Dense RepPoints: Representing Visual Objects with Dense Point Sets

Ze Yang[1,2†⋆], Yinghao Xu[3,4†*], Han Xue[5†*], Zheng Zhang[7]
Raquel Urtasun[6], Liwei Wang[1], Stephen Lin[7], and Han Hu[7]

[1] Peking University
[2] Zhejiang Lab
yangze@pku.edu.cn, wanglw@cis.pku.edu.cn
[3] Zhejiang University
[4] The Chinese University of Hong Kong
justimyhxu@gmail.com
[5] Shanghai Jiao Tong University
xiaoxiaoxh@sjtu.edu.cn
[6] University of Toronto
urtasun@cs.toronto.edu
[7] Microsoft Research Asia
{zhez,stevelin,hanhu}@microsoft.com

Abstract. We present a new object representation, called Dense RepPoints, that utilizes a large set of points to describe an object at multiple levels, including both box level and pixel level. Techniques are proposed to efficiently process these dense points, maintaining near-constant complexity with increasing point numbers. Dense RepPoints is shown to represent and learn object segments well, with the use of a novel distance transform sampling method combined with set-to-set supervision. The distance transform sampling combines the strengths of contour and grid representations, leading to performance that surpasses counterparts based on contours or grids. Code is available at https://github.com/justimyhxu/Dense-RepPoints.

1 Introduction

Representation matters. While significant advances in visual understanding algorithms have been witnessed in recent years, they all rely on proper representation of visual elements for convenient and effective processing. For example, a single image feature, a rectangular box, and a mask are usually adopted to represent input for recognition tasks of different granularity, i.e. image classification [23, 19, 38], object detection [16, 36, 28] and pixel-level segmentation [18, 26, 7], respectively. In addition, the representation at one level of granularity may help the recognition task at another granularity, e.g. an additional mask representation may aid the learning of a coarser recognition task such as object detection [18].

⋆ Equal contribution. †This work was done when Ze Yang, Yinghao Xu and Han Xue were interns at Microsoft Research Asia.

We thus consider the question of whether a unified representation for recognition tasks can be devised over various levels of granularity.

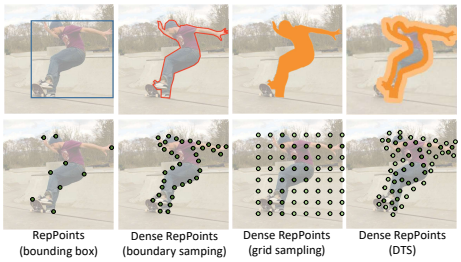

Fig. 1. Visual object in different geometric forms (top row from left to right): bounding box, boudary sampling(Contour), Grid sampling(binary mask), Distance transform sampling. These various object forms can be unified represented by a dense point set, called *Dense RepPoints* (bottom row).

Recently, RepPoints [47] was proposed to represent an object by a small set of adaptive points, simultaneously providing a geometric and semantic description of an object. It demonstrates good performance for the coarse localization task of object detection, and also shows potential to conform to more sophisticated object structures such as semantic keypoints. However, the small number of points (9 by default) limits its ability to reveal more detailed structure of an object, such as pixel-level instance segmentation. In addition, the supervision for recognition and coarse localization also may hinder learning of more fine-grained geometric descriptions.

This paper presents *Dense RepPoints*, which utilizes a large number of points along with optional attributes to represent objects in detail, e.g. for instance segmentation. Because of its high representation flexibility, *Dense RepPoints* can effectively model common object segment descriptors, including contours (polygon) [21, 29, 45] and grid masks [18, 7], as illustrated in columns 2 and 3 of Figure 1. *Dense RepPoints* can also model a *binary boundary mask*, a new geometric descriptor for object segments that combines the description efficiency of contours and the reduced dependence on exact point localization of grid masks, as illustrated in column 4 of Figure 1.

To learn and represent binary boundary masks by *Dense RepPoints*, three techniques are proposed. The first is a distance transform sampling (DTS) method, which converts a ground-truth boundary mask into a point set by probabilistic sampling based on the distance transform map of the object contour. With this conversion, the *Dense RepPoints* prediction and ground truth are both point sets and are thus comparable. The second is a set-to-set supervision loss, in contrast to the commonly used point-to-point supervision loss, e.g. [45, 33]. The set-to-set supervision loss avoids assigning exact geometric meaning for every point, which is usually difficult and semantically inaccurate for instance segmentation but is required by point-to-point methods. The third is a novel conversion method from the learnt non-grid *Dense RepPoints* to an instance mask of any resolution, based on Delaunay triangulation.

With these three novel techniques, *Dense RepPoints* are learnt to well represent the binary boundary map of objects. It also yields better performance than methods based on a contour or grid mask representation. The method achieves 39.1 mask mAP and 45.6 box mAP on the COCO test-dev set using a ResNet-101 backbone network.

In addition to greater representation ability and better accuracy, Dense RepPoints can also be efficiently processed with our proposed techniques. The complexity of vanilla RepPoints increases linearly with the number of points, making it impractical for large point sets, e.g. hundreds of points. To resolve this issue, we propose two techniques, *group pooling* and *shared offset / attribute field*, for object classification and offset / attribute prediction, respectively. These techniques enable near-constant complexity with increasing numbers of points, while maintaining the same accuracy.

