

SimpleClick: Interactive Image Segmentation with Simple Vision Transformers

Qin Liu, Zhenlin Xu, Gedas Bertasius, Marc Niethammer
University of North Carolina at Chapel Hill

<https://github.com/uncbiag/SimpleClick>

Abstract

Click-based interactive image segmentation aims at extracting objects with a limited user clicking. A hierarchical backbone is the de-facto architecture for current methods. Recently, the plain, non-hierarchical Vision Transformer (ViT) has emerged as a competitive backbone for dense prediction tasks. This design allows the original ViT to be a foundation model that can be finetuned for downstream tasks without redesigning a hierarchical backbone for pretraining. Although this design is simple and has been proven effective, it has not yet been explored for interactive image segmentation. To fill this gap, we propose *SimpleClick*, the first interactive segmentation method that leverages a plain backbone. Based on the plain backbone, we introduce a symmetric patch embedding layer that encodes clicks into the backbone with minor modifications to the backbone itself. With the plain backbone pretrained as a masked autoencoder (MAE), *SimpleClick* achieves state-of-the-art performance. Remarkably, our method achieves **4.15** NoC@90 on SBD, improving **21.8%** over the previous best result. Extensive evaluation on medical images demonstrates the generalizability of our method. We further develop an extremely tiny ViT backbone for *SimpleClick* and provide a detailed computational analysis, highlighting its suitability as a practical annotation tool.

1. Introduction

The goal of interactive image segmentation is to obtain high-quality pixel-level annotations with limited user interaction such as clicking. Interactive image segmentation approaches have been widely applied to annotate large-scale image datasets, which drive the success of deep models in various applications, including video understanding [5, 48], self-driving [7], and medical imaging [31, 40]. Much research has been devoted to explore interactive image segmentation with different interaction types, such as bounding boxes [46], polygons [1], clicks [42], scribbles [44], and their combinations [50]. Among them, the click-based approach is most common due to its simplicity and well-

NoC@90 on SBD dataset

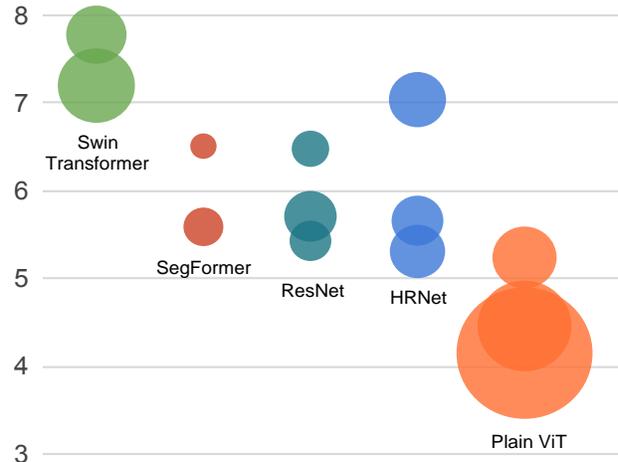


Figure 1. **Interactive segmentation results on SBD [20]**. The metric “NoC@90” denotes the number of clicks required to obtain 90% IoU. The area of each bubble is proportional to the FLOPs of a model variant (Tab. 5). We show that plain ViTs outperform all hierarchical backbones for interactive image segmentation at a moderate computational cost.

established training and evaluation protocols.

Recent advances in click-based approaches mainly lie in two orthogonal directions: 1) the development of more effective backbone networks and 2) the exploration of more elaborate refinement modules built upon the backbone. For the former direction, different hierarchical backbones, including both ConvNets [29, 42] and ViTs [10, 32], have been developed for interactive segmentation. For the latter direction, various refinement modules, including local refinement [10, 29] and click imitation [33], have been proposed to further boost segmentation performance. In this work, we delve into the former direction and focus on exploring a plain backbone for interactive segmentation.

A hierarchical backbone is the predominant architecture for current interactive segmentation methods. This design is deeply rooted in ConvNets, represented by ResNet [22], and has been adopted by ViTs, represented by the Swin Transformer [34]. The motivation for a hierarchical backbone

stems from the locality of convolution operations, leading to insufficient model receptive field size without the hierarchy. To increase the receptive field size, ConvNets have to progressively downsample feature maps to capture more global contextual information. Therefore, they often require a feature pyramid network such as FPN [27] to aggregate multi-scale representations for high-quality segmentation. However, this reasoning no longer applies for a plain ViT, in which global information can be captured from the first self-attention block. Because all feature maps in the ViT are of the same resolution, the motivation for an FPN-like feature pyramid also no longer remains. The above reasoning is supported by a recent finding that a plain ViT can serve as a strong backbone for object detection [25]. This finding indicates a general-purpose ViT backbone might be suitable for other tasks, which then can decouple pretraining from finetuning and transfer the benefits from readily available pretrained ViT models (*e.g.* MAE [21]) to these tasks. However, although this design is simple and has been proven effective, it has not yet been explored in interactive segmentation. In this work, we propose `SimpleClick`, the first plain-backbone method for interactive segmentation. The core of `SimpleClick` is a plain ViT backbone that maintains single-scale representations throughout. We *only* use the last feature map from the plain backbone to build a simple feature pyramid for segmentation, largely decoupling the general-purpose backbone from the segmentation-specific modules. To make `SimpleClick` more efficient, we use a light-weight MLP decoder to transform the simple feature pyramid into a segmentation (see Sec. 3 for details).

We extensively evaluate our method on **10** public benchmarks, including both natural and medical images. With the plain backbone pretrained as a MAE [21], our method achieves **4.15** NoC@90 on SBD, which outperforms the previous best method by **21.8%** without a complex FPN-like design and local refinement. We demonstrate the generalizability of our method by out-of-domain evaluation on medical images. We further analyze the computational efficiency of `SimpleClick`, highlighting its suitability as a practical annotation tool.

Our main contributions are:

- We propose `SimpleClick`, the first plain-backbone method for interactive image segmentation.
- `SimpleClick` achieves state-of-the-art performance on natural images and shows strong generalizability on medical images.
- `SimpleClick` meets the computational efficiency requirement for a practical annotation tool, highlighting its readiness for real-world applications.

2. Related Work

Interactive Image Segmentation Interactive image segmentation is a longstanding problem for which increas-

ingly better solution approaches have been proposed. Early works [6, 16, 18, 39] tackle this problem using graphs defined over image pixels. However, these methods only focus on low-level image features, and therefore tend to have difficulty with complex objects.

Thriving on large datasets, ConvNets [10, 29, 42, 46, 47] have evolved as the dominant architecture for high quality interactive segmentation. ConvNet-based methods have explored various interaction types, such as bounding boxes [46], polygons [1], clicks [42], and scribbles [44]. Click-based approaches are the most common due to their simplicity and well-established training and evaluation protocols. Xu *et al.* [47] first proposed a click simulation strategy that has been adopted by follow-up work [10, 33, 42]. DEXTR [35] extracts a target object from specifying its four extreme points (left-most, right-most, top, bottom pixels). FCA-Net [30] demonstrates the critical role of the first click for better segmentation. Recently, ViTs have been applied to interactive segmentation. FocalClick [10] uses SegFormer [45] as the backbone network and achieves state-of-the-art segmentation results with high computational efficiency. iSegFormer [32] uses a Swin Transformer [34] as the backbone network for interactive segmentation on medical images. Besides the contribution on backbones, some works are exploring elaborate refinement modules built upon the backbone. FocalClick [10] and FocusCut [29] propose similar local refinement modules for high-quality segmentation. PseudoClick [33] proposes a click-imitation mechanism by estimating the next-click to further reduce human annotation cost. Our method differs from all previous click-based methods in its plain, non-hierarchical ViT backbone, enjoying the benefits from readily available pretrained ViT models (*e.g.* MAE [21]).

