Mask Encoding for Single Shot Instance Segmentation*

Rufeng Zhang[1], Zhi Tian[2], Chunhua Shen[2], Mingyu You[1], Youliang Yan[3]
[1] Tongji University, China [2] University of Adelaide, Australia [3] Huawei Noah's Ark Lab

Abstract

To date, instance segmentation is dominated by two-stage methods, as pioneered by Mask R-CNN. In contrast, one-stage alternatives cannot compete with Mask R-CNN in mask AP, mainly due to the difficulty of compactly representing masks, making the design of one-stage methods very challenging. In this work, we propose a simple single-shot instance segmentation framework, termed mask encoding based instance segmentation (MEInst). *Instead of predicting the two-dimensional mask directly,* MEInst *distills it into a compact and fixed-dimensional representation vector, which allows the instance segmentation task to be incorporated into one-stage bounding-box detectors and results in a simple yet efficient instance segmentation framework. The proposed one-stage* MEInst *achieves* 36.4% *in mask AP with single-model (ResNeXt-101-FPN backbone) and single-scale testing on the MS-COCO benchmark. We show that the much simpler and flexible one-stage instance segmentation method, can also achieve competitive performance. This framework can be easily adapted for other instance-level recognition tasks.*

Code is available at: git.io/AdelaiDet

1. Introduction

Instance segmentation enables various visual applications like autonomous driving and robot navigation, to name a few. Instead of separately detecting objects or assigning category labels to pixels, instance segmentation unifies these tasks together, thus being one of the most challenging tasks in computer vision.

Recent advances in deep convolutional neural networks (CNNs) have enabled tremendous progress in instance segmentation, *e.g.*, [13, 15, 17, 22]. One of the mainstream methods employs a two-stage pipeline that first generates proposals and then performs pixel classification within each proposal, as popularized by Mask R-CNN [13]. Almost all the methods in the top rank on the challenging COCO benchmark [20] are built upon Mask R-CNN thus far. One drawback of these two-stage solutions is not sufficiently efficient as their runtime is constrained by the number of instances in an image. On the other hand, one-stage paradigms process the full image straightforward, making the speed stable no matter how many objects present.

(a) Contour-Based (b) Mask-Based

Figure 1: Contour-Based [33] *vs*. Mask-Based. "Hollow Decay" is depicted with **red** dashed rectangles. The contour-based methods exhibit systematic artifacts on "disjointed" objects.

Several works have attempted to incorporate mask prediction into fully convolutional networks (FCNs) [24], resulting in single shot instance segmentation frameworks. These algorithms share a common insight, *i.e.*, encoding the object shape with a set of contour coefficients. Specifically, ESE-Seg [33] designs an "inner-center radius" shape signature for each instance and fits it with Chebyshev polynomials. Concurrently, PolarMask [32] regresses the dense distance of rays between mass-center and contours. These contour-based methods enjoy the advantages of easy optimization and fast inference. The major issue of these methods is that the predicted masks may exhibit "hollow decay" inevitably, since they can only depict instances with a single contour, as shown in Figure 1.

Alternatively, a non-parametric mask representation is more natural for mask prediction as traditionally done, with the price of increasing both design and computation complexity. As natural object masks are not random and akin to natural images, instance masks reside in a much lower intrinsic dimension than that of the pixel space. This in-

*Correspondence shoud be addressed to C. Shen and M. You.

spires us to ask a question, "Is it possible to predict the object mask in the intrinsic low-dimensional space and still achieve competitive accuracy?" Here we provide an affirmative answer: we propose to encode instance masks using a learned dictionary such that only a few scalar coefficients are needed to represent each mask. We demonstrate that such an approach is robust to noise, and efficient, easy to decode for reconstruction.

Then an one-stage detector such as RetinaNet [19], FCOS [29] can be easily extended by adding a branch for predicting these fixed-dimensional mask coefficients, along with the bounding box regression and category classification branches. We build our method on top of FCOS for its simplicity and good detection performance.

We demonstrate that our method can outperform recent one-stage algorithms [3, 32, 33, 35] with this simple design. In particular, experiments on the COCO `val2017` show that MEInst achieves a large gain compared to ESE-Seg [33], outperforming by 11.3% in AP_{50} and 15.9% in AP_{75}, respectively. Our model beats PolarMask [32] in accuracy with similar computational complexity, owing to the lower reconstruction error and more effective reconstruction. This is expected, as the mask representation of our method is more powerful than the parametric representation of [32, 33].

Additionally, we take a closer look at how the object detector influences the performance of instance segmentation based on extensive qualitative experiments. With a careful design based on our finding, MEInst achieves comparable performance with Mask R-CNN [13] with the advantage of being much simpler and flexible.

It is noteworthy that our method is compatible with most one-stage detection frameworks including the anchor-free paradigm. We demonstrate its generality using the FCOS detector, and evaluate the performance on the COCO benchmark [20]. Other anchor-based methods such as YOLO [26], RetinaNet [19] may be used here with minimum modification. Moreover, the vanilla detectors can also benefit from the paralleled mask prediction branch, improving the bounding box detection accuracy.

The main contributions of this work can be summarized as follows.

- We propose to encode a two-dimensional instance mask into a compact representation vector. The compressed vector, takes advantages of the redundancy in the original mask and proves to be effective and efficient for reconstruction.

  Encoding can be done with a few dictionary learning methods, including PCA, sparse coding, and auto-encoders. Here we show that even the simplest PCA already suffices for mask encoding.
- With this mask representation, a new framework is introduced for single shot instance segmentation, termed mask encoding based instance segmentation (MEInst), by extending FCOS [29] with a mask branch for mask coefficient regression. Actually, our mask encoding is completely independent of the mechanism of detectors, and it may be easily incorporated into other detectors.
- We demonstrate a simple and flexible one-stage instance segmentation method. Our best model, attains mask AP of 37.8% on COCO `test-dev`, achieving a good balance between accuracy and speed.

