f-BRS: Rethinking Backpropagating Refinement for Interactive Segmentation

Konstantin Sofiiuk   Ilia Petrov   Olga Barinova   Anton Konushin

{k.sofiiuk, ilia.petrov, o.barinova, a.konushin}@samsung.com

Samsung AI Center – Moscow

Abstract

Deep neural networks have become a mainstream approach to interactive segmentation. As we show in our experiments, while for some images a trained network provides accurate segmentation result with just a few clicks, for some unknown objects it cannot achieve satisfactory result even with a large amount of user input. Recently proposed backpropagating refinement scheme (BRS) [15] introduces an optimization problem for interactive segmentation that results in significantly better performance for the hard cases. At the same time, BRS requires running forward and backward pass through a deep network several times that leads to significantly increased computational budget per click compared to other methods. We propose f-BRS (feature backpropagating refinement scheme) that solves an optimization problem with respect to auxiliary variables instead of the network inputs, and requires running forward and backward passes just for a small part of a network. Experiments on GrabCut, Berkeley, DAVIS and SBD datasets set new state-of-the-art at an order of magnitude lower time per click compared to original BRS [15]. The code and trained models are available at https://github.com/saic-vul/fbrs_interactive_segmentation.

1. Introduction

The development of robust models for visual understanding is tightly coupled with data annotation. For instance, one self-driving car can produce about 1Tb of data every day. Due to constant changes in environment new data should be annotated regularly.

Object segmentation provides fine-grained scene representation and can be useful in many applications, *e.g.* autonomous driving, robotics, medical image analysis, etc. However, practical use of object segmentation is now limited due to extremely high annotation costs. Several large segmentation benchmarks [3, 12] with millions of annotated object instances came out recently. Annotation of these datasets became feasible with the use of automated inter-

active segmentation methods [1, 3].

Interactive segmentation has been a topic of research for a long time [26, 10, 11, 13, 2, 31, 18, 22, 15]. The main scenario considered in the papers is click-based segmentation when the user provides input in a form of positive and negative clicks. Classical approaches formulate this task as an optimization problem [4, 10, 11, 13, 2]. These methods have many built-in heuristics and do not use semantic priors to full extent, thus requiring a large amount of input from the user. On the other hand, deep learning-based methods [31, 18, 22] tend to overuse image semantics. While showing great results on the objects that were present in the training set, they tend to perform poorly on unseen object classes. Recent works propose different solutions to these problems [19, 18, 21]. Still, state-of-the-art networks for interactive segmentation are either able to accurately segment the object of interest after a few clicks or do not provide satisfactory result after any reasonable number of clicks (see Section 5.1 for experiments).

The recently proposed backpropagating refinement scheme (BRS) [15] brings together optimization-based and deep learning-based approaches to interactive segmentation. BRS enforces the consistency of the resulting object mask with user-provided clicks. The effect of BRS is based on the fact that small perturbations of the inputs for a deep network can cause massive changes in the network output [29]. Thus, BRS requires running forward and backward pass multiple times through the whole model, which substantially increases computational budget per click compared to other methods and is not practical for many end-user scenarios.

In this work we propose *f-BRS (feature backpropagating refinement scheme)* that reparameterizes the optimization problem and thus requires running forward and backward passes only through a small part of the network (*i.e.* last several layers). Straightforward optimization for activations in a small sub-network would not lead to the desired effect because the receptive field of the convolutions in the last layers relative to the output is too small. Thus we introduce a set of auxiliary parameters for optimization that are invariant to the position in the image. We show that op-
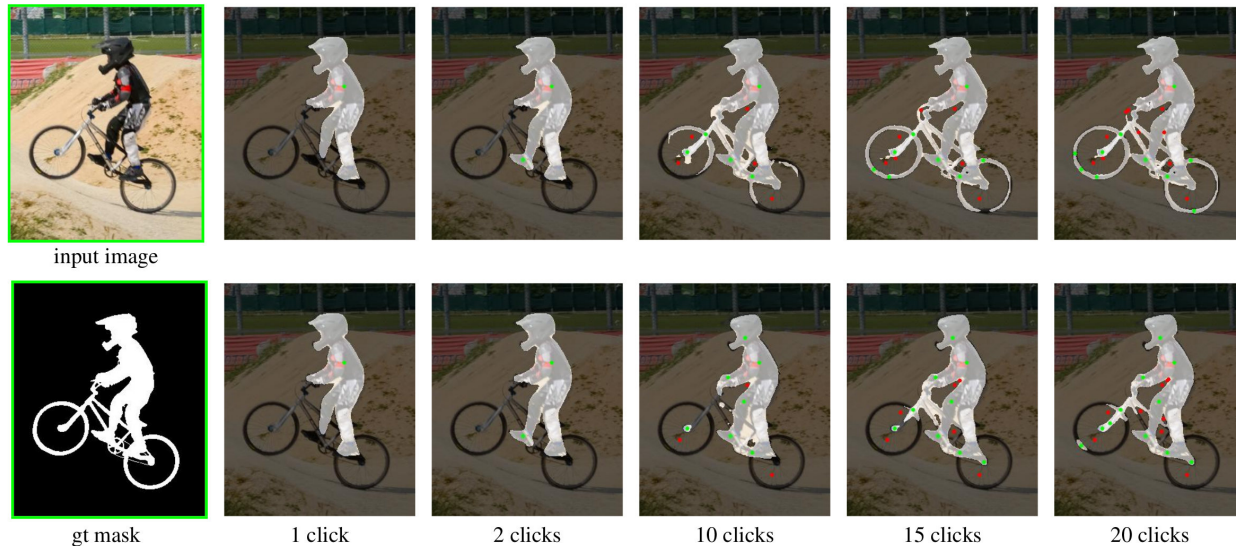
Figure 1. Results of interactive segmentation on an image from DAVIS dataset. First row: using proposed f-BRS-B (Section 3), second row: without BRS. Green dots denote positive clicks, red dots denote negative clicks.

timization with respect to these parameters leads to a similar effect as the original BRS, without the need to compute backward pass through the whole network.