The contributions of this work are summarized as follows:

- We propose a new object representation, called *Dense RepPoints*, that models objects by a large number of adaptive points. The new representation shows great flexibility in representing detailed geometric structure of objects. It also provides a unified object representation over different levels of granularity, such as at the box level and pixel level. This allows for coarse detection tasks to benefit from finer segment annotations as well as enable instance segmentation, in contrast to training through separate branches built on top of base features as popularized in [10, 18].
- We adapt the general *Dense RepPoints* representation model to the instance segmentation task, where three novel techniques of *distance transform sampling* (DTS), *set-to-set supervision loss* and *Delaunay triangulation based conversion* are proposed. *Dense RepPoints* is found to be superior to previous methods built on a contour or grid mask representation.
- We propose two techniques, of *group pooling* and *shared offset / attribute fields*, to efficiently process the large point set of *Dense RepPoints*, yielding near constant complexity with similar accuracy.

2 Related Work

Bounding box representation. Most existing high-level object recognition benchmarks [14, 29, 24] employ bounding box annotations for object detection. The current top-performing two-stage object detectors [17, 16, 36, 11] use bounding boxes as anchors, proposals and final predictions throughout their pipelines. Some early works have proposed to use rotated boxes [20] to improve upon axis-aligned boxes, but the representation remains in a rectangular form. For other high-level recognition tasks such as instance segmentation and human pose estimation, the intermediate proposals in top-down solutions [10, 18] are all based on bounding boxes. However, the bounding box is a coarse geometric representation which only encodes a rough spatial extent of an object.

Non-box object representations. For instance segmentation, the annotation for objects is either as a binary mask [14] or as a set of polygons [29]. While most current top-performing approaches [9, 18, 6] use a binary mask as final predictions, recent approaches also exploit contours for efficient interactive annotation [4, 1, 30] and segmentation [8, 45]. This contour representation, which was popular earlier in computer vision [21, 5, 39–41], is believed to be more compatible

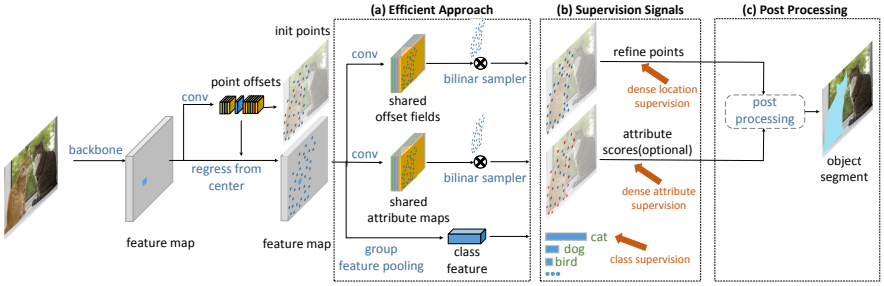

Fig. 2. Overview of *Dense RepPoints*. First, the initial representative points are generated by regressing from the center point as in RepPoints [47]. Then, these initial points are refined by the proposed efficient approaches to obtain refined, attributed representative points. Finally, post-processing is applied to generate the instance segment.

with the semantic concepts of objects [32, 40]. Some works also use edges and superpixels [46, 22] as object representations. Our proposed *Dense RepPoints* has the versatility to model objects in several of these non-box forms, providing a more generalized representation.

Point set representation. There is much research focused on representing point clouds in 3D space [34, 35]. A direct instantiation of ordered point sets in 2D perception is 2D pose [43, 3, 2], which directly addresses the semantic correspondence problem. Recently, there has been increasing interest in the field of object detection on using specific point locations, including corner points [25], extreme points [50], and the center point [49, 13]. These point representations are actually designed to recover a bounding box, which is coarse and lacks semantic information. RepPoints [47] proposes a learnable point set representation trained from localization and recognition feedback. However, it uses only a small number ($n = 9$) of points to represent objects, limiting its ability to represent finer geometry. In this work, we extend RepPoints [47] to a denser and finer geometric representation, enabling usage of dense supervision and taking a step towards dense semantic geometric representation.

3 Methodology

In this section, we first review *RepPoints* [47] for object detection in Sec. 3.1. Then, we introduce *Dense RepPoints* in Sec. 3.2 for strengthening the representation ability of *RepPoints* from object detection to fine-grained geometric localization and recognition tasks, such as extracting an instance mask, by associating an attribute vector with each representative point. In addition, these fine-grained tasks usually require higher resolution and many more representative points than object detection, which makes the computational complexity of vanilla *RepPoints* infeasible. We discuss how to reduce the computational complexity of vanilla *RepPoints* in Sec. 3.3 for representing an instance mask. In Sec. 3.4, we describe how to use *Dense RepPoints* to model instance masks with

different sampling strategies, and then design appropriate supervision signals in Sec. 3.5. Since representative points are usually sparse and non-grid while an instance segment is dense and grid-aligned, we discuss how to transform representative points into an instance segment in Sec. 3.6. An overview of our method is exhibited in Fig. 2.