2. Related Work

We review a few works that are most relevant to ours.

Two-stage Instance Segmentation The mainstream approaches to instance segmentation [9, 13, 15, 22] inherit the pipeline of two-stage object detectors, as pioneered by Mask R-CNN [13]. These methods typically detect instance bounding boxes and then perform binary-class segmentation in boxes. Compared with segmentation-driven ones [1, 21], this group of paradigms lead on most benchmarks [8, 20] in accuracy. In particular, Mask R-CNN [13] replaces ROIPool with ROIAlign to better align features. Following Mask R-CNN, Liu *et al*. [22] present bottom-up path augmentation and adaptive feature pooling for further feature optimization. Mask Scoring R-CNN [15] extends Mask R-CNN with an extra MaskIoU branch, aiming to calibrate the mismatch between mask's quality and the corresponding confidence. The above methods consistently advance the performance.

One-stage Instance Segmentation The second family of solutions [1, 2, 16, 21] are built upon the success of semantic segmentation, *i.e*., generating pixel-wise classification maps firstly and then clustering them into instances. Specifically, InstanceCut [16] addresses the problem with two paralleled sub-tasks, instance-agnostic segmentation and instance-specific boundaries.

In the meantime, dense object segmentation has not witnessed remarkable progress. Impressively, several works have attempted to fill in the gaps lately. For example, TensorMask [7] can be viewed as a precursor to this group of algorithms, in which a structured 4D tensor is introduced to represent the mask over a spatial domain. It achieves similar performance with two-stage methods with the cost of heavy computation overhead in training and testing. In YOLACT [3], a series of global prototypes and individual linear coefficients are assembled for masks, achieving a real-time speed. BlendMask [4] improves YOLACT in both accuracy and speed. Recently, Xie *et al*. propose a general framework named PolarMask [32], which is capable to directly predict the mask without bounding box using a parametric representation of masks. More recently, SOLO

and its improved version SOLOv2 demonstrate promosing results with a simple FCN-like framework [30, 31].

3. Our Method

In this section, we first present the overall architecture of MEInst. We then introduce the instance representation with mask encoding and its optimization. Finally, we explore the correlation between detection quality and mask generation to further improve the performance of MEInst.

3.1. Network Architecture

The object detection modules in our method mainly inherit the pipeline from FCOS[1] [29] for its flexibility and simplicity, including a backbone module [14], a feature pyramid module [18], and two task-specific heads for classification, box regression and center-ness (they share the same head). Then a parallel branch is included for predicting encoded mask coefficients. Additionally, we carefully re-design some parts of the framework, which further boosts the performance. Details are discussed in the following subsection. The overall framework is illustrated in Figure 2.

3.2. Mask Encoding

Given a structured instance mask, we can easily figure out the redundancy in its representation. An example can be seen in Figure 3(b). The discriminative pixels are mainly distributed along the object boundaries while most pixels in its body hold the properties of being category-continuous and category-consistent. In other words, the existing mask representations contains redundant information and it may be highly compressed with negligible loss. In this subsection, we describe how to encode the two-dimensional geometry into a much more compact representation vector in detail.

Compact Representation Let $\mathbf{M}' \in \mathbb{R}^{H\times W}$ represent the ground truth mask and $\boldsymbol{v} \in \mathbb{R}^N$ be the compressed vector, where H, W and N denotes the height/width of two-dimensional mask and the dimension of compact representation vector, respectively. Typically $N \ll H \cdot W$. Note that here $\mathbf{M}'$ is class-agnostic and therefore all the categories are encoded with binary-class encoding, *i.e.*, $\mathbf{M}' \in \{0,1\}^{H\times W}$. The mask is flattened to be a vector for ease of calculation, as $\boldsymbol{u} \in \mathbb{R}^{HW}$. In order to compress $\boldsymbol{u}$ into $\boldsymbol{v}$, we seek a transformation under some criterion to minimize the reconstruction error between $\boldsymbol{u}$ and $\boldsymbol{v}$. Although many approaches can be used for our purpose here, we observe that the simple linear projecting can already perform well in our experiment. In particular, we have,

$$\boldsymbol{v} = \mathbf{T}\boldsymbol{u}; \quad \tilde{\boldsymbol{u}} = \mathbf{W}\boldsymbol{v}. \tag{1}$$

Here $\mathbf{T} \in \mathbb{R}^{N\times HW}$ is the project matrix, used to compress $\boldsymbol{u}$ into $\boldsymbol{v}$. $\boldsymbol{u}$ can be recovered with the reconstruction matrix $\mathbf{W} \in \mathbb{R}^{HW\times N}$. Note that, $\boldsymbol{u}$ is centered by subtracting its mean over the training set, followed with normalization. Finally, we obtain these matrices by minimizing the reconstruction error between $\boldsymbol{u}$ and $\tilde{\boldsymbol{u}}$ on the training set. Mathematically it is written as Eq. (2):

$$\begin{aligned} \mathbf{T}^*, \mathbf{W}^* &= \arg\min_{\mathbf{T},\mathbf{W}} \sum_{\boldsymbol{u}} \|\boldsymbol{u} - \tilde{\boldsymbol{u}}\|^2 \\ &= \arg\min_{\mathbf{T},\mathbf{W}} \sum_{\boldsymbol{u}} \|\boldsymbol{u} - \mathbf{W}\mathbf{T}\boldsymbol{u}\|^2 \end{aligned} \tag{2}$$

We follow the strategy in DUpsampling [28] and optimize this objective by using principal component analysis (PCA). The overall process is illustrated in Figure 3. Please refer to DUpsampling [28] for details. There may be alternative options to minimize the reconstruction loss, *e.g.*, sparse coding or non-linear auto-encoder.

