Pattern Recognition Letters 154 (2022) 53–59

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Regularizing deep networks with label geometry for accurate object localization on small training datasets

Xiaolian Wang[a,b], Xiyuan Hu[c,*], Chen Chen[a,b], Silong Peng[a,b,d]

[a] Institute of Automation, Chinese Academy of Sciences, Beijing, China
[b] University of Chinese Academy of Sciences, Beijing, China
[c] Nanjing University of Science and Technology, Nanjing, Jiangsu, China
[d] Beijing ViSystem Corporation Limited, Beijing, China

ARTICLE INFO

Article history:
Received 15 August 2021
Revised 28 October 2021
Accepted 5 January 2022
Available online 13 January 2022

Edited by Maria De Marsico

Keywords:
Object detection
Object localization
Label geometry
Box evolution
Small dataset
Human-machine interaction

ABSTRACT

Localization is a critical subtask in object detection, which is closely related to spatial information of objects. Most current detectors simply rely on the fitting ability of deep neural networks to regress towards numerical targets such as coordinates of object boxes. Training deep networks for sufficient fitting ability requires a large number of annotations that are expensive to obtain. In this work, we fully exploit limited annotations by extracting label geometry to improve localization performance on small datasets. We generate distance transform of bounding box edges according to localization labels, with which we supervise intermediate outputs of networks pixel by pixel to reconstruct object geometry for localization. Distance transform is sensitive to box edges and provides geometric gradients flowing into boundaries. We learn such gradients to enhance geometric-aware features through a coupled training with regression, and use it to refine regressed boxes in an evolutionary manner in inference. Extensive experiments are implemented to demonstrate the effectiveness of our method. Our method can be applied in applications that required human-machine interaction, such as the driver-assistance system in autonomous driving, by providing accurate detections to assist humans in making better decisions.

1. Introduction

Object detection is a fundamental task in computer vision, and detecting accurate bounding boxes is of great importance in many modern applications such as traffic monitoring [1–5], surveillance in smart Internet of Things (IoT) [6,7], medical pre-diagnosis support [8], smart manufacturing [9], anomaly detection [10], and advanced tasks with unknown objects [11,12]. Object detection also plays an important role in applications involving human-machine interaction, such as the driver-assistance system in autonomous driving, where accurate detections can assist humans in making better decisions. Despite differences in design details of various detection frameworks, most object detectors [13–15] rely on bounding box regression to localize objects, which is effective to predict continuous variables such as offsets. Though this paradigm has achieved impressive performance, it still leaves room for improvement. For example, the localization task is heavily related to spatial information of objects, but the regression targets are usually 4-d vectors for the network to fit, which have neither structure nor geometric information aligned with input images. Thus detectors entail strong fitting ability to bridge the gap between inputs and targets. In addition, detectors employ deep neural networks to extract features for regression, whereas spatial details such as fine object boundaries may be lost during consecutive strides and feature poolings, which limits the precision of localization.

Recently, keypoint-based detectors [16–19] have been greatly developed. They output heatmaps with object keypoints such as corners [16,19] and centers [17,18] highlighted, and then group keypoints with similar embeddings to enclose objects with bounding boxes. These methods convert numerical labels into spatial space, and learn geometry-aware features which are spatially aligned with corresponding keypoints. Keypoints can reflect the exact spatial extent of objects, thus lead to better localization performance. However, as keypoints of multiple objects are simultaneously active on the same heatmap, the keypoint-based detectors often suffer incorrect keypoint clustering which greatly affects detection performance. In addition, as the localization performance highly depends on the accuracy of keypoint detection, this keypoint-based method is susceptible to appearance missing on object boundaries, especially when heavy occlusion occurs.

* Corresponding author.
E-mail address: huxy@njust.edu.cn (X. Hu).

https://doi.org/10.1016/j.patrec.2022.01.004
0167-8655/

To take advantage of both keypoint-based and regression-based methods, we propose an architecture on regression-based FCOS [15] to learn label geometry, with which we regularize the deep regression networks for accurate localization. We convert bounding box labels to 2-d maps, distance transform masks of box edges, to spatially supervise intermediate outputs of the detector. The intermediate supervision constrains the optimization path of neural networks, thus the detector can impose more attention on spatial information in training.

Supervised by distance transform labels, the intermediate outputs reconstruct some weak (i.e., bounding box edges) but sufficient geometric information from deep networks. It is intuitive to integrate the learned geometry into the regression branch to improve localization performance. Inspired by Active Contour Models [20] and its variant of active rays [21,22] which apply constraints of image geometry such as gradients to iterative parameter update, we couple regressed boxes with intermediate masks in a similar evolutionary manner to incorporate geometric regularization into regression. We construct an energy function on predicted distance transforms, and evolve regressed boxes via minimizing the energy function, the minimum of which defines the final sides of bounding boxes.

Our main contributions can be summarized as follows:

1. We propose an architecture of learning object geometry hidden in numerical labels to improve object localization. An intermediate supervision is imposed on deep networks to enforce spatial regularization in training, which benefits side-aware feature learning.
2. We couple reconstructed geometry, distance transform of box edges, with regression in training to enhance regression features, and iteratively evolve regressed boxes in inference to refine object locations.

2. Related works

2.1. Geometry in loss functions

Some methods [23–25] take geometry such as distance and area of boxes into account when designing loss functions for regression. IoU loss in UnitBox [23] is the first to consider overlap between boxes and regress four sides of a predicted box as a whole unit. It is designed to alleviate the sensitivity to variant object scales in commonly adopted ℓ_n-norm losses, such as ℓ_2-norm [26] and ℓ_1-smooth loss [13]. Since then, more geometric properties are considered, such as the smallest convex shapes enclosing predictions and targets in GIoU loss [24], central point distance in DIoU loss [25], and aspect ratio in CIoU loss [25]. These losses converge faster in training, and bring better performance in inference.