Vision Transformers for Non-Interactive Segmentation

Recently, ViT-based approaches [17, 24, 43, 45, 49] have shown competitive performance on segmentation tasks compared to ConvNets. The original ViT [13] is a non-hierarchical architecture that only maintains single-scale feature maps throughout. SETR [51] and Segformer [43] use the original ViT as the encoder for semantic segmentation. To allow for more efficient segmentation, the Swin Transformer [34] reintroduces a computational hierarchy into the original ViT architecture using shifted window attention, leading to a highly efficient hierarchical ViT backbone. SegFormer [45] designs hierarchical feature representations based on the original ViT using overlapped patch merging, combined with a light-weight MLP decoder for efficient segmentation. HRViT [17] integrates a high-resolution multi-branch architecture with ViTs to learn multi-scale representations. Recently, the original ViT has been reintroduced as a competitive backbone for semantic segmentation [8] and object detection [25], with the aid of MAE [21] pretraining and window attention. Inspired by

Model\ Module→	ViT Backbone	Conv. Neck	MLP Head
Ours-ViT-B	83.0 (89.3%)	9.0 (9.7%)	0.9 (1.0%)
Ours-ViT-L	290.8 (94.3%)	16.5 (5.3%)	1.1 (0.4%)
Ours-ViT-H	604.0 (95.7%)	25.8 (4.1%)	1.3 (0.2%)

Table 1. **Number of parameters of our models.** The unit is million. The percentage of parameters is shown in bracket. Most parameters are used by the ViT backbone.

this finding, we explore using a plain ViT as the backbone network for interactive segmentation.

3. Method

Our goal is not to propose new modules, but to adapt a plain-ViT backbone for interactive segmentation with *minimal* modifications so as to enjoy the readily available pre-trained ViT weights. Sec. 3.1 introduces the main modules of SimpleClick. Sec. 3.2 describes the training and inference details of our method.

3.1. Network Architecture

Adaptation of Plain-ViT Backbone We use a plain ViT [13] as our backbone network, which only maintains single-scale feature maps throughout. The patch embedding layer divides the input image into non-overlapping fixed-size patches (*e.g.* 16×16 for ViT-B), each patch is flattened and linearly projected to a fixed-length vector (*e.g.* 768 for ViT-B). The resulting sequence of vectors is fed into a queue of Transformer blocks (*e.g.* 12 for ViT-B) for self-attention. We implement SimpleClick with three backbones: ViT-B, ViT-L, and ViT-H (Tab. 1 shows the number of parameters for the three backbones). The three backbones were pre-trained on ImageNet-1k as MAEs [21]. We adapt the pre-trained backbones to higher-resolution inputs during finetuning using non-shifting window attention aided by a few global self-attention blocks (*e.g.* 2 for ViT-B), as introduced in ViTDet [25]. Since the last feature map is subject to all the attention blocks, it should have the strongest representation. Therefore, we only use the last feature map to build a simple multi-scale feature pyramid.

Simple Feature Pyramid For the hierarchical backbone, a feature pyramid is commonly produced by an FPN [27] to combine features from different stages. For the plain backbone, a feature pyramid can be generated in a much simpler way: by a set of parallel convolutional or deconvolutional layers using *only* the last feature map of the backbone. As shown in Fig. 2, given the input ViT feature map, a multi-scale feature map can be produced by four convolutions with different strides. Though the effectiveness of this simple feature pyramid design is first demonstrated in ViTDet [25] for object detection, we show in this work the effectiveness of this simple feature pyramid design for interactive segmentation. We also propose several additional variants (Fig. 6) as part of an ablation study (Sec. 4.4).

All-MLP Segmentation Head We implement a lightweight segmentation head using only MLP layers. It takes in the simple feature pyramid and produces a segmentation probability map¹ of scale 1/4, followed by an upsampling operation to recover the original resolution. Note that this segmentation head avoids computationally demanding components and only accounts for up to 1% of the model parameters (Tab. 1). The key insight is that with a powerful pre-trained backbone, a lightweight segmentation head is sufficient for interactive segmentation. The proposed all-MLP segmentation head works in three steps. *First*, each feature map from the simple feature pyramid goes through an MLP layer to transform it to an identical channel dimension (*i.e.* C_2 in Fig. 2). *Second*, all feature maps are upsampled to the same resolution (*i.e.* 1/4 in Fig. 2) for concatenation. *Third*, the concatenated features are fused by another MLP layer to produce a single-channel feature map, followed by a sigmoid function to obtain a segmentation probability map, which is then transformed to a binary segmentation given a predefined threshold (*i.e.* 0.5).

Symmetric Patch Embedding and Beyond To fuse human clicks into the plain backbone, we introduce a patch embedding layer that is symmetric to the patch embedding layer in the backbone, followed by element-wise feature addition. The user clicks are encoded in a two-channel disk map, one for positive clicks and the other for negative clicks. The positive clicks should be placed on the foreground, while the negative clicks should be placed on the background. The previous segmentation and the two-channel click map are concatenated as a three-channel map for patch embedding. The two symmetric embedding layers operate on the image and the concatenated three-channel map, respectively. The inputs are patchified, flattened, and projected to two vector sequences of the same dimension, followed by element-wise addition before inputting into the self-attention blocks.

3.2. Training and Inference Settings

Backbone Pretraining Our backbone models are pretrained as MAEs [21] on ImageNet-1K [11]. In MAE pretraining, the ViT models reconstruct the randomly masked pixels of images while learning a universal representation. This simple self-supervised approach turns out to be an efficient and scalable way to pretrain ViT models [21]. In this work, we do not perform pretraining ourselves. Instead, we simply use the readily available pretrained MAE weights from [21].

End-to-end Finetuning With the pretrained backbone, we finetune our model end-to-end on the interactive segmentation task. The finetuning pipeline can be briefly described as follows. *First*, we automatically simulate clicks based on the current segmentation and gold standard segmentation, without a human-in-the-loop providing the clicks. Specifically,

¹This probability map may be miscalibrated and can be improved by calibration approaches [12].

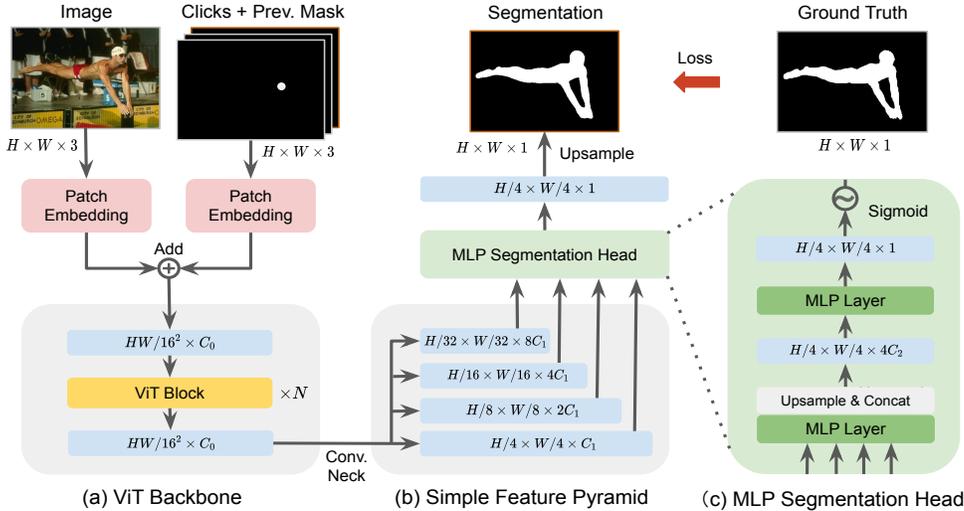


Figure 2. **SimpleClick overview.** Our method consists of three main modules: (a) a plain ViT backbone that maintains single-scale feature maps throughout; (b) a multi-scale simple feature pyramid that is generated from the *last* feature map of the backbone by four parallel convolution or deconvolution layers; (c) a light-weight MLP decoder for segmentation. We also input the previous segmentation to improve the performance and to allow clicking from existing masks. User clicks are encoded as a two-channel disk map, combined with the previous segmentation as input. Boxes in blue are intermediate feature maps. The positional encoding is not shown for brevity.

we use a combination of random and iterative click simulation strategies, inspired by RITM [42]. The random click simulation strategy generates clicks in parallel, without considering the order of the clicks. The iterative click simulation strategy generates clicks iteratively, where the next click should be placed on the erroneous region of a prediction that was obtained using the previous clicks. This strategy is more similar to human clicking behavior. *Second*, we incorporate the segmentation from the previous interaction as an additional input for the backbone, further improving the segmentation quality. This also allows our method to refine from an existing segmentation, which is a desired feature for a practical annotation tool. We use the normalized focal loss [42] (NFL) to train all our models. Previous works [10, 42] show that NFL converges faster and achieves better performance than the widely used binary cross entropy loss for interactive segmentation tasks. Similar training pipelines have been proposed by RITM [42] and its follow-up works [9, 10, 33].

