Gwangmo Song* Heesoo Myeong* Kyoung Mu Lee

Department of ECE, ASRI, Seoul National University, 08826, Seoul, Korea
kfsgm@snu.ac.kr, heesoo.myeong@gmail.com, kyoungmu@snu.ac.kr

Abstract

In this paper, we propose an automatic seed generation technique with deep reinforcement learning to solve the interactive segmentation problem. One of the main issues of the interactive segmentation problem is robust and consistent object extraction with less human effort. Most of the existing algorithms highly depend on the distribution of inputs, which differs from one user to another and hence need sequential user interactions to achieve adequate performance. In our system, when a user first specifies a point on the desired object and a point in the background, a sequence of artificial user input is automatically generated for precisely segmenting the desired object. The proposed system allows the user to reduce the number of input significantly. This problem is difficult to cast as a supervised learning problem because it is not possible to define globally optimal user input at some stage of the interactive segmentation task. Hence, we formulate automatic seed generation problem as Markov Decision Process (MDP) and then optimize it by reinforcement learning with Deep Q-Network (DQN). We train our network on the MSRA10K dataset and show that the network achieves notable performance improvement from inaccurate initial segmentation on both seen and unseen datasets.

1. Introduction

Segmenting the object of interest in an image is one of the fundamental problems in computer vision. However, without knowing the user's intention, automatic object selection has inherent limitations because where and what objects should be extracted differs by users. For this reason, an interactive segmentation approach that receives information on the desired object roughly from a human in the form of a scribble or a bounding box and performs segmentation is widely used to extract the object from an image and video.

*First two authors contributed equally

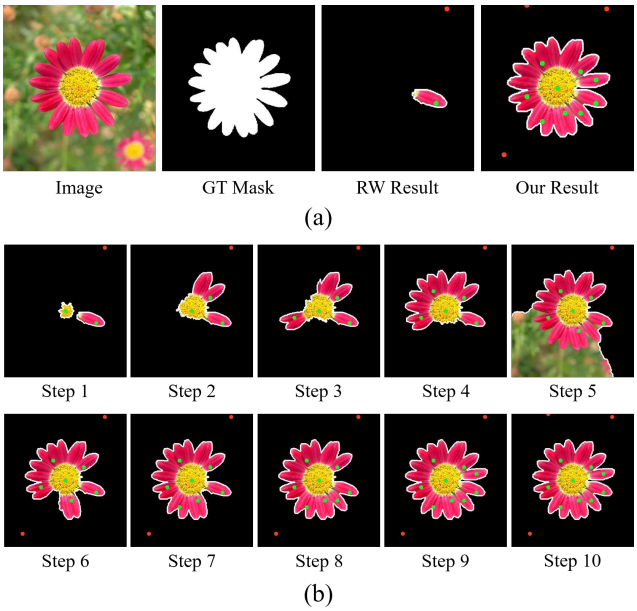

Figure 1. An automatic seed generation example. The green and red dots represent the foreground and the background seeds, respectively. (a) RW result is the output of random walker segmentation [13] algorithm with the initial seeds and our result is the segmentation output with the generated seeds from the SeedNet. (b) Seed generation process through the SeedNet. At each step, the SeedNet creates a new foreground or background seed input.

One of the critical components of the interactive segmentation algorithm is robust object extraction while matching the human intention. For many objects with a complex background, the user often has to spend much effort to refine the results obtained from the algorithm. In this regard, how to reduce human effort while maintaining the performance in interactive segmentation is very important. In [14], the number of additional efforts by users is used as a measure of system performance. In this study, we propose a novel technique to simulate the human process of guiding the interactive segmentation system to obtain the

desired object. When the user enters a point on the desired object and a point on the background, our system automatically generates the sequence of artificial user input to accurately localize the target object of interest, as illustrated in Figure 1. The proposed system is designed to achieve high performance while significantly reducing the number of user input.

In this work, we formulate the automatic seed generation problem as a sequential decision-making problem and train the seed generation agent with deep reinforcement learning. Our agent starts by analyzing the image and the foreground/background segmentation produced with the initial seeds by the user, and then determines a new foreground or background seed. After creating a new segmentation by combining the created seed with the initial seeds, our agent uses this segmentation as a next input and repeats the process of creating seeds. Deep reinforcement learning is suitable for our task because we cannot define globally optimal seed at some stage of interactive segmentation. Additionally, for effective learning, we propose a novel reward function depending on the intersection-over-union (IoU) score. The advantage of the proposed system is that consistent performance has been achieved in images in unobserved datasets as well as in previously observed datasets.

The contributions of this paper include (1) the introduction of a Markov Decision Process (MDP) formulation for the interactive segmentation task where an agent puts seeds on the image to improve segmentation and (2) the novel reward function design to train the agent for automatic seed generation with deep reinforcement learning.

2. Related Works

Interactive segmentation: As one of the major problems in computer vision, interactive segmentation has been studied for a long time. Many interactive segmentation algorithms have tried to segment a desired object with various user input such as contour, scribble or bounding box. Numerous methods such as GrabCut [26], random walks [13, 16], geodesics [5], and methods with shape prior [30, 14] have been proposed.

