S4Net: Single Stage Salient-Instance Segmentation

Ruochen Fan[1,2] Ming-Ming Cheng[3] Qibin Hou[3] Tai-Jiang Mu[1,2] Jingdong Wang[4] Shi-Min Hu[1,2]
[1]Tsinghua University [2]BNRist [3]Nankai University [4]MSRA

Abstract

We consider an interesting problem—salient instance segmentation in this paper. Other than producing bounding boxes, our network also outputs high-quality instance-level segments. Taking into account the category-independent property of each target, we design a single stage salient instance segmentation framework, with a novel segmentation branch. Our new branch regards not only local context inside each detection window but also its surrounding context, enabling us to distinguish the instances in the same scope even with obstruction. Our network is end-to-end trainable and runs at a fast speed (40 fps when processing an image with resolution 320×320*). We evaluate our approach on a public available benchmark and show that it outperforms other alternative solutions. We also provide a thorough analysis of the design choices to help readers better understand the functions of each part of our network. The source code can be found at* https://github.com/RuochenFan/S4Net.

1. Introduction

Rather than recognizing all the objects in a scene, we human only care about a small set of interesting objects/instances [32]. A recent experiment [15] demonstrates that interesting objects are often visually salient, reflecting the importance of detecting salient objects. Locating objects of interest is also essential for a wide range of computer graphics and computer vision applications. Such a capability allows many modern applications (*e.g.* image manipulation/editing [8, 56, 6] and robotic perception [55]) to provide initial regions that might be of interest to users or robots so that they can directly proceed to image editing or scene understanding. Like [33], in this paper, we are also interested in detecting *salient instances* given an input image. Similar to salient object detection, salient instance segmentation aims at detecting the most distinctive objects in a scene, but differently it also identifies each individual instance, *i.e.* outputting an accurate segment for each instance and assigning it a unique label (see Fig. 1).

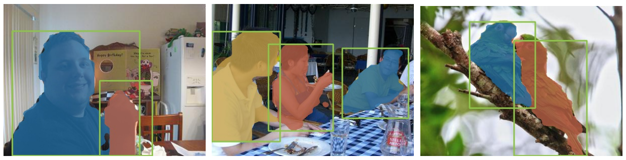


Figure 1: Illustrative examples produced by our approach, which detects and segments salient instances regardless of their semantic categories. Each category-agnostic salient instance is illustrated with a unique color.

Cognitive psychology [54, 14] and neurobiology [41] research suggested that human cortical cells may be *hard wired* to preferentially respond to *high contrast* stimulus, *i.e.* feature separation between foreground and background regions plays a central role in salient object perception [30, 29]. Effectively modeling the foreground/background separation using local and global contrast [7, 31], background prior [62], and Gaussian mixture color model (as in GrabCut [45]), *etc.*, has been proven to be useful in a variety of traditional salient-object/figure-ground segmentation tasks. Recently, convolutional neural networks (CNNs) are becoming the dominant methods in nearly all closely related tasks, *e.g.* salient object detection [26, 35, 52], semantic instance segmentation [11, 21, 22], and generic object detection [20, 44, 12]. While these CNN-based methods have achieved remarkable success by learning powerful multi-level feature representations for capturing different abstracted *appearance variations* of the target categories, they often ignore the important *feature separation ability between the target objects and their nearby background*.

Existing CNN-based instance segmentation methods use either RoIPooling [19, 24], or RoIWarp [10], or RoIAlign [23] to capture the *feature information inside the bounding boxes*. In contrast, we propose a region feature extraction layer, namely *RoIMasking*, to explicitly *incorporate foreground/background separation* for improving salient instance segmentation. Similar to the figure-ground segmentation method, GrabCut [45], we explicitly mark the region surrounding the object proposals as the initial background, and explore the foreground/background feature separations for salient instance segmentation in our segmentation branch. More specifically, we flip the signs of the feature values surrounding the proposals. The RoIMasking layer based segmentation branch is then integrated to the efficient single-stage object detector FPN [37], for detecting the pixel-wise segment of each salient instance. Interestingly, our RoIMasking scheme is quantization-free and scale-preserving, allowing more detailed information to be successfully detected. Furthermore, our model is end-to-end trainable and runs at 40fps on a single GPU when processing a 320 × 320 image.

For verification in the context of the killer application, we apply our salient instance detector to the popular weakly-supervised semantic segmentation task. As done in [53, 27], we use the detected salient instances on the tiny ImageNet dataset [46, 27] as heuristics to train the famous semantic segmentation networks. We evaluate the results on the popular PASCAL VOC 2012 semantic segmentation benchmark [16] and show that our results outperform state-of-the-art methods [53, 27] that leverage traditional salient object cues [31, 26] by a large margin.

To sum up, the contributions of this paper are:

- We propose an end-to-end single-shot salient instance segmentation framework, which not only achieves the state-of-the-art performance but also runs in real time.
- We design a new RoIMasking layer which models feature separation between target objects and its nearby background for high-quality segmentation.

2. Related Works

Salient instance segmentation is relatively a new task. Some seminal methods have recently been proposed by Zheng *et al.* [59] to find salient objects at bounding box level. However, this method misses the important segmentation information, which is essential for applications such as image editing [8, 6] and weakly supervised segmentation [53]. Li *et al.* [33] formally define the salient instance segmentation problem as jointly identifying salient regions as well as individual object instances. They also proposed an MSRNet [33] framework for instance-level salient object segmentation. However, this method was excessively reliant on the quality of the pre-computed edge maps (*e.g.* MCG [43]) and produced sub-optimal results for complicated real-world scenes (see also Sec. 4). Salient instance segmentation is closely related to three major computer vision tasks: salient object detection, object detection, and semantic instance segmentation.