2.2. Geometry in localization targets

Most detectors apply a top-down manner to detect objects, and regress offsets from densely tiled anchors [13] or candidate locations [15] to generate bounding boxes. Their targets are 4-d vectors related to bounding box coordinates, and the prediction of continuous variables enables fine-tuning of object locations. CornerNet [16] introduces a new perspective to the localization task. It detects bounding boxes from high-resolution heatmaps in a bottom-up style. The localization target of this kind of detectors is directly associated with geometry, such as heatmaps of corners [16,19] and centers [17,18], and the geometric structure among keypoints are explicitly considered when grouping them into bounding boxes [17,18]. To take advantage of the two kinds of detectors, we supervise our model with both localization targets. Besides the conventional targets for the regression branch, we apply distance transform of box edges to supervise intermediate outputs of our detector, and further couple the intermediate prediction with regressed boxes for joint training and iterative refinement in inference.

2.3. Geometry in feature fusion

Geometric information can also be integrated into neural networks via feature fusion. Common choices of geometric features are object edges [27], image gradients [27], semantic segmentation [27], masks of bounding boxes [28], corner heatmaps [28], and so on. These features, which are either extracted from other models or jointly trained as a new task along with object detection, are added or concatenated to backbone layers of detectors. Different from these methods that imposing object geometry on features, we directly couple learned geometry with regressed boxes to refine object locations, and enhance side-aware features via distance transform learning.

3. Regularizing deep networks with label geometry

3.1. Learning from label geometry

The localization label of an object is usually coordinates of $(x_{\min}, y_{\min}, x_{\max}, y_{\max})$ for corners or $(x_{\min}, y_{\min}, h, w)$ with the size of bounding box. We convert it to distance transform of box edges to construct spatial supervision. The transformed labels can reflect spatial information of objects, including aspect ratios and scales of objects, the scope of object extension, as well as the absolute position of an object in the image. The distance transform maps also provide geometric gradients pointing to edges of boxes. We consider such spatial information hidden in labels as label geometry, and utilize it to regularize deep networks in the spatial space. Each ground-truth distance transform responds to a unique and complete bounding box of an object. This allows all pixels of box edges including occluded ones to be supervised in the boundary learning. Due to the one-to-one correspondence, the confusion of pixels in the overlapping area belonging to which object can also be avoided. Correspondingly, predictions of different objects should be exclusively generated on different maps, and should also have high resolution to retain sufficient spatial details.

Based on above considerations, we employ the mask branch introduced by CondInst [29] to generate distance transform for each instance, and build the mask branch on FPN layer P3 after a series of (e.g., 4) Conv-ReLU layers to generate high-resolution masks, as shown in Fig. (1). The mask branch applies three 1×1 convolutions with 8 channels to decouple instance-aware features and generate instance masks separately. Kernels $\theta \in \mathbb{R}^d$ of the three convolution filters are dynamically generated by controller heads which employ the same structure as the classification head, i.e., 4 Conv-ReLU layers on each FPN level. Each kernel $\theta_{x,y}$ generated from feature location (x, y) is either associated with one candidate object or the background according to the assignment of positive samples. We feed $\theta_{x,y}$ of K positive locations into the mask branch to generate distance transform of K positive instances separately. The outputs are then upsampled with bilinear interpolation to 1/4 of the original image as applied in CondInst.

In implementation, the mask branch outputs two kinds of distance transform denoted by D and B, and both maps are processed by projection before being sent into the evolution module. We explain the design motivation and details of this part of structure in Section (3.3).

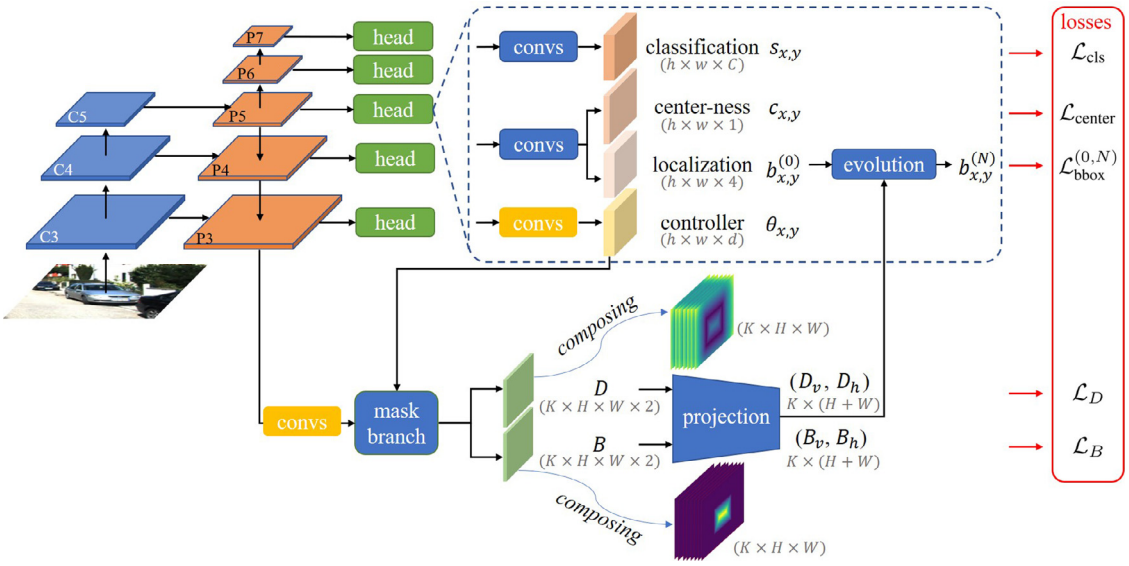

Fig. 1. Illustration of the architecture. Generated by the mask branch, D and B have 2 channels for distance transform along horizontal and vertical directions, which are further projected to 1-d vectors to iteratively evolve box $b^{(0)}$ for refined locations.