Inference There are two inference modes: automatic evaluation and human evaluation. For automatic evaluation, clicks are automatically simulated based on the current segmentation and gold standard. For human evaluation, a human-in-the-loop provides all clicks based on their subjective evaluation of current segmentation results. We use automatic evaluation for quantitative analyses and human evaluation for a qualitative assessment of the interactive segmentation behavior.

4. Experiments

Datasets We conducted experiments on **10** public datasets including 7 natural image datasets and 3 medical datasets. The details are as follows:

- **GrabCut** [39]: 50 images (50 instances), each with clear foreground and background differences.
- **Berkeley** [36]: 96 images (100 instances); this dataset shares a small portion of images with GrabCut.
- **DAVIS** [38]: 50 videos; we only use the same 345 frames as used in [10, 29, 33, 42] for evaluation.
- **Pascal VOC** [14]: 1449 images (3427 instances) in the validation set. We only test on the validation set.
- **SBD** [20]: 8498 training images (20172 instances) and 2857 validation images (6671 instances). Following previous works [10, 29, 42], we train our model on the training set and evaluate on the validation set.
- **COCO** [28]+**LVIS** [19] (C+L): COCO contains 118K training images (1.2M instances); LVIS shares the same images with COCO but has much higher segmentation quality. We combine the two datasets for training.
- **ssTEM** [15]: two image stacks, each contains 20 medical images. We use the same stack that was used in [33].
- **BraTS** [4]: 369 magnetic resonance image (MRI) volumes; we test on the same 369 slices used in [33].
- **OAIZIB** [2]: 507 MRI volumes; we test on the same 150 slices (300 instances) as used in [32].

Evaluation Metrics Following previous works [29, 41, 42], we automatically simulate user clicks by comparing the current segmentation with the gold standard. In this simulation, the next click will be put at the center of the region

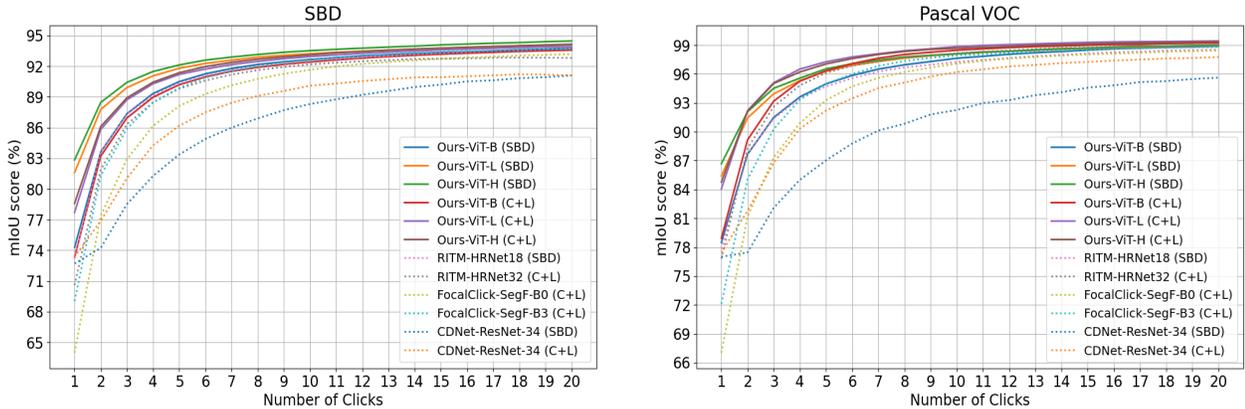


Figure 3. **Convergence analysis** for models trained on either SBD [20] or COCO [28]+LVIS [19] (C+L). We report results on SBD [20] and Pascal VOC [14]. The metric is mean IoU given k clicks ($mIoU@k$). Our models in general require fewer clicks for a given accuracy level.

Method	Backbone	GrabCut		Berkeley		SBD		DAVIS		Pascal VOC	
		NoC85	NoC90	NoC85	NoC90	NoC85	NoC90	NoC85	NoC90	NoC85	NoC90
♪ DIOS [47] _{CVPR16}	FCN	-	6.04	-	8.65	-	-	-	12.58	6.88	-
♪ FCA-Net [30] _{CVPR20}	ResNet-101	-	2.08	-	3.92	-	-	-	7.57	2.69	-
♪ LD [26] _{CVPR18}	VGG-19	3.20	4.79	-	-	7.41	10.78	5.05	9.57	-	-
♪ BRS [23] _{CVPR19}	DenseNet	2.60	3.60	-	5.08	6.59	9.78	5.58	8.24	-	-
♪ f-BRS [41] _{CVPR20}	ResNet-101	2.30	2.72	-	4.57	4.81	7.73	5.04	7.41	-	-
♪ RITM [42] _{Preprint21}	HRNet-18	1.76	2.04	1.87	3.22	3.39	5.43	4.94	6.71	2.51	3.03
♪ CDNet [9] _{ICCV21}	ResNet-34	1.86	2.18	1.95	3.27	5.18	7.89	5.00	6.89	3.61	4.51
♪ PseudoClick [33] _{ECCV22}	HRNet-18	1.68	2.04	1.85	3.23	3.38	5.40	4.81	6.57	2.34	2.74
♪ FocalClick [10] _{CVPR22}	HRNet-18s	1.86	2.06	-	3.14	4.30	6.52	4.92	6.48	-	-
♪ FocalClick [10] _{CVPR22}	SegF-B0	1.66	1.90	-	3.14	4.34	6.51	5.02	7.06	-	-
♪ FocusCut [29] _{CVPR22}	ResNet-50	1.60	1.78	1.85 [†]	3.44	3.62	5.66	5.00	6.38	-	-
♪ FocusCut [29] _{CVPR22}	ResNet-101	1.46	1.64	1.81 [†]	3.01	3.40	5.31	4.85	6.22	-	-
♪ Ours	ViT-B	1.40	1.54	1.44	2.46	3.28	5.24	4.10	5.48	2.38	2.81
♪ Ours	ViT-L	1.38	1.46	1.40	2.33	2.69	4.46	4.12	5.39	1.95	2.30
♪ Ours	ViT-H	1.32	1.44	1.36	2.09	2.51	4.15	4.20	5.34	1.88	2.20
♪ RITM [42] _{Preprint21}	HRNet-32	1.46	1.56	1.43	2.10	3.59	5.71	4.11	5.34	2.19	2.57
♪ CDNet [9] _{ICCV21}	ResNet-34	1.40	1.52	1.47	2.06	4.30	7.04	4.27	5.56	2.74	3.30
♪ PseudoClick [33] _{ECCV22}	HRNet-32	1.36	1.50	1.40	2.08	3.46	5.54	3.79	5.11	1.94	2.25
♪ FocalClick [10] _{CVPR22}	SegF-B0	1.40	1.66	1.59	2.27	4.56	6.86	4.04	5.49	2.97	3.52
♪ FocalClick [10] _{CVPR22}	SegF-B3	1.44	1.50	1.55	1.92	3.53	5.59	3.61	4.90	2.46	2.88
♪ Ours	ViT-B	1.38	1.48	1.36	1.97	3.43	5.62	3.66	5.06	2.06	2.38
♪ Ours	ViT-L	1.32	1.40	1.34	1.89	2.95	4.89	3.26	4.81	1.72	1.96
♪ Ours	ViT-H	1.38	1.50	1.36	1.75	2.85	4.70	3.41	4.78	1.76	1.98

Table 2. **Comparison with previous results.** We report results on five benchmarks: GrabCut [39], Berkeley [36], SBD [20], DAVIS [38], and Pascal VOC [14]. The best results are set in bold. ♪ denotes a model trained on Pascal; ♫ denotes a model trained on SBD; ♫♫ denotes a model trained on COCO [28]+LVIS [19] (C+L); † denotes a number reproduced by the released or retrained models. Our models achieve state-of-the-art performance on all benchmarks.

with the largest error. We use the Number of Clicks (NoC) as the evaluation metric to calculate the number of clicks required to achieve a target Intersection over Union (IoU). We set two target IoUs: 85% and 90%, represented by NoC%85 and NoC%90 respectively. The maximum number of clicks for each instance is set to 20. We also use the average IoU given k clicks ($mIoU@k$) as an evaluation metric to measure the segmentation quality given a fixed number of clicks.

