Action-Driven Object Detection with Top-Down Visual Attentions

Donggeun Yoo, *Student Member, IEEE,* Sunggyun Park, *Student Member, IEEE,* Kyunghyun Paeng, *Student Member, IEEE,* Joon-Young Lee, *Member, IEEE,* and In So Kweon, *Member, IEEE*

Abstract—A dominant paradigm for deep learning based object detection relies on a *"bottom-up"* approach using *"passive"* scoring of class agnostic proposals. These approaches are efficient but lack of holistic analysis of scene-level context. In this paper, we present an *"action-driven"* detection mechanism using our *"top-down"* visual attention model. We localize an object by taking sequential actions that the attention model provides. The attention model conditioned with an image region provides required actions to get closer toward a target object. An action at each time step is weak itself but an ensemble of the sequential actions makes a bounding-box accurately converge to a target object boundary. This attention model we call AttentionNet is composed of a convolutional neural network. During our whole detection procedure, we only utilize the actions from a single AttentionNet without any modules for object proposals nor post bounding-box regression. We evaluate our top-down detection mechanism over the PASCAL VOC series and ILSVRC CLS-LOC dataset, and achieve state-of-the-art performances compared to the major bottom-up detection methods. In particular, our detection mechanism shows a strong advantage in elaborate localization by outperforming Faster R-CNN with a margin of +7.1% over PASCAL VOC 2007 when we increase the IoU threshold for positive detection to 0.7.

Index Terms—Object detection, visual attention model, deep convolutional neural network.

1 Introduction

With the recent advance [1] of deep convolutional neural network (CNN) [2], CNN based object classification methods in computer vision community have reached human-level performances on the ILSVRC [3] classification task; 3.57% [4] and 3.58% [5] in top-5 error which are even superior to human showing 5.1% error [3]. Thus, current research focus in visual recognition is quickly moving towards richer image understanding problems such as object detection, pixel-level semantic segmentation, image description and question answering in a natural language. Our focus is lying on the object detection problem.

There has been a long line of successful works for object detection [6], [7], [8], [9], [10], [11], [12] but significant progress in terms of accuracy and efficiency has been achieved by deep learning approaches [13], [14], [15], [16], [17], [18], [19], [20] for quite recent years. Among a large literature on object detection with deep learning approaches, one major state-of-the-art family [15], [16], [17], [18] is in Region CNN (R-CNN) pipeline; extracting class agnostic object proposals, applying object classifiers and refining bounding-boxes. The researches [21], [22], [23], [24] that incorporate R-CNN [15] reported top scores in ILSVRC'14 and Faster R-CNN [18] won at ILSVRC'15. However, even the most accurate and efficient R-CNN pipeline embeds a limitation of not reflecting the important visual contexts outside a proposal, caused by the passive scoring of the proposal with classifiers. To avoid such limitation, there are alternative top-down approaches [25], [26], [27], [28] which actively explore the location of a target object by taking surrounding context into account. However, the top-down approach for deep learning based object detection has not been much investigated yet.

In this paper, we propose an action-driven method for top-down object detection. We cast an object detection problem as a sequential action problem. We introduce a visual attention model named AttentionNet which acts as an agent determining what action should be taken in the next step. This model takes an image region as input and provides the optimal actions for getting closer toward a target object. In our detection mechanism, this attention model is fully utilized from beginning to the end of object detection pipeline. Starting from a whole image or a large region, our detection mechanism actively explores the location of a target object and finishes by drawing an accurate bounding-box. The background context surrounding a target object is also taken into consideration since the searching scope in its early stage is broad enough.