Recently, learning-based interactive segmentation algorithms have attracted considerable attention. Wu *et al*. [33] considered interactive segmentation problem as a weakly supervised learning problem. In [33], sweeping line multiple instance learning (MIL) technique was presented. The MIL-based classifier is trained with foreground and background bags from user-annotated bounding box. Santner *et al*. [27] also treated the interactive segmentation problem in a weakly supervised learning manner. [27] showed that HoG descriptors learned with random forests successfully segment out a textured object. Kuang *et al*. [18] trained optimal parameters for a single image. The weights for color, texture, and smoothing terms are tuned during the iteration process.

Meanwhile, various studies have been carried out on algorithms for extending seed information. These studies are closely related to our work, in that extended seed information is used. Seeded region growing (SRG) techniques [1] is representative work. In each step of SRG, the most similar pixel among adjacent pixels is taken as an additional seed point. This process extends the seed set. GrowCut [31] also uses a similar algorithm concept. It uses cellular automaton as an image model. Automata evolution models the segmentation process. In each step, a labeled cell tries to attack its neighbors. If the defender cell's strength is lower than that of the attacker, the label of defender cell changes to that of the attacker. However, our method differs in that it proposes new points rather than expanding the seed area.

With the recent development of deep learning, Xu *et al*. [34] proposed a neural network architecture for interactive segmentation. Semantic information is considered by using fully convolutional neural networks (FCN) in their framework. By fine-tuning FCN block, the CNN structure can be used efficiently for interactive segmentation problems. Liew *et al*. [15] improved segmentation performance by creating global and local branches based on CNN architecture. However, our goal is not to train binary mask directly, but to train seed generation step that can help the existing segmentation algorithms.

Deep reinforcement learning: Research on deep reinforcement learning has been actively carried out due to its excellent performance in an Atari game via Deep Q-Network (DQN) [22]. Techniques such as prioritized experience replay [28], double DQN [29], dueling DQN [32], and A3C [21] have been studied to improve the performance of the reinforcement learning algorithm. The reinforcement learning algorithm is often applied in Atari games or robotics problems, but it also has many potential applications in computer vision fields.

A typical application to computer vision using reinforcement learning is the object localization problem. In [9], the authors interpreted the object localization problem as a sequential dynamic decision-making problem. In each decision step, an action is represented by the transformation of a detection box. With a deep representation of an image and previous actions, DQN predicts the action of next step. Similar to [9], [7] used box transformation actions and DQN to predict the next action. They employed a tree-structured search to enable the localization of multiple objects in a single run.

Reinforcement learning framework is also used for image classification problems [4], image captioning [24], video tracking [36], face hallucination [10] and video activity recognition task [35]. Andreas *et al*. [3] applied reinforcement learning to solve the question answering problem. They trained a network structure predictor with rein-

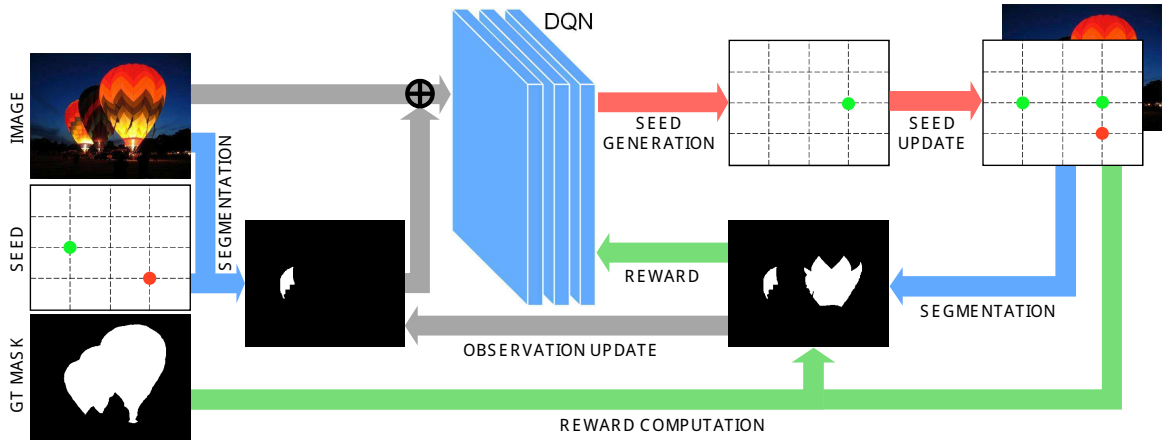

Figure 2. Overview of the proposed SeedNet. The image and the segmentation mask are the input of the DQN. The seed set is updated using the newly created seed from the DQN, and the mask is generated using the revised seed set. The obtained mask is used to calculate the reward value by comparing with the GT mask, and this process is repeated. The gray arrows indicate state-related behavior, red arrows indicate action-related behavior, and green arrows indicate reward-related behavior.

forcement learning technique.

