FastMask: Segment Multi-scale Object Candidates in One Shot

Hexiang Hu* † USC hexiangh@usc.edu

Shiyi Lan* † Fudan University sylan14@fudan.edu.cn

Yuning Jiang Megvii Inc. jyn@megvii.com

Zhimin Cao Megvii Inc. czm@megvii.com

Fei Sha USC feisha@usc.edu

Abstract

Objects appear to scale differently in natural images. This fact requires methods dealing with object-centric tasks (e.g. *object proposal) to have robust performance over variances in object scales. In the paper, we present a novel segment proposal framework, namely FastMask, which takes advantage of hierarchical features in deep convolutional neural networks to segment multi-scale objects in one shot. Innovatively, we adapt segment proposal network into three different functional components (body, neck and head). We further propose a weight-shared residual neck module as well as a scale-tolerant attentional head module for efficient one-shot inference. On MS COCO benchmark, the proposed FastMask outperforms all state-of-the-art segment proposal methods in average recall being 2∼5 times faster. Moreover, with a slight trade-off in accuracy, FastMask can segment objects in near real time (∼13 fps) with 800×600 resolution images, demonstrating its potential in practical applications. Our implementation is available on* https://github.com/voidrank/FastMask.

1. Introduction

Object proposal is considered as the first and fundamental step in object detection task [8, 25, 1, 16, 10, 29]. As the domain rapidly progressed, a renewed interest in object segment proposal has received intensive attentions [6, 20, 21, 5, 2]. Different from traditional object proposal methods, segment proposal algorithms are expected to generate a pixel-wise segment instead of a bounding box for each object. From this perspective, segment proposal inherits from both object proposal and image segmentation, and takes a step further towards simultaneous detection and segmentation [11], which brings more challenges to overcome.

Among all these challenges, how to tackle the scale variances in object appearance remains the most critical one. Compared to bounding-box-based (bbox-based) object proposal, scale variance becomes a more serious problem for segment proposal. It is due to that in segment proposal, a highly matched receptive field is demanded to distinguish the foreground object from background. In Figure 1 two examples are given to explain how a mismatched receptive field affects the segment proposal results: on one hand (Figure 1 (a)), when the receptive field of object proposer is much smaller than the object itself (*e.g.* perceiving only a window of a bus), the bbox-based proposer could still roughly estimate the bounding box with prior knowledge. However, the mission becomes almost impossible for a segment-based proposer as they need to imagine the complete contour of the bus; on the other hand (Figure 1 (b)), too large receptive field may introduce noises from backgrounds and result in the incorrect instance-level segments. For example, a segment-based proposer could be distracted by other people standing nearby the target person, leading to an inaccurate mask covering not only the target person. As a consequence, once the receptive field of a segment-based proposer is fixed, object scale variance will badly affect both segmentation fineness and proposal recall.

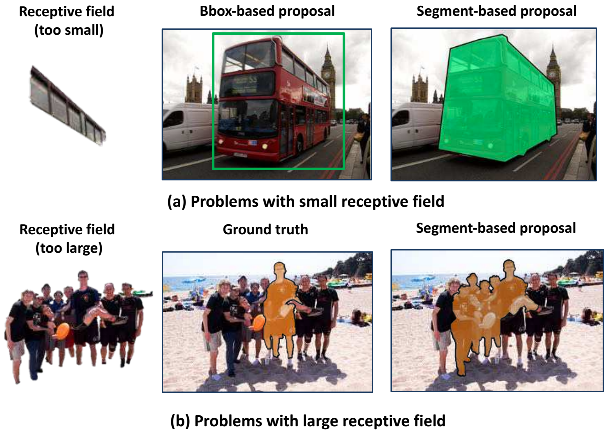
Receptive field (too small)
Bbox-based proposal
Segment-based proposal
(a) Problems with small receptive field
Receptive field (too large)
Ground truth
Segment-based proposal
(b) Problems with large receptive field

Figure 1. How a mismatched receptive field affects the segment proposal results. Refer to text for detailed explanation.

In general, existing methods [6, 20, 21, 5, 2] could be divided into two major categories by how they deal with scale variances. The first category [6, 2] uses extra bbox-based object proposals or object detections as initial inputs.

[*] Equal contribution.
[†] Work was done during their internships at Megvii Inc.

However, its effectiveness and efficiency are highly dependent on the accuracy and speed of pre-processing proposal methods. The second one [20, 21, 5] adopts the image pyramid strategy, in which the original image is rescaled and fed into a fixed-scale object proposer repeatedly for multi-scale inference (see Figure 3(a)). However, such multi-shot methods face a common dilemma: a densely sampled image pyramid becomes the computational bottleneck of the whole framework; nevertheless, reducing the number of the scales of image pyramid leads to performance degradation. Such methods could hardly provide satisfactory accuracy and speed at the same time. With the observation that the original image has already contained all information of an image pyramid, we argue that using one single image should be enough to capture all multi-scale objects in it.