Mask Reconstruction Given the predicted representation vector $\hat{\boldsymbol{v}} \in \mathbb{R}^N$, the two-dimensional mask $\mathbf{M}' \in \mathbb{R}^{H\times W}$ can be reconstructed through Eq. (1) (right). As we employ this operation after non-maximum suppression (the highest scoring 100 samples), the computation cost of such matrix multiplication is negligible.

Loss Function We define our mask loss function as follows:

$$\mathcal{L}_{mask} = \mathbb{1}^{obj} \sum_{i}^{N} d_{mask}(\hat{y}_i, y_i), \tag{3}$$

where $\mathbb{1}^{obj}$ is the indicator function for positive samples. $\hat{y}_i$, y_i denotes the i-th element in prediction and ground-truth vectors, respectively. In our implementation, we have compared different forms of $d_{mask}(\cdot,\cdot)$, *e.g.*, l_1 loss, smooth-l_1 loss, l_2 loss and cosine similarity loss. Finally, we employ l_2 loss for its effectiveness and stability in training. We append it to the overall loss, formally,

$$\mathcal{L} = \lambda_{det} \cdot \mathcal{L}_{det} + \lambda_{mask} \cdot \mathcal{L}_{mask}. \tag{4}$$

Here $\mathcal{L}_{det}$ is the loss for detection, consisting of $\mathcal{L}_{cls}$ for classification, $\mathcal{L}_{reg}$ for bounding box regression and $\mathcal{L}_{cen}$ for center-ness. In particular, $\mathcal{L}_{cls}$ is focal loss as in [19], $\mathcal{L}_{reg}$ is the GIoU loss following FCOS [29]. $\mathcal{L}_{cen}$ denotes the binary cross entropy (BCE) loss for center-ness. All the balance weights in $\mathcal{L}_{det}$ are set to 1 for simplicity in our experiments.

3.3. Correlation Between Boxes and Masks

In general, instance segmentation and object detection are inseparable in detection-driven pipelines. Intuitively,

[1] We use the improved version, including sharing the features between center-ness and regression branch, central sampling and so on. Please refer to [29] for further details.

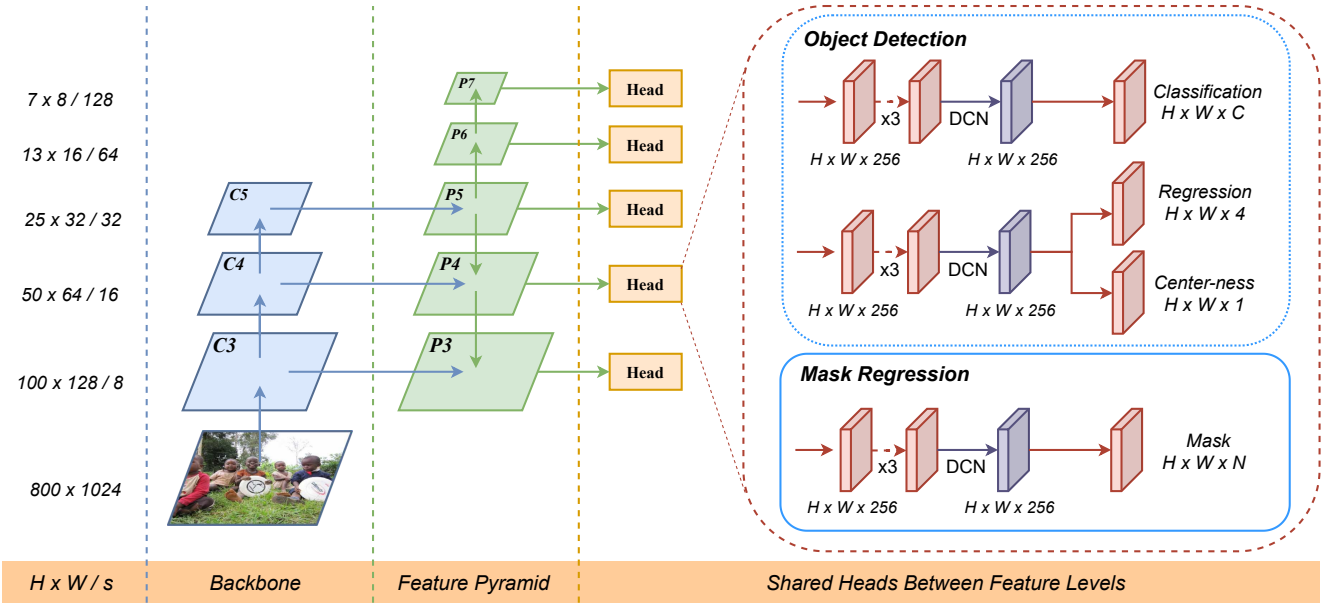

Figure 2: The overall architecture of MEInst, which extends FCOS [29] with a Mask Regression Branch. The model mainly consists of four modules: (a) Backbone for feature extraction. (b) Feature Pyramid. (c) Detection Heads for object detection. (d) Mask Regression Branch for instance segmentation. MEInst detects objects and predicts their mask vectors simultaneously, in which the first three processes are consistent with FCOS. Then the instance masks are reconstructed efficiently through Eq. (1) (right). Here DCN denotes deformable convolution, which is optional (best viewed in color), and N means the dimension of representation vectors (*e.g.*, $N = 60$).

