Deep Regionlets for Object Detection

Hongyu Xu[1,2*], Xutao Lv[2], Xiaoyu Wang[3], Zhou Ren[2] and Rama Chellappa[1]
[1]Department of Electrical and Computer Engineering and the Center for Automation Research, UMIACS
University of Maryland, College Park, MD, USA
[2]Snap Inc. [3]Intellifusion
[1]hyxu@umd.edu [2]lvxutao@gmail.com [3]fanghuaxue@gmail.com
[2]zhou.ren@snap.com [1]rama@umiacs.umd.edu

Abstract

A key challenge in generic object detection is being to handle large variations in object scale, poses, viewpoints, especially part deformations when determining the location for specified object categories. Recent advances in deep neural networks have achieved promising results for object detection by extending the traditional detection methodologies using the convolutional neural network architectures. In this paper, we make an attempt to incorporate another traditional detection schema, Regionlet into an end-to-end trained deep learning framework, and perform ablation studies on its behavior on multiple object detection datasets. More specifically, we propose a "region selection network" and a "gating network". The region selection network serves as a guidance on where to select regions to learn the features from. Additionally, the gating network serves as a local feature selection module to select and transform feature maps to be suitable for detection task. It acts as soft Regionlet selection and pooling. The proposed network is trained end-to-end without additional efforts. Extensive experiments and analysis on the PASCAL VOC dataset and Microsoft COCO dataset show that the proposed framework achieves comparable state-of-the-art results.

1. Introduction

Generic object detection has been extensively studied in computer vision community over the decades [2, 33, 10, 11, 28, 3, 21, 32, 31, 5, 8] due to its great value for both academic research explorations as well as commercial applications. Given an image of interest, the goal of generic 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 specified object categories.

Numerous methods have been proposed based on traditional hand-crafted features (*i.e.* HOG [5], LBP [1], SIFT [24]). Such approaches usually involve an exhaustive search for the possible locations, scales and aspect ratios of the object, by using a sliding windows search. However, regionlet-based detection [32] has gained a lot of attention as it provides the flexibility to deal with different scaling and aspect ratios without performing an exhaustive search. It operates by directly extracting features from regionlets in several selected region within the arbitrary detection bounding box and performing (max) pooling among the regionlets, which lead to improved learning of robust feature representation of object.

Putting this work in context, recently, deep learning has achieved significant success on tasks such as image classification [17, 14], semantic segmentation [23] and object detection [10] using the deep convolutional neural networks (CNNs) architecture. As such, many deep learning-based approaches have achieved very promising performance on detection by introducing CNNs to traditional methods [4, 26].

In this paper, we make an attempt to incorporate traditional Regionlet method into an end-to-end trained deep learning framework. Despite the merits of regionlet-based detection, several drawbacks arise when directly integrated into the deep learning networks. First, Wang *et al*. [32] learns cascade object classifiers after hand-crafted feature extraction in each regionlet, where end-to-end learning is not feasible. Second, regions in regionlet-based detection have to be rectangular, which does not effectively model the deformation in the object with rigid shapes. Moreover, both regions and regionlets are fixed after training is completed.

To this end, we propose a novel deep regionlet approach to introduce deep learning framework to the traditional regionlet method [32]. The overall design of the proposed deep regionlet-based detection system is illustrated in Figure 1. In order to address limitations of the traditional regionlet approach, we propose a Region Selection Network (RSN) to