Therefore, in this paper, we aim to address the scale variances in segment proposal by leveraging the hierarchical feature pyramid[9] from convolutional neural networks (CNN). We adapt segment proposal network into three different functional components, namely *body*, *neck* and *head*. Similar to [20, 21], the body and head module are responsible for extracting semantic feature maps from original images and decoding segmentation masks from feature maps, respectively. Furthermore, We introduce the concept of neck module, whose job is to recurrently zoom out the feature maps extracted by the body module into feature pyramids, and then feed the feature pyramids into the head module for multi-scale inference. We summarize our main contributions as follows:

- First, we learn a novel weight-shared residual neck module to build a feature pyramid of CNN while preserving a well-calibrated feature semantics, for efficient multi-scale training and inference.
- Next, we propose a novel scale-tolerant head module which takes advantage of visual attention and significantly reduces the impact of background noises caused by mismatched scales in receptive fields.
- Finally, together with all those modules, we make a framework capable of one-shot segment proposal. We evaluate our framework on MS COCO benchmark [18] and it achieves the state-of-the-art results while running in near real time.

2. Related Work

Bbox-based object proposal. Most of the bbox-based object proposal methods rely on the dense sliding windows on image pyramid. In EdgeBox [31] and Bing [4], the edge feature is used to make the prediction for each sliding window while the gradient feature is used in [29]. More recently, DeepBox [17] trains a CNN to re-rank the proposals generated by EdgeBox, while MultiBox [7] generates the proposals from convolutional feature maps directly. Ren et. al. [22] presented a region proposal network (RPN) is proposed to handle object candidates in varying scales.

Segment-based object proposal. Segments proposal algorithms aim to find diverse regions in an image which are likely to contain objects. Traditional segment proposal methods such as SelectiveSearch [25], MCG [1] and Geodesic [16] first over-segment image into super pixels and then merge the super pixels in a bottom-up fashion. Inspired by the success of CNNs in image segmentation [23, 3, 28], previous works [6, 2] perform segmentation on the bbox-based object proposal results to obtain object segments. As the state-of-the-arts, DeepMask [20] proposes a body-head structure to decode object masks from CNN feature maps, and SharpMask [21] further adds a backward branch to refine the masks. However, all these methods rely on an image pyramid during inference, which limits their application in practice.

Visual attention. Instead of using holistic image feature from CNN, a number of recent works [26, 19, 30, 27] have explored visual attention to highlight discriminative region inside images and reduce the effects of noisy background. In this paper we apply such attention mechanism to improve the instance-level segmentation performance.

3. From Multi-shot to One-Shot

DeepMask [20] is considered as the representative of the CNN-based multi-shot segment proposal methods, where a body-head structure is proposed. In this section, we briefly review DeepMask to help better understand the multi-shot paradigm and then proceed to our proposed one-shot paradigm.

Patch-based training. DeepMask is trained to predict a segmentation mask and a confidence score given a fixed-size image patch. In training, an image patch is assigned to be positive if it satisfies the *object-centric constrain* [20]; otherwise negative. All the image patches are cropped and rescaled into fixed size (*e.g.* 224×224). These patches are fed into the body network of DeepMask to extract semantic feature maps, and then decoded into the confidence scores and the segmentation masks using the head module.

Multi-shot inference. During multi-shot inference, DeepMask applies the trained model densely at each location, repeatedly across different scales. As shown in Figure 3 (a), at first the input image is resized repeatedly into an image pyramid. Next, the body network of DeepMask extracts a full feature map from each resized image. Finally the head module is applied on every fixed-size sliding window (e.g., 14×14) on multi-scale feature maps, to decodes the confidence score and mask for each sliding window.

For DeepMask and its variants[20, 21, 6], a densely sampled image pyramid is required during inference. However, as the convolutional computation over image pyramid is re-

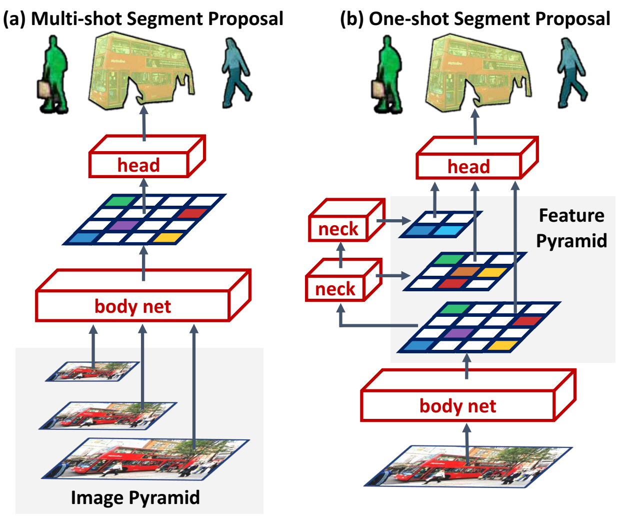
(a) Multi-shot Segment Proposal
head
body net
Image Pyramid
(b) One-shot Segment Proposal
head
neck
neck
Feature Pyramid
body net

Figure 3. Comparison between multi-shot paradigm and our one-shot paradigm.

dundant, the image pyramid has become the computational bottleneck in such multi-shot segment proposal methods.

To overcome the inefficiency brought by image pyramid, we propose a one-shot paradigm that enables efficient training and inference. As shown in Figure 3 (b), we inherit the body-head structure and introduce a new component called *neck*. This neck component could be used on the feature map and zoom it out into feature pyramid while preserving feature semantics. Then, a shared head module is applied on the pyramid of feature maps to decode object segments at different scales. With the proposed body-neck-head structure, we could save the redundant convolutional computation and make efficient use of information to perform segment proposal in one shot. We refer this as ***one-shot segment proposal paradigm*** and derive our proposed segment proposal framework in Section 4.

