Enhanced Object Detection with Deep Convolutional Neural Networks for Advanced Driving Assistance

Jian Wei, Jianhua He†, Yi Zhou, Kai Chen, Zuoyin Tang and Zhiliang Xiong

Abstract—**Object detection is a critical problem for advanced driving assistance systems (ADAS). Recently convolutional neural networks (CNN) achieved large successes on object detection, with performance improvement over traditional approaches, which use hand-engineered features. However, due to the challenging driving environment (e.g., large object scale variation, object occlusion and bad light conditions), popular CNN detectors do not achieve very good object detection accuracy over the KITTI autonomous driving benchmark dataset. In this paper we propose three enhancements for CNN based visual object detection for ADAS. To address the large object scale variation challenge, deconvolution and fusion of CNN feature maps are proposed to add context and deeper features for better object detection at low feature map scales. In addition, soft non-maximal suppression (NMS) is applied across object proposals at different feature scales to address the object occlusion challenge. As the cars and pedestrians have distinct aspect ratio features, we measure their aspect ratio statistics and exploit them to set anchor boxes properly for better object matching and localization. The proposed CNN enhancements are evaluated with various image input sizes by experiments over KITTI dataset. Experiment results demonstrate the effectiveness of the proposed enhancements with good detection performance over KITTI test set.**

I. Introduction

Visual object detection is a long standing and important research problem for computer vision, with a wide range of real world applications, such as robotic vision, surveillance, ADAS and autonomous driving [1]. Its main task is to predict the position and category of objects from images or videos. Traditionally hand-crafted features have been used to detect multiple classes of objects, e.g., over challenge datasets PASCAL [2] and COCO [3]. Deformable parts model (DPM) is a successful traditional object detection approaches [4]. However, since AlexNet achieved large success in the Imagenet challenge in 2012 [5], CNN quickly becomes the dominant object detection approach.

Despite fast growth of CNN in object detection over datasets with a large number of object classes, real time visual object detection in driving environment is still very challenging. It is observed that the object detection performance of the popular CNN detectors including Faster-RCNN [6] and SSD [7] without modification is not very good over the KITTI benchmark datasets [1]. KITTI is the largest public dataset dedicated to ADAS and autonomous driving benchmarking. In addition to radar and Lidar based object detection, camera based visual object detection, which is the focus of this work, provides an economic solution and is also a critical component of hybrid solution for ADAS and autonomous driving. There are many key challenges on visual object detection for ADAS as discussed below, which may not present in the other object detection datasets.

- Most autonomous driving applications have very high detection accuracy and real time requirements. While high false positive ratio (non-targets are falsely detected as targets) or excessively delayed detections are annoying, which may lead to close of the detection based safety applications, high false negative ratio (targets are not detected) can have fatal consequences and should be avoided as much as possible.
- Driving environment is very harsh for visual object detection with poor illumination and weather conditions. Unlike that there are only a few large target objects in images in datasets such as PASCAL, there can be many occluded and truncated objects with large object scale variations in ADAS images. Example images with occluded and truncated cars are shown in Fig. 1.
- Apart from the accuracy performance requirement, computation speed is also a large concern for ADAS object detection. Vehicles are unlikely to be equipped with GPU computers as powerful as used in research environments. Accuracy often has to be compromised due to the computation complexity of advanced CNN detectors.

In view of the above research challenges, in this paper we propose the following enhancements to multi-scale CNN models to increase the visual object detection accuracy for ADAS.

- In the existing multi-scale CNN models [8], feature map from feature output scales are processed separately to predict existence of objects at fixed scales. In this paper deconvolution of CNN features is applied at smaller feature output scales, which is further fused with features at larger feature output scales, to provide richer context for object detection at individual feature output scale. Such enhancement can effectively address the large object scale variation challenge.

Jian Wei is with Onlyou Artificial Intelligence Institute, China. Email: jwei@onlyou.com. Jianhua He and Zuoyin Tang are with School of Engineering and Applied Science, Aston University, UK. Email: {j.he7,z.tang1}@aston.ac.uk. Kai Chen and Yi Zhou are with Institute of Image Communication and Network Engineering, Shanghai Jiaotong University, China. Email: kchen@sjtu.edu.cn, 744623881@qq.com. Zhiliang Xiong is with Forward Innovation Ltd., China. Email: leslie.xiong@forward-innovation.com. †Corresponding Author: Dr Jianhua He.

(a)

(b)

Fig. 1. Example difficult images for object detection.

- In most of existing CNN detectors, non-maximal suppression (NMS) method is used for suppression of overlapping object proposals. With such process there is very little chance for proper detection of occluded objects. But in driving environments occluded objects are normal and are potential driving hazards. To address the object occlusion challenge soft-NMS is applied at object proposals from different feature output scales to strike a balance on the number and quality of object proposals.
- In the existing CNN detectors, default anchor boxes with certain sizes are used to generate object proposals. In the driving environment the interested objects have strong features in shape, for example, the width of a car should not exceed lane width. The distributions of the object aspect ratio can be utilized for anchor box settings. We measure the aspect ratio statistics of objects from KITTI training samples and find proper anchor box settings by exploiting the statistics for better object localization and prediction.

The proposed CNN enhancements are evaluated with various image input sizes by experiments over KITTI benchmark dataset. Good detection performance improvement is observed with both individual and combined CNN enhancements. Compared to the published works over KITTI benchmark test dataset our proposed method ranks the first for pedestrian detection category "Easy" and second for categories "Moderate and "Hard", and is the fastest among the top ten ranked published methods. The object detection time with a GPU computer is 0.08 second per 384×1280 sized image, which can satisfy the real time requirements of driving safety applications.

The remaining of the paper is organized as follows. Section II presents the related works on object detection with both traditional and deep learning models. Section III presents our proposed methods. Evaluation and experimental results are presented in Section IV. Finally the paper is concluded in Section V.

II. Related Work

Visual object detection is a long term research problem. Classic object detectors use hand-crafted features, such as histogram of oriented gradients (HOG) [9], integral channel features (ICF) [10] and aggregated channel features (ACF) [11]. From the aspect of feature enhancement, [12] introduces spatially pooled features to improve the feature robustness. [13] proposes a pedestrian detector by computing features at multiple image scales. A graph-based algorithm in [14] generates proposals of vehicles with better quality than other traditional region proposal approaches [15], [16]. DPM is the latest successful classic object detector with significantly improved detection accuracy. However the computation complexity of DPM is still very high and its detection accuracy is low for driving object detection.

While classic object detection gets stuck in a bottleneck, there is a large breakthrough on visual object detection with deep learning models, especially CNN models. Powered by GPU computers and huge object detection samples, CNN models can automatically learn complex and efficient features from sample images. Widely successful CNN models and applications have been reported within the past several years. In general CNN based object detectors fall into two frameworks: one-stage and two-stage.

Currently two-stage detectors produce the state-of-the-art performance in object detection tasks like PASCAL, COCO. In the line of two-stage CNN detectors, RCNN [17] is a pioneer CNN model, which increases object detection accuracy over classic detectors by a large margin. In the first stage, RCNN applies selective search method [15] to generate sufficient proposal candidates that contain all the objects. In the second stage, RCNN forwards each proposal through convolutional networks, followed by classifying the proposals with SVMs and predicting bounding boxes offsets with linear regression. Fast-RCNN [18] extends RCNN by using one single convolution network to perform shared computation in the second stage, which increases the speed significantly. Furthermore, Faster-RCNN [6] proposes region proposal network (RPN) to replace selective search method in RCNN and makes the whole network trainable in an end to end approach. In addition, many other variants of RCNN-style approaches are proposed [19]–[21].

