Slim Scissors: Segmenting Thin Object from Synthetic Background (Supplementary Material)

Kunyang Han[1,2*], Jun Hao Liew[3], Jiashi Feng[3], Huawei Tian[4], Yao Zhao[1,2], and Yunchao Wei[1,2]

[1] Institute of Information Science, Beijing Jiaotong University
[2] Beijing Key Laboratory of Advanced Information Science and Network Technology
[3] ByteDance
[4] People's Public Security University of China

1 Network Architecture

As described in the main paper, our Slim Scissors is composed of two parallel networks, *i.e.*, a Generic Object Segmentation (GOS) Network (Section 1.1) for acquiring a coarse mask delineating the object's main body and a Thin Parts Refinement (TPR) Network (Section 1.2) for capturing the elongated thin structures based on the user-given scribbles. On top of Slim Scissors, we append a Similarity Detection Module (SDM) (Section 1.4) that enables automated scribbles mining to further reduce user's annotation burden, which we call Fast Slim Scissors. We present the details of each network/module below. The overall architecture can be found in Fig. 2 and 1.

1.1 Generic Object Segmentation (GOS) Network

Our GOS network employs the state-of-the-art interactive segmentation network IOG [6] due to its excellent performance in segmenting general objects. In particular, it consists of a CoarseNet for coarse prediction given the inside-outside guidance, followed by a FineNet that recovers the missing boundary details. In this work, we replace the ResNet-101 backbone in IOG with a lighter ResNet-18 [2] to reduce computational cost. For more details, we refer the interested readers to [6].

1.2 Thin Part Refinement (TPR) Network

First Stage. Similarly, our TPR network also adopts a coarse-to-fine design. Specifically, the first stage employs an encoder-decoder structure, where the encoder takes ResNet-18 [2] as its backbone, with its global average pooling and `FC` layers removed. Following [1], we limit the output stride to 16 by setting the stride and dilation rate of the last residual block (`Layer_4`) to 1 and 2, respectively. To incorporate global context, we append an Atrous Spatial Pyramid Pooling (ASPP) module [1] at the end of the encoder. The decoder, on the

* Work done during an internship at ByteDance.

other hand, progressively upsamples the features and concatenates with earlier features, followed by a 3×3 `conv` layer with `ReLU` activation and Batch Normalization [3], producing a coarse mask and features of 32 channels. A sigmoid activation is appended to constrain the mask prediction to be be in the range of $[0, 1]$

Second Stage (Refiner). The second stage of TPR acts as a refiner for obtaining a sharper prediction along the elongated thin parts. In particular, it takes the concatenation of input image, synthesized background and features from the first stage as input (38 channels in total). To increase the receptive field for refinement, we first downsample the concatenated input by $0.5\times$, followed by two 3×3 `conv` layers. The output features is then upsampled and concatenated with the input image and synthetic background before passed to two other 3×3 `conv` layers for producing the final mask of thin parts. Similarly, we also append a sigmoid activation to the mask prediction.

1.3 Fusion

Lastly, we fuse the features from both the GOS and TPR networks to obtain a final mask that contains both the object's main body and elongated thin parts. Specifically, we first reduce the dimension of the features from FineNet in GOS network using a 1×1 `conv` layer. At the same time, the features from Refiner in TPR network undergoes a 3×3 `conv` layer. Both features are then fused via element-wise summation, followed by a final 1×1 `conv` layer to produce the final segmentation mask.

1.4 Similarity Detection Module (SDM)

As shown in Fig. 1, our SDM adopts ResNet-18 [2]-based encoder-decoder structure as its feature extractor. Similar to TPR, we append ASPP at the end of encoder. The encoder features is then upsampled and concatenated with features from `Layer_1`, followed by three 3×3 `conv` layers (interleaved with `ReLU` and Batch Normalization) and a 1×1 `conv` to produce an embedding feature for matching.

