https://arxiv.org/pdf/1903.12174

Preview:

a. Today’s top performing object detectors rely on **sliding window prediction to generate initial candidate regions**, **a refinement stage** is applied to these candidate regions to obtain more accurate predictions, as pioneered by Faster R-CNN [34] and Mask R-CNN [17] for bounding-box object detection and instance segmentation.
b. bounding-box object detectors which **eschew the refinement step** and focus on direct sliding-window prediction, as exemplified by SSD [27] and RetinaNet [23], have witnessed a resurgence and shown promising results.
c. the field has not witnessed equivalent progress in **dense sliding-window instance segmentation**; there are no direct, dense approaches analogous to SSD / RetinaNet for mask prediction.
d. The goal of this work is to bridge this gap and provide a foundation for exploring dense instance segmentation.

https://towardsdatascience.com/review-deepmask-instance-segmentation-30327a072339

Instance segmentation其实是semantic segmentation和object detection殊途同归的一个结合点，是个挺重要的研究问题. 我非常期待后面能同时结合semantic segmentation和object detection两者优势的instance segmentation算法和网络结构.

TensorMask

1. Treat dense instance segmentation as a prediction task over 4D tensors and present a general framework called TensorMask that explicitly captures this geometry and enables novel operators on 4D tensors.

TensorMask 通用框架， 可以明确的捕捉这种几何结构，并使得在4D Tensor上进行操作成为可能。（挖了个坑）

2. Tensor Representations for Masks

The central idea of the TensorMask framework is to use structured high-dimensional tensors to represent image content (e.g., masks) in a set of densely sliding windows.

例如，如果在特征图 W×H 上有一个 V ×U大小的滑动窗口。那么我们可以使用一个形状为 (C, H, W) 的张量表示所有滑动窗口上的所有 Mask，且每一个 Mask 可以通过 C=V ·U个像素参数化，这就是 DeepMask 中采用的表征。

实际上，这种表征的潜在观点即使用更高维张量——4D 的 (V, U, H, W)。其中子张量 (V, U) 将一个二维空间实体表示为Mask。

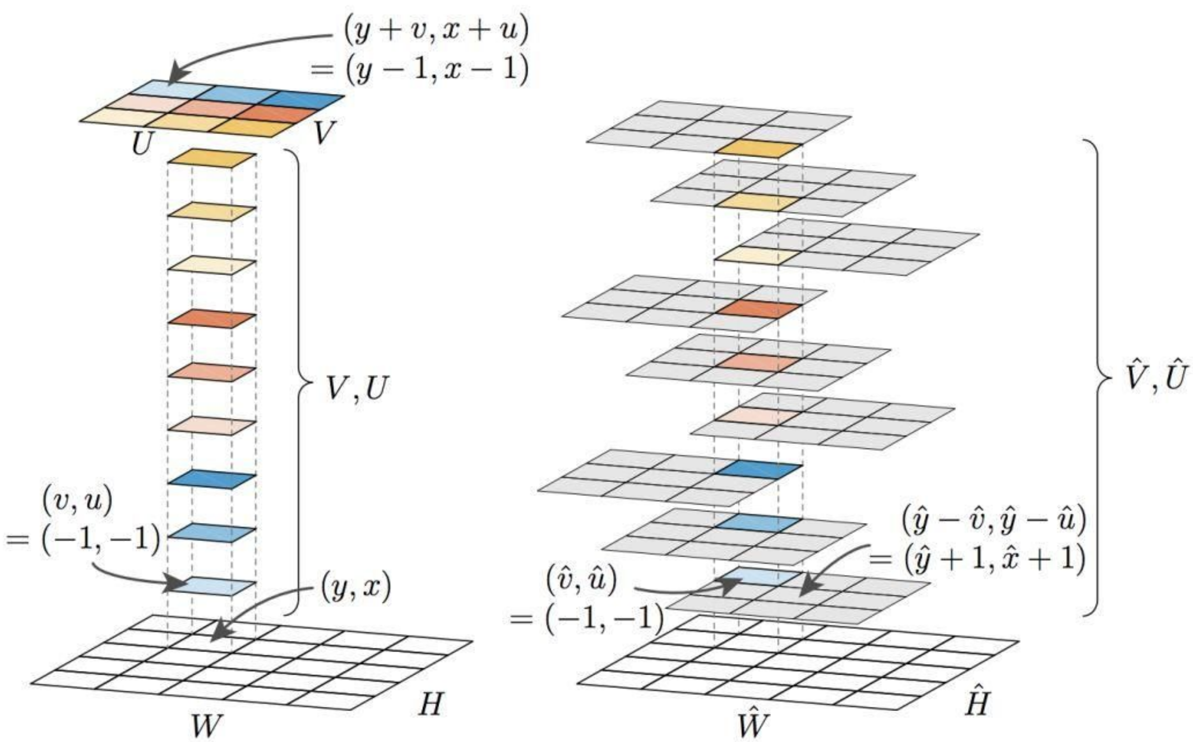
<1> Unit of length:
The unit of an axis defines the length of one pixel along it. Different axes can have different units.
一个轴的单位定义了对应单个像素的长度，不同的轴有不同的单位。例如，H 和 W 轴的单位表示为σ_HW，它定义为有关输入图像的步辐。