3.2. Evolution of bounding boxes

Inspired by Active Contour Models (ACM) [20] and its variant of active rays [21,22], we employ an evolution fashion to couple reconstructed geometry with regressed boxes.

We first parameterize the bounding box via rays. Given a reference center (x, y) inside an object, points located on box sides can be represented by polar coordinates

$$p_{x,y}(\rho, \theta) = \begin{bmatrix} x + \rho\cos(\theta) \\ y + \rho\sin(\theta) \end{bmatrix}, \tag{1}$$

where $\rho \in [0, +\infty)$ is the radius reaching from the reference point to the side, and $\theta \in [0, 2\pi)$ is the angle starting from x-axis. In fact, we do not need every point on sides to form a bounding box. Four points $\{p_i\}_{i=0}^{3}$ are sufficient, i.e., the leftmost and rightmost points horizontally extending from (x, y) to bounding box sides, the topmost and bottommost points vertically extending from (x, y), where

$$p_i = \begin{bmatrix} x + \rho_i\cos(i\Delta\theta) \\ y + \rho_i\sin(i\Delta\theta) \end{bmatrix} \tag{2}$$

with $\Delta\theta$ set to $\pi/2$.

We then design an energy function $E(p)$ of the candidate box for evolution. Inspired by Cheng et al. [22], $E(p)$ consists of a data term E_{data} and a balloon term E_{balloon}, and is defined as

$$E(p) = \sum_{i=0}^{3} [E_{\text{data}}(p_i) + \beta E_{\text{balloon}}(p_i)], \tag{3}$$

where $E_{\text{data}}(p_i)$ provides force from both inside and outside towards the box edge, and $E_{\text{balloon}}(p_i)$ serves as an extra outward expanding force to modulate the whole energy $E(p)$ with a hyperparameter β, as well as encourages outward expansion from a small initialization. Following [22], the data and balloon term in Eq. (3) are formulated as,

$$E_{\text{data}}(p_i) = D(p_i), \tag{4}$$

$$E_{\text{balloon}}(p_i) = B(p_i)\left(1 - \frac{\rho_i}{\rho_{\max}}\right), \tag{5}$$

where D and B are both the simplest Euclidean distance transform and each pixel value is the distance to the nearest bounding box side. The only difference of them is that we mask out the exterior of bounding boxes in B to keep single outward forces, as shown in Fig. (1). Both $D(p_i)$ and $B(p_i)$ are obtained by bilinearly sampling point p_i from distance transform map D and B. Symbol $\rho_{\max}$ is the maximum radius a box can reach within the image.

The minimum of $E(p)$ leads to the optimal box of an object. To find p that minimizes $E(p)$, we compute partial derivatives of each term w.r.t the single variable ρ, and set them to zero. For partial derivatives of $E_{\text{data}}(p)$, we get

$$\frac{\partial E_{\text{data}}(p)}{\partial \rho_i} = \frac{\partial D(p_i)}{\partial x}\cos(i\Delta\theta) + \frac{\partial D(p_i)}{\partial y}\sin(i\Delta\theta), \tag{6}$$

where we resort to Cartesian coordinates to compute derivatives with Sobel filters. As for $E_{\text{balloon}}(p)$, we consider B to be constant within a small vicinity of p_i for approximation, thus we get

$$\frac{\partial E_{\text{balloon}}(p)}{\partial \rho_i} \approx -\frac{B(p_i)}{\rho_{\max}}. \tag{7}$$

We evolve the radius of each ray as below,

$$\rho_i^{(t+1)} = \rho_i^{(t)} - \Delta t\left(\frac{\partial E_{\text{data}}(p)}{\partial \rho_i^{(t)}} + \beta\frac{\partial E_{\text{balloon}}(p)}{\partial \rho_i^{(t)}}\right), \tag{8}$$

where Δt is a hyperparameter of the time step, and $\rho_i^{(0)}$ is initialized with regressed boxes. Through iterative update of radius, we incorporate side information into regression in training. Regressed boxes can be further refined in the same evolutionary manner in inference.

3.3. Projection

Considering that each distance transform D and B only responds to one target, the background pixels will dominate the whole image. In addition, only pixels near regressed box edges are used for evolution. Thus equally supervising all values on the 2-d masks may distract attention from effective pixels to redundant ones. According to Eqs. (6)–(8), the radius is only updated in four single 1-d directions. As $\Delta\theta$ is set to $\pi/2$, the partial derivative of E_{data} involves either $\partial D(p_i)/\partial x$ or $\partial D(p_i)/\partial y$. Therefore, we decouple the 2-d learning of mask D and B to 1-d x and y components to reduce redundant supervision.

We separately predict horizontal and vertical components of energy maps through the mask branch, resulting in $D, B \in \mathbb{R}^{H\times W\times 2}$

as shown in Fig. (1). Maps D and B are then projected along the horizontal and vertical direction with max-pooling to pass salient side information to the final 1-d vectors. Finally, projected D and B are sent to the evolution module to iteratively update regressed boxes. We summarize the evolution process in Algorithm (1).

