Soft Anchor-Point Object Detection

Chenchen Zhu Fangyi Chen Zhiqiang Shen Marios Savvides
Carnegie Mellon University
{chenchez, fangyic, zhiqians, marioss}@andrew.cmu.edu

Abstract

Recently, anchor-free detectors have shown great potential to outperform anchor-based detectors in terms of both accuracy and speed. In this work, we aim at finding a new balance of speed and accuracy for anchor-free detectors. Two questions are studied: 1) how to make the anchor-free detection head better? 2) how to utilize the power of feature pyramid better? We identify attention bias and feature selection as the main issues for these two questions respectively. We propose to address these issues with a novel training strategy that has two soften optimization techniques, i.e. soft-weighted anchor points and soft-selected pyramid levels. 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-based and anchor-free, single-stage and multi-stage detectors. Without bells and whistles, our best model can achieve a single-model single-scale AP of 47.4% on COCO. Our fastest version can run up to 5× faster than other detectors with comparable accuracy.

1. Introduction

Deep learning approaches have achieved remarkable performance in object detection [22, 17, 21, 3, 23, 4, 13, 14, 29, 1], among which most adopt the anchor boxes to address the scale and ratio variation of objects. The anchor boxes have manually selected scales and aspect ratios according to the statistical characteristics of the data. During training, these anchor boxes are treated as references to learn their relations with the ground-truth bounding boxes. In inference, anchor boxes are refined based on the learned relations, so that predictions are generated in a box-to-box style.

Recently, anchor-free object detectors have drawn a lot of attention [11, 35, 26, 34, 25, 10, 33, 6, 27]. They don't rely on anchor boxes. Predictions are generated in a point(s)-to-box style. Compared to conventional anchor-based approaches, 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) potential to outperform anchor-based counterparts in terms of both accuracy and speed.

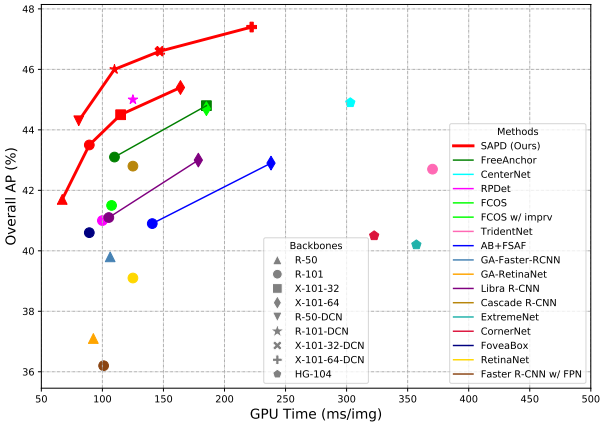


Figure 1. Single-model single-scale speed (ms) vs. accuracy (AP) on COCO `test-dev`. We show variants of our SAPD with and without DCN [4]. Without DCN, our fastest version can run up to 5× faster than other methods with comparable accuracy. With DCN, our SAPD forms an upper envelop of all recent detectors.

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 [9, 28, 35, 26, 25, 10], 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 [11, 34, 6], predict the locations of several key points of the bounding box, e.g. corners, center, or extreme points, and group those key points to form a box.

The above two types of anchor-free methods both have advantages and limitations. Indeed, key-point detectors can achieve relatively high average precision (AP) with relatively small input image size. But they rely on a *single* high-resolution feature map and repeated bottom-up top-down inference [19], thus suffering from high FLOPs and memory consumption, long training and testing time, as

well as low compatibility with popular pretrained backbones. On the contrary, 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 [31, 20, 24]; 4) flexible feature level selection. However, they cannot be as accurate as key-point-based methods under the same image scale of testing, especially less localization accuracy.

A natural question to ask is: could a simple anchor-point detector achieve 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 and more accurate performance. To achieve this, we identify ineffective training as the major obstacle impeding anchor-point detector from achieving state-of-the-art accuracy. Specifically, the current training strategy for anchor-point detector has two overlooked issues, i.e. attention bias and feature selection. By attention bias we mean objects with good views tend to draw more attention from the detector, making other objects being easily missed, due to the ignorance of anchor points' feature misalignment in training. And the feature selection issue lies in assigning instances to pyramid levels based on ad-hoc heuristics or limited to single level per instance, resulting in suboptimal utilization of the feature power. These issues inspire us proposing a novel training strategy with two soften optimization techniques, i.e. soft-weighted anchor points and soft-selected pyramid levels. A positive anchor point is reweighted in the contribution to the network loss based on its distance to the object center and the soft feature selection weights of the instance to which it belongs. A meta-selection network, jointly trained with the detector, is proposed to predict the soft selection weights of the pyramid levels for each instance.