2.1. Salient Object Detection

Salient object detection aims at jointly detecting the most distinguished objects and segmenting them out from a given scene. Early salient object detection methods mostly depended on either global or local contrast cues [**?**, 7, 5, 31]. They designed various hand-crafted features (e.g., color histogram and textures) for each region [2, 17, 48] and fused these features in either manual-designed [7] or learning-based manners [51]. Because of their weak ability to preserve the integrity of salient instances and the instability of hand-crafted features, these methods were gradually taken place by later CNN-based data-driven methods [26, 35, 52, 61, 18, 34, 33]. The key problems of these salient object detection methods when applied to salient instance segmentation task are two-fold. First, the integrity of the salient objects is difficult to be preserved because the distinguished regions might be parts of the interesting instances. Second, salient object detection is a binary problem and hence cannot be competent to instance-level segmentation.

2.2. Object Detection

The goal of object detection is to produce all the bounding boxes for semantic categories. Earlier work mostly relied on hand-engineered features (*e.g.* SIFT [40], SURF [4], and HOG [13]). They built different types of image pyramids to leverage more information across scales. Recently, the emergence of CNNs greatly promoted the development of object detectors. For example, R-CNN [20] and OverFeat [47] regarded CNNs as sliding window detectors for extracting high-level semantic information. Given a stack of pre-computed proposals [50, 9], these methods computed its feature vectors for each proposal using CNNs and then fed the features into a classifier. Later work [24, 19] took as inputs the entire images and applied region-based detectors to feature maps, substantially accelerating the running speed. Faster R-CNN [44] broke through the limitation of using pre-computed proposals by introducing a region proposal network (RPN) into CNNs. In this way, the whole network could be trained end-to-end, offering a better trade-off between accuracy and speed compared to its previous work. However, all the methods discussed above aim at outputting reliable object bounding boxes rather than instance segments.

2.3. Semantic Instance Segmentation.

Earlier semantic instance segmentation methods [11, 21, 22, 42] were mostly based on segment proposals generated by segmentation methods [50, 43, 3]. In [10], Dai *et al.* predicted segmentation proposals by leveraging a multi-stage cascade to refine rectangle regions from bounding box proposals gradually. Li *et al.* [36] proposed to integrate the segment proposal network into an object detection network. More recently, He *et al.* implemented a Mask R-CNN framework, extending the Faster R-CNN [44] architecture by introducing a segmentation branch. Albeit more and more fascinating results, these methods are not suitable for our task for two reasons. First, not all the categories and objects are salient. Second, the semantic instances all belong to a pre-defined category collection, missing the important

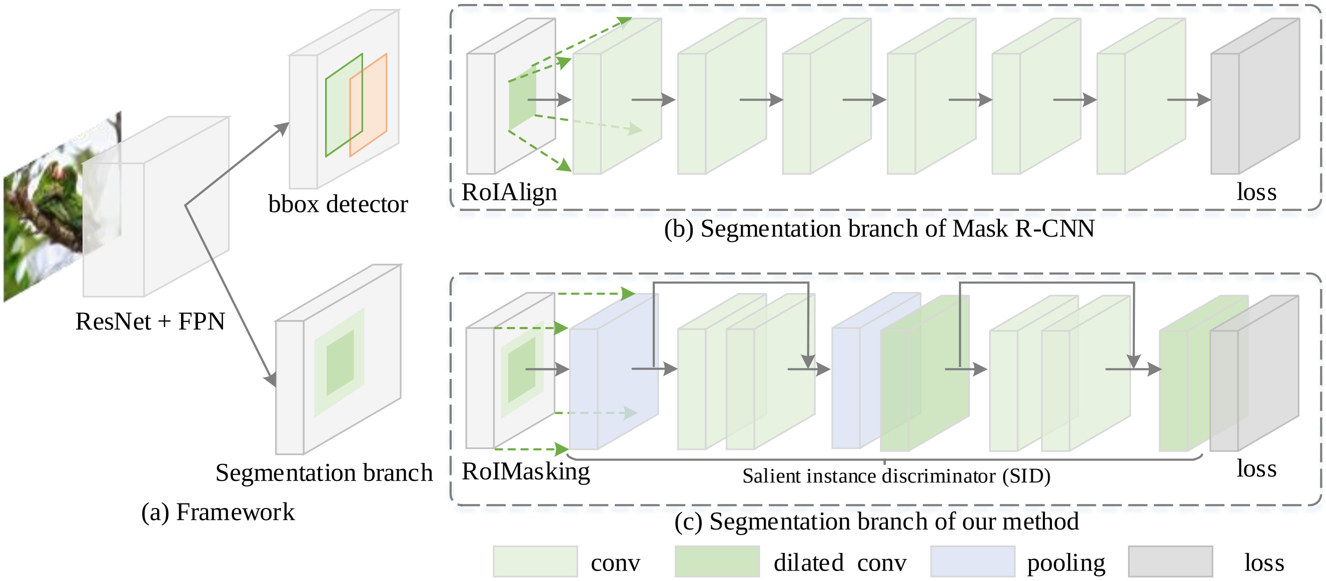

ResNet + FPN
bbox detector
Segmentation branch
(a) Framework
RoIAlign
loss
(b) Segmentation branch of Mask R-CNN
RoIMasking
Salient instance discriminator (SID)
loss
(c) Segmentation branch of our method
conv
dilated conv
pooling
loss

Figure 2: The pipeline of the proposed method. (a) A brief illustration of our framework. For convenience, we do not show the details of the backbone we adopt. Readers may refer to [37] for more information. (b) The segmentation branch proposed in Mask R-CNN [23], which is composed of a stack of consecutive convolutional layers. (c) Our proposed segmentation branch which further enlarges the size of the receptive field but with the same number parameters as in (b).

