PDNet: Towards Better One-stage Object Detection with Prediction Decoupling

Li Yang, Yan Xu, Shaoru Wang, Chunfeng Yuan, Ziqi Zhang, Bing Li, and Weiming Hu

Abstract—Recent one-stage object detectors follow a per-pixel prediction approach that predicts both the object category scores and boundary positions from every single grid location. However, the most suitable positions for inferring different targets, i.e., the object category and boundaries, are generally different. Predicting all these targets from the same grid location thus may lead to sub-optimal results. In this paper, we analyze the suitable inference positions for object category and boundaries, and propose a prediction-target-decoupled detector named PDNet to establish a more flexible detection paradigm. Our PDNet with the prediction decoupling mechanism encodes different targets separately in different locations. A learnable prediction collection module is devised with two sets of dynamic points, i.e., dynamic boundary points and semantic points, to collect and aggregate the predictions from the favorable regions for localization and classification. We adopt a two-step strategy to learn these dynamic point positions, where the prior positions are estimated for different targets first, and the network further predicts residual offsets to the positions with better perceptions of the object properties. Extensive experiments on the MS COCO benchmark demonstrate the effectiveness and efficiency of our method. With a single ResNeXt-64x4d-101 as the backbone, our detector achieves 48.7 AP with single-scale testing, which outperforms the state-of-the-art methods by an appreciable margin under the same experimental settings. Moreover, our detector is highly efficient as a one-stage framework. Our code will be public.

Index Terms—Object detection, prediction decoupling, convolutional neural network.

I. INTRODUCTION

OBJECT detection is a fundamental problem in computer vision aiming to localize and classify objects in digital images. In terms of the prediction stages needed by the detector, existing object detection networks can be generally categorized as the one-stage method [1], [2] and the two-stage method [3], [4]. The one-stage detectors directly produce the classification and localization results from dense grid points in one shot without explicit feature alignment procedure, while the two-stage methods include an additional stage of RoI feature extraction to improve the detection performance in a coarse-to-fine manner [3].

Due to the structural efficiency and competitive performance, one-stage methods have received great attention. The recent cutting-edge methods [5], [6], [7], [8] abandon the anchor box references and develop more straightforward detection frameworks that perform per-pixel classification and regression. For each output grid belonging to an object, these detectors predict the category scores and the current offsets to the leftmost, topmost, rightmost, and bottommost sides of the object. However, given the various object poses and shapes, it could be challenging for the features located at a single grid to accurately perceive the object category and four sides of the bounding box altogether. Intuitively, for example, it could be more difficult for the positions near the object boundary to perceive the semantic information than the positions inside. It could also be less accurate to regress the left border of the object from the positions near the right side. Fig. 1 visualizes the accuracy maps of classification and localization (four sides of the bounding box) predictions at different locations over the object area by a conventional one-stage detection framework [5]. From the accuracy maps, we can tell that the best positions for predicting different targets vary as expected. For the object instance, the regions near each object boundary tend to produce more accurate localization results, while the positions close to the semantic area tend to have

![Fig. 1. Accuracy maps of per-pixel classification and localization results from the current one-stage detector.](image)
wise predictions for the object category. These observations naturally raise a question: is it possible to efficiently separate the prediction targets and obtain the predictions for each target at their respective favorable positions in one shot?

To this end, we propose a Prediction-target-Decoupled detection Network (PDNet), where different targets are inferred separately at their corresponding proper positions. Specifically, unlike the previous one-stage methods that classify and localize an object instance from the same grid location in the feature map, we propose a prediction decoupling mechanism to separate the prediction targets as the object category and four sides of the object bounding box (in an offset manner), which are separately encoded at different locations by the network. To obtain the final detection results, we devise a learnable prediction collection module to collect and aggregate these intermediate predictions for different targets from different locations. Moreover, we analyze the suitable inference positions for localization and classification and propose two sets of dynamic points, i.e., dynamic boundary points and semantic points, to pinpoint these locations respectively. To this end, we introduce a two-step dynamic point generation strategy to facilitate the learning of dynamic points. The network first roughly estimates the prior positions for different targets to initialize the dynamic points, and then further predicts the residual offsets to shift the dynamic points towards the appropriate locations, to better exploit the one-stage detector’s potential. The proposed prediction decoupling mechanism naturally focuses different positions on the prediction of potential. The proposed prediction decoupling mechanism improves the performance, it increases the complexity of the network structure, which motivates researchers to develop more efficient one-stage detectors.