Comprehensive experiments show that the proposed training techniques improve the baseline FSAF [35] module by a large margin *without inference slowdown under the same learning schedule*, e.g. 2.1% AP increase on COCO [15] detection benchmark with ResNet-50 [8]. Unlike key-point detectors, our SAPD can be further improved by adopting better feature pyramid designs. With Balanced Feature Pyramid [20], our complete detector achieves the best speed-accuracy balance among recent state-of-the-art anchor-free and anchor-based 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], by a considerable margin (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 all recent single-stage and multi-stage detectors, surpassing the accurate TridentNet [12] (47.4% vs. 46.8%) and being more than 3× faster.

2. Related Work

Anchor-based detectors Starting from Faster R-CNN [22] and SSD [17], region-based CNN detectors start to adopt anchor boxes as priors for region proposal. Depending on whether there is an additional refinement after region proposal, modern detection frameworks can be broadly classified into two categories: single-stage detectors like YOLO [21], RetinaNet [14], DSOD [23], RefineDet [29], FreeAnchor [30] and multi-stage detectors, including R-FCN [3], FPN [13], Cascade R-CNN [1], Libra R-CNN [20], TridentNet [12], etc.

Anchor-free detectors Despite the dominance of anchor-based methods, anchor-free detectors are continuously under development. Earlier works like DenseBox [9] and UnitBox [28] explore an alternate direction of region proposal. And it has been used in tasks such as scene text detection [32] and pedestrian detection [18]. Recent efforts have pushed the anchor-free detection outperforming the anchor-based counterparts. Most of them are single-stage detectors. For instance, CornerNet [11], ExtremeNet [34] and CenterNet [6] reformulate the detection problem as locating several key points of the bounding boxes. FSAF [35], Guided Anchoring [26], FCOS [25] and FoveaBox [10] encode and decode the bounding boxes as anchor points and point-to-boundary distances. Anchor-free methods can also be applied to two-stage detectors, such as Guided Anchoring [26] and RPDet [27].

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 [26, 25, 10, 27] assign the instances according to scale. The FSAF module [35], on the other hand, makes the assignment by choosing the pyramid level with the minimal instance-dependent loss in a dynamic style during training, which outperforms the heuristic-based selection. In two-stage detectors, some methods consider feature selection in the second stage by feature fusion. PANet [16] 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.

3. Soft Anchor-Point Detector

In this section, we present our Soft Anchor-Point Detector (SAPD). First, we review the basic architecture and optimization of a vanilla anchor-point detector (3.1), which can be seen as the representative of modern anchor-point detectors. Then we introduce our novel training strategy (Figure

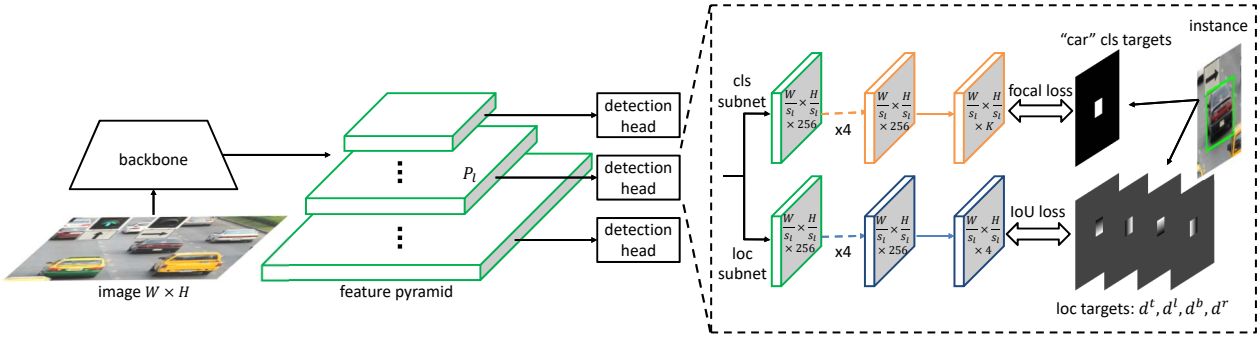


Figure 2. The network architecture of a vanilla anchor-point detector. Details are described in Section 3.1.

