DENSITY-BASED OBJECT DETECTION: LEARNING BOUNDING BOXES WITHOUT GROUND TRUTH ASSIGNMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

In multi-object detection using neural networks, most methods train a network based on ground truth assignment, which makes the training too heuristic and complicated. In this paper, we reformulate the multi-object detection task as a problem of density estimation of bounding boxes. Instead of using a ground-truth-assignment-based method, we train a network by estimating the probability density of bounding boxes in an input image using a mixture model. For this purpose, we propose a novel network for object detection called Mixture Density Object Detector (MDOD), and the corresponding objective function for the density-estimation-based training. Unlike the ground-truth-assignment-based methods, our proposed method gets rid of the cumbersome processes of matching between ground truth boxes and their predictions as well as the heuristic anchor design. It is also free from the problem of foreground-background imbalance. We applied MDOD to MS COCO dataset. Our proposed method not only deals with multi-object detection problems in a new approach, but also improves detection performances through MDOD. CODE WILL BE AVAILABLE.

1 INTRODUCTION

Multi-object detection is the task of finding multiple objects through bounding boxes with class information. Since the breakthrough of the convolutional neural networks (CNN), multi-object detection has been extensively developed in terms of computational efficiency and performance, and is now at a level that can be used in real life and industry.

The fundamental problem in training a multi-object detection network is, "How should the network learn a variable number of bounding boxes in different input images?". As an answer to this question, methods based on ground truth assignment (GTA) have been developed to train multi-object detection networks. These methods train multi-object detection networks by directly assigning a ground truth bounding box to specific locations (usually in a grid) of the network's output feature map with an appropriate criterion such as intersection-over-union (IoU) or center distance. The GTA-based methods have become the mainstream of training multi-object detection networks.

However, in order to successfully train a multi-object detection network using GTA, a thoughtful consideration of several procedures is required. These procedures, described below, make the training of a multi-object detection network too heuristic and complicated, which make the detection performance sensitive to the related hyper-parameters:

Matching ground truths and predictions: To assign a ground truth bounding box to specific locations of the network's output with a specific criterion, we need to determine whether the ground truth and each prediction match or not. Since it is the process of directly generating the target for each prediction and the ground truths that do not matched are ignored, the learning and the performace of a detector network are highly sensitive to the matching algorithm used (Zhang et al., 2020).

Various shape of anchor boxes: In the methods using anchor boxes, a ground truth bounding box must be assigned to one or more anchor box(es) for training a network. When using a generic matching algorithm, it is known that the detection performance is highly dependent on the shape, scale, and the number of anchor boxes (Ren et al., 2015; Lin et al., 2017b). Therefore, anchor boxes with various shapes and scales are needed to cope with various objects in different shapes and sizes.

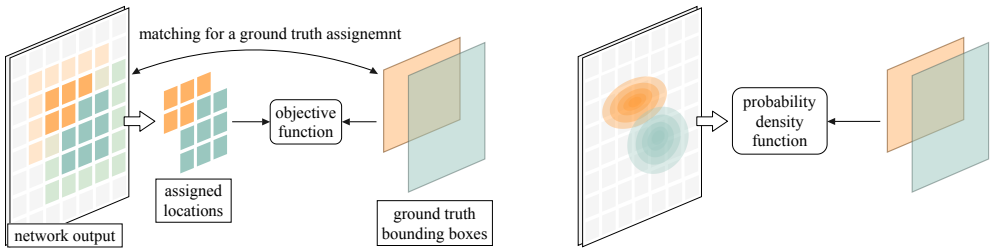

Figure 1: Comparison of ground-truth-assignment-based training (left) and density-estimation-based training (right). Density-estimation-based method trains the detector through a probability density function, without ground-truth-assignment.

Foreground-background imbalance: Generally, in the process of GTA, there exists a severe imbalance between the number of assigned locations that act as foreground and unassigned locations which correspond to backgrounds. This foreground-background imbalance problem makes training difficult. To alleviate this problem, separate processes such as heuristic sampling (Liu et al., 2016) or focal loss (Lin et al., 2017b) are required.

In this paper, we reformulate the multi-object detection task as a density estimation of bounding boxes (See Fig. 1). By doing this, we can train a multi-object detection network through a probability density function instead of using GTA that requires complex and heuristic processes. Our proposed multi-object detection network, Mixture-Density-based Object Detector (MDOD), captures the distribution of bounding boxes for an input image using a mixture model of components consisting of continuous (Cauchy) and discrete (categorical) probability distribution. For each component of the mixture model, the continuous Cauchy distribution is used to represent the distribution of the bounding box coordinates (left, top, right and bottom) and the categorical distribution is used to represent the class probability of that box. The MDOD is trained to maximize the log-likelihood of the estimated parameters for the mixture model given the ground truth bounding boxes of input images. The main contributions of the proposed method are threefold as the following:

1. Unlike the previous methods, we reformulate the multi-object detection task as a density estimation of bounding boxes for an input image. Through this novel approach of density esimation, a detection network can be trained without the ground truth assignment.