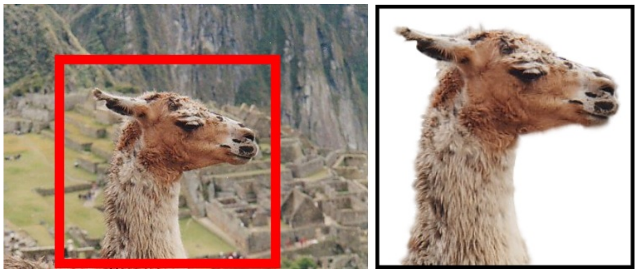


Figure 3: An example of interactive figure-ground segmentation using GrabCut [45].

ability to deal with unknown categories, *i.e. class-agnostic salient instances*.

3. S4Net

The design choices of our method are based on the application requirements of high-quality salient instance segmentation in real time. We design an end-to-end single-shot salient instance segmentation framework —*S4Net*, which is built upon the top of the state-of-the-art single-shot object detector for efficiency consideration.

3.1. Observation

Recent instance-level semantic segmentation methods [23, 36] have shown the strong ability to segmenting semantic instances, using RoIWarp [10], or RoIAlign [23]. However, the segmentation branches of these methods *only focus on the features inside the proposals* to describe the appearance variations of the target instances themselves, lacking the ability to distinguish different instances.

Before CNN-based methods became popular, utilizing feature separation between foreground and background has been the dominant mechanism in similar tasks such as salient object detection [7, 31, 62] and figure-ground segmentation [45]. The ability of effectively modeling the foreground-background feature separation is so powerful that these methods [7, 45, 62] could achieve *remarkable success by utilizing such feature separation in the target image alone, without any additional information by training on many images*. An example is shown in Fig. 3. Users only need to draw a rectangle region (shown in red) around the target object. The GrabCut method [45] initializes the foreground/background color models, *i.e.* Gaussian Mixture Models (GMM), using image pixels inside/outside the rectangle region respectively. Amazing segmentation results could be achieved, without learning from other training images. Notice that the color of some target object regions in this image is very similar to certain background regions (*i.e.* the houses). However, the GMM color model effectively captures the slight color difference (indistinguishable to human eyes) in this specific image. Such slight color difference only exists in this specific image, and could be impossible to learn from training examples of similar scenes. It means that many training examples for color feature modelling will not only be expensive to collect but also be less useful for dealing with such a situation.

Unfortunately, the ability of involving such powerful foreground-background feature separation has been missing in existing CNN-based segmentation methods. Motivated by this, we propose to explicitly leverage more features corresponding to the background area to help make the salient instances more prominent as shown in Fig. 5. This scheme allows more features representing the background area (relative to the salient instance) to be viewed by the segmentation branch, enlarging the receptive field of the segmentation branch and meanwhile enhancing the contrast between foreground and background, especially when there are occlusions.

3.2. Framework

The pipeline of S4Net is shown in Fig. 2, which involves two components: a bounding box detector and a segmentation branch. Both components share the same base model. As in most object detection works, we select ResNet-50 [25] as our base model.

Single-Shot Object Detector. Considering the efficiency of the entire network, we adopt a single-shot object detector [38] with FPN [37] as the base model in order to leverage the multi-level features. To reduce the runtime cost, we discard the lateral connected to conv2 while keeping the rest unchanged (*i.e.* conv3-conv6). Four detection heads are connected to each lateral layer as shown in Fig. 2(a). The head structure is the same to the one used in Faster R-CNN [44], but with different strides to perform detection at multiple scales.

Single-Shot Segmentation Branch. Different from existing instance level semantic segmentation methods, such as Mask R-CNN [23], our segmentation branch is also single-shot. The bounding boxes predicted by the detection branch and the output of the lateral layer with stride 8 in the backbone network are fed into our segmentation branch. As shown in Fig. 2(a), our segmentation branch contains a RoIMasking layer for instance feature extraction and a salient instance discriminator for identifying salient instances.

3.3. RoIMasking

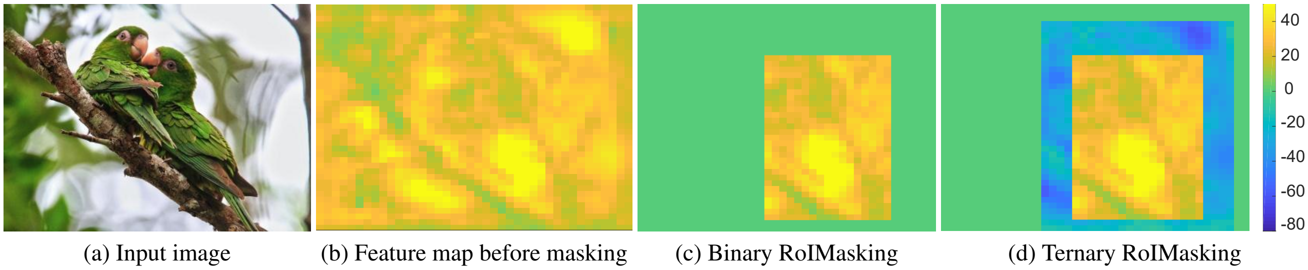
RoIPool [19] and RoIAlign [23] are two standard operations for extracting fixed-size features from the regions of interest. Both RoIPool and RoIAlign sample a region of interest into a fixed spatial extent of $H \times W$, and typically $H = W$, *e.g.* 7×7 in [19] and 28×28 in [23]. The RoIPool first quantizes RoIs by uniformly dividing them into $H \times W$ spatial bins. After max-pooling each spatial bin, the output feature maps with size $H \times W$ can be generated. Since the quantization in RoIPool is performed by rounding operation, it introduces misalignments between the RoI and the extracted features. As a remedy, RoIAlign avoids quantization by using bilinear interpolation.

