This ECCV 2018 paper, provided here by the Computer Vision Foundation, is the author-created version.
The content of this paper is identical to the content of the officially published ECCV 2018
LNCS version of the paper as available on SpringerLink: https://link.springer.com/conference/eccv

Deep Regionlets for Object Detection

Hongyu Xu[1⋆], Xutao Lv[2], Xiaoyu Wang[2], Zhou Ren[3],
Navaneeth Bodla[1] and Rama Chellappa[1]
[1]University of Maryland, College Park, Maryland, USA
[2]Intellifusion [3]Snap Inc.

[1]{hyxu,nbodla,rama}@umiacs.umd.edu
[2]{lvxutao,fanghuaxue}@gmail.com [3]zhou.ren@snap.com

Abstract. In this paper, we propose a novel object detection framework named "Deep Regionlets" by establishing a bridge between deep neural networks and conventional detection schema for accurate generic object detection. Motivated by the abilities of regionlets for modeling object deformation and multiple aspect ratios, we incorporate regionlets into an end-to-end trainable deep learning framework. The deep regionlets framework consists of a region selection network and a deep regionlet learning module. Specifically, given a detection bounding box proposal, the region selection network provides guidance on where to select regions to learn the features from. The regionlet learning module focuses on local feature selection and transformation to alleviate local variations. To this end, we *first* realize *non-rectangular* region selection within the detection framework to accommodate variations in object appearance. Moreover, we design a "gating network" within the regionlet leaning module to enable soft regionlet selection and pooling. The Deep Regionlets framework is trained end-to-end without additional efforts. We perform ablation studies and conduct extensive experiments on the PASCAL VOC and Microsoft COCO datasets. The proposed framework outperforms state-of-the-art algorithms, such as RetinaNet and Mask R-CNN, even without additional segmentation labels.

Keywords: Object Detection, Deep Learning, Deep Regionlets, Spatial Transformation

1 Introduction

Generic object detection has been extensively studied by the computer vision community over several decades [22, 4, 44, 16, 17, 37, 8, 26, 45, 42, 10, 13, 6, 41, 48] due to its appeal to both academic research explorations as well as commercial applications. Given an image of interest, the goal of object detection is to predict the locations of objects and classify them at the same time. The key challenge of the object detection task is to handle variations in object scale, pose, viewpoint and even part deformations when generating the bounding boxes for specific object categories.

⋆ Work started during an internship at Snap Research

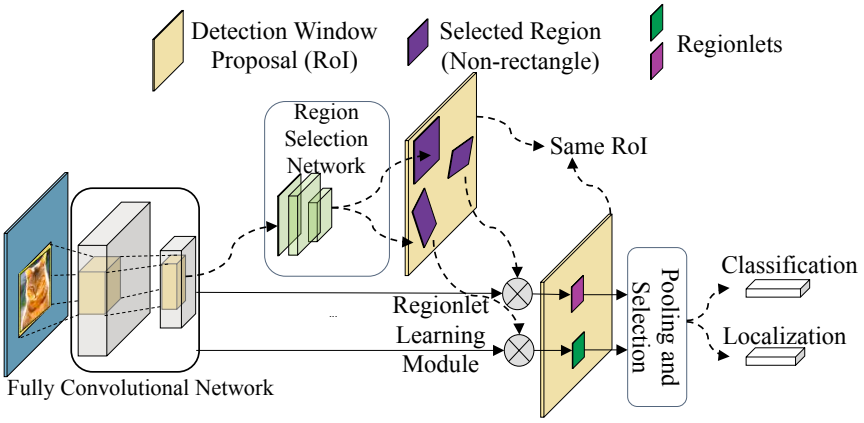

Fig. 1. Architecture of the Deep Regionlets detection framework. It consists of a region selection network (RSN) and a deep regionlet learning module. The region selection network performs *non-rectangular* region selection from the detection window proposal generated by the region proposal network. Deep regionlet learning module learns the regionlets through a spatial transformation and a gating network. The entire pipeline is end-to-end trainable. For better visualization, the region proposal network is not displayed here.

Numerous methods have been proposed based on hand-crafted features (*i.e.* HOG [10], LBP [1], SIFT [30]). These approaches usually involve an exhaustive search for possible locations, scales and aspect ratios of the object, by using the sliding window approach. However, Wang *et al.*'s [45] regionlet-based detection framework has gained a lot of attention as it provides the flexibility to deal with different scales and aspect ratios without performing an exhaustive search. It first introduced the concept of **regionlet** by defining a three-level structural relationship: candidate bounding boxes (sliding windows), regions inside the bounding box and groups of regionlets (sub-regions inside each region). It operates by directly extracting features from regionlets in several selected regions within an arbitrary detection bounding box and performs (max) pooling among the regionlets. Such a feature extraction hierarchy is capable of dealing with variable aspect ratios and flexible feature sets, which leads to improved learning of robust feature representation of the object for region-based object detection.

Recently, deep learning has achieved significant success on many computer vision tasks such as image classification [24, 20, 34], semantic segmentation [29] and object detection [16] using the deep convolutional neural network (DCNN) architecture. Despite the excellent performance of deep learning-based detection framework, most network architectures [37, 8, 28] do not take advantage of successful conventional ideas such as deformable part-based model (DPM) or *regionlets*. Those methods have been effective for modeling object deformation, sub-categories and multiple aspect ratios. Recent advances [33, 9, 32] have

achieved promising results by combining the conventional DPM-based detection methodology with deep neural network architectures.