In most computer vision applications, researchers used a combination of attention models and reinforcement learning. However, we solved the problem of generating seed points by directly using the image space as a large action space.

3. Automatic Seed Generation System

3.1. System Overview

In this work, we propose a novel automatic seed generation system for the task of interactive segmentation. We call it *SeedNet*. When an image and sparse seed information are entered, the ultimate goal of the proposed system is to create additional seed points and obtain accurate segmentation result. The core module of SeedNet is a deep reinforcement learning agent for generating artificial seed points. Also, SeedNet includes an off-the-shelf segmentation model that performs the segmentation operation with the generated seed. The entire system is constructed by learning the DQN [22] agent using the segmentation result.

The overall process of SeedNet is shown in Figure 2. The operation of the system proceeds with the image and the initial seed map given by the user. By utilizing this input information, performing interactive segmentation yields a binary mask. We use Random Walk (RW) segmentation [13] as an off-the-shelf interactive segmentation algorithm. The obtained binary mask and image are concatenated and then input to the DQN. The DQN model proposes new seed information by using the input. The new seed information contains the position and label of the proposed seed. As a result, the seed map is updated by adding the proposed seed point to the existing seed information. In addition, segmentation of the image using the new seed information results in a new binary mask. The obtained binary mask is used for two purposes: the first is to compute the reward signal by comparing the obtained mask with the ground truth (GT) mask. The reward is a value that evaluates the operation of the DQN and is used to update the network. Second, the acquired binary mask is used as an observation of the next iteration.

The sequence of cyclic operations is repeated throughout the training process. However, during the test time, the reward part is omitted, and only the seed generation process is performed. By repeating the steps of generating a seed, a seed map containing several artificial seeds is obtained. In this way, we significantly reduce the human effort on interactive segmentation task.

3.2. Markov Decision Process (MDP)

The core part of the proposed SeedNet is to generate a sequence of seeds by the agent. We define the problem as an MDP consisting of state, action, and reward and the agent operates through the MDP. The agent takes the current state as an input, performs some action, and receives a corresponding reward. This section presents the definition of the proposed MDP.

State: The state should contain enough information to allow the agent to make the best choice. For our problem formulation, information on the whole image is essential. Additionally, the state should include information on observation that changes at each step. We can obtain two kinds of information when a seed is generated at every step: one is the newly created seed map, and the other is a bi-

nary mask using off-the-shelf interactive segmentation algorithm. Given that we want the proposed system to be robust to the seed position, we exclude the seed position information and add only the binary mask information to the state. In addition, past observations are not used, and only the current observation is utilized as the state.

As a result, in our formulation, the state is defined as the current binary segmentation mask and image features. Unlike many existing works, the proposed system does not use any deep feature representation as the state.

Action: Given a state, the agent selects an action within the action space. In our formulation, the action is defined as a positioning new seed point. The agent decides the label (foreground/background) and position of the seed in the 2D grid given the states. If we set the 2D grid to correspond to all the pixels in the image, the action space becomes too large, causing problems in training. Therefore, the 2D grid where the new seed can be placed is sparsely set to 20×20 size. There are a total of 800 kinds of actions because of the foreground and background grids. If an agent selects one of 800 actions, a new seed point is created at the corresponding location. Meanwhile, there is no explicit terminal action because it is hard to define the termination station. Thus, we terminate the process after proposing 10 seed points.

Reward: The reward signal evaluates the result for the action of the agent. Generally, in a game environment, a score or win/loss is used as a reward function. In our system, the results of agent action are seed position and segmentation mask. Thus, we can use the accuracy of the segmentation mask as a score concept. The accuracy of the mask can be determined by comparison with the ground truth (GT) mask. For evaluation, IoU is the common metric. Therefore, the intuitive basic reward function is to use IoU as a reward function. The reward function with IoU is described as R_{IoU}.

$$R_{\text{IoU}} = IoU(M, G), \tag{1}$$

where M denotes the obtained segmentation mask and G denotes the GT mask. Another basic reward function is to use the change trend of IoU. It compares the IoU value of the current mask with the IoU of the previous step mask and gives a success signal if the value is increased and a failure signal if it is decreased. It is like win/loss reward signal in the game environment. In our environment, however, we can obtain the amount of change as well as the direction of change. Therefore, a more flexible reward signal can be designed by using the variation of IoU as the value of reward instead of the binary type reward. It is described as R_{diff}.

$$R_{\text{diff}} = IoU(M, G) - IoU(M_{\text{prev}}, G), \tag{2}$$

where M_{prev} is the segmentation mask of the previous step. In addition, by using an exponential IoU model(R_{exp}) instead of a linear IoU model, we can design a reward signal that gives more attention to changes in high IoU values.

$$R_{\text{exp}} = \frac{exp^{k*IoU(M,G)} - 1}{exp^{k} - 1}, \tag{3}$$

where k is a constant value. Meanwhile, given that we have information on the seed position as well as information about the mask, we can generate an additional signal to assist the IoU reward. Instead of judging success/failure by using the change in IoU, we can judge by comparing GT mask with the newly generated seed. That is, if the label of the new seed matches the GT label of the corresponding location, it is a success; otherwise, it is a failure. With a similar concept, we divide the GT mask into four regions and compare them with the seed label. To divide GT mask into four regions, we create additional boundaries inside and outside the object that give some margin from the object boundary. That is, four regions are generated from three boundaries, including an existing object boundary. These four regions are named strong foreground (SF), weak foreground (WF), weak background (WB), and strong background (SB), in the order from the center of the object to the edge of the image. When a new seed point is assigned, different reward functions are applied to the divided areas according to seed type.