Algorithm 1: Evolution of bounding boxes.

Input:
The set of regressed boxes: $\mathcal{B}$.
Projected distance transform: horizontal D_h and $B_h \in \mathbb{R}^W$, and vertical D_v and $B_v \in \mathbb{R}^H$.
The number of iterations, step length, and the weight of the balloon term: N, Δt, and β.
// Box P_{x_r,y_r} in $\mathcal{B}$ is represented in polar coordinates of $(x_{\min}, y_{\min}, x_{\max}, y_{\max})$ as

$$P_{x_r,y_r} := (x_r - \rho_2, y_r - \rho_3, x_r + \rho_0, y_r + \rho_1).$$

// (x_r, y_r) is a reference point inside an object which can be bounded by box P_{x_r,y_r}.
// ρ_0, ρ_1, ρ_2, ρ_3 are distance from (x_r, y_r) to the right, bottom, left, top edge of P_{x_r,y_r}, respectively.
Output: The set of evolved boxes: $\hat{\mathcal{B}}$
$\hat{\mathcal{B}} \leftarrow \varnothing$
for *each positive* (x_r, y_r) **do**
  for $i = 1, 2, \ldots, N$ **do**
    // According to Eqs.~(6)–(8)
    $\rho_0 \leftarrow \rho_0 - \Delta t\left(\frac{\partial D_h(x)}{\partial x}\big|_{x=x_r+\rho_0} - \beta B_h(x)\big|_{x=x_r+\rho_0}\right)$
    $\rho_1 \leftarrow \rho_1 - \Delta t\left(\frac{\partial D_v(y)}{\partial y}\big|_{y=y_r+\rho_1} - \beta B_v(y)\big|_{y=y_r+\rho_1}\right)$
    $\rho_2 \leftarrow \rho_2 - \Delta t\left(-\frac{\partial D_h(x)}{\partial x}\big|_{x=x_r-\rho_2} - \beta B_h(x)\big|_{x=x_r-\rho_2}\right)$
    $\rho_3 \leftarrow \rho_3 - \Delta t\left(-\frac{\partial D_v(y)}{\partial y}\big|_{y=y_r-\rho_3} - \beta B_v(y)\big|_{y=y_r-\rho_3}\right)$
  end
  $P_{x_r,y_r} \leftarrow (x_r - \rho_2, y_r - \rho_3, x_r + \rho_0, y_r + \rho_1)$
  $\hat{\mathcal{B}} \leftarrow \hat{\mathcal{B}} \cup \{P_{x_r,y_r}\}$
end

3.4. Training and inference

Our detector is optimized by a multi-task loss

$$\mathcal{L} = \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{center}} + \mathcal{L}_{\text{bbox}}^{(0,N)} + \mathcal{L}_D + \mathcal{L}_B, \tag{9}$$

where $\mathcal{L}_{\text{cls}}$ is the focal loss [14] for classification, and $\mathcal{L}_{\text{center}}$ is the cross-entropy loss for center-ness, following FCOS [15]. We employ GIoU loss [24] for $\mathcal{L}_{\text{bbox}}$ to supervise the initial regressed box $b^{(0)}$ and the final evolved box $b^{(N)}$, and adopt dice loss [30] for $\mathcal{L}_D$ and $\mathcal{L}_B$ to supervise projected D and B. We supervise 1-d x and y components of distance transform as explained in Section (3.3). But for D, in implementation, we directly learn 1-d x and y derivatives used in Eq. (6), as we find this leads to faster convergence in training and more robust performance in inference. Ground-truth derivatives are obtained by applying Sobel filters with kernel $[-1, 0, 1]$ on projected D.

In inference, we forward input images through the network to obtain classification scores $s_{x,y}$, center-ness scores $c_{x,y}$, regressed boxes $b_{x,y}^{(0)}$, and kernel parameters $\theta_{x,y}$. We select locations (x, y) with top K scores $s_{x,y}$ to generate D and B, and use them to iteratively evolve $b_{x,y}^{(0)}$. Following FCOS [15], we use the product of $s_{x,y}$ and $c_{x,y}$ to rank evolved boxes $b_{x,y}^{(N)}$, and finally apply non-maximum suppression (NMS) to filter out redundant boxes.

4. Experiments

We conduct experiments on the challenging KITTI [31] dataset to evaluate our method. KITTI contains 7481 images for training and 7518 images for testing. We split the original training set by randomly sampling 2046 images for validation. KITTI applies Average Precision (AP) to evaluate predictions with matching overlap no less than 70%, and provides three levels of difficulties, i.e., *Easy* (E), *Moderate* (M), and *Hard* (H). We also evaluate detailed performance of AP on different matching IoUs and object sizes, i.e., *Small* (S), *Medium* (M), and *Large* (L), as well as Average Recall (AR).

Table 1
Performance of different Δt and β when $N = 1$.

Δt	β	AP	AP_{70}	AP_{80}	AP_{90}	E	M	H
1.0	0	66.0	94.2	85.5	45.4	91.66	92.05	91.93
0.5		**66.7**	**94.4**	**86.5**	**46.4**	96.96	**92.66**	**92.60**
0.1		65.1	93.9	84.6	43.9	**98.76**	92.05	91.82
0.5	0.5	66.1	94.3	85.7	44.9	94.86	92.16	92.13
	0.2	**66.7**	94.4	86.2	**47.2**	**99.03**	**92.67**	**92.76**
	0.1	**66.7**	**94.5**	**86.5**	46.5	98.10	92.47	92.45
	0	**66.7**	94.4	**86.5**	46.4	96.96	92.66	92.60

Table 2
Performance with different number of iterations N in training when $\Delta t = 0.5$ and $\beta = 0.1$.