2. We estimate the density of bounding boxes using a mixture model consisting of continuous (for the location) and discrete (for the class) probability distribution. To this end, we propose a new network architecture, Mixture Density Object Detector (MDOD) and the objective function for it.

3. We measured the performance of our proposed method on MS COCO. We show that the proposed method is superior to GTA-based methods and can improve the detection performance of a state-the-art GTA-based detector, EfficientDet (Tan et al., 2020), by applying MDOD.

2 Related Works

In most modern multi-object detection methods, a ground truth bounding box must be assigned to the network's output based on center locations or anchor boxes. This process is called ground-truth-assignment (GTA). Faster R-CNN (Ren et al., 2015) attempts to represent the space in which a box can exist on an image as much as possible by using a large number of anchor boxes having various scales and aspect ratios. A ground truth bounding box is assigned to an anchor box if the IoU between this anchor box and the ground truth bounding box is above a threshold. In later studies, the use of anchor boxes became a standard. (Liu et al., 2016; Fu et al., 2017; Redmon et al., 2016). However, a large number of anchors worsens the so-called foreground-background imbalance problem, since the unassigned background anchor boxes outnumber the assigned foreground ones, which makes training difficult. Also, a careful design of the anchor is required as the scale and aspect ratio of the anchor affect detection performance much. To alleviate the foreground-background imbalance problem, Hard negative mining (Liu et al., 2016) and OHEM (Shrivastava et al., 2016) sample the negative RoIs (Region of Interests) with a high loss. Focal Loss (Lin et al., 2017b) tackles this problem by concentrating on the loss of hard examples. However, it has the hyperparameters

that should be heuristically searched. To design an anchor box, most of methods inherit the shape heuristically found in previous studies. YOLOv2 (Redmon & Farhadi, 2017) and YOLOv3 (Redmon & Farhadi, 2018) find the optimal anchor boxes through K-means clustering.

Recently, studies not using anchors have been conducted. Tian et al. (2019) learn ground truth bounding boxes based on the center location instead of anchor boxes. Law & Deng (2018); Duan et al. (2019); Zhou et al. (2019) use the keypoint-based method used in pose estimation. They learn the keypoints of the bounding boxes in the form of heatmaps. However, these methods still perform GTA and use focal loss to alleviate the foreground-background imbalance problem.

On other hand, there are also studies dealing with matching criteria in GTA. Zhang et al. (2020) argue that what is important is how to assign the ground truth bounding boxes, not the anchor box shapes, and propose an adaptive method that automatically divides positive and negative samples. FreeAnchor (Zhang et al., 2019) points out that the IoU-based hand-crafted assignment is a problem. It learns the matching between a ground truth bounding box and an anchor through maximum likelihood estimation, so the GTA is not determined by the IoU criterion. However, this only learns matching weights and it still needs to construct the hand-crafted bag of anchors based on IoU.

In the previous studies, the concept of distribution (e.g. Gaussian) in multi-object detection is mainly used to express the uncertainty of bounding box coordinates. For each predicted RoI (roi^k), He et al. (2019) model a ground truth bounding box coordinate (b^i_{coord}) as a Dirac delta function to estimate $p(b^i_{coord}|roi^k, image)$. Choi et al. (2019) estimate the density of a specific bounding box coordinate for a specific anchor ($anchor^k$) as a Gaussian distribution, $i.e.$ $p(b^i_{coord}|anchor^k, image) \sim \mathcal{N}$.

In this paper, we perform multi-object detection by learning the distribution of bounding boxes (b) for an image using a mixture model, $i.e.$ we estimate $p(b|image)$. Unlike the previous methods mentioned above, the GTA and the heuristics caused by GTA are not required to train our MDOD.

3 Problem Formulation: Mixture Model for Object Detection

The bounding box b can be represented as a vector consisting of four coordinates (position) b_p for the location (left-top and right-bottom corners) and an one-hot vector b_c for the object class. In the problem of multi-object detection, the conditional distribution of b for an image may be multi-modal, depending on the number of objects in an image. Therefore, our object detection network must be able to capture the multi-modal distribution. We propose a new model MDOD that can estimate the multi-modal distribution by extending the mixture density network (Bishop, 1994) for object detection. MDOD models the conditional distribution of b for an image using a mixture model whose components consist of continous and discrete probability distribution, which repspectively represents the distribution of bounding box coordinates and the class probability. In this paper, we use the Cauchy distribution as a continuous distribution and the categorical distribution as a discrete distribution. The probability density function (pdf) of this mixture model is defined as follows:

$$p(b|image) = \sum_{k=1}^{K} \pi_k \mathcal{F}(b_p; \mu_k, \gamma_k)\mathcal{P}(b_c; p_k). \tag{1}$$

Here, $\mathcal{F}$ denotes the pdf of Cauchy[1], and $\mathcal{P}$ denotes the probability mass function (pmf) of categorical distribution. The parameters μ_k, γ_k, and π_k are the location, scale, and, mixing coefficient of the k-th component. The C-dimensional vector p_k is the probability for C classes. The Cauchy distribution represents the four-coordinates of the bounding box $b_p = \{b_l, b_t, b_r, b_b\}$. To prevent the model from being overly complicated, we assume that each dimension of the bounding box coordinates is independent from the others. Thus, the pdf of Cauchy for the bounding box coordinates can be factorized as follows:

$$\mathcal{F}(b_p|image) = \prod_{d \in D} \mathcal{F}(b_d; \mu_{k,d}, \gamma_{k,d}), \quad D = \{l, t, r, b\}. \tag{2}$$