<2> Natural Representation:
定义单位后，我们就可以描述 (V, U, H, W) 张量的表征意义。在最简单的定义中，它表示 (H, W) 上的滑动窗口，这可以称为自然表征。
its value at coordinates (v, u, y, x) represents the mask value at (y + αv, x + αu) in the αV ×αU window centered at (y, x).

<3> Aligned Representation: (对齐表征)
For a 4D tensor (V , ˆ U, ˆ H, ˆ W ˆ), its value at coordinates (ˆv, u, ˆ y, ˆ x ˆ) represents the mask value at (ˆy, x ˆ) in the αˆVˆ×αˆUˆ window centered at (ˆy − αˆv, ˆ x ˆ − αˆu ˆ).

下图展示了这两种表征：

左图为自然表征，其中 (V, U) 子张量表示以该像素为中心的窗口。右图为对齐表征，(V hat, U hat) 子张量表示该像素在各窗口的值。



<4> Coordinate Transformation:(坐标转换)
引入了这种方法以在自然表征和为对齐表征之间做转换，这会给设计新架构带来额外的灵活性。

For simplicity, we assume units in both representations are the same: *i.e.*, $\sigma_{\text{HW}}{=}\hat{\sigma}_{\text{HW}}$ and $\sigma_{\text{VU}}{=}\hat{\sigma}_{\text{VU}}$, and thus $\alpha{=}\hat{\alpha}$ (for the more general case see §A.1). Comparing the definitions of natural *vs.* aligned representations, we have the following two relations for x, u: $x{+}\alpha u{=}\hat{x}$ and $x{=}\hat{x}{-}\hat{\alpha}\hat{u}$. With $\alpha{=}\hat{\alpha}$, solving this equation for $\hat{x}$ and $\hat{u}$ gives: $\hat{x}{=}x{+}\alpha u$ and $\hat{u}{=}u$. A similar results hold for y, v. So the transformation from the aligned representation ($\hat{\mathcal{F}}$) to the natural representation ($\mathcal{F}$) is:

$$\mathcal{F}(v, u, y, x) = \hat{\mathcal{F}}(v, u, y + \alpha v, x + \alpha u). \qquad (1)$$

We call this transform `align2nat`. Likewise, solving this set of two relations for x and u gives the reverse transform of `nat2align`: $\hat{\mathcal{F}}(\hat{v}, \hat{u}, \hat{y}, \hat{x}){=}\mathcal{F}(\hat{v}, \hat{u}, \hat{y}{-}\alpha\hat{v}, \hat{x}{-}\alpha\hat{u})$. While all the models presented in this work only use `align2nat`, we present both cases for completeness.

