DynaMask: Dynamic Mask Selection for Instance Segmentation

Ruihuang Li*, Chenhang He*, Shuai Li, Yabin Zhang, Lei Zhang†
The Hong Kong Polytechnic University
{csrhli, csche, cslzhang}@comp.polyu.edu.hk

Abstract

The representative instance segmentation methods mostly segment different object instances with a mask of the fixed resolution, e.g., 28 × 28 grid. However, a low-resolution mask loses rich details, while a high-resolution mask incurs quadratic computation overhead. It is a challenging task to predict the optimal binary mask for each instance. In this paper, we propose to dynamically select suitable masks for different object proposals. First, a dual-level Feature Pyramid Network (FPN) with adaptive feature aggregation is developed to gradually increase the mask grid resolution, ensuring high-quality segmentation of objects. Specifically, an efficient region-level top-down path (r-FPN) is introduced to incorporate complementary contextual and detailed information from different stages of image-level FPN (i-FPN). Then, to alleviate the increase of computation and memory costs caused by using large masks, we develop a Mask Switch Module (MSM) with negligible computational cost to select the most suitable mask resolution for each instance, achieving high efficiency while maintaining high segmentation accuracy. Without bells and whistles, the proposed method, namely DynaMask, brings consistent and noticeable performance improvements over other state-of-the-arts at a moderate computation overhead. The source code: https://github.com/lslrh/DynaMask.

1. Introduction

Instance segmentation (IS) is an important computer vision task, aiming at simultaneously predicting the class label and the binary mask for each instance of interest in an image. It works as the cornerstone of many downstream vision applications, such as autonomous driving, video surveillance, and robotics, to name a few. Recent years have witnessed the significant advances of deep convolutional neural networks (CNNs) based IS methods with a rich amount of training data as the rocket fuel [1, 10, 17, 25, 26]. Existing IS methods can be roughly divided into two major categories: two-stage [6, 10, 17] and single-stage methods [1, 2, 26]. The former first detect a sparse set of proposals and then performs mask predictions based on them, while the latter directly predict classification scores and masks based on the pre-defined anchors. Generally speaking, two-stage methods could produce higher segmentation accuracy but cost more computational resources than single-stage methods.

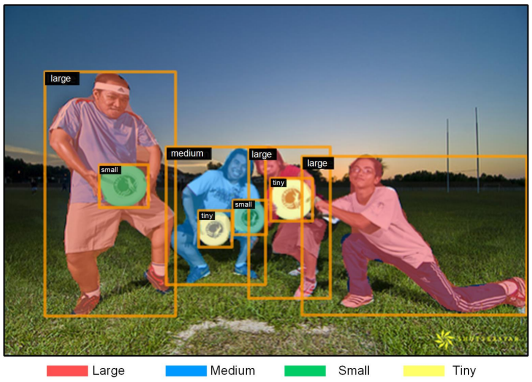

Figure 1. Dynamic mask selection results. Some "hard" samples with irregular shapes like "person" are assigned larger masks, while the "easy" ones like "frisbee" are assigned smaller ones.

Among the many recently developed IS methods, the proposal-based two-stage methods [6, 10, 17], which follow a detection-and-segmentation paradigm, are still the top performers. These methods need to predict a binary grid mask of uniform resolution for all proposals, *e.g.*, 28 × 28, and then upsample it to the original image size. For instance, Mask R-CNN [10] first generates a group of proposals with an object detector and then performs per pixel foreground/background segmentation on the Regions of Interest (RoIs) [24]. Despite achieving promising performance, the low-resolution mask of Mask R-CNN is insufficient to capture more detailed information, resulting in unsatisfactory predictions, especially over object boundaries. An intuitive solution to improve the segmentation quality is to adopt a larger mask. Nevertheless, high-resolution masks often generate excessive predictions on the smooth regions, resulting in high computational complexity.

It is difficult to segment different objects in an image

*denotes the equal contribution, †denotes the corresponding author. This work is supported by the Hong Kong RGC RIF grant (R5001-18).

with masks of the same resolution. Objects with irregular shapes and complicated boundaries demand more fine-grained masks to predict, referred to as "hard" samples, such as the "person" in Fig. 1. In comparison, the "easy" samples with regular shapes and fewer details can be efficiently segmented using coarser masks, like the "frisbee" in Fig. 1. Inspired by the above observations, we propose to adaptively adjust the mask size for each instance for better IS performance. Specifically, instead of using a uniform resolution for all instances, we assign high-resolution masks to "hard" objects and low-resolution masks to "easy" objects. In this way, the redundant computation for "easy" samples can be reduced while the accuracy of "hard" samples can be improved, achieving a balance between accuracy and speed. As shown in Tab. 1, however, directly predicting a high-resolution mask by Mask R-CNN [10] unexpectedly degrades the mask average precision (AP). This attributes to two main reasons. First, the RoI features of larger objects are extracted from higher pyramid levels, which are very coarse due to the downsampling operations [20]. Thus simply increasing the mask size of these RoIs will not bring extra useful information. Second, the mask head of Mask R-CNN is oversimplified, so it cannot make more precise predictions as the mask grid size increases.