To perform matching, we pass the user-provided *coarse* scribbles to our TPR network to first obtain a refined mask of thin parts. In this way, the scribbled region can be separated into a set of foreground and background pixels for matching. The foreground and background matching maps, together with the embedding features, are then passed to three 7×7 `conv`, followed by a 3×3 `conv` and a bilinear upsampling operation to produce a binary mask predicting other thin parts similar to the one(s) highlighted by current scribble(s).

2 Automated Scribble Recommendation

In Fig. 3, we also show some qualitative examples of how our SDM generates additional scribbles to reduce user's annotation burden. In cases when there are

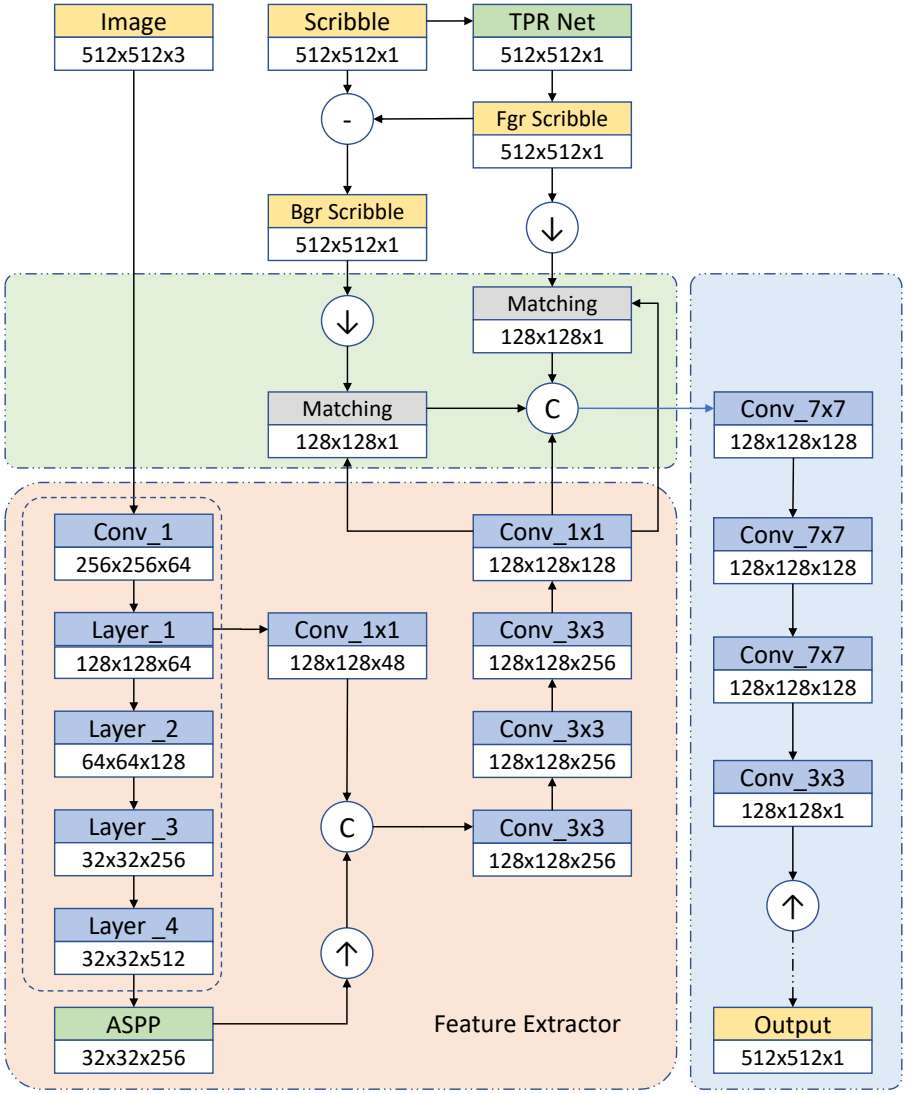


Fig. 1. Similarity Detection Module (SDM). ↑ and ↓ denote bilinear upsamppling and downsampling, respectively. ⊖ implies element-wise reduction while Ⓒ refers to concatenation.