Implementation Details We implement our models using

Python and PyTorch [37]. We implement three models based on three vanilla ViT models (*i.e.* ViT-B, ViT-L, and ViT-H). These backbone models are initialized with the MAE pretrained weights, and then are finetuned end-to-end with other modules. We train our models on either SBD or COCO+LVIS with 55 epochs; the initial learning rate is set to 5×10^{-5} and decreases to 5×10^{-6} after epoch 50. We set the batch size to 140 for ViT-Base, 72 for ViT-Large, and 32 for ViT-Huge to fit the models into GPU memory. All our

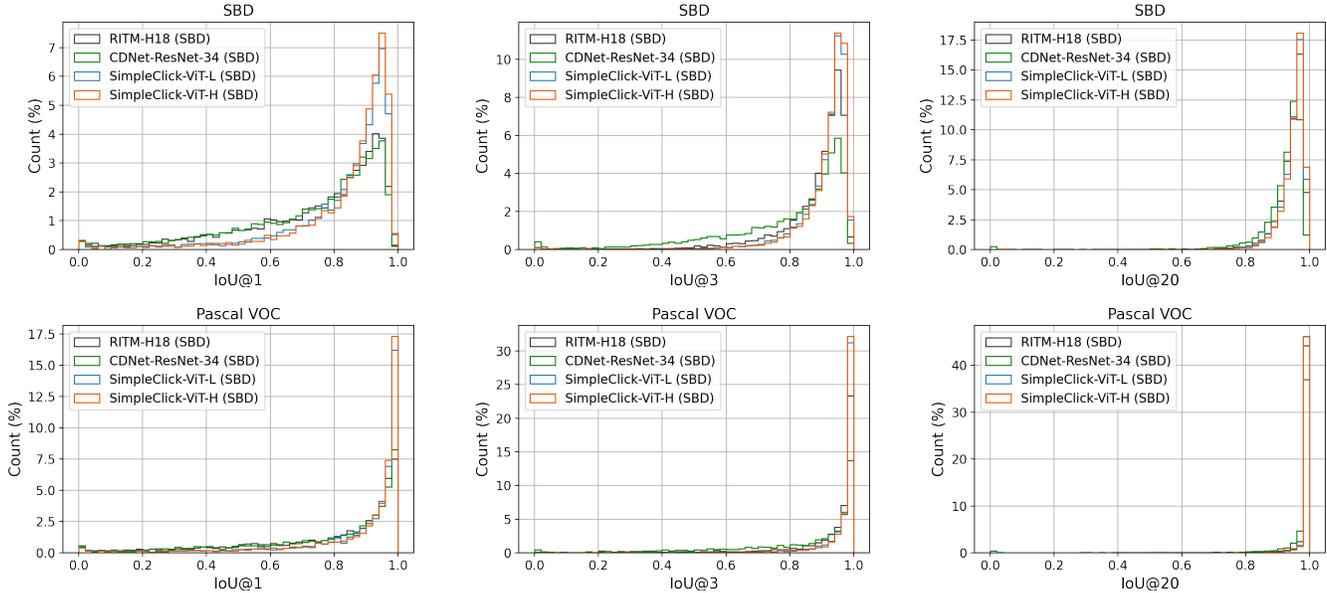


Figure 4. **Histogram analysis** of IoU given a predefined number of clicks k (IoU@ k). We report analysis on SBD [20] and Pascal VOC [14] with models trained on SBD. Compared with the two baselines, our models achieve higher-quality segmentation with fewer failure cases.

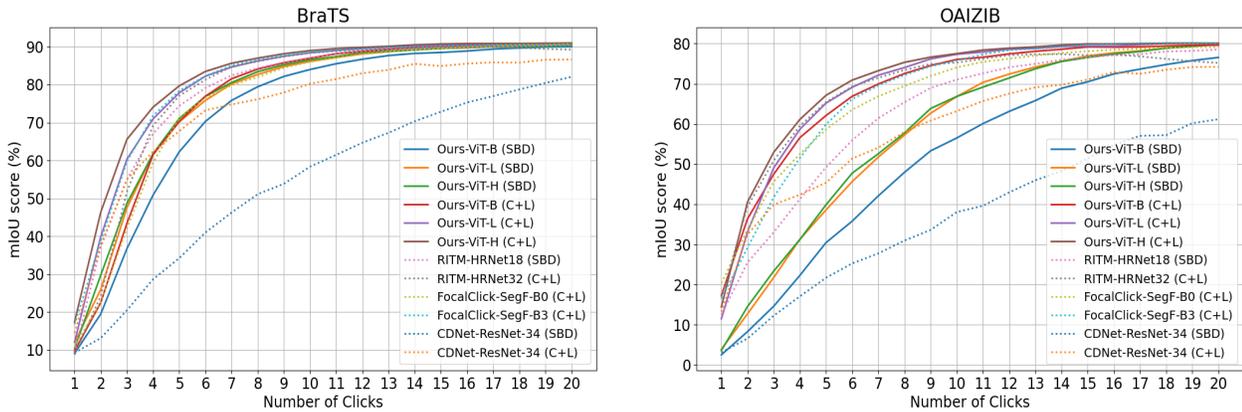


Figure 5. **Convergence analysis** for two medical image datasets: BraTS [4] and OAIZIB [2]. Models are trained on either SBD [20] or COCO [28]+LVIS [19] (denoted as C+L). The metric is mean IoU given k clicks. Overall, our models require fewer clicks for a given accuracy level. The performance gain is more prominent for bigger models (*e.g.* ViT-H) or larger training sets (*e.g.* C+L).

models are trained on four NVIDIA RTX A6000 GPUs. We use the following data augmentation techniques: random resizing (scale range from 0.75 to 1.25), random flipping and rotation, random brightness contrast, and random cropping. Though the ViT backbone was pretrained on images of size 224×224 , we finetune on 448×448 with non-shifting window attention for better performance. We optimize using Adam with $\beta_1 = 0.9$, $\beta_2 = 0.999$.

4.1. Comparison with Previous Results

We show in Tab. 2 the comparisons with previous state-of-the-art results. Our models achieves the best performance on all the five benchmarks. Remarkably, when trained on

SBD training set, our ViT-H model achieves 4.15 NoC@90 on the SBD validation set, outperforming the previous best score by 21.8%. Since the SBD validation set contains the largest number of instances (6671 instances) among the five benchmarks, this improvement is convincing. When trained on COCO+LVIS, our models also achieve state-of-the-art performance on all benchmarks. Fig. 7 shows several segmentation cases on DAVIS, including the worst case. Note that the DAVIS dataset requires high-quality segmentations because all its instances have a high-quality gold standard. Our models still achieve the state-of-the-art on DAVIS without using specific modules, such as a local refinement module [10], which is beneficial for high-quality segmentation.

Model	ssTEM	BraTS	OAIZIB
	mIoU@10	mIoU@10 / 20	mIoU@10 / 20
♪ RITM-H18 [42]	93.15	87.05 / 90.47	71.04 / 78.52
♪ CDN-RN34 [9]	66.72	58.34 / 82.07	38.07 / 61.17
♪ RITM-H32 [42]	94.11	88.34 / 89.25	75.27 / 75.18
♪ CDN-RN34 [9]	88.46	80.24 / 86.63	63.19 / 74.21
♪ FC-SF-B0 [10]	92.62	86.02 / 90.74	74.08 / 79.14
♪ FC-SF-B3 [10]	93.61	88.62 / 90.58	75.77 / 80.08
♪ Ours-ViT-B	93.72	86.98 / 90.67	76.05 / 79.61
♪ Ours-ViT-L	94.34	88.43 / 90.84	77.34 / 79.97
♪ Ours-ViT-H	94.08	88.98 / 91.00	77.50 / 80.10

Table 3. **Out-of-domain evaluation** on three medical image datasets: ssTEM [15], BraTS [4], and OAIZIB [2]. Our models generalize very well on the three datasets, without finetuning.