To overcome the above mentioned problems, we propose a dual-level FPN framework to gradually enlarge the mask grid. Specifically, in addition to traditional image-level FPN (i-FPN), a region-level FPN (r-FPN) is designed to achieve coarse-to-fine mask prediction. Wherein we construct information flows between i-FPN and r-FPN at different pyramid levels, aiming to incorporate complementary contextual and detailed information from multiple feature levels for high-quality segmentation. With the dual-level FPN, we present a data-dependent Mask Switch Module (MSM) with negligible computational cost to adaptively select masks for each instance. The overall approach, namely DynaMask, is evaluated on benchmark instance segmentation datasets to demonstrate its superiority over state-of-the-arts. The major contributions of this work are summarized as follows:

- A dynamic mask selection method (DynaMask) is proposed to adaptively assign appropriate masks to different instances. Specifically, it assigns low-resolution masks to "easy" samples for efficiency while assigning high-resolution masks to "hard" samples for accuracy.
- A dual-level FPN framework is developed for IS. We construct direct information flows from i-FPN to r-FPN at multiple levels, facilitating complementary information aggregation from multiple pyramid levels.
- Extensive experiments demonstrate that DynaMask achieves a good trade-off between IS accuracy and efficiency, outperforming many state-of-the-art two-stage IS methods at a moderate computation overhead.

Method	Resolution	AP	FLOPs
Mask R-CNN [10]	14×14	32.9	0.2G
	28×28	34.7	0.5G
	56×56	33.8	2.0G
	112×112	32.5	8.0G
DynaMask	14×14	32.9	0.2G
	28×28	36.1	0.6G
	56×56	37.1	1.0G
	112×112	37.6	1.4G

Table 1. Mask AP and FLOPs with different mask resolutions. For Mask R-CNN, directly increasing the mask resolution will decrease the mask AP. While for our DynaMask, higher mask resolution results in better performance.

2. Related Work

Instance Segmentation. To date, most of the top-performing IS methods still follow the Mask R-CNN meta-architecture [10]. These proposal-based approaches typically employ an object detector to localize each instance in bounding boxes. Then the instance-wise features are cropped and extracted from FPN features based on the detected bounding boxes by using RoI pooling/align [10, 24]. Finally, a compact segmentation head is deployed to obtain the desired object masks. Mask Scoring R-CNN [14] aligns the mask quality and score by using a branch to explicitly learn the quality of predicted masks. BMask R-CNN [6] leverages boundary details to improve mask localization ability. DCT-Mask [25] encodes high-resolution binary masks into compact vectors through the discrete cosine transform (DCT). PANet [22] constructs two feature pyramids for improving mask prediction.

Some works [3, 17, 30] have been proposed to improve the mask quality by coarse-to-fine refinement. HTC [3] performs cascaded refinement on both detection and segmentation tasks and explores the reciprocal relationships between them. PointRend [17] and RefineMask [30] extract fine-grained features in a multi-stage manner. The former performs point-based predictions over the blurred areas, while the latter refines the entire RoI feature. Despite the promising segmentation results, multiple refinement stages inevitably increase the inference time and memory burden. In this paper, we dynamically select the suitable mask for each instance so that "easy" samples are assigned small masks (with fewer refinement stages) for efficiency, and "hard" samples are assigned large masks (with more refinement stages) for accuracy, achieving a balance between efficiency and accuracy.

Dynamic Networks. Dynamic networks, which aim to adaptively adjust the network architecture according to the input feature, have been widely studied in recent years. Dynamic model compression methods either drop blocks [12, 23, 27] or prune channels [19, 29] for speeding

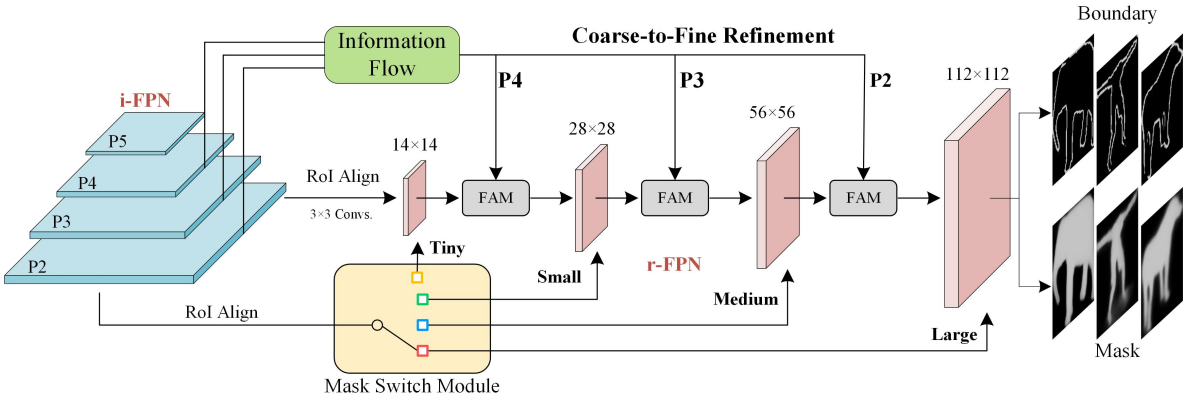

Figure 2. The overall architecture of DynaMask. We propose a dual-level FPN framework to gradually increase the mask size and achieve higher-quality IS. The information flow is constructed between each level of image-level FPN (i-FPN) and region-level FPN (r-FPN), so that region-wise feature hierarchies $\{L_{tiny}, L_{small}, L_{medium}, L_{large}\}$ are constantly enhanced by incorporating complementary information from $\{P2, P3, P4\}$ of i-FPN, resulting in coarse-to-fine mask predictions. To reduce the computation and memory cost, a Mask Switch Module (MSM) is developed to predict the mask resolution for each instance with budgeted resource consumption. Specifically, MSM outputs four different switching states, corresponding to four different mask resolutions, *i.e.*, $[14 \times 14, 28 \times 28, 56 \times 56, 112 \times 112]$.

up the inference. For instance, SkipNet [27] skips convolutional blocks through a reinforcement learning-based gating network. Huang *et al*. propose a multi-scale dense network with multiple classifiers for allocating different computations for "easy" and "hard" samples. Li *et al*. [18] adopt an end-to-end dynamic routing framework to alleviate the scale variance among inputs. DRNet [32] attempts to reduce the redundancy on the input resolution of modern CNNs.

