IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 4, 2017, accepted October 29, 2017, date of publication November 7, 2017,
date of current version November 28, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2770178

G-CNN: Object Detection via Grid Convolutional Neural Network

QISHUO LU[1], CHONGHUA LIU[2], ZHUQING JIANG[1], AIDONG MEN[1], AND BO YANG[1]

[1]Multimedia Technology Center, School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China

[2]China Academy of Space Technology, Beijing 100094, China

Corresponding author: Qishuo Lu (luqishuo@126.com)

This work was supported by the Project of National Natural Science Foundation of China under Grant 61671077 and Grant 61671264.

ABSTRACT We propose an object detection system that depends on position-sensitive grid feature maps. State-of-the-art object detection networks rely on convolutional neural networks pre-trained on a large auxiliary data set (e.g., ILSVRC 2012) designed for an image-level classification task. The image-level classification task favors translation invariance, while the object detection task needs localization representations that are translation variant to an extent. To address this dilemma, we construct position-sensitive convolutional layers, called grid convolutional layers that activate the object's specific locations in the feature maps in the form of grids. With end-to-end training, the region of interesting grid pooling layer shepherds the last set of convolutional layers to learn specialized grid feature maps. Experiments on the PASCAL VOC 2007 data set show that our method outperforms the strong baselines faster region-based convolutional neural network counterpart and region-based fully convolutional networks by a large margin. Our method applied to ResNet-50 improves the mean average precision from 74.8%/74.2% to 79.4% without any other tricks. In addition, our approach achieves similar results on different networks (ResNet-101) and data sets (PASCAL VOC 2012 and MS COCO).

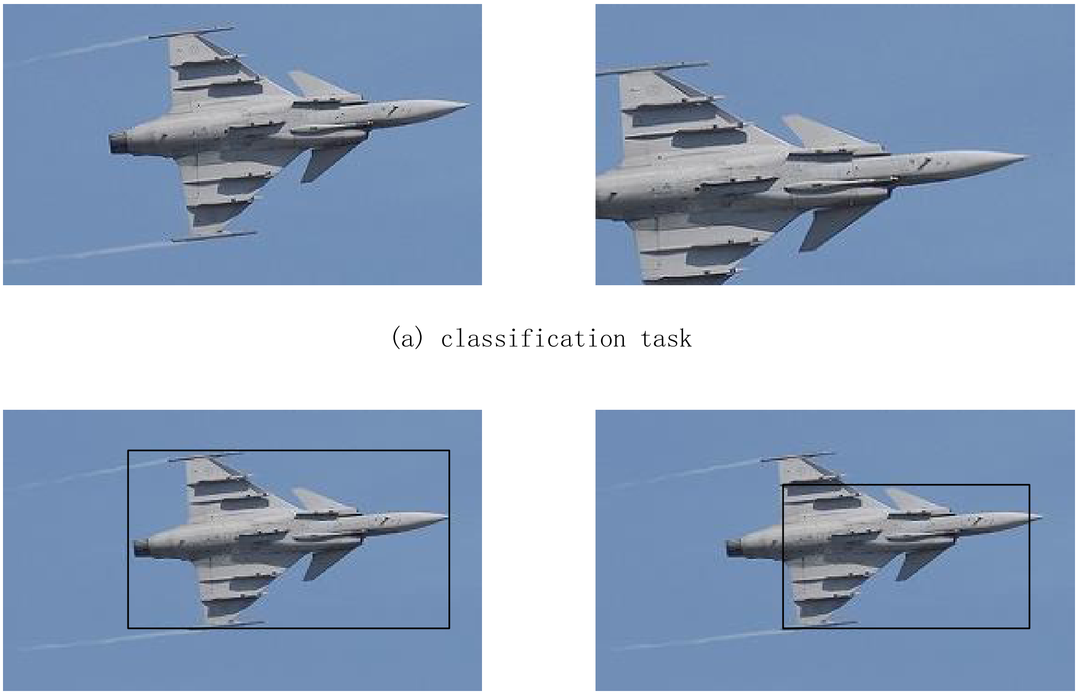
INDEX TERMS Computer vision, deep learning, grid feature map, object detection, region proposal.

I. INTRODUCTION

Object detection is one of the key tasks in the area of computer vision. The task outputs the type of every object and locates it with a tightly surrounding bounding box, and many more advanced tasks rely on object detection (e.g., instance segmentation [3], [4] and human-object interactions [5], [6]). In the last few years, object detection has seen rapid development thanks to significant developments in deep learning [7], [8], especially Convolution Neural Network (CNN) architectures [2], [9]–[12]. Among object detection methods [13]–[16], one of the most notable works is the R-CNN series [16]–[18].

R-CNN [16] uses the Selective Search method [19] to extract region proposals [20]; then, it uses CNNs (i.e., AlexNet [21]) to extract features for each region proposal. Finally, it classifies CNN features with class-specific linear SVMs [22], [23]. R-CNN first pre-trains the networks on a large auxiliary dataset (e.g., ILSVRC 2012 [24]) that is annotated for image classification task, and then, it is fine tuned on the PASCAL VOC [25] dataset, annotated for object detection. Due to the multi-stage training procedure and time-consuming nature of R-CNN, Fast R-CNN [17] and Faster R-CNN [18] have been proposed to improve the training and testing speed as well as the accuracy with simpler pipelines. All these methods are pre-trained on an auxiliary task of image classification. However, image classification task and object detection task present different requirements for the network.