The core of our detection mechanism lies on an idea of taking an action sequence by order of the attention model. The attention model tells us a pair of actions, which should be taken at the top-left and bottom-right corner of an input image, to get closer to a target object. For instance, the action could be "go down" or "go left" at each corner respectively. We then simply take the actions by cropping the input image until the image boundary converges to a target object. Even if each action is inaccurate, taking a set of multiple actions results in an accurate boundary of a target object, such as an ensemble method combines many weak learners to produce a strong learner. Fig. 1 shows real examples of our detection

- *D. Yoo, and I.S. Kweon are with KAIST, South Korea. E-mail: dgyoo@rcv.kaist.ac.kr, iskweon@kaist.ac.kr*
- *S. Park and K. Paeng are with Lunit Inc., South Korea. E-mail: sgpark@lunit.io, khpaeng@lunit.io*
- *J.-Y. Lee is with Adobe Research, CA, USA. E-mail: jolee@adobe.com*

mechanism. Starting from an entire image, the bounding-box moves sequentially then converges to a target object boundary.

Our detection mechanism is radically distinct from the state-of-the-art R-CNN based methods. These methods depend on the bottom-up object proposals and score them with classifiers, while we follow a top-down search strategy. The bottom-up object proposals are based on the characteristic of a local scene so called "objectness". Proposals of [12], [29], [30], [31] are driven from low-level features and those of [16], [18] are from trainable mid-level features. In contrast, our top-down mechanism is controlled by high-level subtasks, i.e. detection boxes are driven from a sequence of actions. The bottom-up approaches are inherently faster than our top-down approach since they are feed-forward while ours is recurrent. However, the top-down approach has its strong property coming from high-level reasoning, such that the context surrounding a target object could be reflected to the action sequence. Thus, this top-down approach can be a complementary way toward the next direction of object detection.

Our detection mechanism with a single attention model does everything necessary for a detection pipeline but yields state-of-the-art performance. With a single attention model, we 1) detect initial regions where a single instance is included, 2) detect objects by taking sequential actions from each initial region, and 3) finally refine the localizations by taking an extra action sequence. Therefore, we do not incorporate any separate modules for object proposals nor post bounding-box regression.

A preliminary version of this work was published in [26], which can detect only a single object class. Since then we have generalized the detection mechanism to handle multiple object classes with a shared attention model. Also, we have given important modifications to determining the scaling factors used for multi-scale training and inference.

1.1 Contributions

In summary, our contributions are three-fold.

1) We suggest an action-driven object detection mechanism, which actively search exact object locations by taking action sequence produced by an attention model.
2) The detection mechanism does not requires any separated modules for object proposal nor post bounding-box regression. Taking sequential actions covers all these.
3) This is a top-down detection mechanism which first achieves the state-of-the-art accuracy compared to the recent bottom-up methods.

2 Related Works

There has been a large literature on object detection for the last few decades. Object models are learned from low-level features [6], [9] or mid-level part based features [10], [11], and the models evaluate image regions in a sliding window fashion. Since then, a raise of object proposal methods [12], [29], [30], [31], which generate thousands of potential bounding-boxes, substantially improves detection efficiency compared to the sliding window search. We refer readers to [32] for an in-depth study on various object proposal methods.

In recent researches of object detection, a significant progress in both of accuracy and efficiency has been achieved by a powerful combination of all three; high-quality object proposals [12], a deep network to represent the proposals [1], and big data for training the network [33]. This framework called R-CNN was proposed by Girshick *et al.* [15] and has become a dominant paradigm for object detection. From here we limit our review to the deep learning approaches.

The successful performance of the R-CNN pipeline triggered engineering challenges to make it run in real-time. An attempt to train a feed-forward network for object proposal [16] speeds up the proposal step but requires per-region classification which is far from a real-time speed. In contrast, per-region pooling over a shared convolution feature map [17], [21] substantially boosts the speed for classifying the proposals but extracting the proposal [12] is a bottleneck. Ren *et al.* [18] design a region proposal network, and make this network and a classification network share the convolution feature maps. This system called Faster R-CNN finally performs in near real-time with an improved accuracy.

