Soft Anchor-Point Object Detection

Chenchen Zhu, Fangyi Chen, Zhiqiang Shen, and Marios Savvides

Carnegie Mellon University, Pittsburgh PA 15213, USA
{chenchez,fangyic,zhiqians,marioss}@andrew.cmu.edu

Abstract. Recently, anchor-free detection methods have been through great progress. The major two families, anchor-point detection and key-point detection, are at opposite edges of the speed-accuracy trade-off, with anchor-point detectors having the speed advantage. In this work, we boost the performance of the anchor-point detector over the key-point counterparts while maintaining the speed advantage. To achieve this, we formulate the detection problem from the anchor point's perspective and identify ineffective training as the main problem. Our key insight is that anchor points should be optimized jointly as a group both within and across feature pyramid levels. We propose a simple yet effective training strategy with soft-weighted anchor points and soft-selected pyramid levels to address the false attention issue within each pyramid level and the feature selection issue across all the pyramid levels, respectively. To evaluate the effectiveness, we train a single-stage anchor-free detector called Soft Anchor-Point Detector (SAPD). Experiments show that our concise SAPD pushes the envelope of speed/accuracy trade-off to a new level, outperforming recent state-of-the-art anchor-free and anchor-based detectors. Without bells and whistles, our best model can achieve a single-model single-scale AP of 47.4% on COCO.

Keywords: Object detection; Anchor-point detector; Soft-weighted anchor points; Soft-selected pyramid levels

1 Introduction

Recently, anchor-free object detectors have drawn a lot of attention [12, 37, 28, 36, 27, 11, 35, 6, 29]. They don't rely on anchor boxes. Predictions are generated in a point(s)-to-box style. Compared to conventional anchor-based approaches [23, 18, 22, 3, 24, 4, 14, 15, 31, 1, 9], anchor-free detectors have a few advantages in general: 1) no manual tuning of hyperparameters for the anchor configuration; 2) usually simpler architecture of detection head; 3) less training memory cost.

The anchor-free detectors can be roughly divided into two categories, i.e. anchor-point detection and key-point detection. Anchor-point detectors, such as [10, 30, 37, 28, 27, 11, 29], encode and decode object bounding boxes as anchor points with corresponding point-to-boundary distances, where the anchor points are the pixels on the pyramidal feature maps and they are associated with the features at their locations just like the anchor boxes. Key-point detectors, such as [12, 36, 6], predict the locations of key points of the bounding

box, e.g. corners, center, or extreme points, using a high-resolution feature map and repeated bottom-up top-down inference [20], and group those key points to form a box. Compared to key-point detectors, anchor-point detectors have several advantages: 1) simpler network architecture; 2) faster training and inference speed; 3) potential to benefit from augmentations on feature pyramids [33, 21, 26]; 4) flexible feature level selection. However, they cannot be as accurate as key-point-based methods under the same image scale of testing.

A natural question to ask is: what hinders a simple anchor-point detector from achieving similar accuracy as key-point detectors? In this work, we push the envelope further: we present Soft Anchor-Point Detector (SAPD), a concise single-stage anchor-point detector with both faster speed and higher accuracy. To achieve this, we formulate the detection problem from the anchor point's perspective and identify ineffective training as the major obstacle impeding anchor-point detector from exploring more potentials of network power both within and across the feature pyramid levels. Specifically, the conventional training strategy has two overlooked issues, i.e. false attention within each pyramid level and feature selection across all pyramid levels. For anchor points on the same pyramid level, those receiving false attention in training will generate detections with unnecessarily high confidence scores but poor localization during inference, suppressing some anchor points with accurate localization but lower score. This can confuse the post-processing step since high-score detections usually have priority to be kept over the low-score ones in non-maximum suppression, resulting in low AP scores at strict IoU thresholds. For anchor points at the same spatial location across different pyramid levels, their associated features are similar but how much they contribute to the network loss is decided without careful consideration. Current methods make the selection based on ad-hoc heuristics like instance scale and usually limited to a single level per instance. This causes a waste of unselected features.

These issues motivate us to propose a novel training strategy with two softened optimization techniques, i.e. soft-weighted anchor points and soft-selected pyramid levels. For anchor points on the same pyramid level, we reduce the false attention by reweighting their contributions to the network loss according to their geometrical relation with the instance box. We argue that the more close to the instance boundaries, the harder for anchor points to localize objects precisely due to feature misalignment, the less they should contribute to the network loss. Additionally, we further reweight an anchor point by the instance-dependent "participation" degree of its pyramid level. We implement a light-weight feature selection network to learn the per-level "participation" degrees given the object instances. The feature selection network is jointly optimized with the detector and not involved in detector inference.

Comprehensive experiments show that the proposed training strategy consistently improves the baseline FSAF [37] module by a large margin *without inference slowdown*, e.g. 2.1% AP increase on COCO [16] detection benchmark with ResNet-50 [8]. The improvements are robust and insensitive to specific hyperparameters and implementations, including advanced feature pyramid de-

signs. With Balanced Feature Pyramid [21], our complete detector achieves the best speed-accuracy balance among recent state-of-the-art anchor-free detectors, see Figure 1. We report single-model single-scale speed/accuracy of SAPD with different backbones, and with or without DCN [4]. The fast variant without DCN outperforms the best key-point detector, CenterNet [6] (45.4% vs. 44.9%) while running about 2× times faster. The accurate variant with DCN forms an upper bound of speed/accuracy trade-offs for recent single-stage and multi-stage detectors, surpassing the accurate TridentNet [13] (47.4% vs. 46.8%) and being more than 3× faster.