We perform experiments on standard datasets: GrabCut [26], Berkeley [23], DAVIS [25] and SBD [13], and show state-of-the-art results, improving over existing approaches in terms of speed and accuracy.

2. Related work

The goal of interactive image segmentation is to obtain an accurate mask of an object using minimal user input. Most of the methods assume such interface where a user can provide positive and negative clicks (seeds) several times until the desired object mask is obtained.

Optimization-based methods. Before deep learning, interactive segmentation was usually posed as an optimization problem. Li *et al.* [17] used graph-cut algorithm to separate foreground pixels from the background using distances from each pixel to foreground and background seeds in color space. Grady *et al.* [10] proposed a method based on random walks, where each pixel is marked according to the label of the first seed that the walker reaches. Later, [11] compute geodesic distances from the clicked points to every image pixel and use them in energy minimisation. In [16], several segmentation maps are first generated for an image. Then optimization algorithm is applied to the cost function that enforces pixels of the same segment to have the same label in the resulting segmentation mask.

Optimization-based methods usually demonstrate predictable behaviour and allow obtaining detailed segmentation masks with enough user input. Since no learning is involved, the amount of input required from a user does not depend on the type of an object of interest. The main drawback of this approach is insufficient use of semantic priors. This requires additional user effort to obtain accurate object masks for known objects compared to recently proposed learning-based methods.

Learning-based methods. The first deep learning-based interactive segmentation method was proposed in [31]. They calculate distance maps from positive and negative clicks, stack them together with an input image and pass into a network that predicts an object mask. This approach was later used in most of the following works. Liew *et al.* [19] propose to combine local predictions on patches containing user clicks and thus refine network output. Li *et al.* [18] notice that learnt models tend to be overconfident in their predictions. In order to improve diversity of the outputs, they generate multiple masks and then select one among them. In [28], user annotations are multiplied automatically by locating foreground and background clicks.

The common problem of all deep-learning-based methods for interactive segmentation is overweighting semantics and making little use of user-provided clicks. This happens because during training user clicks are in perfect correspondence with the semantics of the image and add little information, therefore can be easily downweighted during the training process.

Optimization for activations. Optimization schemes that update activation responses while keeping weights of a neural network fixed have been used for different problems [27, 32, 33, 8, 9]. Szegedy *et al.* [29] formulated an optimization problem for generating adversarial examples, *i.e.* images that are visually indistinguishable from the natural ones, though are incorrectly classified by the network
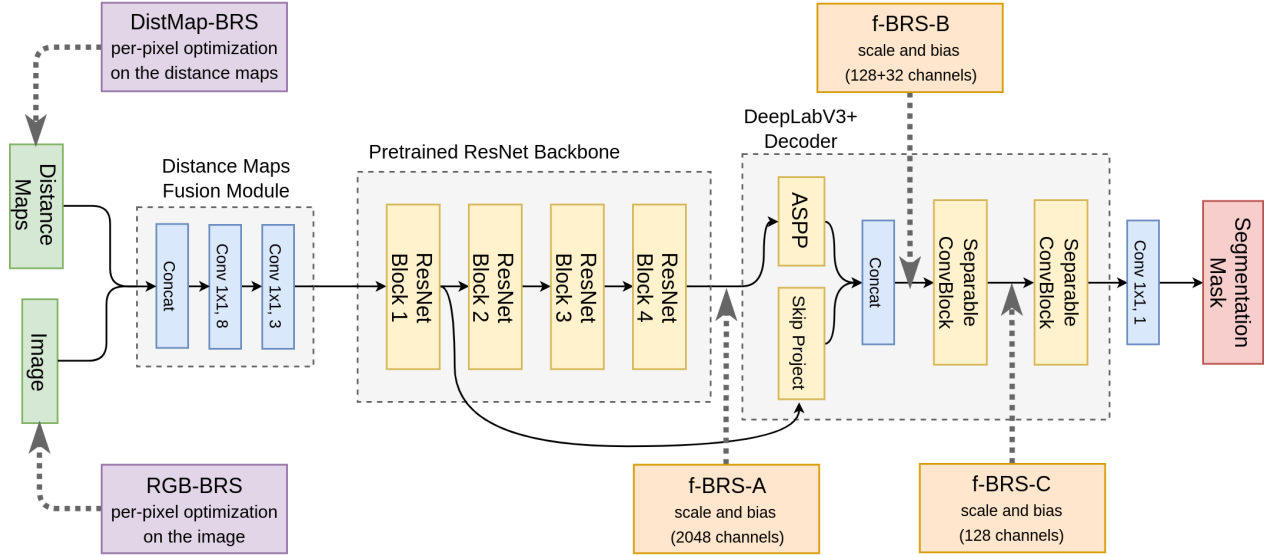
Figure 2. Illustration of the proposed method described in Section 3. f-BRS-A optimizes scale and bias for the features after pre-trained backbone, f-BRS-B optimizes scale and bias for the features after ASPP, f-BRS-C optimizes scale and bias for the features after the first separable convblock. The number of channels is provided for ResNet-50 backbone.