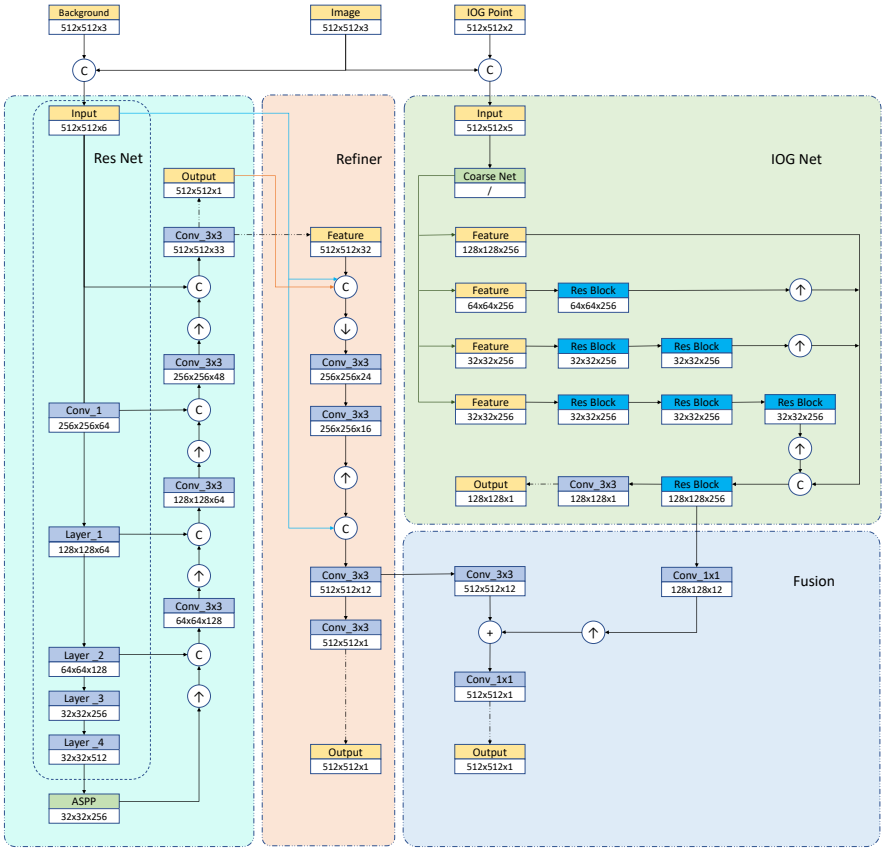
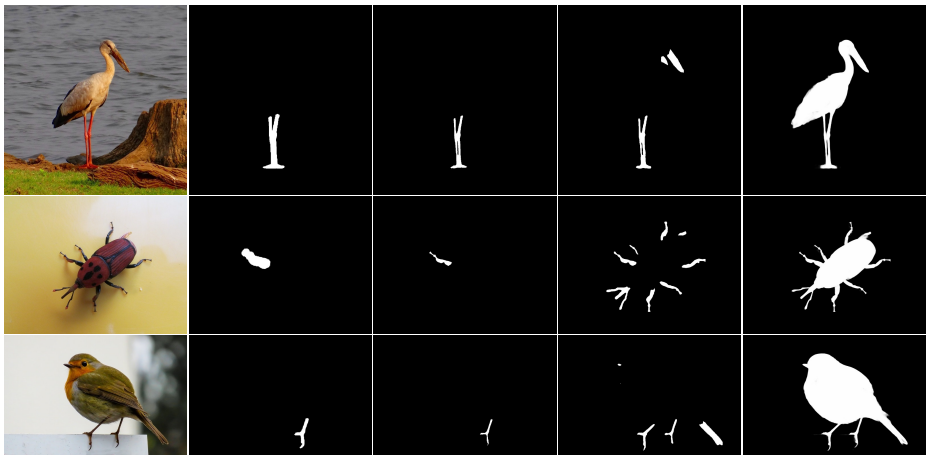


Fig. 2. Slim Scissors. Ⓒ denotes concatenation operation while ↑ refers to bilinear upsampling.

either missing or incorrectly added scribble(s), user can interrupt the interactive segmentation system by introducing new scribbles (Fig. 4 bottom) or removing the wrongly labeled scribbles (Fig, 4 top) until the user is satisfied with the final segmentation quality.