N	AP	AP_{70}	AP_{80}	AP_{90}	E	M	H
1	66.7	94.5	86.5	**46.5**	98.10	92.47	92.45
2	**66.9**	**94.6**	**86.8**	45.6	**98.55**	92.56	92.63
3	66.5	94.4	85.9	46.4	92.04	92.68	92.66
4	66.6	94.0	86.1	**46.5**	93.60	**93.01**	**92.76**
5	66.6	94.4	86.3	46.0	91.88	92.39	92.34

Table 3
Performance with candidate number K in inference.

K	AP	AP_{70}	AP_{80}	AP_{90}	AP_{70}^S	AP_{70}^M	AP_{70}^L
100	66.0	92.9	85.7	45.6	89.0	92.9	95.1
200	66.6	94.2	86.4	45.4	90.2	94.2	95.9
500	66.9	94.6	86.8	45.6	90.2	94.6	96.4
1000	66.9	94.6	86.8	45.6	90.2	94.6	96.4

4.1. Implementation details

Our detector is built on top of FCOS [15], and employs ResNet-50 augmented with FPN as the backbone. We initialize the backbone with weights pretrained on ImageNet [32], and use an SGD optimizer with 0.0001 weight decay and 0.9 momentum for training. We train networks with a batch size of 4 for 12 epochs on a single GTX 2080Ti GPU. The initial learning rate is set to 0.005, and reduced by 10 at epoch 8 and 11, respectively. We also apply a linear warm-up scheme for the first 500 iterations. Input images are resized to [800, 1333] without changing the original ratio. In inference, we apply NMS with a threshold of 0.7, and evaluate results with scores higher than 0.05.

Our method involves four hyperparameters in the bounding box evolution, i.e., the update step Δt, the weight of the balloon term β, the maximum radius $\rho_{\max}$, and training iterations N. In implementation, we do not assign an exact value to $\rho_{\max}$. Since β and $1/\rho_{\max}$ are both coefficients of $B(p_i)$ when putting Eq. (7) into Eq. (8), we simply use β to represent $\beta/\rho_{\max}$ to conduct parameter setting experiments, and choose hyper-parameter β and $\rho_{\max}$ together. According to Tables (1) and (2), we set $\{\Delta t, \beta, N\}$ to {0.5, 0.1, 2} as default in all our experiments. As for the candidate number K in inference, we set it to 500 according to Table (3).

4.2. Ablation study

Reconstructed geometry. We first validate the effectiveness of our method, including the learning of label geometry and coupling reconstructed geometry with regression, as shown in Table (4). Compared with the baseline FCOS (1st row), a simple learning of D and B (2nd row) brings an average of 2.8% AP gain. The improvement is mainly for higher matching IoU, e.g., AP_{70} is improved by 1.7% AP

Table 4
Ablations on the effectiveness of reconstructed geometry.

Learning geometry	Evolving boxes in		AP	AP_{70}	AP_{80}	AP_{90}	AP^S_{70}	AP^M_{70}	AP^L_{70}	AR	AR^S	AR^M	AR^L
	Training	Inference											
✗	✗	✗	61.6	92.0	81.1	37.7	84.0	92.3	95.5	81.0	72.6	80.6	85.8
√	✗	✗	64.4	93.7	83.9	41.7	88.3	93.9	95.8	82.4	76.0	81.9	86.4
√	✗	√	62.5	93.9	83.3	36.2	88.6	93.9	95.9	81.4	73.9	81.0	85.8
√	√	✗	65.5	93.8	85.7	43.3	89.1	94.2	96.2	82.8	76.2	82.5	86.7
√	√	√	66.9	94.6	86.8	45.6	90.2	94.6	96.4	83.5	77.4	83.0	87.5

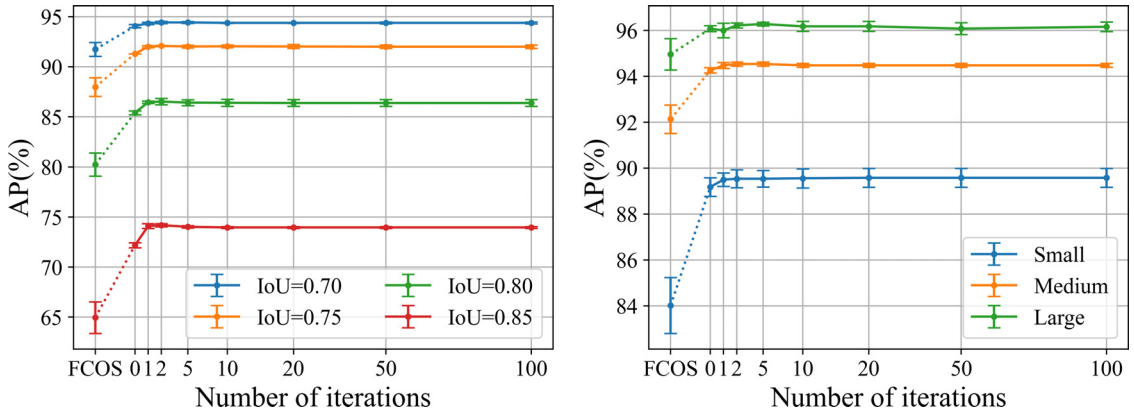

Fig. 2. Performance evaluated under different matching IoUs (left) and object sizes (right) with different iterations in inference. Error bars indicate one standard deviation.