with high confidence. They demonstrated that in deep networks small perturbation of an input signal can cause large changes in activations of the last layers. In [15], the authors apply this idea to the problem of interactive segmentation. They find minimal edits to the input distance maps that result in an object mask consistent with user-provided annotation.

In this work, we also formulate an optimization problem for interactive segmentation. In contrast to [15], here we do not perform optimization over the network inputs but introduce an auxiliary set of parameters for optimization. After such reparameterization, we do not need to run forward and backward passes through the whole network. We evaluate different reparameterizations and the speed and accuracy of the resulting methods. The derived optimization algorithm f-BRS is an order of magnitude faster than BRS from [15].

3. Proposed method

First, let us recall optimization problems from the literature, where the fixed network weights were used. Below we use a unified notation. We denote the space of input images by $\mathbb{R}^m$ and a function that a deep neural network implements by f.

3.1. Background

Adversarial examples generation. Szegedy *et al.* [29] formulate an optimization problem for generating adversarial examples for an image classification task. They find images that are visually indistinguishable from the natural ones, which are incorrectly classified by the network. Let $\mathcal{L}$ denote a continuous loss function that penalizes for incor-

rect classification of an image. For a given image $x \in R^m$ and target label $l \in \{1 \ldots k\}$, they aim to find $x + \Delta x$, which is the closest image to x classified as l by f. For that they solve the following optimization problem:

$$
\begin{aligned}
||\Delta x||_2 \to \min \quad &\text{subject to} \\
1. &\, f(x + \Delta x) = l \\
2. &\, x + \Delta x \in [0, 1]^m
\end{aligned}
\tag{1}
$$

This problem in (1) is reduced to minimisation of the following energy function:

$$
\lambda ||\Delta x||_2 + \mathcal{L}(f(x + \Delta x), l) \to \min_{\Delta x}
\tag{2}
$$

The variable λ in later works is usually assumed a constant and serves as a trade-off between the two energy terms.

Backpropagating refinement scheme for interactive segmentation. Jang *et al.* [15] propose a backpropagating refinement scheme that applies a similar optimization technique to the problem of interactive image segmentation. In their work, a network takes as input an image stacked together with distance maps for user-provided clicks. They find minimal edits to the distance maps that result in an object mask consistent with user-provided annotation. For that, they minimise a sum of two energy functions, *i.e.* corrective energy and inertial energy. Corrective energy function enforces consistency of the resulting mask with user-provided annotation, and inertial energy prevents excessive perturbations in the network inputs.

Let us denote the coordinates of user-provided click by

(u, v) and its label (positive or negative) as $l \in \{0, 1\}$. Let us denote the output of a network f for an image x in position (u, v) as $f(x)_{u,v}$ and the set of all user-provided clicks as $\{(u_i, v_i, l_i)\}_{i=1}^n$. The optimization problem in [15] is formulated as follows:

$$\lambda||\Delta x||_2 + \sum_{i=1}^n \left(f(x + \Delta x)_{u_i, v_i} - l_i \right)^2 \to \min_{\Delta x}, \quad (3)$$

where the first term represents inertial energy, the second term represents corrective energy and λ is a constant that regulates trade-off between the two energy terms. This optimization problem resembles the one from (2) with classification loss for one particular label replaced by a sum of losses for the labels of all user-provided clicks. Here we do not need to ensure that the result of optimization is a valid image, so the energy (3) can be minimised by unconstrained L-BFGS.

The main drawback of this approach is that L-BFGS requires computation of gradients with respect to network inputs, *i.e.* backpropagating through the whole network. It is computationally expensive and results in significant computational overhead.

We also notice that since the first layer of a network f is a convolution, *i.e.* a linear combination of the inputs, one can minimise the energy (3) with respect to input image instead of distance maps and obtain equivalent solution. Moreover, if we minimise it with respect to RGB image which is invariant to an interactive input, we can use the result as an initialisation for optimization of (3) with new clicks. Thus, we set the BRS with respect to an input image as a baseline in our experiments and denote it as RGB-BRS. For a fair comparison, we also implement the optimization with respect to the input distance maps (DistMap-BRS), that was originally introduced in [15].

3.2. Feature backpropagating refinement

In order to speed-up the optimization process, we want to compute backpropagation not for the whole network, but for some part of it. This can be achieved by optimizing some intermediate parameters in the network instead of the input. A naive approach would be to simply optimize the outputs of some of the last layers and thus compute backpropagation only through the head of a network. However, such a naive approach would not lead to the desired result. The convolutions in the last layers have a very small receptive field with respect to the network outputs. Therefore, an optimization target can be easily achieved by changing just a few components of a feature tensor which would cause only minor localized changes around the clicked points in the resulting object mask.