The objective of the MDOD is to accurately estimate the parameters of the mixture model by maximizing the log-likelihood of the ground truth bounding box b, as follows:

$$\theta = \arg\max_{\theta} \mathbb{E}_{b \sim p_{data}(b|image)} \log p(b|image; \theta). \tag{3}$$

[1] $\mathcal{F}(x; \mu, \gamma) = \frac{1}{\pi} \frac{\gamma}{(x-\mu)^2+\gamma^2}$, where μ is the location parameter and γ is the scaling parameter.

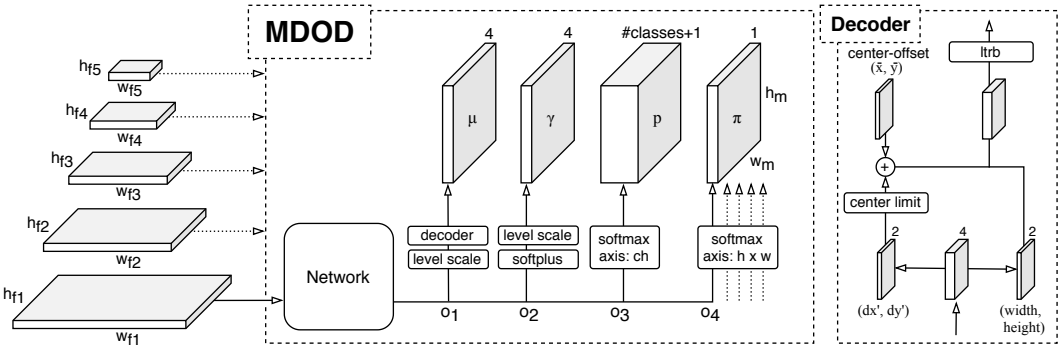

Figure 2: The architecture of MDOD. The parameters of the mixture model (μ, γ, p, and π) are predicted by MDOD. The network produces its intermediate output (o_1 - o_4) from each feature-map of the feature-pyramid.

Here, $p_{data}(b|image)$ is the empirical distribution of b for a given an input image and θ is the parameter vector that includes mixture parameters (μ_k, γ_k, π_k) and the class probability p_k.

4 Mixture Density Object Detector (MDOD)

4.1 Architecture

Fig. 2 shows the architecture of MDOD. The network outputs o_1, o_2, o_3, and o_4 from the input feature-map. The parameter maps of our mixture model, μ-map, γ-map, p-map, and π-map are obtained from o_1, o_2, o_3, and o_4, respectively. The mixture component is represented at each position on the spatial axis of the paramter-maps.

The μ-map is calculated from $o_1 \in \mathbb{R}^{h_m \times w_m \times 4}$. First, each element of o_1 is scaled by a factor of $s = \frac{1}{2^5} \times 2^l$ depending on what level $l \in \{1, \cdots, 5\}$ of feature map in the feature pyramid is used as follows: $o_1' = s \times o_1$. Then, as the decoder block in Fig. 2 depicts, the first two channels (dx', dy') of o_1' which correspond to the deviation from the center-offset are inputted to the center-limit operation. It restricts the output not to deviate too much from the center-offset which is a fixed map with two channels $(\bar{x}, \bar{y})$ that encodes spatial coordinates of each pixel on an input image. By adding the center-offset to the output of center-limit operation, the positions of the mixture components are spatially aligned to match the output of the network. The center-limit operation illustrated in Fig. 3 is implemented by applying $\tanh$ and multiplying the limit factor s_{lim}. In this paper, we set s_{lim} equal to the spacing between adjacent center-offsets (see Fig. 3). The overall computation of a center coordinate in x-direction is as follows: $x = \bar{x} + s_{lim} \times \tanh(dx')$. The same applies also to the y-direction. The last two channels of o_1' acts as the width and height. The lrtb-transformation converts coordinates represented by the center, width, and height $(xywh)$ to the left-top and right-bottom corners $(ltrb)$.

The γ-map is obtained by applying the softplus (Dugas et al., 2001) activation to o_2 and then multiplying the level-scale. The p-map is obtained by applying the softmax function along the channel axis to $o_3 \in \mathbb{R}^{h_m \times w_m \times (C+1)}$, and the π-map is obtained by applying the softmax to the entire five spatial maps of $o_4 \in \mathbb{R}^{h_m \times w_m \times 1}$ such that $\sum_{l=1}^{5} \sum_{h=1}^{h_m^l} \sum_{w=1}^{w_m^l} \pi_{(h,w)}^l = 1$. Here, C denotes the number of object classes and the last channel of o_3 is for the background class.