Detector	AP	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{bb}_{S}	AP^{bb}_{M}	AP^{bb}_{L}
Mask-R-50-FPN	34.2	37.8	59.3	41.1	21.5	41.1	49.9
FCOS-R-50-FPN	34.1(−0.1)	38.7(+0.9)	57.3(−2)	41.9(+0.8)	22.6(+1.1)	42.4(+1.3)	50.1(+0.2)
Mask-R-101-FPN	35.7	40.1	61.7	44.0	23.1	43.4	52.7
FCOS-R-101-FPN	36.6(+0.9)	42.9(+2.8)	61.8(+0.1)	46.3(+2.3)	27.4(+4.3)	46.9(+3.5)	55.4(+2.7)
Mask-X-101-32x8d-FPN	36.9	42.2	63.9	46.1	25.4	46.1	54.7
FCOS-X-101-32x8d-FPN	37.1(+0.2)	44.0(+1.8)	63.2(−0.7)	47.6(+1.5)	27.5(+2.1)	47.6(+1.5)	56.4(+1.7)

Table 1: Comparisons among different algorithms on the COCO `val2017` split. The first row shows Mask R-CNN [13] trained by He *et al.*, while the other is FCOS [29] with the same backbone network. We only employ them to detect objects, as for Mask R-CNN, we discard the mask outputs. AP indicates the performance of instance segmentation, which is predicted by the same model with different pre-detected boxes. The gap between two detectors are highlighted by **green** and **red**, respectively. **green** means **better** and **red** **worse**.

better bounding boxes improves the overall performance in the mask branch. Here we carry out several experiments to validate our assumptions empirically.

Take Mask R-CNN [13] as an example. The inference flow is as follows: 1) A backbone module is used to extract semantic feature from the input image. 2) The extracted feature is then sent to the following modules for classification and object regression. 3) Afterwards, the mask stage computes features using ROIAlign from each detected box. 4) Finally, the regional representation is performed pixel-wise segmentation. It only predicts a binary mask.

In our experiments, the Mask-R-50-FPN model pre-trained by He *et al.* is used as the main backbone. The *step-2* in the above process is replaced with a series of pre-acquired detection results predicted by different detectors, in which case all the variables are kept the same except the boxes. Here we choose Mask R-CNN [13] (two-stage) and FCOS [29] (one-stage) with different backbones as object detectors. In the sequel, AP means *mask* AP and *box* AP is denoted as AP^{bb}. The quantitative results are shown in Table 1 and Figure 4.

As for the same architecture, the detector brings consistent and noticeable gain in mask when the network goes deeper. However, the results of instance segmentation fall below our expectations with different pipelines. Compared with Mask R-CNN, FCOS achieves better detection performances among all backbones under the metric AP^{bb}, measuring 0.9%, 2.8%, 1.8%, respectively. Nevertheless, the corresponding segmentation has not been witnessed equivalent improvement, and even performs worse (34.1% *vs.* 34.2%). It seems counter-intuitive.

We observe that FCOS performs better under all the gen-

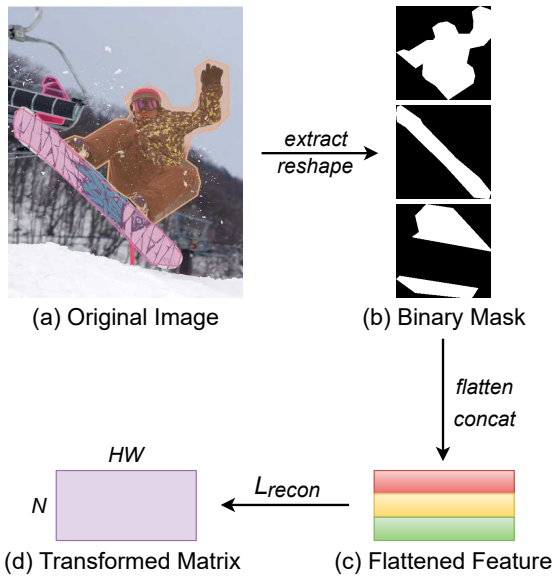

Figure 3: The pipeline of mask encoding. (a) is the original image annotated with instance labels. We extract these annotations and reshape them as (b) $m \times m$ mask (here mask is class-agnostic). Then (c) the flattened feature is compressed for dimensionality reduction. Finally we harvest (d) transformed matrix for mask encoding. The entire procedure is done off-line and it performs very fast. After learning, we freeze all these parameters during network training and inference.

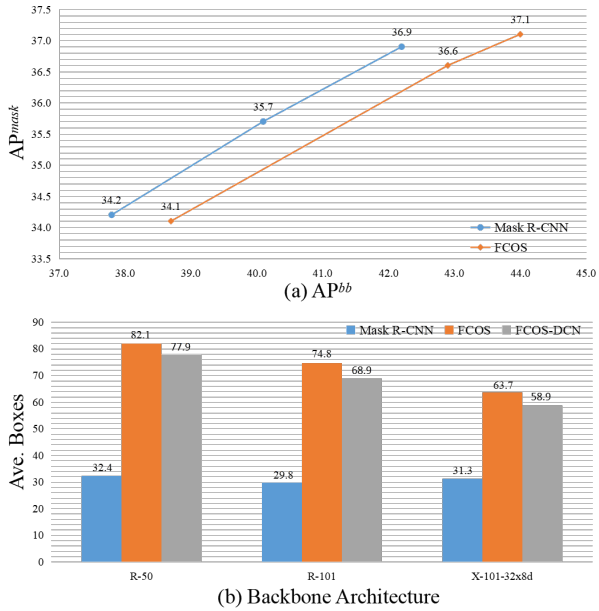

Figure 4: Quantitative analysis of different paradigms on the COCO `val2017` split. (a) AP^{bb} *vs.* AP^{mask}, which shows the correlation between box and mask. As for the same pipeline, better detectors lead to better performances in instance mask. However, this is not the case for FCOS, whose overall detection result is better than the corresponding Mask R-CNN. But FCOS only performs similar or even worse in instance segmentation. (b) Backbone architecture *vs.* Average number of boxes per image: Compared with Mask R-CNN, FCOS outputs more than 2 times more boxes, resulting in lower AP^{bb}_{50}. The phenomenon can be alleviated with a larger receptive field.