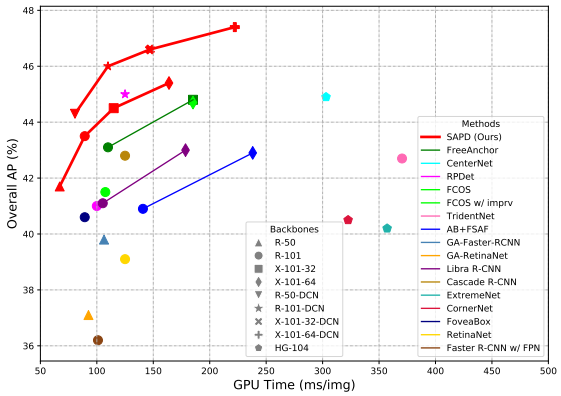

Fig. 1: Single-model single-scale speed (ms) vs. accuracy (AP) on COCO `test-dev`. We show variants of our SAPD with and without DCN [4]. Without DCN, our fastest version can run up to 5× faster than other methods with comparable accuracy. With DCN, our SAPD forms an upper envelop of all recent detectors.

2 Related Work

Anchor-free detectors Despite the dominance of anchor-based methods, anchor-free detectors are continuously under development. Earlier works like DenseBox [10] and UnitBox [30] explore an alternate direction of region proposal. And it has been used in tasks such as scene text detection [34] and pedestrian detection [19]. Recent efforts have pushed the anchor-free detection outperforming the anchor-based counterparts. Most of them are single-stage detectors. For instance, CornerNet [12], ExtremeNet [36] and CenterNet [6] reformulate the detection problem as locating several key points of the bounding boxes. FSAF [37], Guided Anchoring [28], FCOS [27] and FoveaBox [11] encode and decode the bounding boxes as anchor points and point-to-boundary distances. Anchor-free methods can also be in the form of two-stage detectors, such as Guided Anchoring [28] and RPDet [29].

Feature selection in detection Modern object detectors often construct the feature pyramid to alleviate the scale variation problem. With multiple levels in the feature pyramid, selecting the suitable feature level for each instance is a crucial problem. Anchor-based methods make the implicit selection by the anchor matching mechanism, which is based on ad-hoc heuristics like scales and aspect

ratios. Similarly, most anchor-free approaches [28, 27, 11, 29] assign the instances according to scale. The FSAF module [37], on the other hand, makes the assignment by choosing the pyramid level with the minimal instance-dependent loss in a dynamic style during training but limited to one level per instance. In two-stage detectors, some methods consider feature selection in the second stage by feature fusion. PANet [17] proposed the adaptive feature pooling with the element-wise maximize operation. But this requires the input of region proposals for both training and testing, which is not compatible with single-stage methods. Our soft feature selection is designed for single-stage anchor-free methods and can dynamically choose multiple pyramid levels with differentiation.

Soft weighting in detection FCOS [27] predicts the "center-ness" masks and multiplies the confidence scores of anchor points with the masks. But this is in the inference stage so the false attention problem still affects the network training and the extra "center-ness" branches complicate the network architecture. We show that our simple soft-weighting scheme during training is more effective than the "center-ness" masks in the supplementary material. Previous works doing soft weighting in the training stage include Focal Loss [15] and Consistent Loss [25]. They are proposed to reshape the classification loss so that the network down-weights certain samples. But they treat all samples independently. Our training strategy is more direct and comprehensive since we reshape the combination of classification and localization losses and consider jointly weighting a group of anchor points spreading both within and across feature pyramid levels.

3 Soft Anchor-Point Detector

In this section, we present our Soft Anchor-Point Detector (SAPD). First, we formulate the detection problem from the anchor point's perspective in the setting of a vanilla anchor-point detector with a simple head architecture (3.1). Then we introduce our novel training strategy (Figure 3) including soft-weighted anchor points (3.2) and soft-selected pyramid levels (3.3) to address the false attention within pyramid level and feature selection across pyramid levels respectively.

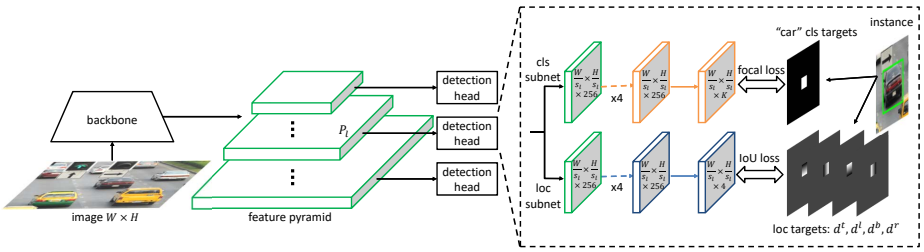

Fig. 2: The network architecture of a vanilla anchor-point detector with a simple detection head.

3.1 Detection Formulation with Anchor Points

The first anchor-point detector can be traced back to DenseBox [10]. The recent modern anchor-point detectors are more or less attaching the detection head of DenseBox with additional convolution layers to multiple levels in the feature pyramids. Here we introduce the general concept of a representative in terms of network architecture, supervision targets, and loss functions.

Network architecture As shown in Figure 2, the network consists of a backbone, a feature pyramid, and one detection head per pyramid level, in a fully convolutional style. A pyramid level is denoted as P_l where l indicates the level number and it has $1/s_l$ resolution of the input image size $W \times H$. s_l is the feature stride and $s_l = 2^l$. A typical range of l is 3 to 7. A detection head has two task-specific subnets, i.e. classification subnet and localization subnet. They both have five 3×3 conv layers. The classification subnet predicts the probability of objects at each anchor point location for each of the K object classes. The localization subnet predicts the 4-dimensional class-agnostic distance from each anchor point to the boundaries of a nearby instance if the anchor point is positive (defined next).