*Work started during an internship at Snap Research

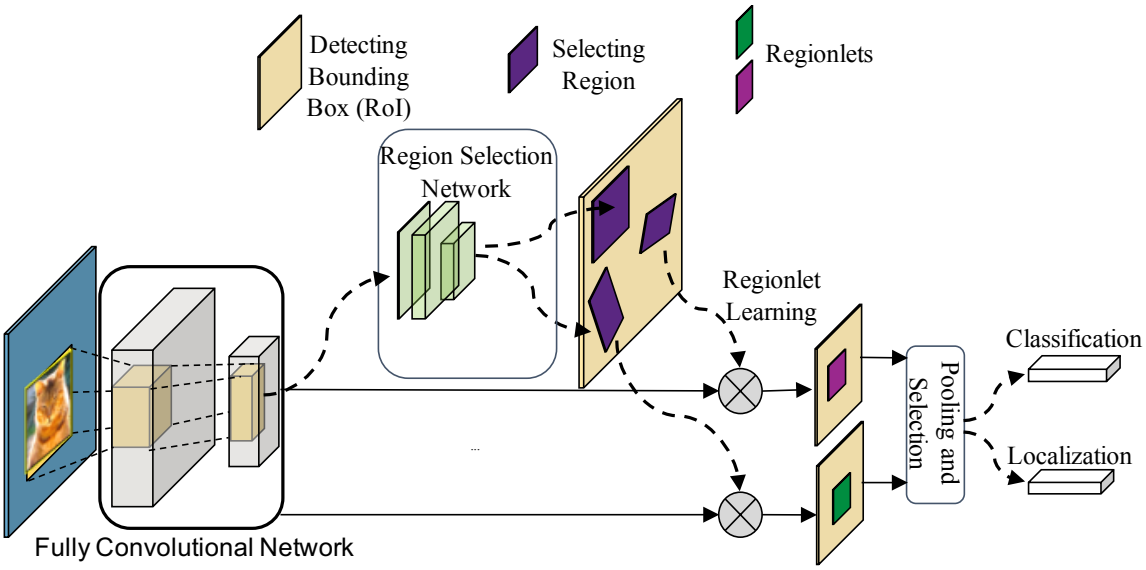

Figure 1: Overview of the proposed deep regionlet approach

perform *non-rectangle* region selection within the detection bounding box. We design a deep regionlet learning module to learn the regionlets through a gating network. By using the proposed gating network, which is a soft regionlet selector, we can generate the final feature representation to be more suitable for detection. The whole system is designed with end-to-end training using only the input images and ground truth bounding boxes.

We conduct a detailed analysis of our approach to understand its merits and properties. Extensive experiments on two detection benchmark datasets, PASCAL VOC [6] and Microsoft COCO [19] show that the proposed deep regionlet approach outperforms several deep-learning based detectors [28, 3, 22, 9] and achieves comparable results with state-of-the-art algorithms [4, 26]. To summarize, we make the following contributions:

- We propose a novel deep regionlet approach for object detection. Our work integrates the traditional regionlet method into the deep learning framework. The system could be trained in a fully end-to-end manner.

- We design a region selection network (RSN), which is able to select non-rectangle regions within the detection bounding box to provide more flexibility in modeling the objects with rigid shape 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 selection to let the network focus on features that are more suited to the detection task.

2. Related Work

Object detection has gained a lot of popularity over the decades. Many approaches have been proposed to solve this problem, including both traditional ones [8, 32, 31] and deep learning-based approaches [11, 28, 22, 27, 3, 10, 13, 4, 26, 9]. Traditional approaches mainly used hand-crafted features (*i.e.* HOG [5], LBP [1]) to train the object detectors using the sliding window paradigm. One of the earliest works [31] used boosted cascaded detectors for face detection, which led to its wide adoption. Deformable Part Model based detection (DPM) [7] further extended the cascaded detectors to more general object categories and proposed the concept of deformable part models to handle the object deformation. Due the rapid development of deep learning techniques [17, 14, 30], the deep learning-based detectors have become dominant object detectors.

Deep learning-based detectors could be further categorized into two classes, single-stage detectors and two-stage detectors, based on whether the detectors have proposal-driven mechanism or not. The single-stage detectors [29, 27, 22, 9, 20] 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 were lower comparing to two-stage detectors.

Two-stage detectors [11, 28, 3, 4, 26] involves two steps. It first generated a sparse set of candidate proposals of detection bounding boxes by 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 boxes regression to predict the objects' class and its corresponding locations.

The two-stage detectors consistently achieve higher accuracy than single-stage detectors and numerous extensions have been proposed [11, 28, 3, 4, 26, 12]. Our method follows the two-stage detector architecture by taking advantage of the region proposal network without the need of dense sampling of object location, 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 proposed deep regionlet approach with end-to-end training. 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 at arbitrary resolution (*i.e.* size and aspect ratio). Wang *at al.* [32] first introduced the concept of regionlet illustrated in Figure 2. In Figure 2, the yellow box is a detection bounding box. R is a *rectangle* 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 the regionlets and regions. However, in the traditional approach, the initialization of regionlets possess randomness and both regions and regionlets are fixed after the training. Moreover, it is based on hand-crafted feature (*i.e.* HOG [5] or LBP descriptors [1]) in each regionlet respectively and hence, it is not end-to-end trainable. We propose the following deep regionlet-based approach to address such limitations.