Our network consists of a convolution layer of 3×3 kernel and three convolution layers of 1×1 kernel. Swish (Ramachandran et al., 2017) is used for the activation function of these layers except the output layer. We use 5-level Feature Pyramid Network (FPN) as a feature extractor (Lin et al., 2017a). Our MDOD estimates only one mixture model from all levels of feature-maps. Thus, the number of components K is the summation of the number of components $(h_m \times w_m)$ of each parameter-map corresponding to the feature-map. Here, each feature-map $(o_1 - o_4)$ and the corresponding parameter-map (μ, γ, p, π) at the same layer have the same dimension.

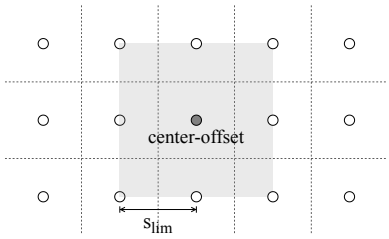

Figure 3: Illustration of the center-limit operation. The circles denote the center-offset. This operation limits μ_k within the gray area.

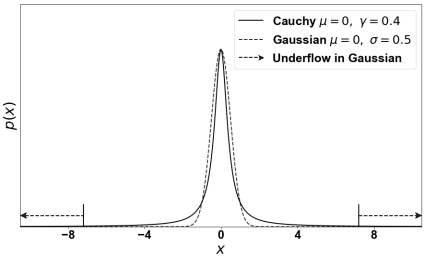

Figure 4: The pdfs of Gaussian and Cauchy distribution. Because of the limited precision of the floating point, for Gaussian, $p(x) = 0$ for $|x| > 7.202$, i.e. underflow in log-likelihood calculation.

4.2 TRAINING

Cauchy vs. Gaussian: Gaussian distibution is one of the representative continous probability distribution. But, the likelihood of Gaussian distribution decreases exponentially as the distance from μ increases. Therefore, even if the predicted coordinate is slightly far away from the ground truth, underflow may arise due to the limited floating point precision in the actual implementation. It causes the problem that the likelihood becomes zero and the loss can not be backpropagated. On the other hand, as can be seen in Fig. 4, the Cauchy distribution has a heavier (quadratically decreasing) tail compared to the Gaussian distribution. Thus, there is much little chance of the underflow problem.

RoI sampling: To take the probability of the negative predictions into account, the class probabilities considering the background are commonly used as the confidence score of the predicted bounding box. But, the set of ground truth bounding boxes $\{b_{gt}\}$ generally does not include the background class. To obtain the bounding boxes of both foreground and background classes, we sample bounding box candidates from the estimated mixture model ignoring the class probability, i.e. a mixture of Cauchy (MoC). If the IoU between a sampled candidate and a ground truth is above a threshold, we label it as the class of the ground truth with the highest IoU, otherwise, we label it as the background. Through this process, we create the RoI set $\{b_{roi}\}$. Since $\{b_{roi}\}$ is sampled from the MoC that is trained to represent the distribution of ground truth bounding box coordinates, the foreground-background imbalance problem does not occur if the MoC estimates the distribution of bounding boxes well. Also, the background bounding boxes in the $\{b_{roi}\}$ can be regarded as hard-negative samples that are acquired stochastically, not heuristically. In the matching of GTA, since ground truths are directly assigned to the network's output, the structure of output should be considered, e.g. anchor design or heatmap. But, in RoI sampling, we need only apply the commonly used criterion (IoU$>$0.5) of the background for RoI labeling.

Loss function: For training MDOD to represent the background probability using $\{b_{roi}\}$, we define the loss function of MDOD into two terms. The first term is the negative log-likelihood of the MoC:

$$\mathcal{L}_{MoC} = -\frac{1}{N_{gt}} \sum_{i=1}^{N_{gt}} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{F}(b^i_{gt,p}; \mu_k, \gamma_k) \right). \tag{4}$$

Here, (π_k, μ_k, γ_k) depends on the image that contains the i-th ground truth bounding box b^i_{gt}. Note that Eq.(4) learns only the distribution of the coordinates of the ground truth bounding box $\{b_{gt,p}\} = \{b^1_{gt,p}, \cdots, b^{N_{gt}}_{gt,p}\}$, excluding class probability using the MoC parameters (π, μ, γ). The second loss term is a complete form of our mixture model including class probability and is calculated as:

$$\mathcal{L}_{MM} = -\frac{1}{N_{roi}} \sum_{j=1}^{N_{roi}} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{F}(b^j_{roi,p}; \mu_k, \gamma_k) \mathcal{P}(b^j_{roi,c}; p_k) \right). \tag{5}$$

$\mathcal{L}_{MM}$ is used to learn the class probability of the estimated mixture model. Note that $\mathcal{L}_{MM}$ is calculated using $\{b_{roi}\} = \{b^1_{roi}, \cdots, b^{N_{roi}}_{roi}\}$ sampled from the estimated MoC. Also, it is trained such that the MoC is not relearned by itself. To this end, the error is not propagated to other parameters of mixture models except class probabilities p_k. The final loss function is defined as:

$$\mathcal{L} = \mathcal{L}_{MoC} + \alpha \mathcal{L}_{MM} \tag{6}$$

Here, α controls the balance between the two terms. In our experiments, we set $\alpha = 2$.