Table 5
Detection performance on different occlusion levels.

Occl.	[0.5, 0.6)	[0.6, 0.7)	[0.7, 0.8)	[0.8, 0.9)	[0.9, 1.0]
FCOS	75.17	64.29	43.92	17.77	13.03
Ours	83.47	75.27	64.62	44.73	20.95
Gains	+8.30	+10.98	+20.70	+26.96	+7.92

Table 6
Performance with similar parameters and computations.

Parm (M) / Comp (GFLOPs)	AP	AP_{70}	AP_{80}
FCOS (32.84 / 154.05)	60.87±1.02	91.7±0.69	80.2±1.15
FCOS* (34.02 / 180.37)	60.96±0.61	91.7±0.25	80.5±0.68
Ours-lite (34.37 / 177.29)	65.80±0.10	94.2±0.08	85.5±0.17

while AP_{90} is improved by 4.0% AP. This indicating that spatially regularizing the intermediate output of deep neural networks benefits the learning of side-aware features for localization. But when we evolve regressed boxes with learned D and B in inference (3rd row), the performance degrades. The reason may be that without supervising evolved boxes in training, D and B are not well learned to properly interact with regressed boxes. In the fourth row, we jointly train evolved and regressed boxes. Even though there is no evolution in inference, results of the single regressed boxes are improved by 3.9% AP over FCOS. With further refinement in inference (the last row), we surpass FCOS by an average of 5.3% AP.

Iterations in inference. We experiment with different iterations in inference, and depict results along with FCOS in Fig. (2), where results reported as error bars are obtained by training with the same configuration for 5 times. The maximum gain appears when increasing iterations from 0 to 1, and the performance becomes stable after 2 iterations, thus 2 iterations in inference is sufficient for our method.

Performance under occlusions. As KITTI does not provide detailed occlusion levels, we calculate occlusion ratios of objects by ourselves. Due to the lack of segmentation labels, we use Intersection over Foreground (IoF), i.e., occluded intersection over current bounding box area, to compute the occlusion ratio. Performance of our detector as well as FCOS under different occlusions are reported in Table (5). We outperform FCOS on occlusions within [0.5,1.0] by an average of +14.97% AP.

Complexity analysis. The full version of our detector involves slightly more parameters (38.14M vs. 32.84M) and computation (205.01G vs. 154.05G) than the baseline FCOS. This overhead mainly occurs at additional controller heads which predict dynamic convolution kernels for the mask branch. Therefore, the inference time is a little longer than FCOS. Given an input image of dimension 800 × 1333, our detector with a ResNet-50 backbone and 2 iterations in inference on average runs 62.9 ms vs. 43.3 ms of FCOS on a single GTX 2080 Ti GPU. However, this overhead can be avoided. As our initially regressed boxes still outperforms the baseline with a large margin as shown in Fig. (2), we can simply keep the additional structures in training, and discard them in inference, to improve the base model without additional parameters or inference time. For a fair comparison with FCOS, we also propose a lite version of our method. We lighten the structure of the controller head as well as *Convs* before the mask branch to one Conv-ReLU layer. We then add one more convolutional layer on FCOS heads to construct a baseline with similar parameters and computations with our lite version (34.02M/180.37G vs. 34.37M/177.29G). As shown in Table (6), our lite version still outperforms FCOS* (65.80% vs 60.96%), indicating improvements by our method are not due to more parameters and computation.

Visualization. Fig. (3) shows some visualization examples of features, where features of different objects especially the small ones generated by our detector separate from each other, whereas features generated by FCOS mostly stick together. This implies that our instance-aware geometric supervision does constrain the learning of features. When predicted bounding boxes are sufficiently separated from each other, missed detection of clustered objects caused by false suppression of NMS may be reduced. Benefitting from this, we improve the recall performance from 81.0% AR of FCOS to 83.5% AR. In addition, features of small objects generated

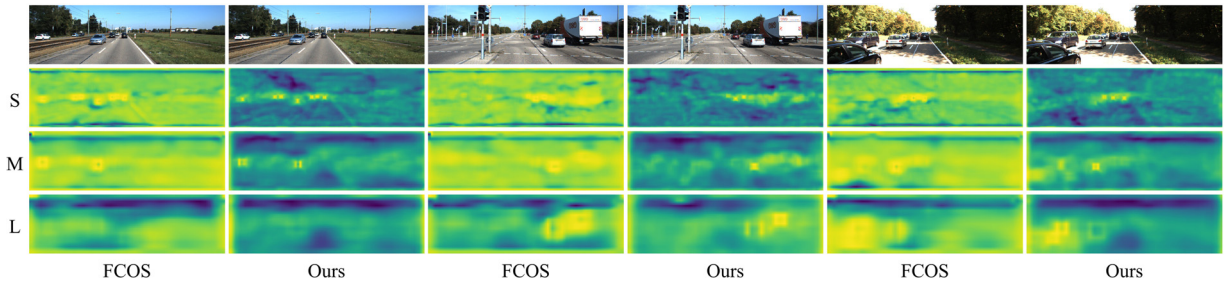

Fig. 3. Visualization of feature maps for small (S), medium (M), and large (L) objects on KITTI.