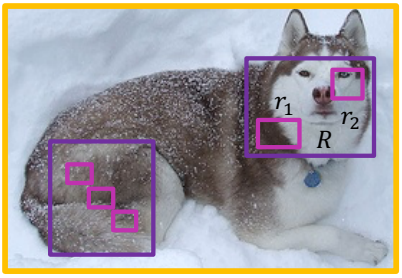

Figure 2: Illustration of relationships among the detection bounding box, a feature extraction region and regionlets. R is a feature extraction region shown as purple rectangle with filled dots. Inside R, two small sub-regions denoted as r_1 and r_2 are the *regionlets*.

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 (CNN). At some layer, the network bifurcates into two branches. One branch, named regional proposal network, generates a set of candidate bounding boxes[1] while the other branch performs classification and regression by pooling the convolutional features inside the proposed bounding box generated by the region proposal network [28, 3]. Taking advantage of the detection network, we introduce the overall design of the proposed deep regionlet-based detection system, as illustrated in Figure 1.

The general architecture consists of a Region Selection Network (RSN) and a regionlet learning module. In particular, our region selection network is used to predict the transformation parameters to select regions in a candidate detecting bounding box, where regionlets are further learned within each selected region. 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 region selection network as well as the regionlet learning module could be simultaneously learned over each selected region within one candidate detecting bounding box.

3.3. Region Selection Network

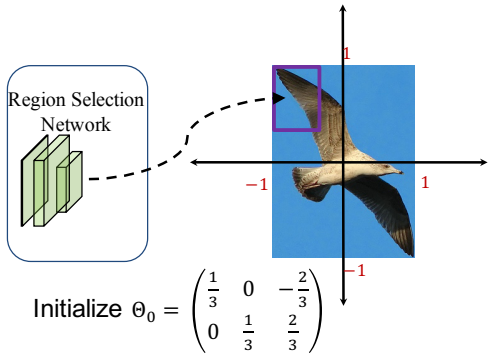

Figure 3: Example of initialization of one affine transformation parameter. Normalized affine transformation parameters $\Theta_0 = [\frac{1}{3}, 0, -\frac{2}{3}; 0, \frac{1}{3}, \frac{2}{3}]$ ($\theta_i \in [-1, 1]$) selects the top-left region in the 3×3 evenly divided detection bounding box, shown as the purple rectangle.

We design the region selection network (RSN) to have the following properties: 1) End-to-end trainable; 2) Simple structure; 3) Generate regions with arbitrary shape. 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 the object with arbitrary shape and deformable parts.

Specifically, we design the RSN using a small neural network with three fully connected layers. The first two fully

[1] [28, 3, 11] also called the detection bounding box as Region of Interest (ROI)

connected layer has output size of 256, with ReLU activation. The last fully connected layer has 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 region proposal network 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 $\Theta = [\theta_1, \theta_2, \theta_3; \theta_4, \theta_5, \theta_6]^T$ ($\theta_i \in [-1, 1]$) to evaluate one selected region within one candidate detecting window at different sizes and aspect ratios without scaling images into multiple resolutions or using multiples components for enumerating possible aspect ratios, like anchors [28, 22, 9].

Initialization of Region Selection Network: Taking advantage of the *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 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. Regionlet Learning

After regions have been selected by the region selection network, 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 box. Therefore, both the selected regions and regionlet learning should be able to be trained by CNN networks. Moreover, we would like the regionlet extracted from the selected regions to better represent objects with rigid shape and deformable parts.

Inspired by the spatial transform network [15], any parameterizable transformation including translation, scaling, rotation, affine or even projective can be learned by 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 selecting region defined by one affine transformation Θ to better match the shapes of objects. This is done with a selecting region R from region selection network, transformation parameters $\Theta = [\theta_1, \theta_2, \theta_3; \theta_4, \theta_5, \theta_6]^T$ 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 selecting 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 regionlet learning module below:

Let (x_p^s, y_p^s) define the spatial location in feature map Z_i. 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