On the other hand, one-stage detectors are faster and easier to train while yielding inferior performance. SSD [7] skips the region proposal stage and directly uses multiple feature maps with different resolutions to perform object localization and classification. YOLO [22] is another one-stage detector that can achieve even faster speed at the expense of accuracy. By introducing improvements of batch normalization, high resolution classifier, convolutional with anchor boxes and dimension clusters to YOLO, YOLOv2 [23] achieves higher accuracy and higher speed.

In the latest research on CNN models, there are increasing interests on exploiting multiple scales feature maps. Based on the conventional pyramidal feature hierarchy in convolutional networks in Faster-RCNN, [20] adds a top-down pathway and lateral connections to merge feature maps from different level. The objective is to strengthen the representational power of low-level feature maps with the semantics conveyed from high-level ones. With this adaptation in Faster-RCNN, [20] shows considerable improvements on the COCO detection benchmark. Similar idea is applied to SSD in [24], where DSSD is proposed to utilize feature maps from smaller scales with more semantics. A fully evaluation of DSSD is conducted with different feature map concatenation approaches, including feature maps pooling and deconvolution [25].

The huge success of deep learning and CNN technologies significantly boost research and development of autonomous driving. The popular models are applied and enhanced for object detection in driving environments. However, the popular models including Faster-RCNN, SSD, YOLO, YOLOv2 did not produce good detection accuracy results over the KITTI test dataset. But with certain modifications and adaptations, the variants of Faster-RCNN and SSD models are taking the top entries in the KITTI object detection leader board. For example, [26] improves the region proposal quality with resource to subcategory information. As it is hard for Faster-RCNN to handle the large object size variation, which is designed to detect all the objects on a single layer, MS-CNN [8] extends the detection over multiple scales of feature layers, which produces good detection performance improvement. Scale dependent pooling and cascaded rejection classifiers are used in [27]. In [28], authors propose a recurrent rolling convolution (RRC) architecture on top of SSD model, which produces top detection performance for pedestrian detection. However, the RRC model is very complex and significantly increases computation time.

Our work presented in this paper are different from the above reported enhancements over KITTI benchmark tests. We use MS-CNN as a baseline network model and add three enhancement building blocks, which show considerable object detection performance improvement but with negligible additional object detection time.

III. Network Architectures

In this section we present the overall architecture of the modified CNN model and the proposed enhancements.

A. Overall Architecture of the Modified CNN Model

The input to the CNN is an image with size $H \times W \times D$, where H and W denote image height and width in pixels, and D denotes the number of color components.

The main building blocks of the modified CNN model is presented in Fig. 2. The baseline network is MS-CNN [8], which detects candidate objects at multiple feature output layers with different scales. To differentiate from the MS-CNN, the proposed enhancements are highlighted by red boxes in Fig. 2. The proposed enhancements to MS-CNN are general and are applicable to other CNN models such as Faster-RCNN and SSD as well.

The proposed network follows the popular two-stages object detection network architecture, which consists of an object proposal network and an object detection network. The proposal network layers are based on the popular reduced VGG-16 net [29], which has 16 weight layers in its original form. Additional convolution layers, pooling layer, proposed deconvolution layers and object proposal layers are added on top of the reduced VGG-16 net. Only a few convolution and pooling layers from hidden layers are presented in Fig. 2 for better visualization. The feature outputs of these layers are directly used for object proposal. The layers selected as feature output layers are labeled as “conv4-3”, “conv5-3”, “conv6-1” and “Pool6”, respectively. The first number in the labels such as 4 and 6 represents the associated hidden layer in VGG-16 net, and the second number represents the ID of the convolution layer in a hidden layer. As the feature output layers are not directly connected (with separation by other convolution layers or pooling layers), dotted lines are used to connect them in Fig. 2.

The original feature outputs are further processed by the deconvolution building blocks (DBB), shown as “DB1”, “DB2” and “DB3” in Fig. 2, to aggregate feature maps from adjacent layers, before being used in object proposal building blocks (OPBB). Each OPBB produces a fixed-size set of proposals including coordinates with respect to the pre-defined anchors and scores of objectiveness. Then a soft-NMS building block is used to remove redundant proposals with heavy overlapping. In the original MS-CNN model, NMS is used to remove redundant proposals. The new building blocks (DBB, OPBB and soft-NMS building blocks) will be introduced in details in the following subsections.

The object detection network has a region of interest (ROI) pooling layer and a fully connected (FC) layer. The outputs of upsampled feature maps from the lowest output feature layer (i.e. “conv4-3”) and object proposals from soft-NMS building block in the proposal networks are used as input to the detection networks. The ROI pooling layer extracts the feature maps of the object proposals using these inputs. The feature maps from “conv4-3” are upsampled twice to improve the capacity for location-aware bounding box regression. Then a fully connected layer maps the ROI feature maps into fixed vectors for classification and bounding box regression.

B. Deconvolution Building Block (DBB)

MS-CNN exploits multi-scale features to produce predictions of different scales, which showed improved object detection performance over Faster-CNN and SSD for KITTI datasets. It is a good idea to use the feature maps at larger scales (lower CNN layers) with smaller receptive fields to detect smaller objects and those in smaller scales (higher layers) to detect larger objects. However, shallow feature maps from the low layers of feature pyramid inherently lack fine semantic information for object recognition. There is an opportunity to augment the shallow feature maps with deeper feature maps from higher feature output layers and improve detection performance.

We propose to add DBB to the baseline MS-CNN model, with additional deconvolution layers and lateral connections to aggregate feature outputs from different layers. Using DBBs the semantics from higher layers can be conveyed into lower layers to increase the representation capacity. There are three DBBs used in the proposed CNN model. Fig. 3 illustrates the architecture of the DBB used in this paper, which connects one feature output layer with its adjacent higher layer counterpart. Specifically, we first connect a convolution layer (“Conv

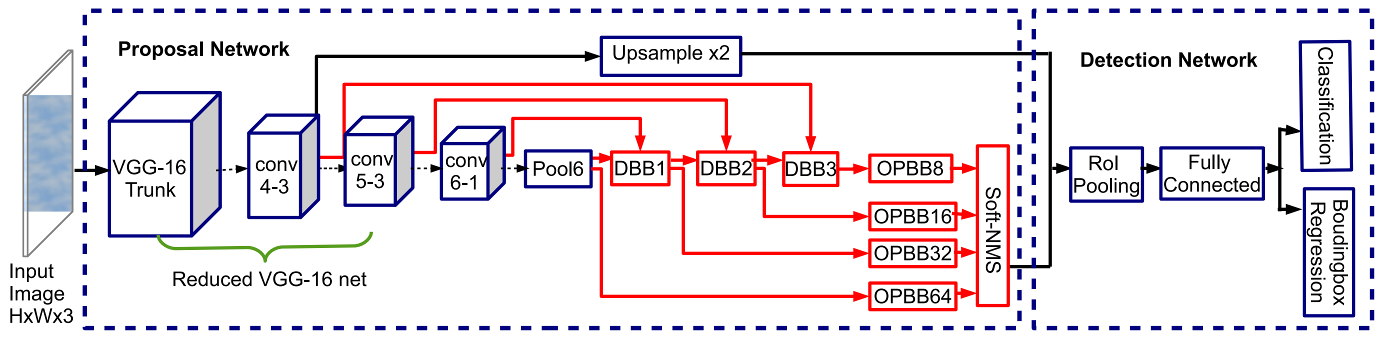


Fig. 2. Overall pipeline of enhanced MS-CNN model.