For example, if the newly given foreground seed belongs to the SF area of the mask, we apply exponential IoU reward. Also, if foreground seed belongs to the WF domain, it is also a success case but is not recommended, so a reduced reward signal is applied. Otherwise, if foreground seed is wrongly suggested on the background area, a fixed reward value of -1 is returned. Likewise, when a new background label seed is given, we can obtain a reward similar to the foreground case. The R_{our} used in this paper is as follows:

$$R_{\text{our}} = \begin{cases} R_{\text{exp}} & \text{if } F_{\text{seed}} \in \text{SF or } B_{\text{seed}} \in \text{SB} \\ R_{\text{exp}} - 1 & \text{if } F_{\text{seed}} \in \text{WF or } B_{\text{seed}} \in \text{WB} \\ -1 & \text{otherwise} \end{cases}, \tag{4}$$

where F_{seed} means foreground seed and B_{seed} means background seed. We obtain a continuous score reward from the mask information and a discrete success/failure reward from the seed information. Finally, we propose a novel reward function by mixing the two types of reward. We compare the differences between the newly proposed reward function and other reward functions in the experimental section.

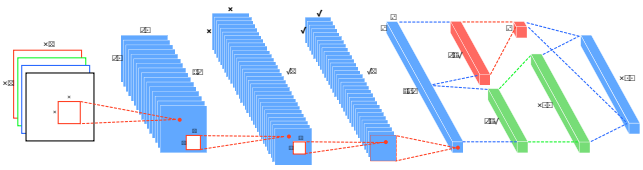

Figure 3. DQN architecture for SeedNet. The red block is the network for the state value function, and the green block is the network for the advantage function.

4. Training an Agent with Deep Reinforcement Learning

4.1. Deep Q-Network (DQN)

With the proposed MDP formulation, the seed generation agent can be trained through the deep reinforcement learning. In this study, we use the DQN algorithm by Mnih *et al*. [22] to train the agent. DQN learns the action-value function $Q(s, a)$, the expected reward that the agent receives when taking action a in a state s. After training, the agent selects the action with the learned Q-function. The Q-learning target can be defined with the given s, a, s':

$$r + \gamma max_{a'} Q(s', a'), \tag{5}$$

where r is the reward, γ is a discount factor, and s' and a' represent the state and action of the next step, respectively. DQN is a technique that approximates the Q-function with a deep neural network. The loss function for training the Q-function can be expressed:

$$Loss(\theta) = \mathbb{E}[(r + \gamma max_{a'} Q(s', a'; \theta) - Q(s, a; \theta))^2]. \tag{6}$$

For effective learning, we employ various techniques from Mnih *et al*. [22]. First, we use a target network to solve the problem of poor learning stability. By introducing a target network separately from the online network, the parameters of the target network during a few iterations are fixed while the online network is updated. This method has significantly improved the stability of learning. Next, we use an ϵ-greedy policy as a behavior policy. The ϵ-greedy policy uses a random action with a probability of ϵ and an action that maximizes the Q-function with a probability of 1-ϵ. The last is experience replay to solve the correlation problem of data used for DQN learning. We created an experience replay buffer, proceeded with the episode, and stored the replay memory in the buffer (s, a, r, s'). During the learning process, samples of the batch size are randomly selected from the buffer to reduce the correlation between the data.

4.2. Model Architecture

The DQN used in this study is shown in Figure 3. The structure of DQN used is almost similar to that of [22]. To improve the performance of the algorithm, we use the double DQN structure of [29] and dueling DQN structure of [32]. The input image and the binary mask resulting from the segmentation at the previous stage are resized to 84×84 and input to the network. Three convolution operations followed by ReLU activation are performed on the input. By taking advantage of the dueling structure, the 512-D layer after the fully-connected operation is split into two parts to learn the advantage function and state value function. Then, through a fully-connected operation, the advantage function $A(a, s)$ comes out as an 800-D output corresponding to the action space size. Meanwhile, the state value function $V(s)$ is a scalar value. Finally, the advantage function is added to the state value function to obtain the Q-function. The action is determined according to the Q-function having the maximum value. If the action label is less than 400, it will be the foreground seed. Otherwise, it will be the background seed and reduces the action label by 400 for conversion to grid coordinates. Finally, converting the action label to 20×20 grid coordinates will determine where the new seed will be located.