Let us reparameterize the function f and introduce auxiliary variables for optimization. Let $\hat{f}(x, z)$ denote the function that depends both on the input x and on the introduced

variables z. With auxiliary parameters fixed $z = p$ the reparameterized function is equivalent to the original one $\hat{f}(x, p) \equiv f(x)$. Thus, we aim to find a small value of Δp, which would bring the values of $\hat{f}(x, p + \Delta p)$ in the clicked points close to the user-provided labels. We formulate the optimization problem as follows:

$$\lambda||\Delta p||_2 + \sum_{i=1}^n \left(\hat{f}(x, p + \Delta p)_{u_i, v_i} - l_i \right)^2 \to \min_{\Delta p}. \quad (4)$$

We call this optimization task *f-BRS (feature backpropagating refinement)* and use unconstrained L-BFGS optimizer for minimization. For f-BRS to be efficient, we need to choose the reparameterization that a) does not have a localized effect on the outputs, b) does not require a backward pass through the whole network for optimization.

One of the options for such reparameterization may be channel-wise scaling and bias for the activations of the last layers in the network. Scale and bias are invariant to the position in the image, thus changes in this parameters would affect the results globally. Compared to optimization with respect to activations, optimization with respect to scale and bias cannot result in degenerate solutions (*i.e.* minor localized changes around the clicked points).

Let us denote the output of some intermediate layer of the network for an image x by $F(x)$, the number of its channels by h, a function that the network head implements by g. Thus, f can be represented by $f(x) \equiv g(F(x))$. Then the reparameterized function $\hat{f}$ looks as follows:

$$\hat{f}(x, s, b) = g\left(s \cdot F(x) + b \right), \quad (5)$$

where $b \in \mathbb{R}^h$ is a vector of biases, $s \in \mathbb{R}^h$ is a vector of scaling coefficients and $\cdot$ denotes a channel-wise multiplication. For $s = \mathbf{1}$ and $b = \mathbf{0}$ we have $\hat{f}(x) \equiv f$, thus we take these values as initial values for optimization.

By varying the part of the network to which auxiliary scale and bias are applied, we achieve a natural trade-off between accuracy and speed. Figure 2 shows the architecture of the network that we used in this work and illustrates different options for optimization. Surprisingly, we found that applying f-BRS to the last several layers causes just a small drop of accuracy compared to full-network BRS, and leads to significant speed-up.

4. Zoom-In for interactive segmentation

Previous works on interactive segmentation often used inference on image crops to achieve speed-up and preserve fine details in the segmentation mask. Cropping helps to infer the masks of small objects, but it also may degrade results in cases when an object of interest is too large to fit into one crop.

In this work, we use an alternative technique (we call it *Zoom-In*), which is quite simple but improves both quality
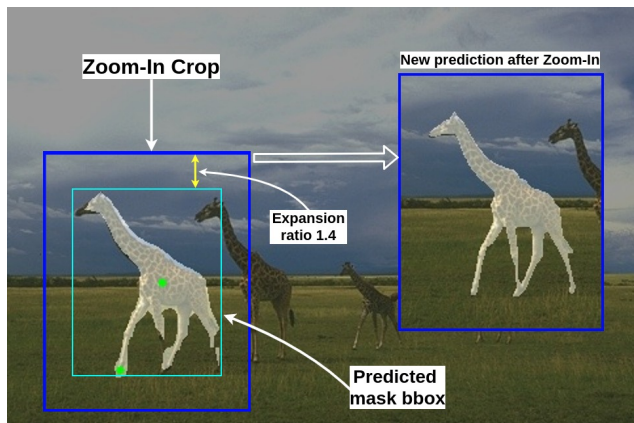
Figure 3. Example of applying zoom-in technique described in Section 4. See how cropping an image allows recovering fine details in the segmentation mask.

and speed of the interactive segmentation. It is based on the ideas from object detection [20, 7]. We have not found any mentions of this exact technique in the literature in the context of interactive segmentation, so we describe it below.

We noticed that the first 1-3 clicks are enough for the network to achieve around 80% IoU with ground truth mask in most cases. It allows us to obtain a rough crop around the region of interest. Therefore, starting from the third click we crop an image according to the bounding box of the inferred object mask and apply the interactive segmentation only to this Zoom-In area. We extend the bounding box by 40% along sides in order to preserve the context and not miss fine details on the boundary. If a user provides a click outside the bounding box, we expand or narrow down the zoom-in area. Then we resize the bounding box so that its longest side matches 400 pixels. Figure 3 shows an example of Zoom-In.

This technique helps the network to predict more accurate masks for small objects. In our experiments, Zoom-In consistently improved the results, therefore we used it by default in all experiments in this work. Table 1 shows a quantitative comparison of the results with and without Zoom-In on GrabCut and Berkeley datasets.

5. Experiments

Following the standard experimental protocol, we evaluate proposed method on the following datasets: SBD [13], GrabCut [26], Berkeley [23] and DAVIS [25].

GrabCut. The GrabCut dataset contains 50 images with a single object mask for each image.

Berkeley. For the Berkeley dataset, we use the same test set as in [24], which includes 96 images with 100 object masks for testing.

DAVIS. The DAVIS dataset is used for evaluating video segmentation algorithms. To evaluate interactive segmen-

tation algorithms one can sample random frames from the videos. We use the same 345 individual frames from video sequences as [15] for evaluation. To follow the evaluation protocol we combine instance-level object masks into one semantic segmentation mask for each image.