Supervision targets We first define the concept of anchor points. An anchor point p_{lij} is a pixel on the pyramid level P_l located at (i, j) with $i = 0, 1, \ldots, W/s_l - 1$ and $j = 0, 1, \ldots, H/s_l - 1$. Each p_{lij} has a corresponding image space location (X_{lij}, Y_{lij}) where $X_{lij} = s_l(i + 0.5)$ and $Y_{lij} = s_l(j + 0.5)$. Next we define the valid box B_v of a ground-truth instance box $B = (c, x, y, w, h)$ where c is the class id, (x, y) is the box center, and w, h are box width and height respectively. B_v is a central shrunk box of B, i.e. $B_v = (c, x, y, \epsilon w, \epsilon h)$, where ϵ is the shrunk factor. An anchor point p_{lij} is positive if and only if some instance B is assigned to P_l and the image space location (X_{lij}, Y_{lij}) of p_{lij} is inside B_v, otherwise it is a negative anchor point. For a positive anchor point, its classification target is c and localization targets are calculated as the normalized distances $\mathbf{d} = (d^l, d^t, d^r, d^b)$ from the anchor point to the left, top, right, bottom boundaries of B respectively (1),

$$\begin{aligned} d^l &= \frac{1}{z s_l}[X_{lij} - (x - w/2)] \quad d^t = \frac{1}{z s_l}[Y_{lij} - (y - h/2)] \\ d^r &= \frac{1}{z s_l}[(x + w/2) - X_{lij}] \quad d^b = \frac{1}{z s_l}[(y + h/2) - Y_{lij}] \end{aligned} \tag{1}$$

where z is the normalization scalar. For negative anchor points, their classification targets are background ($c = 0$), and localization targets are set to null because they don't need to be learned. To this end, we have a classification target c_{lij} and a localization target $\mathbf{d}_{lij}$ for all of each anchor point p_{lij}. A visualization of the classification targets and the localization targets of one feature level is illustrated in Figure 2.

Loss functions Given the architecture and the definition of anchor points, the network generates a K-dimensional classification output $\hat{\mathbf{c}}_{lij}$ and a 4-dimensional localization output $\hat{\mathbf{d}}_{lij}$ per anchor point p_{lij}. Focal loss [15] (l_{FL}) is adopted for the training of classification subnets to overcome the extreme class imbalance

between positive and negative anchor points. IoU loss [30] (l_{IoU}) is used for the training of localization subnets. Therefore, the per anchor point loss L_{lij} is calculated as Eq. (2).

$$L_{lij} = \begin{cases} l_{FL}(\hat{\mathbf{c}}_{lij}, c_{lij}) + l_{IoU}(\hat{\mathbf{d}}_{lij}, \mathbf{d}_{lij}), & p_{lij} \in p^+ \\ l_{FL}(\hat{\mathbf{c}}_{lij}, c_{lij}), & p_{lij} \in p^- \end{cases} \tag{2}$$

where p^+ and p^- are the sets of positive and negative anchor points respectively. The loss for the whole network is the summation of all anchor point losses divided by the number of positive anchor points (3).

$$L = \frac{1}{N_{p^+}} \sum_{l} \sum_{ij} L_{lij} \tag{3}$$

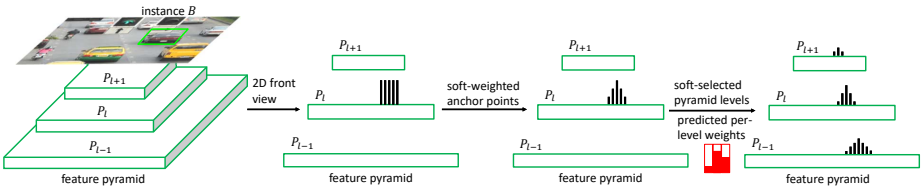

Fig. 3: Illustrative overview of our training strategy with soft-weighted anchor points and soft-selected pyramid levels. The black bars indicate the assigned weights of positive anchor points' contribution to the network loss. The key insight is the joint optimization of anchor points as a group both within and across feature pyramid levels.

3.2 Soft-Weighted Anchor Points

False attention Under the conventional training strategy, we observe that during inference some anchor points generate detection boxes with poor localization but high confidence score, which suppress the boxes with more precise localization but lower score. As a result, the non-maximum suppression (NMS) tends to keep the poorly localized detections, leading to low AP at a strict IoU threshold. We visualize an example of this observation in Figure 4 (a). We plot the detection boxes before NMS with confidence scores indicated by the color. The box with more precise localization of the person is suppressed by other boxes not so accurate but having high scores. Then the final detection (bold box) after NMS doesn't have high IoU with the ground-truth.

So why this is the case? The conventional training strategy treats anchor points independently in Eq. (3), i.e. they receive equal attention. For a group of anchor points inside B_v, their spatial locations and associated features are

different. So their abilities to localize B are also different. We argue that anchor points located close to instance boundaries don't have features well aligned with the instance. Their features tend to be hurt by content outside the instance because their receptive fields include too much information from the background, resulting in less representation power for precise localization. Thus, forcing these anchor points to perform as well as those with powerful feature representation is misleading the network. Less attention should be paid to anchor points close to instance boundaries than those surrounding the center in training. In other words, the network should focus more on optimizing the anchor points with powerful feature representation and reduce the false attention to others.