5. Experiments

We have experimented with several types of datasets. First, we use the MSRA10K saliency dataset [11] to train and compare our results against the initial results from the initial seed. We also conduct a comparative experiment on various single object datasets that were not included in the training dataset.

5.1. Network Learning

In this paper, SeedNet is trained for MSRA10K saliency dataset from scratch. In the training process, 10,000 pre-training steps are preceded to build an experience replay buffer to be used for learning. During the pre-training step, the actual learning does not proceed, but the experience that goes through the episode is stored in the buffer. We used 50,000 experience as a buffer and 32 as a batch size. For exploration, we use ϵ-greedy policy. During training, ϵ decreases from 1 to 0.2 over 10,000 steps. In the subsequent training process, ϵ is fixed to 0.2. As the learning progresses, the action is randomly selected as the probability of ϵ, and the action according to the learned network is selected by the probability of 1-ϵ. The parameters for the specific network size are shown in Figure 3, and the discount factor γ is set to 0.9. Each episode contains a total of 10 seed point generation processes. For training, we use an Adam optimizer [17] and utilize a learning rate of 1e-4. Also, the update rate to the target network is set to 1e-7.

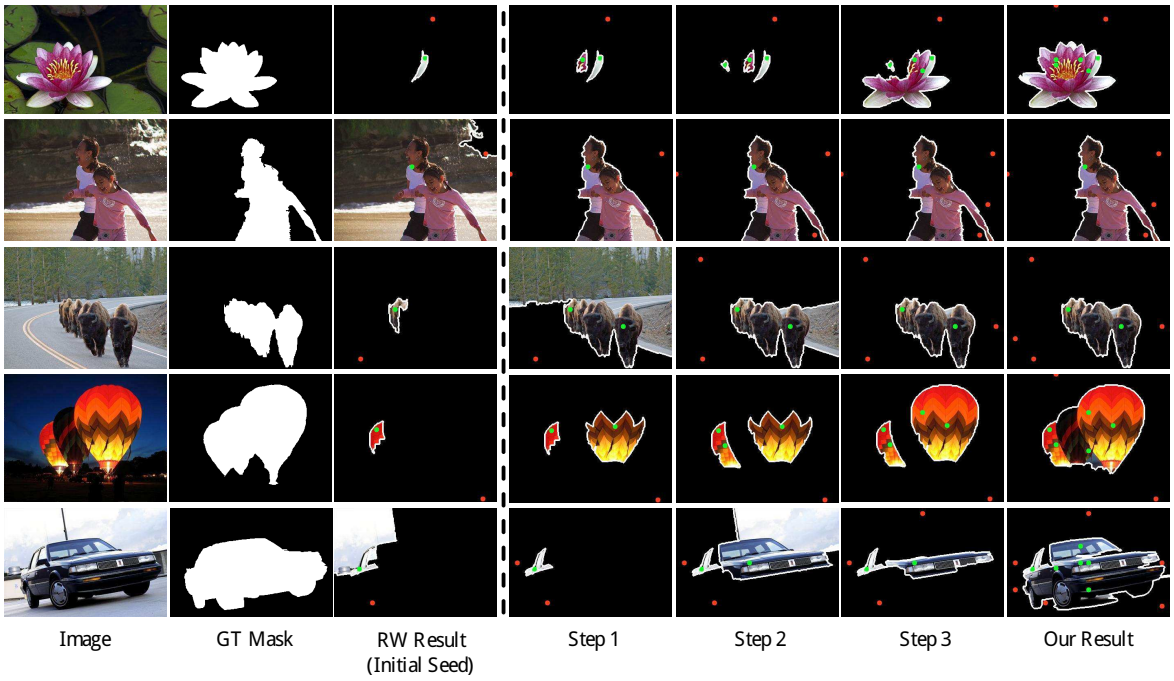

Figure 4. MSRA10K results. The left part shows the input image, GT mask, and initial seed with corresponding RW [13] result. The right part shows the SeedNet result, showing the first three steps and final result.

As previously mentioned, a 20×20 size grid is used as the action space, and the k value of the exponential reward function is set to 5.

5.2. Interactive Segmentation Results

First, our performance evaluation is done on the MSRA10K dataset. The MSRA10K dataset consists of 10,000 images, and we use 9,000 of them as training and the remaining 1,000 as test. Each image consists of an RGB image and a mask representing the GT, and seed information is not included. The size of the image is approximately 400×300 pixels. To accelerate the learning process, each image and GT are reduced to 1/4 size in the learning stage. The same image size of 84×84 is input to the DQN during training and testing. However, when segmentation is performed with a newly generated seed, segmentation is applied to a 1/4 size image in the learning process to obtain a fast result, and the original image size is used in the test time. As the size of segmented images increases, the size of the seeded points increases simultaneously. In training, a circle with a diameter of 3 pixels is used as a seed, and a circle with a diameter of 13 pixels is used as a seed in the test.