4. Our Approach

In this section, we introduce our approach in detail. First, we overview the proposed architecture (FastMask), to give a concrete idea about our body-neck-head structure. We explain our entire pipeline by illustrating the data flow from input image to object segments. Next we study the different designs of the neck module, including both the non-parametric and parametric necks. Finally, we present a novel head module that enables scale-tolerant segmentation mask decoding by taking advantage of the attention model, which plays the key role in improving performance.

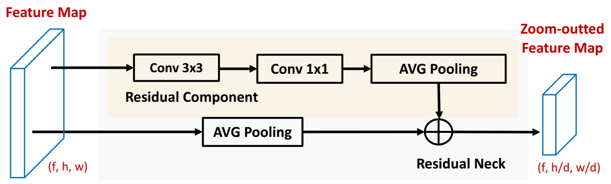

Figure 4. Illustration of the residual neck. We augment the average pooling neck with a learnable residual component.

4.1. Network Architecture

We present our network architecture in Figure 2. Similar to multi-shot methods, the body network extracts semantic feature from the input image. With this base feature map, a shared neck module is applied recursively at it to build feature maps with different scales. This pyramid of feature maps are then input to a 1×1 convolution for reducing dimensionality. Next, we extract dense sliding windows from all these feature maps, and do a batch normalization across all windows to calibrate window features. Note that with a feature map downscaled by a factor m, a sliding window of size (k, k) corresponds to a patch of $(m \times k, m \times k)$ at original image. Finally, a unified head module is used to decode these sliding-window features and produce the output confidence score as well as object mask.

Our approach could be easily adopted to any existing CNN architectures (*e.g.* VGGNet [24], ResNet [12]), by replacing their fully connected layers or some convolutional and pooling layers on the top with the neck and head modules. The reason for removing those top convolutional and pooling layers is to keep feature map in a feasible size, so that a small object could still correspond to a notable region on feature map.

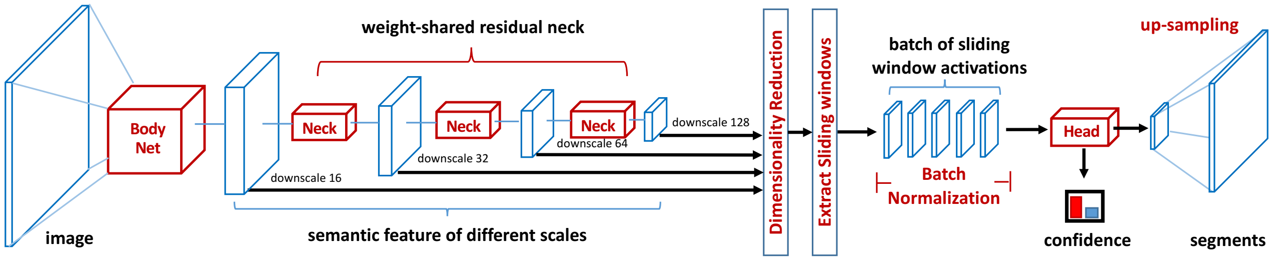

Figure 2. An overview of the proposed FastMask architecture.

4.2. Residual Neck

We consider both non-parametric and parametric methods for encoding feature pyramid. To zoom out feature map, a straightforward choice is non-parametric pooling. Both max pooling and average pooling are widely used components in modern CNN architectures on recognition and detection. In our scenario, we would like to calibrate each feature map for a unified decoding. However, some pooling necks generate sub-optimal empirical results as desired by their natural. In this section, we discuss about several choices of the necks and compare them empirically.

Max pooling neck. Max pooling produces uncalibrated features during encoding. With spatial grids of feature, max pooling takes the max response over each grid for downscaled feature maps. As a result, this process increases the mean of output feature maps. As max pooling is repeatedly applied, the top feature maps would have significantly larger mean than bottom ones.

Average pooling neck. Average pooling smooths out discriminative feature during encoding. Different from max pooling, average pooling maintains the mean of feature maps. Although it helps to keep the means of features in different scales calibrated, it blurs discriminative feature. The lost of discriminative feature makes the head module suffer from distinguishing the object to its background.

Feed-forward neck. To alleviate above side-effects, we propose to learn parametric necks that preserve feature semantics. One naive parametric choice is to learn a feed-forward neck which uses convolutional and pooling layers to zoom out feature maps. However, the feed-forward neck faces the gradient vanishing effect [13] as the number of scales increases. In addition, feature semantics may change substantially since the feature maps on the top go through more convolutional operations than the bottom ones.

Residual neck. Inspired by bottle-neck connection in [12], we design to learn a residual neck as in Figure 4. We augment the non-parametric average pooling with a parametric residual component (using the same structure as in the feed-forward neck, a 3×3 convolutional layer followed by a 1×1 one) to zoom out feature maps, in order to reduce the the smooth effect of average pooling as well as preserve feature semantics.

Method	AR@100	AR^S@100	AR^M@100	AR^L@100
Avg-Pooling	27.9	11.5	36.9	43.9
Max-Pooling	27.8	11.1	36.8	44.2
Feed-Forward	27.1	10.8	35.8	43.4
Residual	**29.3**	**11.7**	**38.3**	**47.2**

Table 1. Comparison on different designs of the neck modules (on COCO benchmark). VGGNet [24] is used as body network for all the necks.