However, employing dynamic masks at different resolutions for segmenting different instances has rarely been explored in the field of IS. Conventional methods [6, 10, 17] typically predict a fixed-size mask irrespective of the object type. This is sufficient for "easy" samples but produces over-smoothing predictions for "hard" samples over the fine-level details. In order to improve the segmentation performance without introducing many additional computation burdens, we devise a dynamic mask selection framework to adaptively allocate suitable masks to different objects according to their segmentation difficulties.

3. Dynamic Mask Selection

The framework of DynaMask is illustrated in Fig. 2. A dual-level FPN architecture is first proposed for improving IS quality, and then a Mask Switch Module (MSM) is developed to dynamically allocate appropriate masks to each instance, so that the resource consumption can be reduced while maintaining superior IS performance. Our DynaMask produces high-quality segmentation with a moderate computation overhead.

3.1. Dual-Level FPN

Original image-level FPN (i-FPN) [20] introduces a top-down path to propagate contextual semantic information from higher layers to lower ones. Actually, the lower-level features contain more fine-grained details than higher-level ones, which are beneficial for high-quality segmentation, especially on boundary regions, but these information is not fully explored in Mask R-CNN [10]. In this work, we propose a region-level FPN (r-FPN) to integrate more detailed information from lower layers of i-FPN into region-wise feature hierarchies. The information flows from each level of i-FPN to r-FPN are shown in Fig. 2.

Region-Level FPN. We follow the original i-FPN to define the layers producing feature maps of the same resolution as one network stage corresponding to one feature level. We use $\{P_2, P_3, P_4, P_5\}$ to denote different feature levels of i-FPN. The r-FPN starts from an RoI-Aligned region-wise feature, and it is gradually enhanced by fusing the complementary information from $\{P_2, P_3, P_4\}$ of i-FPN, resulting in top-down region-based feature hierarchies, denoted by $\{L_{tiny}, L_{small}, L_{medium}, L_{large}\}$. From L_{tiny} to L_{large}, the spatial resolution is progressively increased by a factor of two. We design a Feature Aggregation Module (FAM) to integrate the r-FPN feature L_r and the i-FPN feature P_i.

Feature Aggregation Module (FAM). There exists spatial misalignments between L_r and P_i due to the upsampling and RoI pooling [24] operations, which may degrade the segmentation performance on boundary areas. To overcome this limitation, we propose the FAM to adaptively aggregate multi-scale features. As show in Fig. 3, FAM contains two deformable convolutions [8, 13] which play different roles. The first one (*Deform_Conv1*) adjusts the positions of L_r, making it better aligned to P_i. Here we first concatenate L_r with P_i, and then pass the concatenated features through a 3×3 *Conv* to obtain the offset map, denoted by Δ_o. Finally, L_r is aligned to P_i with the learned offset Δ_o. The

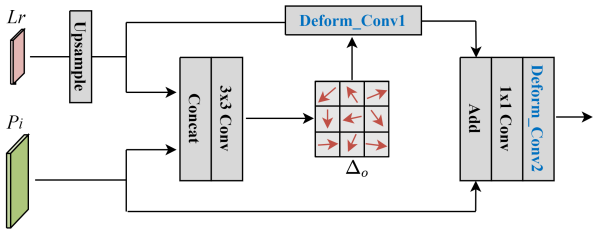

Figure 3. Feature Alignment Moudule (FAM). Deform_Conv1 adjusts the spatial positions of upsampled r-FPN feature L_r and the cropped i-FPN feature P_i. Deform_Conv2 attends to the salient parts of objects.

second one (*Deform_Conv2*) works like an attention mechanism, which attends to the salient parts of objects. The proposed FAM is plugged into different stages of r-FPN and it plays a key role in improving the mask prediction.

3.2. Mask Switch Module (MSM)

The proposed dual-level FPN framework brings a significant performance improvement but at the price of expensive computation and memory burdens. Inspired by the fact that different instances require different mask grids to achieve accurate segmentation, we propose a novel method to adaptively adjust the mask grid resolution for different instances. Specifically, an MSM is developed to perform mask resolution prediction under a budgeted computation consumption, achieving a good trade-off between segmentation accuracy and efficiency.

Optimal Mask Assignment. The MSM module is actually a lightweight classifier, denoted by $f_{MSM}(\cdot)$, which is illustrated in Fig. 4. It contains a channel-wise attention module [11], followed by a few convolutional layers and fully-connected layers. This classifier aims to find the optimal mask resolution from a collection of K candidates, *i.e.*, $[r^1, r^2, \cdots, r^K]$, so that the instance could be accurately segmented with the minimal resource cost. Specifically, MSM takes the cropped region-wise RoI feature as input and outputs a probability vector $P = [p^1, \cdots, p^K]$ by taking a softmax operation. Each element of this vector represents the probability that the corresponding candidate resolution is selected:

$$p^k = \frac{\exp(f^k_{MSM}(x))}{\sum_{k'} \exp(f^{k'}_{MSM}(x))}, \quad k \in \{1, \cdots, K\}, \tag{1}$$

where x is the input RoI feature fed to MSM. The candidate resolution of the largest probability is chosen as the switching state, which decides the resolution of mask grid to segment an object.