Given that seed information is not included in the MSRA10K dataset, we experiment with initial seed point randomly generated using the GT mask information. We apply dilation and erosion separately to the GT mask to form a region slightly distant from the object boundary and randomly select foreground and background seed points from each region. As the initial seed point is determined randomly, we perform five experiments sequentially and evaluate the performance using the average value. We use the RW segmentation method as an off-the-shelf segmentation algorithm in our system. The results obtained using only the initial seed point and the newly proposed seeds of this system are compared and shown in Table 1. The IoU metric is used for evaluation.

Table 1. MSRA10K Result

Method	Set 1	Set 2	Set 3	Set 4	Set 5	Mean
RW [13]	39.59	39.65	39.71	39.77	39.89	**39.72**
SeedNet	60.70	60.12	61.28	61.87	60.90	**60.97**

Table 2. Comparison with supervised methods

Method	FCN [19]	iFCN [34]	*SeedNet*
IoU	37.2	44.6	**60.97**

The results show that the accuracy is significantly increased when seed information generated by the proposed SeedNet is used compared with RW segmentation using only the initial seed. Meanwhile, we change the initial seed distribution from Set 1 to Set 5, but it is not significantly

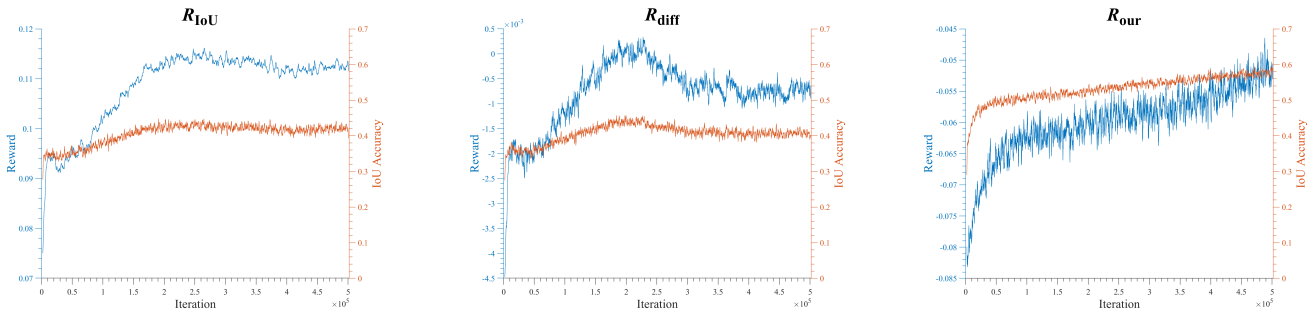

Figure 5. SeedNet learning progress graph using R_{IoU} (left), R_{diff} (center), and R_{our} (right). The reward value is indicated by the blue line and the left axis, and the IoU value is indicated by the orange line and the right axis. A common x-axis represents the progression of the learning iteration. For better visualization, the change is displayed every 100 steps and each point represents the running average value for 1000 steps.

affected by the initial seed distribution, and both RW and SeedNet show similar results. Qualitative results are shown in Figure 4. As shown in the figure, the automatically generated seed information gives better results than the initial seed. The figure 4 also shows the results up to step 3 and the final result. The average number of seeds used until saturation is **5.39** clicks. Therefore, the threshold of the proposed algorithm, which proposes generation up to 10 times, is reasonable. However, given that SeedNet generates a seed on a sparse grid, it is difficult to propose a seed in a finer position as in the case of the third row. Nevertheless, the additional seed is well presented without losing the intention of the initial seed.

Comparison with supervised methods: Additionally, we implement the FCN [19] and the iFCN [34] baseline. We input 80×80 image similar to our network input size, change the fully-connected layer to convolution layer in our network, give padding to make 10×10 output map, and perform deconvolution to the original size. Also, networks are trained from scratch. We add two seed input channels to the RGB channel for iFCN. The results are shown in Table 2. Although it is possible to obtain better performance by using the pretrained network and larger images, it is observed that the supervised segmentation has lower performance in the current configuration.

5.3. Ablation Experiments

To analyze the proposed system, we replace several key components of the system. Experiments are carried out while changing only the corresponding elements and keeping other parts intact.

Reward: Our DQN is updated with a reward comparing the GT with the observation. To verify the effectiveness of the proposed reward function, we train the system using a simple reward described in 3.2. For comparison, R_{IoU} and R_{diff} are used, and the change in reward value according to the learning time and the change in IoU accuracy of the training set according to the learning time are shown in Figure 5.

Table 3. Ablation Experiments : Reward

Method	Set 1	Set 2	Set 3	Set 4	Set 5	Mean
RW [13]	39.59	39.65	39.71	39.77	39.89	**39.72**
R_{IoU}	42.00	42.77	43.69	42.96	41.33	**42.55**
R_{diff}	44.33	44.80	45.09	44.19	43.82	**44.45**
R_{our}	60.70	60.12	61.28	61.87	60.90	**60.97**

Table 4. Ablation Experiments : Segmentation

Method	Set 1	Set 2	Set 3	Set 4	Set 5	Mean
GC [26]	38.15	38.29	38.35	38.70	38.71	**38.44**
SeedNet (GC*ver.*)	52.43	51.89	51.84	52.10	52.26	**52.10**
GSC [14]	57.85	58.10	58.50	58.57	58.70	**58.34**
SeedNet (GSC*ver.*)	63.09	62.70	64.24	63.16	64.19	**63.48**

The reward axis shown on the left has different axes for each graph because the scales are different for each reward function. Meanwhile, the IoU axis on the right has the same axis for all three graphs. Comparing the three graphs, we can see that simple reward functions initially increase in reward value but stay at a certain level, so that IoU no longer improves. Meanwhile, in the proposed reward function, both the reward and IoU values are steadily increased. The result of applying SeedNet learned by each reward function to the test set is shown in Table 3. As expected, we can confirm that the proposed reward function has better results than other reward functions.