However, both RoIPool and RoIAlign focus on the regions inside the proposals, neglecting the rest region. As discussed in Sec. 3.1, the region surrounding the current object RoI contains valuable information for distinguishing between the target object and its background. Unfortunately, although some layer-fusion techniques such as feature pyramid network [37] attempt to embed high-level and comprehensive information in a feature map, both RoIPool and RoIAlign do not explicitly and effectively explore the information surrounding the RoI. Moreover, the sampling process in these two operations makes these operations unable to maintain the aspect ratio and resolution of the regions of interest, possibly hurting the quality of the results. In this subsection, we design a new resolution-preserving and quantization-free layer, called *RoIMasking*, to take the place of RoIPool or RoIAlign. We also attempt to explore feature separation between foreground and background regions for improving segmentation quality.

Binary RoIMasking. We first introduce a simplified version of RoIMasking which we call binary RoIMasking. The binary RoIMasking receives feature maps and proposals predicted by the detection branch. A binary mask is generated according to the position and size of a given rectangle proposal. The values inside the rectangle are set to 1 and otherwise 0. Fig. 5a illustrates a binary version of RoIMasking, in which the bright and dark areas are associated with labels 1 and 0, respectively. The output of the binary RoIMasking layer is the input feature maps multiplied by this mask. In Fig. 4, we show a typical example of the output feature maps. Different from RoIPool [19] and RoIAlign [23], our binary RoIMasking keeps the original aspect ratio and resolution of the feature maps. In Sec. 4, we experimentally verify that the proposed binary RoIMasking outperforms the RoIPool and RoIAlign baselines.

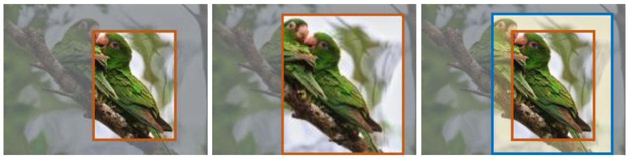
Expanded Binary RoIMasking. In this paragraph, we also consider an extensive version of binary RoIMasking by simply enlarging the proposal region as illustrated in Fig. 5b. Compared to the standard binary RoIMasking, expanded binary RoIMasking takes into account more background/context information, which means the segmentation branch has a larger receptive field. We will show more quantitative comparisons in our experiment section.

Ternary RoIMasking. To make better use of the background information around the regions of interest, we further advance the expanded binary RoIMasking to a ternary case. Because of the ReLU activation function, there are no negative values in the feature maps before RoIMasking. To explicitly notify the segmentation branch that the region



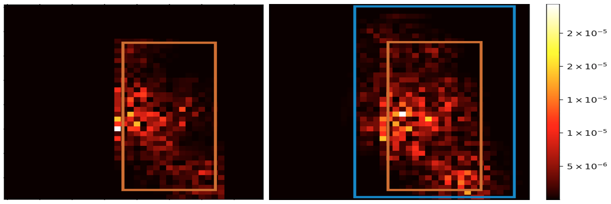
(a) Input image (b) Feature map before masking (c) Binary RoIMasking (d) Ternary RoIMasking

Figure 4: The output feature maps of two different types of RoIMasking layers. (b) Before RoIMasking, all of the values in the feature map are non-negative because of the ReLU layer. (c) After binary RoIMasking, the regions outside the proposal are set to zeros. (d) Ternary RoIMasking additionally considers a larger area, in which feature values are non-positive.



(a) Binary mask (b) Ex-Binary mask (c) Ternary mask

Figure 5: Three different types of masks used in our RoIMasking layer. (a) Binary mask only considers the regions inside the orange rectangle; (b) Expanded binary mask considers a larger rectangle area than binary masking; (b) The ternary mask takes into account both the region inside the orange rectangle and its surrounding regions marked in yellow.



(a) Binary masking (b) Ternary masking

Figure 6: Gradient maps using binary masking and ternary masking. As can be seen, ternary masking considers more perimeter information of the region around the proposal. The input image in this experiment is shown in Fig. 4(a).

outside the proposals should be considered as background, we flip the signs (*i.e.* set the corresponding mask values to -1) of the feature values around the region of interest, which is illustrated in yellow color in Fig. 5c. In this way, the features around regions of interest are distinct from those inside the bounding boxes of the salient instances. This allows the segmentation branch to be able to not only make use of features inside the region of interest as well as the surrounding context (as in extended binary RoIMasking), but also explicitly emphasis on foreground/background feature separation. The feature map after ternary RoIMasking is illustrated in Fig. 4d. It is worth mentioning that this operation introduces no additional computation cost into our model. Ternary RoIMasking leads to a large improvement as we show in the experiment part (Sec. 4). In the following, we *abbreviate ternary RoIMasking as RoIMasking* for notational convenience unless otherwise noted.