SBD. The SBD dataset was first used for evaluating object segmentation techniques in [31]. The dataset contains 8,498 training images and 2,820 test images. As in previous works, we train the models on the training part and use the validation set, which includes 6,671 instance-level object masks, for the performance evaluation.

Evaluation protocol. We report the Number of Clicks (NoC) measure which counts the average number of clicks required to achieve a target intersection over union (IoU) with ground truth mask. We set the target IoU score to 85% or 90% for different datasets, denoting the corresponding measures as NoC@85 and NoC@90 respectively. For a fair comparison, we use the same click generation strategy as in [18, 31] that operates as follows. It finds the dominant type of prediction errors (false positives or false negatives) and generates the next negative or positive click respectively at the point farthest from the boundaries of the corresponding error region.

Network architecture. In this work, we do not focus on network architecture improvements, so in all our experiments we use the standard DeepLabV3+ [5] which is a state-of-the-art model for semantic segmentation. The architecture of our network is shown in Figure 2.

The model contains Distance Maps Fusion (DMF) block for adaptive fusion of RGB image and distance maps. It takes a concatenation of RGB image and 2 distance maps (one for positive clicks and one for negative clicks) as an input. The DMF block processes the 5-channel input with 1×1 convolutions followed by LeakyReLU and outputs a 3-channel tensor which can be passed into the backbone that was pre-trained on RGB images.

Implementation details. We formulate the training task as a binary segmentation problem and use binary cross-entropy loss for training. We train all the models on image crops of size 320×480 with horizontal and vertical flips as augmentations. We randomly resize images from 0.75 to 1.25 of original size before cropping.

We sample clicks during training following the standard procedure first proposed in [31]. We set the maximum number of positive and negative clicks to 10, resulting in a maximum of 20 clicks per image.

In all experiments, we used Adam with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and trained the networks for 120 epochs (100 epochs with learning rate 5×10^{-4}, last 20 epochs with learning rate 5×10^{-5}). The batch size was set to 28 and we used synchronous BatchNorm for all experiments. We trained ResNet-34 and ResNet-50 on 2 GPUs (Tesla P40) and ResNet-101 was trained on 4 GPUs (Tesla P40). The

Method	GrabCut		Berkeley	
	w/o ZI	ZI	w/o ZI	ZI
Ours w/o BRS	3.42	3.32	7.13	5.18
Ours f-BRS-B	2.68	2.98	5.69	4.34

Table 1. Evaluation of the proposed methods with ResNet-50 backbone with and without Zoom-In (ZI) on GrabCut and Berkeley datasets using NoC@90 (see Section 5).

Data	Model	#images ≥ 20	#images ≥ 100	$NoC_{100}@90$
Berkeley	[15] w/o BRS	32	31	33.24
	[15] BRS	10	2	8.77
	Ours w/o BRS	12	9	12.98
	Ours w f-BRS-B	**2**	**0**	**4.47**
DAVIS	[15] w/o BRS	166	157	47.95
	[15] BRS	**77**	51	20.98
	Ours w/o BRS	92	81	27.58
	Ours w f-BRS-B	78	**50**	**20.7**
SBD	Ours w/o BRS	1650	1114	23.18
	Ours w f-BRS-B	1466	**265**	**14.98**

Table 2. Convergence analysis on Berkeley, SBD and DAVIS datasets. We report the number of images that were not correctly segmented after 20 and 100 clicks and the $NoC_{100}@90$ performance measure.
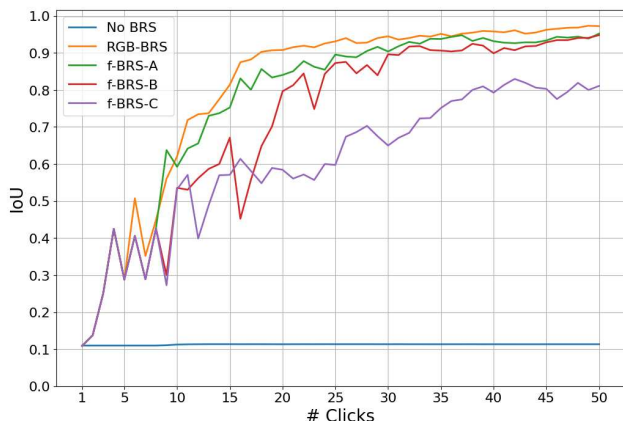

Figure 4. IoU with respect to the number of clicks added by a user for one of the most difficult image from GrabCut dataset (scissors). All results are obtained using the same model with ResNet-50. One can see that without BRS the model does not converge to the correct results.

learning rate for the pre-trained ResNet backbone was 10 times lower than the learning rate for the rest of the network. We set the value of λ to 10^{-3} for RGB-BRS and to 10^{-4} for all variations of f-BRS.

We use MXNet Gluon [6] with GluonCV [14] framework for training and inference of our models. We take pretrained models for ResNet-34, ResNet-50 and ResNet-101 from GluonCV Model Zoo.

5.1. Convergence analysis

An ideal interactive segmentation method should demonstrate predictable performance even for unseen object categories or unusual demand from the user. Moreover, the hard cases that require a significant amount of user input are the most interesting in the data annotation scenario. Thus, the desired property of an interactive segmentation method is convergence, *i.e.* we expect the result to improve with adding more clicks and finally achieve satisfactory accuracy.