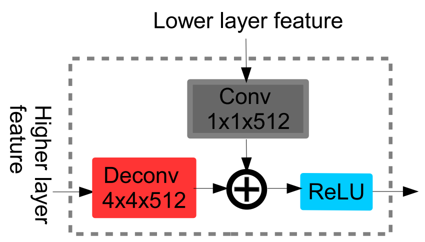


Fig. 3. Feature fusion method for deconvolution building block (DBB).

$1\times1\times512$") with 512 1×1 filters to an output feature layer as shown in the Fig. 3. In addition, in the horizontal direction, a deconvolution layer ("Deconv $4\times4\times512$") with 512 4×4 filters is applied to upsample the corresponding higher-level feature maps. Then the outputs of these two associated feature layers, which have the same spatial size and depth, are merged by element-wise sum and processed by a ReLU layer to produce a new output feature layer. In order to maintain feature aggregation consistence, the number of channels is set to 512 in all DBBs.

There are many possible architecture designs for DBBs. For example, for a given feature output layer, the output feature maps can be merged with those from both higher layers and lower layers. However the computation complexity and memory requirement can be increased significantly. We examined and compared several alternative DBB architectures, some using element-wise multiplication or concatenation instead of element-wise sum used in this paper, and some adding a batch normalization (BN) function block after the convolution and deconvolution layers in the DBB as shown in Fig. 3. However, according to results from extensive experiments, it is found that the implementation shown in Fig. 3 has the best detection performance and low computation complexity. The results demonstrated that the design of DBB is not straightforward and specific consideration should be taken for different baseline CNN models.

C. Object Proposal Building Block (OPBB)

1) OPBB Architecture: The functionality of OPBB is to receive feature map output from the DBBs or Pool6 layer and produce high quality proposals to be further processed by the soft-NMS building block. In this paper we have 4 OPBBs which have the same architecture but different parameters. These OPBBs are labeled as "OPBB8", "OPBB16", "OPBB32" and "OPBB64" as shown in Fig. 2. The number in the OPBB labels is the ratio of the original image size to the spatial size of the feature map input to the OPBBs.

Inside each OPBB there are several similar process pipelines, each associated with one type of anchors. The overall architecture of an OPBB with two types of anchors is shown in Fig. 4. For the first anchor related pipeline, the input feature maps from DBB go through two separate processes: one for classification having a convolution layer with $h1 \times w1 \times (C + 1)$ filters and a softmax module, and the other for bounding box regression with respect to the anchor having a convolution layer with $h1 \times w1 \times 4$ filters. The classification process path produces the softmax scores of C object classes and background class for each feature map location. The regression path produces a bounding box estimation for each feature map location. Then the anchors with estimated classification scores and the bounding box for each feature map location are processed to form good quality proposals.

At each feature map location l, there are two proposals, p_l^n for $n \in \{1, 2\}$, produced from the two anchor pipelines. Each proposal has $(4 + C + 1)$ dimensions, among which 4 dimensions are for bounding box coordinates and C+1 dimensions are for classification scores of each class. The 4 coordinates represent the offsets relative to the associated anchor coordinates. Let B_l^n denote the coordinates vector for proposal p_l^n, $n \in \{1, 2\}$. Let L denote the class label set, $L = \{0, 1, 2, ..., C\}$. Label 0 refers to the background class. Let $F_l^n = (f_l^{n,0}, f_l^{n,1}, ..., f_l^{n,C})$ be the classification score vector for p_l^n, where $n \in \{1, 2\}$, $f_l^{n,c}$ denotes the classification score for class c. Classification score measures the probability distribution over $C+1$ classes. Then a proposal p_l^n at location l can be denoted by p_l^n = (B_l^n, F_l^n).

2) Anchor Boxes: Anchor boxes are critical component of the regional proposal networks for Faster-RCNN model and its variants such as MS-CNN. In the standard Faster-RCNN model there are 9 types of anchor boxes associated to one convolutional filter layer. In the baseline MS-CNN, in each OPBB, there are several convolutional filter layers and each is associated with only one type of anchor boxes. The associated convolutional filter layer and the type of anchor box correspond to one proposal pipeline in an OPBB. The

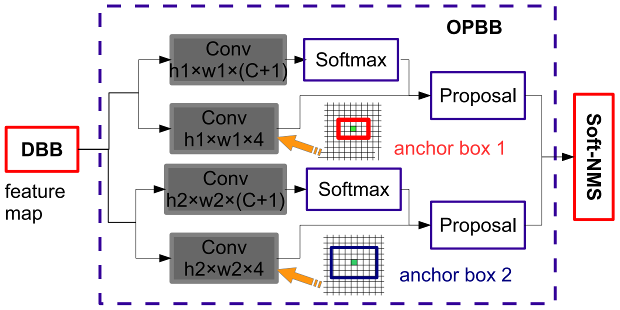


Fig. 4. Object proposal building block.

aspect ratio of MS-CNN anchor boxes is set to 1 for cars and around 0.7 for pedestrians. Although the network can refine the bounding box of proposals by learning to predict the offsets to anchor boxes, a better anchor box setting will help object detection with improved matching to the ground truth bounding boxes, therefore improve both training and inference performance.

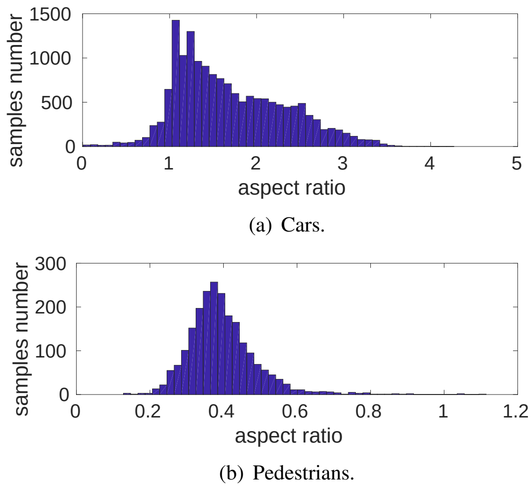


Fig. 5. Distribution of aspect ratios for different object classes in KITTI benchmark training set.

We develop a refined OPBB with better anchor box settings according to the object statistics analyzed over the KITTI training set. Although the experiments of the anchor box setting are conducted with MS-CNN as the baseline CNN model, the idea of setting anchor boxes with sample statistics is general and can be applied to other network architectures as well.

To get insights into better anchor box settings, we collect all the ground truth bounding boxes in the KITTI training set and generate a histogram of object aspect ratios for cars and pedestrians. As shown in Fig. 5, the objects of different classes have distinct distributions of aspect ratios. Car samples have wider boxes with most aspect ratio values in the range of 1 to 3. On the contrary, pedestrians have much smaller aspect ratios. Based on the observation, we resize the square anchor boxes for cars used in MS-CNN to rectangle ones, which are closer to the average aspect ratio of car samples. In addition, for pedestrian objects detection we add one more type of anchor box. So there are three types of anchor boxes in total for an OPBB in the refined OPBB architecture, compared to only two types of anchor boxes used in the baseline MS-CNN. The additional type of anchor box has an aspect ratio 0.5. The original anchor ratio set in the baseline MS-CNN is too narrow to efficiently cover the object variations. More details of anchor configurations are given in Section IV. Our new anchor settings are by no means the best fitting to the KITTI dataset. There may be optimal joint settings on the number, scale and aspect ratio of anchor boxes.