These observations motivate us to establish a bridge between deep convolutional neural network and conventional object detection schema. In this paper, we incorporate the conventional Regionlet method into an end-to-end trainable deep learning framework. Despite being able to handle arbitrary bounding boxes, several drawbacks arise when directly integrating the regionlet methodology into the deep learning framework. First, in [45], Wang *et al.* proposed to learn cascade object classifiers after hand-crafted feature extraction in each regionlet. However, end-to-end learning is not feasible in this framework. Second, regions in regionlet-based detection have to be rectangular, which does not effectively model the deformations of an object which results in variable shapes. Moreover, both regions and regionlets are fixed after training is completed.

To this end, we propose a novel object detection framework named "Deep Regionlets" to integrate the deep learning framework into the traditional regionlet method [45]. The overall design of the proposed detection system is illustrated in Figure 1. It consists of a region selection network (RSN) and a deep regionlet learning module. The region selection network performs *non-rectangular* region selection from the detection window proposal[1] (RoI) to address the limitations of the traditional regionlet approach. We further design a deep regionlet learning module to learn the regionlets through a spatial transformation and a gating network. By using the proposed gating network, which is a soft regionlet selector, the resulting feature representation is more effective for detection. The entire pipeline is end-to-end trainable using only the input images and ground truth bounding boxes.

We conduct a detailed analysis of our approach to understand its merits and evaluate its performance. Extensive experiments on two detection benchmark datasets, PASCAL VOC [11] and Microsoft COCO [27] show that the proposed deep regionlet approach outperforms several competitors [37, 8, 9, 32]. Even without segmentation labels, we outperform state-of-the-art algorithms such as Mask R-CNN [18] and RetinaNet [26]. To summarize, we make the following contributions:

- We propose a novel deep regionlet approach for object detection. Our work extends the traditional regionlet method to the deep learning framework. The system is trainable in an end-to-end manner.
- We design the RSN, which **first** performs **non-rectangular** region selection within the detection bounding box generated from a detection window proposal. It provides more flexibility in modeling objects with variable shapes and deformable parts.
- We propose a deep regionlet learning module, including feature transformation and a gating network. The gating network serves as a soft regionlet selector and lets the network focus on features that benefit detection performance.

[1] The detection window proposal is generated by a region proposal network (RPN) [37, 8, 17]. It is also called region of interest (ROI)

– We present empirical results on object detection benchmark datasets, demonstrating superior performance over state-of-the-art.

2 Related Work

Many approaches have been proposed for object detection including both traditional ones [13, 45, 42] and deep learning-based approaches [17, 37, 28, 35, 8, 16, 19, 9, 32, 6, 21, 51, 52, 50, 48, 43, 41]. Traditional approaches mainly used hand-crafted features to train the object detectors using the sliding window paradigm. One of the earliest works [42] used boosted cascaded detectors for face detection, which led to its wide adoption. Deformable Part Model-based detection (DPM) [12] proposed the concept of deformable part models to handle object deformations. Due to the rapid development of deep learning techniques [24, 20, 40, 5, 49, 34, 47, 2, 46], the deep learning-based detectors have become dominant object detectors.

Deep learning-based detectors could be further categorized into single-stage detectors and two-stage detectors, based on whether the detectors have proposal-driven mechanism or not. The single-stage detectors [38, 35, 28, 14, 25, 26, 48, 50] apply regular, dense sampling windows over object locations, scales and aspect ratios. By exploiting multiple layers within a deep CNN network directly, the single-stage detectors achieved high speed but their accuracy is typically low compared to two-stage detectors.

Two-stage detectors [17, 37, 8] involve two steps. They first generate a sparse set of candidate proposals of detection bounding boxes by the Region Proposal Network (RPN). After filtering out the majority of negative background boxes by RPN, the second stage classifies the proposals of detection bounding boxes and performs the bounding box regression to predict object categories and their corresponding locations. The two-stage detectors consistently achieve higher accuracy than single-stage detectors and numerous extensions have been proposed [9, 32, 18, 6, 41, 21, 7]. Our method follows the two-stage detector architecture by taking advantage of RPN without requiring dense sampling of object locations, scales and aspect ratios.

3 Our Approach

In this section, we first review the traditional regionlet-based detection methods and then present the overall design of the end-to-end trainable deep regionlet approach. Finally, we discuss in detail each module in the proposed end-to-end deep regionlet approach.

3.1 Traditional Regionlet-based Approach

A *regionlet* is a base feature extraction region defined proportionally to a window (*i.e.* a sliding window or a detection bounding box) at arbitrary resolution (*i.e.* size and aspect ratio). Wang *et al.* [45] first introduced the concept of

regionlet, as illustrated in Figure 2. It defines a three-level structure among a detecting bounding box, number of regions inside the bounding box and a group of regionlets (sub-regions inside each region). In Figure 2, the yellow box is a detection bounding box. R is a rectangular feature extraction region inside the bounding box. Furthermore, small sub-regions $r_{i\{i=1...N\}}$(*e.g.* r_1, r_2) are chosen within region R, where we define them as a set of *regionlets*.

The difficulty of the arbitrary detection bounding box has been well addressed by using the *relative* positions and sizes of regionlets and regions. However, in the traditional approach, the initialization of regionlets possess randomness and both regions (R) and regionlets (*i.e.* r_1, r_2) are fixed after the training. Moreover, it is based on hand-crafted features (*i.e.* HOG [10] or LBP [1]) in each regionlet respectively and hence not end-to-end trainable. To this end, we propose the following deep regionlet-based approach to address such limitations.

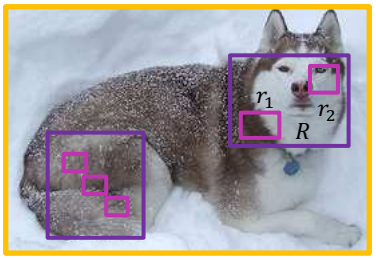

Fig. 2. Illustration of structural relationships among the detection bounding box, feature extraction regions and regionlets. The yellow box is a detection bounding box and R is a feature extraction region shown as a purple rectangle with filled dots inside the bounding box. Inside R, two small sub-regions denoted as r_1 and r_2 are the *regionlets*.

