Feature Selective Anchor-Free Module for Single-Shot Object Detection

Chenchen Zhu Yihui He Marios Savvides
Carnegie Mellon University
{chenchez, he2, marioss}@andrew.cmu.edu

a: RetinaNet (anchor-based, ResNeXt-101) b: Ours (anchor-based + FSAF, ResNet-50)

Figure 1: Qualitative results of the anchor-based RetinaNet [22] using powerful *ResNeXt-101* (left) and our detector with additional FSAF module using just *ResNet-50* (right) under the same training and testing scale. Our FSAF module helps detecting hard objects like tiny person and flat skis with a less powerful backbone network. See Figure 7 for more examples.

Abstract

We motivate and present feature selective anchor-free (FSAF) module, a simple and effective building block for single-shot object detectors. It can be plugged into single-shot detectors with feature pyramid structure. The FSAF module addresses two limitations brought up by the conventional anchor-based detection: 1) heuristic-guided feature selection; 2) overlap-based anchor sampling. The general concept of the FSAF module is online feature selection applied to the training of multi-level anchor-free branches. Specifically, an anchor-free branch is attached to each level of the feature pyramid, allowing box encoding and decoding in the anchor-free manner at an arbitrary level. During training, we dynamically assign each instance to the most suitable feature level. At the time of inference, the FSAF module can work jointly with anchor-based branches by outputting predictions in parallel. We instantiate this concept with simple implementations of anchor-free branches and online feature selection strategy. Experimental results on the COCO detection track show that our FSAF module performs better than anchor-based counterparts while being faster. When working jointly with anchor-based branches, the FSAF module robustly improves the baseline RetinaNet by a large margin under various settings, while introducing nearly free inference overhead. And the resulting best model can achieve a state-of-the-art 44.6% mAP, outperforming all existing single-shot detectors on COCO.

1. Introduction

Object detection is an important task in the computer vision community. It serves as a prerequisite for various downstream vision applications such as instance segmentation [12], facial analysis [1, 39], autonomous driving cars [6, 20], and video analysis [25, 33]. The performance of object detectors has been dramatically improved thanks to the advance of deep convolutional neural networks [16, 29, 13, 34] and well-annotated datasets [7, 23].

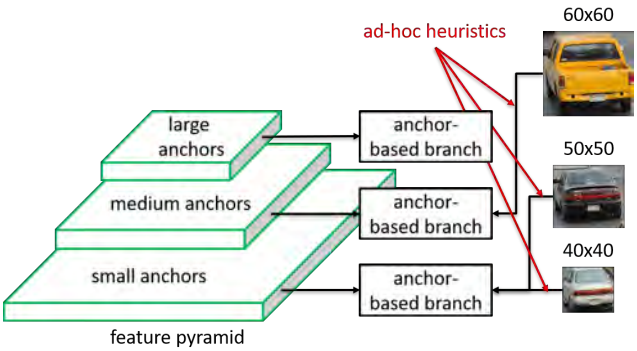

Figure 2: Selected feature level in anchor-based branches may not be optimal.

One challenging problem for object detection is scale variation. To achieve scale invariability, state-of-the-art detectors construct feature pyramids or multi-level feature towers [24, 8, 21, 22, 19, 38]. And multiple scale levels of feature maps are generating predictions in parallel. Besides, anchor boxes can further handle scale variation [24, 28]. Anchor boxes are designed for discretizing the continuous space of all possible instance boxes into a finite number of boxes with predefined locations, scales and aspect ratios. And instance boxes are matched to anchor boxes based on the Intersection-over-Union (IoU) overlap. When integrated with feature pyramids, large anchor boxes are typically associated with upper feature maps, and small anchor boxes are associated with lower feature maps, see Figure 2. This is based on the heuristic that upper feature maps have more semantic information suitable for detecting big instances whereas lower feature maps have more fine-grained details suitable for detecting small instances [11]. The design of feature pyramids integrated with anchor boxes has achieved good performance on object detection benchmarks [7, 23, 9].

However, this design has two limitations: 1) heuristic-guided feature selection; 2) overlap-based anchor sampling. During training, each instance is always matched to the closest anchor box(es) according to IoU overlap. And anchor boxes are associated with a certain level of feature map by human-defined rules, such as box size. Therefore, the selected feature level for each instance is purely based on *ad-hoc heuristics*. For example, a car instance with size 50×50 pixels and another similar car instance with size 60×60 pixels may be assigned to two different feature levels, whereas another 40×40 car instance may be assigned to the same level as the 50×50 instance, as illustrated in Figure 2. In other words, the anchor matching mechanism is inherently heuristic-guided. This leads to a major flaw that the selected feature level to train each instance may not be optimal.

We propose a simple and effective approach named feature selective anchor-free (FSAF) module to address these two limitations simultaneously. Our motivation is to let each instance select the best level of feature freely to optimize the network, so there should be no anchor boxes to constrain the feature selection in our module. Instead, we encode the instances in an anchor-free manner to learn the parameters for classification and regression. The general concept is presented in Figure 3. An anchor-free branch is built per level of feature pyramid, independent to the anchor-based branch. Similar to the anchor-based branch, it consists of a classification subnet and a regression subnet (not shown in figure). An instance can be assigned to arbitrary level of the anchor-free branch. During training, we dynamically select the most suitable level of feature for each instance based on the instance content instead of just the size of instance box. The selected level of feature then learns to detect the assigned instances. At inference, the FSAF module can run independently or jointly with anchor-based branches. Our FSAF module is agnostic to the backbone network and can be applied to single-shot detectors with a structure of feature pyramid. Additionally, the instantiation of anchor-free branches and online feature selection can be various. In this work, we keep the implementation of our FSAF module simple so that its computational cost is marginal compared to the whole network.

Extensive experiments on the COCO [23] object detection benchmark confirm the effectiveness of our method. The FSAF module by itself outperforms anchor-based counterparts as well as runs faster. When working jointly with anchor-based branches, the FSAF module can consistently improve the strong baselines by large margins across various backbone networks, while at the same time introducing the minimum cost of computation. Especially, we improve RetinaNet using ResNeXt-101 [34] by **1.8%** with only **6ms** additional inference latency. Additionally, our final detector achieves a state-of-the-art **44.6**% mAP when multi-scale testing are employed, outperforming all existing single-shot detectors on COCO.

2. Related Work

Recent object detectors often use feature pyramid or multi-level feature tower as a common structure. SSD [24] first proposed to predict class scores and bounding boxes from multiple feature scales. FPN [21] and DSSD [8] proposed to enhance low-level features with high-level semantic feature maps at all scales. RetinaNet [22] addressed class imbalance issue of multi-level dense detectors with focal loss. DetNet [19] designed a novel backbone network to maintain high spatial resolution in upper pyramid levels. However, they all use pre-defined anchor boxes to encode and decode object instances. Other works address the scale variation differently. Zhu et al [41] enhanced the anchor design for small objects. He et al [14] modeled the bounding

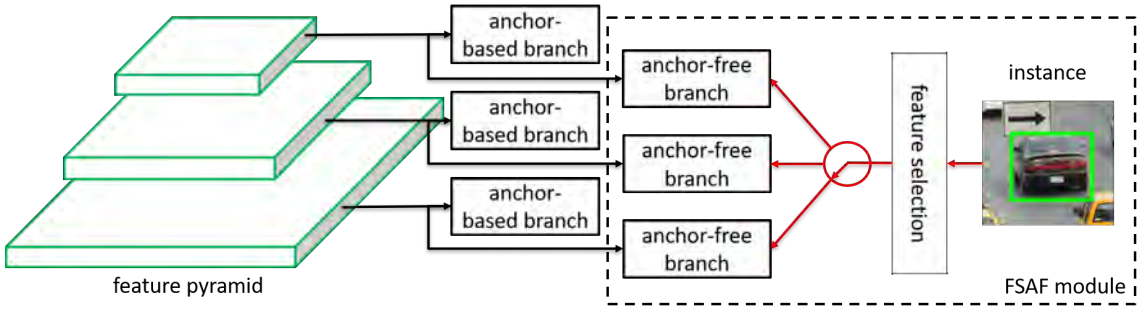

Figure 3: Overview of our FSAF module plugged into conventional anchor-based detection methods. During training, each instance is assigned to a pyramid level via feature selection for setting up supervision signals.

box as Gaussian distribution for improved localization.

The idea of anchor-free detection is not new. DenseBox [15] first proposed a unified end-to-end fully convolutional framework that directly predicted bounding boxes. UnitBox [36] proposed an Intersection over Union (IoU) loss function for better box regression. Zhong et al [40] proposed anchor-free region proposal network to find text in various scales, aspect ratios, and orientations. Recently CornerNet [17] proposed to detect an object bounding box as a pair of corners, leading to the best single-shot detector. SFace [32] proposed to integrate the anchor-based method and anchor-free method. However, they still adopt heuristic feature selection strategies.

3. Feature Selective Anchor-Free Module

In this section we instantiate our feature selective anchor-free (FSAF) module by showing how to apply it to the single-shot detectors with feature pyramids, such as SSD [24], DSSD [8] and RetinaNet [22]. Without lose of generality, we apply the FSAF module to the state-of-the-art RetinaNet [22] and demonstrate our design from the following aspects: 1) how to create the anchor-free branches in the network (3.1); 2) how to generate supervision signals for anchor-free branches (3.2); 3) how to dynamically select feature level for each instance (3.3); 4) how to jointly train and test anchor-free and anchor-based branches (3.4).