Model	Backbone	Pretrained	Params/M	NoC85	NoC90
FocalClick	HRNet-18s-S1	✓	4.22	4.74	7.29
FocalClick	SegFormer-B0-S1	✓	3.72	4.98	7.60
SimpleClick	ViT-xTiny	✗	3.72	4.71	7.09

Table 4. **Comparison results on SBD for tiny models.** All models are trained on C+L with 230 epochs. Our SimpleClick-xTiny model outperforms FocalClick models without pretraining.

Fig. 9 shows that our method converges better than other methods with sufficient clicks, leading to fewer failure cases as shown in Fig. 4. We only report results on SBD and Pascal VOC, the top two largest datasets.

4.2. Out-of-Domain Evaluation on Medical Images

We further evaluate the generalizability of our models on three medical image datasets: ssTEM [15], BraTS [3], and OAIZIB [2]. Tab. 3 reports the evaluation results on these three datasets. Fig. 5 shows the convergence analysis on BraTS and OAIZIB. Overall, our models generalize well to medical images. We also find that the models trained on larger datasets (*i.e.* C+L) generalize better than the models trained on smaller datasets (*i.e.* SBD).

4.3. Towards Practical Annotation Tool

Tiny Backbone To allow for practical applications, especially on low-end devices with limited computational resources, we implement an extremely tiny backbone (*i.e.* ViT-xTiny) for SimpleClick. Compared with ViT-Base, ViT-xTiny decreases the embedding dimension from 768 to 160 and the number of attention blocks from 12 to 8. We end up with a SimpleClick-xTiny model, which is comparable with the tiny FocalClick models in terms of parameters. Comparison results in Tab. 4 show that our model outperforms FocalClick models, even though it is trained from scratch due to the lack of readily available pretrained weights.

Computational Analysis Tab. 5 shows a comparison of computational requirements with respect to model parameters, FLOPs, GPU memory consumption, and speed; the speed is measured by seconds per click (SPC). Fig. 1 shows the interactive segmentation performance of methods in

Backbone	Params/M	FLOPs/G	Mem/G	↓SPC/ms
HR-18s ₄₀₀ [42]	4.22	17.94	0.50	54
HR-18 ₄₀₀ [42]	10.03	30.99	0.52	56
HR-32 ₄₀₀ [42]	30.95	83.12	1.12	86
Swin-B ₄₀₀ [32]	87.44	138.21	1.41	36
Swin-L ₄₀₀ [32]	195.90	302.78	2.14	44
SegF-B0 ₂₅₆ [10]	3.72	3.42	0.10	37
SegF-B3 ₂₅₆ [10]	45.66	24.75	0.32	53
ResN-34 ₃₈₄ [9]	23.47	113.60	0.25	34
ResN-50 ₃₈₄ [29]	40.36	78.82	0.85	331
ResN-101 ₃₈₄ [29]	59.35	100.76	0.89	355
Ours-ViT-xT ₂₂₄	3.72	2.63	0.17	17
Ours-ViT-xT ₄₄₈	3.72	10.52	0.23	29
Ours-ViT-B ₂₂₄	96.46	42.44	0.51	34
Ours-ViT-B ₄₄₈	96.46	169.78	0.87	54
Ours-ViT-L ₄₄₈	322.18	532.87	1.72	86
Ours-ViT-H ₄₄₈	659.39	1401.93	3.22	132

Table 5. **Computation comparison** for model parameters, FLOPs, GPU memory consumption (measured by the maximum GPU memory managed by PyTorch’s caching allocator), and speed (measured by seconds per click). Each method is denoted by its backbone. The small number in front of the model denotes the input size (448×448 for our models by default). Even for our ViT-H model, the speed (132ms) and memory consumption (3.22G) are sufficient to meet the requirements of a practical annotation tool.

FP design	frozen ViT	ViT-B		ViT-L	
		SBD	Pascal	SBD	Pascal
(a) simple FP	✓	11.48	6.93	9.75	5.59
(a) simple FP	✗	5.24	2.53	4.46	2.15
(b) single-scale	✗	6.56	2.80	5.53	2.48
(c) parallel	✗	7.21	3.09	6.26	2.79
(d) partial	✗	8.29	4.34	7.51	4.25

Table 6. **Ablation study** on backbone finetuning and feature pyramid (FP) design (Fig. 6). The metric is NoC@90. We have three findings in this ablation: 1) freezing the ViT backbone during finetuning significantly deteriorates the performance; 2) the multi-scale property matters for the simple feature pyramid; 3) the last feature map from the backbone is sufficient to build an effective feature pyramid.

terms of FLOPs. In Fig. 1 and Tab. 5, each method is denoted by its backbone. For fair comparison, we evaluate all the methods on the same benchmark (*i.e.* GrabCut) and using the same computer (GPU: NVIDIA RTX A6000, CPU: Intel Silver×2). We only calculate the FLOPs in a single forward pass. For methods like FocusCut which require multiple forward passes for each click, the FLOPs may be much higher than reported. By default, our method takes images of size 448×448 as the fixed input. Even for our ViT-H model, the speed (132ms) and memory consumption (3.22G) is sufficient to meet the requirements of a practical annotation tool.

4.4. Ablation Study

In this section, we ablate the backbone finetuning and feature pyramid design. Tab. 6 shows the ablation results.

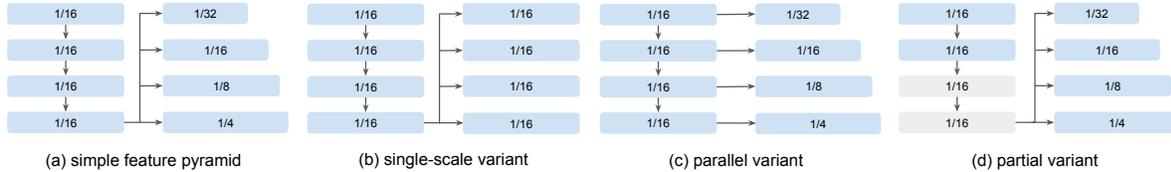


Figure 6. **Simple feature pyramid and its variants**: b) the single-scale variant ablates the multi-scale property; c) the parallel variant evenly extracts features from the backbone; d) the partial variant freezes the second half parameters of the backbone. Tab. 6 shows the comparison results.

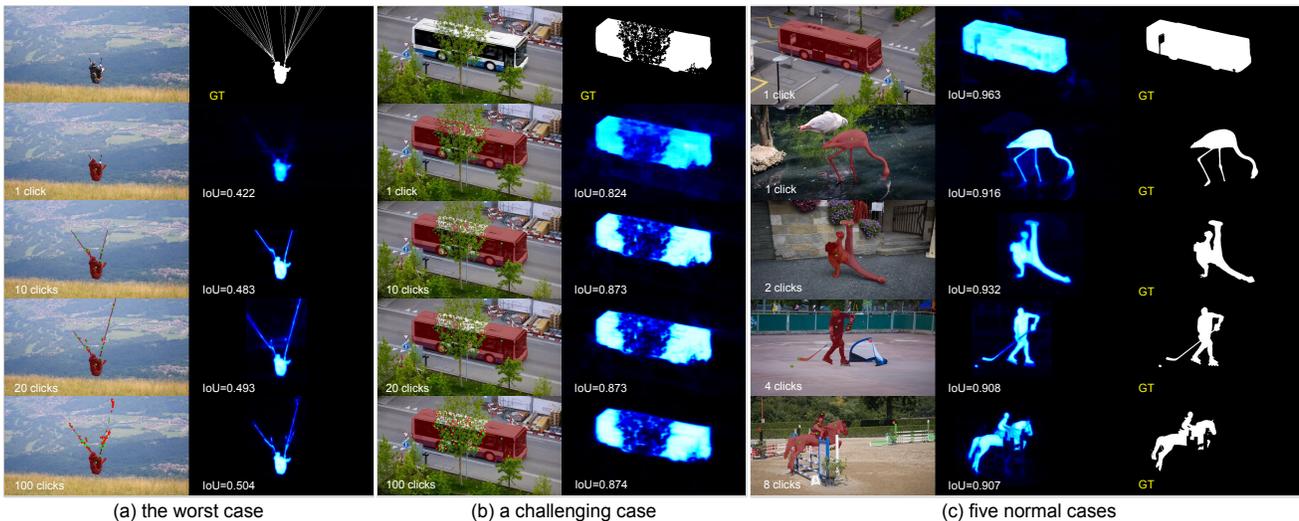


Figure 7. **Segmentation results** on DAVIS [38]: (a) the worst case; (b) a challenging case; (c) five normal cases. The backbone model is ViT-L trained on COCO [28]+LVIS [19]. The segmentation probability maps are shown in blue; the segmentation maps are overlaid in red on the original images. The clicks are shown as green (positive click) or red (negative click) dots on the image. GT denotes ground truth.