3.2 System Architecture

Generally speaking, an object detection network performs a sequence of convolutional operations on an image of interest using a deep convolutional neural network. At some layer, the network bifurcates into two branches. One branch, RPN generates a set of candidate bounding boxes[2] while the other branch performs classification and regression by pooling the convolutional features inside the proposed bounding box generated by the region proposal network [37, 8]. Taking advantage of this detection network, we introduce the overall design of the proposed object detection framework, named "Deep Regionlets", as illustrated in Figure 1.

The general architecture consists of an RSN and a deep regionlet learning module. In particular, the RSN is used to predict the transformation parameters to choose regions given a candidate bounding box, which is generated by the

[2] [37, 8, 17] also called the detection bounding box as detection window proposal

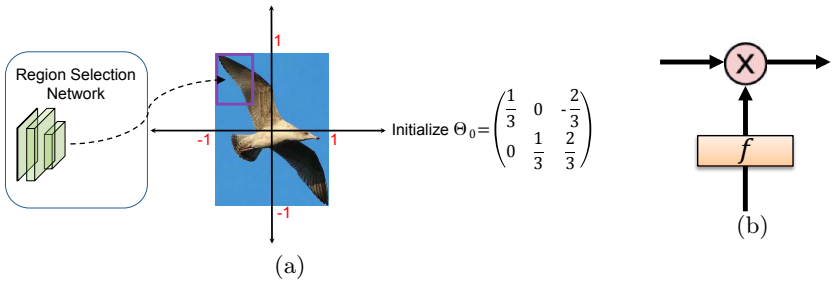

Fig. 3. (a) Example of initialization of one affine transformation parameter. Normalized affine transformation parameters $\Theta_0 = [\frac{1}{3}, 0, -\frac{2}{3}; 0, \frac{1}{3}, \frac{2}{3}]$ ($\theta_i \in [-1, 1]$) selects the top-left region in the 3×3 evenly divided detection bounding box, shown as the purple rectangle. (b) Design of the gating network. f denotes the non-negative gate function

region proposal network. The regionlets are further learned within each selected region defined by the region selection network. The system is designed to be trained in a fully end-to-end manner using only the input images and ground truth bounding box. The RSN as well as the regionlet learning module can be simultaneously learned over each selected region given the detection window proposal.

3.3 Region Selection Network

We design the RSN to have the following properties: 1) End-to-end trainable; 2) Simple structure; 3) Generate regions with arbitrary shapes. Keeping these in mind, we design the RSN to predict a set of *affine* transformation parameters. By using these affine transformation parameters, as well as not requiring the regions to be rectangular, we have more flexibility in modeling objects with arbitrary shapes and deformable parts.

Specifically, we design the RSN using a small neural network with three fully connected layers. The first two fully connected layers have output size of 256, with ReLU activation. The last fully connected layer has the output size of six, which is used to predict the set of affine transformation parameters $\Theta = [\theta_1, \theta_2, \theta_3; \theta_4, \theta_5, \theta_6]$.

Note that the candidate detection bounding boxes proposed by RSN have arbitrary sizes and aspect ratios. In order to address this difficulty, we use *relative* positions and sizes of the selected region within a detection bounding box. The candidate bounding box generated by the RPN is defined by the top-left point (w_0, h_0), width w and height h of the box. We normalize the coordinates by the width w and height h of the box. As a result, we could use the normalized affine transformation parameters $\Theta = [\theta_1, \theta_2, \theta_3; \theta_4, \theta_5, \theta_6]$ ($\theta_i \in [-1, 1]$) to evaluate one selected region within one candidate detection window at different sizes and aspect ratios without scaling images into multiple resolutions or using multiple-components to enumerate possible aspect ratios, like anchors [37, 28, 14].

Initialization of Region Selection Network: Taking advantage of *relative* and *normalized* coordinates, we initialize the RSN by equally dividing the whole detecting bounding box to several sub-regions, named as *cell*s, without any overlap among them. Figure 3(a) shows an example of initialization from one affine transformation (*i.e.* 3×3). The first cell, which is the top-left bin in the whole region (detection bounding box) could be defined by initializing the corresponding affine transformation parameter $\Theta_0 = [\frac{1}{3}, 0, -\frac{2}{3}; 0, \frac{1}{3}, \frac{2}{3}]$. The other eight of 3×3 cells are initialized in a similar way.

3.4 Deep Regionlet Learning

After regions are selected by the RSN, regionlets are further learned from the selected region defined by the normalized affine transformation parameters. Note that our motivation is to design the network to be trained in a fully end-to-end manner using only the input images and ground truth bounding boxes. Therefore, both the selected regions and regionlet learning should be able to be trained by CNN networks. Moreover, we would like the regionlets extracted from the selected regions to better represent objects with variable shapes and deformable parts.

Inspired by the spatial transform network [23], any parameterizable transformation including translation, scaling, rotation, affine or even projective transformation can be learned by a spatial transformer. In this section, we introduce our deep regionlet learning module to learn the regionlets in the selected region, which is defined by the affine transformation parameters.

More specifically, we aim to learn regionlets from one selected region defined by one affine transformation Θ to better match the shapes of objects. This is done with a selected region R from RSN, transformation parameters $\Theta = [\theta_1, \theta_2, \theta_3; \theta_4, \theta_5, \theta_6]$ and a set of feature maps $Z = \{Z_i, i = 1, \ldots, n\}$. Without loss of generality, let Z_i be one of the feature map out of the n feature maps. A selected region R is of size $w \times h$ with the top-left corner (w_0, h_0). Inside the Z_i feature maps, we propose the following regionlet learning module.