The image-level classification task prefers translation invariance, while the object detection task favors localization representations that are translation variant to an extent [1]. Figure 1 illustrates the difference between image classification task and object detection task. The image-level classification task prefers translation invariance – when moving an object inside an image, there should be no discrimination between images. Therefore, for the image classification task, a stronger invariance of the deep neural network provides better results. State-of-the-art image classification networks have very strong invariance, as shown by the excellent performances on the ImageNet classification task [2], [10], [12]. The object detection task needs localization representations that are translation variant to an extent – translating an object inside a candidate box should be discriminative and indicate how well the candidate box overlaps the ground truth [1].



(a) classification task

(b) object detection task

FIGURE 1. The difference between image-level classification task and object detection task. (a) The image-level classification task prefers translation invariance – when moving an object within an image, there should be no discrimination between images. (b) The object detection task needs localization representations that are translation variant to an extent – translating an object inside a candidate box should be discriminative and indicate how well the candidate box overlaps the ground truth.

Deep convolutional neural networks for image classification are less sensitive to translation. To solve this dilemma, the ResNet paper's [2] object detection pipeline puts the RoI pooling layer [26] in the convolutional layers to cut off the translation invariance of the post-RoI convolutional layers. However, the method introduces a large number of RoI-wise layers, thus greatly reducing the speed of training and testing. R-FCN [1] uses position-sensitive score maps, and each score map encodes information with regard to a specific location. R-FCN is a fully convolutional network, and its position-sensitive module consists of a bank of convolutional layers that generate position-sensitive maps and a position-sensitive RoI pooling layer with no learned weight (convolutional/fully connected) layers following. The feature maps in R-FCN are class-aware and position-aware maps, and although the total number of features is large, each class corresponds to very few feature maps. Due to this structural limitation, it is inconvenient to apply this method to a stronger classifier.

Inspired by R-FCN, we propose a type of position-sensitive convolutional layer called Grid Convolutional Layer (GCL). Figure 2 illustrates the main architecture. The input image first passes through some convolutional layers and max pooling layers to generate feature maps. Then, a grid convolutional sub-network is introduced to generate position-sensitive feature maps. The network consists of a set of GCLs that produce position complementary grid features and has an RoI grid pooling layer at the end. Each output of the RoI grid pooling layer comes from a different feature map in an alternating manner. With end-to-end training, the RoI grid pooling layer shepherds the GCL to learn specialized grid feature maps.

Using the 50-layer Residual Net (ResNet-50) [2] as the backbone, experiments on the PASCAL VOC [25] 2007 dataset show that our method outperforms the strong baselines Faster R-CNN counterpart [2] and R-FCN [1] by a large margin. Our method improves the mAP from 74.8%/74.2% to 79.4% without using any tricks. Meanwhile, the test time of our method is 0.16 seconds per image, which is approximately 2× faster than the Faster R-CNN with ResNet-50 counterpart in [2]. In addition, our approach achieves similar results on different networks (ResNet-101) and datasets (PASCAL VOC 2012 and MS COCO).

II. RELATED WORK

Object detection attempts to recognize and locate each object with a bounding box within an image. Object detection methods can be divided into two categories: region-based detection methods, such as R-CNN [16], FPN [27], and RoN [28], and regression-based methods such as YOLO [15], SSD [29], and YOLO9000 [30]. Region-based methods can provide better features for the classifier [18]; therefore, the accuracy is higher. However, each candidate has to go through the

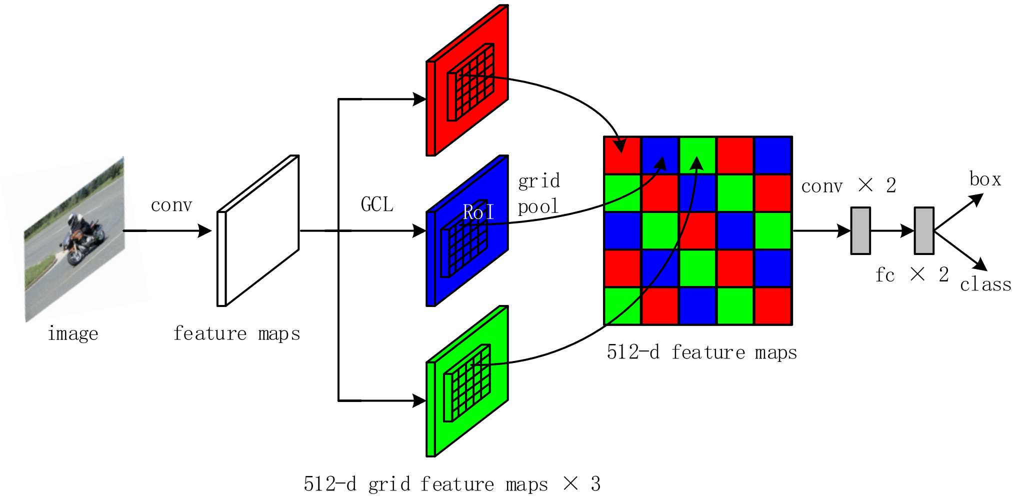


FIGURE 2. The architecture of G-CNN for object detection. The input image first passes through some convolutional layers and max pooling layers to generate feature maps. Then, a grid convolutional sub-network is introduced to generate position-sensitive feature maps. The sub-network consists of a set of position-sensitive convolutional layers (GCLs) that produce position complementary grid features and has an RoI grid pooling layer at the end. Each output of the RoI grid pooling layer comes from a different feature map in an alternating manner. With end-to-end training, the RoI grid pooling layer shepherds the GCL to learn specialized grid feature maps.

region-wise sub-network; therefore, the speed is lower. Regression-based methods can use the fully convolutional network to directly return the object location and category in the image, but they are less accurate.