3.1 Review of RepPoints for object detection

We first review how *RepPoints* [47] detects objects. A set of adaptive representative points $\mathcal{R}$ is used to represent an object in RepPoints:

$$\mathcal{R} = \{p_i\}_{i=1}^{n} \tag{1}$$

where $p_i = (x_i + \Delta x_i, y_i + \Delta y_i)$ is the i-th representative point, x_i and y_i denote an initialized location, Δx_i and Δy_i are learnable offsets, and n is the number of points. The feature of a point $\mathcal{F}(p)$ is extracted from the feature map $\mathcal{F}$ through bilinear interpolation, and the feature of a point set $\mathcal{F}(\mathcal{R})$ is defined as the concatenation of all representative points of $\mathcal{R}$:

$$\mathcal{F}(\mathcal{R}) = \text{concat}(\mathcal{F}(p_1), ..., \mathcal{F}(p_n)) \tag{2}$$

which is used to recognize the class of the point set. The bounding box of a point set can be obtained by conversion functions [47]. In the training phase, explicit supervision and annotation for representative points is not required. Instead, representative points are driven to move to appropriate locations by the classification loss and box localization loss:

$$L_{\text{det}} = L_{\text{cls}}^{b} + L_{\text{loc}}^{b} \tag{3}$$

Both bilinear interpolation used in feature extraction and the conversion function used in bounding box transformation are differentiable with respect to the point locations. These representative points are suitable for representing the object category and accurate position at the same time.

3.2 Dense RepPoints

In vanilla *RepPoints*, the number of representative points is relatively small ($n = 9$). It is sufficient for object detection, since the category and bounding box of an object can be determined with few points. Different from object detection, fine-grained geometric localization tasks such as instance segmentation usually provide pixel-level annotations that require precise estimation. Therefore, the representation capacity of a small number of points is insufficient, and a significantly larger set of points is necessary together with an attribute vector associated with each representative point:

$$\mathcal{R} = \{(x_i + \Delta x_i, y_i + \Delta y_i, \mathbf{a}_i)\}_{i=1}^{n}, \tag{4}$$

where $\mathbf{a}_i$ is the attribute vector associated with the i-th point.

In instance segmentation, the attribute can be a scalar, defined as the foreground score of each point. In addition to the box-level classification and localization terms, L^b_{cls} and L^b_{loc}, we introduce a point-level classification loss L^p_{cls} and a point-level localization loss L^p_{loc}. The objective function of Eq. 3 becomes:

$$L = \underbrace{L^b_{\text{cls}} + L^b_{\text{loc}}}_{L_{\text{det}}} + \underbrace{L^p_{\text{cls}} + L^p_{\text{loc}}}_{L_{\text{mask}}} \tag{5}$$

where L^p_{cls} is responsible for predicting the point foreground score and L^p_{loc} is for learning point localization. This new representation is named *Dense RepPoints*.

3.3 Efficient computation

Intuitively, denser points will improve the capacity of the representation. However, the feature of an object in vanilla *RepPoints* is formed by concatenating the features of all points, so the FLOPs will rapidly increase as the number of points increases. Therefore, directly using a large number of points in *RepPoints* is impractical. To address this issue, we introduce group pooling and shared offset fields to reduce the computational complexity, thereby significantly reducing the extra FLOPs while maintaining performance. In addition, we further introduce a shared attribute map to efficiently predict whether a point is in the foreground.

Group pooling. Group pooling is designed to effectively extract object features and is used in the box classification branch (see Figure 3 top). Given n representative points, we equally divide the points into k groups, with each group having n/k points (if k is not divisible by n, the last group will have fewer points than the others to ensure a total of n points). Then, we aggregate the feature of each point within a group by max-pooling to extract a group feature. Finally, a 1×1 convolution is computed over the concatenated group features from all groups. In this way, the object features are represent by groups instead of points, reducing the computational complexity from $O(n)$ to $O(k)$. The groups are driven by the classification target and will learn different semantics for different groups (though no clear geometric separation) to enhance classification power. We empirically find that the number of groups do not need to be increased when the points become denser, thus the computational complexity is not affected by using a larger set of points. In our implementation, we set k to 9 by default, which works relatively well for classification.

Shared offset fields. The computational complexity of predicting the offsets for the points is $O(n^2)$ in *RepPoints*, making the dense point set representation unsuitable for real applications. Unlike in the classification branch, we need the information of individual points for point location refinement. Hence, we cannot directly apply the grouped features used in classification. Instead, we empirically find that local point features provide enough information for point refinement, in the same spirit as Curve-GCN [30] which uses local features for contour refinement. To share feature computation among points, we propose to first compute n shared offset field maps based on the image feature map.

And then for the i-th representative point, its position is directly predicted via bilinear interpolation at the corresponding location of the i-th offset field (see Figure 3 middle). This reduces the computational complexity of the regression from $O(n^2)$ to $O(n)$. By using group pooling and shared offset fields, even if a large number of points are used, the added FLOPs is still very small compared to that of the entire network (see Sec. 4.3).

Shared attribute map. Predicting the foreground score of each point can be implemented in manner similar to *shared offset fields* by using a shared position-sensitive attribute map, first introduced in R-FCN [11]. In the position-sensitive attribute map, each channel has an explicit positional meaning. Therefore, the foreground score of each representative point can be interpolated on the channel corresponding to its location (see Figure 3 bottom).