3.4. Analysis of RoIMasking

This subsection demonstrates the importance of the background information around the regions of interest in the feature maps and the effectiveness of ternary RoIMasking. To do so, we explore the impact of each activation in the feature maps before RoIMasking on the performance. Inspired by [57], we visualize the function of a specific neuron in this model by drawing a gradients map. After loading the fully trained model weights, we do a forward pass using a specific image. In this process, the activation value of the feature maps before RoIMasking, $H_{i,j,c}$, is extracted and stored. Next, we do a backward pass. Note that in the general training stage, back-propagation is performed to calculate the gradients of the *total loss* with respect to the *weights* in the neural network. But in this experiment, we load the stored $H_{i,j,c}$ as a variable, and regard the convolution kernels as constant. Back-propagation is performed to calculate the gradients of the *instance segmentation loss* with respect to each *feature map* input to RoIMasking, *i.e.* $G_{i,j,c} = \partial L_{sal} / \partial H_{i,j,c}$. The absolute value of $G_{i,j,c}$ reflects the importance of the feature map pixel $H_{i,j,c}$ to the saliency task. After summing up $|G_{i,j,c}|$ along the channel dimension, the gradient map $G_{i,j}$ can be obtained.

Fig. 6 shows the gradient maps for binary RoIMasking and ternary RoIMasking, respectively. The orange rectangle is the ground truth bounding box of a salient instance. By definition, the pixels inside the orange rectangle in the ternary mask are set to 0 and the pixels between the orange and blue boxes are set to -1. It is obvious that there are evident responses in the background (marked as '-1' in the ternary mask) area in Fig. 6b. In Fig. 6a, there are only few responses between the orange and blue boxes. This phenomenon indirectly indicates the importance of the context information around the regions of interest. More experimental results can be found in the experiment section.

3.5. Segmentation Branch

Taking into account the structure of our backbone, we take the feature maps from the lateral layer associated with conv3 with a stride of 8 as the input to our segmentation branch on the trade-off between global context and details. Before connecting our RoIMasking layer, we first add a simple convolutional layer with 256 channels and kernel size 1×1 for compressing the number of channels. Despite the RoIMasking layer, it is still difficult to distinguish the salient instances from the other instances inside the same RoI. To this end, we add a segmentation branch similar to Mask-RCNN [23] to help better distinguish the instances.

As pointed out in [60], enlarging receptive field is helpful for segmentation related tasks. Inspired by [60, 23], we design a new segmentation branch by introducing skip connections and dilated convolutional layers (See Fig. 2c). Other than two residual blocks, we also add two 3×3 max pooling with stride 1 and dilated convolutional layers with dilation rate 2 for enlarging the receptive field. All the convolutional layers have a kernel size 3×3 and stride 1. For the channel numbers, we set the first three to 128 and the rest 64, which we found are enough for salient instance segmentation.

3.6. Loss function

As described above, there are two sibling branches in our framework for detection and saliency segmentation, respectively. The detection branch undertakes objectness classification task and coordinates regression task, and the segmentation branch is for saliency segmentation task. Therefore, we use a multi-task loss L on each training sample to jointly train the model:

$$L = L_{obj} + L_{coord} + L_{seg}. \tag{1}$$

Regarding the fact that positive proposals are far less than negative samples in the detection branch, we adopt the following strategy. Let P and N be the collections of positive and negative proposals, N_P and N_N be the numbers of positive and negative proposals ($N_P \ll N_N$), then we calculate the positive and negative objectness loss separately to avoid the domination of negative gradients during training. Thus we have:

$$L_{obj} = -(\frac{1}{N_P}\sum_{i \in P} \log p_i + \frac{1}{N_N}\sum_{j \in N} \log(1 - p_j)), \tag{2}$$

in which p_i is the probability of the ith proposal being positive.

We use $Smooth_{L1}$ loss as Fast-RCNN [19] for coordinate regression and cross-entropy loss similar to Mask-RCNN [23] for the segmentation branch.

4. Experiments

In this section, we carry out detailed analysis to elaborate the functions of each component in our method by ablation studies. We also perform thorough comparisons with the state-of-the-art methods to exhibit the effectiveness of our approach. We use the dataset proposed in [33] for all experiments. This dataset contains 1,000 images with well-annotated instance-level annotations. For fair comparisons, as done in [33], we randomly select 500 images for training, 200 for validation, and 300 for testing.

Methods	mAP$^{0.5}$	mAP$^{0.7}$	mAP$_O^{0.5}$	mAP$_O^{0.7}$
RoIAlign [23]	85.2%	61.5%	79.2%	47.7%
RoIPool [19]	85.2%	61.1%	80.3%	50.9%
Binary RoIMasking	85.5%	62.4%	80.1%	49.4%
Ternary RoIMasking	86.7%	63.6%	81.2%	51.5%

Table 1: Ablation experiments for analyzing our RoIMasking layer. We also list the results using the RoIAlign and RoIPool proposed in Mask R-CNN [23] and Fast R-CNN [19], respectively. Obviously, our proposed RoIMasking outperforms RoIAlign and RoIPool even for the images with occlusion.

4.1. Implementation Details

Training and Testing. In the training phase, the IoU is used to determine whether a bounding box proposal is a positive or negative sample in the detection branch. A bounding box proposal is positive if it's IoU > 0.5, and negative if IoU < 0.5.

In the testing phase, the bounding boxes fed into the RoIMasking layer are from the detection branch. But in the training phase, we directly feed the ground truth bounding boxes into the RoIMasking layer. This provides the segmentation branch with more stable and valid training data and meanwhile accelerates the training process, as been verified by empirical experiments.

Hyper-parameters. Our proposed network is based on the TensorFlow library [1]. The input images are augmented by horizontal flipping. The hyper-parameters are set as follows: weight decay (0.0001) and momentum (0.9). We train our network on 2 GPUs for 20k iterations, with an initial learning rate of 0.004 which is divided by a factor of 10 after 10k iterations.