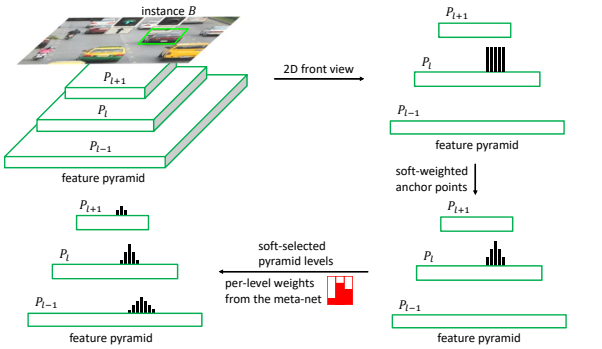


Figure 3. Overview of our training strategy with soft-weighted anchor points and soft-selected pyramid levels. The black bars indicate the assigned weights of positive anchor points' contribution to the network loss.

3) to address the overlooked issues, i.e. attention bias and feature selection. It consists of soft-weighted anchor points (3.2) and soft-selected pyramid levels (3.3), which answer how to make the detection head better and how to utilize the pyramid power better respectively.

3.1. Preliminary on Anchor-Point Detectors

The first anchor-point detector can be traced back to DenseBox [9]. 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, loss functions, and inference.

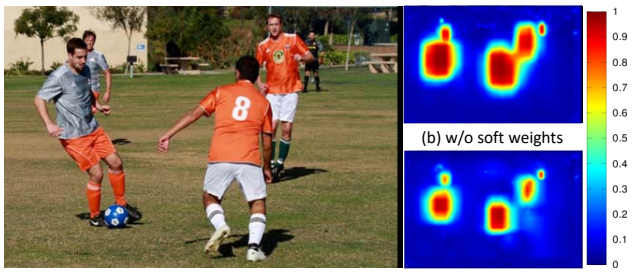
Network architecture As shown in Figure 2, the network consists of a backbone, a feature pyramid, and one detection head per pyramid level, in a fully convolutional style. A pyramid level is denoted as P_l where l indicates the level number and it has $1/s_l$ resolution of the input image size $W \times H$. 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 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),

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

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 [14] (l_{FL}) is



(a) Input image (c) w/ soft weights

Figure 4. Visualization of the confidence score maps (b) and (c) from the classification outputs, given input image (a). (b) and (c) are from detectors optimized w/o and w/ the soft-weighing scheme respectively. (b) reveals the attention bias issue.

adopted for the training of classification subnets to overcome the extreme class imbalance between positive and negative anchor points. IoU loss [28] (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).

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

3.2. Soft-Weighted Anchor Points

Attention bias In natural images, objects may appear under various challenging conditions like occlusion, cluttered background. We find that the vanilla anchor-point detector will have the attention bias issue when handling these conditions, meaning the objects with clear and conspicuous views generate too large high score regions that suppress the score regions of other objects appear as the surrounding context. We visualize an example of this attention bias in Figure 4. The figure contains five soccer players and its score map from classification output is shown in Figure 4(b). The two players in the foreground generate two large dominant regions with high scores in the heatmap. The large regions tend to expand towards the other underrepresented regions corresponding to the remaining players. In worse cases, the dominant regions can even cover the parts of the underrepresented regions. This causes the detector to have biased attention to only foreground instances, suppressing the detection of background instances.

So why is this the case? We argue the problem lies in the unnecessarily high scores at the locations close to the instance boundary due to feature misalignment. Anchor points located close to 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. Therefore it is unreasonable to trust an anchor point close to the instance boundary as much as an anchor point close to the center.

Figure 5. Feature responses from P_3 to P_7. They look similar but the details gradually vanish as the resolution becomes smaller.

Our solution To address the attention bias issue, we propose a soft-weighting scheme. The basic idea is to assign a weight w_{lij} for each anchor point p_{lij}. For each positive anchor point, the weight depends on the distance between its image space location and the corresponding instance center. The longer the distance, the more down-weighted the anchor point gets. Thus, anchor points far away from the center are suppressed, relying more on those close to the center. For negative anchor points, they are kept unchanged, i.e. their weights are all set to 1. This is different from the key-point detectors where only one key point is positive. An illustration is shown in Figure 3.