These bottom-up approaches are inherently feed-forward and fast but lack of holistic analysis of scene-level context. Our top-down approach is relatively slow due to the recurrent actions but can see the larger context while taking actions. There has been three recent works [25], [27], [28] similar to ours in terms of adopting the action-driven top-down approach. Gonzalez-Garcia *et al.* [25] propose an active search strategy which depends on the spatial context and the region scores in the previous state. Caicedo and Lazebnik [27] and Mathe *et al.* [28] also present action-driven detection methods in which an agent determining actions is trained by reinforcement Q-learning [34]. These three works successfully apply the top-down approach to the detection problem, however, the performances are still far from state-of-the-art competitors, and the detector of [27] is class-specific. In this paper, we extend our previous class-specific model [26] to handle multiple classes and achieve state-of-the-art performances.

We introduce another side of detection paradigm where a detection problem is framed as a regression problem. A feed-forward network directly estimates bounding-boxes. Szegedy *et al.* [13] trains a deep network which maps an image to a rectangular mask of an object. Sermanet *et al.* [14] also employ a similar approach but their network directly estimates bounding-box coordinates. These models produce a single bounding-box so should be evaluated on sliding windows to detect multiple instances. Quite recently, Redmon *et al.* [19] and Liu *et al.* [20] develop a regression model that produces multiple bounding-boxes and their class probabilities. They estimate a bounding-box for each grid cell of a convolution feature map, so all the outputs are obtained in a single feed-forward path. All these detection-by-regression approaches are also related to our work in that they do not rely on object proposals and actively produce bounding-boxes. However, ours is distinct from these methods in that our regression proceeds sequentially with high-level reasoning.

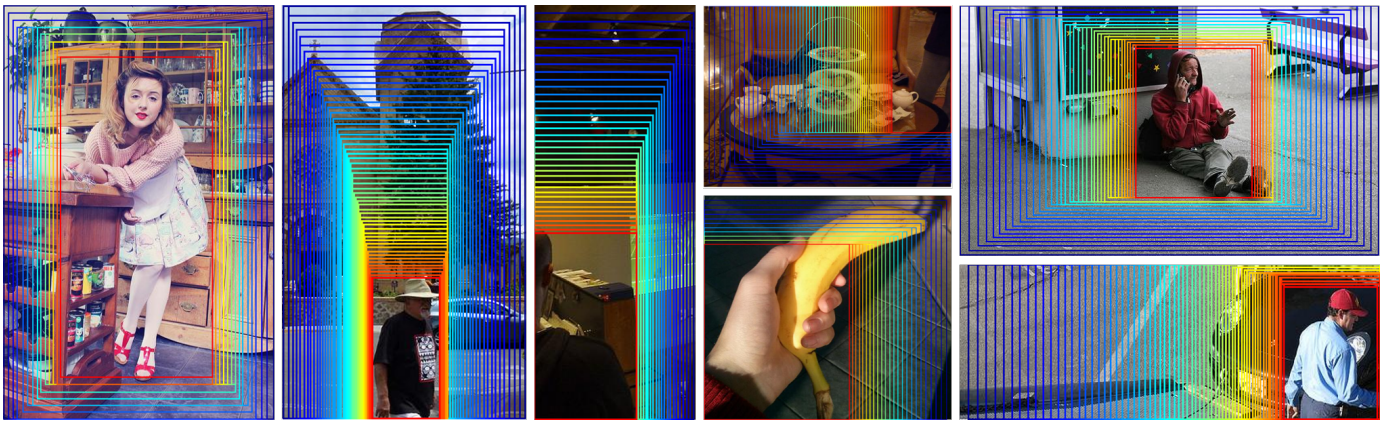


Fig. 1. Real detection examples of our detection mechanism. Starting from an image boundary (dark blue bounding-box), our detection mechanism recursively narrows the bounding-box down to a final human location (red bounding-box). For visual effects, we set the size of each movement small (5 pixels given a 224×224 size input) for all the examples.