By default, we finetune the backbone along with other modules. As an ablation, we freeze the backbone during finetuning, leading to significantly worse performance. This ablation is explainable considering the ViT backbone takes most of the model parameters (Tab. 1). For the second ablation, we compare the default simple feature pyramid design with three variants depicted in Fig. 6 (*i.e.* (b), (c), and (d)). First, we observe that the multi-scale representation matters for the feature pyramid. By ablating the multi-scale property in the simple feature pyramid, the performance drops considerably. We also notice that the *last* feature map from the backbone is strong enough to build the feature pyramid. The parallel feature pyramid generated by multi-stage feature maps from the backbone does not surpass the simple feature pyramid that *only* uses the last feature map of the backbone.

5. Limitations and Remarks

Our best-performing model (ViT-H) is much larger than existing models, leading to concerns about an unfair comparison. We justify the effectiveness of SimpleClick by developing a tiny model and comparing it fairly with other methods. Other than this, our models may fail in some chal-

lenging scenarios such as objects with very thin and elongated shapes or cluttered occlusions((a) and (b) in Fig. 7). We leave the improvements for future work.

We are entering an era of large-scale pretraining on multi-modal foundation models, which is dramatically transforming the landscape of vision and language tasks. In this context, we hope SimpleClick will serve as a strong baseline for a new wave of high-performing interactive segmentation methods based on ViTs and large-scale pretraining.

6. Conclusions

We proposed SimpleClick, the first plain-backbone method for interactive image segmentation. Our method leveraged a general-purpose ViT backbone that can benefit from readily available pretrained ViT models. With the MAE-pretrained weights, SimpleClick achieved state-of-the-art performance on natural images and demonstrated strong generalizability on medical images. We also developed a tiny SimpleClick model and provided a detailed computational analysis, highlighting the suitability of SimpleClick as a practical annotation tool.

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. 1, 2
- [2] Felix Ambellan, Alexander Tack, Moritz Ehlke, and Stefan Zachow. Automated segmentation of knee bone and cartilage combining statistical shape knowledge and convolutional neural networks: Data from the osteoarthritis initiative. *Medical image analysis*, 52:109–118, 2019. 4, 6, 7, 12, 13
- [3] Xue Bai and Guillermo Sapiro. A geodesic framework for fast interactive image and video segmentation and matting. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007. 7, 13
- [4] Ujjwal Baid, Satyam Ghodasara, Suyash Mohan, Michel Bilello, Evan Calabrese, Errol Colak, Keyvan Farahani, Jayashree Kalpathy-Cramer, Felipe C Kitamura, Sarthak Pati, et al. The rsna-asnr-miccai brats 2021 benchmark on brain tumor segmentation and radiogenomic classification. *arXiv preprint arXiv:2107.02314*, 2021. 4, 6, 7, 12
- [5] Gedas Bertasius and Lorenzo Torresani. Classifying, segmenting, and tracking object instances in video with mask propagation. In *CVPR*, pages 9739–9748, 2020. 1
- [6] 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. 2
- [7] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, pages 11621–11631, 2020. 1
- [8] Wuyang Chen, Xianzhi Du, Fan Yang, Lucas Beyer, Xiaohua Zhai, Tsung-Yi Lin, Huizhong Chen, Jing Li, Xiaodan Song, Zhangyang Wang, et al. A simple single-scale vision transformer for object localization and instance segmentation. *arXiv preprint arXiv:2112.09747*, 2021. 2
- [9] Xi Chen, Zhiyan Zhao, Feiwu Yu, Yilei Zhang, and Manni Duan. Conditional diffusion for interactive segmentation. In *ICCV*, pages 7345–7354, 2021. 4, 5, 7
- [10] Xi Chen, Zhiyan Zhao, Yilei Zhang, Manni Duan, Donglian Qi, and Hengshuang Zhao. FocalClick: Towards practical interactive image segmentation. In *CVPR*, pages 1300–1309, 2022. 1, 2, 4, 5, 6, 7, 11
- [11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 3
- [12] Zhipeng Ding, Xu Han, Peirong Liu, and Marc Niethammer. Local temperature scaling for probability calibration. In *ICCV*, pages 6889–6899, 2021. 3
- [13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2, 3
- [14] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 4, 5, 6, 11
- [15] Stephan Gerhard, Jan Funke, Julien Martel, Albert Cardona, and Richard Fetter. Segmented anisotropic sstem dataset of neural tissue. *figshare*, pages 0–0, 2013. 4, 7, 12, 13
- [16] Leo Grady. Random walks for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 28(11):1768–1783, 2006. 2
- [17] Jiaqi Gu, Hyoukjun Kwon, Dilin Wang, Wei Ye, Meng Li, Yu-Hsin Chen, Liangzhen Lai, Vikas Chandra, and David Z Pan. Multi-scale high-resolution vision transformer for semantic segmentation. In *CVPR*, pages 12094–12103, 2022. 2
- [18] 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. 2
- [19] Agrim Gupta, Piotr Dollár, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *CVPR*, pages 5356–5364, 2019. 4, 5, 6, 8, 11, 13
- [20] 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, 4, 5, 6, 11, 13
- [21] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021. 2, 3, 11
- [22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 1
- [23] Won-Dong Jang and Chang-Su Kim. Interactive image segmentation via backpropagating refinement scheme. In *CVPR*, pages 5297–5306, 2019. 5
- [24] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *ACM computing surveys (CSUR)*, 54(10s):1–41, 2022. 2
- [25] Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. Exploring plain vision transformer backbones for object detection. *arXiv preprint arXiv:2203.16527*, 2022. 2, 3, 11
- [26] Zhuwen Li, Qifeng Chen, and Vladlen Koltun. Interactive image segmentation with latent diversity. In *CVPR*, pages 577–585, 2018. 5
- [27] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 2117–2125, 2017. 2, 3
- [28] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014. 4, 5, 6, 8, 11, 13
- [29] Zheng Lin, Zheng-Peng Duan, Zhao Zhang, Chun-Le Guo, and Ming-Ming Cheng. FocusCut: Diving into a focus view