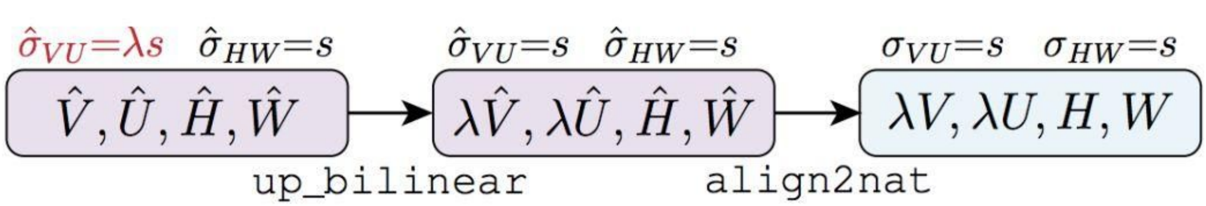
<5> Upscaling Transformation:(放大转换)

对齐表征允许使用粗粒度的子张量 (V hat, U hat)创建细粒度的子张量 (V, U)。

The aligned representation enables the use of a *coarse* $(\hat{V}, \hat{U})$ sub-tensors to create finer (V, U) sub-tensors, which proves quite useful. Fig. 4 illustrates this transformation, which we call `up_align2nat` and describe next.

The `up_align2nat` op accepts a $(\hat{V}, \hat{U}, \hat{H}, \hat{W})$ tensor as input. The $(\hat{V}, \hat{U})$ sub-tensor is $\lambda\times$ coarser than the desired output (so its unit is $\lambda\times$ bigger). It performs bilinear upsampling, `up_bilinear`, in the $(\hat{V}, \hat{U})$ domain by λ, reducing the underlying unit by $\lambda\times$. Next, the `align2nat` op converts the output into the natural representation. The full `up_align2nat` op is shown in Fig. 4.

As our experiments demonstrate, the `up_align2nat` op is effective for generating high-resolution masks without inflating channel counts in preceding feature maps. This in turn enables novel architectures, as described next.



<6> Bipyramid:

在目标框检测中，使用特征金字塔非常常见。为此在 Mask 张量中，我们不再使用 V ×U 个单元表示不同尺度的Mask，我们提出了这种基于尺度来调整 Mask 像素数量的方法。

Tensor bipyramid: *A tensor bipyramid is a list of tensors of shapes:* $(2^kV, 2^kU, \frac{1}{2^k}H, \frac{1}{2^k}W)$, *for* k=0, 1, 2, . . ., *with units* $\sigma_{VU}^{k+1} = \sigma_{VU}^{k}$ *and* $\sigma_{HW}^{k+1} = 2\sigma_{HW}^{k}, \forall k$.

Because the units σ_{VU}^{k} are the same across all levels, a $2^kV \times 2^kU$ mask has $4^k\times$ more pixels in the input image. In the (H, W) domain, because the units σ_{HW}^{k} increase with k, the number of predicted masks decreases for larger masks, as desired. Note that the total size of each level is the same (it is $V{\cdot}U{\cdot}H{\cdot}W$). A tensor bipyramid can be constructed using the `swap_align2nat` operation, described next.

This `swap_align2nat` op is composed of two steps: first, an input tensor with fine $(\hat{H}, \hat{W})$ and coarse $(\hat{V}, \hat{U})$ is upscaled to $(2^kV, 2^kU, H, W)$ using `up_align2nat`. Then (H, W) is subsampled to obtain the final shape. The combination of `up_align2nat` and `subsample`, shown in Fig. 5, is called `swap_align2nat`: the units before and after this op are *swapped*. For efficiency, it is not necessary to compute the intermediate tensor of shape $(2^kV, 2^kU, H, W)$ from `up_align2nat`, which would be prohibitive. This is because only a small subset of values in this intermediate tensor appear in the final output after subsampling. So although Fig. 5 shows the conceptual computation, in practice we implement `swap_align2nat` as a single op that only performs the necessary computation and has complexity $O(V{\cdot}U{\cdot}H{\cdot}W)$ regardless of k.