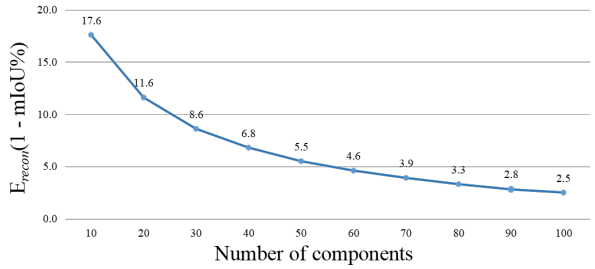

Figure 5: The reconstruction error E_{recon} *vs.* Number of components to keep on COCO `train2017` split.

eral metrics except AP^{bb}_{50}, which indicates that the boxes predicted by FCOS are location-accurate but with more false-positive (FP). Figure 4(b) shows the average number of bounding boxes predicted by different models. FCOS predicts significantly more bounding boxes than Mask R-CNN with the same confidence threshold (*e.g.*, 0.05), which may degrade the performance under the metric AP^{bb}_{50}. Mask R-CNN employs a two-stage pipeline, *i.e.*, first proposes candidates and then refines the boxes, in which case most mis-proposed boxes can be filtered out effectively. However, one-stage paradigm such as FCOS outputs results directly for faster inference, resulting in the redundant boxes. Actually almost all the one-stage methods [19, 23, 27] suffer from this dilemma.

We hypothesize that the issue may be related to the effective receptive field (ERF). Zhou *et al*. [34] declare that the effective receptive field is much smaller than the theoretical receptive field, since CNN tends to capture information from central regions. The insufficient ERF may lead to many false-positive (FP) boxes as the network can not "see" the objects. To tackle this issue, we simply employ deformable convolution [36] that has the capacity to focus on salient regions and enlarge the ERF to some extent. Specifically, we replace the last vanilla convolutional layer in multi-head branches respectively. Note that other modules such as dilated convolution [6] and Large Kernel [25], which are beneficial to ERF, may also boost the performance. We provide further comparisons in the experimental section.

4. Experiments

Our experiments are conducted on the challenging MS COCO benchmark [20] using the standard metrics for instance segmentation. All models are trained on the COCO `train2017` split (∼118k images) and evaluated

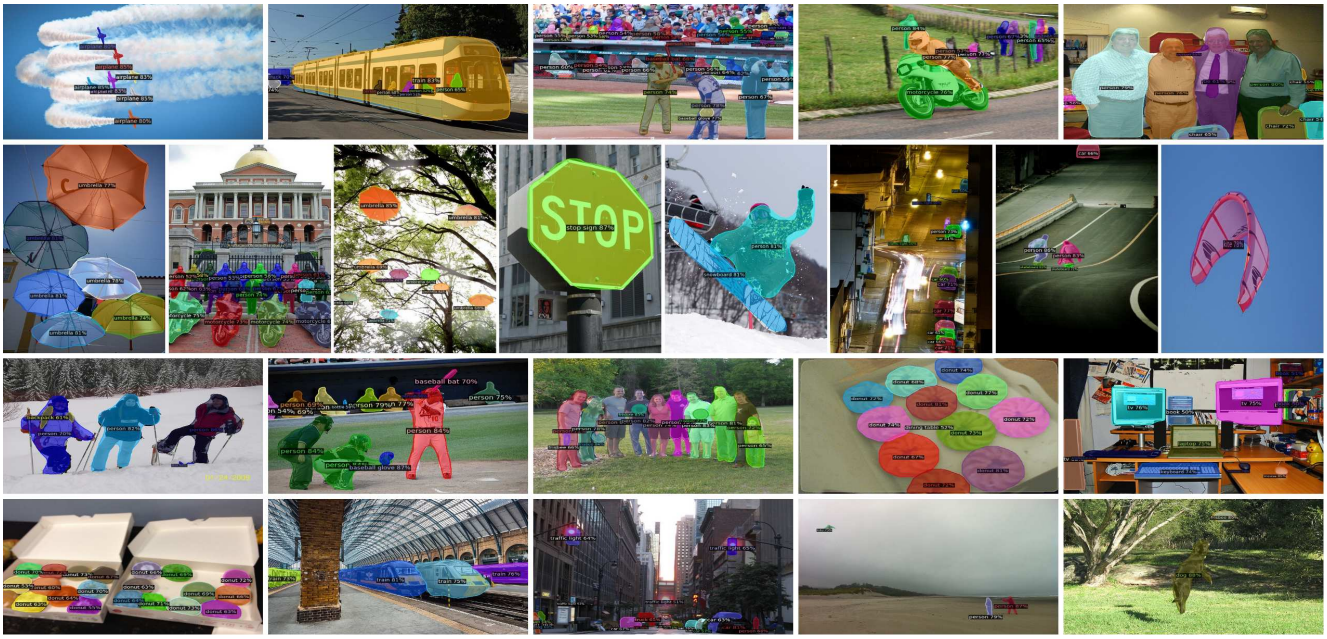

Figure 6: Visualization of MEInst **on COCO images** with ResNeXt-101-FPN, achieving 36.4% mask AP (Table 8).

with `val2017` (5k images). The final results are reported on `test-dev` (20k images). Moreover, we adopt the $1\times$ training strategy [5, 12], single scale training and testing unless otherwise specified.