Let s denote the source and t denote target, we define (x_p^s, y_p^s) as the spatial location in original feature map Z_i and (x_p^s, y_p^s) as the spatial location in the output feature maps after spatial transformation. U_{nm}^c is the value at location (n, m) in channel c of the input feature. The total output feature map V is of size $H \times W$. Let $V(x_p^t, y_p^t, c|\Theta, R)$ be the output feature value at location (x_p^t, y_p^t) ($x_p^t \in [0, H]$, $y_p^t \in [0, W]$) in channel c, which is computed as

$$V(x_p^s, y_p^s, c|\Theta, R) = \sum_n^H \sum_m^M U_{nm}^c \max(0, 1 - |x_p^s - m|) \max(0, 1 - |y_p^s - n|) \quad (1)$$

Back Propagation through Spatial Transform To allow back propagation of the loss through the regionlet learning module, we can define the gradients with respect to both feature maps and the region selection network. In this layer's `backward` function, we have partial derivative of the loss function with

respect to both feature map variable U^c_{mn} and affine transform parameter $\Theta = [\theta_1, \theta_2, \theta_3; \theta_4, \theta_5, \theta_6]$. Motivated by [23], the partial derivative of the loss function with respect to the feature map is:

$$\frac{\partial V(x^s_p, y^s_p, c|\Theta, R)}{\partial U^c_{nm}} = \sum_n^H \sum_m^M \max(0, 1 - |x^s_p - m|) \times \max(0, 1 - |y^s_p - n|) \quad (2)$$

Moreover, during back propagation, we need to compute the gradient with respect to each affine transformation parameter $\Theta = [\theta_1, \theta_2, \theta_3; \theta_4, \theta_5, \theta_6]$. In this way, the region selection network could also be updated to adjust the selected region. We take θ_1 as an example due to space limitations and similar derivative can be computed for other parameters $\theta_i (i = 2, \ldots, 6)$ respectively.

$$\frac{\partial V(x^s_p, y^s_p, c|\Theta, R)}{\partial \theta_1} = x^t_p \sum_n^H \sum_m^M U^c_{nm} \max(0, 1 - |y^s_p - n|) \times \begin{cases} 0 \text{ if } |m - x^s_p| \geq 1 \\ 1 \text{ if } m > x^s_p \\ -1 \text{ if } m < x^s_p \end{cases} \quad (3)$$

It is worth noting that (x^t_p, y^t_p) are normalized coordinates in range $[-1, 1]$ so that it can to be scaled with respect to w and h with start position (w_0, h_0).

Gating Network The gating network, which serves as a soft regionlet selector, is used to assgin regionlets with different weights and generate regionlet feature representation. We design a simple gating network using a fully connected layer with `sigmoid` activation, shown in Figure 3(b). The output values of the gating network are within range of $[0, 1]$. Given the output feature maps $V(x^s_p, y^s_p, c|\Theta, R)$ described above, we use a fully connected layer to generate the same number of output as feature maps $V(x^s_p, y^s_p, c|\Theta, R)$, which is followed by an activation layer `sigmoid` to generate the corresponding weight respectively. The final feature representation is generated by the product of feature maps $V(x^s_p, y^s_p, c|\Theta, R)$ and their corresponding weights.

Regionlet Pool Construction Object deformations may occur at different scales. For instance, deformation could be caused by different body parts in person detection. Same number of regionlets (size $H \times W$) learned from small selected region have higher extraction density, which may lead to non-compact regionlet representation. In order to learn a *compact*, *efficient* regionlet representation, we further perform the pooling (*i.e.* max/ave) operation over the feature maps $V(x^s_p, y^s_p, c|\Theta, R)$ of size $(H \times W)$. We reap two benefits from the pool construction: (1) Regionlet representation is compact (small size). (2) Regionlets learned from different size of selected regions are able to represent such regions in the same efficient way, thus to handle object deformations at different scales.

3.5 Relations to Recent Works

Our deep regionlet approach is related to some recent works in different aspects. We discuss both similarities and differences in detail in the supplementary material section.

4 Experiments

In this section, we present comprehensive experimental results of the proposed approach on two challenging benchmark datasets: PASCAL VOC [11] and MS-COCO [27]. There are in total 20 categories of objects in PASCAL VOC [11] dataset. We follow the common settings used in [37, 4, 8, 17] to enable fair comparsions.

More specifically, we train our deep model on (1) VOC 2007 `trainval` and (2) union of VOC 2007 `trainval` and 2012 `trainval` and evaluate on VOC2007 `test`. We also report results on VOC 2012 `test`, following the suggested settings in [37, 4, 8, 17]. In addition, we report the results on the VOC2007 `test` split for ablation studies. MS-COCO [27] contains 80 object categories. Following the official settings in COCO website , we use the COCO 2017 `trainval` split (union of 135k images from `train` split and 5k images from `val` split) for training. We report the COCO-style average precision (AP) on `test-dev` 2017 split, which requires evaluation from the MS-COCO server.

For the base network, we choose both VGG-16 [40] and ResNet-101 [20] to demonstrate the generalization of our approach regardless of which network backbone we use. The *á trous* algorithm [29, 31] is adopted in stage 5 of ResNet-101. Following the suggested settings in [8, 9], we also set the pooling size to 7 by changing the conv5 stage's effective stride from 32 to 16 to increase the feature map resolution. In addition, the first convolution layer with stride 2 in the conv5 stage is modified to 1. Both backbone networks are intialized with the pre-trained ImageNet [20, 24] model. In the following sections, we report the results of a series of ablation experiments to understand the behavior of the proposed deep regionlet approach. Furthermore, we present comparisons with state-of-the-art detectors [37, 8, 9, 18, 26, 25] on both PASCAL VOC [11] and MS COCO [27] datasets.