To summarize, the contributions of this work are:

- We analyze the dense predictions of the conventional one-stage detector and find that the best positions for inferring the object category and boundary positions are different. Inspired by the phenomena, we propose the PDNet with a prediction decoupling mechanism to flexibly collect and aggregate the predictions for different targets from different locations.
- We devise two sets of dynamic points, i.e., dynamic boundary points and semantic points, and propose a two-step dynamic point generation strategy to facilitate the learning of suitable point positions for localization and classification.
- Without bells and whistles, our method achieves state-of-the-art performance on the MS COCO benchmark. With a single ResNetXt-64×4d-101 as the backbone, our detector achieves 48.7 AP with single-scale testing, outperforming the other methods by an appreciable margin under the same experimental settings.

II. RELATED WORK

In this section, we first briefly introduce the two-stage and one-stage object detection methods, and then present various detection head designs for object localization and classification.

A. Two-stage Detection

The Faster R-CNN [3] demonstrates an exemplar two-stage detection framework, which has been successfully deployed in many applications. It first uses a region proposal network (RPN) to generate object region proposals and then employs a region-based convolutional neural network (R-CNN) [9, 10] to perform classification and refine the bounding box locations. Following this basic detection pipeline, a series of methods are proposed to further improve the performance, including network architecture reform [4, 11, 12, 13], object feature extraction [14, 15, 16], better region proposals [17, 18], training strategies and loss functions [19, 20, 21, 22, 23, 24]. While this multi-stage prediction strategy improves the performance, it increases the complexity of the network structure, which motivates researchers to develop more efficient one-stage detectors.

B. One-stage Detection

The emergence of SSD [1], RetinaNet [2], etc., establishes simple and effective one-stage detection frameworks. These methods preset the anchor boxes of various sizes at each grid location, and directly refine these box locations and classify the corresponding objects as well. Since then, many significant improvements have been made based on this framework [25, 26, 27, 28, 29, 30]. Another branch of one-stage detection is anchor-free methods [31, 32]. More recently, the anchor-free methods achieve comparable or even better performance than previous methods through more flexible and straightforward frameworks, such as FCOS [5], FoveaBox [6], FASF [7], SAPD [8], etc., which have received great attention. Without anchor boxes, they directly predict the category scores and the offsets to four sides of the object bounding box at each spatial grid of the feature maps. However, they generally infer all the properties (location and category) of an object from the same grid location, which may limit the potential of one-stage detectors. In our work, we follow this dense prediction manner but propose to more flexibly collect the localization and classification predictions from different positions that are more suitable.

C. Detection Head Design for Localization and Classification

Copious detection head design manners have been explored to push the performance boundary of object detection. Based on the two-stage detection pipeline, Double-Head [33] discusses the detection heads suitable for classification and localization, respectively. TSD [34] generates different region proposals to perform classification and localization. Grid R-CNN [35] implements localization via segmenting the grid points of the object bounding box. SABL [36] constructs
side-aware features to localize each boundary in a coarse-to-fine manner. Unlike these methods that rely on RoI features for prediction, we inherit the prediction scheme of one-stage methods and collect the localization and classification results at proper positions to achieve accurate and efficient detection.

There are also recent works that incorporate object feature extraction into one-stage detectors to make more accurate localization and classification predictions. RepPoints [37, 38] formulates the object as a set of representative points for feature sampling. AlignDet [39] proposes RoIConv to align the convolution features with object proposals to make predictions. Based on the detections of PCOS [5], BorderDet [40] gathers the border features to refine the classification and localization predictions. While exhibiting better performance, these methods all need additional detection branches to make predictions on extracted features, which may sacrifice inference efficiency. In comparison, we only collect predictions from the regression and classification maps of the conventional one-stage detection pipeline, which achieves higher accuracy while keeping the efficiency advantage.

Another family of research works follows a bottom-up approach to localize and classify objects. CornerNet [41] proposes to detect an object bounding box as a pair of keypoints, the top-left corner and the bottom-right corner. It first predicts the heatmaps of corner points for different categories and then groups the corner points with similar embeddings to form the detection results. CenterNet [42] extends CornerNet by introducing the detection of center keypoints to improve accuracy and recall. Zhou et al. [43] also detects the object centers and regresses the object sizes. ExtremeNet [44] detects four extreme points and one center point of objects and proposes center grouping to produce the detection results. Compared with these methods, we predict dynamic points to collect localization and classification results, and no additional embedding or grouping operations are required during post-processing.

III. Our method

In this section, we first analyze the suitability of inference positions for different targets, i.e., object boundaries and categories, in one-stage object detection. Based on the analysis, we propose a prediction decoupling mechanism to focus different locations on the prediction of different targets. Then, we further devise dynamic points to locate the appropriate positions for collecting predictions. Finally, we elaborate on the network architecture and the details of training and inference.