Training Details ResNet-50 [14] is used as the backbone network and all hyper-parameters are kept consistent with FCOS [29] unless specified. Specifically, we use the stochastic gradient descent (SGD) optimizer, weight decay 0.0001, momentum 0.9 with 90K iterations in all. The initial learning rate is set to 0.01 and divided by 10 at iteration 60K and 80K, respectively. We use a mini-batch of 16 images and all models are trained with 8 GPUs. The backbone is initialized with the pre-trained weights on ImageNet [10] and other newly added layers are initialized as in [19]. The shorter side of images is fixed as 800 pixels with the longer side being 1333 or less. Moreover, we sum up all the losses directly, *i.e.*, $\lambda_{det} = \lambda_{mask} = 1$ in Eq. (4). We expect that the performance may be better with a careful parameter tuning.

Inference Details The inference process is kept the same as FCOS since we only append one more prediction to the predicted boxes. An input image goes through the network and then predicts boxes with several attributes, such as categories and mask coefficients. We peform mask reconstruction after non-maximum suppression (NMS) to avoid unnecessary computational overhead (the highest scoring 100 samples). Since the matrix multiplication is fast, MEInst introduces slight overhead to its FCOS counterpart.

4.1. Ablation Study

Analysis of Upper Bound We first reshape all the annotations into 28×28 binary-class masks. Afterwards, these masks are encoded and recovered to two-dimensional matrices with Eq. (1). Finally we use the metric of *mIoU* to evaluate the quality of reconstructed masks. The reconstruction error on the COCO `train2017` split is shown in Figure 5. It is evident that the reconstruction error goes down consistently with the increase of the number of components kept, and can even reach an extremely low level when the dimension goes to 100 (only 2.5%). Moreover, we observe that the class-agnostic matrix achieves a similar result to class-specific one (up to C times in dimensions). Thus, the former is a better choice for memory-conserving consideration.

Dimension of Encoding Representation It plays a very fundamental role in MEInst. As shown in Table 2, the performance grows steadily with the increase of dimension and reaches saturation at last. For example, there is an improvement of 2% from 20 to 60 and it remains stable beyond 60. The reconstruction has a great influence at the beginning. However, when adequate components can reconstruct the mask well, it is no longer the main factor constraining the performance. We choose $N = 60$ in our experiments unless otherwise specified.

Learning without Explicit Encoding Alternatively, the mask can be learned without explicit encoding. That is, instead of compressing the redundant label into a fix-dimensional vector, we recover the predicted mask with the reconstruction matrix $\mathbf{W}$ and perform pixel-wise classification on it. This projecting process is essentially identical to

N	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
20	29.8	52.4	30.2	14.5	32.0	43.0
40	31.4	53.3	32.5	14.6	34.0	44.9
60	31.8	**53.9**	**32.9**	**15.9**	34.2	**45.7**
80	**31.9**	**53.9**	32.6	15.4	**34.4**	45.5

Table 2: Number of components: MEInst attains consistent gain with more components and reaches saturation at last.

employing a 1×1 convolution along the spatial dimensions, with convolutional kernels stored in $\mathbf{W}$. Note that these parameters are frozen during training. Moreover, we also explore the potential of learning without mask encoding, *i.e.*, the network straightly outputs the high-dimensional masks (*e.g.*, $28 \times 28 = 784$). The results are shown in Table 3. The over-high dimension makes it hard to optimize, resulting in a performance drop. Particularly, AP_{75} and AP_L decrease considerably, measuring by 1.3% and 2.0%, respectively. The relatively compact vector is not only for faster inference, but also beneficial for optimization. With the same dimension, our method still performs better under all the metrics, which further proves the effectiveness of mask encoding.

Loss Function As discussed above, mask encoding converts the task of instance segmentation into a set of coefficient regression problems. We try several popular losses in our experiments to supervise the regression problems, more specifically, smooth-l_1 loss, l_1 loss and l_2 loss. λ_{mask} in Eq. (4) is set to 1 for simplicity. As shown is Table 4, l_2 loss performs better than others. We also consider the case to view the mask vector as a whole, so we apply cosine similarity loss. However, the performance goes worse, which indicates that mask encoding has already eased the redundancy in original representation, and now the elements in vectors are independent.

Large Receptive Field Here we demonstrate the importance of large receptive field. Firstly, we apply large kernel [25] (LK) in the mask prediction layer. The LK layer is a combination of $1 \times k + k \times 1$ and $k \times 1 + 1 \times k$ convolutions. k is set to 9 in our experiments. Compared with 3×3 convolution, it introduces negligible overhead. As shown in Table 5, LK in prediction layer achieves 0.7% AP gains. We also explore the potential of deformable convolution (DCN). Specifically, we only use it in the last layer of head to keep our model efficient. With the ability of capturing more meaningful and larger receptive features, it obtains 1.5% improvement in AP.