Segmentation: SeedNet uses RW as an off-the-shelf segmentation algorithm, which can be replaced by other algorithms. SeedNet is trained using GrabCut (GC) [26] and GSCseq (GSC) [14], respectively. The results are shown in

Figure 6. MSRA10K result with SeedNet GC (upper two rows) and GSC versions (bottom two rows).

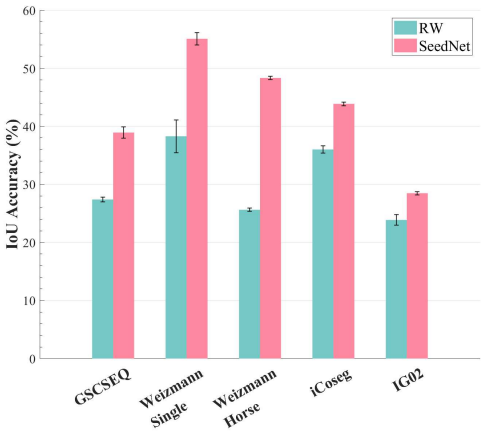

Figure 7. Results for unseen datasets. The horizontal axis represents each dataset, and the vertical axis represents the average IoU accuracy.

Table 4. Both the GC and GSC versions of SeedNet show an increase in IoU compared with the initial results. As other segmentation algorithms can be applied in this way, better results can be expected using CNN based algorithms, such as iFCN [34]. The results of using GC and GSC are shown in Figure 6.

5.4. Unseen Dataset Experiments

To verify the scalability of the proposed SeedNet, we conducted experiments on an unseen dataset. As this system is learned using the saliency dataset, MSRA10K, we test our agent on various single-object binary segmentation datasets instead of the validation images of the MSRA10K datasets. The experimental setup is the same as that of MSRA10K, and the evaluation is also performed with an average IoU for five random initial seeds.

GSCSEQ [14]: This dataset consists of a total of 151 images, including 49 pieces from the GrabCut dataset [26], 99 pieces from the Pascal VOC dataset [12], and 3 pieces from the Alpha matting dataset [25]. The dataset includes RGB images, GT binary masks, and scribble information. However, in this experiment, seed points are generated from the mask without using scribble information.

Weizmann Single Object [2]: The Weizmann single object dataset consists of 100 single object images, including three types of GT binary masks for each image. The three types of GT are slightly different depending on the subject of the labeling user, and we only use the first GT for evaluation.

Weizmann Horse [8]: A total of 328 images contain a side view of the horse. The dataset contains images and GT binary masks.

iCoseg [6]: iCoseg is a dataset mainly used for cosegmentation, and it has 38 categories and consists of 643 images in total. There are GT binary masks for each image.

IG02 [20]: The new annotation of the Graz-02 dataset [23] from INRIA consists of three categories: bikes, cars, and people. A total of 479 test images from each category are used for this experiment. Some images contain several objects, but only one object is tested in this experiment.

The experimental results are shown in Figure 7. In all five datasets, we can see that the result of using seed generated through SeedNet is significantly improved compared with the initial seed. In particular, the Weizmann Horse dataset shows an increase in accuracy of more than 20%. SeedNet, on the other hand, is relatively weak for the IG02 dataset, where multiple objects exist because we only train from a single object case. Nevertheless, we can confirm that the proposed SeedNet is applied well even though it is a dataset of different nature that has never been seen during training.

6. Conclusion

We have proposed a novel interactive segmentation agent for assisting a user to segment an object accurately. The agent can predict the user's intention and reduce the user's effort. Also, this approach has the potential to leverage the user's intent in various computer vision problems such as semantic segmentation. Furthermore, our agent can help to reduce the cost of pixelwise labeling task.