D. Soft-NMS Building Block

After the object proposal layers, soft-NMS building block is used to filter out highly overlapped proposals from the object proposal layers. As NMS algorithm has been applied to remove redundant neighbor proposals in most state-of-the-art object detection CNN models including MS-CNN, we have a brief introduction to NMS before the presentation of soft-NMS. For a proposal p, any other proposal that has an overlap more than a pre-defined threshold T with proposal p is called a neighbor proposal of proposal p. Mathematically, let $P_{\text{in}} = \{p_1, p_2, ...p_n\}$ denote an initial proposal set output from the object proposal layers, in which the proposals are sorted by their objectiveness scores. Here the objectiveness score S_i for proposal p_i is the maximum value in the classification score vector of p_i. The traditional NMS method works as follows:

Algorithm 1 NMS.

```
Input: Proposal set $P_{\text{in}}$
Output: Proposal set $P_{\text{out}}$, which is initialized to an empty set
1: Create a temporary proposal set $P_{\text{temp}}$, which is initialized to $P_{\text{in}}$.
2: Check if any proposal remains in proposal set $P_{\text{temp}}$.
3: If yes, go to Step 4; else, terminate the NMS process and return output $P_{\text{out}}$.
4: Move the first proposal (with the highest objectiveness score) in $P_{\text{temp}}$ to $P_{\text{out}}$, which is called winning proposal, denoted by $p_{\text{win}}$.
5: Update set $P_{\text{temp}}$ by removing all the neighbor proposals of proposal $p_{\text{win}}$ from set $P_{\text{temp}}$.
6: Go to Step 2.
```

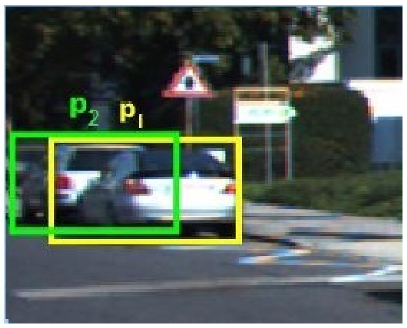


Fig. 6. Example of overlapped proposals.

In many object detection challenge datasets neighbor proposals usually correspond to the same object. But due to heavy object occlusion in KITTI dataset, NMS may remove positive proposals unexpectedly. For example, there are two proposals p_1 and p_2 from an image with large overlap in Fig. 6. The proposal p_2 for the occluded back car may be removed with high probability by the traditional NMS method. To address the NMS issue with occluded objects, we apply soft-NMS for suppression of overlapped objects [30]. With soft-NMS the neighbor proposals of a winning proposal are not completely suppressed. Instead they are suppressed according to updated objectiveness scores of the neighbor proposals, which are computed according to the overlap level of the neighbor proposals and the winning proposal. NMS can be viewed as a specific case of soft-NMS, in which the updated objectiveness scores of the neighbor proposals of a winning proposal are simply set to zero.

Let p_i be a winning proposal and p_j be a neighbor proposal of p_i. Let S_j be the objectiveness score of p_j computed from object proposal layers. The updated objectiveness score of p_j (denoted by S_j^{u}) is computed with a linear function by the following formula (1) [30]:

$$S_j^{\text{u}} = S_j(1 - O_{p_i,p_j}), \tag{1}$$

where O_{p_i,p_j} represents the intersection of union (IoU) between p_i and p_j. O_{p_i,p_j} is computed by the following formula:

$$O_{p_i,p_j} = \frac{area(p_i \cap p_j)}{area(p_i \cup p_j)}. \tag{2}$$

As a whole, the term $1 - O_{p_i,p_j}$ acts as a weighting function with higher overlap leading to larger penalty to objectiveness score for neighbor proposals.

The operation of soft-NMS method is presented below.

Algorithm 2 Soft-NMS.

```
Input: Proposal set $P_{\text{in}}$
Output: Proposal set $P_{\text{out}}$
1: Create a temporary proposal set $P_{\text{temp}}$, which is initialized to $P_{\text{in}}$.
2: Check if any proposal remains in proposal set $P_{\text{temp}}$.
3: If yes, go to Step 4; else, terminate the NMS process and return output $P_{\text{out}}$.
4: Move the winning proposal $p_{\text{win}}$ in $P_{\text{temp}}$ in this round to $P_{\text{out}}$.
5: Compute the updated score of the neighbor proposals of proposal $p_{\text{win}}$ in $P_{\text{temp}}$ according to (1).
6: Update set $P_{\text{temp}}$ by removing the neighbor proposals of $p_{\text{win}}$ if their updated scores are lower than a pre-defined threshold $T_{\text{s}}$.
7: Go to Step 2.
```

In this paper, the neighbor proposal threshold T is set to 0.4 and the score updating threshold T_{s} is set to 0.001 for soft-NMS method by cross-validation.

E. Training and Inference

The whole network training includes two phases. Firstly, train the object proposal network with object proposal training samples. Secondly, train both the object proposal network and the object detection network. For both phases of network training, training samples with object classes and bounding boxes are needed. Next we introduce the construction of training samples, then present the loss function to be used for network training.

1) Training Samples: The class and bounding box of a proposal with regard to an anchor for a feature map location is mainly determined by the convolution layers in the OPBB. However, their weights are learned from training process with ground truth samples and configured anchors. Without loss of generality, let A_l denote an anchor with a given scale and aspect ratio from one type of anchor boxes centered at a feature map location l. The coordinates of the anchor includes its center (x_l, y_l), anchor width (w_l) and height (h_l). To create a training sample for this anchor, we first find the best matching ground truth box for it based on their IoU overlap. Let gt_l denote the best matching ground truth box for anchor A_l, and O_{A_l,gt_l} be the IoU overlap between anchor A_l and ground truth box gt_l. Then class label (denoted by c_l) for this anchor can be determined according to the IoU with the matched ground truth box. If O_{A_l,gt_l} is higher than 0.5, the anchor A_l is assigned a class label c_{gt_l}, which is the class label of the matched ground truth object. If O_{A_l,gt_l} is lower than 0.2, the anchor A_l is labeled as 0 (i.e., background class). Otherwise the anchor is assigned a class value of -1. The class label determination can be expressed in the following formula:

$$c_l = \begin{cases} c_{gt_l} & O_{A_l,gt_l} > 0.5 \\ 0 & O_{A_l,gt_l} < 0.2 \\ -1 & otherwise \end{cases}, \tag{3}$$

Note that anchors that labeled -1 will be discarded and are not used as training samples. The regression of the bounding box can be obtained from the anchor coordinates and the ground truth bounding box in a similar way presented in [6].