3.1. Network Architecture

From the network's perspective, our FSAF module is surprisingly simple. Figure 4 illustrates the architecture of the RetinaNet [22] with the FSAF module. In brief, RetinaNet is composed of a backbone network (not shown in the figure) and two task-specific subnets. The feature pyramid is constructed from the backbone network with levels from P_3 through P_7, where l is the pyramid level and P_l has $1/2^l$ resolution of the input image. Only three levels are shown for simplicity. Each level of the pyramid is used for detecting objects at a different scale. To do this, a classification subnet and a regression subnet are attached to P_l. They are both small fully convolutional networks. The classification subnet predicts the probability of objects at each spatial location for each of the A anchors and K object classes. The regression subnet predicts the 4-dimensional class-agnostic offset from each of the A anchors to a nearby instance if exists.

On top of the RetinaNet, our FSAF module introduces only two additional conv layers per pyramid level, shown as the dashed feature maps in Figure 4. These two layers are responsible for the classification and regression predictions in the anchor-free branch respectively. To be more specific, a 3×3 conv layer with K filters is attached to the feature map in the classification subnet followed by the sigmoid function, in parallel with the one from the anchor-based branch. It predicts the probability of objects at each spatial location for K object classes. Similarly, a 3×3 conv layer with four filters is attached to the feature map in the regression subnet followed by the ReLU [26] function. It is responsible for predicting the box offsets encoded in an anchor-free manner. To this end the anchor-free and anchor-based branches work jointly in a multi-task style, sharing the features in every pyramid level.

3.2. Ground-truth and Loss

Given an object instance, we know its class label k and bounding box coordinates $b = [x, y, w, h]$, where (x, y) is the center of the box, and w, h are box width and height respectively. The instance can be assigned to arbitrary feature level P_l during training. We define the projected box $b_p^l = [x_p^l, y_p^l, w_p^l, h_p^l]$ as the projection of b onto the feature pyramid P_l, i.e. $b_p^l = b/2^l$. We also define the effective box $b_e^l = [x_e^l, y_e^l, w_e^l, h_e^l]$ and the ignoring box $b_i^l = [x_i^l, y_i^l, w_i^l, h_i^l]$ as proportional regions of b_p^l controlled by constant scale factors ϵ_e and ϵ_i respectively, i.e. $x_e^l = x_p^l, y_e^l = y_p^l, w_e^l = \epsilon_e w_p^l, h_e^l = \epsilon_e h_p^l$, $x_i^l = x_p^l, y_i^l = y_p^l, w_i^l = \epsilon_i w_p^l, h_i^l = \epsilon_i h_p^l$. We set $\epsilon_e = 0.2$ and $\epsilon_i = 0.5$. An example of ground-truth generation for a car instance is

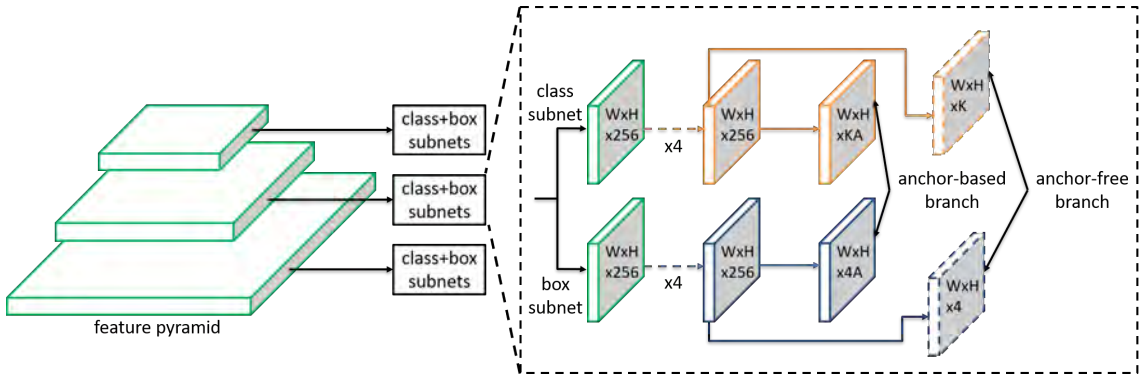

Figure 4: Network architecture of RetinaNet with our FSAF module. The FSAF module only introduces two additional conv layers (dashed feature maps) per pyramid level, keeping the architecture fully convolutional.

illustrated in Figure 5.

Classification Output: The ground-truth for the classification output is K maps, with each map corresponding to one class. The instance affects kth ground-truth map in three ways. First, the effective box b_e^l region is the positive region filled by ones shown as the white box in "car" class map, indicating the existence of the instance. Second, the ignoring box excluding the effective box $(b_i^l - b_e^l)$ is the ignoring region shown as the grey area, which means that the gradients in this area are not propagated back to the network. Third, the ignoring boxes in adjacent feature levels (b_i^{l-1}, b_i^{l+1}) are also ignoring regions if exists. Note that if the effective boxes of two instances overlap in one level, the smaller instance has higher priority. The rest region of the ground-truth map is the negative (black) area filled by zeros, indicating the absence of objects. Focal loss [22] is applied for supervision with hyperparameters $\alpha = 0.25$ and $\gamma = 2.0$. The total classification loss of anchor-free branches for an image is the summation of the focal loss over all non-ignoring regions, normalized by the total number of pixels inside all effective box regions.

Box Regression Output: The ground-truth for the regression output are 4 offset maps agnostic to classes. The instance only affects the b_e^l region on the offset maps. For each pixel location (i, j) inside b_e^l, we represent the projected box b_p^l as a 4-dimensional vector $\mathbf{d}_{i,j}^l = [d_{t_{i,j}}^l, d_{l_{i,j}}^l, d_{b_{i,j}}^l, d_{r_{i,j}}^l]$, where d_t^l, d_l^l, d_b^l, d_r^l are the distances between the current pixel location (i, j) and the top, left, bottom, and right boundaries of b_p^l, respectively. Then the 4-dimensional vector at (i, j) location across 4 offset maps is set to $\mathbf{d}_{i,j}^l / S$ with each map corresponding to one dimension. S is a normalization constant and we choose $S = 4.0$ in this work empirically. Locations outside the effective box are the grey area where gradients are ignored. IoU loss [36] is adopted for optimization. The total regression loss of anchor-free branches for an image is the average of the IoU loss over all effective box regions.

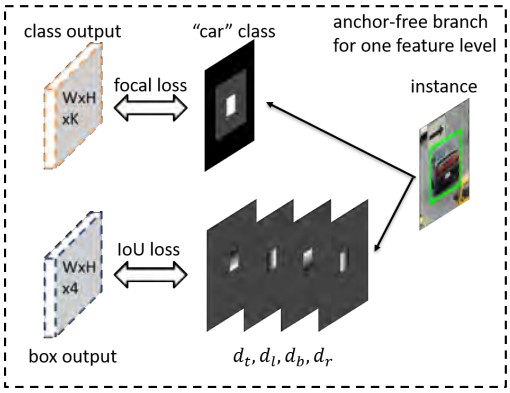

Figure 5: Supervision signals for an instance in one feature level of the anchor-free branches. We use focal loss for classification and IoU loss for box regression.

During inference, it is straightforward to decode the predicted boxes from the classification and regression outputs. At each pixel location (i, j), suppose the predicted offsets are $[\hat{o}_{t_{i,j}}, \hat{o}_{l_{i,j}}, \hat{o}_{b_{i,j}}, \hat{o}_{r_{i,j}}]$. Then the predicted distances are $[S\hat{o}_{t_{i,j}}, S\hat{o}_{l_{i,j}}, S\hat{o}_{b_{i,j}}, S\hat{o}_{r_{i,j}}]$. And the top-left corner and the bottom-right corner of the predicted projected box are $(i - S\hat{o}_{t_{i,j}}, j - S\hat{o}_{l_{i,j}})$ and $(i + S\hat{o}_{b_{i,j}}, j + S\hat{o}_{r_{i,j}}])$ respectively. We further scale up the projected box by 2^l to get the final box in the image plane. The confidence score and class for the box can be decided by the maximum score and the corresponding class of the K-dimensional vector at location (i, j) on the classification output maps.

3.3. Online Feature Selection

The design of the anchor-free branches allows us to learn each instance using the feature of an arbitrary pyramid level P_l. To find the optimal feature level, our FSAF module selects the best P_l based on the instance content, instead of the size of instance box as in anchor-based methods.

Given an instance I, we define its classification loss and

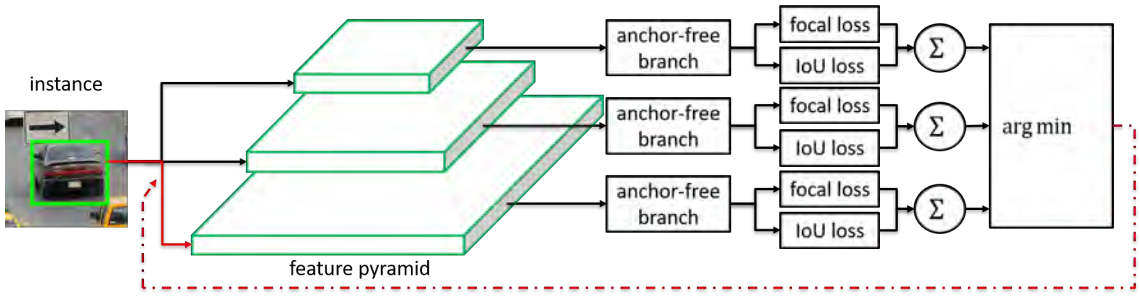

Figure 6: Online feature selection mechanism. Each instance is passing through all levels of anchor-free branches to compute the averaged classification (focal) loss and regression (IoU) loss over effective regions. Then the level with minimal summation of two losses is selected to set up the supervision signals for that instance.

box regression loss on P_l as $L^I_{FL}(l)$ and $L^I_{IoU}(l)$, respectively. They are computed by averaging the focal loss and the IoU loss over the effective box region b^l_e, i.e.

$$L^I_{FL}(l) = \frac{1}{N(b^l_e)} \sum_{i,j \in b^l_e} FL(l,i,j)$$
$$L^I_{IoU}(l) = \frac{1}{N(b^l_e)} \sum_{i,j \in b^l_e} IoU(l,i,j) \quad (1)$$

where $N(b^l_e)$ is the number of pixels inside b^l_e region, and $FL(l,i,j)$, $IoU(l,i,j)$ are the focal loss [22] and IoU loss [36] at location (i,j) on P_l respectively.