Our solution To address the false attention issue, we propose a simple and effective soft-weighting scheme. The basic idea is to assign an attention weight w_{lij} for each anchor point's loss L_{lij}. For each positive anchor point, the weight depends on the distance between its image space location and the corresponding instance boundaries. The closer to a boundary, the more down-weighted the anchor point gets. Thus, anchor points close to boundaries are receiving less attention and the network focuses more on those surrounding the center. For negative anchor points, they are kept unchanged since they are not involved in localization, i.e. their weights are all set to 1. Mathematically, w_{lij} is defined in Eq. (4):

$$w_{lij} = \begin{cases} f(p_{lij}, B), & \exists B, p_{lij} \in B_v \\ 1, & \text{otherwise} \end{cases} \tag{4}$$

where f is a function reflecting how close p_{lij} is to the boundaries of B. Closer distance yields less attention weight. We instantiate f using a generalized version of centerness function [27], i.e. $f(p_{lij}, B) = [\frac{\min(d^l_{lij}, d^r_{lij})}{\max(d^l_{lij}, d^r_{lij})} \frac{\min(d^t_{lij}, d^b_{lij})}{\max(d^t_{lij}, d^b_{lij})}]^\eta$, where η controls the decreasing steepness. An illustration of the soft-weighted anchor points is shown in Figure 3.

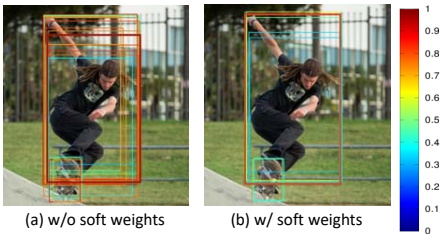

Fig. 4: (a) Poorly localized detection boxes with high scores are generated by anchor points receiving false attention. (b) Our soft-weighting scheme effectively improves localization. Box score is indicated by the color bar.

Fig. 5: Feature responses from P_3 to P_7. They look similar but the details gradually vanish as the resolution becomes smaller. Selecting a single level per instance causes the waste of network power.

3.3 Soft-Selected Pyramid Levels

Feature selection Unlike anchor-based detectors, anchor-free methods don't have constraints from anchor matching to select feature levels for instances from the feature pyramid. In other words, we can assign each instance to arbitrary feature level(s) in anchor-free methods during training. And selecting the right feature levels can make a big difference [37].

We approach the issue of feature selection by looking into the properties of the feature pyramid. Indeed, feature maps from different pyramid levels are somewhat similar to each other, especially the adjacent levels. We visualize the response of all pyramid levels in Figure 5. It turns out that if one level of feature is activated in a certain region, the same regions of adjacent levels may also be activated in a similar style. But the similarity fades as the levels are farther apart. This means that features from more than one pyramid level can participate together in the detection of a particular instance, but the degrees of participation from different levels should be somewhat different.

Inspired by the above analysis, we argue there should be two principles for proper pyramid level selection. Firstly, the selection should be related to the pattern of feature response, rather than some ad-hoc heuristics. And the instance-dependent loss can be a good reflection of whether a pyramid level is suitable for detecting some instances. This principle is also supported by [37]. Secondly, we should allow features from multiple levels involved in the training and testing for each instance, and each level should make distinct contributions. FoveaBox [11] has shown that assigning instances to multiple feature levels can improve the performance to some extent. But assigning to too many levels may instead hurt the performance severely. We believe this limitation is caused by the hard selection of pyramid levels. For each instance, the pyramid levels in FoveaBox are either selected or discarded. The selected levels are treated equally no matter how different their feature responses are.

Therefore, the solution lies in reweighting the pyramid levels for each instance. In other words, a weight is assigned to each pyramid level according to the feature response, making the selection soft. This can also be viewed as assigning a proportion of the instance to a level.

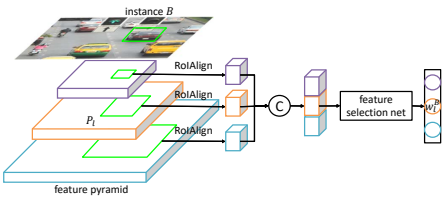

Fig. 6: The weights prediction for soft-selected pyramid levels. "C" indicates the concatenation operation.

layer type	output size	layer setting	activation
input	$1280 \times 7 \times 7$	n/a	n/a
conv	$256 \times 5 \times 5$	$3 \times 3, 256$	relu
conv	$256 \times 3 \times 3$	$3 \times 3, 256$	relu
conv	$256 \times 1 \times 1$	$3 \times 3, 256$	relu
fc	5	n/a	softmax

Table 1: Architecture of the feature selection network. The conv layers have no padding.

Our solution So how to decide the weight of each pyramid level per instance? We propose to train a feature selection network to predict the weights for soft feature selection. The input to the network is instance-dependent feature responses extracted from all the pyramid levels. This is realized by applying the RoIAlign layer [7] to each pyramid feature followed by concatenation, where the RoI is the instance ground-truth box. Then the extracted feature goes through a feature selection network to output a vector of the probability distribution, as shown in Figure 6. We use the probabilities as the weights of soft feature selection.

There are multiple architecture designs for the feature selection network. For simplicity, we present a light-weight instantiation. It consists of three 3×3 conv layers with no padding, each followed by the ReLU function, and a fully-connected layer with softmax. Table 1 details the architecture. The feature selection network is jointly trained with the detector. Cross entropy loss is used for optimization and the ground-truth is a one-hot vector indicating which pyramid level has minimal loss as defined in the FSAF module [37].