4.1 Ablation Study

For a fair comparison, we adopt ResNet-101 as the backbone network for ablation studies. We train our model on the union set of VOC 2007 + 2012 `trainval` and evaluate on the VOC2007 `test` set. The shorter side of image is set to be 600 pixels, as suggested in [17, 37, 8]. The training is performed for 60k iterations with an effective mini-batch size 4 on 4 GPUs, where the learning rate is set at 10^{-3} for the first 40k iterations and at 10^{-4} for the remaining 20k iterations. First we investigate the proposed approach to understand each component (1) RSN, (2) Deep regionlet learning and (3) Soft regionlet selection by comparing it with several baselines:

(1) Global RSN. RSN only selects one global region and it is initialized as identity transformation (*i.e.* $\Theta_0 = [1, 0, 0; 0, 1, 0]$). This is equivalent to global regionlet learning within the RoI.

(2) Offset-only RSN. We set the RSN to only learn the offset by enforcing $\theta_1, \theta_2, \theta_4, \theta_5$ not to change during the training process. In this way, the region

Methods	Global RSN	Offset-only RSN [9, 32]	Non-gating	Ours
mAP@0.5(%)	30.27	78.5	81.3 (+2.8)	82.0 (+3.5)

Table 1. Ablation study of each component in deep regionlet approach. Output size $H \times W$ is set to 4×4 for all the baselines

Regionlets Density / # of Regions	2×2	3×3	4×4	5×5	6×6
$4(2 \times 2)$ regions	78.0	79.2	79.9	80.2	80.3
$9(3 \times 3)$ regions	79.6	80.3	80.9	81.5	81.3
$16(4 \times 4)$ regions	80.0	81.0	82.0	81.6	80.8

Table 2. Results of ablation studies when the RSN selects different number of regions and regionlets are learned at different level of density.

selection network only selects the rectangular region with offsets to the initialized region. This baseline is similar to the Deformable RoI Pooling in [9] and [32].

(3) Non-gating selection: Deep regionlet without soft selection. No soft regionlet selection is performed after the regionlet learning. In this case, each regionlet learned has the same contribution to the final feature representation.

Results are shown in Table 1. First, when the region selection network only selects one global region, the RSN reduces to the single localization network [23]. In this case, regionlets will be extracted in a global manner. It is interesting to note that selecting only one region by the region selection network is able to converge, which is different from [37, 8]. However, the performance is extremely poor. This is because no discriminative regionlets could be explicitly learned within the region. More importantly, when we compare our approach and offset-only RSN with global RSN, the results clearly demonstrate that the RSN is *indispensable* in the deep regionlet approach.

Moreover, offset-only RSN could be viewed as similar to deformable RoI pooling in [9, 32]. These methods all learn the offset of the rectangle region with respect to its reference position, which lead to improvement over [37]. However, non-gating selection outperforms offset-only RSN by 2.8% while selecting the non-rectangular region. The improvement demonstrates that non-rectangular region selection could provide more flexibility around the original reference region, thus could better model the non-rectangular objects with sharp shapes and deformable parts. Last but not least, by using the gate function to perform soft regionlet selection, the performance can be further improved by 0.7%.

Next, we present ablation studies on the following questions in order to understand more deeply on the region selection network and regionlet learning module: (1) How many regions should we learn using the region selection network? (2) How many regionlets should we learn in a selected region (density is of size $H \times W$)?

How many regions should we learn using the region selection network? We investigate how the detection performance varies when different number of regions are selected by the region selection network. All the regions are initialized

Methods	training data	mAP@0.5(%)	training data	mAP@0.5(%)
Regionlet [45]	07	41.7	07 + 12	N/A
Faster R-CNN [37]	07	70.0	07 + 12	73.2
R-FCN [8]	07	69.6	07 + 12	76.6
SSD 512 [28]	07	71.6	07 + 12	76.8
Soft-NMS [4]	07	71.1	07 + 12	76.8
Ours	07	***73.0***	07 + 12	***79.2***
Ours[§]	07	**73.8**	07 + 12	**80.1**

Table 3. Detection results on PASCAL VOC using VGG16 as backbone architecture. Training data: "07": VOC2007 `trainval`, "07 + 12": VOC 2007 and 2012 `trainval`. Ours[§] denotes applying the soft-NMS [4] in the test stage.

as described in Section 3.3 without any overlap between regions. Without loss of generality, we report results for $4(2 \times 2)$, $9(3 \times 3)$ and $16(4 \times 4)$ regions in Table 2. We observe that the mean AP increases when the number of selected regions is increased from $4(2 \times 2)$ to $9(3 \times 3)$ for a fixed regionlets learning number, but gets saturated with $16(4 \times 4)$ selected regions.

How many regionlets should we learn in one selected region? Next, we investigate how the detection performance varies when different number of regionlets are learned in one selected region by varying H and W. Without loss of generality, we set $H = W$ and vary the H value from 2 to 6. In Table 2, we report results when we set the number of regionlets at $4(2 \times 2)$, $9(3 \times 3)$, $16(4 \times 4)$, $25(5 \times 5)$, $36(6 \times 6)$ before the regionlet pooling construction.