Acknowledgements

This work was partly supported by the National Research Foundation of Korea(NRF) grant funded by the Korea Government(MSIT) (No. NRF-2017R1A2B2011862)

References

[1] R. Adams and L. Bischof. Seeded region growing. In *PAMI*. IEEE, 1994. 2

[2] S. Alpert, M. Galun, R. Basri, and A. Brandt. Image segmentation by probabilistic bottom-up aggregation and cue integration. In *CVPR*. IEEE, 2007. 8

[3] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Learning to compose neural networks for question answering. In *NAACL*. Association for Computational Linguistics, 2016. 2

[4] J. Ba, V. Mnih, and K. Kavukcuoglu. Multiple object recognition with visual attention. In *ICLR*, 2015. 2

[5] X. Bai and G. Sapiro. Geodesic matting: A framework for fast interactive image and video segmentation and matting. In *IJCV*. Springer, 2009. 2

[6] D. Batra, A. Kowdle, D. Parikh, J. Luo, and T. Chen. icoseg: Interactive co-segmentation with intelligent scribble guidance. In *CVPR*. IEEE, 2010. 8

[7] M. Bellver, X. Giro-i Nieto, F. Marques, and J. Torres. Hierarchical object detection with deep reinforcement learning. In *Deep Reinforcement Learning Workshop, NIPS*, 2016. 2

[8] E. Borenstein and S. Ullman. Learning to segment. In *ECCV*. Springer, 2004. 8

[9] J. C. Caicedo and S. Lazebnik. Active object localization with deep reinforcement learning. In *ICCV*. IEEE, 2015. 2

[10] Q. Cao, L. Lin, Y. Shi, X. Liang, and G. Li. Attention-aware face hallucination via deep reinforcement learning. In *CVPR*. IEEE, 2017. 2

[11] M.-M. Cheng, N. J. Mitra, X. Huang, P. H. S. Torr, and S.-M. Hu. Global contrast based salient region detection. In *PAMI*. IEEE, 2015. 5

[12] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge 2009. In *2th PASCAL Challenge Workshop*, 2009. 8

[13] L. Grady. Random walks for image segmentation. In *PAMI*. IEEE, 2006. 1, 2, 3, 6, 7

[14] V. Gulshan, C. Rother, A. Criminisi, A. Blake, and A. Zisserman. Geodesic star convexity for interactive image segmentation. In *CVPR*. IEEE, 2010. 1, 2, 7, 8

[15] J. Hao Liew, Y. Wei, W. Xiong, S.-H. Ong, and J. Feng. Regional interactive image segmentation networks. In *ICCV*. IEEE, 2017. 2

[16] T. H. Kim, K. M. Lee, and S. U. Lee. Generative image segmentation using random walks with restart. In *ECCV*. Springer, 2008. 2

[17] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *CoRR*, 2014. 5

[18] Z. Kuang, D. Schnieders, H. Zhou, K.-Y. K. Wong, Y. Yu, and B. Peng. Learning image-specific parameters for interactive segmentation. In *CVPR*. IEEE, 2012. 2

[19] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*. IEEE, 2015. 6, 7

[20] M. Marszalek and C. Schmid. Accurate object localization with shape masks. In *CVPR*. IEEE, 2007. 8

[21] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *ICML*, 2016. 2

[22] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. In *Nature*. Nature Research, 2015. 2, 3, 5

[23] A. Opelt, A. Pinz, M. Fussenegger, and P. Auer. Generic object recognition with boosting. In *PAMI*. IEEE, 2006. 8

[24] Z. Ren, X. Wang, N. Zhang, X. Lv, and L.-J. Li. Deep reinforcement learning-based image captioning with embedding reward. In *CVPR*. IEEE, 2017. 2

[25] C. Rhemann, C. Rother, J. Wang, M. Gelautz, P. Kohli, and P. Rott. A perceptually motivated online benchmark for image matting. In *CVPR*. IEEE, 2009. 8

[26] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ToG*. ACM, 2004. 2, 7, 8

[27] J. Santner, M. Unger, T. Pock, C. Leistner, A. Saffari, and H. Bischof. Interactive texture segmentation using random forests and total variation. In *BMVC*. BMVA, 2009. 2

[28] T. Schaul, J. Quan, I. Antonoglou, and D. Silver. Prioritized experience replay. In *ICLR*, 2016. 2

[29] H. Van Hasselt, A. Guez, and D. Silver. Deep reinforcement learning with double q-learning. In *AAAI*, 2016. 2, 5

[30] O. Veksler. Star shape prior for graph-cut image segmentation. In *ECCV*. Springer, 2008. 2

[31] V. Vezhnevets and V. Konouchine. Growcut: Interactive multi-label nd image segmentation by cellular automata. In *Graphicon*, 2005. 2

[32] Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, and N. De Freitas. Dueling network architectures for deep reinforcement learning. In *ICML*, 2016. 2, 5

[33] J. Wu, Y. Zhao, J.-Y. Zhu, S. Luo, and Z. Tu. Milcut: A sweeping line multiple instance learning paradigm for interactive image segmentation. In *CVPR*. IEEE, 2014. 2

[34] N. Xu, B. Price, S. Cohen, J. Yang, and T. S. Huang. Deep interactive object selection. In *CVPR*. IEEE, 2016. 2, 6, 7, 8

[35] S. Yeung, O. Russakovsky, G. Mori, and L. Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *CVPR*. IEEE, 2016. 2

[36] S. Yun, J. Choi, Y. Yoo, K. Yun, and J. Young Choi. Action-decision networks for visual tracking with deep reinforcement learning. In *CVPR*. IEEE, 2017. 2