Reparameterization with Gumbel-Softmax. The soft output P of MSM should be transformed into a one-hot prediction, denoted by $Y = [y^1, \cdots, y^K]$, $y^k \in \{0, 1\}$. This process can be done by discrete sampling, which however is non-differentiable and does not support end-to-end training. To allow the gradient back-propagation for updating MSM, we introduce a reparameterization method [15], called the Gumbel-Softmax. Given a categorical distribution with class probabilities $P = [p^1, \cdots, p^K]$, we can draw a group of masks through the rule $y = \text{one\_hot}\Big(argmax_k(logp^k + g^k)\Big)$, where $\{g^k\}^K_{i=k}$ are i.i.d. samples drawn from the $Gumbel(0, 1)$ distribution, which is defined by $g = -log\big(-log(u)\big)$ with $u \sim Uniform(0, 1)$. Then we use the Gumbel-softmax function as a continuous, differentiable approximation to the original $\mathrm{softmax}$ function:

$$y^k = \frac{\exp\big((logp^k + g^k)/\tau\big)}{\sum_{k'} \exp\big((logp^{k'} + g^{k'})/\tau\big)}, \tag{2}$$

where τ denotes a temperature parameter. When τ approches 0, the Gumbel-softmax is close to one-hot.

3.3. Objective Function

The proposed framework gradually enlarges the mask resolution by a factor of two to improve the segmentation performance. On the one hand, FAM adaptively aggregates complementary information from multiple stages of i-FPN and r-FPN to enhance the feature hierarchies. On the other hand, MSM dynamically allocates masks of different resolutions for different instances in an image, reducing resource costs without sacrificing accuracy. In this subsection, we elaborate on the loss function used to train the mask head.

Mask Loss. Given a positive instance x_i, we first predict its mask switching state $Y = [y^1, \cdots, y^K]$ by MSM, and obtain a group of mask prediction maps at K different resolutions $\{\hat{m}^1_i, \cdots, \hat{m}^K_i\}$ by passing this instance through different stages of r-FPN. We define the mask loss function as follows:

$$\mathcal{L}_{mask} = \sum_{i=1}^{N} \sum_{k=1}^{K} y^k \ell(\hat{m}^k_i, m_i), \tag{3}$$

where $\hat{m}^k_i$ denotes the k-th mask prediction of x_i, and m_i represents its corresponding ground truth mask grid. y^k is the indicator for whether the k-th mask resolution is selected as the output resolution. ℓ is defined as the binary cross-entropy loss in this paper.

Edge Loss. In Eq. 3, we assume that the mask producing smaller loss should have higher quality, so that the most accurate mask could be selected by minimizing the mask loss. However, our empirical results show that the mask loss produced on different masks is very close, making it difficult to distinguish the mask quality. In comparison, as shown in Fig. 5, the edge loss produced by masks of different resolutions varies greatly, which could better reveal the mask quality. Given the output of MSM $Y = [y^1, \cdots, y^K]$

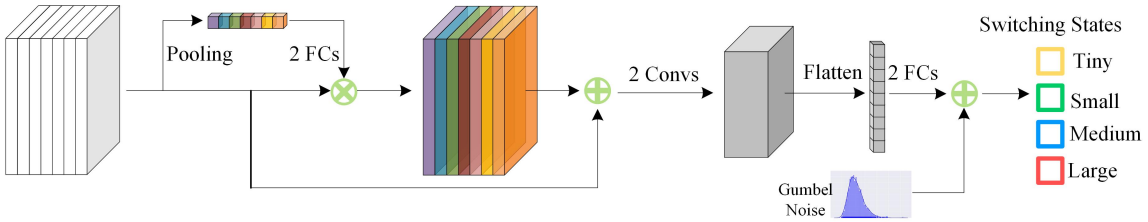

Figure 4. The Mask Switch Module (MSM), which is a CNN-based classifier and consists of a SE-block [11], two convolutional layers and two fully-connected layers.

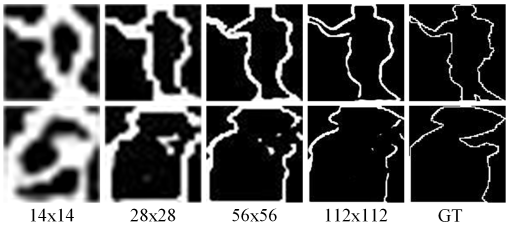

Figure 5. The predicted object edges using the masks of different resolutions. The edge loss computed on boundary regions could reflect the quality of mask.

and the edge maps of different resolutions, denoted by $\{\hat{e}_i^1, \cdots, \hat{e}_i^K\}$, the edge loss is defined as follows:

$$\mathcal{L}_{edge} = \sum_{i=1}^{N}\sum_{k=1}^{K} y^k \ell(\hat{e}_i^k, e_i), \tag{4}$$

where e_i denotes the ground-truth edge, which is generated by first applying the Laplacian operator on the ground-truth mask m_i to obtain a soft edge map, and then converting it into a binary edge map by thresholding [6]. Fig. 5 visualizes the object edges predicted by using masks of different resolutions. As can be seen, the edge loss could better reveal the quality of masks, *i.e.*, higher-resolution masks produce edges closer to GT (smaller loss), while lower-resolution masks produce edges more different from GT (larger loss).