3. TensorMask:

这些模型有一个预测 Mask 的 Head，它在滑动窗口中生成 Mask；同时也有一个进行分类的Head，它可以预测目标类别。它们类似于滑动窗口目标检测器中的边界框回归和分类分支。边界框预测对于 TensorMask模型并不是必要的，但可以便捷地包含进来。

图 (C, H, W)。图 6: 基线 Mask 预测 Head，这四种 Head 都从通道为 C 的特征图开始。图 7: 使用基线 Head 的特征金字塔，与 Tensor Bipyramid 的对比。图 8: 使用 Tensor Bipyramid 将 FPN 特征图从__转换到(C, H, W)。

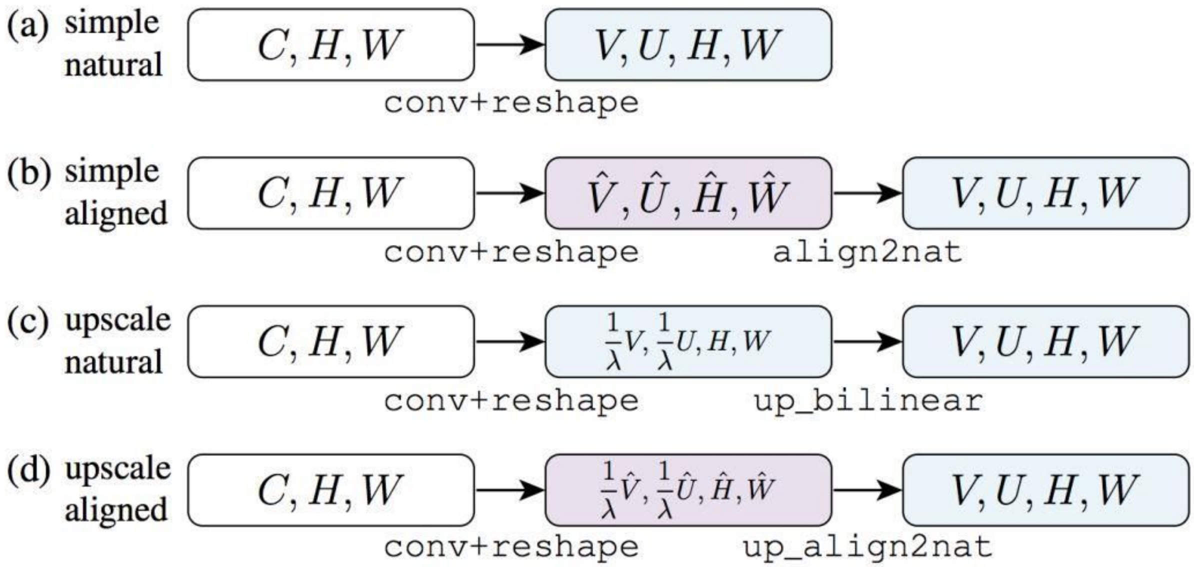


图 6: 基线 Mask 预测 Head，这四种 Head 都从通道为 C 的特征图开始。

Figure 6. **Baseline mask prediction heads**: Each of the four heads shown starts from a feature map (*e.g.*, from a level of an FPN [22]) with an arbitrary channel number C. Then a 1×1 `conv` layer projects the features into an appropriate number of channels, which form the specified 4D tensor by `reshape`. The output units of these four heads are the same, and $\sigma_{\mathrm{VU}}=\sigma_{\mathrm{HW}}$.