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

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 the feature maps and the region selection network. In this layer's `backwards` function, we have partial derivative of the loss function with respect to both feature map variable U_{mn}^c and affine transform parameter $\Theta = [\theta_1, \theta_2, \theta_3; \theta_4, \theta_5, \theta_6]^T$. Motivate by [15], the partial derivative of the loss function with respect to the feature map is:

$$\frac{\partial V(x_p^s, y_p^s, c|\Theta, R)}{\partial U_{nm}^c} = \sum_n^H \sum_m^M \max(0, 1 - |x_p^s - m|) \times \max(0, 1 - |y_p^s - n|) \tag{2}$$

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

$$\frac{\partial V(x_p^s, y_p^s, c|\Theta, R)}{\partial \theta_1} = \frac{\partial V(x_p^s, y_p^s, c|\Theta, R)}{\partial x_p^s} \frac{\partial x_p^s}{\partial \theta_1}$$
$$= x_p^t \sum_n^H \sum_m^M U_{nm}^c \max(0, 1 - |y_p^s - n|) \times \begin{cases} 0 \text{ if } |m - x_p^s| \geq 1 \\ 1 \text{ if } m \geq x_p^s \\ -1 \text{ if } m \leq x_p^s \end{cases} \tag{3}$$

It is worth noting that the height and width normalized coordinates are used in (x_p^t, y_p^t) (*i.e.* $-1 \leq x_p^t, y_p^t \leq 1$), so that it can to be scaled with respect to w and h with start position (w_0, h_0).v

Gating Network: The gating network, which serves as a soft regionlet selection module, is used to assgin regionlets with different weight and generate regionlet feature representation. We design a simple gating network based on the `sigmoid` activation function for its non-negative property.

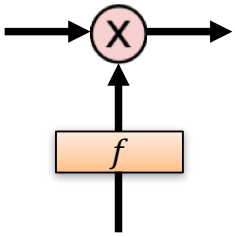

Figure 4: Design of the gating network. f denotes the non-negative gate function (*i.e.* sigmoid)

Given the output feature maps $V(x_p^s, y_p^s, c|\Theta, R)$ described above, we use a fully connected layer to generate the same number of output as feature maps $V(x_p^s, y_p^s, c|\Theta, R)$, 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 $V(x_p^s, y_p^s, c|\Theta, R)$ and their corresponding weights.

Regionlet Pool Construction Object deformation may occur at different scales. For instance, in person detection, deformation could be caused by different body parts. Same number of regionlets (size $H \times W$) learned from small selecting 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 $V(x_p^s, y_p^s, c|\Theta, R)$ 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 selecting regions are able to represent such regions in the same efficient way, thus to handle the object deformations at different scales.

3.5. In Context of Related Work

Our deep regionlet approach is related to some recent works in different aspects. In this section, we discuss both similarities and differences in details.

Spatial Transform Networks (STN) [15] It is the first work to propose the spatial transformer module to provide spatial transformation capabilities into a deep neural network. It only learns *one global parametric transformation* (scaling, rotations as well as affine transformation). Such learning is known difficult to apply on the semi-dense vision tasks (*e.g.*, object detection) and the transformation is on the entire feature map, which means the transformation is manipulated identically across all the regions in the feature map.

Our region selection network learns a set of affine transformation and could be considered as simulating multiple cases of the localization network in [15]. However, our regionlet learning is different from image sampling [15] method as it adopts a region-based parameter transformation and feature wrapping. By learning the transformation locally in the detection bounding box, our method provide the flexibility of learning the compact, efficient feature representation of object with rigid shape and deformable parts.

Deformable Part Model (DPM) [7] and **its deep learning extensions** [26, 4]. Deformable Part Model (DPM) [7] explicitly models spatial deformations of object parts via latent variables. A root filter is learned to model the global appearance of the objects, while the part filters are designed to extract to describe the local parts in the objects. However, DPM is a shallow model and the training process involves heuristic choices to select components and part sizes, making end-to-end training inefficient.

Both works [4, 26] extend the DPM with end-to-end training in deep CNNs. Motivated by DPM [8] to allow parts to slightly move around their reference position (partition of the initial regions), they share the similar idea of learning local shifts [2] to model the local element and pool the features at their corresponding locations after the shift. While [4, 26] show promising improvement over other deep learning-based object detectors [11, 28], it still lacks the flexibility of modeling the non-rectangle objects with sharp shapes and deformable parts.