Figure 6 shows our online feature selection process. First the instance I is forwarded through all levels of feature pyramid. Then the summation of $L^I_{FL}(l)$ and $L^I_{IoU}(l)$ is computed in all anchor-free branches using Eqn. (1). Finally, the best pyramid level P_{l^*} yielding the minimal summation of losses is selected to learn the instance, i.e.

$$l^* = \arg\min_l L^I_{FL}(l) + L^I_{IoU}(l) \quad (2)$$

For a training batch, features are updated for their correspondingly assigned instances. The intuition is that the selected feature is currently the best to model the instance. Its loss forms a lower bound in the feature space. And by training, we further pull down this lower bound. At the time of inference, we do not need to select the feature because the most suitable level of feature pyramid will naturally output high confidence scores.

In order to verify the importance of our online feature selection, we also conduct a heuristic feature selection process for comparison in the ablation studies (4.1). The heuristic feature selection depends purely on box sizes. We borrow the idea from the FPN detector [21]. An instance I is assigned to the level $P_{l'}$ of the feature pyramid by:

$$l' = \lfloor l_0 + \log_2(\sqrt{wh}/224) \rfloor \quad (3)$$

Here 224 is the canonical ImageNet pre-training size, and l_0 is the target level on which an instance with $w \times h = 224^2$ should be mapped into. In this work we choose $l_0 = 5$ because ResNet [13] uses the feature map from 5th convolution group to do the final classification.

3.4. Joint Inference and Training

When plugged into RetinaNet [22], our FSAF module works jointly with the anchor-based branches, see Figure 4. We keep the anchor-based branches as original, with all hyperparameters unchanged in both training and inference.

Inference: The FSAF module just adds a few convolution layers to the fully-convolutional RetinaNet, so the inference is still as simple as forwarding an image through the network. For anchor-free branches, we only decode box predictions from at most 1k top-scoring locations in each pyramid level, after thresholding the confidence scores by 0.05. These top predictions from all levels are merged with the box predictions from anchor-based branches, followed by non-maximum suppression with a threshold of 0.5, yielding the final detections.

Initialization: The backbone networks are pre-trained on ImageNet1k [5]. We initialize the layers in RetinaNet as in [22]. For conv layers in our FSAF module, we initialize the classification layers with bias $-\log((1-\pi)/\pi)$ and a Gaussian weight filled with $\sigma = 0.01$, where π specifies that at the beginning of training every pixel location outputs objectness scores around π. We set $\pi = 0.01$ following [22]. All the box regression layers are initialized with bias b, and a Gaussian weight filled with $\sigma = 0.01$. We use $b = 0.1$ in all experiments. The initialization helps stabilize the network learning in the early iterations by preventing large losses.

Optimization: The loss for the whole network is combined losses from the anchor-free and anchor-based branches. Let L^{ab} be the total loss of the original anchor-based RetinaNet. And let L^{af}_{cls} and L^{af}_{reg} be the total classification and regression losses of anchor-free branches, respectively. Then total optimization loss is $L = L^{ab} + \lambda(L^{af}_{cls} + L^{af}_{reg})$, where λ controls the weight of the anchor-free branches. We set $\lambda = 0.5$ in all experiments, although results are robust to the exact

	Anchor-based branches	Anchor-free branches		AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
		Heuristic feature selection Eqn. (3)	Online feature selection Eqn. (2)						
RetinaNet	✓			35.7	54.7	38.5	19.5	39.9	47.5
Ours		✓		34.7	54.0	36.4	19.0	39.0	45.8
			✓	35.9	55.0	37.9	19.8	39.6	48.2
	✓	✓		36.1	55.6	38.7	19.8	39.7	48.9
	✓		✓	**37.2**	**57.2**	**39.4**	**21.0**	**41.2**	**49.7**

Table 1: Ablative experiments for the FSAF module on the COCO `minival`. ResNet-50 is the backbone network for all experiments in this table. We study the effect of anchor-free branches, heuristic feature selection, and online feature selection.

Backbone	Method	AP	AP_{50}	Runtime (ms/im)
R-50	RetinaNet	35.7	54.7	131
	Ours(FSAF)	35.9	55.0	107
	Ours(AB+FSAF)	37.2	57.2	138
R-101	RetinaNet	37.7	57.2	172
	Ours(FSAF)	37.9	58.0	148
	Ours(AB+FSAF)	39.3	59.2	180
X-101	RetinaNet	39.8	59.5	356
	Ours(FSAF)	41.0	61.5	288
	Ours(AB+FSAF)	41.6	62.4	362

Table 2: Detection accuracy and inference latency with different backbone networks on the COCO `minival`. **AB**: Anchor-based branches. **R**: ResNet. **X**: ResNeXt.

value. The entire network is trained with stochastic gradient descent (SGD) on 8 GPUs with 2 images per GPU. Unless otherwise noted, all models are trained for 90k iterations with an initial learning rate of 0.01, which is divided by 10 at 60k and again at 80k iterations. Horizontal image flipping is the only applied data augmentation unless otherwise specified. Weight decay is 0.0001 and momentum is 0.9.

4. Experiments

We conduct experiments on the detection track of the COCO dataset [23]. The training data is the COCO `trainval35k` split, including all 80k images from `train` and a random 35k subset of images from the 40k `val` split. We analyze our method by ablation studies on the `minival` split containing the remaining 5k images from `val`. When comparing to the state-of-the-art methods, we report COCO AP on the `test-dev` split, which has no public labels and requires the use of the evaluation server.

4.1. Ablation Studies

For all ablation studies, we use an image scale of 800 pixels for both training and testing. We evaluate the contribution of several important elements to our detector, including anchor-free branches, online feature selection, and backbone networks. Results are reported in Table 1 and 2.

Anchor-free branches are necessary. We first train two detectors with *only* anchor-free branches, using two feature selection methods respectively (Table 1 2nd and 3rd entries). It turns out anchor-free branches only can already achieve decent results. When jointly optimized with anchor-based branches, anchor-free branches help learning instances which are hard to be modeled by anchor-based branches, leading to improved AP scores (Table 1 5th entry). Especially the AP_{50}, AP_S and AP_L scores increase by 2.5%, 1.5%, and 2.2% respectively with online feature selection. To find out what kinds of objects the FSAF module can detect, we show some qualitative results of the head-to-head comparison between RetinaNet and ours in Figure 7. Clearly, our FSAF module is better at finding challenging instances, such as tiny and very thin objects which are not well covered by anchor boxes.

Online feature selection is essential. As stated in Section 3.3, we can select features in anchor-free branches either based on heuristics just like the anchor-based branches, or based on instance content. It turns out selecting the right feature to learn plays a fundamental role in detection. Experiments show that anchor-free branches with heuristic feature selection (Eqn. (3)) only are not able to compete with anchor-based counterparts due to less learnable parameters. But with our online feature selection (Eqn. (2)), the AP is improved by **1.2%** (Table 1 3rd vs 2nd entries), which overcomes the parameter disadvantage. Additionally, Table 1 4th and 5th entries further confirm that our online feature selection is essential for anchor-free and anchor-based branches to work well together.

How is optimal feature selected? In order to understand the optimal pyramid level selected for instances, we visualize some qualitative detection results from only the anchor-free branches in Figure 8. The number before the class name indicates the feature level that detects the object. It turns out the online feature selection actually follows the rule that upper levels select larger instances, and lower levels are responsible for smaller instances, which

Figure 7: More qualitative comparison examples between anchor-based RetinaNet (top, Table 1 1st entry) and our detector with additional FSAF module (bottom, Table 1 5th entry). Both are using ResNet-50 as backbone. Our FSAF module helps finding more challenging objects.

Figure 8: Visualization of online feature selection from anchor-free branches. The number before the class name is the pyramid level that detects the instance. We compare this level with the level to which as if this instance is assigned in the anchor-based branches, and use *red* to indicate the disagreement and *green* for agreement.

is the same principle in anchor-based branches. However, there are quite a few exceptions, *i.e.* online feature selection chooses pyramid levels different from the choices of anchor-based branches. We label these exceptions as red boxes in Figure 8. Green boxes indicate agreement between the FSAF module and anchor-based branches. By capturing these exceptions, our FSAF module can use better features to detect challenging objects.