A. Analysis

The conventional state-of-the-art one-stage detectors generally infer the object locations and categories from the central areas of objects [5, 6]. However, as shown in Fig. 1 the most suitable positions for inferring different targets might differ as well. Directly inferring the object category and boundaries from the same grid location, which has been widely adopted by the conventional one-stage methods, might require rethinkings. To find the optimal inference locations for different targets and build up a more powerful detector, we first conduct some experiments to evaluate the performance of dense classification and regression predictions in object regions.

We train the one-stage detection network [5], where we assign all the grids inside the object bounding box as positive samples for object localization and classification during training. Then, we evaluate the detection network on the validation set [45] and analyze its dense detection results. Specifically, for each object instance with ground-truth bounding box annotation, we sample the grids containing detection results that have IoU > 0.5 w.r.t. the corresponding ground-truth bounding box, and statistically analyze the grid locations where the most accurate predictions are generated for different targets (i.e., the object bounding box boundaries and the object category). The 2D histograms in Fig. 2 exhibit the distributions of the favorable grid locations for different targets. It can be observed that the regions around the object are more suitable for inferring the locations of four sides of the object bounding box, while the areas covering the object tend to have higher confidence in category classification. This phenomenon reveals the different key factors for localization and classification, which also meets the intuition that it would be easier for the areas near the object contour to perceive the boundary, while the object category needs to be identified from the inner semantic regions of the object. Therefore, we argue that the predictions for different targets should be obtained from their more appropriate locations. In the following sections, we will discuss the prediction decoupling mechanism and propose a unified detection framework with the prediction decoupling to push the one-stage detection performance boundary.

B. Prediction Decoupling

Given an input image, one-stage detection networks [5, 6] generate multi-level dense prediction maps containing the category scores and the boundary locations of objects. We let \( P_\tau \) denote a prediction map for a specific target \( \tau \in \{c, l, t, r, b\} \), i.e., either the category \( c \) or the boundary locations for each side indexed by \( l, t, r, \) and \( b \). In the conventional one-stage paradigm, for the object corresponding to the grid \((x, y)\), the prediction result \( R_\tau(x, y) \) for each target \( \tau \) identically comes from the same location in the prediction map \( P_\tau \), i.e., \( R_\tau(x, y) = P_\tau(x, y) \). However, as mentioned above, such prediction manners tend to be sub-optimal, since the prediction results for different targets may need to be obtained from different locations. To this end, we need to learn a map \( G_\tau \) for
each target $\tau$, with which we can locate the suitable location $(x', y')$ to collect the predictions for the object at the current location $(x, y)$. We formulate this collection process as:

$$\begin{align*}
(x', y') &= G_\tau(x, y) \\
R_\tau(x, y) &= P_\tau(x', y'),
\end{align*}$$

(1)

where the prediction result for the object at $(x, y)$ can be collected flexibly from the more appropriate location $(x', y')$ on the prediction map. The above operations essentially assign the tasks of different target predictions to the respective more advantageous locations, and we thus named this mechanism prediction decoupling.

Eq. (1) only allows flexible prediction collection for different targets. However, for each target, multiple prediction results may need to be incorporated for better modeling. Specifically, the object category may need the predictions from different semantic parts to jointly determine, while the boundary locations could be better estimated by choosing the predictions at proper scale levels (from multi-scale localization predictions). Thus, we further extend the above formulation Eq. (1) into a more general version that can utilize multiple predictions for each target:

$$\begin{align*}
\begin{bmatrix} x' \\ y' \end{bmatrix} &= G_\tau(x, y), \quad X' \in \mathbb{R}^{K \times 2} \\
R_\tau(x, y) &= \Phi(P_\tau X'),
\end{align*}$$

(2)

Compared with Eq. (1), here we model each target by $K$ collected predictions from different locations, arranged in a matrix as $P_{\tau, X'} = \begin{bmatrix} x^{(1)}(y^{(1)}), \ldots, x^{(K)}(y^{(K)}) \end{bmatrix}^T \in \mathbb{R}^{K \times 2}$. Specifically, for each target $\tau \in \{c, l, t, r, b\}$, we generate the multiple collection locations as $X' = \begin{bmatrix} x^{(1)}(y^{(1)}) \ldots, x^{(K)}(y^{(K)}) \end{bmatrix}^T \in \mathbb{R}^{K \times 2}$ to obtain the predictions $P_{\tau, X'}$ from the respective prediction maps $\{P^{(i)}_{\tau}\}_{i=1}^{N}$ (for multiple levels or semantic parts, detailed in Section III-C). Then, we use the aggregate function $\Phi(\cdot)$ to produce the final results. Note that for localization and classification targets, we propose different locations to collect and aggregate predictions, which will be elaborated in Section III-C. The prediction collection process incurs almost negligible overhead, which can be easily integrated into the dense detection pipelines [5], [6].