So far, each instance B is associated with a per level weight w_l^B via the feature selection network. Together with the soft-weighting scheme in Section 3.2, the anchor point loss L_{lij} is down-weighed further if B is assigned to P_l and p_{lij} is inside B_v. We assign each instance B to top$k$ feature levels with k minimal instance-dependent losses during training. Thus, Eq. (4) is augmented into Eq. (5).

$$w_{lij} = \begin{cases} w_l^B f(p_{lij}, B), & \exists B, p_{lij} \in B_v \\ 1, & \text{otherwise} \end{cases} \tag{5}$$

The total loss of the whole model is the weighted sum of anchor point losses plus the classification loss ($L_{\text{select-net}}$) from the feature selection network, as in Eq. (6). Figure 3 shows the effect of applying soft-selection weights.

$$L = \frac{1}{\sum_{p_{lij} \in p^+} w_{lij}} \sum_{lij} w_{lij} L_{lij} + \lambda L_{\text{select-net}} \tag{6}$$

where λ is the hyperparameter that controls the proportion of classification loss $L_{\text{select-net}}$ for feature selection.

3.4 Implementation Details

Initialization We follow [37] for the initialization of the detection network. Specifically, the backbone networks are pre-trained on ImageNet1k [5]. The classification layers in the detection head are initialized with bias $-\log((1-\pi)/\pi)$ where $\pi = 0.01$, and a Gaussian weight. The localization layers in the detection head are initialized with bias 0.1, and also a Gaussian weight. For the newly added feature selection network, we initialize all layers in it using a Gaussian weight. All the Gaussian weights are filled with $\sigma = 0.01$.

Optimization The entire detection network and the feature selection network are jointly trained with stochastic gradient descent on 8 GPUs with 2 images per

GPU using the COCO `train2017` set [16]. Unless otherwise noted, all models are trained for 12 epochs (∼90k iterations) with an initial learning rate of 0.01, which is divided by 10 at the 9th and the 11th epoch. Horizontal image flipping is the only data augmentation unless otherwise specified. For the first 6 epochs, we don't use the output from the feature selection network. The detection network is trained with the same online feature selection strategy as in the FSAF module [37], i.e. each instance is assigned to only one feature level yielding the minimal loss. We plug in the soft selection weights and choose the top$k$ levels for the second 6 epochs. This is to stabilize the feature selection network first and to make the learning smoother in practice. We use the same training hyper-parameters for the shrunk factor $\epsilon = 0.2$ and the normalization scalar $z = 4.0$ as [37]. We set $\lambda = 0.1$ although results are robust to the exact value.

Inference At the time of inference, the network architecture is as simple as in Figure 2. The feature selection network is *not* involved in the inference so the runtime speed is not affected. An image is forwarded through the network in a fully convolutional style. Then classification prediction $\hat{\mathbf{c}}_{lij}$ and localization prediction $\hat{\mathbf{d}}_{lij}$ are generated for all each anchor point p_{lij}. Bounding boxes can be decoded using the reverse of Eq. (1). We only decode box predictions from at most 1k top-scoring anchor points in each pyramid level, after thresholding the confidence scores by 0.05. These top predictions from all feature levels are merged, followed by non-maximum suppression with a threshold of 0.5, yielding the final detections.

4 Experiments

We conduct experiments on the COCO [16] detection track using the MMDetection [2] codebase. All models are trained on the `train2017` split including around 115k images. We analyze our method by ablation studies on the `val2017` split containing 5k images. When comparing to the state-of-the-art detectors, we report the Average Precision (AP) scores on the `test-dev` split.

	SW	SS	BFP	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
FSAF [37]				35.9	55.0	37.9	19.8	39.6	48.2
	✓			37.0	55.8	39.5	20.5	40.1	48.5
	✓	✓		38.0	56.9	40.5	21.2	41.2	50.2
			✓	36.8	57.4	38.8	21.6	40.9	47.6
SAPD	✓	✓	✓	**38.8**	**58.7**	**41.3**	**22.5**	**42.6**	**50.8**

Table 2: Ablative experiments for the SAPD on the COCO `val2017`. ResNet-50 is the backbone network for all experiments in this table. We study the effect of **SW**: soft-weighted anchor points, **SS**: soft-selected pyramid levels, and **BFP** [21]: augmented feature pyramids.

4.1 Ablation Studies

All results in ablation studies are based on models trained and tested with an image scale of 800 pixels. We study the contribution of each proposed component by gradually applying these components to the baseline FSAF [37] module. For the soft-weighted anchor points and soft-selected pyramid levels, we first study the effect of varying hyperparameters on them and then apply each component with the best hyperparameter to the baseline. We also report more ablative experiments in the supplementary material.

η	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
0.10	36.8	56.2	39.0	20.6	40.3	48.3
0.50	36.9	55.8	39.4	20.2	40.1	48.7
1.0	**37.0**	55.8	39.5	20.5	40.1	48.5
2.0	36.6	55.1	39.1	19.8	40.3	47.8

Table 3: Varying η for the generalized centerness function in soft-weighted anchor points.

top$k$	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
2	37.9	56.9	40.5	21.1	41.0	50.1
3	**38.0**	56.9	40.5	21.2	41.2	50.2
4	37.9	56.9	40.3	21.2	41.1	50.2
5	37.9	56.8	40.5	21.0	41.1	50.2

Table 4: Varying k for number of selected levels in soft-selected pyramid level with $\eta = 1.0$.