FSAF module is robust and efficient. We also evaluate the effect of backbone networks to our FSAF module in terms of accuracy and speed. Three backbone networks include ResNet-50, ResNet-101 [13], and ResNeXt-101 [34]. Detectors run on a single Titan X GPU with CUDA 9 and CUDNN 7 using a batch size of 1. Results are reported in

Method	Backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Multi-shot detectors							
CoupleNet [42]	ResNet-101	34.4	54.8	37.2	13.4	38.1	50.8
Faster R-CNN+++ [28]		34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w/ FPN [21]		36.2	59.1	39.0	18.2	39.0	48.2
Regionlets [35]		39.3	59.8	n/a	21.7	43.7	50.9
Fitness NMS [31]		41.8	60.9	44.9	21.5	45.0	57.5
Cascade R-CNN [3]		42.8	62.1	46.3	23.7	45.5	55.2
Deformable R-FCN [4]	Aligned-Inception-ResNet	37.5	58.0	n/a	19.4	40.1	52.5
Soft-NMS [2]		40.9	62.8	n/a	23.3	43.6	53.3
Deformable R-FCN + SNIP [30]	DPN-98	45.7	67.3	51.1	29.3	48.8	57.1
Single-shot detectors							
YOLOv2 [27]	DarkNet-19	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [24]	ResNet-101	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [8]		33.2	53.3	35.2	13.0	35.4	51.1
RefineDet512 [37] (single-scale)		36.4	57.5	39.5	16.6	39.9	51.4
RefineDet [37] (multi-scale)		41.8	62.9	45.7	25.6	45.1	**54.1**
RetinaNet800 [22]		39.1	59.1	42.3	21.8	42.7	50.2
GHM800 [18]		39.9	60.8	42.5	20.3	43.6	**54.1**
Ours800 (single-scale)		40.9	61.5	44.0	24.0	44.2	51.3
Ours (multi-scale)		**42.8**	**63.1**	**46.5**	**27.8**	**45.5**	53.2
CornerNet511 [17] (single-scale)	Hourglass-104	40.5	56.5	43.1	19.4	42.7	53.9
CornerNet [17] (multi-scale)		42.1	57.8	45.3	20.8	44.8	56.7
GHM800 [18]	ResNeXt-101	41.6	62.8	44.2	22.3	45.1	**55.3**
Ours800 (single-scale)		42.9	63.8	46.3	26.6	46.2	52.7
Ours (multi-scale)		**44.6**	**65.2**	**48.6**	**29.7**	**47.1**	54.6

Table 3: Object detection results of our best *single* model with the FSAF module vs. state-of-the-art single-shot and multi-shot detectors on the COCO `test-dev`.

Table 2. We find that our FSAF module is robust to various backbone networks. The FSAF module by itself is already better and faster than anchor-based RetinaNet. On ResNeXt-101, the FSAF module outperforms anchor-based counterparts by **1.2%** AP while being **68ms** faster. When applied jointly with anchor-based branches, our FSAF module consistently offers considerable improvements. This also suggests that *anchor-based branches are not utilizing the full power of backbone networks.* Meanwhile, our FSAF module introduces marginal computation cost to the whole network, leading to negligible loss of inference speed. Especially, we improve RetinaNet by **1.8%** AP on ResNeXt-101 with only **6ms** additional inference latency.

4.2. Comparison to State of the Art

We evaluate our final detector on the COCO `test-dev` split to compare with recent state-of-the-art methods. Our final model is RetinaNet with the FSAF module, i.e. anchor-based branches plus the FSAF module. The model is trained using scale jitter over scales {640, 672, 704, 736, 768, 800} and for 1.5× longer than the models in Section 4.1. The evaluation includes single-scale and multi-scale versions, where single-scale testing uses an image scale of 800 pixels and multi-scale testing applies test time augmentations. Test time augmentations are testing over scales {400, 500, 600, 700, 900, 1000, 1100, 1200} and horizontal flipping on each scale, following Detectron [10]. All of our results are from single models *without* ensemble.

Table 3 presents the comparison. With ResNet-101, our detector is able to achieve competitive performance in both single-scale and multi-scale scenarios. Plugging in ResNeXt-101-64x4d further improves AP to **44.6%** , which outperforms previous state-of-the-art single-shot detectors by a large margin.

5. Conclusion

This work identifies heuristic feature selection as the primary limitation for anchor-based single-shot detectors with feature pyramids. To address this, we propose FSAF module which applies online feature selection to train anchor-free branches in the feature pyramid. It significantly improves strong baselines with tiny inference overhead and outperforms recent state-of-the-art single-shot detectors.

References

[1] C. Bhagavatula, C. Zhu, K. Luu, and M. Savvides. Faster than real-time facial alignment: A 3d spatial transformer network approach in unconstrained poses. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 1

[2] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis. Soft-nmsimproving object detection with one line of code. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 5562–5570. IEEE, 2017. 8

[3] Z. Cai and N. Vasconcelos. Cascade r-cnn: Delving into high quality object detection. *arXiv preprint arXiv:1712.00726*, 2017. 8

[4] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 764–773. IEEE, 2017. 8

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009. 5

[6] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 304–311. IEEE, 2009. 1

[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html. 1, 2

[8] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg. Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*, 2017. 2, 3, 8

[9] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 2

[10] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He. Detectron. https://github.com/facebookresearch/detectron, 2018. 8

[11] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 447–456, 2015. 2

[12] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017. 1

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 1, 5, 7

[14] Y. He, C. Zhu, J. Wang, M. Savvides, and X. Zhang. Bounding box regression with uncertainty for accurate object detection. *arXiv preprint arXiv:1809.08545*, 2018. 2

[15] L. Huang, Y. Yang, Y. Deng, and Y. Yu. Densebox: Unifying landmark localization with end to end object detection. *arXiv preprint arXiv:1509.04874*, 2015. 3

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 1

[17] H. Law and J. Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 734–750, 2018. 3, 8

[18] B. Li, Y. Liu, and X. Wang. Gradient harmonized single-stage detector. In *Thirty-Third AAAI Conference on Artificial Intelligence*, 2019. 8

[19] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, and J. Sun. Detnet: A backbone network for object detection. *arXiv preprint arXiv:1804.06215*, 2018. 2

[20] X. Liang, T. Wang, L. Yang, and E. Xing. Cirl: Controllable imitative reinforcement learning for vision-based self-driving. *arXiv preprint arXiv:1807.03776*, 2018. 1

[21] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. In *CVPR*, page 3, 2017. 2, 5, 8

[22] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. *IEEE transactions on pattern analysis and machine intelligence*, 2018. 1, 2, 3, 4, 5, 8

[23] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, Cham, 2014. 1, 2, 6

[24] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016. 2, 3, 8

[25] X. Ma, Y. He, X. Luo, J. Li, M. Zhao, B. An, and X. Guan. Vehicle traffic driven camera placement for better metropolis security surveillance. *IEEE Intelligent Systems*, 2018. 1

[26] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010. 3

[27] J. Redmon and A. Farhadi. Yolo9000: Better, faster, stronger. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6517–6525. IEEE, 2017. 8

[28] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 2, 8

[29] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1

[30] B. Singh and L. S. Davis. An analysis of scale invariance in object detection–snip. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3578–3587, 2018. 8

[31] L. Tychsen-Smith and L. Petersson. Improving object localization with fitness nms and bounded iou loss. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 8

[32] J. Wang, Y. Yuan, G. Yu, and S. Jian. Sface: An efficient network for face detection in large scale variations. *arXiv preprint arXiv:1804.06559*, 2018. 3

[33] X. Wang and A. Gupta. Videos as space-time region graphs. In *The European Conference on Computer Vision (ECCV)*, September 2018. 1

[34] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 5987–5995. IEEE, 2017. 1, 2, 7

[35] H. Xu, X. Lv, X. Wang, Z. Ren, and R. Chellappa. Deep regionlets for object detection. *arXiv preprint arXiv:1712.02408*, 2017. 8

[36] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang. Unitbox: An advanced object detection network. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 516–520. ACM, 2016. 3, 4, 5

[37] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li. Single-shot refinement neural network for object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 8

[38] Q. Zhao, T. Sheng, Y. Wang, Z. Tang, Y. Chen, L. Cai, and H. Ling. M2det: A single-shot object detector based on multi-level feature pyramid network. In *Thirty-Third AAAI Conference on Artificial Intelligence*, 2019. 2

[39] Y. Zheng, D. K. Pal, and M. Savvides. Ring loss: Convex feature normalization for face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5089–5097, 2018. 1

[40] Z. Zhong, L. Sun, and Q. Huo. An anchor-free region proposal network for faster r-cnn based text detection approaches. *arXiv preprint arXiv:1804.09003*, 2018. 3

[41] C. Zhu, R. Tao, K. Luu, and M. Savvides. Seeing small faces from robust anchor's perspective. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2

[42] Y. Zhu, C. Zhao, J. Wang, X. Zhao, Y. Wu, H. Lu, et al. Couplenet: Coupling global structure with local parts for object detection. In *Proc. of Intl Conf. on Computer Vision (ICCV)*, volume 2, 2017. 8

CVPR2019 Paper Translation

姓名：陈豪博

学号：2016302862

班号：10011604

用于单镜头目标检测的无锚特征选择模块

Chenchen Zhu Yihui He Marios Savvides
Carnegie Mellon University
{chenchez, he2, marioss}@andrew.cmu.edu

a: RetinaNet (anchor-based, ResNeXt-101) b: Ours (anchor-based + FSAF, ResNet-50)

图 1: 在相同的训练和测试尺度下，使用有锚RetinaNet [22] 的强大*ResNeXt-101* (左)和我们的仅带有FSAF模块的*ResNet-50*（右）检测器的定性结果。我们的FSAF模块仅在不太强大主干网上就可以帮助检测像小人和平板滑雪板这样的高难度对象，有关更多示例，请参见图 7。