2) Training Loss Function: After the training samples are prepared, the network can be trained with properly designed loss function. In this paper the objective loss function is to minimize the weighted sum of localization loss L_{loc} and classification loss L_{cls} for the proposal and detection networks [8]:

$$\min \left[\sum_{l,c_l \geq 1} \lambda L_{loc}(loc_l, loc_{gt_l}) + \sum_{l} L_{cls}(F_l, c_l) \right], \tag{4}$$

$$L_{loc}(loc_l, loc_{gt_l}) = 0.25 * smooth_{L1}(loc_l - loc_{gt_l}), \tag{5}$$

$$L_{cls}(F_l, c_l) = -log(f_l^{c_l}), \tag{6}$$

$$smooth_{L1}(x) = \begin{cases} 0.5x^2 & if|x| < 1 \\ |x| - 0.5 & otherwise \end{cases}, \tag{7}$$

where l is the index of an anchor in the set of training samples, λ denotes a weight term, $F_l = (f_l^0, f_l^1, ..., f_l^C)$ is the classification score vector for proposal p_l, c_l is the anchor label class, loc_l is the bounding box coordinates for p_l, loc_{gt_l} is the coordinates of matched ground truth box. With the above objective function, the network can be trained by standard back-propagation and stochastic gradient descent strategies.

During inference process, a feed-forward pass of the network is run on the test images. The proposal network generates proposal candidates with bounding boxes and classification confidences and detection network further refines the location and class scores for proposals processed by soft-NMS.

IV. Experiments

A. Dataset

We evaluate the enhanced CNN model over the KITTI 2D object detection benchmark dataset. The dataset contains 14999 images with 7481 for training and 7518 for testing. The image size is 384×1280 pixels. There are over 80000 annotated objects, which are divided into three categories (car, pedestrian and cyclist). Three object detection evaluation categories ("Easy", "Moderate" and "Hard") are set up for each object class, according to object height, occlusion and truncation level, which are presented in Table I. For evaluation, average precision (AP) with different IoU thresholds (0.7 for car, 0.5 for pedestrian and cyclist) is used as the main metric of interest. The AP is computed as the mean precision at a set of equally spaced recall levels [2].

TABLE I
Three object difficulty levels for KITTI dataset.

Levels	Description		
	Min. height	Max. occlusion level	Max. truncation
Easy	40 pixels	Fully visible	15%
Moderate	25 pixels	Partly occluded	30%
Hard	25 pixels	Difficult to see	50%

B. Implementation Details

As a widely adopted practice, the proposed network is fine-tuned on the reduced VGG-16 model, which is pre-trained on the ILSVRC CLS-LOC dataset [31]. We split the raw training dataset into training set and validation set for local performance evaluation.

As the number of samples for different object classes are highly imbalanced, detectors are trained separately for detection of cars and pedestrians. The training procedure consists of two stages. In the first stage, only the proposal network is trained by 10000 iterations, with weight term λ of 0.05, initial learning rate of 0.00005, momentum of 0.9, weight decay of 0.0005. Following the proposal network training, in the second stage the whole network (including both proposal network and detection network) is trained for another 25000 iterations. The learning rate for the second stage is initially set to 0.0005 and is divided by 10 every 10000 iterations. The weight term λ is 1. The experiments are run with an Intel i7-7700k 4.20GHz server with 8 CPU cores and 32 GB memory and a Nvidia GeForce GTX 1080 GPU. Training time ranges from 6 to 10 hours for the models used in this paper.

In order to examine the effectiveness of the proposed network enhancements, ablation experiments are designed and conducted. We let letters "D", "AR" and "S" denote the proposed network enhancements on deconvolution, anchor box resize and soft-NMS, respectively. The baseline MS-CNN network is denoted by letter "M". In the ablation experiments various enhancements are added on top of the baseline MS-CNN network. The network variants with baseline network and different network enhancements are denoted by "M+D", "M+AR", "M+S", "M+AR+S" and "M+D+AR+S", respectively. As there are much smaller number of cyclist samples compared to those for car and pedestrian in the dataset, only car and pedestrian evaluation results are presented.

In addition to the various network enhancements, input layer image size impact is also investigated. We train the network with 3 input image sizes, small image 384×1280 (the original image size), medium image 576×1920 and large image 768×2560. The enlargement of images does not increase image resolution. The experiments carried out with different input image size are denoted by the object class and the input image height. For example, experiments for car detection with image size 384×1280 are denoted by "Car-384". Anchor sizes are set differently for different types of experiments. The anchor and associated filter size configurations for different image sizes and different object classes are shown in Table II. Note that the other parameters are kept unchanged through all the experiments.

C. Experimental Results on Validation Set

In this subsection we examine and compare the performance of the proposed CNN enhancements for object detection over KITTI benchmark dataset. As the ground truth of the KITTI test set is not publicised and only one submission of the KITTI test results to the benchmark website is allowed, performance comparison of the proposed enhancements is performed over the KITTI training and validation set.

The AP results of the compared CNN models as configured in the previous subsection are reported in Table III for both car and pedestrian detection. The CNN models include the original MS-CNN with and without the proposed enhancements. The AP results for the detection categories "Easy", "Moderate" and "Hard" are presented in Table III(a), III(b) and III(c), respectively. In the tables the maximal AP values from the compared CNN models for each image size are displayed in bold font. As MS-CNN training with deconvolution building block and image size 768×2560 was not completed due to high GPU memory requirement, the results of related CNN models with DBB enhancement ("M+D+$*$") are not presented for large input image size.

In addition the network inference time per image is reported in Table IV. The inference speed of the original MS-CNN and the proposed CNN networks are very fast (0.08 second per image for small image size). The introduction of anchor box resize ("AR") and soft-NMS ("S") add negligible time. The deconvolution building block introduce a little extra computation time (0.01 second per image).

1) The effectiveness of proposed enhancements: First we check the effectiveness of the individual proposed enhancements. Comparing the results of CNN variants "M+D", "M+AR" and "M+S" to the baseline MS-CNN model "M", it can be observed that there are good performance improvement for most input image sizes and object classes. Among the individual enhancements, soft-NMS produces the largest and consistent performance gain for both car and pedestrian detection in most cases. The performance improvement with

TABLE II
CONFIGURATIONS OF ANCHOR SIZE AND FILTER SIZE (WIDTH×HEIGHT) WITH DIFFERENT IMAGE SIZE.

(a) car.

		OPBB-8		OPBB-16		OPBB-32		OPBB-64	
Car-384	anchor	40×24	56×36	80×48	112×72	160×96	224×144	320×192	
	filter	5×5	7×7	5×5	7×7	5×5	7×7	5×5	
Car-576	anchor	60×40	84×54	120×80	168×108	240×160	336×216	480×320	
	filter	5×5	7×7	5×5	7×7	5×5	7×7	5×5	
Car-768	anchor	60×40	84×54	120×80	168×108	240×160	336×216	480×320	672×432
	filter	5×5	7×7	5×5	7×7	5×5	7×7	5×5	7×7

(b) pedestrian.

		OPBB-8			OPBB-16			OPBB-32			OPBB-64	
Ped	anchor	28×40	28×56	36×56	56×80	56×112	72×112	112×160	112×224	144×224	224×320	
-384	filter	3×5	3×7	5×7	3×5	3×7	5×7	3×5	3×7	5×7	3×5	
Ped	anchor	40×60	40×84	56×84	80×120	80×168	112×168	160×240	160×336	224×336	320×480	
-576	filter	3×5	3×7	5×7	3×5	3×7	5×7	3×5	3×7	5×7	3×5	
Ped	anchor	40×60	40×84	56×84	80×120	80×168	112×168	160×240	160×336	224×336	320×480	448×672
-768	filter	3×5	3×7	5×7	3×5	3×7	5×7	3×5	3×7	5×7	3×5	5×7

TABLE III
PERFORMANCE COMPARISON OF CNN VARIANTS ON VALIDATION SET.