It is noticeable that the regionlet learning on the selecting region is a generalization of [4, 26]. First, we generalize the selecting region to be non-rectangle by learning the affine transformation parameters. Such non-rectangle regions could provide the capabilities of *scaling*, *shifting* and *rotation* around the original reference region. If we only enforce the region selection network to learn the shift, our regionlet learning mechanism would degenerate to similar deformable RoI pooling as in [4, 26]

Spatial-based RoI pooling [18, 16, 13]. Traditional spatial pyramid pooling [18] performs pooling over hand crafted regions at different scales. With the help of deep CNNs, [13] proposes to use pyramid pooling in deep learning-based object detection. However, as the pooling regions over image pyramid still need to be carefully designed and chosen to learn the spatial layout of the pooling regions, therefore the end-to-end training is not well facilitated. Our deep regionlet learning approach learns pooling regions end-to-end in deep CNNs. Moreover, the region selection step for learning regionlets accommodates different sizes of the regions. Hence, we are able to handle the object deformations at different scales without generating the feature pyramid.

4. Experiments

In this section, we present experiment results of the proposed approach on two challenging benchmark datasets, PASCAL VOC [6] and MS-COCO [19]. We first introduce the datasets and experimental settings. It is then followed by discussion of experimental results.

[2] [4] uses term offset while [26] uses term displacement

4.1. Datasets and Settings

4.1.1 Datasets

There are in total 20 categories of objects in PASCAL VOC [6] dataset, which including rigid objects such as cars and deformable objects like cats. For PASCAL VOC dataset, we follow the same settings used in [28, 2, 3, 11]. More specifically, in order to draw complete comparisons, we train our deep model on two different training datasets: (1) VOC2007 `trainval` and (2) union of VOC2007 `trainval` and VOC2012 `trainval`. Evaluation is on VOC 2007 `test` partition, which contains a total 4952 images. In addition, we report the results on the VOC2007 `test` split for ablation study.

MS-COCO [19] is a widely used challenging detection dataset, which contains 80 object categories. For training, following the official settings in COCO website[3], we use the COCO 2017 `train` and 2017 `val` split (union of 135k images from `train` split and 5k images from `val` split). For testing, we report COCO average precision (AP) on the 2017 `test-dev` split, which has no public labels available and requires evaluation from the MS-COCO server[4].

4.1.2 Settings

Base Network: We run our experiments based on both VGG-16 [30] and ResNet-101 [14] to demonstrate the generalization of our approach regardless of which network backbone we use. The public VGG-16 [30] model has 13 convolutional layers and 3 fully-connected layers. For the ResNet-101 [14], following the suggestion settings in [3, 4], we change 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. Following the *á trous* algorithm [23, 25], we also increase the dilation from 1 to 2 to fix the "holes" if the convolutional layers in conv5 stage have kernel size larger than 1.

Training and Inference: For all the experiments evaluated on the VOC 2007 `test`, we set the shorter side of image to be 600 pixels, as suggested in [11, 28, 3]. For the first experiment which training set is from VOC2007 `trainval`, we start learning rate at 10^{-3} for the first 40k iterations, then we decrease it to 10^{-4} for the rest 20k iterations with single GPU. Next, due to more training data, increasing the number of iteration is needed on the union of VOC 2007 `trainval` and 2012 `trainval`. The training is performed for 30k iterations with effective mini-batch size 8 on 8 GPUs, where the learning rate is set as 10^{-3} for the first 20k iterations and 10^{-4} for the rest 10k iterations. We take the VGG16 or ResNet-101 model from Image-Net as the pre-trained model for our approach. For COCO, The training is performed for total 240k iterations with effective mini-batch size 8. the learning rate is set as 10^{-3} for the first $\frac{2}{3}$ and 10^{-4} for the rest $\frac{1}{3}$ iterations.

Methods	training data	mAP@0.5(%)
Regionlet [32]	07	41.7
Faster R-CNN [28]	07	70.0
R-FCN [3]	07	69.6
SSD 512 [22]	07	71.6
Soft-NMS [2]	07	71.1
Ours	07	**72.1**
Ours with soft-NMS	07	**73.4**
Faster-RCNN [28]	07 + 12	73.2
R-FCN [3]	07 + 12	76.6
SSD 512 [22]	07 + 12	76.8
Ours	07 + 12	**77.3**
Ours with soft-NMS	07 + 12	**78.4**

Table 1: Detection results on PASCAL VOC 2007 test set using VGG16 as backbone architecture. Training data: "07": VOC 2007 `trainval`, "07 + 12": union set of VOC 2007 `trainval` and VOC 2012 `trainval`. With soft-NMS denotes we apply the soft-NMS in the test stage.