4.2. Ablation Studies

To evaluate the effectiveness of each component in our proposed framework, we train our model on the salient

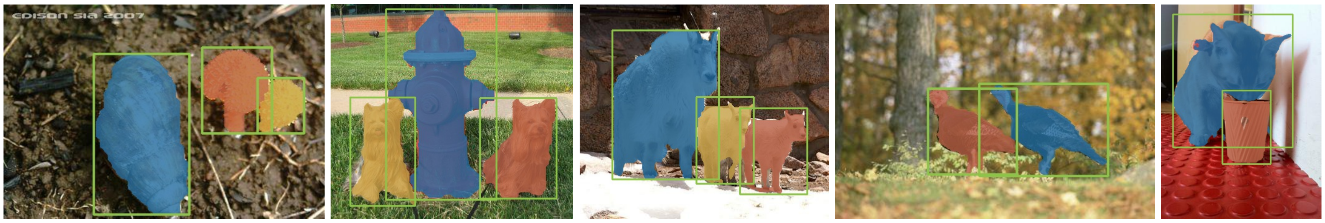


Figure 7: Selected examples of instance-level saliency segmentation results on the dataset proposed by [33] (the line above) and COCO [39] (the line below). Even obstructed instances can be well distinguished and segmented by our S4Net.

instance segmentation dataset [33]. Following the standard COCO metrics [39], we report results on mAP (averaged precision over IoU thresholds), including mAP$^{0.5}$ and mAP$^{0.7}$. Furthermore, to analyze the ability to distinguish different instances, we also consider another set which only contains instances with occlusion, which is denoted by mAP$_O$.

Effect of RoIMasking. To evaluate the performance of the proposed RoIMasking layer, we also consider using RoIPool [19] and RoIAlign [23]. We replace our RoIMasking with RoIPool and RoIAlign to perform two comparison experiments while keep all other network structures and experimental settings unchanged. Quantitative evaluation results are listed in Tab. 1. As can be seen, our proposed binary RoIMasking and ternary RoIMasking both outperform RoIPool and RoIAlign in mAP$^{0.7}$. Specifically, our ternary RoIMasking result improves the result using RoIAlign by around 2.1 points. This reflects that considering more context information outside the proposals does help for salient instance segmentation.

To further verify the effectiveness of our RoIMasking layer, we also consider a binary masking case in which the values in the yellow area of Fig. 5d are all set to 1. The penultimate line in Table 1 shows the corresponding results. As can be seen, the results are even worse when the binary masking is used. This fact reflects that simply enlarging the regions of interest is not helpful for discriminating salient instances. However, when the signs of the extended regions in the mask are flipped (ternary RoIMasking), the best results can be obtained (the bottom line of Table 1). This demonstrates that changing the signs of the extended regions in the mask can explicitly increase the contrast between the salient instances and background. More importantly, the non-salient regions inside the proposals will tend to be predicted to the same class as the extended regions. Therefore, the feature separation ability between the target objects and their nearby background plays an important role in our approach.

Size of Context Regions. To better understand our RoIMasking layer, we analyze how large the context regions should be here. Suppose the bounding box size of a salient instance is (w, h), where w and h are the width and height, respectively. We define an expansion coefficient α to denote the width of the '-1' region in the RoI mask. So, the size of the valid region is $(w + 2\alpha w, h + 2\alpha h)$. By default, we set α to 1/3. We also try different values of α to explore its influence on the final results as shown in Table 2 but found both larger and smaller values of α slightly harms the performance. This indicates that a region size of $(w + 2w/3, h + 2h/3)$ is enough for discriminating different instances as larger '-1' region may make more salient instances be viewed, weakening the performance of identifying the 'real' salient instances.

α	0	1/6	1/3	1/2	2/3	1
mAP$^{0.5}$	85.9%	86.4%	**86.7%**	86.5%	86.2%	85.9%
mAP$^{0.7}$	62.5%	63.4%	**63.6%**	63.3%	62.4%	62.0%

Table 2: Performance of S4Net with different expanded areas. All the results shown here are based on ResNet-50 [25]. As can be observed, when $\alpha = 1/3$ we obtain the best result.

Number of Proposals. The number of proposals sent to the segmentation branch also effects the performance. According to our experiments, more proposals lead to better performance but more computational costs. Notice that the performance gain is not obvious when the number of proposals exceeds 20. Specifically, when we set the number of proposals to 100, only around 1.5% improvement can be achieved but the runtime cost increases dramatically. Taking this into account, we take 20 proposals as a trade-off during the inference phase. Users may decide the number of proposals by their tailored tasks.

Base Models. Besides the base model of ResNet-50 [25], we also try another three popular base models, including Resnet-101 [25], VGG16 [49], and MobileNet [28]. Tab. 3 lists the results when different base models are utilized. As one can see, base models with better performance on classification also works better in our experiments. For speed, real-time processing can be achieved by our pro-

Base models	mAP@0.5	mAP@0.7	Speed (FPS)
ResNet-101 [25]	**88.1%**	**66.8%**	33.3
ResNet-50 [25]	86.7%	63.6%	40.0
VGG16 [49]	82.2%	53.0%	43.5
MobileNet [28]	62.9%	33.5%	90.9

Table 3: Performance of S4Net when using different base models. When we change the default ResNet-50 to ResNet-101, another 3.2% improvement can be obtained in spite of a little sacrifice on time cost. We also attempt to use the recent MobileNet [28] as our base model and yield a frame rate of more than 90 fps on a GTX 1080 Ti GPU.