- in interactive segmentation. In *CVPR*, pages 2637–2646, 2022. 1, 2, 4, 5, 7
- [30] Zheng Lin, Zhao Zhang, Lin-Zhuo Chen, Ming-Ming Cheng, and Shao-Ping Lu. Interactive image segmentation with first click attention. In *CVPR*, pages 13339–13348, 2020. 2, 5
- [31] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017. 1
- [32] Qin Liu, Zhenlin Xu, Yining Jiao, and Marc Niethammer. iSegFormer: Interactive segmentation via transformers with application to 3d knee mr images. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 464–474. Springer, 2022. 1, 2, 4, 7
- [33] Qin Liu, Meng Zheng, Benjamin Planche, Srikrishna Karanam, Terrence Chen, Marc Niethammer, and Ziyang Wu. PseudoClick: Interactive image segmentation with click imitation. *arXiv preprint arXiv:2207.05282*, 2022. 1, 2, 4, 5, 11
- [34] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pages 10012–10022, 2021. 1, 2
- [35] Kevis-Kokitsi Maninis, Sergi Caelles, Jordi Pont-Tuset, and Luc Van Gool. Deep extreme cut: From extreme points to object segmentation. In *CVPR*, pages 616–625, 2018. 2
- [36] 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. 4, 5, 11
- [37] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 5
- [38] 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 *CVPR*, pages 724–732, 2016. 4, 5, 8, 11
- [39] 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. 2, 4, 5, 11
- [40] Dinggang Shen, Guorong Wu, and Heung-Il Suk. Deep learning in medical image analysis. *Annual review of biomedical engineering*, 19:221, 2017. 1
- [41] Konstantin Sofiiuk, Ilia Petrov, Olga Barinova, and Anton Konushin. f-brs: Rethinking backpropagating refinement for interactive segmentation. In *CVPR*, pages 8623–8632, 2020. 4, 5
- [42] Konstantin Sofiiuk, Ilia A Petrov, and Anton Konushin. Reviving iterative training with mask guidance for interactive segmentation. *arXiv preprint arXiv:2102.06583*, 2021. 1, 2, 4, 5, 7, 11
- [43] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *ICCV*, pages 7262–7272, 2021. 2
- [44] Jiajun Wu, Yibiao Zhao, Jun-Yan Zhu, Siwei Luo, and Zhuowen Tu. Milcut: A sweeping line multiple instance learning paradigm for interactive image segmentation. In *CVPR*, pages 256–263, 2014. 1, 2
- [45] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. SegFormer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*, 34:12077–12090, 2021. 2
- [46] Ning Xu, Brian Price, Scott Cohen, Jimei Yang, and Thomas Huang. Deep grabcut for object selection. *arXiv preprint arXiv:1707.00243*, 2017. 1, 2
- [47] Ning Xu, Brian Price, Scott Cohen, Jimei Yang, and Thomas S Huang. Deep interactive object selection. In *CVPR*, pages 373–381, 2016. 2, 5
- [48] Ning Xu, Linjie Yang, Yuchen Fan, Dingcheng Yue, Yuchen Liang, Jianchao Yang, and Thomas Huang. Youtube-vos: A large-scale video object segmentation benchmark. *arXiv preprint arXiv:1809.03327*, 2018. 1
- [49] Yuhui Yuan, Rao Fu, Lang Huang, Weihong Lin, Chao Zhang, Xilin Chen, and Jingdong Wang. Hrformer: High-resolution vision transformer for dense predict. *Advances in Neural Information Processing Systems*, 34:7281–7293, 2021. 2
- [50] Shiyin Zhang, Jun Hao Liew, Yunchao Wei, Shikui Wei, and Yao Zhao. Interactive object segmentation with inside-outside guidance. In *CVPR*, pages 12234–12244, 2020. 1
- [51] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*, pages 6881–6890, 2021. 2

A. Datasets

This section supplements the “Datasets” (Sec. 4) in the main paper. Our models are trained either using SBD [20] or the combined COCO [28]+LVIS [19] datasets. Before RITM [42], most of the deep learning-based interactive segmentation models were trained either using the SBD [20] or Pascal VOC [14] datasets. These two datasets only cover 20 categories of general objects such as persons, transportation vehicles, animals, and indoor objects. The authors of RITM constructed the combined COCO+LVIS dataset, which contains 118k training images of 80 diverse object classes, for interactive segmentation. This large and diverse training dataset contributes to the state-of-the-art performance of RITM models. Inspired by RITM and its follow-up works [10, 33], we use SBD and COCO+LVIS as our training datasets.

B. Implementation Details

B.1. Architectures

This section supplements Sec. 3.1 “Network Architecture” in the main paper. Tab. 7 shows the main architecture parameters of our models. By default, our models use an input size of 448×448 during training and evaluation. Our ViT-B and ViT-L models use a patch size of 16×16 , while the ViT-H model uses a smaller patch size of 14×14 . This leads to a higher resolution representation in terms of the number of patches. Each patch is flattened and projected to an embed dimension of C_0 through the patch embedding layer. The tokens generated by the patch embedding layer are processed by N self-attention blocks, which N is a hyper-parameter inherited from plain ViT models [21]. Inspired by ViTDet [25], we build a simple feature pyramid with the four resolutions $\{\frac{1}{32}, \frac{1}{16}, \frac{1}{8}, \frac{1}{4}\}$. The $\frac{1}{16}$ resolution uses the last feature map of the ViT backbone. The $\frac{1}{32}$ resolution is built by a 2×2 convolutional layer with a stride of 2. The $\frac{1}{8}$ (or $\frac{1}{4}$) resolution is built by one (or two) 2×2 transposed convolution layer(s) with a stride of 2. We use a 1×1 convolution layer with layer normalization to convert the channels of each feature map to predefined dimensions. Specifically, feature maps of resolutions $\{\frac{1}{32}, \frac{1}{16}, \frac{1}{8}, \frac{1}{4}\}$ are converted to channel dimensions of $\{8C_1, 4C_1, 2C_1, C_1\}$, respectively. Each feature map is then converted to the same dimension of C_2 through an MLP layer in the segmentation head, followed by upsampling to the $\frac{1}{4}$ resolution. At this point, the four feature maps have the same resolution and the same number of channels. They are concatenated as a single feature map with $4C_2$ channels. Another MLP layer in the segmentation head converts this multi-channel feature map to a one-channel feature map, followed by a sigmoid function to obtain the final binary segmentation. We use C_1 and C_2 as hyper-parameters without tuning.

Model	H, W	Patch Size	N	C_0, C_1, C_2
Ours-ViT-B	448, 448	16×16	12	768, 128, 256
Ours-ViT-L	448, 448	16×16	24	1024, 192, 256
Ours-ViT-H	448, 448	14×14	32	1280, 240, 256

Table 7. **Architecture parameters** of SimpleClick models. N denotes the number of self-attention blocks. $C_0, C_1,$ and C_2 denote the feature map dimensions at different levels.

B.2. Clicks Encoding

This section also supplements Sec. 3.1 in the main paper. We encode clicks, which are represented by the coordinates in an image, as disks with a small radius of 5 pixels. Positive and negative clicks are encoded separately. In our implementation, we also attach the previous segmentation as an additional channel, resulting in a three-channel disk map. Two patch embedding layers, which are of the same structure, process the three-channel disk map and the RGB image separately. The tokens of the two inputs after the patch embedding layers are added element by element, without changing the input dimensions for the self-attention blocks. This design is more efficient than other designs such as concatenation and allows our ViT backbones to be initialized with pretrained ViT weights.

B.3. Finetuning on Higher-Resolution Images

This section supplements Sec. 3.2 “Training and Inference Settings” in the main paper. Our models are pretrained on an image size of 224×224 but are finetuned on an image size of 448×448 . We first interpolate the positional encoding to the high resolution. Then, we perform non-overlapping window attention [25] with a few global blocks for cross-window attention. The high-resolution feature map is divided into regular non-overlapping windows. The non-global blocks perform self-attention within each window, while global blocks perform global self-attention. We set the number of global blocks to 2, 6, and 8 for the ViT-B, ViT-L, and ViT-H models, respectively.

C. Additional Comparison Results

This section supplements Sec. 4.1 “Comparison with Previous Results” in the main paper. Fig. 9 shows convergence results for our models on four datasets: GrabCut [39], Berkeley [36], DAVIS [38], and COCO [28]. Overall, our models perform better than other models on these datasets. However, the results in Fig. 9 are not as compelling as the results on SBD [20] or Pascal VOC [14] (shown in Fig. 3 of the main paper). This is likely due to the limited number of images in these datasets (*e.g.* GrabCut only contains 50 instances, while SBD contains 6671 instances for evaluation).