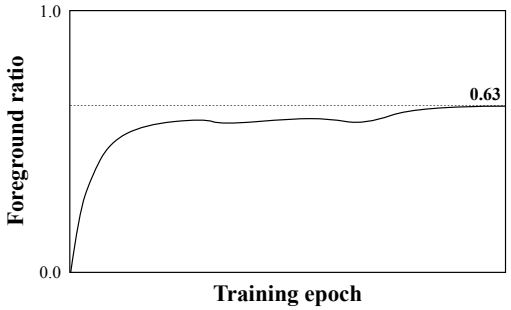

Figure 5: The ratio of foreground samples in the set $\{b_{roi}\}$ which is sampled from the mixture of Cauchy distribution at each training epoch.

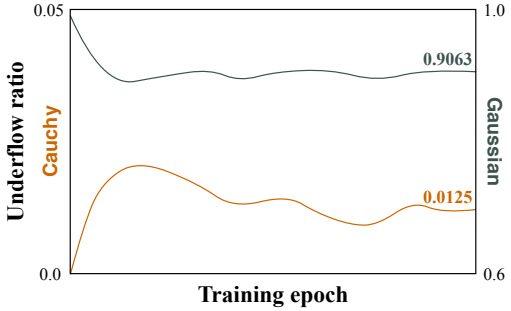

Figure 6: The ratios of underflowed components for Cauchy and Gaussian distributions at each training epoch.

4.3 Inference

In the inference phase, we choose μ's of mixture components as coordinates of the predicted bounding boxes. We assume that these μ's have a high possibility to be close to the local maxima of the estimated mixture model by MDOD. In the aspect of mixture-model-based clustering, we consider the μ's as representative values for the corresponding clusters. Before performing the non-maximum suppression (NMS), we can filter out the mixture components with relatively low $p(c)$ or π values. Since the scale of π depends on the input image, we filter mixture components through normalized-π (π'), which is obtained by normalizing π-vector with its maximum element, i.e. $\pi' = \pi / \max(\pi)$.

5 Experiments

5.1 Analysis for MDOD

To analyze the MDOD architecture, we use the MS COCO (Lin et al., 2014) 'train2017' and 'val2017' for training and evaluation. Input images are resized to 320×320, and ResNet50 (He et al., 2016) with FPN is used. Training details are described in the appendix (Section A.1).

Foreground-background balance: Since we perform sampling from the estimated MoC, the sampled set $\{b_{roi}\}$ contains both foreground and background samples. In order to check the balance of foreground and background in $\{b_{roi}\}$, we measure the foreground ratio (#foreground / #total) of $\{b_{roi}\}$. In Fig. 5, the foreground ratio is initially low but increases as training progresses and converges to a certain value. This shows that the foreground-background imbalance problem is solved naturally as the training progresses (#foreground : #background = 1.7 : 1 at the final epoch) .

Underflow ratio of Gaussian and Cauchy: In practice, the likelihood of Gaussian and Cauchy distribution can be zero due to underflow caused by the limited floating-point precision. In order to show this problem during training, we measure the ratio of components where underflow occurs due to a large distance from a ground truth bounding box coordinate. As can be seen in Fig. 4, in the Cauchy distribution, underflow rarely occurs, whereas in Gaussian, underflow occurs at a high ratio throughout the training process (about 0.9 ratio). The resultant APs of MDOD using Gaussian and Cauchy distribution are 32.7 and 33.8, respectively.

Table 1: The size of $\{b_{roi}\}$ (N_{roi}) and detection performances (APs).

N_{roi}	AP	AP_{50}
10	33.4	53.1
100	33.7	53.4
$N_{gt} \times 1$	33.8	53.3
$N_{gt} \times 3$	33.8	53.4
$N_{gt} \times 5$	33.9	53.3

Table 2: The effectiveness of the network components of MDOD.

	MDOD			
ltrb	✓	✓	✓	
center-limit	✓	✓		✓
level-scale	✓			✓
AP	33.8	32.9	32.3	32.8
AP_{50}	53.4	52.9	51.6	52.5

Table 3: Comparison of Baseline and EfficientDet with MDOD on MS COCO 'test-dev' dataset.

method	feature extractor	input size	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Baseline	ResNet50-FPN	320x320	30.1	45.9	32.4	6.4	34.7	50.8
Baseline	ResNet50-FPN	512x512	35.0	53.2	38.1	15.0	40.2	50.7
MDOD	ResNet50-FPN	320x320	33.9	53.8	35.5	14.7	35.1	49.6
MDOD	ResNet50-FPN	512x512	37.9	59.1	40.2	19.8	40.7	50.5
Baseline	ResNet101-FPN	320x320	31.1	46.8	33.6	6.7	36.1	52.3
Baseline	ResNet101-FPN	512x512	36.6	54.5	39.8	15.6	42.0	53.2
MDOD	ResNet101-FPN	320x320	35.0	54.8	36.8	14.4	36.5	15.8
MDOD	ResNet101-FPN	512x512	40.0	60.7	42.6	20.7	43.1	53.8
EfficientDet	Efficient-D0	512x512	33.8	52.2	35.8	12.0	38.3	51.2
EfficientDet	Efficient-D1	640x640	39.6	58.6	42.3	17.9	44.3	56.0
MDOD	Efficient-D0	512x512	35.2	56.5	36.8	16.9	37.3	48.7
MDOD	Efficient-D1	640x640	40.5	62.0	42.8	21.5	42.8	55.3