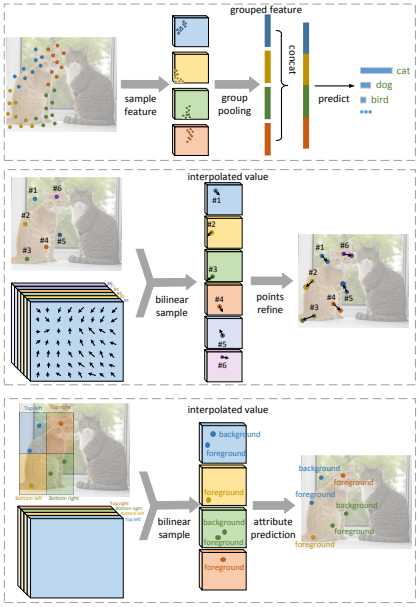

Fig. 3. Illustration of efficient feature extraction for *Dense RepPoints*. **Top:** group pooling operation. **Middle:** shared offset fields for each point index. **Bottom:** shared attribute maps for each relative position.

3.4 Different sampling strategies

How to represent object segments effectively is a core problem in visual perception. Contours and binary masks are two typical representations widely used in previous works [18, 7, 45, 33]. In *Dense RepPoints*, these representations can be simulated by different sampling strategies: a binary mask by uniformly sampling grid points over the bounding box of an object, and a contour as all sampling points along the object boundary. We call these two sampling strategies *grid sampling* and *boundary sampling*, respectively, and discuss them in this section. In addition, we introduce a new sampling strategy, named *distance transform sampling*, which combines the advantages of both grid sampling and boundary sampling.

Boundary sampling (Contour). An instance segment can be defined as the inner region of a closed object contour. Contour points is a compact object description because of its 1-D nature (defined by a sequence of points). In our method, the contour representation can be simulated through supervising the offsets of representative points along the object boundary, with the score of points set to 1 by default.

Grid sampling (Binary Mask). A binary mask can be represented as a set of uniformly sampled grid points over the bounding box of an object, and each

sampled point has a binary score to represent its category, i.e. foreground or background. This sampling strategy (representation) is widely used, such as in Mask R-CNN [18] and Tensor Mask [7]. In *Dense RepPoints*, grid sampling can be implemented by constraining the offsets of representative points as:

$$\Delta x_i = \alpha(\frac{i}{\sqrt{n}} - 0.5), \quad \Delta y_i = \beta(\frac{i}{\sqrt{n}} - 0.5), \quad i \in \{1...n\} \tag{6}$$

where n is the number of sampling points, and α and β are two learnable parameters.

Distance transform sampling (Binary Boundary Mask). Boundary sampling and grid sampling both have their advantages and applications. In general, boundary sampling (contour) is more compact for object segment description, and grid sampling is easier for learning, mainly because its additional attribute (foreground score) avoids the need for precise point localization. To take advantage of both sampling strategies, we introduce a new sampling method called distance transform sampling. In this sampling strategy, points near the object boundary are sampled more and other regions are sampled less. During the training phase, the ground truth is sampled according to distance from the object boundary:

$$\mathcal{P}(p) = \frac{g(\mathcal{D}(p))}{\sum_q g(\mathcal{D}(q))} \tag{7}$$

$$\mathcal{D}(p) = \frac{\min_{e \in \mathcal{E}} \|p - e\|_2}{\sqrt{\max_{e,e' \in \mathcal{E}} |e_x - e'_x| \cdot \max_{e,e' \in \mathcal{E}} |e_y - e'_y|}} \tag{8}$$

where $P(p)$ is the sampling probability of point p, $D(p)$ is the normalized distance from the object boundary of point p, $\mathcal{E}$ is the boundary point set, and g is a decreasing function. In our work, we use the Heaviside step function for g:

$$g(x) = \begin{cases} 1 & x \leq \delta \\ 0 & x > \delta \end{cases} \tag{9}$$

Here, we use $\delta = 0.04$ by default. Intuitively, points with a distance less than δ (close to the contour) have a uniform sampling probability, and points with a distance greater than δ (away from the contour) are not sampled.

3.5 Sampling supervision

The point classification loss L^p_{cls} and the point localization loss L^p_{loc} are used to supervise the different segment representations during training. In our method, L^p_{cls} is defined as a standard cross entropy loss function with softmax activation, where a point located in the foreground is labeled as positive and otherwise its label is negative.

For localization supervision, a point-to-point approach could be taken, where each ground truth point is assigned an exact geometric meaning, e.g. using the

polar assignment method in PolarMask [45]. Each ground truth with exact geometric meaning also corresponds to a fixed indexed representative point in *Dense RepPoints*, and the L2 distance is used as the point localization loss L_{loc}^{p}:

$$L_{point}(\mathcal{R}, \mathcal{R}') = \frac{1}{n} \sum_{k=1}^{n} \|(x_i, y_i) - (x'_i, y'_i)\|_2 \tag{10}$$

where $(x_i, y_i) \in \mathcal{R}$ and $(x'_i, y'_i) \in \mathcal{R}'$ represent the point in the predicted point set and ground-truth point set, respectively.

However, assigning exact geometric meaning to each point is difficult and may be semantically inaccurate for instance segmentation. Therefore, we propose set-to-set supervision, rather than supervise each individual point. The point localization loss is measured by *Chamfer distance* [15, 37] between the supervision point set and the learned point set:

$$L_{set}(\mathcal{R}, \mathcal{R}') = \frac{1}{2n} \left(\sum_{i=1}^{n} \min_{j} \left\|(x_i, y_i) - (x'_j, y'_j)\right\|_2 + \sum_{j=1}^{n} \min_{i} \left\|(x_i, y_i) - (x'_j, y'_j)\right\|_2 \right)$$

where $(x_i, y_i) \in \mathcal{R}$ and $(x'_j, y'_j) \in \mathcal{R}'$. We evaluate these two forms of supervision in Section 4.3.