Soft-weighted anchor points improve the localization. We first apply the soft-weighting scheme (Eq. (4)) to the training of the baseline FSAF module. Results are reported in Table 2 and 3. Soft-weighted anchor points offer a significant improvement (up to 1.1% AP) over the baseline, being insensitive to various hyperparameters. More importantly, AP_{75} is increased by 1.6%, indicating better localization accuracy at a strict IoU threshold. We also visualize the effect of our soft-weighting scheme in Figure 4(b). The precise box (marked as bold) is kept while the other poorly localized boxes are suppressed, reducing the false attention issue effectively.

Soft-selected pyramid levels utilize the feature power better. Next, we further apply the soft feature selection on top of the soft-weighting scheme, so that each anchor point is down-weighted as in Eq. (5). Table 2 and 4 reports the ablative results. We find that as long as each instance is assigned to more than one pyramid level, we can observe robust $\sim$1.0% absolute AP improvements over the FSAF module plus soft-weighted anchor points. This indicates that allowing instances to optimize multiple pyramid levels is essential to utilize the feature power as much as possible. Empirically, we assign each instance to the top 3 feature levels with the minimal instance-dependent losses according to Table 4. To understand how does the feature selection network assign instances, we visualize the predicted soft selection weights in Figure 7. It turns out that larger instances tend to be assigned high weights for higher pyramid levels. The majority of instances can be learned with no more than two levels. Very rare instances need to be modeled by more than two levels, e.g. the sofa in the top right sub-figure of Figure 7. This is consistent with the results in Table 4.

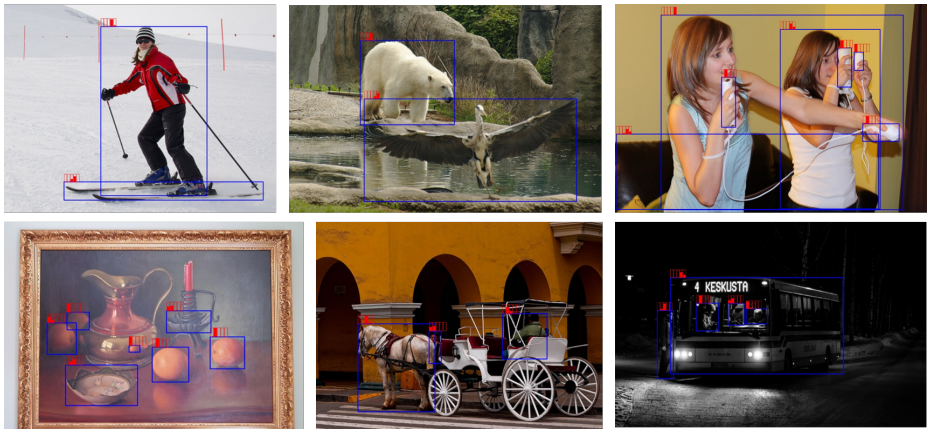

Fig. 7: Visualization of the soft feature selection weights from the feature selection network. Weights (the top-left red bars) ranging from 0 to 1 of five pyramid levels (P_3 to P_7) are predicted for each instance (blue box). The more filled a red bar is, the higher the weight. *Best viewed in color when zoomed in.*

Joint training of the feature selection network has a negligible effect on performance. As shown in Figure 6, the feature selection network takes in feature extracted from the shared feature pyramid and is jointly trained with the detector. One may argue that the performance improvement by the soft-selected pyramid levels is due to the multi-task learning of the feature selection network and the detector. We prove this is not the case. We conduct an experiment in which the feature selection network is jointly trained with the detector but its predicted soft selection weights are not used. In other words, the weights for anchor points are still following Eq. (4) and the feature selection strategy is the same as the baseline FSAF module. It turns out the final AP is 37.1%, only 0.1% higher than the 2nd entry and 0.9% lower than the 3rd entry in Table 2. This means that the major contribution of the soft-selected pyramid levels is actually from *softly selecting multiple levels* rather than the multi-task learning effect from the feature selection network.

Our training strategy works well with augmented feature pyramids. Different from key-point detectors that use a single high-resolution feature map, the SAPD can enjoy the merits brought by the advanced designs of feature pyramids. Here we adopt the Balanced Feature Pyramid (BFP) [21] and achieve further improvement. The BFP pushes our model with ResNet-50 to a 38.8% AP, which is 2.9% higher than the baseline FSAF module. More importantly, our proposed training strategy can robustly work with advanced feature pyramids, offering a steady 2% AP gain (see 4th and 5th entries in Table 2).

SAPD is robust and efficient. Our SAPD can consistently provide robust performance using deeper and better backbone networks, while at the same time keeping the detection head as simple as possible. We report the head-to-head

Backbone	Method	AP	AP_{50}	FPS
R-50	RetinaNet	35.7	54.7	11.6
	AB+FSAF	37.2	57.2	9.0
	SAPD(Ours)	**38.8**	**58.7**	**14.9**
R-101	RetinaNet	37.7	57.2	8.0
	AB+FSAF	39.3	59.2	7.1
	SAPD(Ours)	**41.0**	**60.7**	**11.2**
X-101-64x4d	RetinaNet	39.8	59.5	4.5
	AB+FSAF	41.6	62.4	4.2
	SAPD(Ours)	**43.1**	**63.7**	**6.1**

Table 5: Head-to-head comparisons of anchor-based RetinaNet, anchor-based plus FSAF module, and our purely anchor-free SAPD with different backbone networks on the COCO `val2017` set. **AB**: Anchor-based branches. **R**: ResNet. **X**: ResNeXt.

comparisons with anchor-based RetinaNet and the more complex anchor-based plus FSAF detector in terms of detection accuracy and speed in Table 5. Except for the head architectures, all other settings are the same. All detectors run on a single GTX 1080Ti GPU with CUDA 10 using a batch size of 1. It turns out that our SAPD gets both sides of two worlds. Our SAPD with purely anchor-free heads can not only run faster than the anchor-based counterparts due to simpler head architecture, but also outperform the *combination* of anchor-based and anchor-free heads by significant margins, i.e. 1.6%, 1.7%, and 1.5% absolute AP increases on ResNet-50, ResNet-101, and ResNeXt-101-64x4d backbones respectively.