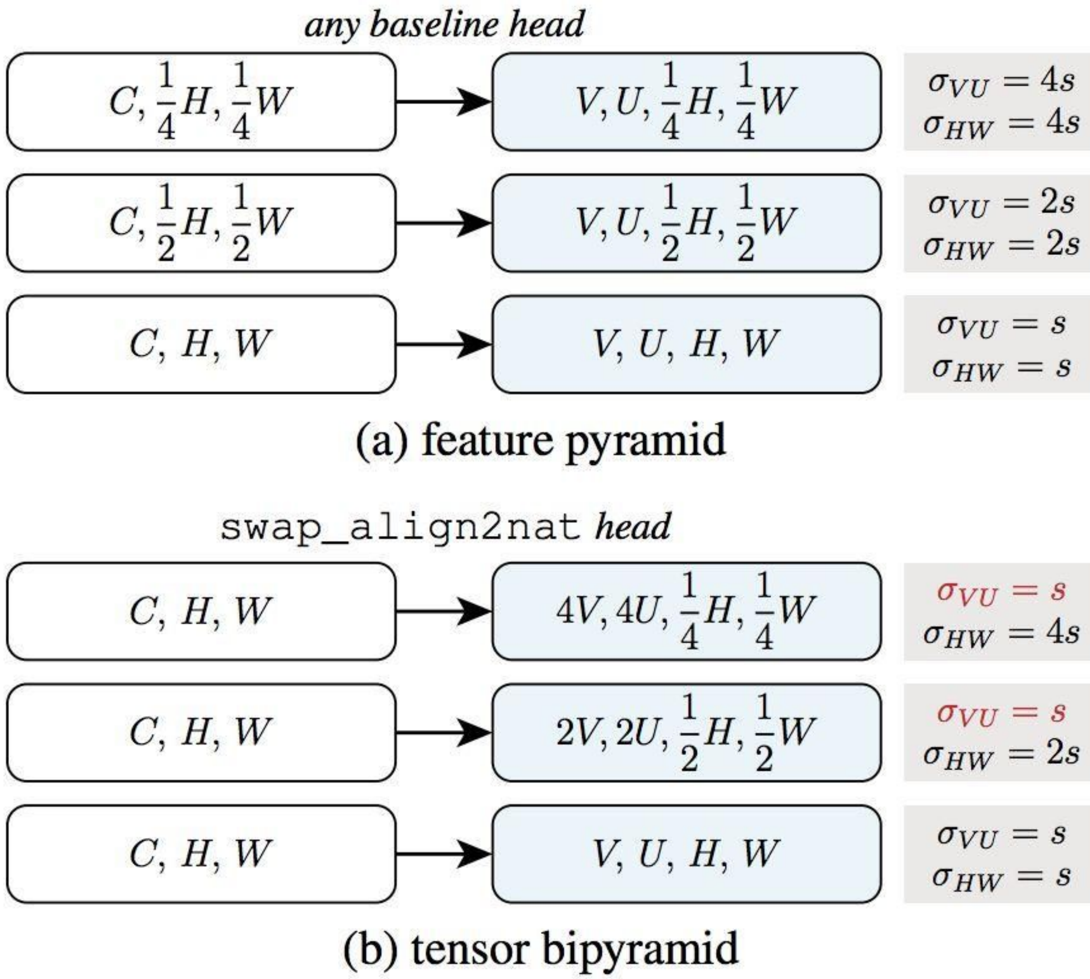


图 7: 使用基线 Head 的特征金字塔，与 Tensor Bipyramid 的对比。

Figure 7. Conceptual comparison between: (a) a **feature pyramid** with any one of the baseline heads (Fig. 6) attached, and (b) a **tensor bipyramid** that uses `swap_align2nat` (Fig. 5). A baseline head on the feature pyramid has $\sigma_{\mathrm{VU}}=\sigma_{\mathrm{HW}}$ for each level, which implies that masks for large objects and small objects are predicted using the same number of pixels. On the other hand, the `swap_align2nat` head can keep the mask resolution high (*i.e.*, σ_{VU} is the same across levels) despite the HW resolution changes.