However, neither the training procedure nor the inference in feed-forward networks for interactive segmentation guarantee convergence. Therefore, we noticed that when using feed-forward networks, the result does not converge for a significant number of images, *i.e.* additional user clicks do not improve the resulting segmentation mask. An example of such behaviour can be found in Figure 4. We observe very similar behaviour with different network architectures, namely with an architecture from [15] and with DeepLabV3+. Below we describe our experiments.

Motivation for using NoC_{100} metric. Previous works usually report NoC with the maximum number of generated clicks limited to 20 (we simply call this metric NoC). However, for a large portion of images in the standard datasets, this limit is exceeded. In terms of NoC, images that require 20 clicks and 2000 clicks to obtain accurate masks will get the same penalty. Therefore, NoC does not distinguish between the cases where an interactive segmentation method requires slightly more user input to converge and the cases where it fails to converge (i.e. unable to achieve satisfactory results after any reasonable number of user clicks).

In the experiments below we analyse NoC with the maximum number of clicks limited to 100 (let us call this metric NoC_{100}). NoC_{100} is better for the convergence analysis, allowing us to identify the images where interactive segmentation fails. We believe that NoC_{100} is substantially more adequate for comparison of interactive segmentation methods than NoC.

Experiments and discussion. In Table 2 we report the number of images that were not correctly segmented even after 20 and 100 clicks, and NoC_{100} for the target IoU=90% ($NoC_{100}@90$).

One can see that both DeepLabV3+ and the network architecture from [15] without BRS were unable to produce accurate segmentation results on a relatively large portion of images from all datasets even with 100 user clicks provided. Interestingly, this percentage is also high for the SBD dataset which has the closest distribution to the training set. The images that could not be segmented with 100 user clicks are clear failure cases for the method. The use of both original BRS and proposed f-BRS allows to reduce the number of such cases by several times and results in significant improvement in terms of NoC_{100}.
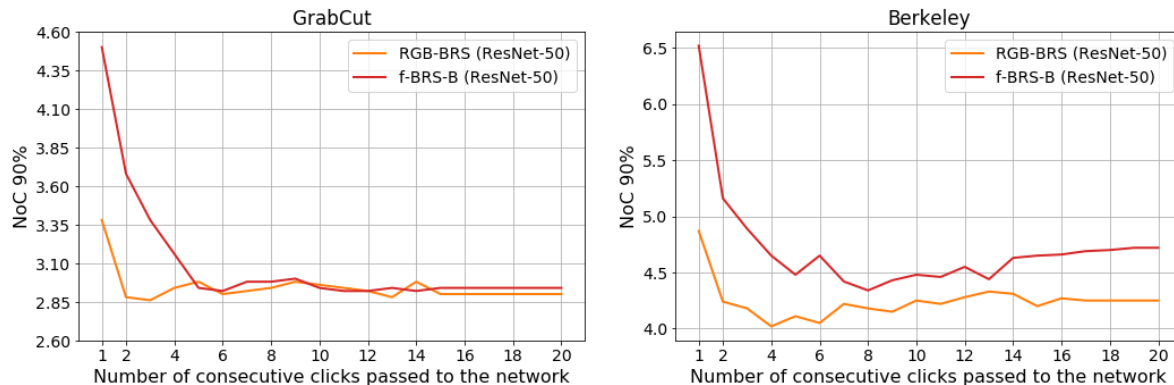
Figure 5. Evaluation of different click-processing strategies on GrabCut and Berkeley datasets. The plots show NoC@90 with respect to the number of consecutive clicks passed to the network.

We believe that the use of optimization-based backpropagating refinement results not just in metrics improvement, but more importantly, it changes the behaviour of the interactive segmentation system and its convergence properties.

5.2. Evaluation of the importance of clicks passed to the network

We have noticed that the results do not always improve with the increasing number of clicks passed to the network. Moreover, too many clicks can cause unpredictable behaviour of the network. On the other hand, the formulation of the optimization task for backpropagating refinement enforces the consistency of the resulting mask with user-provided annotation.

One may notice that we can handle user clicks only as a target for BRS loss function without passing them to the network through distance maps. We initialise the state of the network by making a prediction with the first few clicks. Then we iteratively refine the resulting segmentation mask only with BRS according to the new clicks.

We studied the relation between the number of consecutive clicks passed to the network and resulting NoC@90 on GrabCut and Berkeley datasets. The results of this study for RGB-BRS and f-BRS-B are shown in Figure 5. The results show that providing all clicks to the network is not an optimal strategy. It is clear that for RGB-BRS, the optimum is achieved by limiting the number of clicks to 4, and for f-BRS-B – by 8 clicks. This shows that both BRS and f-BRS can adapt the network output to user input, without explicitly passing clicks to the network.

In all other experiments, we have limited the number of clicks passed to the network to 8 for f-BRS algorithms and to 4 for RGB-BRS.

5.3. Comparison with previous works

Comparison using the standard protocol. Table 3 compares with previous works across the standard protocol

and report the average NoC with two IoU thresholds: 85% and 90%.

The proposed f-BRS algorithm requires fewer clicks than conventional algorithms, which indicates that the proposed algorithm yields accurate object masks with less user effort.