C. Dynamic Points in Prediction Decoupling

We propose to establish two sets of dynamic points to locate the appropriate positions for predicting localization and classification targets. However, directly having the network learn such locations automatically may be difficult, and the optimization process can easily fall into local optimums. To alleviate this, we propose a two-step dynamic point generation module that initializes the dynamic points to the prior positions for different targets and further shifts the points with the residual positional offsets predicted by the network. We will elaborate on the different dynamic point configurations for localization and classification separately in the ensuing parts.

1) Dynamic Boundary Points for Localization: As analyzed in Section III-A, the areas near the object edges are more suitable for boundary localization. Thus we need several dynamic points in these areas to accurately locate the object, which we refer to as dynamic boundary points in the following. To effectively find the appropriate dynamic boundary points for each object, we decompose the point generation into two steps with the dynamic point generation module, which is branched...
from the network’s dense prediction head. Specifically, we first estimate a coarse object box by a convolution layer and initialize the dynamic boundary points at the midpoints of these coarse box boundaries, which should be already close to the object edges. Thereafter, we further adjust the point locations along the coarse boundaries with the offsets generated by another convolution layer (parallel to the one generating the previous coarse box). In this manner, the dynamic boundary points can be further pushed closer to the object edges and obtain the final position \((x_\tau, y_\tau) (\tau \in \{l, t, r, b\})\) for the dynamic boundary point on each side, i.e., the left, top, right, and bottom sides. An example is demonstrated on the right side of Fig. 5 for better understanding.

After having these optimized dynamic boundary points, we collect the respective regression predictions used to pinpoint the boundaries of the object bounding box. Let \(P_l, P_t, P_r, P_b\) denote the respective dense regression maps for the per-pixel positional offsets w.r.t. the four box boundaries of objects. Then, the four boundaries of the final bounding box \(B\) can be obtained as:

\[
B_l = P_l(x_l, y_l) + x_l, \quad B_t = P_t(x_t, y_t) + y_t, \\
B_r = P_r(x_r, y_r) + x_r, \quad B_b = P_b(x_b, y_b) + y_b,
\]

where \(P_{\tau}(x_\tau, y_\tau) (\tau \in \{l, t, r, b\})\) is the collected regression offset to each side of the object bounding box.

Considering the various scales and aspect ratios of objects, the regression map of a specific scale level may be insufficient to perceive and localize the object boundaries well. We hence propose to choose the predictions at the regression maps of more suitable levels to collect the localization results. Concretely, when localizing the object corresponding to the level \(s_0\), we collect the predictions from the current scale level \(s_0\) as well as the adjacent levels (denoted by \(\mathbb{N}(s_0)\)), and select the most confident localization predictions with a differentiable weighting mechanism. With the collected predictions \(P_{\tau} \in \mathbb{R}^{K \times 1}\) (where each element is the prediction \(P_{\tau}(s)(x_\tau(s), y_\tau(s))\) collected from the level \(s \in \mathbb{N}(s_0)\)) and the learned soft weights \(W_{\tau} \in \mathbb{R}^{K \times 1}\) for \(K\) different levels (\(K = \mathbb{N}(s_0)\)), we take an aggregate function \(\Phi(P_{\tau}; W_{\tau}) = W_{\tau}^T P_{\tau} = \sum_{s \in \mathbb{N}(s_0)} W_{\tau}^{(s)} P_{\tau}^{(s)}(x_\tau^{(s)}, y_\tau^{(s)})\) to weight these collected offset predictions from multiple scale levels \(\mathbb{N}(s_0)\), which essentially selects the suitable levels to obtain the final localization results, as shown in Fig. 5. By enhancing Eq. (3) with this weighted multi-level aggregation, we have the regression equation for each side of the bounding box. For instance, the left boundary location is calculated as:

\[
B_l = \Phi(P_l; W_l) + x_l = \sum_{s \in \mathbb{N}(s_0)} W_l^{(s)} P_l^{(s)}(x_l^{(s)}, y_l^{(s)}) + x_l,
\]

where the generated soft weights \(W_l^{(s)}\) are normalized to satisfy \(\sum_{s \in \mathbb{N}(s_0)} W_l^{(s)} = 1\). Compared with Eq. (3), for each obtained dynamic boundary point \((x_l^{(s)}, y_l^{(s)})\), we map it to the adjacent scale levels as \((x_l^{(s)}, y_l^{(s)})\) according to the interpolation rules, to fetch the corresponding prediction results. During training, the dynamic boundary points are pushed towards the positions with better localization results by the regression loss \(L_{reg}\) (Section III-E1). This enables the dynamic boundary points to flexibly adapt to the object silhouettes, which is crucial for achieving accurate bounding box prediction.