Comparison. To verify the effectiveness of the proposed necks, we empirically evaluate all these designs and report their performance in Table 1. Here we report overall AR@100 and AR@100 for objects in different sizes (details in Section 6). The results confirm that the residual neck component beats all other necks in terms of average recall. Note that we obtain a large margin in average recall for objects in large scale, which are decoded from the top feature maps. This verifies the effectiveness of the residual neck in encoding feature pyramid.

4.3. Attentional Head

Following [20, 21], we use a combination of convolutional layers and fully connected layers to assemble a head module for decoding mask and object confidence. However, in the context of feature pyramid decoding, we found that simply applying this head leads to a suboptimal performance. A likely reason is that, comparing to original DeepMask [20], our feature pyramid is sparser in scales. To be concrete, after the neck module is applied, the feature map is downscaled by a factor of two, which means that the scale gap between two adjacent feature maps is two (while the scale gap in DeepMask is $2^{0.5}$). The sparse feature pyramid raises the possibility that no suitable feature maps exists for an object to decode, and also increases the risk of introducing background noises because an object may not matches well with the size of receptive field (sliding window).

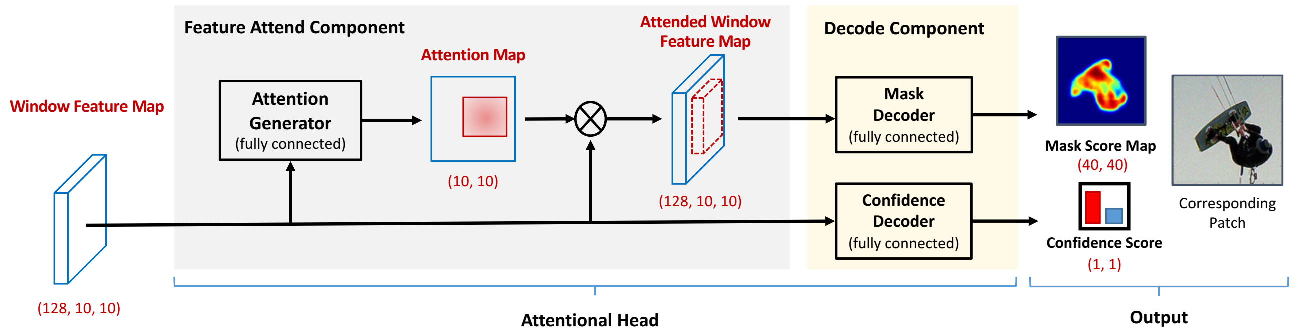

Figure 5. Details of the attentional head. It presents the data flow starting from a feature map to the confidence score and segment mask inside each sliding window. (Notations in round brackets indicate the dimensionality)

Such observations drive us to propose two alternative solutions alleviating such problem. First, we tried to expand our network into two stream, to simply increase the scale density (we defer this part to Section 5). Second, we develop a novel head module that learns to attend salient region during decoding. With visual attention, a decoding head could reduce the noises from the backgrounds in a sliding window and alleviate the mismatch between the size of receptive field and object. Note that such attention also brings the tolerance to shift disturbance (*i.e.* when a object is not well centered), which further improves its robustness.

Figure 5 gives the detailed implementation of our attentional head. Given the feature map of a sliding window as input, we first compute a spatial attention through a fully connected layer. This spatial attention is then applied to window feature map via an element-wise multiplication across channels. Such operation enables the head module to highlight features on the salient region, which indicates the rough location for the target object. Finally, this attended feature map is input into a fully connected layer to decode the segmentation mask of the object.

Comparison. To verify the effectiveness of the proposed attentional head, we do experimental comparisons between FastMask with a standard head and FastMask with an attentional head, as reported in Table 2. From the table we can see that with the tolerance to scale and shift disturbance, the attentional head significantly improves the segment proposal accuracy.

Visualization. To further justify the effectiveness of regional attention in denoising, we visualize two examples (Figure 6) as exemplars. In the top example, a skateboard is the central object and the person riding it is the noisy. As a consequence, generated attention weight regions close to skateboard with higher confidence to highlight the central object. Similarly, the bottom example indicates the same spirit, while in a vice versus manner that person becomes the central object and skateboard is the noise.

Method	AR@10	AR@100	AR@1k
Standard Head	12.7	24.8	33.2
Attentional Head	**15.2**	**29.3**	**38.6**

Table 2. Comparison of different head modules on the COCO benchmark. VGGNet [24] is used as the body network.

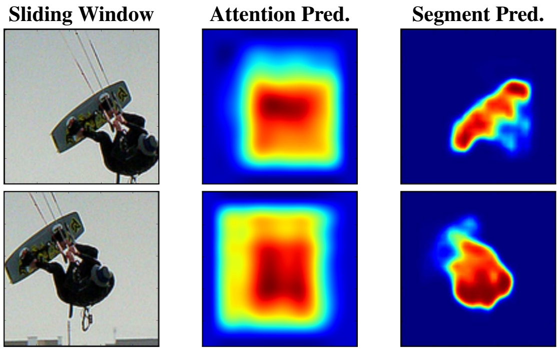

Figure 6. **Attention exemplars.** Attentional head helps to locate important central object feature for mask decoder. (Color towards red represents high score)

5. Implementation Details

In this section we first present an practical technique for obtaining more scales in feature pyramid. Then we give all the details about training, optimization and inference in our framework. We made our code public available on: https://github.com/voidrank/FastMask.