4.2 Comparison to State of the Art

We evaluate our complete SAPD on the COCO `test-dev` set to compare with recent state-of-the-art anchor-free and anchor-based detectors. All of our models are trained using scale jitter by randomly scaling the shorter side of images in the range from 640 to 800 and for $2\times$ number of epochs as the models in Section 4.1 with the learning rate change points scaled proportionally. Other settings are the same as Section 4.1.

For a fair comparison, we report the results of single-model single-scale testing for all methods, as well as their corresponding inference speeds in Table 6. A visualization of the accuracy-speed trade-off is shown in Figure 1. The inference speeds are measured by Frames-per-Second (FPS) on the same machine with a GTX 1080Ti GPU using a batch size of 1 whenever possible. A "n/a" indicates the case that the method doesn't provide trained models nor self-timing results from the original paper.

Our proposed SAPD pushes the envelope of accuracy-speed boundary to a new level. We report the results of two series of the backbone models, one without DCN and the other with DCN. Without DCN, our fastest SAPD version based on ResNet-50 can reach a 14.9 FPS while maintaining a 41.7% AP, outperforming some of the methods [14, 15, 11, 27, 21] using ResNet-101. With DCN, our SAPD forms an upper envelope of state-of-the-art anchor-free detectors and some recent anchor-based detectors. The closest competitor, RPDet [29], is 1.0% AP worse

Method	Backbone	Anchor free?	FPS	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Multi-stage detectors									
Faster R-CNN w/ FPN [14]	R-101		9.9	36.2	59.1	39.0	18.2	39.0	48.2
Cascade R-CNN [1]	R-101		8.0	42.8	62.1	46.3	23.7	45.5	55.2
GA-Faster-RCNN [28]	R-50	✓	9.4	39.8	59.2	43.5	21.8	42.6	50.7
Libra R-CNN [21]	R-101		9.5	41.1	62.1	44.7	23.4	43.7	52.5
Libra R-CNN [21]	X-101-64x4d		5.6	43.0	64.0	47.0	25.3	45.6	54.6
RPDet [29]	R-101	✓	10.0	41.0	62.9	44.3	23.6	44.1	51.7
RPDet [29]	R-101-DCN	✓	8.0	45.0	66.1	49.0	26.6	48.6	57.5
TridentNet [13]	R-101		2.7	42.7	63.6	46.5	23.9	46.6	56.6
TridentNet [13]	R-101-DCN		1.3	46.8	67.6	51.5	28.0	51.2	60.5
Single-stage detectors									
RetinaNet [15]	R-101		8.0	39.1	59.1	42.3	21.8	42.7	50.2
CornerNet [12]	HG-104	✓	3.1	40.5	56.5	43.1	19.4	42.7	53.9
AB+FSAF [37]	R-101		7.1	40.9	61.5	44.0	24.0	44.2	51.3
AB+FSAF [37]	X-101-64x4d		4.2	42.9	63.8	46.3	26.6	46.2	52.7
GA-RetinaNet [28]	R-50	✓	10.8	37.1	56.9	40.0	20.1	40.1	48.0
ExtremeNet [36]	HG-104	✓	2.8	40.2	55.5	43.2	20.4	43.2	53.1
FoveaBox [11]	X-101	✓	n/a	42.1	61.9	45.2	24.9	46.8	55.6
FCOS [27]	R-101	✓	9.3	41.5	60.7	45.0	24.4	44.8	51.6
FCOS [27] w/ imprv	X-101-64x4d	✓	5.4	44.7	64.1	48.4	27.6	47.5	55.6
CenterNet [6]	HG-104	✓	3.3	44.9	62.4	48.1	25.6	47.4	57.4
FreeAnchor [32]	R-101		9.1	43.1	62.2	46.4	24.5	46.1	54.8
FreeAnchor [32]	X-101-32x8d		5.4	44.8	64.3	48.4	27.0	47.9	56.0
SAPD (Ours)	R-50	✓	14.9	41.7	61.9	44.6	24.1	44.6	51.6
SAPD (Ours)	R-101	✓	11.2	43.5	63.6	46.5	24.9	46.8	54.6
SAPD (Ours)	X-101-64x4d	✓	6.1	45.4	65.6	48.9	27.3	48.7	56.8
SAPD (Ours)	R-50-DCN	✓	12.4	44.3	64.4	47.7	25.5	47.3	57.0
SAPD (Ours)	R-101-DCN	✓	9.1	46.0	65.9	49.6	26.3	49.2	59.6
SAPD (Ours)	X-101-64x4d-DCN	✓	4.5	47.4	67.4	51.1	28.1	50.3	61.5

Table 6: *Single-model and single-scale* accuracy and inference speed of SAPD vs. recent state-of-the-art detectors on the COCO `test-dev` set. FPS is measured on the same machine with a single *GTX 1080Ti* GPU using the official source code whenever possible. "n/a" means that both trained models and timing results from original papers are not available. **R**: ResNet. **X**: ResNeXt. **HG**: Hourglass. For a visualized comparison, please refer to Figure 1.

and 15ms slower than ours. Compared to key-point anchor-free detectors [6, 36, 12] using Hourglass, our SAPD enjoys significantly faster inference speed (up to 5× times) and a 2.5% AP improvement (47.4% vs. 44.9%) over the best key-point detector, CenterNet [6].