Most state-of-the-art region-based object detection pipelines follow the Region-based Convolutional Neural Network (R-CNN) [16]. In R-CNN, region proposals are first generated by a manually designed method (e.g., Edge Box [31], MCG [32] and Selective Search [19]) from the input image. R-CNN then uses a CNN to extract feature maps for each region proposal. Finally, the bounding box regression and classification are performed to discriminate the target objects. The object classifier, bounding box regressor and CNN are trained separately through a multi-stage training pipeline, and thus, training R-CNN is expensive in terms of computation and memory requirements. To improve the computational efficiency and detection accuracy, Fast R-CNN [17] has been developed to address the above problems. First, the training process is a single-stage process – feature extraction, classification and bounding box regression are performed by a network; region proposals within the same image share their calculation, which greatly improves the speed of the training and testing phases. R-CNN and Fast R-CNN are based on region proposals; the region proposal generation process is computational expensive and affects the overall speed. To further reduce the time of generating region proposals, Faster R-CNN [18] introduces a novel Region Proposal Network (RPN), which can be embedded in the Fast R-CNN framework for region proposal generation. RPN shares the convolutional computation of the entire image, and the region proposal generation is almost performed at zero cost. The RPN simultaneously predicts the object bounding box and the classification score at each location. RPN training is end-to-end and produces high-quality region proposals.

YOLO [15] treats object detection as a regression problem. Based on a single end-to-end network, it directly predicts the object position and category from the input image. The network divides the image into regions and predicts the bounding boxes and probabilities of each region. These bounding boxes are weighed by the predicted probabilities. YOLO detects the entire image at the test time so that the forecast is based on the global contextual information of the entire image. This technique is simply based on a single network to generate one prediction and does not include region-wise sub-networks that operate thousands of times on the region proposals such as in Fast R-CNN [17]. SSD [29] outputs the predicted bounding boxes from a set of default boxes over different scales and aspect ratios of each feature map location. At the testing phase, the network produces probabilities for each predicted object category per default box and generates adjustments to the default box to achieve better localization. Moreover, SSD detects objects on multiple feature maps and predicts objects of different scales on the corresponding resolution feature maps.

However, those methods are first pre-trained on a large auxiliary dataset for image classification. To solve this dilemma, the ResNet paper's [2] object detection pipeline inserts the RoI pooling layer into the convolutional layer to cut off the translation invariance of the post-RoI convolutional layers. However, the method introduces a considerable number of RoI-wise layers, thus greatly increasing the training and testing times. R-FCN [1] utilizes position-sensitive score maps, where each score map encodes information with regard to a specific location. It is a fully convolutional network, and its position-sensitive module contains a set of

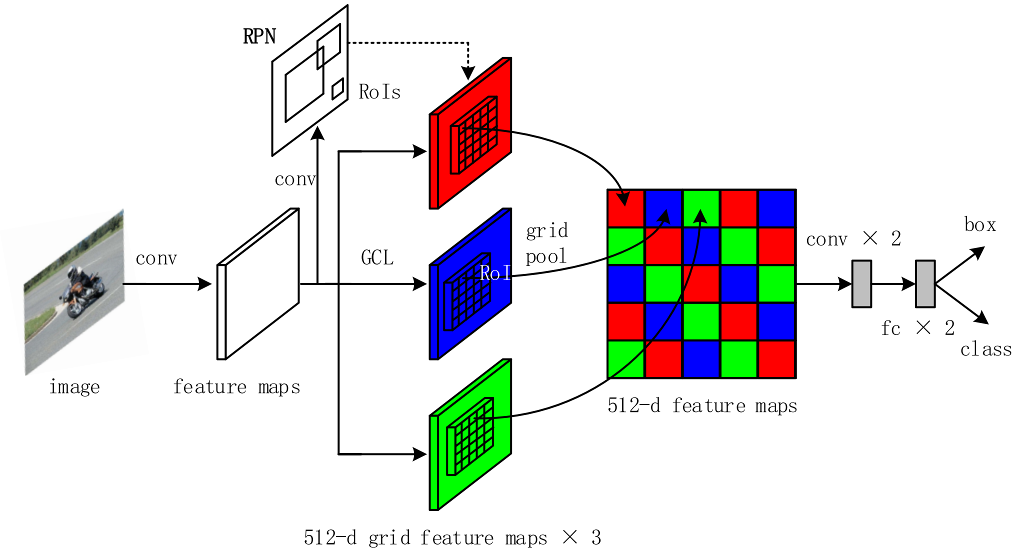


FIGURE 3. Overview of G-CNN. The input image first passes through some convolutional layers and max pooling layers to generate the feature maps. Then, the grid convolutional sub-network is introduced to generate position-sensitive feature maps. The sub-network consists of a set of position-sensitive convolutional layers (GCL) that produce position complementary grid features and has an RoI grid pooling layer at the end.

convolutional layers that generate position-sensitive maps and a position-sensitive RoI pooling layer with no learned weight (convolutional/fully connected) layers following. The feature maps of R-FCN are class-aware and position-aware maps, and although the total number of features is considerable, each class corresponds to very few feature maps. Due to this structural limitation, it is inconvenient to apply to a stronger classifier.

