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].

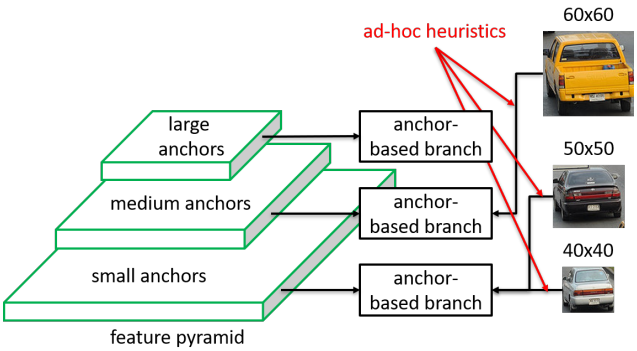

Figure 2: Selected feature level in anchor-based branches may not be optimal.

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

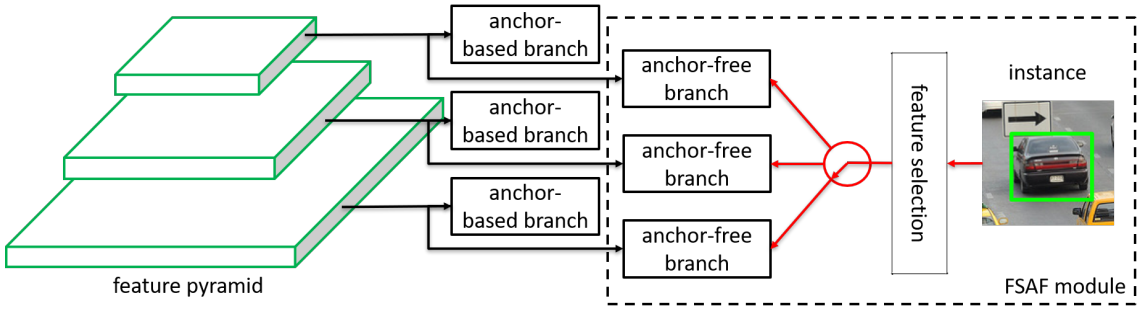

Figure 3: Overview of our FSAF module plugged into conventional anchor-based detection methods. During training, each instance is assigned to a pyramid level via feature selection for setting up supervision signals.

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

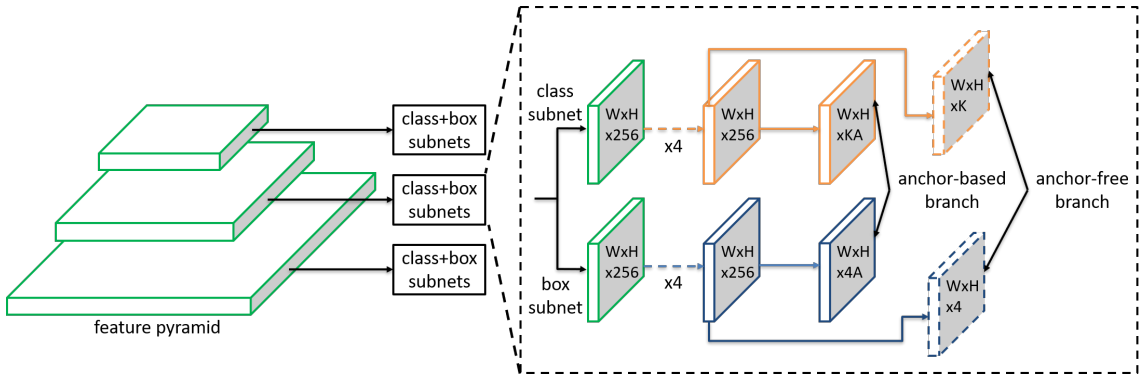

Figure 4: Network architecture of RetinaNet with our FSAF module. The FSAF module only introduces two additional conv layers (dashed feature maps) per pyramid level, keeping the architecture fully convolutional.

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.

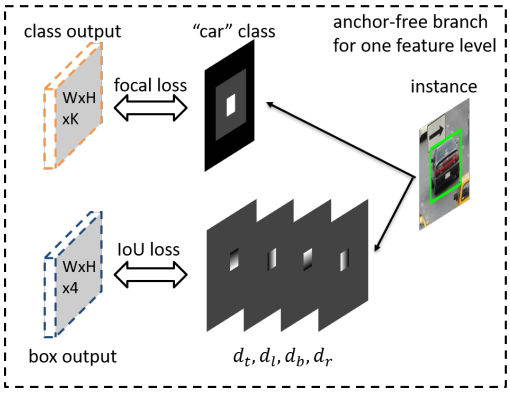

Figure 5: Supervision signals for an instance in one feature level of the anchor-free branches. We use focal loss for classification and IoU loss for box regression.

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

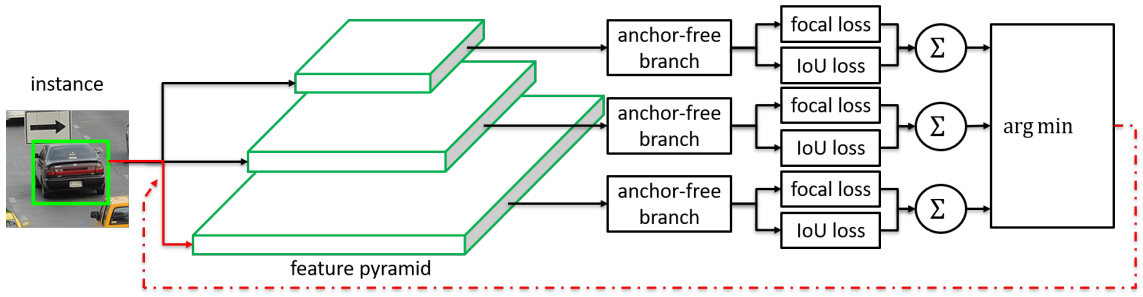

Figure 6: Online feature selection mechanism. Each instance is passing through all levels of anchor-free branches to compute the averaged classification (focal) loss and regression (IoU) loss over effective regions. Then the level with minimal summation of two losses is selected to set up the supervision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