Tensor bipyramid head. Unlike the baseline heads, the tensor bipyramid head (§3.6) accepts a feature map of fine resolution (H, W) at all levels. Fig. 8 shows a minor modification of FPN to obtain these maps. For each of the resulting levels, now all (C, H, W), we first use `conv+reshape` to produce the appropriate 4D tensor, then run a mask prediction head with `swap_align2nat`, see Fig. 7b. The tensor bipyramid model is the most effective TensorMask variant explored in this work.

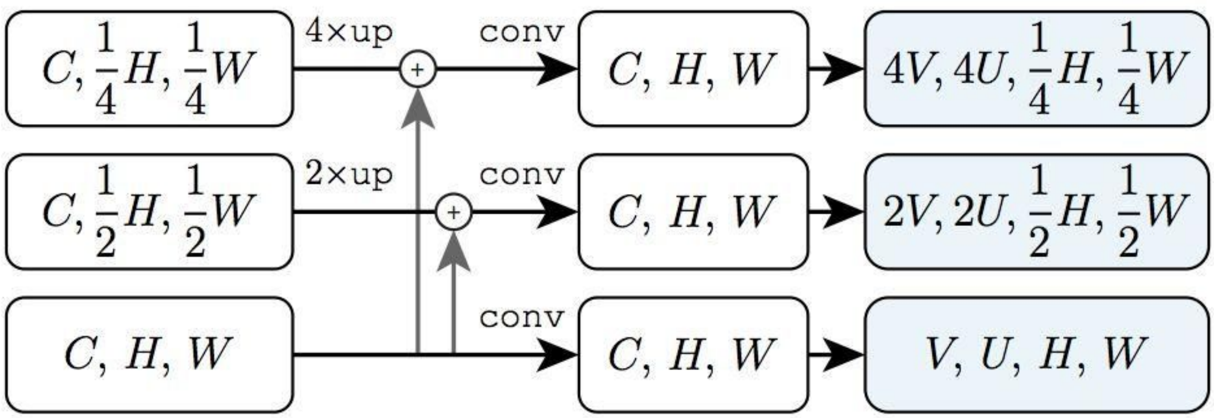


图 8: 使用 Tensor Bipyramid 将 FPN 特征图从__

$$\left(C, \tfrac{1}{2^k}H, \tfrac{1}{2^k}W\right)$$ *转换到*

(C, H, W)。

head	λ	AP	AP$_{50}$	AP$_{75}$	Δ aligned - natural		
natural	1.5	28.8	52.0	28.9	*+0.9*	*+0.5*	*+1.7*
aligned		**29.7**	**52.5**	**30.6**			
natural	3	25.4	48.8	23.7	*+4.1*	*+3.4*	*+6.6*
aligned		**29.5**	**52.2**	**30.3**			
natural	5	13.5	33.9	9.0	*+15.6*	*+18.2*	*+20.8*
aligned		**29.1**	**52.1**	**29.8**			

(a) **Upscaling heads**: natural *vs.* aligned heads (Fig. 6c *vs.* 6d). The $V \times U$=15×15 output is upscaled by $\lambda\times$: `conv+reshape` uses $\frac{1}{\lambda^2}VU$ output channels as input. *The aligned representation has a large gain over its natural counterpart when λ is large.*

head	λ	AP	AP$_{50}$	AP$_{75}$	Δ bilinear - nearest		
nearest	1.5	29.4	52.4	30.1	*+0.3*	*+0.1*	*+0.5*
bilinear		**29.7**	**52.5**	**30.6**			
nearest	3	28.5	51.3	28.8	*+1.0*	*+0.9*	*+1.5*
bilinear		**29.5**	**52.2**	**30.3**			
nearest	5	25.9	47.8	25.6	*+3.2*	*+4.3*	*+4.2*
bilinear		**29.1**	**52.1**	**29.8**			