Learning Masks boosts Object Detection As mentioned in [11], learning with instance mask prediction can usually boost the performance of one-stage detectors. We also find the similar phenomenon in our experiments, *i.e.*, our MEInst outperforms FCOS [29] by 0.8% AP in box, as demonstrated in Table 6. Compared with RetinaMask [11] which employs a few tricks, our method is simpler yet achieving the same performance.

encoding	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
✓	**31.8**	**53.9**	**32.9**	**15.9**	**34.2**	**45.7**
–	30.8	53.3	31.6	14.5	33.1	43.7
w/o	29.7	52.7	29.9	14.5	32.0	43.4

Table 3: Mask encoding: Learning with mask encoding achieves a better performance. Note that, the difference between "–" and "w/o" is that, the former one leverages implicit mask encoding, while the other does not.

loss	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
smooth l_1	30.8	53.2	31.5	14.8	33.0	44.7
l_1	31.4	53.4	32.4	15.3	33.8	44.8
l_2	**31.8**	**53.9**	**32.9**	**15.9**	**34.2**	**45.7**
cosine	28.9	51.1	29.1	13.1	30.5	42.8

Table 4: Different loss functions: smooth l_1, l_1 and l_2 loss functions show no significant difference, and l_2 works slightly better.

larger?	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
	30.3	53.0	31.1	14.2	33.2	43.4
LK	31.0	52.7	31.9	14.7	33.8	44.5
DC	**31.8**	**53.9**	**32.9**	**15.9**	**34.2**	**45.7**

Table 5: Large receptive field matters: Improving performance with a larger receptive field.

w/mask	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{bb}_S	AP^{bb}_M	AP^{bb}_L
	39.6	58.2	42.7	22.5	43.4	52.1
✓	**40.4**	**58.5**	**43.5**	**24.5**	**43.8**	**52.7**

Table 6: Learning mask boosts object detection: The performance of detection is advanced by multi-task learning.

Scale	Method	AP	AP_{50}	AP_{75}	FPS
416	ESE-Seg [33]	21.6	48.7	22.4	**38.5**
400	MEInst	23.9	42.4	24.1	28.2
600	MEInst	28.4	49.3	28.8	18.5
800	MEInst	**30.3**	**53.0**	**31.1**	12.8

Table 7: Mask-Based *vs*. Contour-Based: MEInst outperforms ESE-Seg [33] by a large margin. All models are based on ResNet-50 and the FPS is reported on GTX 1080Ti.

Mask-Based *vs*. Contour-Based We compare MEInst against the recent contour-based method termed ESE-Seg [33]. To make this a fair comparison, we do not apply any deformable convolutions in our model. As shown in Table 7, MEInst shows a large gain compared to the ESE-Seg method. Additionally, when the input scale becomes smaller (*e.g.*, 400), our model still achieves a better performance at a real-time speed. Note that we do not specifically train a new model here. It indicates that MEInst can not only achieve good performance in mask AP, but also shows promises for real-time applications. Besides the performance, our mask-based method also shows a detail-preserving advantage that ESE-Seg lacks, which is illustrated in Figure 1. Experiments demonstrate that the pro-

Method	Backbone	epochs	aug.	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage									
MNC [9]	ResNet-101-C4	12	–	24.6	44.3	24.8	4.7	25.9	43.6
FCIS [17]	ResNet-101-C5-dilated	12	–	29.2	49.5	–	7.1	31.3	50.0
Mask R-CNN [13]	ResNeXt-101-FPN	12	–	37.1	60.0	39.4	16.9	**39.9**	**53.5**
One-stage									
ExtremeNet [35]	Hourglass-104	100	✓	18.9	44.5	13.7	10.4	20.4	28.3
TensorMask [7]	ResNet-101-FPN	72	✓	37.1	59.3	39.4	17.4	39.1	51.6
YOLACT [3]	ResNet-101-FPN	48	✓	31.2	50.6	32.8	12.1	33.3	47.1
PolarMask [32]	ResNet-101-FPN	12	–	30.4	51.9	31.0	13.4	32.4	42.8
PolarMask [32]	ResNeXt-101-FPN	12	–	32.9	55.4	33.8	15.5	35.1	46.3
MEInst	ResNet-101-FPN	12	–	33.0	56.4	34.0	15.2	35.3	46.3
MEInst	ResNeXt-101-FPN	12	–	35.5	59.7	36.7	17.5	38.0	49.0
MEInst	ResNet-101-FPN-DCN	12	–	34.9	58.8	36.0	16.3	37.0	49.6
MEInst	ResNeXt-101-FPN-DCN	12	–	36.8	**61.6**	38.4	18.1	39.2	51.8
MEInst	ResNet-101-FPN	36	✓	33.9	56.2	35.4	19.8	36.1	42.3
MEInst	ResNeXt-101-FPN	36	✓	36.4	60.0	38.3	21.3	38.8	45.7
MEInst	ResNeXt-101-FPN-DCN	36	✓	**37.8**	61.4	**40.0**	**21.8**	39.8	48.8

Table 8: Instance segmentation *mask* AP on the COCO `test-dev`. Here "aug." denotes data augmentation, *e.g.*, multi-scale. ✓ means training with "aug."

posed method enjoys desirable properties comparing with contour-based algorithms such as PolarMask [32] and ESE-Seg [33].

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

We evaluate MEInst on COCO `test-dev` and compare our results with some state-of-the-art methods, including both one-stage and two-stage models. The results are shown in Table 8 and Figure 6. Without bells and whistles, MEInst achieves a mask AP of 36.4%, which outperforms most one-stage methods by a large margin. Note that we do not use any tricks in our experiments, *e.g.*, auxiliary semantic segmentation supervision. Our performance may be further improved with those tricks. Moreover, the gap between TensorMask [7] and ours is mainly because 1) Tensormask uses a very long training schedule, as well as 2) bipyramid and aligned representation. Considering that these modules are time- and memory-consuming, we do not plug them into our model.

4.3. Advantages and Limitations

MEInst has the capacity to better deal with "disjointed" objects. An example can be found in Figure 6 (row 3 column 1).