First, it is observed that increasing the number of regionlets from $4(2 \times 2)$ to $25(5 \times 5)$ results in improved performance. As more regionlets are learned from one region, more spatial and shape information from objects could be learned. The proposed approach could achieve the best performance when regionlets are extracted at $16(4 \times 4)$ or $25(5 \times 5)$ density level. It is also interesting to note that when the density increases from $25(5 \times 5)$ to $36(6 \times 6)$, the performance degrades slightly. When the regionlets are learned at a very high density level, some redundant spatial information may be learned without being useful for detection, thus affecting the region proposal-based decision to be made. In all the experiments, we present the results from 16 selected regions from the RSN and set output size $H \times W = 4 \times 4$.

4.2 Experiments on PASCAL VOC

In this section, we compare our results with a traditional regionlet method [45] and several state-of-the-art deep learning-based object detectors as follows: Faster R-CNN [37], SSD [28], R-FCN [8], soft-NMS [4], DP-FCN [32] and D-F-RCNN/D-R-FCN [9].

We follow the standard settings as in [37, 8, 4, 9] and report mean average precision (mAP) scores using IoU thresholds at 0.5 and 0.7. For the first experiment, while training from VOC 2007 `trainval`, we use a learning rate of 10^{-3} for the first 40k iterations, then decrease it to 10^{-4} for the remaining 20k iterations with

Methods	mAP@0.5 / @0.7(%)	Methods	mAP@0.5 / @0.7(%)
Faster R-CNN [37]	78.1 / 62.1	SSD [28]	76.8 / N/A
DP-FCN [32]	78.1 / N/A	ION [3]	79.4 / N/A
LocNet [15]	78.4 / N/A	Deformable ConvNet [9]	78.6 / 63.3
Deformable ROI Pooling [9]	78.3 / 66.6	D-F-RCNN [9]	79.3 / 66.9
Ours	***82.0*** / ***67.0***	Ours[§]	**83.1** / **67.9**

Table 4. Detection results on PASCAL VOC using ResNet-101 [20] as backbone acchitecture. Training data: union set of VOC 2007 and 2012 `trainval`. Ours[§] denotes applying the soft-NMS [4] in the test stage.

Methods	FRCN [37]	YOLO9000 [36]	FRCN OHEM	DSSD [14]	SSD* [28]
mAP@0.5(%)	73.8	73.4	76.3	76.3	78.5
Methods	ION [3]	R-FCN [8]	DP-FCN [32]	Ours	Ours[§]
mAP@0.5(%)	76.4	77.6	79.5	***80.4***	**81.2**

Table 5. Detection results on VOC2012 `test` set using training data "07++12": 2007 `trainvaltest` and 2012 `trainval`. SSD* denotes the new data augmentation. Ours[§] denotes applying the soft-NMS [4] in the test stage.

a single GPU. Next, due to more training data, an increase in the number of iterations is needed on the union of VOC 2007 and VOC 2012 `trainval`. We perform the same training process as described in Section 4.1. Moreover, we use 300 RoIs at test stage from a single-scale image testing and set the shorter side of the image to be 600. For a fair comparison, we do not deploy the multi-scale training/testing or online hard example mining(OHEM) [39], although it is shown in [4, 9] that such enhancements could enhance the performance.

The results on VOC2007 `test` using VGG16 [40] backbone are shown in Table 3. We first compare with a traditional regionlet method [45] and several state-of-the-art object detectors [37, 28, 4] when training using small size dataset (VOC 2007 `trainval`). Next, we evaluate our method as we increase the training dataset (union set of VOC 2007 and 2012 `trainval`). With the power of deep CNNs, the deep regionlet approach significantly improves the detection performance over the traditional regionlet method [45]. We also observe that more data always helps. Moreover, it is encouraging that soft-NMS [4] is only applied in the test stage without modification in the training stage, which could directly improve over [37] by 1.1%. In summary, our method consistently outperform all the compared methods and the performance could be further improved if we replace NMS with soft-NMS [4]

Next, we change the network backbone from VGG16 [40] to ResNet-101 [20] and present corresponding results in Table 4. In addition, we also compare with D-F-RCNN/D-R-FCN [9] and DP-FCN [32].

First, compared to the performance in Table 3 using VGG16 [40] network, the mAP can be significantly increased by using deeper networks like ResNet-101 [20]. Second, comparing with DP-FCN [32] and Deformable ROI Pooling in [9][3], we

[3] [9] reported best result using OHEM, We only compare the results reported in [9] without deploying OHEM.

Methods	Training Data	mmAP 0.5:0.95	mAP @0.5	mAP small	mAP medium	mAP large
Faster R-CNN [37]	trainval	24.4	45.7	7.9	26.6	37.2
SSD*[28]	trainval	31.2	50.4	10.2	34.5	49.8
DSSD [14]	trainval	33.2	53.5	13.0	35.4	51.1
R-FCN [8]	trainval	30.8	52.6	11.8	33.9	44.8
D-F-RCNN [9]	trainval	33.1	50.3	11.6	34.9	51.2
D-R-FCN [9]	trainval	34.5	55.0	14.0	37.7	50.3
Mask R-CNN [18]	trainval	38.2	60.3	20.1	41.1	50.2
RetinaNet500 [26]	trainval	34.4	53.1	14.7	38.5	49.1
Ours	trainval	**39.3**	59.8	**21.7**	**43.7**	50.9

Table 6. Object detection results on MS COCO 2017 `test-dev` using ResNet-101 backbone. Training data: 2017 `train` and `val` set. SSD* denotes the new data augmentation.

outperform these two methods by **3.9**% and **2.7**% respectively. This provides the empirical support that our deep regionlet learning method could be treated as a *generalization* of Deformable RoI Pooling in [9, 32], as discussed in Section 3.5. In addition, the results demonstrate that selecting *non-rectangular* regions from our method provides more capabilities including *scaling*, *shifting* and *rotation* to learn the feature representations. In summary, our method achieves state-of-the-art performance on the object detection task when using ResNet-101 as backbone network.