(b) **Upscaling: bilinear *vs.* nearest-neighbor** interpolation for the aligned head (Fig. 6d). The output has $V \times U$=15×15. With nearest-neighbor interpolation, the aligned upscaling head is similar to the InstanceFCN [7] head. *Bilinear interpolation shows a large gain when λ is large.*

head	AP	AP$_{50}$	AP$_{75}$	AP$_S$	AP$_M$	AP$_L$
feature pyramid, best	29.7	52.5	30.6	15.1	32.2	40.7
tensor bipyramid	**33.8**	**54.8**	**35.8**	**16.1**	**36.3**	**47.7**
Δ	*+4.1*	*+2.3*	*+5.2*	*+1.0*	*+4.1*	*+7.0*

(c) The **tensor bipyramid** substantially improves results compared to the best baseline head (Tab. 2a, row 2) on a *feature pyramid* (Fig. 7a).

$V \times U$	AP	AP$_{50}$	AP$_{75}$	AP$_S$	AP$_M$	AP$_L$
15×15	33.8	54.8	35.8	16.1	36.3	47.7
15×15, 11×11	**35.4**	**56.5**	**37.5**	**16.4**	**37.9**	**50.0**
Δ	*+1.6*	*+1.7*	*+1.7*	*+0.3*	*+1.6*	*+2.3*

(d) **Window sizes**: extending from one $V \times U$ window size (per level) to two increases all AP metrics. Both rows use the tensor bipyramid.

Table 2. **Ablations on TensorMask representations** on COCO `val2017`. All variants use ResNet-50-FPN and a 72 epoch schedule.

5.1. TensorMask Representations

First we explore various tensor representations for masks using V=U=15 and a ResNet-50-FPN backbone. We report quantitative results in Tab. 2 and show qualitative comparisons in Figs. 2 and 9.

Simple heads. Tab. 1 compares natural *vs.* aligned representations with simple heads (Fig. 6a *vs.* 6b). Both representations perform similarly, with a marginal gap of 0.2 AP. The simple natural head can be thought of as a *class-specific* variant of DeepMask [31] with an FPN backbone [22] and focal loss [23]. As we aim to use lower-resolution intermediate representations, we explore upscaling heads next.

Upscaling heads. Tab. 2a compares natural *vs.* aligned representations with upscaling heads (Fig. 6c *vs.* 6d). The output size is fixed at $V \times U$=15×15. Given an upscaling factor λ, the conv in Fig. 6 has $\frac{1}{\lambda^2}VU$ channels, *e.g.*, 9 channels with λ=5 (*vs.* 225 channels if no upscaling). The difference in accuracy is big for large λ: the aligned variant improves AP *+15.6* over the natural head (115% relative) when λ=5.

The visual difference is clear in Fig. 9a (natural) *vs.* 9c (aligned). The upscale aligned head still produces sharp masks with large λ. *This is critical for the tensor bipyramid, where we have an output of* $2^kV \times 2^kU$*, which is achieved with a large upscaling factor of* λ=2^k (e.g., 32)*;* see Fig. 5.

Interpolation. The tensor view reveals the $(\hat{V}, \hat{U})$ subtensor as a 2D spatial entity that can be manipulated. Tab. 2b compares the upscale aligned head with bilinear (default) *vs. nearest-neighbor* interpolation on $(\hat{V}, \hat{U})$. We refer to this latter variant as *unaligned* since quantization breaks pixel-to-pixel alignment. The unaligned variant is related to InstanceFCN [7] (see §A.2).

We observe in Tab. 2b that bilinear interpolation yields solid improvements over nearest-neighbor interpolation, especially if λ is large (ΔAP=*3.2*). These interpolation methods lead to striking visual differences when objects overlap: see Fig. 9b (unaligned) *vs.* 9c (aligned).