5 Conclusion

This work studied the anchor-point object detection and discovered the key insight lies in the joint optimization of a group of anchor points both within and across the feature pyramid levels. We proposed a novel training strategy addressing two underexplored issues of anchor-point detection approaches, i.e. the false attention issue within each pyramid level and the feature selection issue across all pyramid levels. Applying our training strategy to a simple anchor-point detector leads to a new upper envelope of the speed-accuracy trade-off.

References

1. Cai, Z., Vasconcelos, N.: Cascade r-cnn: Delving into high quality object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 6154–6162 (2018)
2. Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C.C., Lin, D.: MMDetection: Open mmlab detection toolbox and benchmark. arXiv preprint arXiv:1906.07155 (2019)
3. Dai, J., Li, Y., He, K., Sun, J.: R-fcn: Object detection via region-based fully convolutional networks. In: Advances in neural information processing systems. pp. 379–387 (2016)
4. Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., Wei, Y.: Deformable convolutional networks. In: Proceedings of the IEEE international conference on computer vision. pp. 764–773 (2017)
5. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
6. Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., Tian, Q.: Centernet: Object detection with keypoint triplets. In: Proceedings of the IEEE International Conference on Computer Vision (2019)
7. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 2961–2969 (2017)
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
9. He, Y., Zhu, C., Wang, J., Savvides, M., Zhang, X.: Bounding box regression with uncertainty for accurate object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
10. Huang, L., Yang, Y., Deng, Y., Yu, Y.: Densebox: Unifying landmark localization with end to end object detection. arXiv preprint arXiv:1509.04874 (2015)
11. Kong, T., Sun, F., Liu, H., Jiang, Y., Shi, J.: Foveabox: Beyond anchor-based object detector. arXiv preprint arXiv:1904.03797 (2019)
12. Law, H., Deng, J.: Cornernet: Detecting objects as paired keypoints. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 734–750 (2018)
13. Li, Y., Chen, Y., Wang, N., Zhang, Z.: Scale-aware trident networks for object detection. In: Proceedings of the IEEE International Conference on Computer Vision (2019)
14. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2117–2125 (2017)
15. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. pp. 2980–2988 (2017)
16. 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: European conference on computer vision. pp. 740–755. Springer (2014)
17. Liu, S., Qi, L., Qin, H., Shi, J., Jia, J.: Path aggregation network for instance segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8759–8768 (2018)

18. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: European conference on computer vision. pp. 21–37. Springer (2016)
19. Liu, W., Liao, S., Ren, W., Hu, W., Yu, Y.: High-level semantic feature detection: A new perspective for pedestrian detection. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
20. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: European conference on computer vision. pp. 483–499. Springer (2016)
21. Pang, J., Chen, K., Shi, J., Feng, H., Ouyang, W., Lin, D.: Libra r-cnn: Towards balanced learning for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 821–830 (2019)
22. Redmon, J., Farhadi, A.: Yolo9000: better, faster, stronger. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7263–7271 (2017)
23. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems. pp. 91–99 (2015)
24. Shen, Z., Liu, Z., Li, J., Jiang, Y.G., Chen, Y., Xue, X.: Dsod: Learning deeply supervised object detectors from scratch. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1919–1927 (2017)
25. Sun, F., Kong, T., Huang, W., Tan, C., Fang, B., Liu, H.: Feature pyramid reconfiguration with consistent loss for object detection. IEEE Transactions on Image Processing **28**(10), 5041–5051 (2019)
26. Sun, K., Xiao, B., Liu, D., Wang, J.: Deep high-resolution representation learning for human pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5693–5703 (2019)
27. Tian, Z., Shen, C., Chen, H., He, T.: Fcos: Fully convolutional one-stage object detection. In: Proceedings of the IEEE International Conference on Computer Vision (2019)
28. Wang, J., Chen, K., Yang, S., Loy, C.C., Lin, D.: Region proposal by guided anchoring. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2965–2974 (2019)
29. Yang, Z., Liu, S., Hu, H., Wang, L., Lin, S.: Reppoints: Point set representation for object detection. In: Proceedings of the IEEE International Conference on Computer Vision (2019)
30. Yu, J., Jiang, Y., Wang, Z., Cao, Z., Huang, T.: Unitbox: An advanced object detection network. In: Proceedings of the 24th ACM international conference on Multimedia. pp. 516–520. ACM (2016)
31. Zhang, S., Wen, L., Bian, X., Lei, Z., Li, S.Z.: Single-shot refinement neural network for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4203–4212 (2018)
32. Zhang, X., Wan, F., Liu, C., Ji, R., Ye, Q.: Freeanchor: Learning to match anchors for visual object detection. In: Advances in neural information processing systems (2019)
33. Zhao, Q., Sheng, T., Wang, Y., Tang, Z., Chen, Y., Cai, L., Ling, H.: M2det: A single-shot object detector based on multi-level feature pyramid network. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 9259–9266 (2019)
34. Zhong, Z., Sun, L., Huo, Q.: An anchor-free region proposal network for faster r-cnn based text detection approaches. arXiv preprint arXiv:1804.09003 (2018)
35. Zhou, X., Wang, D., Krähenbühl, P.: Objects as points. arXiv preprint arXiv:1904.07850 (2019)

36. Zhou, X., Zhuo, J., Krahenbuhl, P.: Bottom-up object detection by grouping extreme and center points. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 850–859 (2019)
37. Zhu, C., He, Y., Savvides, M.: Feature selective anchor-free module for single-shot object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)