(a) Easy.

	Car			Pedestrian		
Image height	384	576	768	384	576	768
M	89.34	90.62	91.12	76.25	79.72	80.02
M+D	90.96	92.39	-	77.93	80.25	-
M+AR	94.44	90.47	91.38	77.97	79.92	80.25
M+S	91.77	91.09	91.50	78.96	79.82	80.41
M+AR+S	**94.78**	92.72	**91.68**	**80.28**	79.98	**80.58**
M+D+AR+S	93.76	**93.12**	-	78.50	**80.28**	-

(b) Moderate.

	Car			Pedestrian		
Image height	384	576	768	384	576	768
M	88.84	89.86	90.04	70.57	74.68	76.49
M+D	89.00	89.74	-	71.39	75.92	-
M+AR	89.36	89.88	90.08	71.63	75.59	76.38
M+S	89.44	89.99	90.29	73.04	75.07	76.64
M+AR+S	**89.57**	90.20	**90.35**	**73.05**	75.85	**76.93**
M+D+AR+S	89.37	**90.23**	-	72.42	**76.69**	-

(c) Hard.

	Car			Pedestrian		
Image height	384	576	768	384	576	768
M	77.59	79.04	79.86	62.58	66.55	68.02
M+D	77.22	78.80	-	63.53	68.06	-
M+AR	77.86	79.50	79.83	63.41	66.85	68.02
M+S	77.16	79.50	80.31	64.70	66.74	68.03
M+AR+S	**78.40**	80.04	**80.39**	**64.88**	66.92	**68.41**
M+D+AR+S	78.23	**80.33**	-	64.15	**68.25**	-

TABLE IV
AVERAGE INFERENCE TIME FOR VARIOUS NETWORK ARCHITECTURES.

	Car			Pedestrian		
Image height	384	576	768	384	576	768
M	0.08s	0.17s	0.24s	0.06s	0.14s	0.20s
M+AR+S	0.08s	0.17s	0.24s	0.06s	0.14s	0.20s
M+D+AR+S	0.09s	0.18s	-	0.07s	0.15s	-

soft-NMS is more obvious for pedestrian detection with image size 384×1280. For example, the AP with soft-NMS increases from 76.25% for "M" to 78.96% for pedestrian detection category "Easy" with small image size. These results demonstrate the effectiveness of soft-NMS on tackling the object occlusion issues in ADAS environments. Anchor resize ("AR") enhancement shows consistent performance gain over the baseline network as well. But the largest performance gain with "AR" comes mainly with the small image size, e.g., 5.1% performance gain with "AR" for car detection category "Easy". On the other hand, deconvolution ("D") enhancement shows consistent performance gain for pedestrian detection and large performance gain for car detection category "Easy" with medium image size, but there is a slight performance loss for car category "Hard".

Next combinations of the proposed enhancements are examined. The best AP performance is always achieved with combined network enhancements for all object classes, object detection categories and input image sizes. For example, for medium image size, the best network for both car and pedestrian detection is "M+D+AR+S" for category "Easy", "Moderate" and "Hard". These results show that the proposed enhancements can work together and effectively boost object detection performance.

An interesting observation is on the experiment results with combination of "AR" and "S" enhancements. For both car and pedestrian detection with small image size, both anchor resize and soft-NMS enhancements bring performance gains: anchor resize has much larger gains for car detection, while soft-NMS has larger gains for pedestrian detection. The combination of "AR" and "S" enhancements have consistent and larger gains than the individual enhancement.

2) The impact of input image size: Apart from the proposed network enhancement, it is also observed that increasing image size has a large positive impact on object detection. For any given studied MS-CNN variant, the AP performance improves with larger image size in most studied cases. There is a substantial performance gain with image size for the baseline network "M", especially for pedestrian detection. For example the AP increases from 70.57% with small image size to 76.49% with large image size.

However the performance gains with larger image size for some MS-CNN variants (such as "M+AR+S" and "M+D+AR+S") are much smaller. For the baseline MS-CNN network, the largest AP for car detection category "Easy" is 91.12% with large image size. But the enhanced network "M+AR+S" has 94.78% AP with small image size.

It is worth noting that the performance gains with large image size do not come without cost. According to Table IV, the average inference time per image for car detection increases from 0.08 second for small image size to 0.17 second for medium image size and 0.24 second for large image size. Similar inference time performance for pedestrian detection is observed. As the best detection performance with "M+D+AR+S" with medium image size is already very close to or even better than the best available performance with large image size, "M+D+AR+S" network model with medium image size is recommended for joint considerations on detection precision and speed. More specifically, the "M+AR+S" network architecture with small image size offers the highest speed and best detection AP (94.78% versus 91.68% with large image) for car detection category "Easy" and slightly lower AP (80.28% versus 80.58% for large image) for pedestrian. For some driving safety assistance applications with targets of detecting easy objects, such as forward collision warning, the "M+AR+S" network architecture with small image size can be the first choice.

To visually assess the effectiveness of the proposed method, some example KITTI images with annotations of detected objects by the baseline MS-CNN model (shown in the left column) and our method (shown in the right column) are presented in Fig. 7. To be fair, the image size is set 768×2560 for both models. The first three rows Fig. 7(a)-Fig. 7(c) are for car detection and the last row Fig. 7(d) is for pedestrian detection. Compared the detection results with MS-CNN and our method, we can find that our method improves the detection performance from several aspects:

- Our method can reduce false proposals as shown in Fig. 7(a) and in Fig. 7(b). In the left image of Fig. 7(a), there are two false proposals produced by MS-CNN around the orange car in the bottom left side. In Fig. 7(b) the MS-CNN method produce two false proposals, one in the right cluster of cars and one in the left cluster of cars.
- Our method can detect more small objects that are missed by MS-CNN as shown in Fig. 7(c). The MS-CNN method missed the remote small car on the road and a car in right shadow area.
- Our method can avoid producing multiple bounding boxes for one object. For example in Fig. 7(d), the MS-CNN model produces two bounding boxes for each detected pedestrian.

D. Experimental Results on KITTI Test Set

Next we present the experiment results over the KITTI test set and compare our results with those of recently published approaches.

As the KITTI leader board ranks the approaches based on the AP for "Moderate" detection category, we select the network "M+AR+S" with large image size (768×2560) for competition, which produced the best AP for "Moderate" category over validation set. The results are submitted to the KITTI test set evaluation server.

The AP and inference time results of our proposed method and other top ranked published approaches are presented in Table V. While the original CNN models (Faster-RCNN, SSD and YOLOv2) without adaption to the KITTI datasets have much lower object detection performance over KITTI test set, they are also listed in Table V for information.

A simple comparison of our own results on KITTI test data set to those on validation test shows that there are considerable performance loss possibly due to harder images in the test set. However similar performance loss was observed for the baseline MS-CNN model.