We use an attention model as an agent, so a long line of the attention models is also related to ours. For recent years, incorporating the visual attention idea into a deep network [35], [36], [37], [38] has been proposed to select regions that need more attention for better visual recognition such as classification [38], [39], [40] and caption generation [41], [42]. Our use of the attention model differs from them in its aim and supervision. We incorporate the attention model in order to determine the optimal action to get closer to an object whereas their models aim at more focused representation to help their target recognition tasks. Also, we can train our model in a supervised fashion with optimal action labels determined from ground-truth bounding-boxes. In contrast, attention of these methods cannot be directly supervised since their target labels do not include locations of significant objects. For this reason, they often employ the reinforcement Q-learning, or design a differentiable model that could be optimized with a back-propagation in a weakly supervised fashion.

This paper is organized as follows. We first introduce our class-specific detection mechanism in Sec. 3 and evaluate the method by a human detection task in Sec. 4. We then generalize the detection mechanism to multiple object classes in Sec. 5 and also evaluate that in Sec. 6. We finally conclude this study in Sec. 7.

3 Detection Mechanism

We introduce our detection mechanism under an assumption that an input image includes a single object instance only. Extension of the mechanism to multiple instances will be described in Sec. 3.4.

Fig. 2 shows how we frame an object detection problem as a sequential action problem. We first warp an input to a fixed size image and feed it to the attention model named AttentionNet. The attention model then tells us a pair of actions required for the input to get closer to the target object. The actions will be applied to the top-left corner (TL) and the bottom-right corner (BR) of the input image respectively. We define a high-level action set for TL as follows; go right $\rightarrow$, go right-down $\searrow$, go down $\downarrow$, stop $\bullet$ and reject $\times$. We also define the action set for BR in this way but the directions of the movement actions are opposite to those of TL.

The attention model is indicating $\downarrow_{\text{TL}}$ and $\nwarrow_{\text{BR}}$ in Fig. 2. We then apply the actions at both corner in the way of cropping the input. The amount of movement l is constant. The cropped image is fed to the attention model again until the image meets one of the two terminal conditions; $\times$ at both corners, or $\bullet$ at both corners. An image given $\times$ at both corners is regarded as a background while an image given $\bullet$ at both corners is a detection result. The detected image boundary is back-projected to a bounding-box in the original input image domain. Given a stopped (detected) bounding-box b and its corresponding output activations $\mathbf{y}_{\text{TL}}, \mathbf{y}_{\text{BR}} \in \Re^5$ before a softmax normalization, the detection score s^b is discriminatively defined as

$$
\begin{aligned}
s^b &= \frac{1}{2}\left(s^b_{\text{TL}} + s^b_{\text{BR}}\right), \quad \text{s.t.} \\
s^b_{\text{TL}} &= y^{\bullet}_{\text{TL}} - (y^{\rightarrow}_{\text{TL}} + y^{\searrow}_{\text{TL}} + y^{\downarrow}_{\text{TL}} + y^{\times}_{\text{TL}}), \\
s^b_{\text{BR}} &= y^{\bullet}_{\text{BR}} - (y^{\leftarrow}_{\text{BR}} + y^{\nwarrow}_{\text{BR}} + y^{\uparrow}_{\text{BR}} + y^{\times}_{\text{BR}}).
\end{aligned} \tag{1}
$$

Compared with the R-CNN framework which depends on object proposals, our detection starts from a large area and actively reaches at a terminal point with *stop signals*. In early stage of this procedure, we can take the *large context* surrounding an object into consideration. Such a large context is an important cue for identifying the class of the object. This benefit will be highlighted with an experiment in Sec. 3.3 again.

Compared with the previous detection-by-regression approaches [13], [14], [19], [20], we solve the regression problem by iterative classifications of high-level actions. Even if the actions in early stage could be inaccurate, subsequent actions become stronger as the searching scope is gradually narrowed down to an object.