There are a few choices to design the soft-weighting scheme, as long as the weight is a monotonously decreasing function of the distance between the anchor point and instance center. To this end, we propose a generalized centerness function defined in Eq. (4).

$$w_{lij} = \begin{cases} [\frac{\min(d^l_{lij}, d^r_{lij})}{\max(d^l_{lij}, d^r_{lij})} \frac{\min(d^t_{lij}, d^b_{lij})}{\max(d^t_{lij}, d^b_{lij})}]^\eta, & p_{lij} \in p^+ \\ 1, & p_{lij} \in p^- \end{cases} \tag{4}$$

where η controls the decreasing steepness and the value of w_{lij} is always between 0 and 1. Eq. (4) guarantees that anchor points on the instance boundaries have zero weight and anchor point at the instance center has weight of 1.

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/levels in anchor-free methods during

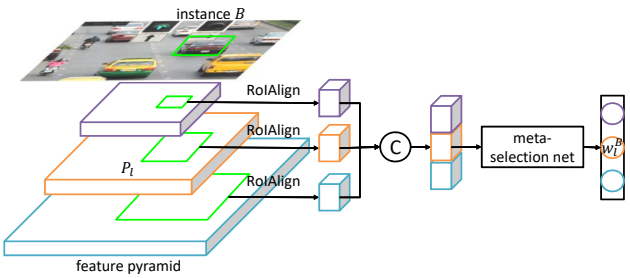


Figure 6. Overview of the weights prediction for soft-selected pyramid levels. "C" indicates the concatenation operation.

training. And selecting the right feature levels can make a big difference [35].

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 contribute together to the detection of a particular instance, but the extent of contributions 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 needs to follow 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 [35]. 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 [10] 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 re-weighting 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.

Our solution So how to decide the weight of each pyramid level per instance? We propose to train a meta-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 the meta-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.

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 meta-selection network.

The meta-selection network is light-weight. 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 meta-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 [35]. Note that there may be other architecture designs for the meta-selection network. Here we just present a simple instantiation.

So far, each instance B is associated with a per level weight w_l^B via the meta-selection network. Together with the soft-weighting scheme in Section 3.2, the anchor point p_{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).

$$w_{lij} = \begin{cases} w_l^B [\frac{\min(d^l_{lij}, d^r_{lij})}{\max(d^l_{lij}, d^r_{lij})} \frac{\min(d^t_{lij}, d^b_{lij})}{\max(d^t_{lij}, d^b_{lij})}]^\eta, & \exists B, p_{lij} \in B_v \\ 1, & \text{otherwise} \end{cases} \tag{5}$$

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

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

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

3.4. Implementation Details

Initialization We follow [35] 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

	SW	SS	BFP	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	AR_1	AR_{10}	AR_{100}	AR_S	AR_M	AR_L
FSAF				35.9	55.0	37.9	19.8	39.6	48.2	30.6	49.1	52.3	31.8	56.8	67.3
	✓			37.0	55.8	39.5	20.5	40.1	48.5	31.7	51.5	54.5	33.7	59.2	70.0
	✓	✓		38.0	56.9	40.5	21.2	41.2	50.2	**32.0**	52.0	55.0	34.4	59.2	70.5
SAPD	✓	✓	✓	**38.8**	**58.7**	**41.3**	**22.5**	**42.6**	**50.8**	31.9	**52.4**	**55.7**	**35.6**	**60.5**	**70.6**

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** [20]: augmented feature pyramids.

0.1, and also a Gaussian weight. For the newly added meta-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 meta-selection network are jointly trained with stochastic gradient descent on 8 GPUs with 2 images per GPU using the COCO `train2017` set [15]. 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 meta-selection network. The detection network is trained with the same online feature selection strategy as in the FSAF module [35], 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 meta-selection network first and to make the learning smoother in practice. We use the same training hyperparameters for the shrunk factor $\epsilon = 0.2$ and the normalization scalar $z = 4.0$ as [35]. 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 meta-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 [15] 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.

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
0.25	36.9	56.1	39.5	20.3	40.4	48.8
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 $\eta = 1.0$.

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 [35] detector. 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.

Soft-weighted anchor points improve the detection head. 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. It is evident that soft-weighted anchor points can offer a significant improvement (up to 1.1% AP and 1.6% AP_{75}) over the baseline under a wide range of hyperparameter choices. As for the hyperparameter η, the performance reaches the best when $\eta = 1.0$, which is different from the "center-ness" in [25] where $\eta = 0.5$. So $\eta = 1.0$ for all of the following experiments. We also visualize the classification heatmap in Figure 4(c). The figure clearly shows that the high score regions become more compact to the center of instances, reducing the overlapping between dominant regions and underrepresented regions. With our soft-weighting scheme, the attention bias issue is effectively alleviated and each detection head be-