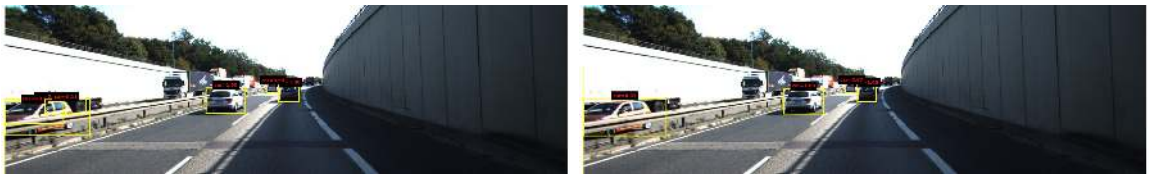
Comparing the AP and the inference time results in Table V, it can be concluded that there is no absolute winner with dominant performance over all the comparison aspects. Among the compared leading approaches, our proposed method ranked the first in network inference speed, the best in the pedestrian category "Easy", second in pedestrian categories "Moderate" and "Hard", third in car detection category "Moderate". D_MANTA [35] ranked the first in car category "Easy". RRC [28] has four number one positions in all detection categories. However, RRC has the second longest inference time (3.6 second), which is 15 times our inference time, even it is based on the fast SSD baseline and used much higher specification GPU computer.

The highest AP for car category "Easy" achieved by "M+AR+S" with small image is 94.78% over the validation set, while the highest AP over the test set is only 90.49%. One reason for the performance gap is that "M+AR+S" with large image size is selected as the only model for competition. Therefore the good performance with "M+AR+S" model and small image size is compromised.

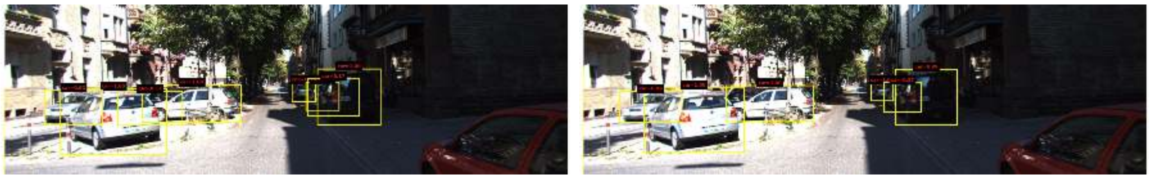
TABLE V
PERFORMANCE COMPARISON OF RECENT PUBLISHED WORKS AND OUR METHOD ON THE TEST SET.

Method	Pedestrian			Car			Time (s)
	Easy	Mod	Hard	Easy	Mod	Hard	
Faster-RCNN [6]	78.35	65.91	61.19	87.90	79.11	79.19	2
SSD [7]	23.14	16.30	16.06	83.89	67.17	59.09	0.06
YOLOv2 [23]	20.80	16.19	15.43	28.37	19.31	15.94	0.02
spLBP [12]	-	-	-	80.16	77.39	60.59	1.5
Mono3D [32]	77.30	66.66	63.44	90.27	87.86	78.09	4.2
MS-CNN [8]	83.70	73.62	68.28	90.46	88.83	74.76	0.4
Deep3D [33]	-	-	-	90.47	88.86	77.60	1.5
SubCNN [26]	83.17	71.34	66.36	90.75	88.86	79.24	2.0
MV3D [34]	-	-	-	90.53	89.17	80.16	0.36
SDP+RPN [27]	79.98	70.20	64.84	89.90	89.42	78.54	0.4
D_MANTA [35]	-	-	-	**97.25**	90.03	80.62	0.7
RRC [28]	84.14	**75.33**	**70.39**	90.61	**90.22**	**87.44**	3.6
Our method	**85.12**	74.52	69.35	90.49	89.64	77.95	**0.24**

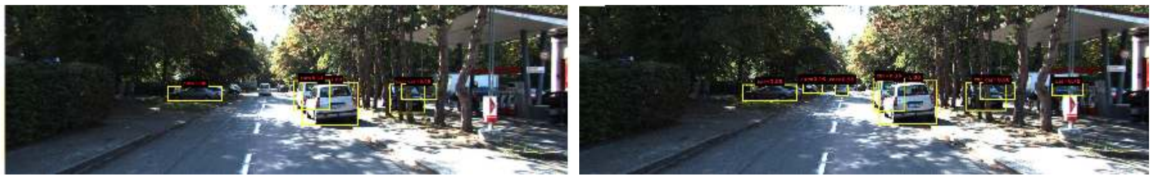
According to the object detection results presented in Table V and in KITTI benchmark website, it can be observed that the car detection performance for category "Moderate" is almost saturated with very little performance gap over the top 20 detection methods. However, there is still large performance improvement space for pedestrian and cyclist detection. For example the highest AP from the published works is 85.12% and 75.33% for pedestrian category "Easy" and "Moderate", respectively.



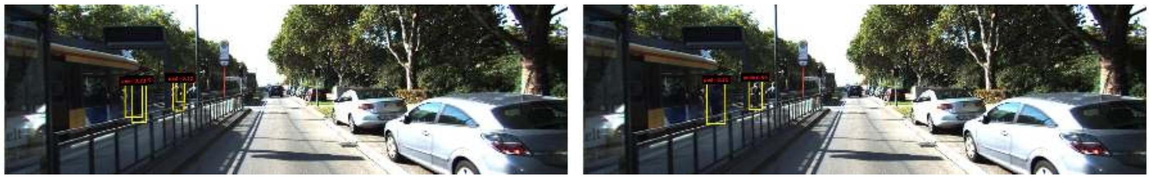
(a)



(b)



(c)



(d)

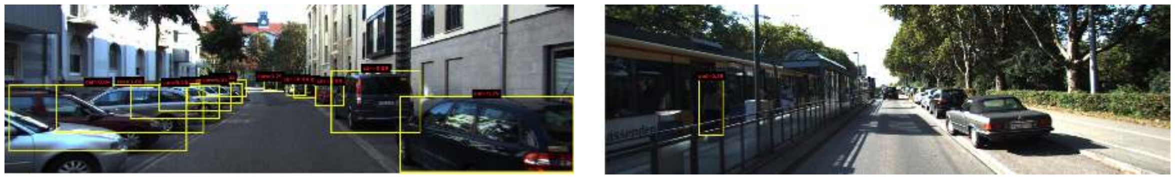
Fig. 7. Object detection examples on KITTI testing set with MS-CNN and our method.

The main challenges of the pedestrian and cyclist detection still come from the small size, heavy occlusion or truncation of the objects. In addition other external factors like illumination change and cluttered background can affect the accuracy of our detection method. And compared to the number of car samples in the KITTI dataset, the number of pedestrian and cyclist samples are much smaller, which may be another cause of the relatively poor detection performance for pedestrian detection.

We present two example images in which some samples are not correctly detected by our method in Fig. 8. These detection examples may help understand the existing detection challenges. In Fig. 8(a) the white car in the bottom left side is not detected due to heavy truncation. In Fig. 8(b) one person near the train is not detected due to occlusion and poor illumination conditions.