Budget Constraint. By optimizing the edge loss in Eq. 4, the model tends to converge to a sub-optimal solution that all instances are segmented with the largest mask, *i.e.*, 112×112, which incorporates more detailed information and hence has the minimal prediction loss. Actually, not all samples require the largest mask for segmentation. The redundant computations of segmenting the "easy" samples can be saved for efficiency. In order to reduce the computational cost and avoid the above mentioned sub-optimal solution, we propose to train the MSM with a budget constraint. Specifically, let $\mathcal{C}$ denote the computational cost (*e.g.*, FLOPs) corresponding to the selected mask resolution. We add a penalty to the model when the expectation of FLOPs computed on current batch data, denoted by $\mathbb{E}(\mathcal{C})$, exceeds the target FLOPs, denoted by $\mathcal{C}_t$. The budget constraint is defined as follows:

$$\mathcal{L}_{budget} = \max(\frac{\mathbb{E}(\mathcal{C})}{\mathcal{C}_t} - 1, 0). \tag{5}$$

We further introduce an information entropy loss to balance the resolution predictions of MSM. Given a group of output probability vectors $P_1, P_2, \cdots, P_N$, where N is the number of instances of the current batch, the frequency of the k-th resolution is calculated by: $f^k = \frac{1}{N}\sum_i p_i^k$. Then the information entropy loss is defined as follows:

$$\mathcal{L}_{entropy} = \frac{1}{K}\sum_k f^k log f^k. \tag{6}$$

The above entropy loss tends to push each element f^k to be $\frac{1}{K}$ so that MSM could select different resolutions with similar probabilities.

Finally, the total objective function of the mask branch is obtained as follows:

$$\mathcal{L}_{total} = \mathcal{L}_{mask} + \lambda_1 \mathcal{L}_{edge} + \lambda_2 \mathcal{L}_{reg}, \tag{7}$$

where λ_1 and λ_2 are the trade-off hyper-parameters. $\mathcal{L}_{reg}$ denotes the regularization term which is obtained by combining the budget constraint in Eq. 5 and the information entropy loss in Eq. 6, *i.e.*, $\mathcal{L}_{reg} = \mathcal{L}_{budget} + \mathcal{L}_{entropy}$.

4. Experiments

We perform extensive experiments on the benchmark instance segmentation datasets: COCO [21] and Cityscapes [7]. COCO [21] contains about 115k images (`train2017`) with instance-level annotations of 80 categories for training. We use the `val2017` set (around 5k images) for the ablation study and the `test-dev2017` set (about 20k images) for comparison with other methods. The Cityscapes dataset [7] contains 2975, 500, and 1525 images collected from 8 categories of urban scenes for training, validation, and testing, respectively. On both COCO and Cityscapes datasets, we use the standard mask AP as the evaluation metric, which computes the average precision over varying IoU thresholds (from 0.5 to 0.95).

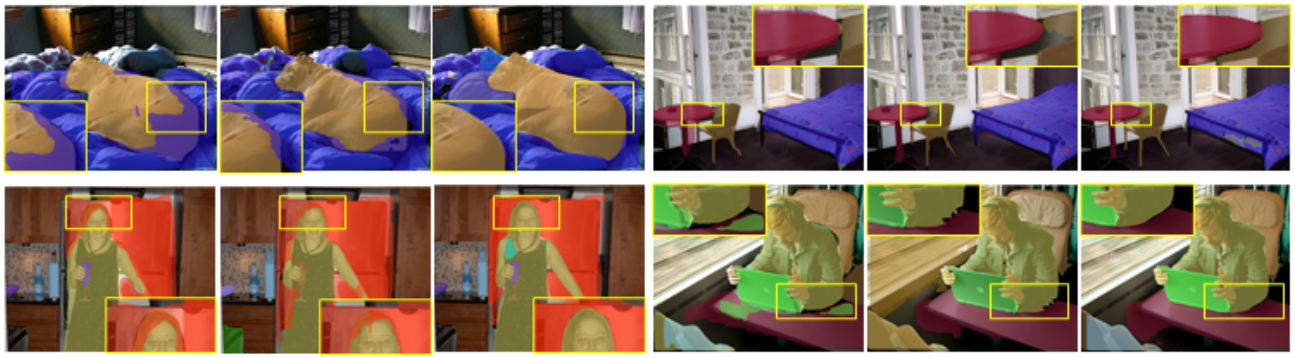

Figure 6. Example results of Mask R-CNN [10] (left), PointRend [17] (middle), and DynaMask (right), using ResNet-50-FPN as backbone.

4.1. Implementation Details

We adopt Mask R-CNN [10] as our baseline. The backbone is pre-trained on ImageNet. The hyper-parameters and loss functions are set the same as Mask R-CNN implemented in MMDetection [4] unless specified. The proposed MSM has four switching states, which correspond to four candidate resolutions, *i.e.*, $[14 \times 14, 28 \times 28, 56 \times 56, 112 \times 112]$. The hyper-parameter λ_1 is set as 0.1. We first pre-train the model without MSM using mask loss at all resolutions for one epoch. The initial learning rate is 0.02, and the batch size is 16 on 8 GPUs. Then we train all modules for 12 epochs using SGD with the same initial learning rate and batch size and reduce the learning rate by a factor of 0.1 after 8 and 11 epochs, respectively. Multi-scale training is used with the shorter side randomly sampled from $[640, 800]$, while for inference, the short side is resized to 800. In the ablation study, we use the standard $1\times$ training schedule and data augmentation defined in MMDetection [4].

4.2. Main Results

Comparison with Mask R-CNN. We first compare the performance of DynaMask with the baseline Mask R-CNN on COCO by using ResNet-50 and ResNet-101 backbones. As shown in Tab. 2, our method outperforms Mask R-CNN by a large margin. The performance is improved by 2.9% AP and 2.8% AP with "$1\times$" and "$2\times$" schedules, respectively, when ResNet-50-FPN backbone is used. Particularly, DynaMask outperforms the baseline by 3.3%, and 3.6% with "$1\times$" and "$2\times$" schedules in terms of AP_{75}, respectively. Similar observations can be obtained when ResNet-101-FPN is used as the backbone. This is because the proposed dual-level FPN structure incorporates complementary semantic and detailed information from multiple levels of FPN, resulting in more precise mask localization and higher-quality segmentation.