Methods	mAP@0.5 / @0.7(%)
Faster R-CNN [28]	78.1 / 62.1
SSD [22]	76.8 / N/A
DP-FCN [26]	78.1 / N/A
Deformable ConvNet [4]	78.6 / 63.3
Deformale ROI Pooling [4]	78.3 / 66.6
Deformable ConvNet + ROI Pooling [4]	79.3 / ***66.9***
Ours	***80.7*** / 66.5
Ours with soft-NMS	**81.6 / 67.1**

Table 2: Detection results on PASCAL VOC 2007 test set using ResNet-101 [14] as backbone acchitecture. Training data: union set of VOC 2007 `trainval` and VOC 2012 `trainval`. With soft-NMS denotes we apply the soft-NMS in the test stage.

4.2. Experiments on PASCAL VOC

In this section, we compare our results with traditional regionlet method [32] and several state-of-the-art deep learning-based object detectors as follows: Faster R-CNN [28], SSD [22], R-FCN [3], soft-NMS [2]. All deep learning-based methods are trained on the same VGG-16 [30, 11] network and ResNet-101 [14] network.

The mean average precision (mAP) is used for evaluation on the VOC 2007 dataset. We follow the standard settings as in [28, 3, 2, 4] on VOC 2007 `test` and report mAP scores using IoU thresholds at 0.5. Although it is shown in [2, 4] that multi-scale testing could lead to the performance boost in the mAP, in order to draw fair comparison, we use 300

[3] http://cocodataset.org/#detections-challenge2017

[4] The updated settings (2017) are different from the previous settings (2016, 2015) in [2, 21, 4, 3, 21], as it includes different train/val sets.

RoIs at test stage from single-scale image testing with setting the image shorter side to be 600.

In Table 1, we first compare with traditional regionlet method [32] and several state-of-the-art object detectors [28, 22, 2] when training from small size dataset (VOC2007 `trainval`). Next, we evaluate our method as we increase the training dataset from VOC2007 `trainval` to union set of VOC 2007 and 2012 `trainval`. With the power of deep CNNs, the deep regionlet approach has significantly improved the detection performance over the traditional regionlet method [32]. We also observe that more data always helps improves. Moreover, it is encouraging that soft-NMS [2] is only applied in the test stage without modification in the training stage, which could directly improve over [28] by 1.1%. In all, our method consistently outperform all the compared methods from VOC2007 `trainval` to the VOC 2007 + 2012 `trainval` and the performance could be further gained if we apply soft-NMS.

Next, we change the network backbone from VGG16 [30] to ResNet-101 [14] and present corresponding results in Table 2. In addition, we also compare with Deformable Convolutional Network [4] and DP-FCN [26][5]. First, compared to the performance in Table 1 using VGG16 [30] network, the mAP can be significantly increased by using deeper networks like ResNet-101 [14]. Second, comparing with DP-FCN [26] and Deformable ROI Pooling [4], we outperform these two methods by 2.5% and 1.3% respectively. As discussed in Section 3.5, our deep regionlet learning method could be treated as a generalization of [4, 26]. The results demonstrate that selecting non-rectangle regions from our method provide more capabilities including *scaling*, *shifting* and *rotation* to learn the feature representations. In all, our method achieves state-of-the-art performance on object detection task when using ResNet-101 as a backbone network.

4.3. Ablation Study

We perform series of ablation experiments on PASCAL VOC [6] to understand the behavior of the proposed deep regionlet approach. We train the model on the union set of VOC 2007 `trainval` and 2012 `trainval` and evaluate on the VOC 2007 `test` set. Mean Average precision (mAP) is reported.

First we investigate our deep regionlet approach and compare it with several other baselines described below to understand each component: (1) Region selection network, (2) Deep regionlet learning and (3) Soft regionlet selection.

- Global RSN. Region selection network 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.
- Shifting-only RSN. We set the region selection network to only learn the shift by enforcing $\theta_1, \theta_2, \theta_4, \theta_5$ not change during the training process. In this way, the region selection network only selects the rectangular region with shifts to the initialized region. This setting could also be viewed as similar to the Deformable RoI Pooling in [4] and [26].
- 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 3a. First, when the Region Selection Network only selects one global region, the Region Selection Network reduces to the single localization network [15]. 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 [28, 3]. However, the performance is extremely poor. This is because no discriminative regionlets could be explicitly learned within the region. More importantly, comparing our approach and shifting-only RSN to global RSN, the results clearly demonstrate that the region selection network (RSN) is indispensable in the deep regionlet approach.