Table 4: Inference time (ms) comparison of Baseline and MDOD. 'net-time', 'pp-time' and 'total-time' mean network inference, post processing and total inference time, respectively.

method	feature extractor	input size	net-time	pp-time	total-time	FPS
Baseline	ResNet50-FPN	320x320	17	4	21	47.6
Baseline	ResNet50-FPN	512x512	22	6	28	37.5
MDOD	ResNet50-FPN	320x320	16	2	18	55.6
MDOD	ResNet50-FPN	512x512	21	2	23	43.5

The number of RoIs: Table 1 shows the performance changes according to N_{roi}, the size of $\{b_{roi}\}$. We either set N_{roi} proportional to N_{gt}, or fixed it independent of N_{gt}, the number of the ground truth bounding boxes. As a result of the experiment, the performance is not sensitive to the N_{roi}. In this paper, N_{roi} is set to three-times N_{gt}.

Ablation study: MDOD has components that play a specific role in the intermidate feature-map. In this experiment, we change the following components in the MDOD architecture one by one to see the effect: *ltrb-transformation (ltrb), center-limit* and *level-scale* operation. Table 2 shows the results. MDOD that uses all the components shows the best performance. Removing *center-limit* and *level-scale* operation results in a slight decrease in performance. The *center-limit* and *level-scale* operation seems to have a positive effect on detection results. If *ltrb-transformation* is not used, bounding box is learned in $xywh$ coordinate. In our method, learning through the $ltrb$ coordinate shows around 1.0 better APs than learning through $xywh$.

5.2 Evaluation result comparison

We compared MDOD with other object detection methods. MS COCO 'train2017' dataset is used as the training-set and 'test-dev2017' is used for evaluation. The frame-per-second (FPS) for MDOD is measured using a single nvidia Geforce 1080Ti including the post processing with batch size 1 without using tensorRT. Likewise, the FPSs for the other compared methods are also measured by the GPU with Nvidia Pascal architecture.

Comparison with GTA-based baseline: We set up a GTA-based simple baseline to compare the GTA-based methods and MDOD. In order to compare the two methods as fairly as possible, we use the completely same batch size, augmentation strategy, and network architecture excluding the output layer to the baseline and MDOD. The baseline network is trained by smooth l1 and the cross entropy with hard negative mining. And, the baseline uses nine shapes of anchor boxes per each cell of output. As can be seen in the Table 3, MDOD outperforms the baseline. Also, in Table 4, MDOD shows a faster inference speed than the baseline. The reasons are as follows: The predictions of

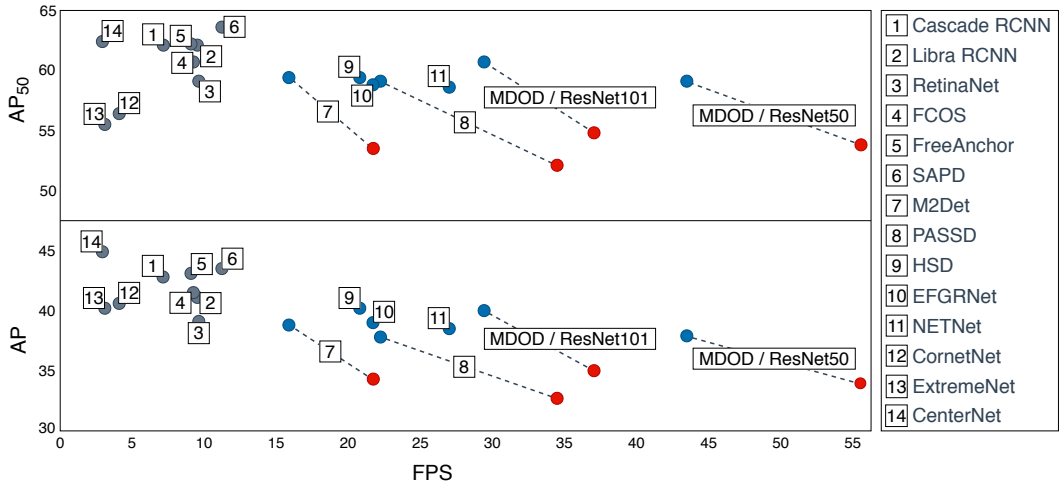

Figure 7: Comparison of speed and AP on MS COCO 'test-dev' dataset. Horizontal axis is the detection speed (FPS). Vertical axis of the top is AP_{50} and the bottom is AP. The red and blue circles denote 320x320 and 512x512 input sizes, respectively.

MDOD is only 1 for each cell in the output. Thus, the number of filters of output layer becomes smaller than that of the baseline (MDOD: 90, Baseline: 765). In addition, MDOD predicts fewer boxes than the Baselines (MDOD: 2134, Baseline: 19206). The number of predictions can affect the speed of the post-processing using NMS.