5.1. Two-stream network

As mentioned in Section 4.3, to make the feature pyramid denser, we craft the body network to branches in the middle through applying pooling layers with different strides (*e.g.* 2 and 3 in our implementation) and feed these differently scaled features to the shared neck. This augments the body network to produce feature maps of different sizes, not necessarily limited to a multiple of two. Due to the space limit, we defer more details (*including an illustration figure*) to the supplementary material.

5.2. Training

The key difference between training FastMask and standard DeepMask [20] is that FastMask could be trained by images in varying scales, rather than cropped fixed-scale patches. To enable this training scheme, we introduce our strategies on ground truth assignment, learning objective and optimization details.

Ground truth assignment. During training, we need to determine which sliding window a ground truth object belongs to. For each ground truth object, we assign it to a sliding window if *(i)* it fully contains this object, and *(ii)* the object fits into the scale range of [0.4, 0.8] with regard to the window, and *(iii)* the object is roughly centered in the window (object center in central 10×10 rectangle region of window). Once an object is assigned to a window, we extract the segmentation mask as segmentation ground truth (denoted by s) and use the surrounding bounding as attention ground truth (denoted by a).

Learning objective. The overall objective function of FastMask is a weighted sum of the confidence loss ($\mathcal{L}_{conf}$), segmentation loss ($\mathcal{L}_{seg}$) and region attention loss ($\mathcal{L}_{att}$). Note that c, a, s stand for ground truth label for confidence, region attention and segmentation mask, while $\hat{c}$, $\hat{a}$, $\hat{s}$ stand for corresponding prediction.

$$\mathcal{L}(c,a,s) = \frac{1}{N}\sum_k^N \Big[\mathcal{L}_{conf}(c_k,\hat{c}_k) \\ +\mathbb{1}(c_k)\cdot\big(\mathcal{L}_{seg}(s_k,\hat{s}_k)+\mathcal{L}_{att}(a_k,\hat{a}_k)\big)\Big]. \qquad (1)$$

Here $\mathbb{1}(c_k)$ is an indicator function which returns 1 if c_k is true and 0 otherwise. Equation 1 indicates that we only back-propagate gradients when $c_k = 1$. It is critical to get good performance by computing $\mathcal{L}_{seg}$ and $\mathcal{L}_{att}$ only with positive object samples. We normalize this weighted sum with the total number of sliding windows across mini-batches. For each loss components, we compute the cross entropy function between the prediction and ground truth as following:

$$\mathcal{L}_{conf}(c,\hat{c}) \quad = -\mathcal{E}(s_{i,j},\hat{s}_{i,j}) \tag{2}$$

$$\mathcal{L}_{seg}(s,\hat{s}) \quad = -[\frac{1}{w\cdot h}\sum_{i,j}^{h,w}(\mathcal{E}(s_{i,j},\hat{s}_{i,j}))] \tag{3}$$

$$\mathcal{L}_{att}(a,\hat{a}) \quad = -[\frac{1}{w\cdot h}\sum_{i,j}^{h,w}(\mathcal{E}(a_{i,j},\hat{a}_{i,j}))]. \tag{4}$$

For $\mathcal{L}_{seg}$ and $\mathcal{L}_{att}$, we normalize spatially across the window to balance the gradients between three loss components. $\mathcal{E}(y,\hat{y})$ is a standard binary cross entropy function with sigmoid activation function (denoted by $\sigma(y)$), in the following form:

$$\mathcal{E}(y,\hat{y}) \quad = y\cdot log(\sigma(\hat{y})) + (1-y)\cdot log(1-\sigma(\hat{y})). \tag{5}$$

Optimization. We optimize the objective by standard stochastic gradient descent (SGD) with batch size equals 1, momentum equals 0.9 and weight decay equals 0.00005. We train our network for approximately 15 epochs and choose the best models through a different subset of COCO validation set. Following the practice of [22, 8], we balance positive and negative samples by a certain ratio (*e.g.* roughly 1:1 in our case) after collecting all sliding-windows in training. In our practice, due to the limitation of GPU Memory, we train our two-stream network with totally 7-scale feature maps, by taking zooming out 4 times on the stream with *stride* = 2, and 3 times on the stream with *stride* = 3.

5.3. Inference

During inference, we process an image in one shot and extract windows at multi-scale feature maps as same as the training stage. First the confidence score of each window is predicted, and then only the top-*k* confident windows are selected for object segment decoding. In addition, as the residual neck is weight shared, we could add or reduce the number of neck components during inference. This enables us to make easy trade-off between the effectiveness and efficiency, via adjusting the number of neck components. Therefore, although trained by 7 scales, the two-stream network could still be equipped by more than 7 neck modules to generate a denser feature pyramid. In the following experiments, unless specified, we use the two-stream network with 8 scales in the inference stage.

6. Experiments

We analyze and evaluate our network on MS COCO benchmark, which contains 80k training images and a total of nearly 500k instance annotations. Following the experimental setting of [20, 21, 5], we report our result on the first 5k COCO validation images. We use another non-overlapped 5k images for validation.