Moreover, shifting-only RSN could be viewed as similar to deformable RoI pooling in [4, 26]. These methods all learn the shift of the rectangle region with respect to its reference position, which lead to improvement over [28]. However, non-gating selection outperforms shifting-only RSN by 1.6% with selecting non-rectangle region. The improvement demonstrates that non-rectangle region selection could provide more flexibility around the original reference region, thus could better model the non-rectangle objects with sharp shapes and deformable parts. Last but not least, by using the gate function to perform soft regionlet selection, the performance could be further improved by 0.6%.

Next, in order to understand more deeply the region selection network and regionlet learning module, we present ablation studies on the following questions:

- How many regions should we learn by region selection network (RSN)?
- How many regionlets should we learn in one selected region (density is of size $H \times W$)?

How many regions should we learn by region selection network (RSN)? We investigate how the detection performance varies when different number of regions are selected by the region selection network. All the regions are initialized as described in 3.3 without any overlap between each region. Without loss of generality, we report results when selecting $4(2 \times 2)$, $9(3 \times 3)$ and $16(4 \times 4)$ regions by the region selection network in Table 3b. We observe that

[5] Note [26] reported best result using ResNeXt-101 (64x4d) [34], we only compare the results in [26] using ResNet-101 [14] backbone for fair comparison.

Methods	mAP@0.5(%)
Global RSN	9.27
Shift-only RSN [4, 26]	78.5
Non-gating	80.1
Ours	**80.7**

(a) Ablation study to understand each component in deep regionlet approach. Output size $H \times W$ is set to 4×4 for all the baselines

# of Regions \ Regionlets Density	2×2	3×3	4×4	5×5	6×6
$4(2 \times 2)$ regions	77.0	78.2	78.6	78.9	79.0
$9(3 \times 3)$ regions	78.4	79.3	79.8	80.4	80.2
$16(4 \times 4)$ regions	78.9	79.9	80.7	80.3	79.6

(b) Ablation study on PASCAL VOC when Region Selection Network (RSN) selects different number of regions and regionlets are learned at different level of density.

Table 3: Ablation study on PASCAL VOC 2007 `test`. mAP@0.5 are reported. ResNet-101 [14] is used as the backbone architecture. Training data: VOC 2007 + 2012 `trainval`. Soft-NMS [2] is not conducted in the test stage

Methods	Training Data	mAP 0.5 : 0.95	mAP @0.5	mAP small	mAP medium	mAP large
F-RCNN [28]	trainval	24.4	45.7	7.9	26.6	37.2
R-FCN [3]	trainval	30.8	52.6	11.8	33.9	44.8
Deformable F-RCNN [4]	trainval	33.1	50.3	11.6	34.9	51.2
Ours	trainval	31.7	50.4	11.2	34.5	50.6

Table 4: Object detection results on MS COCO 2017 `test-dev` using ResNet-101 [14] as backbone acchitecture. Training data: union set of 2017 `train` and 2017 `val` set.

the mean AP increases when the number of selected regions is increased from $4(2)$ to $9(3 \times 3)$ for 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 selecting region by varying H and W. Without loss of generality, we set $H = W$ throughout our experiments and varying value from 2 to 6 due to the space limitation. In Tabbe 3b, we report results when we learn 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 $25(5 \times 5)$ can result 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 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. Throughout all the experiments in the paper, we report the results from 16 selecting regions from region selection network and set output size $H \times W = 4 \times 4$.

4.4. Experiments on MS COCO

In this section, we evaluate the proposed deep regionlet approach on MS COCO [19] 2017 `test-dev` set and compare with other state-of-the-art object detectors: Faster R-CNN [28], R-FCN [3] and Deformable Convolutional Network [4]. For fair comparison, ResNet-101 [14] is used as the backbone architecture for all the comparisons[6]. We report results using either the released models or the code from original authors, since the updated 2017 settings are different from previous 2016, 2015 as it contains different `train`/`val` set[7]. We report the results in Table 4. First, both [4] and our method outperform [28, 3] by wide margin. Moreover, it can be seen that our method achieves results comparable to [4], which demonstrates the effectiveness of the proposed deep regionlet approach. Note that we only deploy single-scale image testing without the iterative boudning box average, although these tricks could further boost performance (mAP).