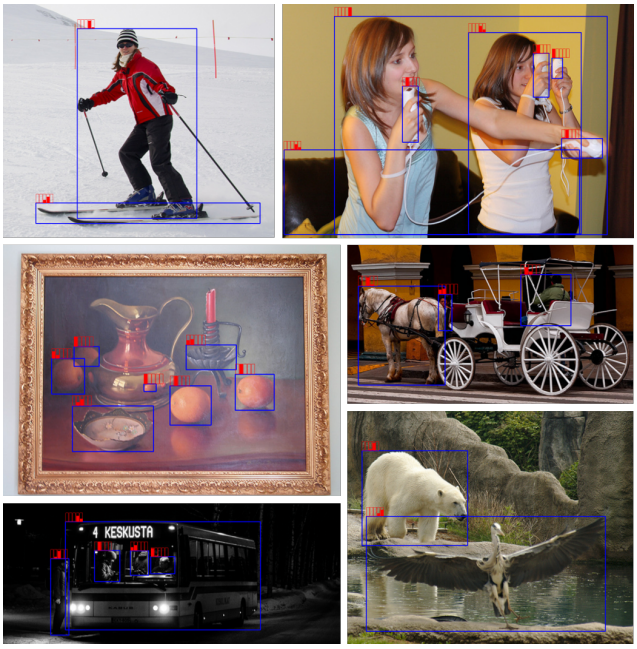


Figure 7. Visualization of the soft feature selection weights from the meta-selection network. Weights (the top-left red bars) ranging from 0 to 1 of five pyramid levels (P_3 to P_7) are predicted for each instance (blue box). The more filled a red bar is, the higher the weight. *Best viewed in color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

comes better.

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 ∼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 meta-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.

Joint training of the meta-selection network has a negligible effect on performance. As shown in Figure 6, the meta-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 meta-selection network and the detector. We prove this is not the case. We conduct an experiment in which the meta-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 meta-selection network.

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.

SAPD benefits from augmented feature pyramids. Different from key-point anchor-free 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) [20] 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.

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 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 training and testing 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.

Method	Backbone	Anchor free?	FPS	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Multi-stage detectors									
Faster R-CNN w/ FPN [13]	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 [26]	R-50	✓	9.4	39.8	59.2	43.5	21.8	42.6	50.7
Libra R-CNN [20]	R-101		9.5	41.1	62.1	44.7	23.4	43.7	52.5
Libra R-CNN [20]	X-101-64x4d		5.6	43.0	64.0	47.0	25.3	45.6	54.6
RPDet [27]	R-101	✓	10.0	41.0	62.9	44.3	23.6	44.1	51.7
RPDet [27]	R-101-DCN	✓	8.0	45.0	66.1	49.0	26.6	48.6	57.5
TridentNet [12]	R-101		2.7	42.7	63.6	46.5	23.9	46.6	56.6
TridentNet [12]	R-101-DCN		1.3	46.8	67.6	51.5	28.0	51.2	60.5
Single-stage detectors									
RetinaNet [14]	R-101		8.0	39.1	59.1	42.3	21.8	42.7	50.2
CornerNet [11]	HG-104	✓	3.1	40.5	56.5	43.1	19.4	42.7	53.9
AB+FSAF [35]	R-101		7.1	40.9	61.5	44.0	24.0	44.2	51.3
AB+FSAF [35]	X-101-64x4d		4.2	42.9	63.8	46.3	26.6	46.2	52.7
GA-RetinaNet [26]	R-50	✓	10.8	37.1	56.9	40.0	20.1	40.1	48.0
ExtremeNet [34]	HG-104	✓	2.8	40.2	55.5	43.2	20.4	43.2	53.1
FoveaBox [10]	R-101	✓	11.2	40.6	60.1	43.5	23.3	45.2	54.5
FoveaBox [10]	X-101	✓	n/a	42.1	61.9	45.2	24.9	46.8	55.6
FCOS [25]	R-101	✓	9.3	41.5	60.7	45.0	24.4	44.8	51.6
FCOS [25] w/ imprv	X-101-64x4d	✓	5.4	44.7	64.1	48.4	27.6	47.5	55.6
CenterNet [6]	HG-52	✓	4.4	41.6	59.4	44.2	22.5	43.1	54.1
CenterNet [6]	HG-104	✓	3.3	44.9	62.4	48.1	25.6	47.4	57.4
FreeAnchor [30]	R-101		9.1	43.1	62.2	46.4	24.5	46.1	54.8
FreeAnchor [30]	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-101-32x4d	✓	8.7	44.5	64.7	47.8	26.5	47.8	55.8
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-101-32x4d-DCN	✓	6.8	46.6	66.6	50.0	27.3	49.7	60.7
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. **DCN**: Deformable Convolutional Network