V. Conclusion

Real time accurate object detection is one of the most critical problems for advanced driving assistance systems (ADAS) and autonomous driving. Recently convolutional neural networks (CNN) achieved huge successes on visual object detection over traditional object detectors, which use hand-engineered features. However, due to the challenging driving environment (e.g., large object scale variation, object occlusion and bad light conditions), popular CNN detectors including Faster-RCNN and SSD do not produce good detection performance over the KITTI driving benchmark dataset. In this paper we proposed three enhancements on a multiple scale CNN network model for ADAS object detection. Firstly, CNN feature maps deconvolution and fusion was proposed to add context and deeper features for better object detection at lower scale of feature maps, to address the large object scale variation challenge. Then, soft non-maximal suppression (NMS) was applied across object proposals at different image scales to address the object occlusion challenge. As the cars and pedestrians have distinct aspect ratio features, we measured their aspect ratio statistics and exploited them to set anchor boxes properly for better object matching and localization. The proposed CNN enhancements with various input image sizes were individually and jointly evaluated by extensive experiments over KITTI dataset. The effectiveness of the proposed enhancements was verified by experiment results with improved or comparable detection performance over KITTI test set. The average precision (AP) for pedestrian detection category “Easy” and the computation speed rank the first among the published works, the second for pedestrian category “Moderate” and “Hard”, the third for car category “Moderate”. And the network inference time for cars per 384×1280 image is only 0.08 second, much faster than the other top ranked published methods in KITTI leader board. In our future works we will investigate more CNN models and enhancements to improve object detection for safe and intelligent transport.



(a) Undetected car due to heavy truncation

(b) Undetected pedestrians due to occlusion and poor illumination conditions

Fig. 8. Example images from KITTI testing set with false object detection by our method.

Acknowledgment

The authors would like to thank the reviewers for their constructive comments. This project has received funding from the European Unions Horizon 2020 research and innovation programme under the Marie Skodowska-Curie grant agreement No 824019 and the FP7 grant DETERMINE under the FP7-PEOPLE-2012-IRSES grant agreement No 318906.

References

[1] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 3354–3361, IEEE, 2012.

[2] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.

[3] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*, pp. 740–755, Springer, 2014.

[4] P. Felzenszwalb, R. B. Girshick, and D. McAllester, "Cascade object detection with deformable part models," In *CVPR*, pp. 2241-2248, 2010.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," In *NIPS*, pp. 1097-1105, 2012.

[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, pp. 91–99, 2015.

[7] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*, pp. 21–37, Springer, 2016.

[8] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos, "A unified multi-scale deep convolutional neural network for fast object detection," in *European Conference on Computer Vision*, pp. 354–370, Springer, 2016.

[9] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1, pp. 886–893, IEEE, 2005.

[10] P. Dollár, Z. Tu, P. Perona, and S. Belongie, "Integral channel features," 2009.

[11] P. Dollár, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 8, pp. 1532–1545, 2014.

[12] Q. Hu, S. Paisitkriangkrai, C. Shen, A. van den Hengel, and F. Porikli, "Fast detection of multiple objects in traffic scenes with a common detection framework," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1002–1014, 2016.

[13] R. N. Rajaram, E. Ohn-Bar, and M. M. Trivedi, "Looking at pedestrians at different scales: A multiresolution approach and evaluations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 12, pp. 3565–3576, 2016.

[14] X. Yuan, S. Su, and H. Chen, "A graph-based vehicle proposal location and detection algorithm," *IEEE Transactions on Intelligent Transportation Systems*, 2017.

[15] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

[16] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *European Conference on Computer Vision*, pp. 391–405, Springer, 2014.

[17] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

[18] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

[19] 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*, pp. 379–387, 2016.

[20] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *CVPR*, 2017.

[21] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *International conference on computer vision*, 2017.

[22] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788, 2016.

[23] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[24] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "Dssd: Deconvolutional single shot detector," *arXiv preprint arXiv:1701.06659*, 2017.

[25] J. Jeong, H. Park, and N. Kwak, "Enhancement of ssd by concatenating feature maps for object detection," *arXiv preprint arXiv:1705.09587*, 2017.

[26] Y. Xiang, W. Choi, Y. Lin, and S. Savarese, "Subcategory-aware convolutional neural networks for object proposals and detection," in *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*, pp. 924–933, IEEE, 2017.

[27] F. Yang, W. Choi, and Y. Lin, "Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers," in *CVPR*, 2016.

[28] J. Ren, X. Chen, J. Liu, W. Sun, J. Pang, Q. Yan, Y.-W. Tai, and L. Xu, "Accurate single stage detector using recurrent rolling convolution," in *CVPR*, 2017.

[29] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *NIPS*, 2015.

[30] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis, "Improving object detection with one line of code," *arXiv preprint arXiv:1704.04503*, 2017.

[31] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[32] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun, "Monocular 3d object detection for autonomous driving," in *CVPR*, 2016.

[33] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka, "3d bounding box estimation using deep learning and geometry," in *CVPR*, 2017.

[34] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *CVPR*, 2017.

[35] F. Chabot, M. Chaouch, J. Rabarisoa, C. Teulire, and T. Chateau, "Deep manta: A coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image," in *CVPR*, 2017.

Jian Wei received the B.E. degree in electronic engineering from Shanghai Jiao Tong University, Shanghai, China, in 2013 and the Ph.D. degree in Engineering and Applied Science from Aston University, UK, in 2017. He is currently a researcher with the Onlyou Artificial Intelligence Institute, China. His current research interests include machine learning and data mining, object detection and recognition, and natural language processing.

Jianhua He received a BEng degree and MEng in Electronics Engineering from Huazhong University of Science and Technology, Ph.D. degree from Nanyang Technological University, Singapore, in 1995, 1998 and 2002, respectively. He is a Lecturer of Aston University, U.K. He has authored or co-authored over 100 technical papers in major international journals and conferences. His current research interests include 4G/5G technologies, ADAS, connected autonomous vehicles, Internet of Things, mobile edge computing, machine learning and big data analytics. Dr. He was Editor or Guest Editor of several international journals, including Wireless Communications and Mobile Computing, IEEE Access, International Journals of Communication Systems, International Journal of Distributed Sensor Networks, and KSII Transactions on Internet and Information Systems. He acted as TPC chair for international conference ICAIT2010 and ICONI2010. He is a TPC member of many international conferences including IEEE Globecom and IEEE ICC.

Yi Zhou received the Ph.D. degree from Shanghai Jiao Tong University, Shanghai, China, in 2010. She is with the Institute of Image Communication and Network Engineering, Shanghai Jiao Tong University. Her research on big data analytics and Chinese characters recognition has been supported by Huawei and the National Natural Science Foundation of China. Her current research interests includes big data mining and object recognition.

Kai Chen received the Ph.D. degree from Shanghai Jiao Tong University, Shanghai, China. He is with the Institute of Image Communication and Network Engineering, Shanghai Jiao Tong University. He is a key member of the Institute on Network Engineering Research. He was the principle investigator of several key nation projects and many industry-academia research projects. His current research interests include information retrieving, object recognition and big data mining.

Zuoyin Tang received the Ph.D. degree from the University of Bath, Bath, U.K., in 2008. She is currently a Lecturer with the School of Engineering and Applied Science, Aston University, U.K. She has authored or co-authored over 40 technical papers in major international journals and conferences. Her current research interests include resource management for cellular networks, Internet of Things, wireless sensor networks, and fog computing.

Zhiliang Xiong received the B.E. degree in Computer Science from Huazhong University of Science and Technology in 1996. He was a software engineer at Nortel from 1996 to 1997. He was the head of engineering research and development department at C-Cube untill 2013. He is now the CEO of Forward Innovation Ltd., which developed several advanced driving assistance system products, including forward collision warning (FCW), lane departure warning (LDW), adaptive cruise control (ACC) and lane keeping assist (LKA).