III. OUR APPROACH

A. OVERVIEW

Following R-CNN, we adopt a prevalent two-stage object detection pipeline [27], [33]–[36] that consists of generating region proposals and proposal classification. We generate candidate boxes by the Region Proposal Network (RPN) [18] and share features between RPN and Grid Convolutional Neural Network (G-CNN). Figure 3 shows an overview of the architecture.

The input image first passes through some convolutional layers and max pooling layers to generate feature maps. Then, the grid convolutional sub-network is introduced to generate position-sensitive feature maps. The sub-network consists of a set of GCLs that produce position complementary grid features and has an RoI grid pooling layer at the end. The set of GCLs generates m sets of feature maps, where each set of feature maps has c channels.

G-CNN ends with a grid pooling layer. The layer aggregates all the location feature maps from the output of different convolutional layers. Each output of the RoI pooling layer comes from a different feature map in an alternating manner. With the end-to-end training, the RoI grid pooling layer shepherds the GCLs to learn specialized grid feature maps. Ren *et al.* [37] showed that networks on convolutional feature maps improve the object detection process. Given that finding, G-CNN is followed by two convolutional layers and fully connected layers that ultimately branch into two sibling output layers: one produces softmax probability outputs over all detection classes plus a "background" class, and the other sibling produces four real-valued numbers for each class [26]. Each set of four values encodes a refined bounding box position for one of the classes. For fair comparison, the RPN is built on top of the conv4 stage, as is the case for Faster R-CNN in [2].

B. GRID FEATURE MAPS AND RoI GRID POOLING

We divide each RoI region into $n \times n$ bins using a regular grid. For a $w \times h$ RoI region, the size of a bin is approximately $\frac{w}{n} \times \frac{h}{n}$ [26]. Each bin is derived from a different feature map in an alternating manner; therefore, the technique explicitly encodes all position information into the RoI. In this model, the GCL generates m sets of feature maps. Inside the $(i,j)-th$ bin $(0 \le i, j \le n-1)$, we define an RoI grid pooling operation that pools only over the $((j \times n+i)\%m)-th$ set of grid feature maps.

$$r_c(i,j \mid \Theta) = \max_{(x,y)\in bin(i,j)} z_{i,j,c,l}(x+x_0, y+y_0 \mid \Theta) \quad (1)$$

$$l = (j \times n + i)\%m \quad (2)$$

Here, $r_c(i,j)$ is the pooled response in the $(i,j) - th$ bin for the $c - th$ channel; $z_{i,j,c,l}$ is the feature map of the $c - th$ channel from the $l - th$ set of grid feature maps; and (x_0, y_0) denotes the top-left corner of an RoI. The $(i,j) - th$ bin spans $\lfloor i \cdot \frac{w}{n} \rfloor \le x \le \lceil (i+1) \cdot \frac{w}{n} \rceil$ and $\lfloor j \cdot \frac{h}{n} \rfloor \le y \le \lceil (j+1) \cdot \frac{h}{n} \rceil$. The operation of Eq.(1) is illustrated in Figure 3. Eq. (1) performs the max pooling operation.

The concept of G-CNN is partially inspired by R-FCN [1], which was developed for efficient object detection and whose detector is fully convolutional, with almost all computations shared across the entire image. The RoI pooling layer conducts selective pooling, and each of the $n \times n$ bins aggregates from one score map out of the bank of $n \times n$ score maps and corresponds to a specific class. Therefore, the feature maps of R-FCN are class-aware and position-aware feature maps – although the total number of features is large, each class corresponds to very few features. Due to these structural limitations, this technique is inconvenient to apply to a stronger classifier; therefore, to obtain higher accuracy, we propose the G-CNN.

C. BACKBONE ARCHITECTURE

In this paper, the architecture is based on ResNet [2], although other networks [10], [12] are also feasible. ResNet computes a feature hierarchy consisting of feature maps at several scales with a scaling step of 2. There are many layers producing output maps of the same size and we say these layers are in the same network stage. We denote the forth stage and fifth stage as the conv4 stage and conv5 stage respectively. ResNet-50 (ResNet-101) has 50 (101) convolutional layers, followed by an average pooling layer and a 1000-way fully connected layer. We only use convolutional layers to extract feature maps; the average pooling and the fully connected layers are removed. The output of the last convolutional block in ResNet-50 is $2048-d$, and we attach a $1024-d$ 1×1 convolutional layer randomly initialized to reduce the dimension. ResNet-50's effective stride is 32 pixels. To obtain a higher resolution of the feature map, we reduce the effective stride to 16 pixels. All the layers before and on the conv4 stage [2] remain unchanged. In addition, the stride of 2 operations in the first conv5 block is changed to have a stride of 1, and all other conv5 stage convolutional filters are modified using the "hole algorithm" [38], [39] to compensate for the step reduction.

D. OPTIMIZATION

We randomly initialize all new layers by drawing weights from a zero-mean Gaussian distribution with standard deviation 0.01. Using a single-scale training, the shorter side of the image is scaled to 600 pixels, and the longer side does not exceed 1000 pixels. Each GPU holds one image and selects 128 region proposals. We train the net using 8 GPUs; therefore, the effective mini-batch is eight. We use a momentum of 0.9 and a weight decay of 0.0005 as in [21]. We fine-tune G-CNN using a learning rate of 0.001 for 20k mini-batches and 0.0001 for the subsequent 10k mini-batches on the PASCAL VOC dataset. G-CNN shares features with RPN and uses the approximate joint training [18].