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 [13, 14, 10, 25, 20] using ResNet-101. With DCN, our

SAPD forms an upper envelope of recent state-of-the-art anchor-based and anchor-free detectors. The closest competitor, RPDet [27], is 1.0% AP worse and 15ms slower than ours. Compared to key-point anchor-free detectors [6, 34, 11] 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 paper proposed a novel training strategy addressing two underexplored issues of anchor-free object detection, i.e. attention bias and feature selection. Applying our strategy to the vanilla anchor-point detector leads to a new level of balance between speed and accuracy.

References

[1] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6154–6162, 2018. 1, 2, 8

[2] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 6

[3] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, pages 379–387, 2016. 1, 2

[4] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017. 1, 2

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 5

[6] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Object detection with keypoint triplets. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 1, 2, 8, 9

[7] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 5

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[9] Lichao Huang, Yi Yang, Yafeng Deng, and Yinan Yu. Densebox: Unifying landmark localization with end to end object detection. *arXiv preprint arXiv:1509.04874*, 2015. 1, 2, 3

[10] Tao Kong, Fuchun Sun, Huaping Liu, Yuning Jiang, and Jianbo Shi. Foveabox: Beyond anchor-based object detector. *arXiv preprint arXiv:1904.03797*, 2019. 1, 2, 5, 8

[11] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 734–750, 2018. 1, 2, 8, 9

[12] Yanghao Li, Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Scale-aware trident networks for object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 2, 8

[13] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 1, 2, 8

[14] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 1, 2, 3, 8

[15] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 2, 6

[16] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8759–8768, 2018. 2

[17] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016. 1, 2

[18] Wei Liu, Shengcai Liao, Weiqiang Ren, Weidong Hu, and Yinan Yu. High-level semantic feature detection: A new perspective for pedestrian detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[19] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499. Springer, 2016. 1

[20] Jiangmiao Pang, Kai Chen, Jianping Shi, Huajun Feng, Wanli Ouyang, and Dahua Lin. Libra r-cnn: Towards balanced learning for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 821–830, 2019. 2, 6, 7, 8

[21] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017. 1, 2

[22] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 1, 2

[23] Zhiqiang Shen, Zhuang Liu, Jianguo Li, Yu-Gang Jiang, Yurong Chen, and Xiangyang Xue. Dsod: Learning deeply

supervised object detectors from scratch. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1919–1927, 2017. 1, 2

[24] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5693–5703, 2019. 2

[25] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 1, 2, 6, 8

[26] Jiaqi Wang, Kai Chen, Shuo Yang, Chen Change Loy, and Dahua Lin. Region proposal by guided anchoring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2965–2974, 2019. 1, 2, 8

[27] Ze Yang, Shaohui Liu, Han Hu, Liwei Wang, and Stephen Lin. Reppoints: Point set representation for object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 1, 2, 8, 9

[28] Jiahui Yu, Yuning Jiang, Zhangyang Wang, Zhimin Cao, and Thomas Huang. Unitbox: An advanced object detection network. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 516–520. ACM, 2016. 1, 2, 4

[29] Shifeng Zhang, Longyin Wen, Xiao Bian, Zhen Lei, and Stan Z Li. Single-shot refinement neural network for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4203–4212, 2018. 1, 2

[30] Xiaosong Zhang, Fang Wan, Chang Liu, Rongrong Ji, and Qixiang Ye. Freeanchor: Learning to match anchors for visual object detection. In *Advances in neural information processing systems*, 2019. 2, 8

[31] Qijie Zhao, Tao Sheng, Yongtao Wang, Zhi Tang, Ying Chen, Ling Cai, and Haibin Ling. M2det: A single-shot object detector based on multi-level feature pyramid network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9259–9266, 2019. 2

[32] Zhuoyao Zhong, Lei Sun, and Qiang Huo. An anchor-free region proposal network for faster r-cnn based text detection approaches. *arXiv preprint arXiv:1804.09003*, 2018. 2

[33] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 1

[34] Xingyi Zhou, Jiacheng Zhuo, and Philipp Krahenbuhl. Bottom-up object detection by grouping extreme and center points. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 850–859, 2019. 1, 2, 8, 9

[35] Chenchen Zhu, Yihui He, and Marios Savvides. Feature selective anchor-free module for single-shot object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 2, 5, 6, 8