2) Dynamic Semantic Points for Classification: From the analysis in Section III-A, the inner regions of an object are more suitable for inferring the class labels, since these regions may contain richer semantic information helpful for determining the category. To efficiently pinpoint these semantic region positions, we define another set of dynamic points, named dynamic semantic points. We also use the dynamic point generation module to generate these semantic points in a two-step manner. First, we borrow the coarse object box estimated in Section III-C1 and initially distribute \(N\) points over the coarse box as the prior positions for the semantic points, as shown in Fig. 5 (right side, \(N = 9\) for demonstration). These prior positions roughly cover different parts of the object. Then, we further generate \(N\) offsets with a convolution layer in the dynamic point generation module to shift these points to positions that better perceive various semantic regions of the object.

In our implementation, we predict \(N\) classification maps \(\{P_c^{(i)}\}_{i=1}^N\) (each with \(C\) channels for \(C\) classes) in parallel to model \(N\) different semantic parts of objects. Each dynamic semantic point that represents a certain semantic part, is associated with a specific dense classification map. Specifically, after obtaining the position \((x_l^{(i)}, y_l^{(i)})\) of the \(i\)-th semantic point, we will collect the class scores voted by this point from its associated classification map, obtaining \(P_c^{(i)}(x_l^{(i)}, y_l^{(i)}) \in \mathbb{R}^C\). To jointly determine the object category from multiple semantic parts, we gather the voting score vectors from \(N\) different point locations \(\{(x_l^{(i)}, y_l^{(i)})\}_{i=1}^N\) as \(P_c = [P_c^{(1)}(x_l^{(1)}, y_l^{(1)}), ..., P_c^{(N)}(x_l^{(N)}, y_l^{(N)})]^T \in \mathbb{R}^{N \times C}\) and aggregate them by function \(\Phi(P_c) = 1^T P_c = \sum_{i=1}^N P_c^{(i)}(x_l^{(i)}, y_l^{(i)})\) with a sigmoid function to produce the final classification results:

\[
s_c = \frac{1}{1 + \exp \left( -\sum_{i=1}^N P_c^{(i)}(x_l^{(i)}, y_l^{(i)}) \right)}
\]

Since the final object score is directly voted from the classification scores of each point, the classification loss can automatically drive the points towards the representative areas where the corresponding object category can be better perceived. This in turn helps us to collect better classification results from the optimized semantic points, making a more confident detection.

D. The Network Architecture

The overall architecture of our detection network is illustrated in Fig. 5. We employ a paradigm similar to other one-stage detectors [2] [5], including an image processing backbone [46], a feature pyramid network [11], and multiple detection heads for multi-scale object detection. In each detection head, following the dense prediction convention, the regression and classification branches produce dense prediction maps. The regression predictions (illustrated as green blocks) are divided along the channel dimension into four regression maps that contain the relative offsets to four sides of objects respectively, which are used for locating the object bounding boxes. Besides, the classification predictions (represented as
yellow blocks in Fig. 3 contain \( N \) classification maps, which model the different semantic parts of objects.

To achieve the prediction decoupling and collection mentioned in Section III-B as shown in Fig. 3 in parallel with the regression branch, we devise a two-step dynamic point generation module to produce the dynamic boundary points and semantic points at each grid as in Section III-C. These two kinds of points are optimized to approach the edges or semantic regions of the target object. After having the densely predicted classification and regression maps, we perform prediction collection guided by these dynamic points, where bilinear interpolation is used to approximate the collected predictions. For the regression maps of multiple scale levels, we use dynamic boundary points to collect the positional offset prediction for each side, with which to produce the final object bounding box. For the classification maps, we incorporate the scores at the dynamic semantic points to jointly identify the object. The overall dense detection results are the combination of bounding boxes and classification scores produced by dynamic point sets at all grid locations. Through prediction decoupling, our proposed network effectively reuses the dense predictions for classification and localization, thereby achieving an accurate and efficient detection.

E. Training and Inference

1) Training: Our detection network is trained with the following loss:

\[
L = L_{cls} + \lambda_1 L_{reg} + \lambda_2 L_{reg_2}
\]

(6)

where \( L_{cls} \) and \( L_{reg} \) are the standard classification and regression losses to supervise the final detection results, while \( L_{reg_2} \) is the additional regression loss to supervise the learning of coarse object boxes used for dynamic point generation. \( \lambda_1 \) and \( \lambda_2 \) are hyper-parameters to balance these losses during training. In our implementation, focal loss \([2]\) is adopted for the classification loss \( L_{cls} \), while GIoU loss \([47]\) is used for the regression losses \( L_{reg} \) and \( L_{reg_2} \).