Metrics. We measure the mask accuracy by Intersection over Union(IoU) between predicted mask and ground truth annotation. As average recall correlates well with object proposal quality [15], we summarize Average Recall (AR) between IoU 0.5 and 0.95 for a fixed number N of proposals, denoted as "AR@N" in order to measure the performance of algorithms. (We use N equals to 10, 100 and 1000)

Scales. As COCO dataset contains objects in a wide range of scales, a more fine-grained evaluation tends to measures metrics with regards to object scales. Practically, objects are divided into three groups according to their pixel areas a: small ($a < 32^2$), medium ($32^2 < a < 96^2$), large ($a > 96^2$). In our experiments, we denote the metrics for different scales by adding superscripts S, M, L respectfully.

Methods. By default, we compare our method with recent state-of-the-arts for segment proposal, including *DeepMask* [20], *SharpMask* [21] and *InstanceFCN* [5]. Note that we also provide results from a revised DeepMask architecture from [21], denoted as *DeepMask**. Different from original DeepMask, it is implemented based on 39-layer residual net with a revised head component. These methods not only achieve good Average Recall but also provide strong efficiency during inference.

Our network is general and could be plug-in to different body networks. In our experiments, we adopt 39-layer Residual Net [12] for best accuracy as well as fair comparison and PvaNet [14] for best efficiency.

6.1. Comparison with State-of-the-art Methods

Table 3 compares the performance of our FastMask to other state-of-the-art methods. We report results on both bounding box and segment proposals (by deriving a tight bounding box from a mask proposal). Here we do not include the SharpMaskZoom^2 result because they use images with extra scales (2 ^1/2 larger) to obtain superior performance.

We compare our two-stream FastMask with all those image pyramid based methods since our one-stream network does not contain the same density in its feature pyramid. To address the influence of feature scale density to performance as well as efficiency, we conduct separate controlled experiments in Section 6.2.

Method	Body Net	Box Proposals			Segmentation Proposals		
		AR@10	AR@100	AR@1k	AR@10	AR@100	AR@1k
MCG	-	10.1	24.6	39.8	7.7	18.6	29.9
DeepMask [20]	VGG	15.3	31.3	44.6	12.6	24.5	33.1
DeepMaskZoom [20]	VGG	15.0	32.6	48.2	12.7	26.1	36.6
DeepMask* [21]	Res39	18.0	34.8	47.0	14.1	25.8	33.6
SharpMask [21]	Res39	19.2	36.2	48.3	15.4	27.8	36.0
SharpMaskZoom [21]	Res39	19.2	39.0	53.2	15.6	30.4	**40.1**
InstanceFCN [5]	VGG	-	-	-	16.6	**31.7**	39.2
FastMask+two streams	Res39	**22.6**	**43.1**	**57.4**	**16.9**	**31.3**	**40.6**
FastMask+two streams	Pva	**24.1**	**43.6**	**56.2**	**17.5**	30.7	39.0

Table 3. Object segment proposal results on COCO validation set for box and segmentation proposals. Note that we also report the body network for each corresponding method.

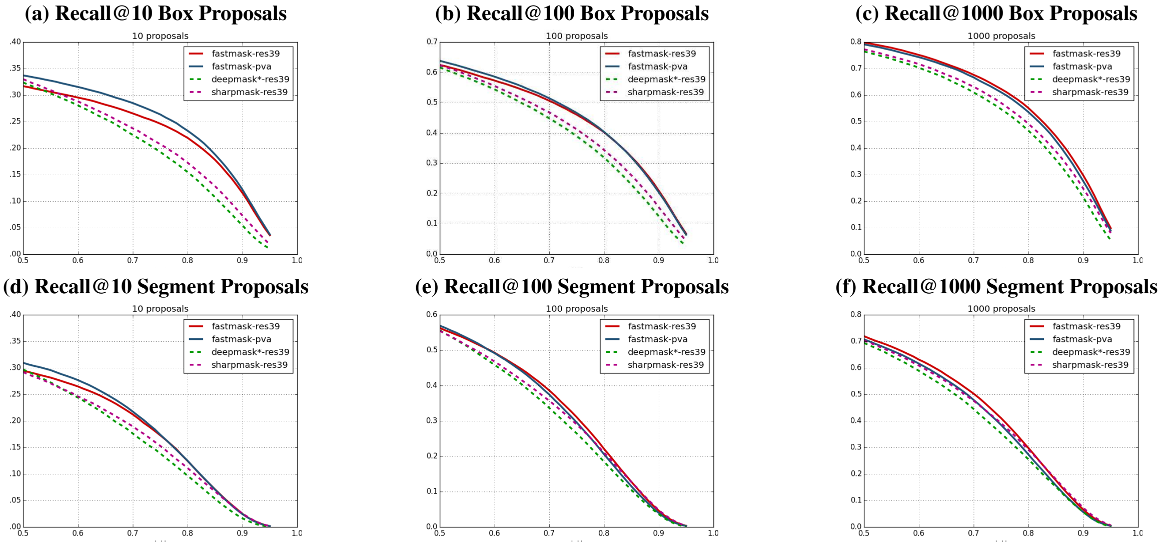
(a) Recall@10 Box Proposals
(b) Recall@100 Box Proposals
(c) Recall@1000 Box Proposals
(d) Recall@10 Segment Proposals
(e) Recall@100 Segment Proposals
(f) Recall@1000 Segment Proposals

Figure 7. Proposal recall curve. (a-c) show detailed box proposal recall while (d-e) show detailed segment proposal recall.