E. ANALYSIS OF POSITION SENSITIVITY

Figure 4 illustrates how the G-CNN works. For clarity of explanation, it is assumed that only the CNN feature maps corresponding to the ground-truth bounding box region are activated. (a) Normal Network. The region proposal on the left is well covered by the object, and the activation level of the corresponding feature maps is relatively high. The right region proposal is not as well covered by the object, and the feature maps of the overlapping area can still be activated. We hope that the score on the left is higher than that on the right so that we can achieve a better localization. However, state-of-the-art image classification networks have very strong translation invariance, therein usually failing to achieve a good distinction between them. (b) G-CNN. The region proposal on the left is well overlapped with the object, and the activation level of the corresponding feature maps is relatively high. The right region proposal is not as well covered by the object, and the feature maps of the overlapping region cannot be activated. This is because the GCL obtains position-sensitive grid feature maps. The RoI pooling layer divides each RoI region into $n \times n$ bins, where each bin comes from a different feature map in an alternating manner. Therefore, the score on the better region proposal is higher; we can distinguish them explicitly and achieve a better localization.

IV. EXPERIMENTS

A. EXPERIMENTS ON PASCAL VOC

We evaluate our method on the PASCAL VOC dataset [25], which includes 20 categories. We train the system on the union set of VOC 2012 *trainval* and VOC 2007 *trainval* ("07+12"), and we evaluate the system on the VOC 2007 *test* set. The evaluation criterion for object detection is the mean average precision (mAP).

We conduct comparative experiments on the following related networks:

G-CNN without position sensitivity. By setting $k = 1$ to remove the position sensitivity, this network is equivalent to global pooling within each RoI.

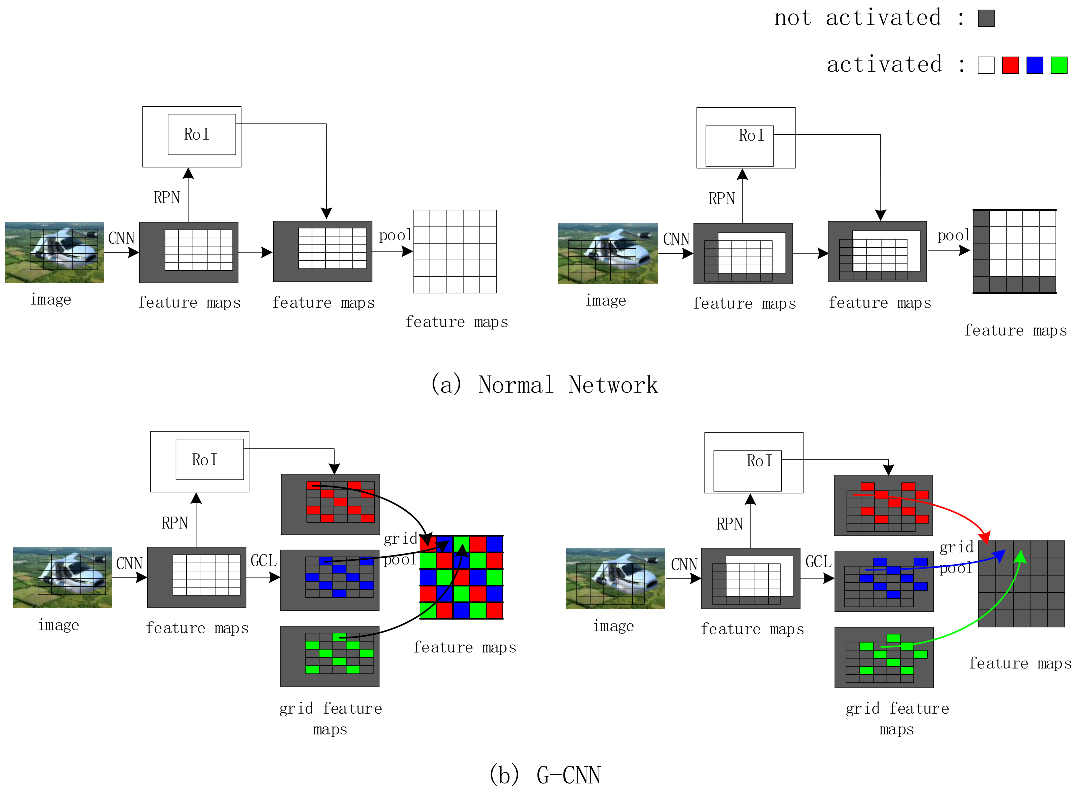
R-FCN [1]. This is a region-based, fully convolutional neural network with almost all computations being shared across the entire image. Its convolutional layer is followed by a position-sensitive pooling layer but is not followed by any weight layers.

Faster R-CNN counterpart [2]. In this architecture, the layers prior to the conv4 stage (including conv4) are used to extract the feature maps. This network inserts the ROI pooling layer between conv4 and conv5; the other layers are followed as the classifier for every RoI. Therefore, the design sacrifices training and testing efficiency because it introduces heavy region-wise layers.

TABLE 1. Detection results of PASCAL VOC 2007 *test* set using the ResNet-50 model.