3.6 Representative Points to Object Segment

Dense RepPoints represents an object segment in a sparse and non-grid form, and thus an extra post-processing step is required to transform the non-grid points into a binary mask. In this section, we propose two approaches, Concave Hull [31] and Triangulation, for this purpose.

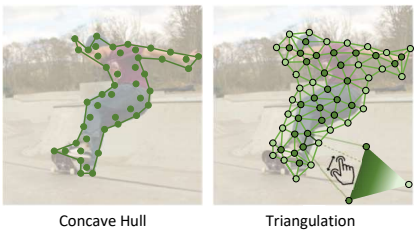

Fig. 4. Post-Processing. Generating image segments by Concave Hull and Triangulation.

Concave Hull. An instance mask can be defined as a concave hull of a set of foreground points (see Figure 4 left), which is used by many contour-based methods. In *Dense RepPoints*, boundary sampling naturally uses this post-processing. We first use a threshold to binarize the predicted points by their foreground scores, and then compute their concave hull to obtain the binary mask. In our approach, we empirically set a threshold of 0.5 by default.

Triangulation. Triangulation is commonly used in computer graphics to obtain a mesh from a point set representation, and we introduce it to generate an object segment. Specifically, we first apply Delaunay triangulation to partition the space into triangles with vertices defined by the learned point set. Then, each pixel in the space will fall inside a triangle and its point score is obtained by linearly interpolating from the triangle vertices in the Barycentric coordinates (Figure 4 right). Finally, a threshold is used to binarize the interpolated score map to obtain the binary mask.

4 Experiments

Fig. 5. Visualization of points and instance masks by DTS. **Top**:The learned points (225 points) is mainly distributed around the mask boundary. **Bottom**: The foreground masks generated by triangulation post-processing on COCO `test-dev` images with ResNet-50 backbone under '3x' training schedule.

4.1 Datasets

We present experimental results for instance segmentation and object detection on the COCO2017 benchmark [29], which contains 118k images for training, 5k images for validation (`val`) and 20k images for testing (`test-dev`). The standard mean average precision (mAP) is used to measure accuracy. We conduct an ablation study on the validation set, and compare with other state-of-the-art methods on the test-dev set.

4.2 Implementation Details

We follow the training settings of *RepPoints* [47]. Horizontal image flipping augmentation, group normalization [44] and focal loss [28] are used during training. If not specified, ResNet-50 [19] with FPN [27] is used as the default backbone in the ablation study, and weights are initialized from the ImageNet [12] pre-trained model. Distance transform sampling with set-to-set supervision is used as the default training setting, and triangulation is chosen as the default post-processing. For predicting attribute scores, we follow SOLO [42] by using seven 3×3 convs in the attribute score head, and the feature map of P3 is used to fuse with feature maps of other levels through addition operation, which is inspired by the conversion FPN used in TensorMask [7],

The models are trained on 8 GPUs with 2 images per GPU for 12 epochs ($1\times$ settings). In SGD training, the learning rate is initialized to 0.01 and then divided by 10 at epochs 8 and 11. The weight decay and momentum parameters are set to 10^{-4} and 0.9, respectively. In inference, we follow SOLO [42] to refine the classification score by using the mask prediction, and we use NMS with IoU threshold of 0.5, following RetinaNet [28].

4.3 Ablation Study

Components for greater efficiency. We validate group pooling (GP) and shared offset fields (SOF) by adding them to vanilla *RepPoints* [47] and evaluating the performance on object detection. Results are shown in Table 1. We

n	G FLOPS			mAP		
	Base	+ GP	+SOF	Base	+ GP	+ SOF
9	211.04	208.03	202.05	38.1	37.9	37.9
25	255.14	237.80	205.93	37.7	37.8	37.7
49	321.28	278.86	209.18	37.7	37.6	37.5
81	409.46	331.03	212.60	37.5	37.5	37.5

Table 1. Validating the proposed components for greater efficiency. With group pooling (GP) and shared offset fields (SOF), the mAP constantly improve as the number of points increase, while the FLOPS is nearly unaffected.

number of points	9	25	81	225	729
Contour	**19.7**	23.9	26.0	25.2	24.1
Grid points	5.0	17.6	29.7	31.6	32.8
DTS	13.9	**24.5**	**31.5**	**32.8**	**33.8**

Table 2. Comparison of different mask representations.

present the results under different numbers of points: $n = 9, 25, 49, 81$. By using group pooling, FLOPs significantly decreases with increasing number of points compared to vanilla *RepPoints* with similar mAP. By introducing shared offset fields, while mAP is not affected, FLOPs is further reduced and nearly constant with respect to n. Specifically, for $n = 81$, our efficient approach saves 197G FLOPs in total. This demonstrates the effectiveness of our efficient approach representation and makes the use of more representative points in instance segmentation possible.