We tested three backbones on all datasets. Surprisingly, there is no significant difference in performance between these models. The smallest ResNet-34 model shows the best quality on GrabCut dataset outperforming much heavier models such as ResNet-101. However, during training there was a significant difference in the values of the target loss function on the validation set between these models. This shows that the target loss function is poorly correlated with the NoC metric.

Running time analysis. We measure the average running time of the proposed algorithm in seconds per click (SPC) and measure the total running time to process a dataset. The SPC shows the delay between placing a click and getting the updated result. The second metric indicates the total time a user needs to spend to obtain a satisfactory image annotation. In these experiments, we set the threshold on the number of clicks per image to 20. We test it on Berkeley and DAVIS datasets using a PC with an AMD Ryzen Threadripper 1900X CPU and a GTX 1080 Ti GPU.

Table 4 shows the results for different versions of the proposed method and for our implemented baselines: without BRS and with RGB-BRS. The running time of f-BRS is an order of magnitude lower compared to RGB-BRS and adds just a small overhead with respect to a pure feedforward model.

5.4. Comparison of different versions of f-BRS.

The choice of a layer where to introduce auxiliary variables provides a trade-off between speed and accuracy of f-BRS. We compare three options of applying scale and bias to intermediate outputs in different parts of the network: af-

Method	GrabCut		Berkeley	SBD		DAVIS	
	NoC@85	NoC@90	NoC@90	NoC@85	NoC@90	NoC@85	NoC@90
Graph cut [4]	7.98	10.00	14.22	13.6	15.96	15.13	17.41
Geodesic matting [11]	13.32	14.57	15.96	15.36	17.60	18.59	19.50
Random walker [10]	11.36	13.77	14.02	12.22	15.04	16.71	18.31
Euclidean star convexity [11]	7.24	9.20	12.11	12.21	14.86	15.41	17.70
Geodesic star convexity [11]	7.10	9.12	12.57	12.69	15.31	15.35	17.52
Growcut [30]	–	16.74	18.25	–	–	–	–
DOS w/o GC [31]	8.02	12.59	–	14.30	16.79	12.52	17.11
DOS with GC [31]	5.08	6.08	–	9.22	12.80	9.03	12.58
Latent diversity [18]	3.20	4.79	–	7.41	10.78	**5.05**	9.57
RIS-Net [19]	–	5.00	–	6.03	–	–	–
CM guidance [21]	–	3.58	5.60	–	–	–	–
BRS [15]	2.60	3.60	5.08	6.59	9.78	5.58	8.24
Ours w/o BRS ResNet-34	2.52	3.20	5.31	5.51	8.58	5.47	8.51
Ours w/o BRS ResNet-50	2.64	3.32	5.18	5.10	<u>8.01</u>	5.39	8.18
Ours w/o BRS ResNet-101	2.50	3.18	6.25	5.28	8.13	<u>5.12</u>	8.01
Ours f-BRS-B ResNet-34	**2.00**	**2.46**	4.65	5.25	8.30	5.39	8.21
Ours f-BRS-B ResNet-50	2.50	2.98	**4.34**	<u>5.06</u>	8.08	5.39	<u>7.81</u>
Ours f-BRS-B ResNet-101	<u>2.30</u>	<u>2.72</u>	<u>4.57</u>	**4.81**	**7.73**	**5.04**	**7.41**

Table 3. Evaluation results of GrabCut, Berkeley, SBD and DAVIS datasets. The best and the second best results are written in bold and underlined respectively.

Method	Berkeley				Davis			
	NoC@90	#images ≥20	SPC	Time, s	NoC@90	#images ≥20	SPC	Time, s
Ours w/o BRS	5.18	12	0.091	49.9	8.18	92	0.21	585.9
Ours RGB-BRS	4.08	4	1.117	455.7	7.58	72	2.89	7480.8
Ours DistMap-BRS	4.17	4	0.669	276.4	7.93	73	1.47	4028.4
Ours f-BRS-A	4.36	3	0.281	119.3	7.54	72	0.75	1980.5
Ours f-BRS-B	4.34	2	0.132	55.07	7.81	78	0.32	889.4
Ours f-BRS-C	4.91	8	0.138	61.4	7.91	84	0.31	848.2

Table 4. Comparison of the results without BRS and with f-BRS types A, B and C with ResNet-50 backbone.

ter the backbone (f-BRS-A), before the first separable convolutions block in DeepLabV3+ (f-BRS-B), and before the second separable convolutions block in DeepLabV3+ (f-BRS-C). As a baseline for our experiments, we report the results for a feed-forward network without BRS. We also implement RGB-BRS, employing the optimization with respect to an input image. In these experiments, we used the ResNet-50 backbone.

We report NoC@90 and the number of images where the satisfactory result was not obtained after 20 user clicks. We also measure SPC (seconds per click) and Time (total time to process a dataset). Notice that direct comparison of the timings with the numbers reported in previous works is not valid due to differences in used frameworks and hardware. Therefore, only relative comparison makes sense.

The results of the evaluation for Berkeley and DAVIS datasets are shown in Table 4. One can notice that all versions of f-BRS perform better than the baseline without BRS. The f-BRS-B is about 8 times faster than the RGB-

BRS while showing very close results in terms of NoC. Therefore, we chose it for the comparative experiments.