Method	Training Dataset	mAP (%) on VOC07
Faster R-CNN [2]	07+12	74.8%
R-FCN [1]	07+12	74.2%
G-CNN [ours]	07+12	**79.4**%

In Table 1, we provide comparisons with current state-of-the-art results, including Faster R-CNN and R-FCN. We note



(a) Normal Network

(b) G-CNN

FIGURE 4. Analysis of position sensitivity. For clarity of explanation, it is assumed that only the CNN feature maps corresponding to the ground-truth bounding box region are activated. (a) Normal Network. The region proposal on the left is well covered with the object, and the activation level of the corresponding feature maps is relatively high. The right region proposal is not as well covered by the object, and the feature maps of the overlapping area can still be activated. We hope that the score on the left is higher than that on the right so that we can achieve a better localization. However, state-of-the-art image classification networks have very strong translation invariance, therein usually failing to make a good distinction between them. (b) G-CNN. The region proposal on the left is well overlapped with the object, and the activation level of the corresponding feature maps is relatively high. The right region proposal is not as well covered by the object, and the feature maps of the overlapping region cannot be activated. This is because the GCL obtains position-sensitive grid feature maps. The RoI pooling layer divides each RoI region into *n* × *n* bins, where each bin comes from a different feature map in an alternating manner. Therefore, the score on the better region proposal is higher; we can distinguish them explicitly and achieve a better localization.

that all methods in Table 1 are based on RPN built on the top of conv4 blocks.

Our methods achieves 79.4% mAP on the PASCAL VOC 2007 test set. Faster R-CNN and R-FCN achieve mAPs of 74.8% and 74.2%, respectively. The accuracy of G-CNN exceeds Faster R-CNN and R-FCN by a large margin. Faster R-CNN inserts the ROI pooling layer between conv4 and conv5 to break down the translation invariance. However, this model introduces a substantial number of region-wise layers, therein failing to fully utilize the depth of the network to extract more semantic features. R-FCN is a region-based, fully connected network, and there is no parameter layer that can be learned after the RoI pooling layer. The detection speed of R-FCN is higher, but R-FCN cannot be applied to heavier classifiers, thus limiting its accuracy.

The importance of G-CNN is further proved by setting $m = 1$ (Table 2), upon which the accuracy drops to 78.2%. However, if m is set to 5, the accuracy drops to 78.6%. The object detection process also requires a certain degree of translation invariance. If m is set to 5, the network's translation variance becomes too strong.

TABLE 2. The effect of the types (*m*) of grid feature maps. A higher value of *m* indicates a stronger sensitivity and vice versa.

Method	m	mAP (%) on VOC07
G-CNN without position sensitivity	1	78.2%
G-CNN with position sensitivity	3	**79.4%**
G-CNN with position sensitivity	5	78.6%

TABLE 3. The comparisons of speeds on the PASCAL VOC 2007 *test* set using ResNet-50. Timing is evaluated on a single Nvidia Titan X GPU. A total of 300 RoIs per image are computed in the forward pass, and 128 samples are selected for back propagation. A total of 300 RoIs are used for testing.

Method	Training Time (sec/img)	Testing Time (sec/img)
Faster R-CNN [2]	1.12	0.35
R-FCN	**0.36**	**0.08**
G-CNN	0.64	0.16

Table 3 shows a comparison in terms of speed. With 300 RoIs at testing time, Faster R-CNN, R-FCN, and G-CNN require 0.35 s, 0.08 s, and 0.16 s per image, respectively. Faster R-CNN utilizes a large region-wise sub-network

TABLE 4. The comparisons on the PASCAL VOC 2007 *test* set using ResNet-101. "Faster R-CNN + + + [2]" uses context, multi-scale testing and iterative box regression. R-FCN [1] and G-CNN use OHEM [40] with 300 RoIs.

Method	Training Data	with OHEM	mAP (%) on VOC07
Faster R-CNN [2]	07+12		76.4
Faster R-CNN +++ [2]	07+12+CoCo		**85.6**
R-FCN	07+12	√ (300 RoIs)	79.5
R-FCN $_{multi-sc\ train}$	07+12	√ (300 RoIs)	80.6
R-FCN $_{multi-sc\ train}$	07+12+CoCo	√ (300 RoIs)	**83.5**
G-CNN	07+12	√ (300 RoIs)	82.5
G-CNN $_{multi-sc\ train}$	07+12	√ (300 RoIs)	83.6
G-CNN $_{multi-sc\ train}$	07+12+CoCo	√ (300 RoIs)	**86.5**

TABLE 5. The comparisons on the PASCAL VOC 2012 *test* set using ResNet-101. "07++12" denotes the union set of 2007 *trainval*+*test* and 2012 *trainval*.

Method	Training Data	mAP (%)	Test Time (sec/img)
Faster R-CNN [2]	07++12	73.8	0.44
Faster R-CNN +++ [2]	07++12+COCO	83.8	3.50
R-FCN $_{multi-sc\ train}$ [1]	07++12	77.6	**0.16**
R-FCN $_{multi-sc\ train}$ [1]	07++12+COCO	82.0	**0.16**
G-CNN $_{multi-sc\ train}$ [ours]	07++12	79.6	0.22
G-CNN $_{multi-sc\ train}$ [ours]	07++12+COCO	**84.8**	0.22

TABLE 6. The comparisons on the MS COCO dataset using ResNet-101. We evaluate the mAP on COCO's standard metric averaged for *IoU* ∈ [0.5 : 0.05 : 0.95] (simply denoted as mAP@[.5, .95]) and mAP@0.5 (PASCAL VOC's metric).