Different sampling strategies. We compare different strategies for sampling object points. Since different sampling strategies perform differently under different post-processing, we compare them with the their best-performing post-processing method for fair comparison. Therefore, we use triangulation (Figure 4 right) for distance transform sampling, bilinear interpolation (`imresize`) for grid sampling, and concave hull (Figure 4 left) for boundary sampling. Please see the Appendix for more details on the post-processing. Results are shown in Table 2. Boundary sampling has the best performance with few points. When $n = 9$, boundary sampling obtains 19.7 mAP, and grid sampling has only 5.0 mAP. Distance transform sampling has 13.9 mAP, which lies in the middle. The reason is that boundary sampling only samples points on the boundary, which is the most efficient way to represent object masks, so relatively good performance can be achieved with fewer points. Both grid sampling and distance transform sampling need to sample non-boundary points, so their efficiency is lower than boundary sampling, but distance transform sampling samples more points around the boundary than in other regions, thus it performs much better than grid sampling.

When using more points, grid sampling and distance transform sampling perform better than boundary sampling. For $n = 729$, grid sampling and distance transform sampling achieve 32.8 mAP and 33.8 mAP, respectively, while boundary sampling only obtains 24.1 mAP. This is due to the limited representation capability of boundary sampling since it only takes boundary points into consideration. In addition, distance transform sampling outperforms grid sampling

# of points	9	25	49	81	225
Concave-Hull	9.7	21.0	21.3	20.6	23.4
Triangulation	13.9	24.5	29.6	31.5	**32.8**

Table 3. Comparison of triangulation and concave hull.

number of points	9	25	81	225	729
point-to-point	10.7	20.7	27.8	31.3	32.6
set-to-set	13.9	24.5	31.5	32.8	**33.8**

Table 4. Comparison of point-to-point and set-to-set supervision.

in all cases, which indicates that distance transform sampling is more efficient than grid sampling while maintaining the same representation capability.

Concave Hull vs. Triangulation. Concave hull and triangulation both can transform a point set to a binary mask. Here, we compare them using distance transform sampling. Results are shown in Table 3. Triangulation outperforms concave hull consistently with different numbers of points, indicating that triangulation is more suitable for distance transform sampling (DTS). It is noted that concave hull with DTS is worse than contour sampling, because DTS does not strictly sample on the boundary but usually samples points near the boundary. Besides, it also samples points farther from the boundary.

Different supervision strategies. Point-to-point is a common and intuitive supervision strategy and it is widely used by other methods [45, 50, 33]. However, this kind of supervision may prevent *Dense RepPoints* from learning better sampling strategies, since it is restrictive and ignores the relationships among points. This motivates the proposed set-to-set supervision in Section 3.5. We compare the two forms of supervision using distance transform sampling. Results are shown in Table 4. Set-to-set supervision consistently outperforms point-to-point supervision, especially for a small number of points.

More representative points. *Dense RepPoints* can take advantage of more points than vanilla *RepPoint* [47], and its computation cost does not change as the number of points increases. Table 5 shows the performance of *Dense RepPoints* on different numbers of points using distance transform sampling and triangulation inference. In general, more points bring better performance, but as the number of points increases, the improvement saturates.

Benefit of Dense RepPoints on detection. Instance segmentation benefits object detection via multi-task learning as reported in Mask R-CNN [18]. Here, we examine whether *Dense RepPoints* can improve object detection performance as well. Results are shown in Table 6. Surprisingly, *Dense RepPoints* not only takes advantage of instance segmentation to strengthen object detection, but also brings greater improvement when more points are used. Specifically, when $n = 81$, *Dense RepPoints* improves detection mAP by 1.9 points. As a comparison, Mask R-CNN improves by 0.9 points compared to Faster R-CNN. This indicates that multi-task learning benefits more from better representation. This suggests that *Dense RepPoints* models a finer geometric representation. This novel application of explicit multi-task learning also verifies the necessity

number of points	81	225	441	729
AP	31.5	32.8	33.3	**33.8**
AP@50	54.2	54.2	54.5	**54.8**
AP@75	32.7	34.4	35.2	**35.9**

Table 5. Results of *Dense RepPoints* on different numbers of points.

	Dense RepPoints				Mask R-CNN
	n=9	n=25	n=49	n=81	
w.o. Inst	37.9	37.7	37.5	37.5	36.4
w. Inst	38.1	38.7	39.2	39.4	37.3
improve	+0.2	+1.0	+1.7	**+1.9**	+0.9

Table 6. Effects of dense supervision on detection.

of using a denser point set, and it demonstrates the effectiveness of our unified representation.

Method	Backbone	epochs	jitter	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Mask R-CNN [18]	ResNet-101	12		35.7	58.0	37.8	15.5	38.1	52.4
Mask R-CNN [18]	ResNeXt-101	12		37.1	60.0	39.4	16.9	39.9	53.5
TensorMask [7]	ResNet-101	72	✓	37.1	59.3	39.4	17.4	39.1	51.6
SOLO [42]	ResNet-101	72	✓	37.8	59.5	40.4	16.4	40.6	54.2
ExtremeNet [50]	HG-104	100	✓	18.9	-	-	10.4	20.4	28.3
PolarMask [45]	ResNet-101	24	✓	32.1	53.7	33.1	14.7	33.8	45.3
Ours*	ResNet-50	12		33.9	55.3	36.0	17.5	37.1	44.6
Ours	ResNet-50	12		34.1	56.0	36.1	17.7	36.6	44.9
Ours	ResNet-50	36	✓	37.6	60.4	40.2	20.9	40.5	48.6
Ours	ResNet-101	12		35.8	58.2	38.0	18.7	38.8	47.1
Ours	ResNet-101	36	✓	39.1	62.2	42.1	21.8	42.5	50.8
Ours	ResNeXt-101	36	✓	40.2	63.8	43.1	23.1	43.6	52.0
Ours	ResNeXt-101-DCN	36	✓	41.8	65.7	45.0	24.0	45.2	54.6

Table 7. Performance of instance segmentation on COCO [29] `test-dev`. Our method significantly surpasses all other state-of-the-arts. '*' indicates training without ATSS [48] assigner and 'jitter' indicates using scale-jitter during training.