We calculate the coarse object box regression loss \( L_{reg_2} \) by measuring the difference between each ground-truth bounding box and the corresponding coarse object box predicted from the grid closest to the ground-truth bounding box center. To compute the classification loss \( L_{cls} \) and regression loss \( L_{reg} \), we first find the coarse box predictions with IoU larger than 0.6 w.r.t. the nearest ground-truth bounding box, and assign the dynamic points associated with these coarse boxes as positive samples for different targets, i.e., class label and object boundary. Then, we take the corresponding ground-truth labels to guide the classification and localization predictions from these dynamic points as well as the position learning for these dynamic points.

2) Inference: During inference, the detection network first densely predicts the classification and regression maps from each level of the feature pyramids. Then two sets of dynamic points are generated for each grid location to collect predictions and produce the final classification scores and object bounding boxes. Finally, the non-maximum suppression (NMS) with IoU threshold 0.6 is used to determine the final detection results.

IV. EXPERIMENTS

The experiments are conducted on the challenging MS COCO 2017 benchmark \([45]\). We train the detection model on the train2017 split and evaluate our model on the val2017 split. We also compare with other methods on the test-dev split, which is the official test set without public ground-truth labels for benchmarking purpose.

A. Implementation Details

Following the common experimental conventions \([2, 5, 29]\), we use ResNet-50 \([46]\) with FPN \([11]\) as the backbone in most of our experiments except when comparing with other cutting-edge methods. The backbone has been pre-trained on the ImageNet dataset \([48]\). Our detection model is trained with the synchronized stochastic gradient descent (SGD) on 4 GPUs with 16 images per minibatch. The training procedure lasts for 90k iterations with an initial learning rate of 0.01, which decays by a factor of 10 after 60k iterations and 80k iterations, respectively. The input images are resized to make the shorter edges equal to 800 and the longer sides no larger than 1333. Besides, only random horizontal image flipping is used in data augmentation. Moreover, for Eq. (6), we set \( \lambda_1 = 2.0 \) and \( \lambda_2 = 0.5 \). Unless otherwise specified, we adopt \( N = 9 \) in generating the dynamic semantic points.

B. Ablation Study

1) Prediction Decoupling: To demonstrate the effectiveness of our prediction decoupling mechanism for accurate detection, we conduct a thorough ablation study. The third row of Table I shows our baseline without prediction decoupling mechanism, where the detector follows the prediction manner of the previous one-stage methods ATSS \([29]\) and FCOS \([5]\). This baseline achieves 39.5 AP, similar to the previous one-stage methods listed in the first two rows of Table I. We first individually add the prediction decoupling to the localization or classification branches and find the performance is improved by 1.3 AP and 1.1 AP respectively, as shown in the 4th and 5th rows in Table I. Furthermore, the last row of Table I shows the detection performance of our model with the prediction decoupling applied to both the localization and classification, which improves the baseline from 39.5 AP to 41.8 AP (+2.3 AP) and achieves the best results among all these ablation variants.

| Method | \( D_{loc} \) | \( D_{cls} \) | AP  | AP75 | AP50 | AP90 | AP
|--------|--------------|--------------|-----|------|------|------|---|
| FCOS \([5]\) | 38.6 | 57.4 | 41.4 | 22.3 | 42.3 | 49.8 |
| ATSS \([29]\) | 39.3 | 57.5 | 43.0 | 22.3 | 43.5 | 51.3 |
| Ours: | | | | | | | |
| PDNet ✓ | 39.5 | 57.4 | 43.0 | 22.3 | 43.5 | 52.0 |
| PDNet ✓ | 40.8 | 58.4 | 43.6 | 23.5 | 45.0 | 53.5 |
| PDNet ✓ | 40.6 | 59.1 | 44.2 | 23.4 | 44.3 | 53.0 |
| PDNet ✓ ✓ | 41.8 | 60.0 | 45.1 | 24.7 | 45.8 | 55.2 |