Method	Training Data	Test Data	mAP@0.5	mAP@[.5, .95]	mAP@[.5, .95]	mAP@[.5, .95]	mAP@[.5, .95]
					small	medium	large
Faster R-CNN [2]	train	val	48.4	27.2	6.6	28.6	45.0
R-FCN	train	val	48.9	27.6	8.9	30.5	42.0
R-FCN $_{mutil-sc\ train}$	train	val	49.1	27.8	8.8	30.8	42.2
G-CNN	train	val	52.3	29.5	**9.5**	32.6	44.9
G-CNN $_{mutil-sc\ train}$	train	val	**52.4**	**29.7**	9.4	**32.9**	**45.1**
Faster R-CNN +++ [2]	trainval	test-dev	55.7	**34.9**	**15.6**	**38.7**	**50.9**
R-FCN	trainval	test-dev	51.5	29.2	10.3	32.4	43.3
R-FCN $_{mutil-sc\ train}$	trainval	test-dev	51.9	29.9	10.8	32.8	45.0
R-FCN $_{mutil-sc\ train,\ test}$	trainval	test-dev	53.2	31.5	14.3	35.5	44.2
G-CNN	trainval	test-dev	53.5	31.0	10.9	34.3	45.9
G-CNN $_{mutil-sc\ train}$	trainval	test-dev	54.9	31.6	11.4	34.7	47.6
G-CNN $_{mutil-sc\ train,\ test}$	trainval	test-dev	**56.0**	33.2	15.1	37.4	46.5

to achieve good accuracy, R-FCN has a negligible per-region cost, and our method includes a region-wise sub-network consisting of two convolutional layer and three fully connected layers. Therefore, R-FCN is the fastest method, followed by our method, and Faster R-CNN is the slowest.

Table 4 shows the results of using ResNet-101 network. R-FCN [1] and G-CNN use online hard example mining (OHEM [40]). The mAP of G-CNN is 82.5%. Moreover, using the multi-scale training strategy in [26], we resize the scale of image in each iteration to a random value chosen from {400, 500, 600, 700, 800} pixels. We continue to performing testing at a single scale of 600 pixels, leading to no additional testing time. The mAP is 83.6%. In addition, we train the system on the MS COCO [41] *trainval* and then fine-tune the system on the PASCAL VOC dataset. The mAP is 86.5%, which is 0.9% higher than the strongest competitor "Faster R-CNN +++" [2]. "Faster R-CNN +++" uses context, multi-scale testing, iterative box regression and a 10-layer sub-network to evaluate each region proposal; therefore, using OHEM is time consuming. The results of multi-scale testing are aggregated as in [2].

Table 5 shows the results on the PASCAL VOC 2012 *test* set and presents similar conclusions; this proves that our method has a certain degree of generalizability.

B. EXPERIMENTS ON MS COCO

Next, we evaluate our method on the MS COCO object detection dataset [41]. This dataset possesses 80 object categories. We experiment with 80k images on the *train* set, 40k images on the *val* set and 20k images on the *test-dev* set. We evaluate the mAP on COCO's standard metric averaged for $IoU \in [0.5 : 0.05 : 0.95]$ (simply denoted as mAP@[.5, .95]) and mAP@0.5 (PASCAL VOC's metric). The learning rate is 0.001 for the first 90k iterations but subsequently is reduced to 0.0001 for the next 30k iterations.

The results are presented in Table 6. Our single-scale trained G-CNN with the *train* set achieves a *val* result of 52.3%/29.5%. This is better than that of the Faster R-CNN (48.4%/27.2%) and R-FCN (48.9%/27.6%). It is worth noting that our method is more accurate for small objects defined by [41]. Our multi-scale trained (still single-scale tested) model with the *train* set achieves a result of 52.4%/29.7% on the *val* set, while the multi-scale trained R-FCN achieves 49.1%/27.8%. The multi-scale trained G-CNN with the *trainval* set achieves a result of 54.9%/31.6% on the *test-dev* set, while the multi-scale trained R-FCN achieves 51.9%/29.9%. Considering the wide range of the scale distribution of the COCO dataset, we further consider using multi-scale testing, and the scales in this testing process are $\{200, 400, 600, 800, 1000\}$. The mAP is 56.0%/33.2%, and the R-FCN is 53.2%/31.5%. This result is on par with the strongest baseline "Faster R-CNN + + +" (55.7%/34.9%) and is simpler, with no other bells and whistles (e.g., iterative box regression or context).

V. CONCLUSION

We introduced the Grid Convolutional Neural Network, a simple but accurate and efficient architecture. Our system adopts a state-of-the-art region-based object detection architecture, therein using the prevalent ResNet and RPN datasets. With the proposed GCLs, our method outperforms the Faster R-CNN counterpart [2] in terms of both speed and accuracy by a large margin. The RPN is built on normal feature maps, and in the future, we can consider how to use position-sensitive feature maps to generate better region proposals.

REFERENCES

[1] Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 379–387.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[3] J. Dai, K. He, and J. Sun, "Instance-aware semantic segmentation via multi-task network cascades," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3150–3158.

[4] K. He, G. Gkioxari, P. Dollár, and R. Girshick. (Mar. 2017). "Mask R-CNN." [Online]. Available: https://arxiv.org/abs/1703.06870

[5] G. Gkioxari, R. Girshick, P. Dollár, and K. He. (Apr. 2017). "Detecting and recognizing human-object interactions." [Online]. Available: https://arxiv.org/abs/1704.07333

[6] P. Wei, Y. Zhao, N. Zheng, and S.-C. Zhu, "Modeling 4D human-object interactions for event and object recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 3272–3279.