摘要

我们提出了一种简单有效的可用于单镜头目标探测器的无锚特征选择(FSAF)模块构件。它可以嵌入具有特征金字塔结构的单镜头探测器。FSAF模块聚焦于传统基于锚框检测方法的两个局限性：1）启发式引导的特征选择；2）基于重叠的锚框采样。FSAF模块的总体概念是应用于多级无锚框分支训练的在线特征选择。具体来说，一个无锚分支被附加到特征金字塔的每一层，允许检测框在任意一层以无锚方式进行编码和解码。在训练期间，我们动态地将每个实例分配到最合适的特征层。在推理时，FSAF模块可以通过并行输出预测结果与基于锚框的分支协同工作。我们用无锚分支的简单实现和在线特性选择策略实例化了这个概念。在COCO检测跟踪上的实验结果表明，我们的FSAF模块性能优于基于锚框的同类模块，但速度更快。当与基于锚框的分支协同工作时，FSAF模块在各种设置下显著提高了RetinaNet网络的基准，同时几乎不具有推理开销。并且所得到的最佳模型可以获得最高水准的44.6%的mAP，性能优于COCO上现有的所有单镜头探测器。

1. 介绍

目标检测是计算机视觉领域的一项重要任务。它是各种下游视觉应用的先决条件，如实例分割 [12]、人脸分析 [1, 39]、自动驾驶汽车 [6, 20]和视频分析 [25, 33]。由于深度卷积神经网络 [16, 29, 13, 34]和注释良好的数据集 [7, 23]的发展，目标检测器的性能得到

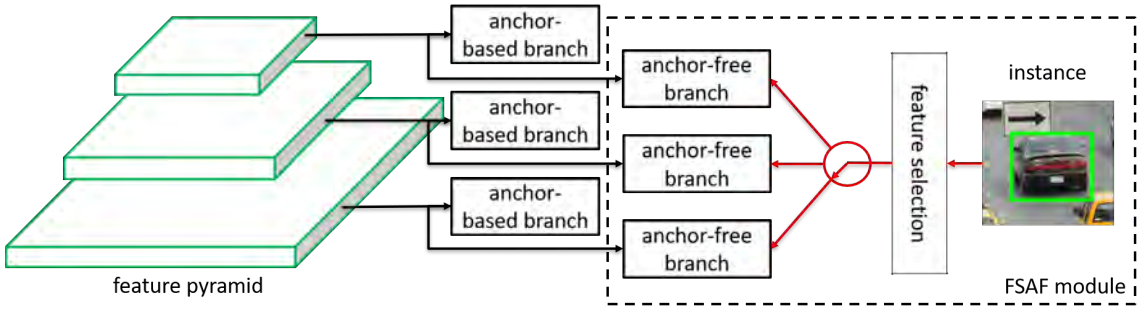

图 3: 嵌入FSAF模块的传统有锚检测方法的概述。在训练过程中，通过特征选择将每个实例分配到一个金字塔层以设置监控信号。

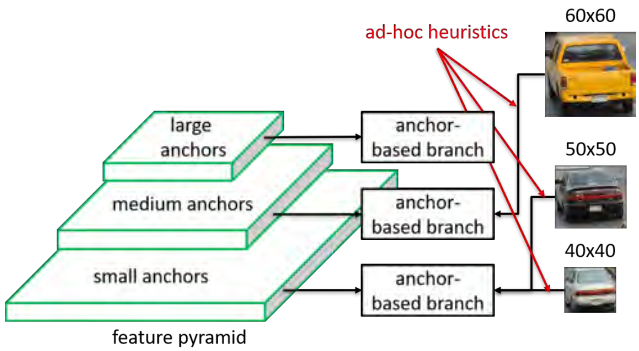

图 2: 基于锚框分支的特征选择层可能不是最优的。

了显著提高。

尺度变化是目标检测的一个难题。为了实现尺度不变性，最先进的探测器构建了特征金字塔或多级特征塔 [24, 8, 21, 22, 19, 38]，同时多尺度层次的特征图并行产生预测。此外，锚框还可以进一步处理尺度变化 [24, 28]。锚框被用于将所有可能的具有连续空间的实例框离散为有限数量的，具有预定义位置、比例和纵横比的框。同时实例框通过其与锚框的重叠交并比（IoU）来将其与锚定框相匹配。当与特征金字塔集成时，大型锚框通常与上部特征图相关联，而小型锚框则与下部特征图相关联，请参见图 2 。这是基于上部特征图具有多更适合检测大实例的语义信息，而下部特征图具有更多适合检测小实例的细粒度细节信息的启发 [11] 。结合锚框的特征金字塔设计在目标检测基准上取得了良好的性能 [7, 23, 9]。

然而，这种设计有两个局限性：1）启发式引导的特征选择；2）基于重叠的锚框取样。在训练过程中，每个实例总是根据重叠交并比（IoU）匹配到最近的锚框，而锚框则通过人为定义的规则(如框的大小)与特定级别的特征图相关联。因此，为每个实例选择的特征级别完全基于*特定的启发式*。例如,一个50×50 像素的汽车实例大小和另一个类似的60×60 像素的汽车实例可能分配到两个不同的特征层，而另一个40×40 像素的汽车实例可能被分配到与一个50×50 像素的汽车相同层，如图 2 所示。换句话说，锚框匹配机制本质上是启发式引导的，这将导致一个主要缺陷，即用于训练每个实例的所选特征级别可能不是最优的。

为了同时解决这两个限制，我们提出了一种简单有效的方法——无锚特征选择(FSAF)模块。我们的动机是让每个实例自由地选择最佳的特征层来优化网络，因此在我们的模块中不应该有锚框来约束特征的选择。我们采用了一种方法代替它，以无锚的方式对实例进行编码从而学习用于分类和回归的参数。图 3 给出了其一般概念。特征金字塔的每一层都构建一个独立于有锚分支的无锚分支。与有锚分支类似，它由分类子网和回归子网组成(图上没有展示)。一个实例可以被分配到无锚分支的任意层。训练期间，我们根据实例内容动态地为每个实例选择最合适的特征层，而不是仅仅根据实例框的大小。然后，所选的特性层将学习检测所分配的实例。在推理时，FSAF模块可以独立运行，也可以与有锚分支联合运行。我们的FSAF模块与主干网无关，可以应用于具有特征金字塔结构的单镜头检测器。此外，无锚分支的实例化和在线特征选择可以是多种多样的。在这项工作中，我们的FSAF模块的实现尽量保持简单，从而使其计算成本比起整个网络来说是可忽视的。

在COCO [23]目标检测基准上的大量实验验证了

该方法的有效性。FSAF模块本身性能优于有锚对应模块，同时运行速度更快。在与有锚分支协同工作时，FSAF模块可以在不同主干网之间持续改进健壮的基准，同时引入最小的计算成本。特别是，我们使用ResNeXt-101 [34] 以仅增加了**6ms**的推断延迟在RetinaNet上获取了**1.8%**的提高。此外，我们的最终探测器在多尺度测试中实现了最先进的**44.6%**的mAP，超过所有现有的单镜头探测器在COCO上的记录。

2. 相关工作

近年来，目标检测器常用特征金字塔或多级特征塔作为一种常见结构。SSD [24]首先提出从多个特征尺度来预测类别分数和边界框。FPN [21]和DSSD [8] 提出了在所有尺度上用高级语义特征图来增强低级特征。RetinaNet [22] 解决了具有焦点损失的多级密集探测器的类不平衡问题。DetNet [19] 设计了一种新的骨干网络，从而在金字塔上层保持高的空间分辨率。但是，他们都使用预定义的锚框对对象实例进行编码和解码。其他工作则以不同的方式处理尺度的变化。Zhu等人 [41] 增强了小目标的锚设计。He等人 [14] 将边界框建模为高斯分布，以改进定位功能。

无锚检测的概念并不新鲜。Dense-Box [15] 首先提出了一个统一的端到端全卷积框架，该框架直接预测了边界盒。UnitBox [36] 提出了一种基于交并比（IoU）损失函数，用于更好的边框回归。Zhong等人 [40] 提出了无锚候选区域网络来寻找不同尺度、纵横比和方向的文本。最近CornerNet [17] 提出将一个物体边界框检测为一对角，从而得到了最好的单镜头检测器。SFace [32] 提出将有锚的方法与无锚方法相结合。然而，它们仍然采用启发式特征选择的策略。

3. 无锚特征选择模块

在本节中，我们实例化了我们的无锚特征选择(FSAF)模块，展示了如何将它应用于具有特征金字塔的单镜头检测器，如SSD [24]、DSSD [8]和RetinaNet [22]。在不失一般性的前提下，我们将FSAF模块应用到最先进的RetinaNet [22]中，并从以下几个方面展示了我们的设计：1）如何在网络中创建无锚分支(3.1)；2）如何生成无锚分支的监控信号(3.2)；3）如何为每个实例动态选择特征层(3.3)；4）如何联合训练和测试无锚分支和有锚分支(3.4)。

3.1. 网络结构

从网络的角度来看，我们的FSAF模块非常简单。图 4 说明了带有FSAF模块的RetinaNet [22] 的体系结构。简而言之，RetinaNet由一个主干网络(图中未展示)和两个特定于任务的子网络组成。特征金字塔由P_3层到P_7层的骨干网构成，其中l为金字塔层数，P_l具有输入图像$1/2^l$的分辨率。为了简单起见，只展示了三层。金字塔的每一层都用于探测不同尺度的物体。为此，P_l附加了一个分类子网络和一个回归子网络，它们都是小的全卷积网络。分类子网络为每A个锚框和K个对象类预测对象在每个空间位置的概率。如果存在，回归子网络预测每A个锚到附近实例的4维类无关偏移量。