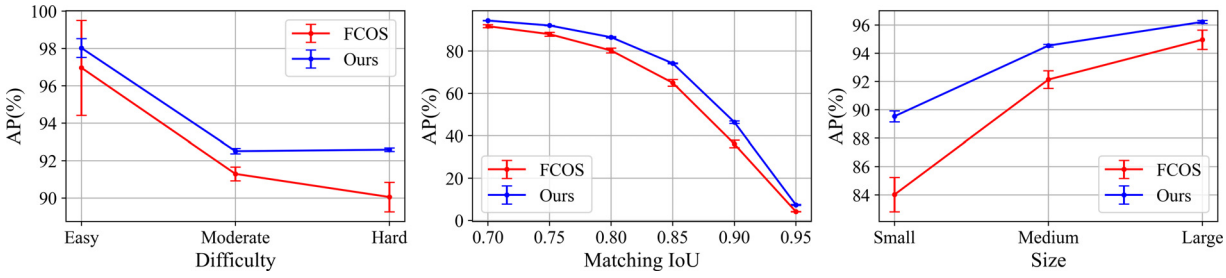

Fig. 4. The overall performance comparison of FCOS and our detector under different metrics on KITTI.

Table 7
Comparisons to state-of-the-art detectors with the best/second best AP (%) on KITTI.

Method	Backbone	Epoch	Car			Cyclist			Pedestrian		
			Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
CornerNet [16]	Hourglass-104	210	81.71	80.87	74.16	83.84	79.04	75.73	55.56	46.32	42.17
CentripetalNet [17]	Hourglass-104	210	87.90	88.50	87.78	84.98	83.88	81.67	69.66	59.55	54.72
RetinaNet [14]	ResNet-50	12	86.96	87.18	85.62	39.42	38.71	36.80	26.23	24.44	21.05
RepPoints [33]	ResNet-50	12	96.87	90.41	88.94	87.60	78.36	75.25	53.37	48.73	42.93
FCOS [15]	ResNet-50	12	97.90	91.24	90.31	87.57	79.02	77.54	52.50	46.82	42.78
FoveaBox [34]	ResNet-50	12	98.42	93.44	89.57	79.54	76.41	74.90	52.73	47.90	42.31
GFL [35]	ResNet-50	12	99.49	91.94	91.69	86.40	76.50	73.74	53.69	47.55	42.66
Ours	ResNet-50	12	98.55	92.56	92.63	86.97	80.29	77.22	64.47	57.17	51.66

Table 8
Performance of our method extended to FoveaBox.

Method	AP	AP_{70}	AP_{80}	AP_{90}	AP^S_{70}	AP^M_{70}	AP^L_{70}
FoveaBox	61.3	91.8	80.7	37.6	83.0	92.9	94.3
+Ours	64.4	92.6	84.0	43.6	85.3	93.5	95.4
Gains	+3.1	+0.8	+3.3	+6.0	+2.3	+0.6	+1.1

by our detector are highly illuminated against the background, explaining the large improvement on AP_S shown in Fig. (4), where an average of 5.5% AP gain is obtained (89.5% vs. 84.0%).

4.3. Comparison to state-of-the-art detectors

We compare the proposed method to other state-of-the-art detectors as shown in Table (7). Our detector is trained with the same configuration as well as hyperparameters as in ablation experiments except that the initial learning rate is set to 0.001 on *Cyclist* and *Pedestrian*, and the NMS threshold is set to 0.5 in inference. Without bells and whistles, our method achieves comparable results to state-of-the-art detectors.

4.4. Extension to other detectors

Our method can also be applied to other state-of-the-art detectors such as FoveaBox [34]. Training with the original configuration of FoveaBox, our approach obtains an average of 3.1% AP gain over the baseline as shown in Table (8). The improvement is mainly for higher matching IoU and small objects, e.g., AP_{90} is improved by 6.0% and AP^S_{70} increases by 2.3%, which is consistent with the improvement over FCOS.

5. Conclusion

In this paper, we propose an architecture to learn geometric information hidden in numerical localization labels. By spatially supervising intermediate outputs and coupling reconstructed geometry with regressed boxes, our detector gets improved performance on localization. Experiments constructed on KITTI demonstrate the effectiveness of our method.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by the National Key R&D Program of China under Grant 2021YFF0602101, the National Science Foundation of China under Grant NSFC 61906194, and Liaoning Collaboration Innovation Center For CSLE.

References

[1] A. Guezzaz, Y. Asimi, M. Azrour, A. Asimi, Mathematical validation of proposed machine learning classifier for heterogeneous traffic and anomaly detection, Big Data Min. Anal. 4 (1) (2021) 18–24.

[2] Y.N. Malek, M. Najib, M. Bakhouya, M. Essaaidi, Multivariate deep learning approach for electric vehicle speed forecasting, Big Data Min. Anal. 4 (1) (2021) 56–64.

[3] F. Dai, P. Huang, X. Xu, L. Qi, M.R. Khosravi, Spatio-temporal deep learning framework for traffic speed forecasting in IoT, IEEE Internet Things J. 3 (4) (2020) 66–69.

[4] C. Hu, W. Fan, E. Zen, Z. Hang, F. Wang, L. Qi, M.Z.A. Bhuiyan, A digital twin-assisted real-time traffic data prediction method for 5G-enabled internet of vehicles, IEEE Trans. Ind. Inform. 18 (4) (2022) 2811–2819.

[5] L. Bai, L. Yao, C. Li, X. Wang, C. Wang, Adaptive graph convolutional recurrent network for traffic forecasting, in: Proceedings of the Advances in Neural Information Processing Systems, 2020.

[6] X. Zhou, X. Yang, J. Ma, K.I.-K. Wang, Energy efficient smart routing based on link correlation mining for wireless edge computing in IoT, IEEE Internet Things J. (2021).

[7] X. Zhou, X. Xu, W. Liang, Z. Zeng, Z. Yan, Deep learning enhanced multi-target detection for end-edge-cloud surveillance in smart IoT, IEEE Internet Things J. 8 (16) (2021) 12588–12596.