5. Conclusion

In this paper, we present a novel deep regionlet-based approach for object detection. The region selection network is proposed, which can select non-rectangle region within the detection bounding box, and hence the object with rigid shape and deformable parts can be better modeled. We also design the deep regionlet learning module so that both the

[6] [4] also reported results using Aligned-Inception-ResNet model, which is in unpublished work and not publicly available. We only compare the results reported in [4] using ResNet-101.

[7] MS COCO server does not accept 2016 and 2015 test-dev evaluation. As a result, we are not able to report results on 2016, 2015 test-dev set.

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 for generic object detection. The experimental results shows competitive performance over state-of-the-art.

6. Acknowledgement

This research is based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA R&D Contract No. D17PC00345. 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 the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

References

[1] T. Ahonen, A. Hadid, and M. Pietikäinen. Face recognition with local binary patterns. In *European Conference on Computer Vision (ECCV)*, pages 469–481, 2004. 1, 2, 3

[2] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis. Soft-nms – improving object detection with one line of code. In *IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 1, 6, 7, 8

[3] J. Dai, Y. Li, K. He, and J. Sun. R-FCN: object detection via region-based fully convolutional networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 379–387, 2016. 1, 2, 3, 6, 7, 8

[4] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. *CoRR*, abs/1703.06211, 2017. 1, 2, 3, 5, 6, 7, 8

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 886–893, 2005. 1, 2, 3

[6] M. Everingham, L. J. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010. 2, 5, 6, 7

[7] P. F. Felzenszwalb, R. B. Girshick, and D. A. McAllester. Cascade object detection with deformable part models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2241–2248, 2010. 2, 5

[8] P. F. Felzenszwalb, R. B. Girshick, D. A. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010. 1, 2, 5

[9] C. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg. DSSD : Deconvolutional single shot detector. *CoRR*, abs/1701.06659, 2017. 2, 4

[10] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 1, 2

[11] R. B. Girshick. Fast R-CNN. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015. 1, 2, 3, 5, 6

[12] K. He, G. Gkioxari, P. Dollar, and R. Girshick. Mask r-cnn. In *IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 3

[13] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European Conference on Computer Vision (ECCV)*, pages 346–361, 2014. 2, 5

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 1, 2, 6, 7, 8

[15] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2017–2025, 2015. 4, 5, 7

[16] Y. Jia, C. Huang, and T. Darrell. Beyond spatial pyramids: Receptive field learning for pooled image features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3370–3377, 2012. 5

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 1097–1105. 2012. 1, 2

[18] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2169–2178, 2006. 5

[19] T. Lin, M. Maire, S. J. 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 (ECCV)*, pages 740–755, 2014. 2, 5, 6, 8

[20] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2

[21] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar. Focal loss for dense object detection. In *IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 1, 6

[22] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg. SSD: single shot multibox detector. In *European Conference on Computer Vision (ECCV)*, pages 21–37, 2016. 2, 4, 6, 7

[23] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, 2015. 1, 6

[24] D. G. Lowe. Object recognition from local scale-invariant features. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1150–1157, 1999. 1

[25] S. Mallat. *A Wavelet Tour of Signal Processing, 2nd Edition*. Academic Press, 1999. 6

[26] T. Mordan, N. Thome, M. Cord, and G. Henaff. Deformable part-based fully convolutional network for object detection. *CoRR*, abs/1707.06175, 2017. 1, 2, 3, 5, 6, 7, 8

[27] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016. 2

[28] S. Ren, K. He, R. B. Girshick, and J. Sun. 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. 1, 2, 3, 4, 5, 6, 7, 8

[29] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. Lecun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2014. 2

[30] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 2, 6, 7

[31] P. A. Viola and M. J. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 511–518, 2001. 1, 2

[32] X. Wang, M. Yang, S. Zhu, and Y. Lin. Regionlets for generic object detection. In *IEEE International Conference on Computer Vision (ICCV)*, pages 17–24, 2013. 1, 2, 3, 6, 7

[33] X. Wang, M. Yang, S. Zhu, and Y. Lin. Regionlets for generic object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(10):2071–2084, 2015. 1

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