Comparison with EfficientDet: We compared the detection performance of MDOD with that of EfficientDet (Tan et al., 2020), a state-of-the-art GTA-based method. For the sake of fairness, the feature extractor used in EfficientDet is also applied to MDOD. In Table 3, this version of MDOD taking the structural superiority of EfficientDet's feature extractor shows better APs than the original EfficientDet in all the cases using the same feature extractor and input size. Especially, MDOD with Efficient-D1 achieved the highest AP (40.5) in this table. What is remarkable about these results is that this improvement is not caused by structural changes, heuristic or complex processing, but by a novel approach of learning distribution of bounding boxes in multi-object detection networks.

Comparison with other methods: We compared MDOD with other methods using the similar feature extractor to focus on the methodology of multi-object detection. Fig. 7 shows the APs and FPSs of these object detectors. MDOD shows better performance than others in both terms of detection performance and speed. The contribution of MDOD in terms of computation and speed is not prominent. But, since MDOD has the advantages mentioned in the comparison with baseline and does not use modified convolution modules that require more computation, MDOD shows faster inference speed than other methods when using the same input size. The corresponding table for this figure is attached in the appendix (Section A.3, Table 5).

6 Conclusion

In this paper, we treat the multi-object detection task as a density estimation of bounding boxes for an input image. Through this density-esimation-based approach, the detection network can be trained without heuristic and complex GTA processing. We proposed a new multi-object detector, MDOD, and the objective function to train it. MDOD captures the distribution of bounding boxes using the mixture model whose components consist of Cauchy and categorical distribution. Through thorough analysis, we verified that MDOD does not incur foreground-background imbalance problem and each component of MDOD contributes to the performance enhancement. Our MDOD shows improved detection performance compared to GTA-based methods. Notably, this performance is achieved not by structural changes or by heuristic and complex processings, but by a new approach of training multi-object detection networks. We believe that MDOD laid an initial step towards a new direction to multi-object detection which has a large room for improvements that can be achieved by further research and development.

REFERENCES

Christopher M Bishop. Mixture density networks. Technical report, Citeseer, 1994.

Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S Davis. Soft-nms–improving object detection with one line of code. In *Proceedings of the IEEE international conference on computer vision*, pp. 5561–5569, 2017.

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

Jiale Cao, Yanwei Pang, Jungong Han, and Xuelong Li. Hierarchical shot detector, 2019.

Jiwoong Choi, Dayoung Chun, Hyun Kim, and Hyuk-Jae Lee. Gaussian yolov3: An accurate and fast object detector using localization uncertainty for autonomous driving. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 502–511, 2019.

Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Object detection with keypoint triplets. *arXiv preprint arXiv:1904.08189*, 2019.

Charles Dugas, Yoshua Bengio, François Bélisle, Claude Nadeau, and René Garcia. Incorporating second-order functional knowledge for better option pricing. In *Advances in neural information processing systems*, pp. 472–478, 2001.

Cheng-Yang Fu, Wei Liu, Ananth Ranga, Ambrish Tyagi, and Alexander C Berg. Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pp. 770–778, 2016.

Yihui He, Chenchen Zhu, Jianren Wang, Marios Savvides, and Xiangyu Zhang. Bounding box regression with uncertainty for accurate object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2888–2897, 2019.

Ho-Deok Jang, Sanghyun Woo, Philipp Benz, Jinsun Park, and In So Kweon. Propose-and-attend single shot detector. *arXiv preprint arXiv:1907.12736*, 2019.

Hisham Cholakkal Fahad Shahbaz Khan, Yanwei Pang Ling Shao Jing Nie, Rao Muhammad Anwer. Enriched feature guided refinement network for object detection. 2019.

Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 734–750, 2018.

Yazhao Li, Yanwei Pang, Jianbing Shen, Jiale Cao, and Ling Shao. Netnet: Neighbor erasing and transferring network for better single shot object detection, 2020.

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

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

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017b.

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

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

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

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pp. 1310–1318, 2013.

Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions, 2017.

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

Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, pp. 91–99, 2015.

Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 761–769, 2016.

Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10781–10790, 2020.

Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. *arXiv preprint arXiv:1904.01355*, 2019.

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

Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9759–9768, 2020.

Xiaosong Zhang, Fang Wan, Chang Liu, Rongrong Ji, and Qixiang Ye. Freeanchor: Learning to match anchors for visual object detection. In *Advances in Neural Information Processing Systems*, pp. 147–155, 2019.

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

Xingyi Zhou, Jiacheng Zhuo, and Philipp Krahenbuhl. Bottom-up object detection by grouping extreme and center points. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 850–859, 2019.

Chenchen Zhu, Fangyi Chen, Zhiqiang Shen, and Marios Savvides. Soft anchor-point object detection. *arXiv preprint arXiv:1911.12448*, 2019.

A APPENDIX

A.1 TRAINING DETAILS OF MDOD

For training our network, we use the stochastic gradient descent optimization method with a momentum factor of 0.9. The learning rate is decayed at epoch 120 and 150 with a decay rate 0.1, and the network is trained up to 160 epochs. Here, the batch size is 32. Gradient clipping (Pascanu et al., 2013) is applied with a cutoff threshold of 7.0. We perform the generally used data augmentation process: the expansion, cropping and the horizotal flip described in (Liu et al., 2016). These are the same processes used in EfficientDet.