TABLE I

The ablation studies of the prediction decoupling. The \( D_{loc} \) and \( D_{cls} \) refer to applying the prediction decoupling on the localization and classification branches, respectively.
2) Points for localization: To demonstrate the effectiveness of our dynamic boundary points, in Table II we compare different configurations of point positions used to collect the localization predictions for each side of the object bounding box. Specifically, these different configurations include the following as shown in Fig. 4: (a) The original grid point location. (b) A point generated from the grid (with the positional offset predicted by the network). (c) The center point on each boundary of the estimated coarse object box. (d) Our proposed dynamic boundary point. As shown in Table II learning an offset from the original grid improves the AP by 0.2, while the variant with the boundary center point achieves 0.4 AP higher than that with the grid point, indicating that better localization results can be obtained near the object boundaries. By introducing the two-step dynamic point generation strategy (i.e., estimating the coarse boundary centers first and then shifting them with the predicted offsets), our proposed dynamic boundary points can further improve the performance to 41.4 AP and consistently boost the AP of various IOU metrics, which testifies that the two-step generation can better model the object edges. The significant improvement of AP$_{90}$ (+1.8 points) shows the great advantage of our method in high-quality localization. We also evaluate different numbers of dynamic boundary points used for localizing each side of the object bounding box. However, as shown in Table II further increasing the points has little effect on the final performance, which indicates that a single dynamic boundary point is sufficient to model each of the box edges well.

3) Points for classification: Similarly, in Table III we compare the detection performance when applying different configurations of point positions to collect the classification predictions. Fig. 5 presents these different configurations, including the following: (a) The original grid location. (b) A set of divergent points generated from the grid (with N offsets predicted by the network). (c) Our proposed dynamic semantic points. We first evaluate the influence of point numbers N used to collect the predictions. As shown in Table III the AP value increases (from 40.4) as more points are employed until it reaches the saturation point of 41.2 AP near N = 9. This shows that the object category can be better recognized with multiple classification predictions from different semantic parts of the object. With the dynamic semantic points established by the two-step generation process (i.e., predicting the coarse object box first and then the offsets from the prior positions), the performance can be further improved to 41.4 AP. These improvements testify that the two-step dynamic semantic point generation can more easily find various semantic regions of the object and thereby obtain better classification results.
### C. Comparisons with State-of-the-art Methods

We compare our detector PDNet on the test-dev split of MS COCO benchmark with other state-of-the-art methods in Table VI. As in previous works [2, 5], we adopt the multi-scale training strategy by randomly scaling the shorter side of the image to the range from 640 to 800. The training iterations are also doubled to 180k, with the learning rate reduced by 10 times at 120k and 160k iterations respectively. To compare with the methods that adopt a wider scale range [480:960] for multi-scale training, we also apply this strategy in training our detection model for fair comparison. The other settings are kept the same as the previous experiments.

As shown in Table VI, our detector with the ResNet-101 backbone achieves 45.7 AP without bells and whistles, and outperforms other methods using the same backbone, including FCOS [4] (41.5 AP), FreeAnchor [27] (43.1 AP), ATSS [29] (43.6 AP), and GFL [30] (45.0 AP, with a wider training scale range [480:800]). Compared with the recently proposed BorderDet [40] and RepPoints v2 [38], our method also performs better with a much simpler network architecture. With a larger backbone ResNeXt-64x4d-101, we can further improve the AP from 45.7 to 47.4, which is significantly higher than BorderDet [40] (46.5 AP) under the same setting. By utilizing a wider training scale range [480:960], our method with a single ResNeXt-64x4d-101 reaches 48.7 AP, outperforming other methods by an appreciable margin.

| Method       | Backbone   | AP   | AP_{50} | AP_{75} | AP_{S} | AP_{M} | AP_{L} |
|--------------|------------|------|---------|---------|--------|--------|--------|
| FPN [11]     | ResNet-101 | 36.2 | 59.1    | 39.0    | 18.2   | 39.0   | 48.2   |
| Mask R-CNN [14] | ResNet-101 | 38.2 | 60.3    | 41.7    | 20.1   | 41.1   | 50.2   |
| Cascade R-CNN [15] | ResNet-101 | 42.8 | 62.1    | 46.3    | 23.7   | 45.5   | 55.2   |
| RetinaNet [16] | ResNet-101 | 39.1 | 59.1    | 42.3    | 21.8   | 42.7   | 50.2   |
| FoveaBox [17] | ResNet-101 | 40.8 | 61.4    | 44.0    | 24.1   | 45.3   | 53.2   |
| FSAF [18]    | ResNet-101 | 40.9 | 61.5    | 44.0    | 24.0   | 44.2   | 51.3   |
| FCOS [19]    | ResNet-101 | 41.5 | 60.7    | 46.3    | 23.7   | 45.5   | 55.2   |
| FreeAnchor [27] | ResNet-101 | 43.1 | 62.2    | 46.4    | 24.5   | 46.1   | 54.8   |
| FreeAnchor† [27] | ResNeXt-64x4d-101 | 46.0 | 65.6    | 49.8    | 27.8   | 49.5   | 57.7   |
| ATSS [29]    | ResNet-101 | 43.6 | 62.1    | 47.4    | 26.1   | 47.0   | 53.6   |
| ATSS [29]    | ResNeXt-64x4d-101 | 45.6 | 64.6    | 49.7    | 28.5   | 47.0   | 53.6   |
| GFL [30]     | ResNet-101 | 45.0 | 63.7    | 48.9    | 27.2   | 48.8   | 54.5   |
| AlignDet [39] | ResNet-101 | 42.0 | 62.4    | 46.5    | 24.6   | 44.8   | 53.3   |
| RepPoints [37] | ResNet-101-DCN | 45.0 | 66.1    | 49.0    | 26.6   | 48.6   | 57.5   |
| RepPoints v2 [33] | ResNet-101 | 46.0 | 65.3    | 49.5    | 27.4   | 48.9   | 57.3   |
| RepPoints v2† [33] | ResNeXt-64x4d-101 | 47.8 | 67.3    | 51.7    | 29.3   | 50.7   | 59.5   |
| BorderDet [40] | ResNet-101 | 45.4 | 64.1    | 48.8    | 26.7   | 48.3   | 56.5   |
| BorderDet [40] | ResNeXt-64x4d-101 | 46.5 | 65.7    | 50.5    | 29.1   | 49.4   | 57.5   |