Comparison with state-of-the-art methods. We then compare the segmentation performance of DynaMask with many other state-of-the-art methods on COCO. The results are listed in Tab. 3. All models are trained on COCO `train2017` and evaluated on COCO `test-dev2017`. Without bells and whistles, DynaMask surpasses these methods with visible margins. Furthermore, we compare our method with other representative two-stage IS methods on Cityscapes `val` set in Tab. 4. Our method outperforms Mask R-CNN [10] by 4.2%. Noticeably, its performance on large objects is 5.8% AP higher than Mask R-CNN. This is because DynaMask employs high-resolution masks to achieve high-quality segmentation on large and difficult objects. RefineMask [30] also produces outstanding performance since it refines the mask prediction with fine-grained features, but it costs many computations due to the multi-stage refinement.

Visualizations of segmentation results. In Fig. 6, we visualize the segmentation results of DynaMask together with two representative methods: Mask R-CNN [10], and PointRend [17]. One can see that our method achieves finer and more accurate predictions around the objects' boundaries. This is because Dynamask introduces an r-FPN to fuse complementary information from multiple stages, so that difficult instances can be segmented with large masks which contain more fine-grained details. Mask R-CNN [10] employs a uniform coarse mask for prediction, while PointRend [17] makes point-wise refinement only at the blurred regions, which is not enough to capture sufficient details.

4.3. Ablation Study

Mask resolution prediction. To better understand how DynaMask selects suitable masks for different instances, we show the mask selection results in Fig. 7. As can be seen, the "hard" objects with irregular shapes and complex boundaries are assigned large masks, such as the "potted plant", "person", "giraffe", *etc*. On the contrary, the "easy" samples with regular shapes and less details are assigned small masks, such as the "skis", "suitcase", "dining table" and so on. It is worth mentioning that the choice of mask

Method	Backbone	Sched.	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Mask R-CNN [10]	ResNet-50-FPN	1×	34.7	55.7	37.2	18.3	37.4	47.2
DynaMask	ResNet-50-FPN	1×	37.6	57.4	40.5	20.7	40.4	50.3
Mask R-CNN [10]	ResNet-50-FPN	2×	35.4	56.4	37.9	19.1	38.6	48.4
DynaMask	ResNet-50-FPN	2×	38.2	58.1	41.5	20.5	40.8	52.7
Mask R-CNN [10]	ResNet-101-FPN	1×	36.1	57.5	38.6	18.8	39.7	49.5
DynaMask	ResNet-101-FPN	1×	38.7	58.8	41.8	20.9	41.8	52.4
Mask R-CNN [10]	ResNet-101-FPN	2×	36.6	57.9	39.1	19.2	40.2	50.5
DynaMask	ResNet-101-FPN	2×	39.0	59.1	42.2	20.9	42.1	53.3

Table 2. Comparison with Mask R-CNN on COCO `val2017`.

Method	Backbone	sched.	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
MEInst [31]	ResNet-101-FPN	3×	33.9	56.2	35.4	19.8	36.1	42.3
Mask R-CNN [10]	ResNet-101-FPN	3×	38.5	60.0	41.6	19.2	41.6	55.8
BMask R-CNN [6]	ResNet-101-FPN	1×	37.7	59.3	40.6	16.8	39.9	54.6
TensorMask [5]	ResNet-101-FPN	6×	37.1	59.3	39.4	17.4	39.1	51.6
Mask Scoring [14]	ResNet-101-FPN	18e	38.3	58.8	41.5	17.8	40.4	54.4
CondInst [26]	ResNet-101-FPN	3×	39.1	60.9	42.0	21.5	41.7	50.9
BlendMask [2]	ResNet-101-FPN	3×	38.4	60.7	41.3	18.2	41.5	53.3
PointRend [17]	ResNeXt-101-FPN	3×	41.4	63.3	44.8	24.2	43.9	53.2
SOLOv2 [28]	ResNet-101-FPN	6×	39.7	60.7	42.9	17.3	42.9	57.4
HTC [3]	ResNet-101-FPN	20e	39.7	61.8	43.1	21.0	42.2	53.5
DCT-Mask† [25]	ResNet-101-FPN	3×	40.1	61.2	43.6	22.7	42.7	51.8
RefineMask [30]	ResNet-101-FPN	3×	39.4	-	-	-	-	-
Mask Transfiner† [16]	ResNet-101-FPN	3×	40.7	-	-	23.1	42.8	53.8
QueryInst† [9]	ResNet-101-FPN	3×	41.7	64.4	45.3	24.2	43.9	53.9
DynaMask	ResNet-101-FPN	3×	39.6	61.4	42.9	21.2	42.4	53.2
DynaMask†	ResNet-101-FPN	3×	41.3	62.5	45.2	22.8	43.4	54.0
DynaMask†	ResNeXt-101-FPN	3×	42.0	62.9	46.3	23.4	44.0	54.8

Table 3. Comparison with state-of-the-art methods for instance segmentation on COCO `test-dev2017`, † denotes multi-scale training.

size is independent of object size. For example, in the third image, the small but hard object "potted plant" is allocated a large mask, while the large but easy sample "suitcase" in the fifth image can be accurately segmented with a small mask. We provide quantitative analysis of the correlations between the predicted mask resolutions and the class in **supplemental files**.