(a) Input image (b) Coarse scribble (c) Refined scribble (d) Output of SDM (e) Final mask

Fig. 3. Examples of automated scribble recommendation by our Similarity Detection Module (SDM). (b) The user first provides some coarse scribbles around the elongated thin parts, which are then passed to our TPR network to extract the thin parts within those scribbles (c), separating the scribbled areas to foreground and background sets for matching. (d) The matching outputs are then passed to our SDM to generate additional scribble(s). (e) Finally, all the scribbles are used to produce the final segmentation output.

3 Challenging Cases

In Fig. 5, we also studied the performance of our Slim Scissors on challenging cases, such as racket strings, soccer goal nets, cage, *etc*. In the left figure, we use extremely coarse scribbles to quickly cover all the thin regions whereas in the right figure, we randomly scribble on one of the thin parts and rely on our SDM module to automatically generate the scribbles for the remaining thin parts. As shown in Fig. 5, our Slim Scissors yield satisfactory performance even on these challenging cases.

4 Additional Qualitative Results

In addition to the qualitative examples shown in our main paper, we also provide more qualitative results here. Fig. 6 and 7 show the result of our Slim Scissors on COIFT [4] and HRSOD [5] dataset, respectively.

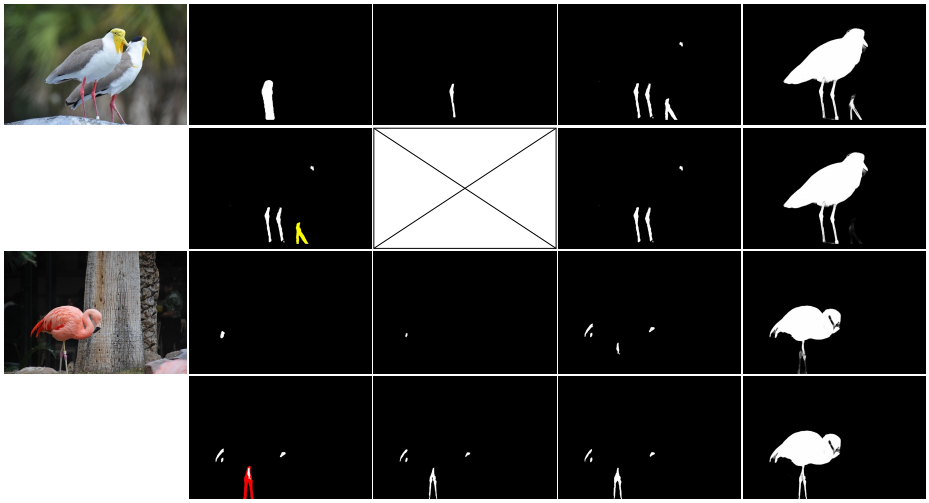


Fig. 4. More examples of automated scribbles recommendation. (**top**) When our SDS incorrectly synthesizes additional scribbles (*e.g.*, another bird's legs), user can simply "erase" the wrongly generated scribbles to remove them (highlighted in yellow). Note that we need not run SDM when removing a scribble. (**bottom**) When there are missing thin parts, user can simply introduce additional scribbles (highlighted in red) to guide both the scribbles generation and subsequent segmentation process.