Quantitative evaluation. According to Table 3, we outperform all state-of-the-art methods in bounding-box proposal by a large margin and obtain very competitive results with segmentation proposals (outperform all methods on AR@10 and AR@1k, and show competitive performance on AR@100). It is worth noting that our two-stream network significantly improves the box proposal quality comparing to all other methods, which provides a guidance on its potential for bbox-based object detection. Our two-stream FastMask model with 39-layers Resnet achieves approximately 18%, 11%, 8% relative improvement on AR@10, AR@100, AR@1k metrics respectively, over previous best SharpMask model. In order to give a better picture of our proposal quality, we plot the recall versus IoU threshold for different of segmentation proposals in COCO dataset as Figure 7. There is a clear gap in the plot, which indicates that FastMask produce better mask quality overall.

While obtaining superior performance, our method also yields better efficiency than all image pyramid based approaches. We did some controlled experiments and report the speed/performance in Section 6.2.

Qualitative visualization. We visualize some results in Figure 8 showing exemplars on which our method improves over baselines. Generally, we observe that our method is more robust to scale variance and invariant to noisy background. Not like SharpMask, FastMask does not perform any mask refinement at all. It is possible to further boost mask quality by leveraging mask refinement.

6.2. Efficiency Study

In this section, we evaluate two threads to support our argument that FastMask outperforms image pyramid methods on both efficiency and performance. On the first thread, we provide experimental results on DeepMask and SharpMask, with restriction on the scale density of their image pyramids. We construct a fair environment that both these methods and our method take equivalently many scales and evaluate both inference speed and performance. On the other thread, we provide the performance and speed of state-of-the-art methods and compare our best model as well as fastest model to them.

Trade-off scale density with speed. We conduct a fair

Method	Scales	AR@10	AR@100	Speed
DeepMask* [21]	8	14.3	27.3	0.45s
DeepMask* [21]	4	11.3	22.2	0.24s
FastMask	8	16.9	31.3	0.26s
FastMask	4	13.3	26.6	0.14s

Table 4. Trade-off between scale density and performance.

study to analyze the trade-off by decreasing scale density. In the DeepMaskZoom* and SharpMaskZoom, they inference on images scaled from 2ˆ[-2.5, -2.0, -1.5, -1.0, -0.5, 0, 0.5, 1] to obtain superior performance on a diverse range of object segments. This is similar to our two-stream network, where we input a image up-sampled by two. To improve the inference efficiency, we made a trade-off in scale density by reducing our network to one-stream without re-training, which is identical to reduce scale density for DeepMaskZoom* and SharpMaskZoom to 2ˆ[-2.5, -1.5, -0.5, 0.5].

Figure 4 illustrates the performance degradation and efficiency increase with scale density trade-off. We measure only AR@10 and AR@100 as a sparse scale density leads to less total proposal number. These controlled experiments are tested using NVIDIA Titan X GPU. We do multiple runs and average their time to obtain an estimation of runtime speed. Our method achieves to preserve the best performance while increase the inference speed by almost 2×. Note that retraining a network with reduced scale density can boost up performance.

Speed evaluation. We evaluate the inference speed of all state-of-the-art methods. Two variant of our models, our most effective model (FastMask-acc) and most efficient model (FastMask-fast), are reported. Our most effective model takes a two-stream structure with 39-layer ResNet; Our fastest model takes a one-stream structure with PvaNet [14], which is light-weight and fast.

Method	Body Net	AR@10	AR@100	AR@1k	Speed
DeepMask [20]	VGG	12.6	24.5	33.1	1.60s
DeepMask* [21]	Res39	14.1	25.8	33.6	0.46s
SharpMask [21]	Res39	15.4	27.8	36.0	0.76s
SharpMaskZoom [21]	Res39	15.6	30.4	**40.1**	∼1.5s
InstanceFCN [5]	VGG	16.6	**31.7**	39.2	1.50s
FastMask-acc	Res39	**16.9**	**31.3**	**40.6**	0.26s
FastMask-fast	Pva	**17.2**	29.4	36.4	0.07s

Table 5. Speed Study with state-of-the-art methods.

Figure 5 compare our best and fastest model with other networks. Our best model produces superior proposal performance while preserving good efficiency. With slight trade-off in performance, our fastest model obtains almost real-time efficiency (∼13 FPS by NVIDIA Titan X Maxwell).

7. Conclusion

In this paper we present an innovative framework, *i.e.* FastMask, for efficient segment-based object proposal. Instead of building pyramid of input image, FastMask learns to encode feature pyramid by a neck module, and performs one-shot training and inference. Along with with process, a scale-tolerant head module is proposed to highlight the foreground object from its background noises, havesting a significant better segmentation accuracy. On MS COCO benchmark, FastMask outperforms all state-of-the-art segment proposal methods in average recall while keeping several times faster. More impressively, with a slight trade-off in accuracy, FastMast can segment objects in nearly real time (∼13 fps) with images at 800×600 resolution. As an effective and efficient segment proposal method, FastMask is believed to have great potentials in other tasks.