Influence of the budget constraint. In Tab. 5, we explore the influence of budget constraint on the model complexity (FLOPs). The average FLOPs are calculated by taking the average over all instances of the validation set. By tuning hyper-parameters $\mathcal{C}_t$ and λ_2 to different values, we get models of different computational costs. For example, the FLOPs are reduced by about 19% without sacrificing the segmentation performance by setting $\mathcal{C}_t$ and λ_2 to 1.0 and 0.4, respectively. This demonstrates that there exists much redundancy when using the large mask (112×112) for all instances. Actually, many instances in COCO dataset can be efficiently segmented with smaller masks. Thus the redundant computation for easier samples can be reduced, while the accuracy of harder samples can be maintained by still using larger masks. By setting the target FLOPs $\mathcal{C}_t$ to a smaller value (*e.g.*, 0.4, 0.6, 0.8), more significant computation reduction can be achieved at a slight degradation of accuracy. For instance, by setting both $\mathcal{C}_t$ and λ_2 to 0.4, more than half of the FLOPs (around 54%) are reduced while still producing competitive segmentation results (36.8% AP).

Speed comparison of different methods. To validate the efficiency of our model, we compare the model accuracy, FLOPs, and runtime of different two-stage IS methods in Tab. 6. Inference time is tested on one NVIDIA TITAN RTX with the input size 800×1333. Compared to these methods, our DynaMask method achieves visible performance gain at a small amount of extra computational cost. Specifically, DynaMask ($\mathcal{C}_t = 0.4$) outperforms PointRend [17] and DCT-mask [25] by 1.2% AP and 0.3% AP, respectively, with comparable runtime. Though HTC [3] produces a similar segmentation result (0.2% lower AP) to DynaMask, it is two times slower than DynaMask because it performs hybrid cascade refinement on both detection and segmentation tasks, resulting in a large amount of memory and computation overhead.

Influence of mask size. The Mask Switch Module (MSM) outputs four different candidate mask resolutions $[14 \times 14, 28 \times 28, 56 \times 56, 112 \times 112]$. We choose one of them as the uniform output mask size and report the corresponding results in Tab. 7. As can be seen, models with larger mask resolutions could achieve higher segmentation performance, especially on large objects, but the computational cost also increases obviously. For example, the performance is improved by 3.2%, 4.2%, and 4.7% AP when the mask resolution is increased from 14×14 to 28×28, 56×56

Figure 7. Mask selection results by DynaMask. Each color corresponds to a candidate resolution (red→large, blue→medium, green→small, yellow→tiny).

Method	AP	AP_S	AP_M	AP_L
Mask R-CNN [10]	33.8	12.0	31.5	51.8
PointRend [17]	35.8	-	-	-
BMask R-CNN [6]	35.8	-	-	-
RefineMask [30]	37.6	14.6	34.0	58.1
DynaMask	38.0	14.8	35.1	57.6

Table 4. The results on Cityscapes `val` set with ResNet-50-FPN backbone.

$\mathcal{C}_t$	λ_2	FLOPs	Δ	AP
1.0	0.3	1.27G	-9%	37.6
1.0	0.4	1.13G	-19%	37.6
1.0	0.5	1.06G	-24%	37.5
0.8	0.4	0.92G	-34%	37.4
0.6	0.4	0.83G	-41%	37.2
0.4	0.4	0.64G	-54%	36.8

Table 5. The influence of the budget constraint.

Method	AP	FLOPs	FPS
Mask R-CNN [10]	34.7	0.5G	12.4
PointRend [17]	35.6	0.9G	9.4
DCT-Mask† [25]	36.5	0.5G	11.8
HTC [3]	37.4	-	3.9
DynaMask	37.6	1.4G	8.3
DynaMask ($\mathcal{C}_t = 0.4$)	36.8	0.64G	11.2

Table 6. Speed comparison of different methods on `val` set, †: multi-scale training.

Mask Size	FLOPs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
14×14	0.23G	32.9	55.3	34.4	19.3	35.8	43.6
28×28	0.62G	36.1	57.1	38.7	20.4	39.2	48.2
56×56	1.01G	37.1	57.3	40.0	20.5	40.0	49.7
112×112	1.40G	37.6	57.4	40.5	20.7	40.4	50.3

Table 7. Performance of using different mask sizes.

Mask Size	FLOPs	AP	Tiny	Small	Medium	Large
28×28	0.62G	36.1	0	1	0	0
Size-based	0.56G	35.9	47%	32%	14%	8%
DynaMask ($\mathcal{C}_t = 0.4$)	0.64G	36.8	35%	34%	21%	10%
DynaMask	1.4G	37.6	0	0	0	1

Table 8. Comparison with size-based mask selection method.

and 112 × 112, respectively. Nevertheless, the performance tends to saturate as the mask size increases further.

Size-based mask selection method. We compare the performance and mask distributions of different mask selection methods in Tab. 8. The baseline is performed by using a uniform mask size (28 × 28) for all objects. Size-based method denotes assigning masks according to the size of object. Specifically, we assign a mask $\hat{m}_i^k$ to the instance of width w and height h by the following rule:

$$k = \left\lfloor k_0 + log_2(\sqrt{wh}/\sqrt{w_0 h_0}) \right\rfloor, \tag{8}$$

where w_0 and h_0 denote the width and height of the input image, respectively. and k_0 denotes the index of the highest mask resolution 112 × 112, *i.e.*, $k_0 = 4$. Intuitively, Eq. 8 means that the larger object will be assigned a higher-resolution mask. This conforms to our common sense, since the larger objects usually contain many details which need finer-grained masks to achieve high-quality prediction.