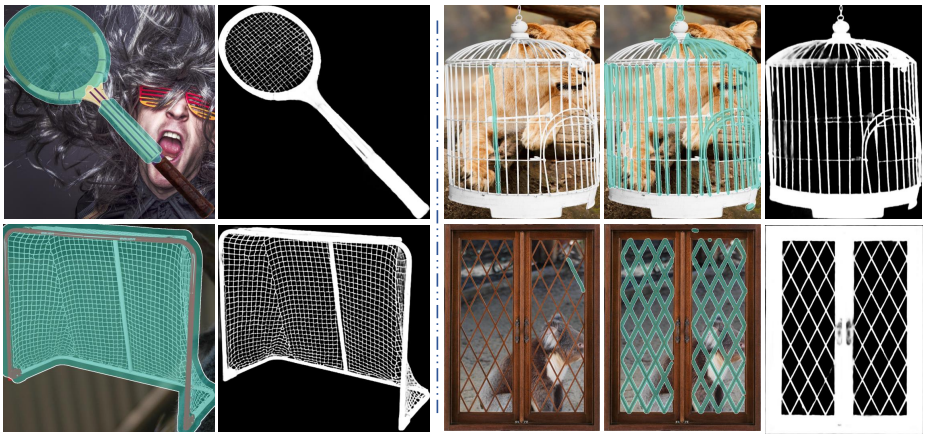


Fig. 5. **Challenging cases.** From left to right: (i) extremely coarse scribble annotation, (ii) prediction, (iii) human-annotated scribble, (iv) SDM-generated scribbles, (v) prediction.