4.4 Comparison with other SOTA methods

A comparison is conducted with other state-of-the-arts methods in object detection and instance segmentation on the COCO `test-dev` set. We use 729 representative points by default, and trained by distance transform sampling and set-to-set supervision. ATSS [48] is used as the label assignment strategy if not specified. In the inference stage, the instance mask is generated by adopting triangulation as post-processing.

We first compare with other state-of-the-art instance segmentation methods. Results are shown in Table 7. With the same ResNet-101 backbone, our method achieves 39.1 mAP with the 1x setting, outperforming all other methods. By further integrating ResNeXt-101-DCN as a stronger backbone, our method reaches 41.8 mAP.

We then compare with other state-of-the-arts object detection methods. Results are shown in Table 8. With ResNet-101 as the backbone, our method

Method	Backbone	epochs	jitter	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster R-CNN[27]	ResNet-101	12		36.2	59.1	39.0	18.2	39.0	48.2
Mask R-CNN[18]	ResNet-101	12		38.2	60.3	41.7	20.1	41.1	50.2
Mask R-CNN[18]	ResNeXt-101	12		39.8	62.3	43.4	22.1	43.2	51.2
RetinaNet[28]	ResNet-101	12		39.1	59.1	42.3	21.8	42.7	50.2
RepPoints[47]	ResNet-101	12		41.0	62.9	44.3	23.6	44.1	51.7
ATSS[48]	ResNeXt-101-DCN	24	✓	47.7	66.5	51.9	29.7	50.8	59.4
CornerNet[25]	HG-104	100	✓	40.5	56.5	43.1	19.4	42.7	53.9
ExtremeNet[50]	HG-104	100	✓	40.1	55.3	43.2	20.3	43.2	53.1
CenterNet [49]	HG-104	100	✓	42.1	61.1	45.9	24.1	45.5	52.8
Ours*	ResNet-50	12		39.4	58.9	42.6	22.2	43.0	49.6
Ours	ResNet-50	12		40.1	59.7	43.3	22.8	42.8	50.4
Ours	ResNet-50	36	✓	43.9	64.0	47.6	26.7	46.7	54.1
Ours	ResNet-101	12		42.1	62.0	45.6	24.0	45.1	52.9
Ours	ResNet-101	36	✓	45.6	65.7	49.7	27.7	48.9	56.6
Ours	ResNeXt-101	36	✓	47.0	67.3	51.1	29.3	50.1	58.0
Ours	ResNeXt-101+DCN	36	✓	48.9	69.2	53.4	30.5	51.9	61.2

Table 8. Object detection on COCO [29] `test-dev`. Our method significantly surpasses all other state-of-the-arts. '*' indicates training without ATSS [48] assigner and 'jitter' indicates using scale-jitter during training.

achieves 42.1 mAP with the 1x setting, outperforming RepPoints [47] and Mask R-CNN by 1.1 mAP and 3.9 mAP, respectively. With ResNeXt-101-DCN as a stronger backbone, our method achieves 48.9 mAP, surpassing all other anchor-free SOTA methods.

5 Conclusion

In this paper, we present *Dense RepPoints*, a dense attributed point set representation for 2D objects. By introducing efficient feature extraction and employing dense supervision, this work takes a step towards learning a unified representation for top-down object recognition pipelines, enabling explicit modeling between different visual entities, *e.g.* coarse bounding boxes and fine instance masks. Besides, we also propose a new point sampling method to describe masks, shown to be effective in our experiments. Experimental results show that this new dense 2D representation is not only applicable for predicting dense masks, but also can help improve other tasks such as object detection via its novel multi-granular object representation. We also analyze the upper bound for our representation and plan to explore better score head designs and system-level performance improvements particularly on large objects.

Acknowledgement We thank Jifeng Dai and Bolei Zhou for discussion and comments about this work. Jifeng Dai was involved in early discussions of the work and gave up authorship after he joined another company.