8. Acknowledgement

H.X. Hu and F. Sha are partially supported by NSF IIS-1065243, 1451412, 1513966, 1208500, CCF-1139148, a Google Research Award, an Alfred. P. Sloan Research Fellowship and ARO# W911NF-15-1-0484

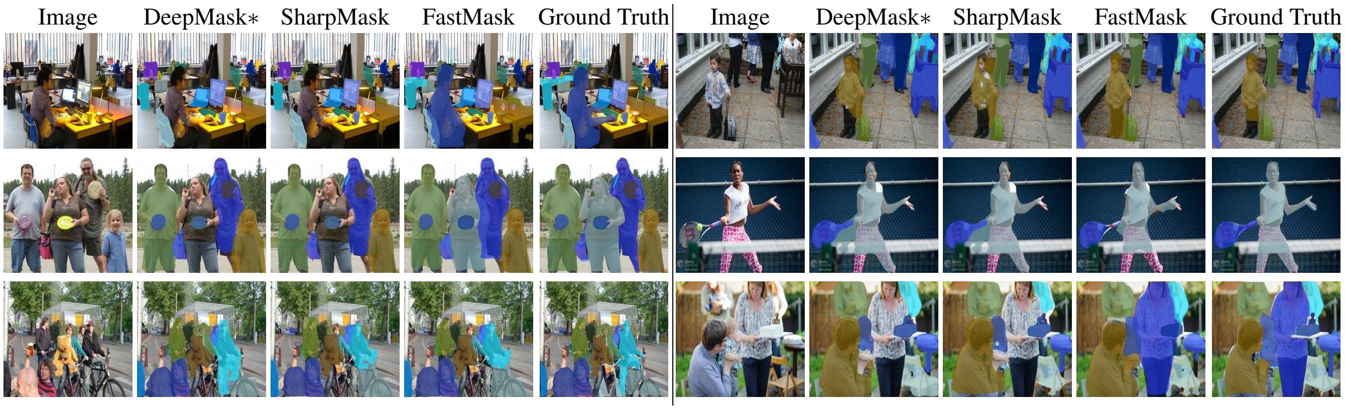

Figure 8. Visualization of the object candidate segmentation results on sampled MS COCO images. We compare our FastMask with DeepMask∗ [21] and SharpMask [21]. We also show the origin images and the ground-truth annotations for reference.

References

[1] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *CVPR*, 2014. 1, 2

[2] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In *CVPR*, 2016. 1, 2

[3] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *ICLR*, 2015. 2

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

[5] J. Dai, K. He, Y. Li, S. Ren, and J. Sun. Instance-sensitive fully convolutional networks. In *ECCV*, 2016. 1, 2, 6, 7, 8

[6] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. In *CVPR*, 2016. 1, 2

[7] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov. Scalable object detection using deep neural networks. In *CVPR*, 2014. 2

[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 1, 6

[9] R. Girshick, F. Iandola, T. Darrell, and J. Malik. Deformable part models are convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 437–446, 2015. 2

[10] R. Gokberk Cinbis, J. Verbeek, and C. Schmid. Segmentation driven object detection with fisher vectors. In *CVPR*, 2013. 1

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

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3, 4, 6

[13] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *ECCV*, 2016. 4

[14] S. Hong, B. Roh, K.-H. Kim, Y. Cheon, and M. Park. Pvanet: Lightweight deep neural networks for real-time object detection. *arXiv preprint arXiv:1611.08588*, 2016. 6, 8

[15] J. Hosang, R. Benenson, P. Dollár, and B. Schiele. What makes for effective detection proposals? *IEEE T-PAMI*, 2016. 6

[16] P. Krähenbühl and V. Koltun. Geodesic object proposals. In *ECCV*, 2014. 1, 2

[17] W. Kuo, B. Hariharan, and J. Malik. Deepbox: Learning objectness with convolutional networks. In *ICCV*, 2015. 2

[18] 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. 2

[19] C. Liu, J. Mao, F. Sha, and A. Yuille. Attention correctness in neural image captioning. *arXiv preprint arXiv:1605.09553*, 2016. 2

[20] P. O. Pinheiro, R. Collobert, and P. Dollar. Learning to segment object candidates. In *NIPS*, 2015. 1, 2, 4, 5, 6, 7, 8

[21] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár. Learning to refine object segments. In *ECCV*, 2016. 1, 2, 4, 6, 7, 8

[22] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 2, 6

[23] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. *IEEE T-PAMI*, 2016. 2

[24] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 3, 4, 5

[25] K. E. Van de Sande, J. R. Uijlings, T. Gevers, and A. W. Smeulders. Segmentation as selective search for object recognition. In *ICCV*. IEEE, 2011. 1, 2

[26] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2015. 2

[27] Z. Yang, X. He, J. Gao, L. Deng, and A. Smola. Stacked attention networks for image question answering. 2016. 2

[28] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016. 2

[29] Z. Zhang, J. Warrell, and P. H. Torr. Proposal generation for object detection using cascaded ranking svms. In *CVPR*, 2011. 1, 2

[30] Y. Zhu, O. Groth, M. Bernstein, and L. Fei-Fei. Visual7w: Grounded question answering in images. 2016. 2

[31] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014. 2