在RetinaNet的顶部，我们的FSAF模块仅为每个金字塔层引入两个额外的卷积层，如图 4 中虚线特征图所示。这两层分别负责无锚分支的分类和回归预测。更具体地说，一个3×3卷积层带有K个滤波器，附在分类子网中的特征图上，后面接着sigmoid函数，与有锚分支的卷积层并行。它为K个对象类预测对象在每个空间位置的概率。同样的，回归子网中的特征图上也附加了一个3×3卷积层，带有四个滤波器，然后是ReLU [26] 函数。它负责预测以无锚方式编码的预测框偏移量。为此，无锚和有锚的分支以多任务的方式联合工作，共享金字塔每层的特征。

3.2. 真值和损失

给定一个对象实例，我们知道它的类标签k和边界框坐标$b = [x, y, w, h]$, (x, y)为边框的中心，w, h分别为边框的宽度和高度。在训练期间，可以将实例分配到任意的特征层P_l。定义投影框$b_p^l = [x_p^l, y_p^l, w_p^l, h_p^l]$为$b$在特征金字塔$P_l$上的投影，即$b_p^l = b/2^l$。我们还定义了有效框$b_e^l = [x_e^l, y_e^l, w_e^l, h_e^l]$和忽略框$b_i^l = [x_i^l, y_i^l, w_i^l, h_i^l]$为$b_p^l$的比例区域，分别受常数尺度因子$\epsilon_e$和$\epsilon_i$控制，即$x_e^l = x_p^l, y_e^l = y_p^l \text{fi} w_e^l = \epsilon_e w_p^l \text{fi} h_e^l = \epsilon_e h_p^l$，$x_i^l = x_p^l \text{fi} y_i^l = y_p^l \text{fi} w_i^l = \epsilon_i w_p^l \text{fi} h_i^l = \epsilon_i h_p^l$。我们设置$\epsilon_e = 0.2$，$\epsilon_i = 0.5$。图 5 显示了一个为汽车实例生成真值的示例。

分类输出：分类输出的真值是K个图，每个图对应一个类。实例以三种方式影响第k个真值。首先，有效框b_e^l区域是由“汽车”类图中白色框所表示的区域所填充的正区域，表示实例的存在。第二，忽略框除

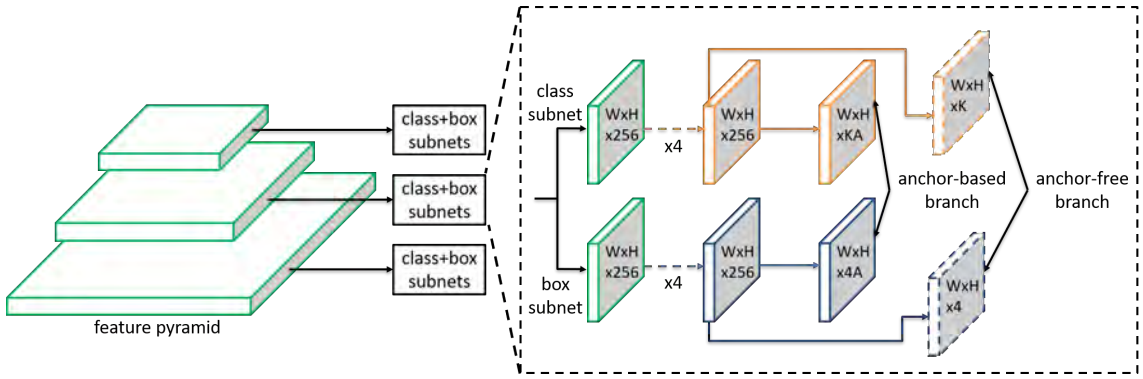

图 4: 使用我们的FSAF模块的RetinaNet网络架构。FSAF模块在每个金字塔层只引入了两个额外的卷积层(虚线特征图)，保持了架构的完全卷积性。

去有效框($b_i^l - b_e^l$) 是显示为灰色的忽略区域，这意味着这一地区的梯度不回到网络进行传播。第三，相邻特征层的忽略框(b_i^{l-1}, b_i^{l+1}) 如果存在的话也是忽视区域，。注意，如果两个实例的有效框在同一层上重叠，则较小的实例具有更高的优先级。其余区域的真值图是由零填充的负(黑色)区域，表示没有对象。焦点损失[22]用于监督，其中超参数$\alpha = 0.25, \gamma = 2.0$。一张图像的无锚分支的分类总损失是所有非忽略区域上的焦点损失之和，并由所有有效框区域内的像素总数归一化。

框回归输出：回归输出的真值是4个与类无关的偏移图。实例只影响偏移量图上的b_e^l 区域。对于每个b_e^l 里面的像素位置(i, j)，我们将投影的框b_p^l 表示为一个四维向量$\mathbf{d}_{i,j}^l = [d_{t_{i,j}}^l, d_{l_{i,j}}^l, d_{b_{i,j}}^l, d_{r_{i,j}}^l]$，其中$d_t^l$, d_l^l, d_b^l, d_r^l 分别为当前像素位置(i, j) 和b_p^l 的上、左、下、右边界。然后(i, j) 处的四维向量经过4个偏移图，$\mathbf{d}_{i,j}^l/S$，每个图对应一个维度。S是一个归一化常数，我们根据经验选择$S = 4.0$。有效框外的位置是忽略梯度的灰色区域，采用交并比损失 [36] 进行优化。图像的无锚分支的总回归损失是所有有效框区域交并比损失的平均值。

在推理过程中，很容易从分类和回归输出中解码预测框。在每个像素位置(i, j)，假设预测偏移量为$[\hat{o}_{t_{i,j}}, \hat{o}_{l_{i,j}}, \hat{o}_{b_{i,j}}, \hat{o}_{r_{i,j}}]$。则预测距离为$[S\hat{o}_{t_{i,j}}, S\hat{o}_{l_{i,j}}, S\hat{o}_{b_{i,j}}, S\hat{o}_{r_{i,j}}]$。预测投影框的左上角和右下角分别是$(i - S\hat{o}_{t_{i,j}}, j - S\hat{o}_{l_{i,j}})$ 和$(i + S\hat{o}_{b_{i,j}}, j + S\hat{o}_{r_{i,j}})$。我们进一步将投影框放大$2^l$，得到图像平面中的最终框。框的置信度分数和类别由最大分数和对应类的分类输出图(i, j) 位置的K 维向量决定。

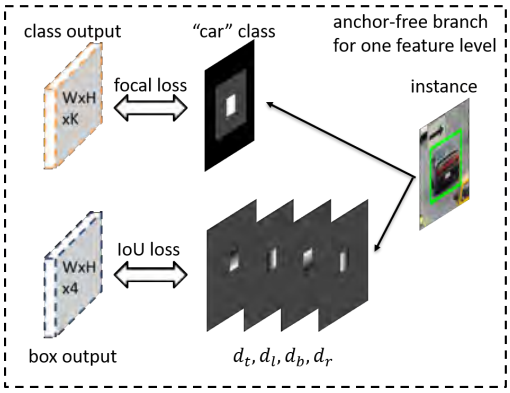

图 5: 一个实例在无锚分支其中一个特征层的监视信号。我们使用焦点损失进行分类，交并比损失用于框回归。

3.3. 在线特征选择

无锚分支的设计允许我们使用任意金字塔层P_l 的特征来学习每个实例。为了找到最优特性层，我们的FSAF模块根据实例内容选择最佳P_l，而不是像基于锚框的方法中那样选择实例框的大小。

以I 为例，将其在P_l的分类损失和框回归损失分别定义为$L_{FL}^I(l)$ 和$L_{IoU}^I(l)$。计算方法是将有效框区域b_e^l 上的焦点损失和交并比损失平均，即

$$L_{FL}^I(l) = \frac{1}{N(b_e^l)} \sum_{i,j \in b_e^l} FL(l, i, j)$$
$$L_{IoU}^I(l) = \frac{1}{N(b_e^l)} \sum_{i,j \in b_e^l} IoU(l, i, j) \quad (1)$$

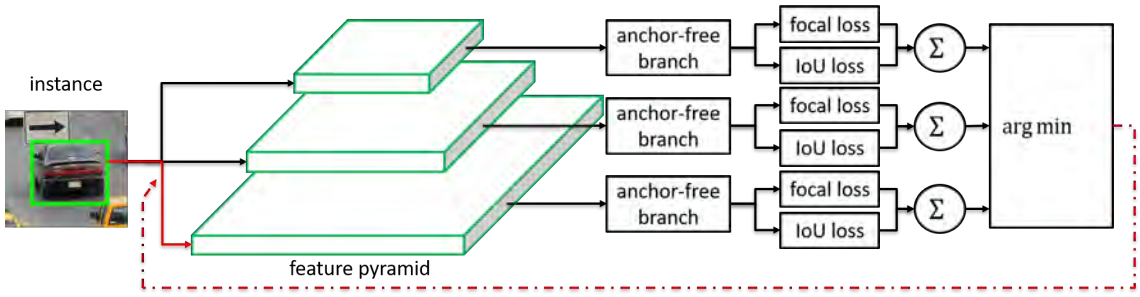

图 6: 在线特征选择机制。每个实例都通过所有层无锚分支来计算有效区域上的平均分类(焦点)损失和回归(交并比)损失。然后选择两个损失之和最小的层，为该实例设置监控信号。

其中$N(b_e^l)$为b_e^l区域内的像素个数，$FL(l,i,j)$, $IoU(l,i,j)$为P_l的(i,j)处的焦点损失 [22]和交并比损失 [36]。