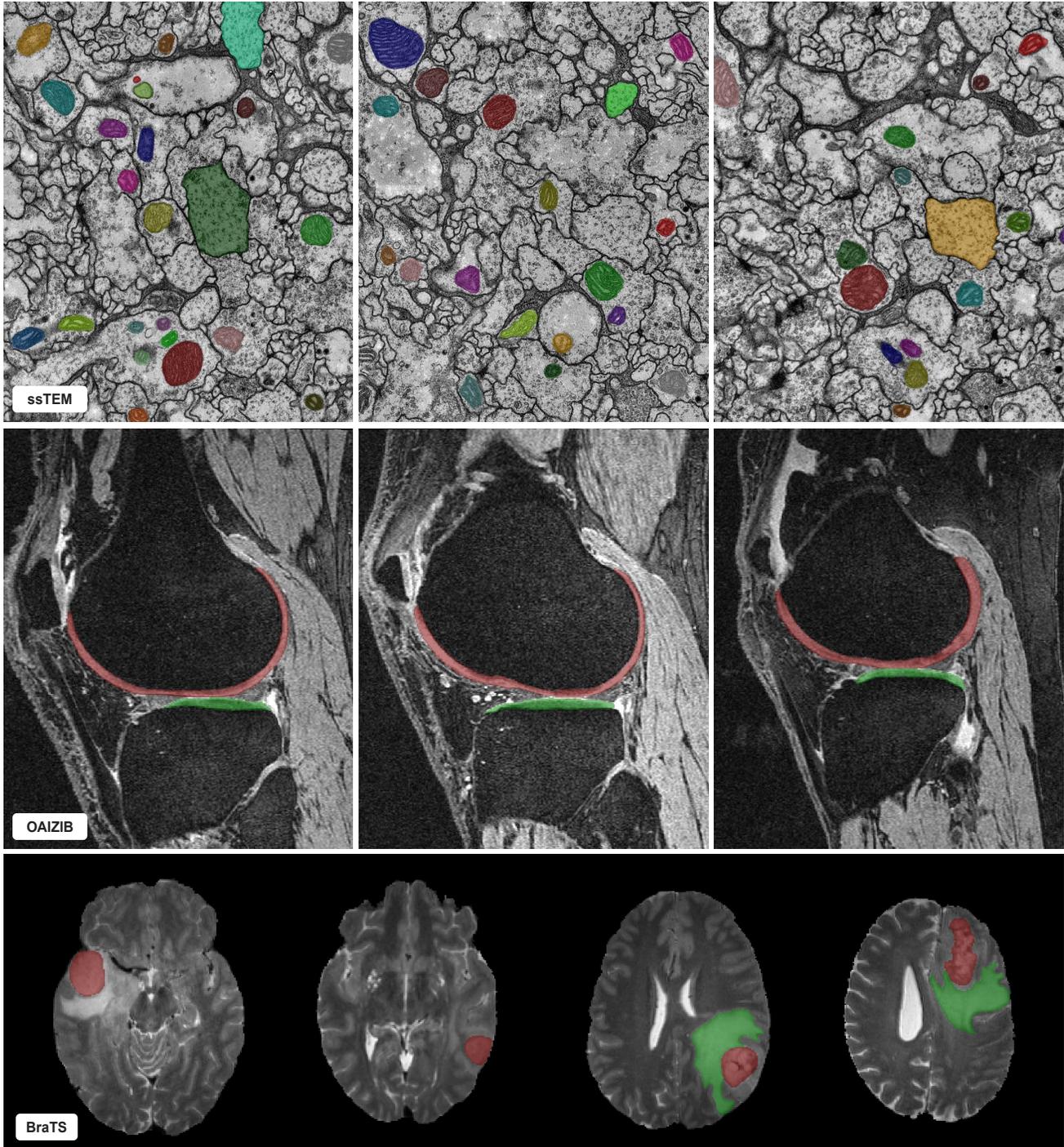


Figure 8. **Qualitative results** of human evaluation on three medical image datasets: ssTEM [15], OAIZIB [2], and BraTS [4]. All the results are obtained by a human-in-the-loop providing the clicks. Though our models are evaluated on medical images without finetuning, they generalize well to all the unseen objects given a few clicks, as shown in the demo videos (<https://github.com/uncbiag/SimpleClick>).

D. Human Evaluation on Medical Images

This section supplements Sec. 4.2 “Out-of-Domain Evaluation on Medical Images” in the main paper. In the main

paper, we report quantitative results on medical images using an automatic evaluation mode where clicks are automatically simulated. In this section, we perform human eval-

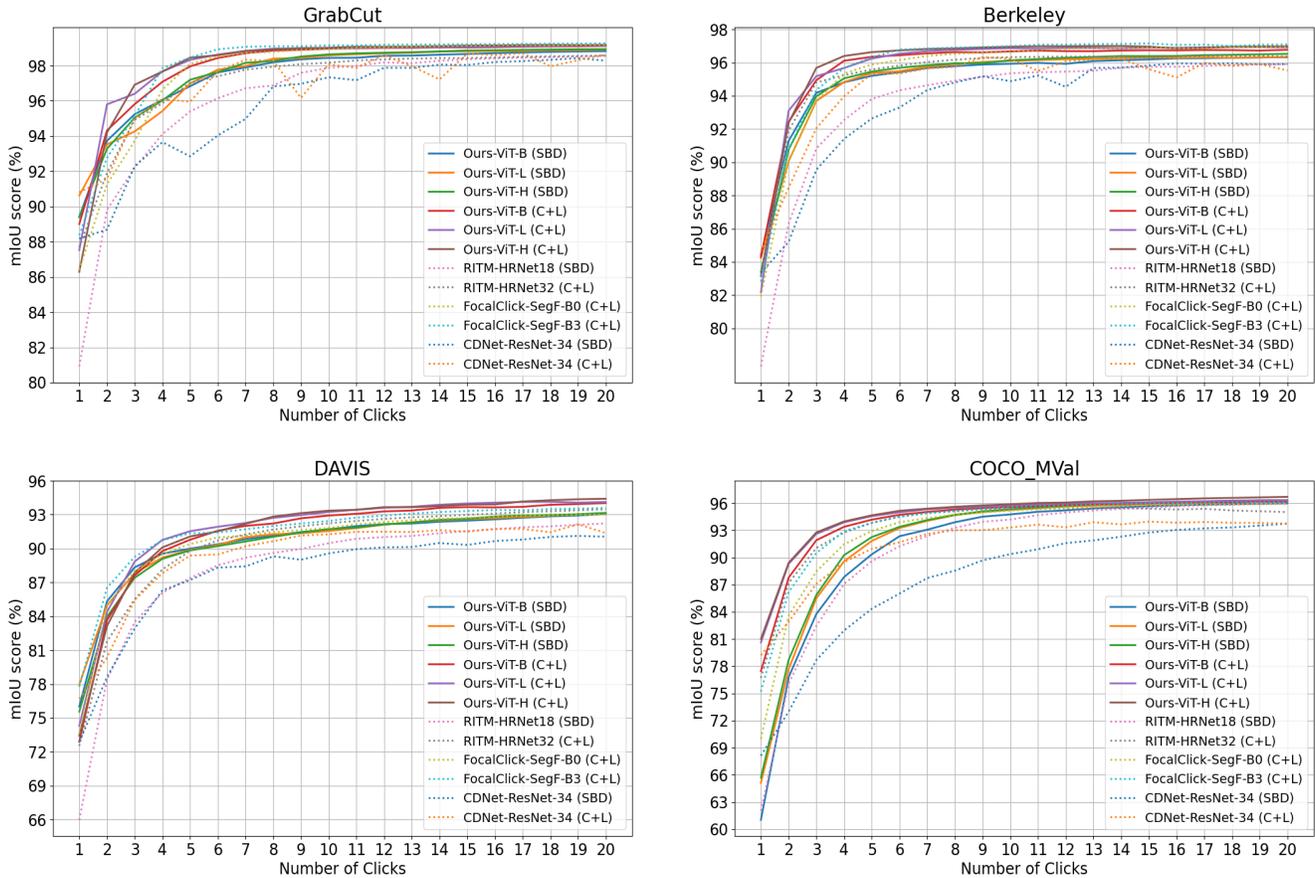


Figure 9. **Convergence analysis** for GrabCut, Berkeley, DAVIS, and COCO. All models are trained on either SBD [20] or COCO [28]+LVIS [19] (C+L). The metric is mean IoU given k clicks ($mIoU@k$). In general, our models require fewer clicks for a given accuracy level.

uations where a human-in-the-loop provides all the clicks. Fig. 8 shows qualitative results on three medical image datasets: ssTEM [15], OAIZIB [2], and BraTS [3]. For simple objects such as cell nuclei in ssTEM, it may take as little as one click for a good segmentation. However, for more challenging objects such as knee cartilage in the OAIZIB dataset or brain tumors in the BraTS dataset, it may take more than ten clicks until a high-quality segmentation is obtained. Considering our models are not finetuned on the label-scarce medical imaging datasets, our observed performance is quite promising. The attached videos demonstrate the evaluation process.