| Ours:     |           |      |         |        |        |        |        |
|-----------|-----------|------|---------|--------|--------|--------|--------|
| PDNet     | ResNet-50 | 44.3 | 62.9    | 48.0    | 26.5   | 47.6   | 54.9   |
| PDNet†    | ResNet-50 | 45.0 | 63.5    | 48.6    | 26.9   | 48.4   | 55.9   |
| PDNet     | ResNet-101 | 45.7 | 64.5    | 49.7    | 27.6   | 49.2   | 56.7   |
| PDNet†    | ResNet-101 | 46.6 | 65.3    | 50.6    | 28.0   | 50.2   | 58.0   |
| PDNet     | ResNeXt-64x4d-101 | 47.4 | 66.6    | 51.5    | 29.6   | 50.6   | 58.5   |
| PDNet†    | ResNeXt-64x4d-101 | 48.7 | 67.6    | 52.9    | 30.5   | 52.2   | 60.2   |
D. Visualization of the Regression and Classification Maps

As shown in Fig. 6, we visualize the dense offset regression maps used to locate the left, top, right, and bottom edges of the object bounding box. For clear illustration, we only show the offset predictions in the areas around the object boundary. It can be observed that the positional offsets estimated at the grids near the object edges accurately match the residual distances to the corresponding boundaries of the object bounding box. This provides the foundation for accurate object localization.

In Fig. 7 for the same input image, we present the predicted classification maps for the person. As shown in Fig. 7 these classification maps produce strong activations on different parts of this person, e.g., the head, feet, body, etc. This implies that the different classification maps can model the semantic information of different parts of the object, which allows us to gather predictions from them to jointly identify the object category.

E. Visualization of Detection Results

In Fig. 8 we demonstrate some detection results on the MS COCO val2017 split [45], as well as the dynamic boundary points (in green) and semantic points (in yellow) used to produce these detection results. These results manifest that the predicted dynamic boundary points are located near the object edges where the bounding box boundaries can be better inferred, and the dynamic semantic points are more likely to disperse over different object parts to collect more reasonable classification predictions. For example, in the first image of Fig. 8 the leftmost border of the cat is accurately localized with a dynamic boundary point on its tail, while the semantic points tend to scatter over the cat’s body to comprehensively classify the cat.

F. Efficiency

We evaluate the inference time of our proposed PDNet and other recent dense detection methods for efficiency comparison. All the experiments are conducted using a single NVIDIA 1080Ti GPU. As shown in Table VII our detector runs almost as fast as the one-stage detection methods FCOS [5] and ATSS [29], since our prediction collection process is lightweight and incurs negligible overhead. Compared with the recent RepPoints v2 [38] and BorderDet [40] with more
complicated detection heads, our method achieves a higher AP while being significantly faster, which demonstrates the advantages of our method in both efficiency and precision.

V. CONCLUSION

In this work, we propose an accurate and efficient object detector PDNet that infers different targets (i.e., the object category and boundary locations) at their corresponding appropriate positions. Specifically, based on the dense prediction approach, we propose the PDNet with a prediction decoupling mechanism to flexibly collect different target predictions from different locations and aggregate them for the final detection results. Moreover, we devise two sets of dynamic points, i.e., dynamic boundary points and semantic points, and incorporate a two-step generation strategy to facilitate the learning of suitable inference positions for localization and classification. Extensive experiments on the MS COCO benchmark demonstrate the state-of-the-art performance and efficiency of our method.
[47] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and 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.

[48] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in 2009 IEEE conference on computer vision and pattern recognition. Ieee, 2009, pp. 248–255.