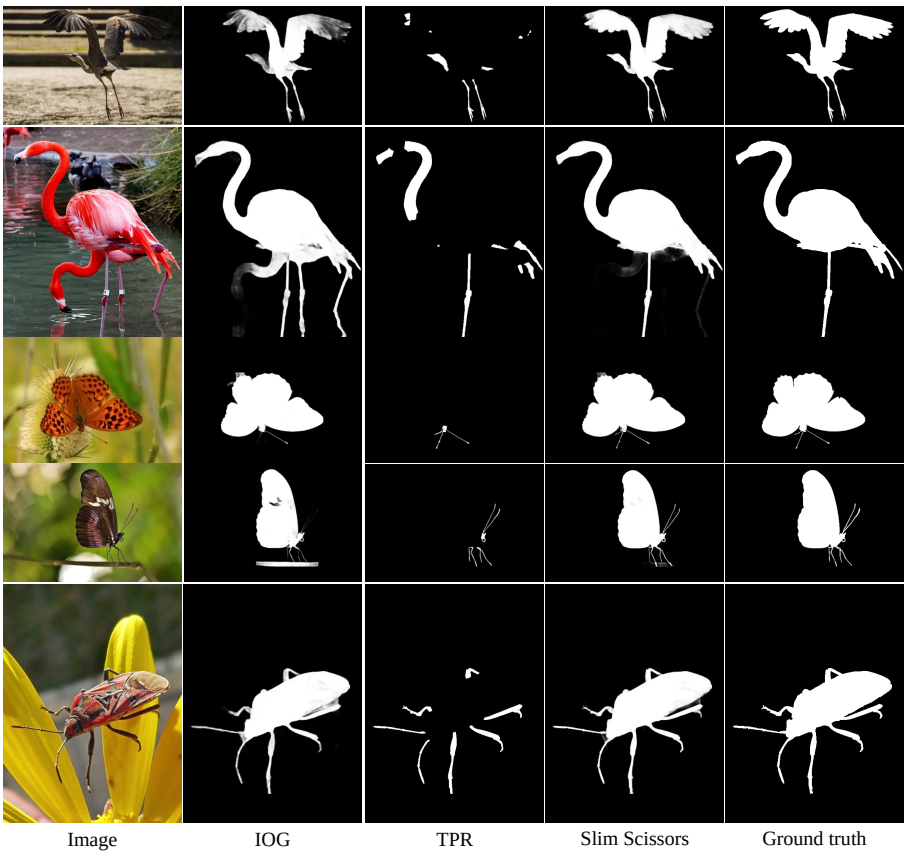


Fig. 6. Additional qualitative result on COIFT dataset

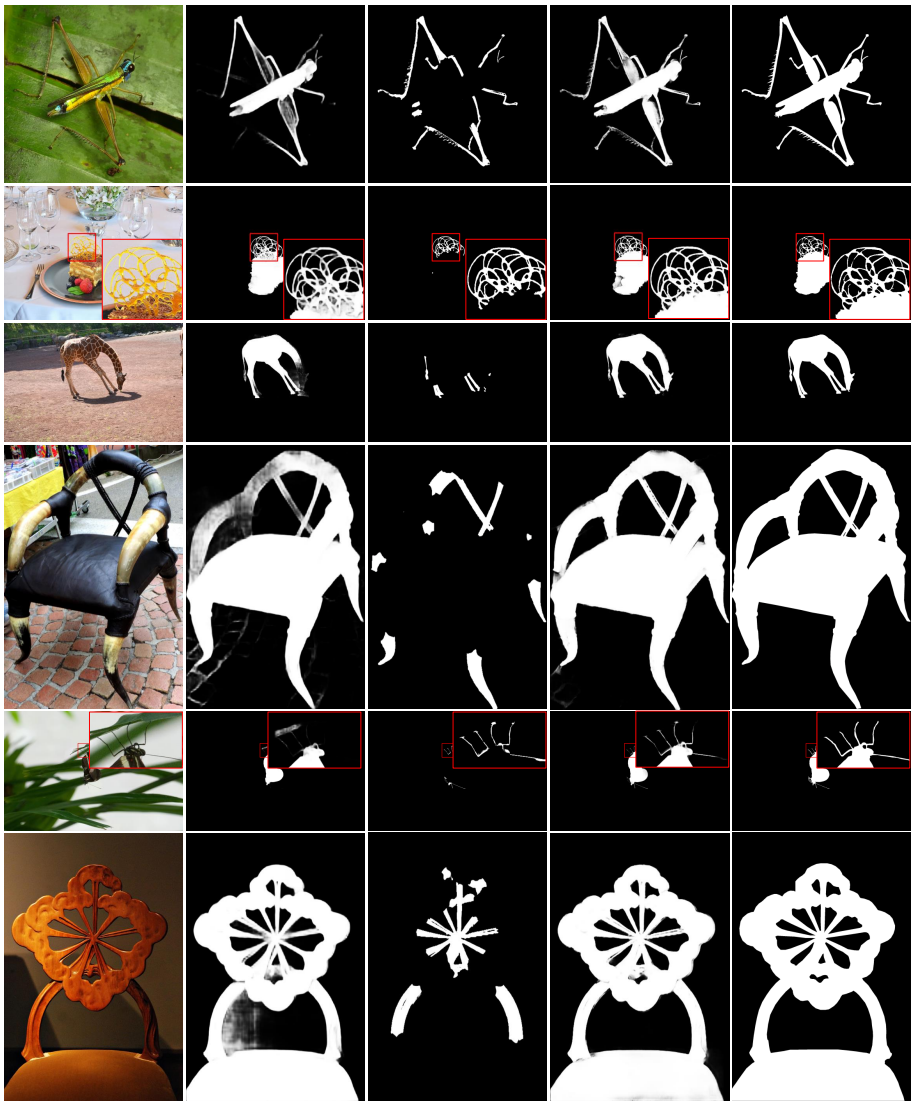


Fig. 7. Additional qualitative result on HRSOD dataset

References

1. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. TPAMI (2018)
2. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
3. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift (2015)
4. Mansilla, L.A., Miranda, P.A.: Oriented image foresting transform segmentation: Connectivity constraints with adjustable width. In: SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI) (2016)
5. Zeng, Y., Zhang, P., Zhang, J., Lin, Z., Lu, H.: Towards high-resolution salient object detection. In: ICCV (2019)
6. Zhang, S., Liew, J.H., Wei, Y., Wei, S., Zhao, Y.: Interactive object segmentation with inside-outside guidance. In: CVPR (2020)