[8] X. Zhou, Y. Li, W. Liang, CNN-RNN based intelligent recommendation for online medical pre-diagnosis support, IEEE ACM Trans. Comput. Biol. Bioinform. 18 (3) (2021) 912–921.

[9] X. Zhou, X. Xu, W. Liang, Z. Zeng, S. Shimizu, L.T. Yang, Q. Jin, Intelligent small object detection based on digital twinning for smart manufacturing in industrial CPS, IEEE Trans. Ind. Inform. (2021).

[10] X. Zhou, W. Liang, S. Shimizu, J. Ma, Q. Jin, Siamese neural network based few-shot learning for anomaly detection in industrial cyber-physical systems, IEEE Trans. Ind. Inform. 17 (8) (2021) 5790–5798.

[11] L. Zhang, X. Wang, L. Yao, F. Zheng, Zero-shot object detection with textual descriptions using convolutional neural networks, in: Proceedings of the International Joint Conference on Neural Networks, 2020, pp. 1–6.

[12] L. Zhang, X. Wang, L. Yao, L. Wu, F. Zheng, Zero-shot object detection via learning an embedding from semantic space to visual space, in: Proceedings of the International Joint Conference on Artificial Intelligence, 2020, pp. 906–912.

[13] S. Ren, K. He, R.B. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, in: Proceedings of the Advances in Neural Information Processing Systems, 2015, pp. 91–99.

[14] T. Lin, P. Goyal, R.B. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE International Conference Computing Vision, 2017, pp. 2999–3007.

[15] Z. Tian, C. Shen, H. Chen, T. He, FCOS: a simple and strong anchor-free object detector, IEEE Trans. Pattern Anal. Mach. Intell. (2020). 1–1

[16] H. Law, J. Deng, CornerNet: detecting objects as paired keypoints, Int. J. Comput. Vis. 128 (3) (2020) 642–656.

[17] Z. Dong, G. Li, Y. Liao, F. Wang, P. Ren, C. Qian, CentripetalNet: pursuing high-quality keypoint pairs for object detection, in: Proceedings of the IEEE Computer Vision and Pattern Recognition, 2020, pp. 10516–10525.

[18] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, Q. Tian, CenterNet: keypoint triplets for object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 6568–6577.

[19] K. Duan, L. Xie, H. Qi, S. Bai, Q. Huang, Q. Tian, Corner proposal network for anchor-free, two-stage object detection, in: Proceedings of the European Conference on Computer Vision, 2020, pp. 399–416.

[20] M. Kass, A.P. Witkin, D. Terzopoulos, Snakes: active contour models, Int. J. Comput. Vis. 1 (4) (1988) 321–331.

[21] J. Denzler, H. Niemann, Active rays: polar-transformed active contours for real-time contour tracking, Real Time Imaging 5 (3) (1999) 203–213.

[22] D. Cheng, R. Liao, S. Fidler, R. Urtasun, DARNet: deep active ray network for building segmentation, in: Proceedings of the IEEE Conference on Conference on Computer Vision and Pattern Recognition, 2019, pp. 7431–7439.

[23] J. Yu, Y. Jiang, Z. Wang, Z. Cao, T.S. Huang, UnitBox: an advanced object detection network, in: Proceedings of the ACM Conference on Multimedia, MM, 2016, pp. 516–520.

[24] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I.D. Reid, S. Savarese, Generalized intersection over union: a metric and a loss for bounding box regression, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 658–666.

[25] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, D. Ren, Distance-IoU loss: faster and better learning for bounding box regression, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2020, pp. 12993–13000.

[26] J. Redmon, S.K. Divvala, R.B. Girshick, A. Farhadi, You only look once: unified, real-time object detection, in: Proceedings of the IEEE Computer Vision and Pattern Recognition, 2016, pp. 779–788.

[27] J. Mao, T. Xiao, Y. Jiang, Z. Cao, What can help pedestrian detection? in: Proceedings of the IEEE Computer Vision and Pattern Recognition, 2017, pp. 6034–6043.

[28] Y. Chen, Z. Zhang, Y. Cao, L. Wang, S. Lin, H. Hu, RepPoints v2: verification meets regression for object detection, in: Proceedings of the Advances in Neural Information Processing Systems, 2020.

[29] Z. Tian, C. Shen, H. Chen, Conditional convolutions for instance segmentation, in: Proceedings of the European Conference on Computer Vision, 2020, pp. 282–298.

[30] F. Milletari, N. Navab, S. Ahmadi, V-Net: fully convolutional neural networks for volumetric medical image segmentation, in: Proceedings of the International Conference on 3D Vision, 2016, pp. 565–571.

[31] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? The KITTI vision benchmark suite, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 3354–3361.

[32] J. Deng, W. Dong, R. Socher, L. Li, K. Li, F. Li, ImageNet: a large-scale hierarchical image database, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255.

[33] Z. Yang, S. Liu, H. Hu, L. Wang, S. Lin, RepPoints: point set representation for object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 9656–9665.

[34] T. Kong, F. Sun, H. Liu, Y. Jiang, L. Li, J. Shi, Foveabox: beyond anchor-based object detection, IEEE Trans. Image Process. 29 (2020) 7389–7398.

[35] X. Li, W. Wang, L. Wu, S. Chen, X. Hu, J. Li, J. Tang, J. Yang, Generalized focal loss: learning qualified and distributed bounding boxes for dense object detection, in: Proceedings of the Advances in Neural Information Processing Systems, 2020.