As can be seen from Tab. 8, the size-based method attains comparable performance with the baseline at a lower cost. Our DynaMask ($\mathcal{C}_t = 0.4$) outperforms the baseline by 0.7% AP and the size-based method by 0.9% AP at a slight extra computation cost, demonstrating that the proposed mask selection strategy could better partition different objects according to their segmentation difficulties and assign more suitable masks to achieve better performance. More analyses about the distribution of predicted mask scales can be found in **supplemental materials**.

5. Conclusion

In this work, we proposed a simple yet effective method to improve instance segmentation performance at a small amount of computation and memory overhead. We devised a dual-level FPN structure for better exploring complementary contextual and detailed information from multiple pyramid levels. Specifically, in addition to traditional image-level FPN (i-FPN), we augmented a region-level top-down path (r-FPN) to gradually enlarge the mask size and incorporate more details from i-FPN. Furthermore, to reduce the computation and memory cost led by using large masks, we introduced a Mask Switch Module (MSM) to adaptively select suitable masks for each proposal so that the redundant computation for easy samples can be reduced by using smaller masks. Extensive experimental results demonstrated that our method achieved significant performance gains with moderate extra computation overhead.

References

[1] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. Yolact: Real-time instance segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9157–9166, 2019. 1

[2] Hao Chen, Kunyang Sun, Zhi Tian, Chunhua Shen, Yongming Huang, and Youliang Yan. Blendmask: Top-down meets bottom-up for instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8573–8581, 2020. 1, 7

[3] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, et al. Hybrid task cascade for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4974–4983, 2019. 2, 7, 8

[4] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 6

[5] Xinlei Chen, Ross Girshick, Kaiming He, and Piotr Dollár. Tensormask: A foundation for dense object segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2061–2069, 2019. 7

[6] Tianheng Cheng, Xinggang Wang, Lichao Huang, and Wenyu Liu. Boundary-preserving mask r-cnn. In *European conference on computer vision*, pages 660–676. Springer, 2020. 1, 2, 3, 5, 7, 8

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 5

[8] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017. 3

[9] Yuxin Fang, Shusheng Yang, Xinggang Wang, Yu Li, Chen Fang, Ying Shan, Bin Feng, and Wenyu Liu. Instances as queries. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6910–6919, 2021. 7

[10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 1, 2, 3, 6, 7, 8

[11] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 4, 5

[12] Gao Huang and Danlu Chen. Multi-scale dense networks for resource efficient image classification. *ICLR 2018*, 2018. 2

[13] Shihua Huang, Zhichao Lu, Ran Cheng, and Cheng He. Fapn: Feature-aligned pyramid network for dense image prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 864–873, 2021. 3

[14] Zhaojin Huang, Lichao Huang, Yongchao Gong, Chang Huang, and Xinggang Wang. Mask scoring r-cnn. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6409–6418, 2019. 2, 7

[15] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *stat*, 1050:5, 2017. 4

[16] Lei Ke, Martin Danelljan, Xia Li, Yu-Wing Tai, Chi-Keung Tang, and Fisher Yu. Mask transfiner for high-quality instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4412–4421, 2022. 7

[17] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9799–9808, 2020. 1, 2, 3, 6, 7, 8

[18] Yanwei Li, Lin Song, Yukang Chen, Zeming Li, Xiangyu Zhang, Xingang Wang, and Jian Sun. Learning dynamic routing for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8553–8562, 2020. 3

[19] Ji Lin, Yongming Rao, Jiwen Lu, and Jie Zhou. Runtime neural pruning. *Advances in neural information processing systems*, 30, 2017. 2

[20] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 2, 3

[21] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 5

[22] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8759–8768, 2018. 2

[23] Ravi Teja Mullapudi, William R Mark, Noam Shazeer, and Kayvon Fatahalian. Hydranets: Specialized dynamic architectures for efficient inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8080–8089, 2018. 2

[24] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 1, 2, 3

[25] Xing Shen, Jirui Yang, Chunbo Wei, Bing Deng, Jianqiang Huang, Xian-Sheng Hua, Xiaoliang Cheng, and Kewei Liang. Dct-mask: Discrete cosine transform mask representation for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8720–8729, 2021. 1, 2, 7, 8

[26] Zhi Tian, Chunhua Shen, and Hao Chen. Conditional convolutions for instance segmentation. In *European Conference on Computer Vision*, pages 282–298. Springer, 2020. 1, 7

[27] Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. Skipnet: Learning dynamic routing in

convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 409–424, 2018. 2, 3

[28] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. Solov2: Dynamic and fast instance segmentation. *Advances in Neural information processing systems*, 33:17721–17732, 2020. 7

[29] Zhonghui You, Kun Yan, Jinmian Ye, Meng Ma, and Ping Wang. Gate decorator: Global filter pruning method for accelerating deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 32, 2019. 2

[30] Gang Zhang, Xin Lu, Jingru Tan, Jianmin Li, Zhaoxiang Zhang, Quanquan Li, and Xiaolin Hu. Refinemask: Towards high-quality instance segmentation with fine-grained features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6861–6869, 2021. 2, 6, 7, 8

[31] Rufeng Zhang, Zhi Tian, Chunhua Shen, Mingyu You, and Youliang Yan. Mask encoding for single shot instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10226–10235, 2020. 7

[32] Mingjian Zhu, Kai Han, Enhua Wu, Qiulin Zhang, Ying Nie, Zhenzhong Lan, and Yunhe Wang. Dynamic resolution network. *Advances in Neural Information Processing Systems*, 34, 2021. 3