Methods	$\text{mAP}^{0.5}$	$\text{mAP}^{0.7}$	$\text{mAP}_O^{0.5}$	$\text{mAP}_O^{0.7}$
MSRNet [33]	65.3%	52.3%	-	-
S4Net	**86.7%**	**63.6%**	81.2%	51.5%

Table 4: Quantitative comparisons with existing methods on the 'test' set. As the instance segmentation maps of [33] and related code are not available, thus we use '-' to fill the blank cells.

posed S4Net. When the size of input images is 320 × 320, S4Net has a frame rate of 40.0 fps on a GTX 1080 Ti GPU. Furthermore, using MobileNet [28] as our base model, S4Net runs very fast at a speed of 90.9 fps.

4.3. Comparisons with the State-of-the-Arts

Unlike salient object detection which has been studied for years, salient instance detection is a relatively new problem such that there is only one related work MSRNet [33] that can be used for direct comparison. In this experiment, we compare our S4Net based on ResNet-50 [25] with the MSRNet method. We report the results on the 'test' set using $\text{mAP}^{0.5}$ and $\text{mAP}^{0.7}$ metric.

Quantitative Analysis. Two datasets are used in our comparison experiments. The results of comparative experiments on dataset proposed by [33] are listed in Tab. 4. Our proposed S4Net achieves better results in both $\text{mAP}^{0.5}$ and $\text{mAP}^{0.7}$ compared to MSRNet [33]. Specifically, our approach improves the baseline results presented in MSRNet [33] by about 21 points in $\text{mAP}^{0.5}$. Regarding $\text{mAP}^{0.7}$, we also have a great improvement on the same dataset.

5. Applications

In this section, we apply our proposed S4Net to a popular vision task—weakly-supervised semantic segmentation. For training samples with multiple keywords, such as the images in PASCAL VOC [16], discriminating different instances is even essential for keyword assignment. The detailed methodology can be seen in our supplementary material. The results is shown in Table 5. It is obvious that training with our instance segmentation on the same dataset works much better than the settings in which other heuristic cues are used. Our approach obtains a 4.8% performance gain compared to using the DSS salient object detector [26].

Model	Heuristic cues	val set
DeepLab-VGG16	Sal maps [31]	49.8%
DeepLab-VGG16	Sal maps [26]	52.6%
DeepLab-VGG16	Att [58] + Sal [26]	53.8%
DeepLab-VGG16	Salient Instances	57.4%
DeepLab-ResNet101	Salient Instances	**61.8%**

Table 5: Semantic segmentation results with different initial heuristic cues on the PASCAL VOC validation set. The best result is highlighted in **bold**. Due to the space limitation, we use abbreviations for convenience. As can be seen, training with our instance-level saliency cues greatly outperforms settings with regular saliency cues.

6. Conclusions

In this paper, we present a single stage salient-instance segmentation framework, which is able to segment instance-level salient objects in real time. The key novelties include (i) the ROIMasking layer, which takes into account both the information inside the proposals and the context information outside the proposals and preserves the original resolution and aspect ratio of the regions of interest, and (2) an advanced salient instance discriminator which enlarges the receptive field of our segmentation branch and thus boosts the performance. Thorough experiments show that the proposed RoIMasking greatly outperforms RoIAlign and RoIPool, especially for distinguishing instances in the same scope. Our S4Net achieves the state-of-the-art performance on a publicly available benchmark.

Acknowledgements

This research was supported by NSFC (61521002, 61572264, 61620106008), the Joint NSFC-ISF Research Program (project number 61561146393), Tsinghua-Tencent Joint Laboratory for Internet Innovation Technology, the national youth talent support program, Tianjin Natural Science Foundation (17JCJQJC43700, 18ZXZNGX00110) and the Fundamental Research Funds for the Central Universities (Nankai University, NO. 63191501). Shi-Min Hu is the correcponding author of the paper.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016. 6

[2] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE TPAMI*, 2012. 2

[3] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE TPAMI*, 33(5):898–916, 2011. 2

[4] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008. 2

[5] A. Borji, M.-M. Cheng, H. Jiang, and J. Li. Salient object detection: A benchmark. *IEEE TIP*, 24(12):5706–5722, 2015. 2

[6] T. Chen, M.-M. Cheng, P. Tan, A. Shamir, and S.-M. Hu. Sketch2photo: Internet image montage. *ACM TOG*, 28(5):124:1–10, 2009. 1, 2

[7] M. Cheng, N. J. Mitra, X. Huang, P. H. Torr, and S. Hu. Global contrast based salient region detection. *IEEE TPAMI*, 37(3):569–582, 2015. 1, 2, 3

[8] M.-M. Cheng, F.-L. Zhang, N. J. Mitra, X. Huang, and S.-M. Hu. Repfinder: finding approximately repeated scene elements for image editing. In *ACM Transactions on Graphics (TOG)*, volume 29, page 83. ACM, 2010. 1, 2

[9] M.-M. Cheng, Z. Zhang, W.-Y. Lin, and P. Torr. Bing: Binarized normed gradients for objectness estimation at 300fps. In *CVPR*, 2014. 2

[10] J. Dai, K. He, Y. Li, S. Ren, and J. Sun. Instance-sensitive fully convolutional networks. In *ECCV*, 2016. 1, 2, 3

[11] J. Dai, K. He, and J. Sun. Convolutional feature masking for joint object and stuff segmentation. In *CVPR*, 2015. 1, 2

[12] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks. In *NIPS*, 2016. 1

[13] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 2

[14] R. Desimone and J. Duncan. Neural mechanisms of selective visual attention. *Annual review of neuroscience*, 18(1):193–222, 1995. 1