[7] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[8] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[9] Y. Chen, J. Li, H. Xiao, X. Jin, S. Yan, and J. Feng. (Jul. 2017). "Dual path networks." [Online]. Available: https://arxiv.org/abs/1707.01629

[10] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[11] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.

[12] K. Simonyan and A. Zisserman. (Sep. 2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556

[13] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2008, pp. 1–8.

[14] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. (Dec. 2013). "OverFeat: Integrated recognition, localization and detection using convolutional networks." [Online]. Available: https://arxiv.org/abs/1312.6229

[15] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[16] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Columbus, OH, USA, Jun. 2014, pp. 580–587.

[17] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1440–1448.

[18] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[19] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, 2013.

[20] J. Hosang, R. Benenson, P. Dollár, and B. Schiele, "What makes for effective detection proposals?" *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 4, pp. 814–830, Apr. 2016.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[22] C. Cortes and V. Vapnik, "Support vector machine," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[23] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining Knowl. Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[24] O. Russakovsky *et al.* (Sep. 2014). "Imagenet large scale visual recognition challenge." [Online]. Available: https://arxiv.org/abs/1409.0575

[25] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Sep. 2009.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015.

[27] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. (Dec. 2016). "Feature pyramid networks for object detection." [Online]. Available: https://arxiv.org/abs/1612.03144

[28] T. Kong, F. Sun, A. Yao, H. Liu, M. Lu, and Y. Chen. (Jul. 2017). "RON: Reverse connection with objectness prior networks for object detection." [Online]. Available: https://arxiv.org/abs/1707.01691

[29] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Proc. 14th Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 21–37.

[30] J. Redmon and A. Farhadi. (Dec. 2016). "YOLO9000: Better, faster, stronger." [Online]. Available: https://arxiv.org/abs/1612.08242

[31] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *Proc. ECCV*, 2014, pp. 391–405.

[32] J. Pont-Tuset, P. Arbeláez, J. T. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping for image segmentation and object proposal generation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 1, pp. 128–140, Jan. 2017.

[33] T. Kong, A. Yao, Y. Chen, and F. Sun, "HyperNet: Towards accurate region proposal generation and joint object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 845–853.

[34] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick, "Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2874–2883.

[35] S. Gidaris and N. Komodakis, "Object detection via a multi-region and semantic segmentation-aware CNN model," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1134–1142.

[36] X. Wang, A. Shrivastava, and A. Gupta. (Apr. 2017). "A-Fast-RCNN: Hard positive generation via adversary for object detection." [Online]. Available: https://arxiv.org/abs/1704.03414

[37] S. Ren, K. He, R. Girshick, X. Zhang, and J. Sun, "Object detection networks on convolutional feature maps," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 7, pp. 1476–1481, Jul. 2017.

[38] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published.

[39] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 640–651, Apr. 2017.

[40] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 761–769.

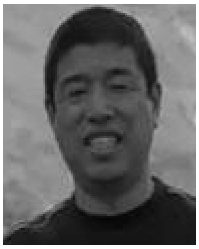
[41] T.-Y. Lin *et al.* (Feb. 2014). "Microsoft COCO: Common objects in context." [Online]. Available: https://arxiv.org/abs/1405.0312

QISHUO LU received the B.S. degree in electronic information engineering and the M.S. degree in information and communication engineering from the China University of Petroleum (East China). He is currently pursuing the Ph.D. degree in information and communication engineering with the Beijing University of Posts and Telecommunications. His research interests include object detection and image classification.

CHONGHUA LIU received the master's degree in circuit and system from BUPT in 1987. She is currently a Research Scientist with the China Academy of Space Technology, China. Her research interest includes satellite communication on measurement and control.

ZHUQING JIANG received the B.S. degree from Beijing Forestry University in 2008 and the Ph.D. degree from the Beijing University of Posts and Telecommunications, Beijing, in 2014, where he is currently pursuing the Ph.D. degree. Since 2014, he has been a Lecturer in communication and information engineering with the Beijing University of Posts and Telecommunications. He has published several papers in journals and international conference. His research interests include satellite communications and multimedia signal processing.

AIDONG MEN received the B.S., M.S., and Ph.D. degrees from the Department of Radio Engineering, Beijing University of Posts and Telecommunications, Beijing, in 1994. From 1994 to 2000, he was an Associate Professor with the Department of Radio Engineering, Beijing University of Posts and Telecommunications. Since 2000, he has been a Professor with the Telecom Engineering College, Beijing University of Posts and Telecommunications. He is currently a fellow of the Chinese Institute of Electronics and the China Institute of Communications. He has published over 100 papers in journals and international conference. His research interests include multimedia communication, digital TV, and images and speech signal processing and transmission. He is also an invited fellow of the Science and Technology Committee of State Administration of Radio, Film and Television.



BO YANG received the B.S. degree from the Department of Radio Engineering, Beijing University of Posts and Telecommunications, Beijing, in 1982. Since 1982, he has been an Associate Professor in communication and information engineering with the Beijing University of Posts and Telecommunications. He is currently a fellow of the China Institute of Communications. He has presided a number of research projects on multimedia communication. His research interests include multimedia communication and mobile communication. He is also a Committee Member of the National Ministry of Science and Technology Confidentiality Review Expert Committee.