图 6展示了在线特征选择过程。首先，实例I通过特征金字塔的所有层。在此基础上，利用等式 (1)对所有无锚分支的$L_{FL}^I(l)$和$L_{IoU}^I(l)$进行求和。最后选择最好的金字塔层P_{l^*}产生的最小损失和学习此实例，即

$$l^* = \arg\min_l L_{FL}^I(l) + L_{IoU}^I(l) \tag{2}$$

对于批处理训练为其相应分配的实例更新特征。直观的感觉是，所选的特征目前是对实例建模的最佳特征。它的损失在特征空间中形成一个下界。通过训练，我们进一步拉低这个下界。在推理时，我们不需要选择特征，因为最合适的特征金字塔层自然会输出较高的置信度分数。

为了验证我们的在线特征选择的重要性，我们还在消融研究中进行了启发式特征选择过程的比较(4.1)。启发式特征选择完全依赖于框的大小。我们借用了FPN检测器 [21]的思想。将一个实例I分配给特征金字塔的$P_{l'}$级:

$$l' = \lfloor l_0 + \log_2(\sqrt{wh}/224) \rfloor \tag{3}$$

这里224是标准的ImageNet预训练大小，l_0是将$w \times h = 224^2$的实例映射到的目标层。在本工作中，我们选择$l_0 = 5$，因为ResNet [13] 使用特征图中第5卷积组的进行最终分类。

3.4. 联合推理和训练

当嵌入到RetinaNet [22] 时，我们的FSAF模块与有锚分支协同工作，如图 4所示。我们保持了有锚分支的原始性，在训练和推理过程中所有超参数都保持不变。

推理： FSAF模块只是在全卷积RetinaNet上增加了几个卷积层，所以推理仍然像图像通过网络一样简单。对于无锚分支，我们只解码每个金字塔层中得分最高的1k个位置的框预测，然后将置信值用0.05阈值化。这些来自各个层的最高预测与有锚分支的框预测合并，然后是阈值为0.5的非最大抑制，得到最终的检测结果。

初始化： 主干网络在ImageNet1k [5] 上进行预训练。我们在像在[22] 中一样初始化RetinaNet的各层。对于FSAF模块中的卷积层，分类层我们以$-\log((1-\pi)/\pi)$初始化偏移量，以$\sigma = 0.01$的高斯分布初始化权重，π特指在一开始的训练每个像素位置输出对象分数在π左右。我们在按照[22] 中设置$\pi = 0.01$。所有的框回归层都初始化偏移量为b，权重为$\sigma = 0.01$的高斯分布。我们在所有的实验中都使用$b = 0.1$。初始化有助于在早期迭代中稳定网络学习，避免较大的损失。

优化： 整个网络的损失是无锚和有锚的分支的综合损失。设L^{ab}为原始有锚的RetinaNet的总损失。设L_{cls}^{af}和L_{reg}^{af}分别为无锚分支的分类总损失和回归总损失, 则总优化损失为$L = L^{ab} + \lambda(L_{cls}^{af} + L_{reg}^{af})$, 其中$\lambda$控制无锚分支的权重。我们在所有的实验中都设置了$\lambda = 0.5$，尽管结果对精确值具有鲁棒性。整个网络在8个GPU上进行随机梯度下降(SGD)训练，每个GPU 2张图像。除非另有说明，所有模型都经过90k次初始学习率为0.01的迭代训练，在60k时学习率除以10，80k时再除以10。除非另有说明，否则水平图像翻转是唯一应用的数据增强方法。权重衰减为0.0001，动量

	Anchor-based branches	Anchor-free branches		AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
		Heuristic feature selection Eqn. (3)	Online feature selection Eqn. (2)						
RetinaNet	✓			35.7	54.7	38.5	19.5	39.9	47.5
Ours		✓		34.7	54.0	36.4	19.0	39.0	45.8
			✓	35.9	55.0	37.9	19.8	39.6	48.2
	✓	✓		36.1	55.6	38.7	19.8	39.7	48.9
	✓		✓	**37.2**	**57.2**	**39.4**	**21.0**	**41.2**	**49.7**

表 1: COCO `minival` 上FSAF模块消融实验。ResNet-50是本表所有实验的骨干网络，研究了无锚分支、启发式特征选择和在线特征选择的效果。

Backbone	Method	AP	AP_{50}	Runtime (ms/im)
R-50	RetinaNet	35.7	54.7	131
	Ours(FSAF)	35.9	55.0	107
	Ours(AB+FSAF)	37.2	57.2	138
R-101	RetinaNet	37.7	57.2	172
	Ours(FSAF)	37.9	58.0	148
	Ours(AB+FSAF)	39.3	59.2	180
X-101	RetinaNet	39.8	59.5	356
	Ours(FSAF)	41.0	61.5	288
	Ours(AB+FSAF)	41.6	62.4	362

表 2: 不同骨干网在COCO `minival` 上的检测精度和推理延迟。**AB**：有锚分支。**R**: ResNet。**X**: ResNeXt。

为0.9。

4. 实验

我们在COCO数据集 [23]的检测跟踪上进行了实验。训练数据是COCO `trainval35k` split，包括来自`train`的所有80k张图像和来自`val` split的40k 张图像中的随机35k张图像子集。我们在`minival` split数据集上，包含来自`val` 的剩余5k张图像，通过消融研究分析了我们的方法。当与最先进的方法进行比较时，我们报告了`test-dev` split上的COCO AP，它没有公共标签，并需要使用评估服务器。

4.1. 消融研究

对于所有消融研究，我们使用800像素的图像比例尺进行训练和测试。评估了几个重要元素对检测器的贡献，包括无锚分支、在线特征选择和主干网络。结果见表1和表2。

无锚分支是必要的。我们首先使用两种特征选择方法(Table 1 第2项和第3项)分别训练两个仅仅具有无锚分支的检测器。事实证明，没有锚定的分支只能取得不错的效果。当与有锚分支联合优化时，无锚分支有助于学习有锚的分支难以建模的实例，从而提高AP分数(Table 1 第5项)。尤其是AP_{50}, AP_S 和AP_L评分，通过在线特征选择，分别提高了2.5%、1.5%、2.2%。为了了解FSAF模块可以检测到哪些类型的对象，我们在图 7 中展示了我们与RetinaNet进行比较的一些定性结果。显然，我们的FSAF模块更擅长于寻找具有挑战性的实例，例如锚框不能很好地覆盖的微小和非常薄的对象。

在线特性选择是必不可少的。正如 3.3 节所述，我们可以像有锚分支一样，根据启发式选择无锚分支中的特征，或者根据实例内容选择特征。事实证明，选择正确的特征来学习在检测中起着至关重要的作用。实验表明，采用启发式特征选择的无锚分支(等式(3))具有较好的鲁棒性。由于参数的可学习性较差，无法与有锚的对手竞争。但有在线特征选择(等式 (2))，AP提高了**1.2%** (表 1 第3项vs 第2项)，克服了参数劣势。此外，表 1第4项和第5项进一步证实，我们的在线特征选择对于无锚和有锚分支的良好协同工作是必不可少的。

如何选择最优特征？为了理解为实例选择的最优金字塔层，我们可视化了图 8 中仅来自无锚分支的一些定性检测结果。类名前的数字表示检测对象的特性

图 7: 有锚的RetinaNet(顶部, Table 1 第1项)和带有附加FSAF模块的检测器(底部, Table 1 第5项)之间的定性比较示例。两者都使用ResNet-50作为骨干。我们的FSAF模块帮助寻找更具挑战性的对象。

图 8: 从无锚定分支中在线特性选择的可视化。类名前的数字是检测实例的金字塔级别。我们将此级别与基于锚的分支中分配此实例的级别进行比较，并使用红色表示不一致，使用绿色表示一致。

级别。事实证明，在线特性选择实际上遵循这样的规则:上层选择较大的实例，下层负责较小的实例，这与有锚分支中的原则相同。然而，也有相当多的例外，即在线特征选择选择金字塔层不同于有锚分支的选择。我们将这些异常标记为图 8 中的红色框。绿框表示FSAF模块和有锚分支之间一致。通过捕获这些异常，我们的FSAF模块可以使用更好的特征来检测具有挑

FSAF模块是健壮高效的。我们也评估了主干网对我们的FSAF模块在精度和速度方面的影响。三个主干网络包括ResNet-50、ResNet-101 [13]和ResNeXt-101 [34]。检测器运行在一个单一的具有CUDA 9和CUDNN

Method	Backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Multi-shot detectors							
CoupleNet [42]	ResNet-101	34.4	54.8	37.2	13.4	38.1	50.8
Faster R-CNN+++ [28]		34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w/ FPN [21]		36.2	59.1	39.0	18.2	39.0	48.2
Regionlets [35]		39.3	59.8	n/a	21.7	43.7	50.9
Fitness NMS [31]		41.8	60.9	44.9	21.5	45.0	57.5
Cascade R-CNN [3]		42.8	62.1	46.3	23.7	45.5	55.2
Deformable R-FCN [4]	Aligned-Inception-ResNet	37.5	58.0	n/a	19.4	40.1	52.5
Soft-NMS [2]		40.9	62.8	n/a	23.3	43.6	53.3
Deformable R-FCN + SNIP [30]	DPN-98	45.7	67.3	51.1	29.3	48.8	57.1
Single-shot detectors							
YOLOv2 [27]	DarkNet-19	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [24]	ResNet-101	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [8]		33.2	53.3	35.2	13.0	35.4	51.1
RefineDet512 [37] (single-scale)		36.4	57.5	39.5	16.6	39.9	51.4
RefineDet [37] (multi-scale)		41.8	62.9	45.7	25.6	45.1	**22**
RetinaNet800 [18]		39.1	59.1	42.3	21.8	42.7	50.2
GHM800 [17]		39.9	60.8	42.5	20.3	43.6	**54.1**
Ours800 (single-scale)		40.9	61.5	44.0	24.0	44.2	51.3
Ours (multi-scale)		**42.8**	**63.1**	**46.5**	**27.8**	**45.5**	53.2
CornerNet511 [17] (single-scale)	Hourglass-104	40.5	56.5	43.1	19.4	42.7	53.9
CornerNet [17] (multi-scale)		42.1	57.8	45.3	20.8	44.8	56.7
GHM800 [18]	ResNeXt-101	41.6	62.8	44.2	22.3	45.1	**55.3**
Ours800 (single-scale)		42.9	63.8	46.3	26.6	46.2	52.7
Ours (multi-scale)		**44.6**	**65.2**	**48.6**	**29.7**	**47.1**	54.6