An interesting phenomenon is that, MEInst surpasses Mask R-CNN [13] when the detected object is small (21.3% *vs*. 16.9%) while performs worse when the object becomes larger (45.7% *vs*. 53.5%). We argue that the main reasons are two folds:

- For small objects, the capacity of the single feature vector in our work is not a problem. While in Mask R-CNN, it requires the mask prediction head to label each pixel of a small object, which is challenging when the object is very small. That is why we outperform Mask R-CNN for small objects.
- As for large objects, a compact representation vector is difficult to accommodate all the details of the mask. In this case, non-parametric pixel labelling shows advantages. Additional modules to encode details are needed in this case.

5. Conclusion

In this work, we have introduced a new, simple single-shot instance segmentation framework termed MEInst. Different from previous works that typically solve mask prediction as binary classification in a spatial layout, MEInst represents the mask with a fixed-dimensional and compact vector, and casts the task into a regression task. The reformation allows the challenging task to be solved by appending a parallel regression branch to existing one-stage object detectors. Experimental analyses demonstrate that the proposed framework achieves competitive accuracy and speed among one-stage paradigms. In the future, we will explore the possibility of using other dictionary learning methods for encoding instance masks, and the possibility of applying this idea to other instance recognition tasks.

References

[1] Anurag Arnab and Philip Torr. Pixelwise instance segmentation with a dynamically instantiated network. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 441–450, 2017. 2

[2] Min Bai and Raquel Urtasun. Deep watershed transform for instance segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 5221–5229, 2017. 2

[3] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. YOLACT: real-time instance segmentation. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 9157–9166, 2019. 2, 8

[4] Hao Chen, Kunyang Sun, Zhi Tian, Chunhua Shen, Yongming Huang, and Youliang Yan. BlendMask: Top-down meets bottom-up for instance segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2020. 2

[5] 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

[6] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(4):834–848, 2017. 5

[7] Xinlei Chen, Ross Girshick, Kaiming He, and Piotr Dollár. Tensormask: A foundation for dense object segmentation. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 2061–2069, 2019. 2, 8

[8] 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 *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 3213–3223, 2016. 2

[9] Jifeng Dai, Kaiming He, Yi Li, Shaoqing Ren, and Jian Sun. Instance-sensitive fully convolutional networks. In *Proc. Eur. Conf. Comp. Vis.*, pages 534–549. Springer, 2016. 2, 8

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 248–255, 2009. 6

[11] Cheng-Yang Fu, Mykhailo Shvets, and Alexander C. Berg. RetinaMask: Learning to predict masks improves state-of-the-art single-shot detection for free. *arXiv preprint arXiv:1901.03353*, 2019. 7

[12] Ross Girshick, Ilija Radosavovic, Georgia Gkioxari, Piotr Dollár, and Kaiming He. Detectron, 2018. 6

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 2961–2969, 2017. 1, 2, 4, 8

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 770–778, 2016. 3, 6

[15] Zhaojin Huang, Lichao Huang, Yongchao Gong, Chang Huang, and Xinggang Wang. Mask scoring R-CNN. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 6409–6418, 2019. 1, 2

[16] Alexander Kirillov, Evgeny Levinkov, Bjoern Andres, Bogdan Savchynskyy, and Carsten Rother. Instancecut: from edges to instances with multicut. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 5008–5017, 2017. 2

[17] Yi Li, Haozhi Qi, Jifeng Dai, Xiangyang Ji, and Yichen Wei. Fully convolutional instance-aware semantic segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 2359–2367, 2017. 1, 8

[18] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 2117–2125, 2017. 3

[19] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 2980–2988, 2017. 2, 3, 5, 6

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Proc. Eur. Conf. Comp. Vis.*, pages 740–755. Springer, 2014. 1, 2, 5

[21] Shu Liu, Jiaya Jia, Sanja Fidler, and Raquel Urtasun. Sgn: Sequential grouping networks for instance segmentation. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 3496–3504, 2017. 2

[22] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 8759–8768, 2018. 1, 2

[23] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander Berg. SSD: Single shot multibox detector. In *Proc. Eur. Conf. Comp. Vis.*, pages 21–37. Springer, 2016. 5

[24] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 3431–3440, 2015. 1

[25] Chao Peng, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. Large kernel matters–improve semantic segmentation by global convolutional network. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 4353–4361, 2017. 5, 7

[26] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 779–788, 2016. 2

[27] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 7263–7271, 2017. 5

[28] Zhi Tian, Tong He, Chunhua Shen, and Youliang Yan. Decoders matter for semantic segmentation: Data-dependent decoding enables flexible feature aggregation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 3126–3135, 2019. 3

[29] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: Fully convolutional one-stage object detection. In *Proc. IEEE Int. Conf. Comp. Vis.*, 2019. 2, 3, 4, 6, 7

[30] Xinlong Wang, Tao Kong, Chunhua Shen, Yuning Jiang, and Lei Li. SOLO: Segmenting objects by locations. *arXiv preprint*, 2019. 3

[31] Xinlong Wang, Rufeng Zhang, Tao Kong, Chunhua Shen, and Lei Li. SOLOv2: Dynamic, faster and stronger. *arXiv preprint*, 2019. 3

[32] Enze Xie, Peize Sun, Xiaoge Song, Wenhai Wang, Xuebo Liu, Ding Liang, Chunhua Shen, and Ping Luo. PolarMask: Single shot instance segmentation with polar representation. *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2020. 1, 2, 8

[33] Wenqiang Xu, Haiyang Wang, Fubo Qi, and Cewu Lu. Explicit shape encoding for real-time instance segmentation. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 5168–5177, 2019. 1, 2, 7, 8

[34] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. *arXiv preprint arXiv:1412.6856*, 2014. 5

[35] Xingyi Zhou, Jiacheng Zhuo, and Philipp Krahenbuhl. Bottom-up object detection by grouping extreme and center points. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 850–859, 2019. 2, 8

[36] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 9308–9316, 2019. 5