A.2 CONSIDERATIONS FOR FAIR COMPARISON WITH BASELINES

We came up with our own baseline detector model because the variables of each detector such as batch size, augmentation methods, network architecture and so on are so much different from detector to detector. We found it very hard to conduct fair and proper comparisons between various detectors due to these variables. Therefore, we designed our own baseline GTA-based model with our own controlled variables so that we could perform fair experiments for comparison. We use the completely same batch size, augmentation strategy, and network architecture excluding the output layer to the baseline and MDOD. And then, we tuned our baseline model by trying different hyper-parameters (positive-negative ratio, loss weight between regression and classification, weight-decay and learning rate) and tried to find the best that show the best results for the baseline model (30.1 AP when using ResNet50 and 320x320 size input image).

A.3 COMPARISON WITH OTHER METHODS

Table 5 shows comparison of speed and accuracy with 2-stage methods and 1-stage methods.

2-stage method: Cascade (Cai & Vasconcelos, 2018) and Libra (Pang et al., 2019)

1-stage method: RetinaNet (Lin et al., 2017b), FCOS (Tian et al., 2019), ATSS (Zhang et al., 2020), FreeAnchor (Zhang et al., 2019), SAPD (Zhu et al., 2019), RefineDet (Zhang et al., 2018), M2Det (Zhao et al., 2019), PASSD (Jang et al., 2019), HSD (Cao et al., 2019) , EFGRNet (Jing Nie, 2019) and NETNet (Li et al., 2020). CornerNet (Law & Deng, 2018), ExtremeNet (Zhou et al., 2019), and CenterNet (Duan et al., 2019). Keypoint-based methods are compared using Hourglass (Newell et al., 2016) as feature extractor due to those characteristics.

Table 5: Comparison of various results with MDOD on MS COCO 'test-dev' dataset. '$\circ$' and '$\star$' denote soft-nms (Bodla et al., 2017) and flip test (Law & Deng, 2018), respectively. The 'short-x' means to use an image that shorter side is resized as x while maintaining the aspect ratio, and the 'ori.' means using the original size input image in test.

Method	Feature extractor	Input size	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	FPS
Cascade R-CNN	ResNet-101 FPN+	short-800	42.8	62.1	46.3	23.7	45.5	55.2	7.1
Libra R-CNN	ResNet-101 FPN	short-800	41.1	62.1	44.7	23.4	43.7	52.5	9.5
RetinaNet	ResNet-101 FPN	short-800	39.1	59.1	42.3	21.8	42.7	50.2	9.6
FCOS	ResNet-101 FPN	short-800	41.5	60.7	45.0	24.4	44.8	51.6	9.3
ATSS	ResNet-101 FPN	short-800	43.6	62.1	47.4	26.1	47.0	53.6	-
FreeAnchor	ResNet-101 FPN	short-800	43.1	62.2	46.4	24.5	46.1	54.8	9.1
SAPD	ResNet-101	short-800	43.5	63.6	46.5	24.9	46.8	54.6	11.2
RefineDet320	ResNet-101 TCB	320x320	32.0	51.4	34.2	10.5	34.7	50.4	-
RefineDet512	ResNet-101 TCB	512x512	36.4	57.5	39.5	16.6	39.9	51.4	-
M2Det320	ResNet-101 MLFPN	320x320	34.3	53.5	36.5	14.8	38.8	47.9	21.7
M2Det512	ResNet-101 MLFPN	512x512	38.8	59.4	41.7	20.5	43.9	53.4	15.8
PASSD320$\circ$	ResNet-101 FPN	320x320	32.7	52.1	35.3	10.8	36.5	50.2	34.5
PASSD512$\circ$	ResNet-101 FPN	512x512	37.8	59.1	41.4	19.3	42.6	51.0	22.2
HSD	ResNet-101	512x512	40.2	59.4	44.0	20.0	44.4	50.2	20.8
EFGRNet	ResNet-101	512×512	39.0	58.8	42.3	17.8	43.6	54.5	21.7
NETNet	ResNet-101	512×512	38.5	58.6	41.3	19.0	42.3	53.9	27.0
CornerNet$\circ\star$	Hourglass-104	511x511 (ori.)	40.6	56.4	43.2	19.1	42.8	54.3	4.1
ExtremeNet$\circ\star$	Hourglass-104	511x511 (ori.)	40.2	55.5	43.2	20.4	43.2	53.1	3.1
CenterNet$\circ\star$	Hourglass-104	511x511 (ori.)	44.9	62.4	48.1	25.6	47.4	57.4	2.9
MDOD320	ResNet-50 FPN	320x320	33.9	53.8	35.5	14.7	35.1	49.6	55.6
MDOD512	ResNet-50 FPN	512x512	37.9	59.1	40.2	19.8	40.7	50.5	43.5
MDOD320	ResNet-101 FPN	320x320	35.0	54.8	36.8	14.4	36.5	15.8	37.0
MDOD512	ResNet-101 FPN	512x512	40.0	60.7	42.6	20.7	43.1	53.8	29.4