6. Conclusions

We proposed a novel backpropagating refinement scheme (f-BRS) that operates on intermediate features in the network and requires running forward and backward passes just for a small part of a network. Our approach was evaluated on four standard interactive segmentation benchmarks and set new state-of-the-art results in terms of both accuracy and speed. The conducted experiments demonstrated a better convergence of backpropagating refinement schemes compared to pure feed-forward approaches. We analysed the importance of first clicks passed to the network and showed that both BRS and f-BRS can successfully adapt the network output to user input, without explicitly passing clicks to the network.

References

[1] Eirikur Agustsson, Jasper RR Uijlings, and Vittorio Ferrari. Interactive full image segmentation by considering all regions jointly. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11622–11631, 2019. 1

[2] Junjie Bai and Xiaodong Wu. Error-tolerant scribbles based interactive image segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 392–399, 2014. 1

[3] Rodrigo Benenson, Stefan Popov, and Vittorio Ferrari. Large-scale interactive object segmentation with human annotators. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11700–11709, 2019. 1

[4] Yuri Y Boykov and M-P Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *Proceedings eighth IEEE international conference on computer vision. ICCV 2001*, volume 1, pages 105–112. IEEE, 2001. 1, 8

[5] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018. 5

[6] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274*, 2015. 6

[7] Mingfei Gao, Ruichi Yu, Ang Li, Vlad I Morariu, and Larry S Davis. Dynamic zoom-in network for fast object detection in large images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6926–6935, 2018. 5

[8] Leon Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. In *Advances in neural information processing systems*, pages 262–270, 2015. 2

[9] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016. 2

[10] Leo Grady. Random walks for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 28(11):1768–1783, 2006. 1, 2, 8

[11] Varun Gulshan, Carsten Rother, Antonio Criminisi, Andrew Blake, and Andrew Zisserman. Geodesic star convexity for interactive image segmentation. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3129–3136. IEEE, 2010. 1, 2, 8

[12] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5356–5364, 2019. 1

[13] Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *2011 International Conference on Computer Vision*, pages 991–998. IEEE, 2011. 1, 2, 5

[14] Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of tricks for image classification with convolutional neural networks. *arXiv preprint arXiv:1812.01187*, 2018. 6

[15] Won-Dong Jang and Chang-Su Kim. Interactive image segmentation via backpropagating refinement scheme. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5297–5306, 2019. 1, 3, 4, 5, 6, 8

[16] Tae Hoon Kim, Kyoung Mu Lee, and Sang Uk Lee. Nonparametric higher-order learning for interactive segmentation. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3201–3208. IEEE, 2010. 2

[17] Yin Li, Jian Sun, Chi-Keung Tang, and Heung-Yeung Shum. Lazy snapping. *ACM Transactions on Graphics (ToG)*, 23(3):303–308, 2004. 2

[18] Zhuwen Li, Qifeng Chen, and Vladlen Koltun. Interactive image segmentation with latent diversity. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 577–585, 2018. 1, 2, 5, 8

[19] JunHao Liew, Yunchao Wei, Wei Xiong, Sim-Heng Ong, and Jiashi Feng. Regional interactive image segmentation networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2746–2754. IEEE, 2017. 1, 2, 8

[20] Yongxi Lu, Tara Javidi, and Svetlana Lazebnik. Adaptive object detection using adjacency and zoom prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2351–2359, 2016. 5

[21] Soumajit Majumder and Angela Yao. Content-aware multi-level guidance for interactive instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11602–11611, 2019. 1, 8

[22] Kevis-Kokitsi Maninis, Sergi Caelles, Jordi Pont-Tuset, and Luc Van Gool. Deep extreme cut: From extreme points to object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 616–625, 2018. 1

[23] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 416–423. IEEE, 2001. 2, 5

[24] Kevin McGuinness and Noel E O'connor. A comparative evaluation of interactive segmentation algorithms. *Pattern Recognition*, 43(2):434–444, 2010. 5

[25] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 724–732, 2016. 2, 5

[26] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. " grabcut" interactive foreground extraction using iterated graph cuts. *ACM transactions on graphics (TOG)*, 23(3):309–314, 2004. 1, 2, 5

[27] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013. 2

[28] Gwangmo Song, Heesoo Myeong, and Kyoung Mu Lee. Seednet: Automatic seed generation with deep reinforcement learning for robust interactive segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1760–1768, 2018. 2

[29] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 1, 2, 3

[30] Vladimir Vezhnevets and Vadim Konouchine. Growcut: Interactive multi-label nd image segmentation by cellular automata. In *proc. of Graphicon*, volume 1, pages 150–156. Citeseer, 2005. 8

[31] Ning Xu, Brian Price, Scott Cohen, Jimei Yang, and Thomas S Huang. Deep interactive object selection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 373–381, 2016. 1, 2, 5, 8

[32] Qiong Yan, Li Xu, Jianping Shi, and Jiaya Jia. Hierarchical saliency detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1155–1162, 2013. 2

[33] Jianming Zhang, Sarah Adel Bargal, Zhe Lin, Jonathan Brandt, Xiaohui Shen, and Stan Sclaroff. Top-down neural attention by excitation backprop. *International Journal of Computer Vision*, 126(10):1084–1102, 2018. 2