Results evaluated on VOC2012 `test` are shown in Table 5. We follow the same settings as in [8, 37, 14, 28, 32] and train our model using VOC"07++12": VOC 2007 `trainvaltest` and 2012 `trainval` set. It can be seen that our method outperform all the competing methods. In particular, we outperform DP-FCN [32], which further proves the generalization of our method over [32].

4.3 Experiments on MS COCO

In this section, we evaluate the proposed deep regionlet approach on the MS COCO [27] dataset and compare with other state-of-the-art object detectors: Faster R-CNN [37], SSD [28], R-FCN [8], D-F-RCNN/D-R-FCN [9], Mask R-CNN [18], RetinaNet [26].

We adopt ResNet-101 as the backbone architecture of all the methods for a fair comparison. Following the settings in [18, 9, 26, 8], we set the shorter edge of the image to 800 pixels. Training is performed for 280k iterations with an effective mini-batch size 8 on 8 GPUs. We first train the model with a learning rate of 10^{-3} for the first 160k iterations, followed by learning rates of 10^{-4} and 10^{-5} subsequent for another 80k iterations and the last 40k iterations respectively. Five scales and three aspect ratios are deployed as anchors. We report results using either the released models or the code from the original authors. It is noted that we only deploy single-scale image training without the iterative bounding box average, although these enhancements could further boost performance (mmAP).

Table 6 shows the results on 2017 `test-dev` set, which contains 20,288 images. Compared with the baseline methods Faster R-CNN [37], R-FCN [8] and SSD [28], both D-F-RCNN/D-R-FCN [9] and our method provides significant improvements

over [37, 8, 28] (+3.7% and +8.5%). Moreover, it can be seen that the proposed method outperforms D-F-RCNN/D-R-FCN [9] by a wide margin(∼**4**%). This observation further supports that our deep regionlet learning module could be treated as a *generalization* of Deformable RoI Pooling in [9, 32]. It is also noted that although most recent state-of-the-art object detectors such as Mask R-CNN [18] utilize multi-task training with segmentation labels, we still outperform Mask R-CNN [18] by 1.1%. In addition, the focal loss in [26], which overcomes the obstacle caused by the imbalance of positive/nagetive samples, is complimentary to our method. We believe it can be integrated into our method to further boost performance. In summary, compared with Mask R-CNN [18] and RetinaNet[4] [26], our method achieves competitive performance over state-of-the-art on MS COCO when using ResNet-101 as a backbone network.

5 Conclusion

In this paper, we present a novel deep regionlet-based approach for object detection. The proposed RSN can select *non-rectangular* regions within the detection bounding box, and hence an object with rigid shape and deformable parts can be better modeled. We also design the deep regionlet learning module so that both the selected regions and the regionlets can be learned simultaneously. Moreover, the proposed system can be trained in a fully end-to-end manner without additional efforts. Finally, we extensively evaluate our approach on two detection benchmarks and experimental results show competitive performance over state-of-the-art.

6 Acknowledgement

This research is based upon work supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/Interior Business Center (DOI/IBC) contract number D17PC00345. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes not withstanding any copyright annotation theron. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied of IARPA, DOI/IBC or the U.S. Government.

We thank the reviewers for their valuable comments and suggestions.

References

1. Ahonen, T., Hadid, A., Pietikäinen, M.: Face recognition with local binary patterns. In: European Conference on Computer Vision (ECCV). pp. 469–481 (2004)

[4] [26] reported best result using multi-scale training for 1.5× longer iterations, we only compare the results without scale jitter during training. In addition, we only compare the results in [18] using ResNet-101 backbone for fair comparison.

2. Bansal, A., Sikka, K., Sharma, G., Chellappa, R., Divakaran, A.: Zero-shot object detection. CoRR **abs/1804.04340** (2018)
3. Bell, S., Zitnick, C.L., Bala, K., Girshick, R.B.: Inside-Outside Net: Detecting objects in context with skip pooling and recurrent neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2874–2883 (2016)
4. Bodla, N., Singh, B., Chellappa, R., Davis, L.S.: Soft-NMS - improving object detection with one line of code. In: IEEE International Conference on Computer Vision (ICCV). pp. 5562–5570 (2017)
5. Bodla, N., Zheng, J., Xu, H., Chen, J., Castillo, C.D., Chellappa, R.: Deep heterogeneous feature fusion for template-based face recognition. In: IEEE Winter Conference on Applications of Computer Vision (WACV). pp. 586–595 (2017)
6. Cai, Z., Vasconcelos, N.: Cascade r-cnn: Delving into high quality object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
7. Cheng, B., Wei, Y., Shi, H., Feris, R.S., Xiong, J., Huang, T.S.: Revisiting RCNN: on awakening the classification power of faster RCNN. CoRR **abs/1803.06799** (2018)
8. Dai, J., Li, Y., He, K., Sun, J.: R-FCN: object detection via region-based fully convolutional networks. In: Advances in Neural Information Processing Systems (NIPS). pp. 379–387 (2016)
9. Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., Wei, Y.: Deformable convolutional networks. In: IEEE International Conference on Computer Vision (ICCV). pp. 764–773 (2017)
10. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 886–893 (2005)
11. Everingham, M., Gool, L.J.V., Williams, C.K.I., Winn, J.M., Zisserman, A.: The pascal visual object classes (VOC) challenge. International Journal of Computer Vision **88**(2), 303–338 (2010)
12. Felzenszwalb, P.F., Girshick, R.B., McAllester, D.A.: Cascade object detection with deformable part models. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2241–2248 (2010)
13. Felzenszwalb, P.F., Girshick, R.B., McAllester, D.A., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE Transactions on Pattern Analysis and Machine Intelligence **32**(9), 1627–1645 (2010)
14. Fu, C., Liu, W., Ranga, A., Tyagi, A., Berg, A.C.: DSSD : Deconvolutional single shot detector. CoRR **abs/1701.06659** (2017)
15. Gidaris, S., Komodakis, N.: LocNet: Improving localization accuracy for object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 789–798 (2016)
16. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014)
17. Girshick, R.B.: Fast R-CNN. In: IEEE International Conference on Computer Vision (ICCV). pp. 1440–1448 (2015)
18. He, K., Gkioxari, G., Dollár, P., Girshick, R.B.: Mask R-CNN. In: IEEE International Conference on Computer Vision (ICCV). pp. 2980–2988 (2017)
19. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. In: European Conference on Computer Vision (ECCV). pp. 346–361 (2014)

20. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016)
21. Hu, H., Gu, J., Zhang, Z., Dai, J., Wei, Y.: Relation networks for object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
22. Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., Murphy, K.: Speed/accuracy trade-offs for modern convolutional object detectors. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3296–3297 (2017)
23. Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K.: Spatial transformer networks. In: Advances in Neural Information Processing Systems (NIPS). pp. 2017–2025 (2015)
24. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems (NIPS), pp. 1097–1105 (2012)
25. Lin, T., Dollár, P., Girshick, R.B., He, K., Hariharan, B., Belongie, S.J.: Feature pyramid networks for object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 936–944 (2017)
26. Lin, T., Goyal, P., Girshick, R.B., He, K., Dollár, P.: Focal loss for dense object detection. In: IEEE International Conference on Computer Vision (ICCV). pp. 2999–3007 (2017)
27. Lin, T., Maire, M., Belongie, S.J., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: common objects in context. In: European Conference on Computer Vision (ECCV). pp. 740–755 (2014)
28. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S.E., Fu, C., Berg, A.C.: SSD: single shot multibox detector. In: European Conference on Computer Vision (ECCV). pp. 21–37 (2016)
29. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3431–3440 (2015)
30. Lowe, D.G.: Object recognition from local scale-invariant features. In: IEEE International Conference on Computer Vision (ICCV). pp. 1150–1157 (1999)
31. Mallat, S.: A Wavelet Tour of Signal Processing, 2nd Edition. Academic Press (1999)
32. Mordan, T., Thome, N., Cord, M., Henaff, G.: Deformable part-based fully convolutional network for object detection. In: Proceedings of the British Machine Vision Conference (BMVC) (2017)
33. Ouyang, W., Zeng, X., Wang, X., Qiu, S., Luo, P., Tian, Y., Li, H., Yang, S., Wang, Z., Li, H., Wang, K., Yan, J., Loy, C.C., Tang, X.: DeepID-Net: Object detection with deformable part based convolutional neural networks. IEEE Transactions on Pattern Analysis and Machine Intelligence **39**(7), 1320–1334 (2017)
34. Ranjan, R., Bansal, A., Xu, H., Sankaranarayanan, S., Chen, J., Castillo, C.D., Chellappa, R.: Crystal loss and quality pooling for unconstrained face verification and recognition. CoRR **abs/1804.01159** (2018)
35. Redmon, J., Divvala, S.K., Girshick, R.B., Farhadi, A.: You only look once: Unified, real-time object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 779–788 (2016)
36. Redmon, J., Farhadi, A.: YOLO9000: better, faster, stronger. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6517–6525 (2017)

37. Ren, S., He, K., Girshick, R.B., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence **39**(6), 1137–1149 (2017)
38. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., Lecun, Y.: Overfeat: Integrated recognition, localization and detection using convolutional networks. In: International Conference on Learning Representations (ICLR) (2014)
39. Shrivastava, A., Gupta, A., Girshick, R.B.: Training region-based object detectors with online hard example mining. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 761–769 (2016)
40. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR **abs/1409.1556** (2014)
41. Singh, B., Davis, L.S.: An analysis of scale invariance in object detection - SNIP. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
42. Viola, P.A., Jones, M.J.: Rapid object detection using a boosted cascade of simple features. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 511–518 (2001)
43. Wang, H., Wang, Q., Gao, M., Li, P., Zuo, W.: Multi-scale location-aware kernel representation for object detection. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
44. Wang, X., Yang, M., Zhu, S., Lin, Y.: Regionlets for generic object detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **37**(10), 2071–2084 (2015)
45. Wang, X., Yang, M., Zhu, S., Lin, Y.: Regionlets for generic object detection. In: IEEE International Conference on Computer Vision (ICCV). pp. 17–24 (2013)
46. Wu, Z., Bodla, N., Singh, B., Najibi, M., Chellappa, R., Davis, L.S.: Soft sampling for robust object detection. CoRR **abs/1806.06986** (2018)
47. Xu, H., Zheng, J., Alavi, A., Chellappa, R.: Cross-domain visual recognition via domain adaptive dictionary learning. CoRR **abs/1804.04687** (2018)
48. Zhang, S., Wen, L., Bian, X., Lei, Z., Li, S.Z.: Single-shot refinement neural network for object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
49. Zhang, S., Zhu, X., Lei, Z., Shi, H., Wang, X., Li, S.Z.: S^3fd: Single shot scale-invariant face detector. In: IEEE International Conference on Computer Vision (ICCV). pp. 192–201 (2017)
50. Zhang, Z., Qiao, S., Xie, C., Shen, W., Wang, B., Yuille, A.L.: Single-shot object detection with enriched semantics. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
51. Zhao, X., Liang, S., Wei, Y.: Pseudo mask augmented object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
52. Zhou, P., Ni, B., Geng, C., Hu, J., Xu, Y.: Scale-transferrable object detection. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)