Tensor bipyramid. Replacing the best feature pyramid model with a tensor bipyramid yields a large 4.1 AP improvement (Tab. 2c). Here, the mask size is $V \times U$=15×15 on level k=0, and is $32V \times 32U$=480×480 for k=5; see Fig. 7b. The higher resolution masks predicted for large objects (*e.g.*, at k=5) have clear benefit: AP_L jumps by

7.0 points. This improvement does *not* come at the cost of denser windows as the $k{=}5$ output is at $(\frac{H}{32}, \frac{W}{32})$ resolution.

Again, we note that it is intractable to have, *e.g.*, a 480^2-channel `conv`. The upscaling aligned head with bilinear interpolation is key to making tensor bipyramid possible.

Multiple window sizes. Thus far we have used a single window size (per-level) for all models, that is, $V{\times}U{=}15{\times}15$. Analogous to the concept of *anchors* in RPN [34] that are also used in current detectors [33, 27, 23], we extend our method to multiple window sizes. We set $V{\times}U{\in}\{15{\times}15, 11{\times}11\}$, leading to two heads per level. Tab. 2d shows the benefit of having two window sizes: it increases AP by 1.6 points. More window sizes and aspect ratios are possible, suggesting room for improvement.

表 3 总结了测试-开发集上的最好 TensorMask 模型，并与当前 COCO 实例分割的主流模型 Mask RCNN 进行了对比。

method	backbone	aug	epochs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Mask R-CNN [13]	R-50-FPN		24	34.9	57.2	36.9	15.4	36.6	50.8
Mask R-CNN, *ours*	R-50-FPN		24	34.9	56.8	36.8	15.1	36.7	50.6
Mask R-CNN, *ours*	R-50-FPN	✓	72	**36.8**	**59.2**	**39.3**	**17.1**	**38.7**	**52.1**
TensorMask	R-50-FPN	✓	72	35.5	57.3	37.4	16.6	37.0	49.1
Mask R-CNN, *ours*	R-101-FPN	✓	72	**38.3**	**61.2**	**40.8**	**18.2**	**40.6**	**54.1**
TensorMask	R-101-FPN	✓	72	37.3	59.5	39.5	17.5	39.3	51.6

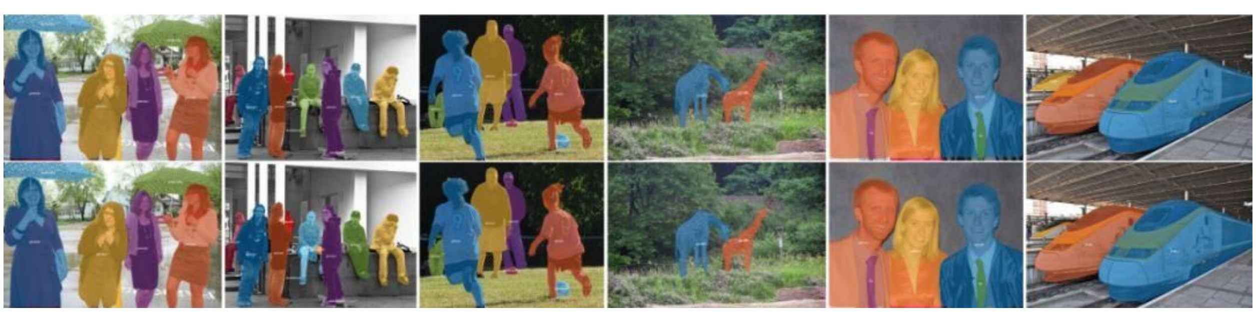


Figure 10. More results of Mask R-CNN [17] (top row per set) and TensorMask (bottom row per set) on the last 65 `val2017` images (continued in Fig. 11). These models use a ResNet-101-FPN backbone and obtain 38.3 and 37.3 AP, on `test-dev`, respectively. Visually, TensorMask gives sharper masks compared to Mask R-CNN although its AP is 1 point lower. Best viewed in a digital format with zoom.