表 3: 我们具有FSAF模块的最好单一模型在COCO `test-dev` 上的目标检测结果对比最先进的单镜头和多镜头检测器的结果。

7的泰坦X GPU上，使用批量大小为1。结果见表 2。我们发现我们的FSAF模块对各种主干网都是鲁棒的。FSAF模块本身已经比有锚的RetinaNet更好更快。在ResNeXt-101上，FSAF模块的性能比有锚的模块高出**1.2%**，同时速度快**68ms**。当与有锚分支联合应用时，我们的FSAF模块始终能够提供相当大的改进，这也表明有锚分支没有充分利用主干网络的能力。同时，我们的FSAF模块在整个网络中引入了边际计算成本，推理速度损失可以忽略不计。特别地，我们在ResNeXt- 101上提高了**1.8%** 的AP，仅增加了**6ms** 的推断延迟。

4.2. 与最前先进技术比较

我们评估了在COCO `test-dev` split上的最终检测器，并与最新的最先进的方法进行了比较。我们最后的模型是带有FSAF模块的RetinaNet，即有锚分支加上FSAF模块。该模型使用{640, 672, 704, 736, 768, 800}中的比例尺抖动进行训练，是第 4.1 节的模型训练时间1.5 倍长。评估包括单尺度和多尺度版本，其中单尺度测试使用800像素的图像尺度，多尺度测试使用测试时间增广。测试时间增广是测试在{400, 500, 600,

700, 900, 1000, 1100, 1200}中的尺度，并且在每个尺度水平翻转，跟检测器一样 [10]。我们所有的结果都来自于没有集成的单个模型。

表 3 给出了比较。通过ResNet-101，我们的检测器能够在单尺度和多尺度场景中获得良好的表现。嵌入到ResNeXt-101-64x4d后，AP进一步提高到**44.6%**，大大超过了以前最先进的单镜头探测器。

5. 结论

该工作确定了启发式特征选择是有锚单镜头特征金字塔检测器的主要限制。为了解决这一问题，我们提出了FSAF模块，该模块应用在线特征选择来训练特征金字塔中的无锚分支。它以微小的推理开销显著提高了本就健壮的基准，并优于目前最先进的单镜头探测器。

References

[1] C. Bhagavatula, C. Zhu, K. Luu, and M. Savvides. Faster than real-time facial alignment: A 3d spatial transformer network approach in unconstrained poses. In The IEEE International Conference on Computer Vision (ICCV), Oct 2017. 1 1

[2] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis. Softnmsimproving object detection with one line of code. In Computer Vision (ICCV), 2017 IEEE International Conference on, pages 5562–5570. IEEE, 2017. 8 8

[3] Z. Cai and N. Vasconcelos. Cascade r-cnn: Delving into high quality object detection. arXiv preprint arXiv:1712.00726, 2017. 8 8

[4] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In Computer Vision (ICCV), 2017 IEEE International Conference on, pages 764–773. IEEE, 2017. 8 8

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei- Fei. Imagenet: A large-scale hierarchical image database. In Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, pages 248–255. IEEE, 2009. 5 5

[6] P. Doll´ar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, pages 304–311. IEEE, 2009. 1 1

[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascalnetwork.org/challenges/VOC/voc2007/workshop/index.html. 1, 2 1, 2

[8] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg. Dssd: Deconvolutional single shot detector. arXiv preprint arXiv:1701.06659, 2017. 2, 3, 8 2, 3, 8

[9] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In Conference on Computer Vision and Pattern Recognition (CVPR), 2012. 2 2

[10] R. Girshick, I. Radosavovic, G. Gkioxari, P. Doll´ar, and K. He. Detectron. https://github.com/ facebookresearch/detectron, 2018. 8 9

[11] B. Hariharan, P. Arbel´aez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 447–456, 2015. 2 2

[12] K. He, G. Gkioxari, P. Doll´ar, and R. Girshick. Mask r-cnn. In Computer Vision (ICCV), 2017 IEEE International Conference on, pages 2980–2988. IEEE, 2017. 1 1

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 770–778, 2016. 1, 5, 7 1, 5, 7

[14] Y. He, C. Zhu, J. Wang, M. Savvides, and X. Zhang. Bounding box regression with uncertainty for accurate object detection. arXiv preprint arXiv:1809.08545, 2018. 2 3

[15] L. Huang, Y. Yang, Y. Deng, and Y. Yu. Densebox: Unifying landmark localization with end to end object detection. arXiv preprint arXiv:1509.04874, 2015. 3 3

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097–1105, 2012. 1 1

[17] H. Law and J. Deng. Cornernet: Detecting objects as paired keypoints. In Proceedings of the European Conference on Computer Vision (ECCV), pages 734–750, 2018. 3, 8 3, 8

[18] B. Li, Y. Liu, and X. Wang. Gradient harmonized singlestage detector. In Thirty-Third AAAI Conference on Artificial Intelligence, 2019. 8 8

[19] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, and J. Sun. Detnet: A backbone network for object detection. arXiv preprint arXiv:1804.06215, 2018. 2 2, 3

[20] X. Liang, T. Wang, L. Yang, and E. Xing. Cirl: Controllable imitative reinforcement learning for vision-based selfdriving. arXiv preprint arXiv:1807.03776, 2018. 1 1

[21] T.-Y. Lin, P. Doll´ar, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. In CVPR, page 3, 2017. 2, 5, 8 2, 3, 5, 8

[22] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Doll´ar. Focal loss for dense object detection. IEEE transactions on pattern analysis and machine intelligence, 2018. 1, 2, 3, 4, 5, 8 1, 2, 3, 4, 5

[23] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Doll´ar, and C. L. Zitnick. Microsoft coco: Common objects in context. In European conference on computer vision, pages 740–755. Springer, Cham, 2014. 1, 2, 6 1, 2, 6

[24] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.- Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In European conference on computer vision, pages 21–37. Springer, 2016. 2, 3, 8 2, 3, 8

[25] X. Ma, Y. He, X. Luo, J. Li, M. Zhao, B. An, and X. Guan. Vehicle traffic driven camera placement for better metropolis security surveillance. IEEE Intelligent Systems, 2018. 1 1

[26] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th international conference on machine learning (ICML-10), pages 807–814, 2010. 3 3

[27] J. Redmon and A. Farhadi. Yolo9000: Better, faster, stronger. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 6517–6525. IEEE, 2017. 8 8

[28] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems, pages 91–99, 2015. 2, 8 2, 8

[29] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014. 1 1

[30] B. Singh and L. S. Davis. An analysis of scale invariance in object detection–snip. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3578–3587, 2018. 8 8

[31] L. Tychsen-Smith and L. Petersson. Improving object localization with fitness nms and bounded iou loss. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018. 8 8

[32] J. Wang, Y. Yuan, G. Yu, and S. Jian. Sface: An efficient network for face detection in large scale variations. arXiv preprint arXiv:1804.06559, 2018. 3 3

[33] X. Wang and A. Gupta. Videos as space-time region graphs. In The European Conference on Computer Vision (ECCV), September 2018. 1 1

[34] S. Xie, R. Girshick, P. Doll´ar, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on, pages 5987–5995. IEEE, 2017. 1, 2, 7 1, 3, 7

[35] H. Xu, X. Lv, X. Wang, Z. Ren, and R. Chellappa. Deep regionlets for object detection. arXiv preprint arXiv:1712.02408, 2017. 8 8

[36] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang. Unitbox: An advanced object detection network. In Proceedings of the 2016 ACM on Multimedia Conference, pages 516–520. ACM, 2016. 3, 4, 5 3, 4, 5

[37] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li. Singleshot refinement neural network for object detection. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018. 8 8

[38] Q. Zhao, T. Sheng, Y. Wang, Z. Tang, Y. Chen, L. Cai, and H. Ling. M2det: A single-shot object detector based on multi-level feature pyramid network. In Thirty-Third AAAI Conference on Artificial Intelligence, 2019. 2 2

[39] Y. Zheng, D. K. Pal, and M. Savvides. Ring loss: Convex feature normalization for face recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 5089–5097, 2018. 1 1

[40] Z. Zhong, L. Sun, and Q. Huo. An anchor-free region proposal network for faster r-cnn based text detection approaches. arXiv preprint arXiv:1804.09003, 2018. 3 3

[41] C. Zhu, R. Tao, K. Luu, and M. Savvides. Seeing small faces from robust anchor’ s perspective. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018. 2 3

[42] Y. Zhu, C. Zhao, J.Wang, X. Zhao, Y.Wu, H. Lu, et al. Couplenet: Coupling global structure with local parts for object detection. In Proc. of Intl Conf. on Computer Vision (ICCV), volume 2, 2017. 8 8