References

1. Acuna, D., Ling, H., Kar, A., Fidler, S.: Efficient interactive annotation of segmentation datasets with polygon-rnn++ (2018)
2. Alp Güler, R., Neverova, N., Kokkinos, I.: Densepose: Dense human pose estimation in the wild. In: CVPR. pp. 7297–7306 (2018)
3. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2d pose estimation using part affinity fields. In: CVPR. pp. 7291–7299 (2017)
4. Castrejon, L., Kundu, K., Urtasun, R., Fidler, S.: Annotating object instances with a polygon-rnn. In: CVPR. pp. 5230–5238 (2017)
5. Chan, T.F., Vese, L.A.: Active contours without edges. IEEE Transactions on image processing **10**(2), 266–277 (2001)
6. Chen, L.C., Hermans, A., Papandreou, G., Schroff, F., Wang, P., Adam, H.: Masklab: Instance segmentation by refining object detection with semantic and direction features. In: CVPR. pp. 4013–4022 (2018)
7. Chen, X., Girshick, R.B., He, K., Dollár, P.: Tensormask: A foundation for dense object segmentation. In: ICCV (2019)
8. Cheng, D., Liao, R., Fidler, S., Urtasun, R.: Darnet: Deep active ray network for building segmentatio. arXiv preprint arXiv:1905.05889 (2019)
9. Dai, J., He, K., Li, Y., Ren, S., Sun, J.: Instance-sensitive fully convolutional networks. In: ECCV. pp. 534–549. Springer (2016)
10. Dai, J., He, K., Sun, J.: Instance-aware semantic segmentation via multi-task network cascades. In: CVPR. pp. 3150–3158 (2016)
11. Dai, J., Li, Y., He, K., Sun, J.: R-fcn: Object detection via region-based fully convolutional networks. In: NeurIPS. pp. 379–387 (2016)
12. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database (2009)
13. Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., Tian, Q.: Centernet: Object detection with keypoint triplets. arXiv preprint arXiv:1904.08189 (2019)
14. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. IJCV **88**(2), 303–338 (2010)
15. Fan, H., Su, H., Guibas, L.J.: A point set generation network for 3d object reconstruction from a single image. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 605–613 (2017)
16. Girshick, R.: Fast r-cnn. In: ICCV. pp. 1440–1448 (2015)
17. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR. pp. 580–587 (2014)
18. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: ICCV. pp. 2961–2969 (2017)
19. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016)
20. Huang, C., Ai, H., Li, Y., Lao, S.: High-performance rotation invariant multiview face detection. PAMI **29**(4), 671–686 (2007)
21. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. IJCV **1**(4), 321–331 (1988)
22. Kirillov, A., Levinkov, E., Andres, B., Savchynskyy, B., Rother, C.: Instancecut: from edges to instances with multicut. In: CVPR. pp. 5008–5017 (2017)
23. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NeurIPS. pp. 1097–1105 (2012)

24. Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., Krasin, I., Pont-Tuset, J., Kamali, S., Popov, S., Malloci, M., Duerig, T., et al.: The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. arXiv preprint arXiv:1811.00982 (2018)
25. Law, H., Deng, J.: Cornernet: Detecting objects as paired keypoints. In: ECCV. pp. 734–750 (2018)
26. Li, Y., Qi, H., Dai, J., Ji, X., Wei, Y.: Fully convolutional instance-aware semantic segmentation. In: CVPR. pp. 2359–2367 (2017)
27. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: ICCV. pp. 2117–2125 (2017)
28. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: ICCV. pp. 2980–2988 (2017)
29. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV. pp. 740–755. Springer (2014)
30. Ling, H., Gao, J., Kar, A., Chen, W., Fidler, S.: Fast interactive object annotation with curve-gcn. In: CVPR (2019)
31. Moreira, A., Santos, M.Y.: Concave hull: A k-nearest neighbours approach for the computation of the region occupied by a set of points (2007)
32. Palmer, S.E.: Vision science: Photons to phenomenology. MIT press (1999)
33. Peng, S., Jiang, W., Pi, H., Bao, H., Zhou, X.: Deep snake for real-time instance segmentation. arXiv preprint arXiv:2001.01629 (2020)
34. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: CVPR (2017)
35. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: NeurIPS. pp. 5099–5108 (2017)
36. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: NeurIPS. pp. 91–99 (2015)
37. Rubner, Y., Tomasi, C., Guibas, L.J.: The earth mover's distance as a metric for image retrieval. IJCV **40**(2), 99–121 (2000)
38. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)
39. Srinivasan, P., Zhu, Q., Shi, J.: Many-to-one contour matching for describing and discriminating object shape. In: CVPR (2010)
40. Toshev, A., Taskar, B., Daniilidis, K.: Shape-based object detection via boundary structure segmentation. IJCV **99**(2), 123–146 (2012)
41. Wang, X., Bai, X., Ma, T., Liu, W., Latecki, L.J.: Fan shape model for object detection. In: CVPR. pp. 151–158. IEEE (2012)
42. Wang, X., Kong, T., Shen, C., Jiang, Y., Li, L.: Solo: Segmenting objects by locations. arXiv preprint arXiv:1912.04488 (2019)
43. Wei, S.E., Ramakrishna, V., Kanade, T., Sheikh, Y.: Convolutional pose machines. In: CVPR (2016)
44. Wu, Y., He, K.: Group normalization. In: ECCV. pp. 3–19 (2018)
45. Xie, E., Sun, P., Song, X., Wang, W., Liu, X., Liang, D., Shen, C., Luo, P.: Polarmask: Single shot instance segmentation with polar representation. arXiv preprint arXiv:1909.13226 (2019)
46. Yang, J., Price, B., Cohen, S., Lee, H., Yang, M.H.: Object contour detection with a fully convolutional encoder-decoder network. In: CVPR. pp. 193–202 (2016)
47. Yang, Z., Liu, S., Hu, H., Wang, L., Lin, S.: Reppoints: Point set representation for object detection. In: CVPR (2019)

48. Zhang, S., Chi, C., Yao, Y., Lei, Z., Li, S.Z.: Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. arXiv preprint arXiv:1912.02424 (2019)
49. Zhou, X., Wang, D., Krähenbühl, P.: Objects as points. arXiv preprint arXiv:1904.07850 (2019)
50. Zhou, X., Zhuo, J., Krähenbühl, P.: Bottom-up object detection by grouping extreme and center points. In: CVPR (2019)