[15] L. Elazary and L. Itti. Interesting objects are visually salient. *Journal of vision*, 2008. 1

[16] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *IJCV*, 2015. 2, 8

[17] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 2004. 2

[18] L. Gayoung, T. Yu-Wing, and K. Junmo. Deep saliency with encoded low level distance map and high level features. In *CVPR*, 2016. 2

[19] R. Girshick. Fast r-cnn. In *ICCV*, pages 1440–1448, 2015. 1, 2, 4, 6, 7

[20] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 1, 2

[21] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *ECCV*, 2014. 1, 2

[22] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *CVPR*, pages 447–456, 2015. 1, 2

[23] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *ICCV*, 2017. 1, 3, 4, 6, 7

[24] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE TPAMI*, 2015. 1, 2

[25] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4, 7, 8

[26] Q. Hou, M.-M. Cheng, X. Hu, A. Borji, Z. Tu, and P. Torr. Deeply supervised salient object detection with short connections. In *CVPR*, 2017. 1, 2, 8

[27] Q. Hou, P. K. Dokania, D. Massiceti, Y. Wei, M.-M. Cheng, and P. Torr. Bottom-up top-down cues for weakly-supervised semantic segmentation. In *EMMCVPR*, 2017. 2

[28] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 7, 8

[29] L. Itti and C. Koch. Computational modeling of visual attention. *Nature reviews neuroscience*, 2(3):194–203, 2001. 1

[30] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 20(11):1254–1259, 1998. 1

[31] H. Jiang, J. Wang, Z. Yuan, Y. Wu, N. Zheng, and S. Li. Salient object detection: A discriminative regional feature integration approach. In *CVPR*, pages 2083–2090, 2013. 1, 2, 3, 8

[32] F.-F. Li, R. VanRullen, C. Koch, and P. Perona. Rapid natural scene categorization in the near absence of attention. *Proceedings of the National Academy of Sciences*, 2002. 1

[33] G. Li, Y. Xie, L. Lin, and Y. Yu. Instance-level salient object segmentation. In *CVPR*, 2017. 1, 2, 6, 7, 8

[34] G. Li and Y. Yu. Visual saliency based on multiscale deep features. In *CVPR*, pages 5455–5463, 2015. 2

[35] G. Li and Y. Yu. Deep contrast learning for salient object detection. In *CVPR*, 2016. 1, 2

[36] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei. Fully convolutional instance-aware semantic segmentation. In *CVPR*, 2017. 2, 3

[37] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. In *CVPR*, volume 1, page 4, 2017. 2, 3, 4

[38] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 4

[39] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 7

[40] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004. 2

[41] S. K. Mannan, C. Kennard, and M. Husain. The role of visual salience in directing eye movements in visual object agnosia. *Current biology*, 19(6):R247–R248, 2009. 1

[42] P. O. Pinheiro, R. Collobert, and P. Dollár. Learning to segment object candidates. In *NIPS*, 2015. 2

[43] J. Pont-Tuset, P. Arbelaez, J. T. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping for image segmentation and object proposal generation. *IEEE TPAMI*, 2017. 2

[44] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE TPAMI*, 2017. 1, 2, 4

[45] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM TOG*, 2004. 1, 3

[46] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 2

[47] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *ICLR*, 2014. 2

[48] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE TPAMI*, 2000. 2

[49] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 7, 8

[50] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *IJCV*, 2013. 2

[51] J. Wang, H. Jiang, Z. Yuan, M.-M. Cheng, X. Hu, and N. Zheng. Salient object detection: A discriminative regional feature integration approach. *International Journal of Computer Vision*, 123(2):251–268, 2017. 2

[52] L. Wang, H. Lu, X. Ruan, and M.-H. Yang. Deep networks for saliency detection via local estimation and global search. In *CVPR*, pages 3183–3192, 2015. 1, 2

[53] Y. Wei, X. Liang, Y. Chen, X. Shen, M.-M. Cheng, J. Feng, Y. Zhao, and S. Yan. Stc: A simple to complex framework for weakly-supervised semantic segmentation. *IEEE TPAMI*, 2016. 2

[54] J. M. Wolfe and T. S. Horowitz. What attributes guide the deployment of visual attention and how do they do it? *Nature reviews neuroscience*, 5(6):495, 2004. 1

[55] C. Wu, I. Lenz, and A. Saxena. Hierarchical semantic labeling for task-relevant rgb-d perception. In *Robotics: Science and systems*, 2014. 1

[56] H. Wu, Y.-S. Wang, K.-C. Feng, T.-T. Wong, T.-Y. Lee, and P.-A. Heng. Resizing by symmetry-summarization. In *ACM Transactions on Graphics (TOG)*, volume 29, page 159, 2010. 1

[57] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015. 5

[58] J. Zhang, Z. Lin, J. Brandt, X. Shen, and S. Sclaroff. Top-down neural attention by excitation backprop. In *ECCV*, 2016. 8

[59] J. Zhang, S. Sclaroff, Z. Lin, X. Shen, B. Price, and R. Mech. Unconstrained salient object detection via proposal subset optimization. In *CVPR*, 2016. 2

[60] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *CVPR*, 2017. 6

[61] R. Zhao, W. Ouyang, H. Li, and X. Wang. Saliency detection by multi-context deep learning. In *CVPR*, pages 1265–1274, 2015. 2

[62] W. Zhu, S. Liang, Y. Wei, and J. Sun. Saliency optimization from robust background detection. In *CVPR*, pages 2814–2821, 2014. 1, 3