3.1 Attention Model

Our detection mechanism requires an agent which determines the optimal actions to be applied to both corner of an input. The agent can be a regression model that tells us a location coordinate but the regression is a more difficult task for a network compared to a classification task that classifies

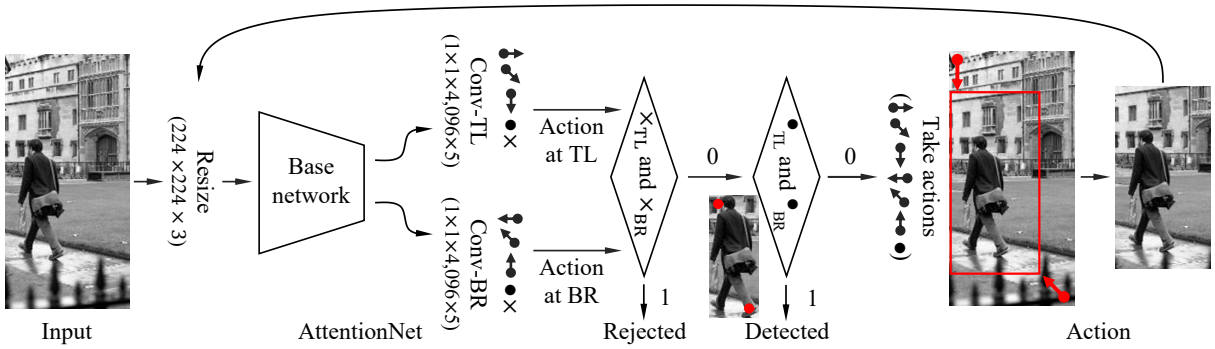


Fig. 2. Action-driven detection mechanism. The attention model tells us a pair of actions which should be taken at the top-left corner (TL) and the bottom-right corner (BR) of an input image. The action set for TL is defined as follows; go right →, go right-down ↘, go down ↓, stop • and reject ×. The action set for BR is also defined in this way but with opposite directions. If the attention model produces the action "reject ×" in both corners, we reject the input. If not, we apply the actions to the input and feed it to the model again until it meets the action "stop •" in both corners.

quantized directions. Thus we choose a classification model, which is trainable with a softmax loss. The end of the model is composed of two classification layers for both corners, and each layer classifies the five actions including the three movement actions (→, ↘, ↓ for TL) and the two termination actions (•, ×). The five fully connected filters of 1×1×4,096 size determine the five action scores for each corner. We can choose a base network for this model from any popular convolutional network architectures. The illustration of this model is shown in Fig. 2.

3.2 Training

The required actions are determinant to the location of a target object. Always we can determine the optimal action for an arbitrary region to get closer to the ground-truth bounding-box. Selecting the optimal action at each time step does not depend on the previous action sequence. Thus, we can train our attention model regardless of that.

Caicedo and Lazebnik [27] also present an action-driven detection mechanism with an agent. Their action set are differently defined such as horizontal moves, vertical moves, scale changes and aspect ratio changes. Given this action set, they adopt the reinforcement Q-learning [34] with an IoU (Intersection over Union) based reward. Despite their interesting application of reinforcement learning for object detection, the reinforcement method inherently has a high variance in the gradient of the expected reward so it is difficult to accurately find a Q value which is approximated by a deep network with a limited size of training set. In contrast, since our actions are designed to be optimally chosen to increase the IoU at any states, we can train our attention model with the softmax loss.

To make the attention model operate in the scenario we devise, it is important to process original training images to a suitable form. During the inference stage, the number of possible action pairs is 17 (=4×4+1) such as $\{\rightarrow, \searrow, \downarrow, \bullet\}_{\mathrm{TL}} \times \{\leftarrow, \nwarrow, \uparrow, \bullet\}_{\mathrm{BR}}$ for positive regions and $\{\times_{\mathrm{TL}}, \times_{\mathrm{BR}}\}$ for negative regions. To evenly cover these 17 cases in training, we augment the original training images into a reformed region set.

Fig. 3 shows examples how we process an original training image to multiple augmented regions. We randomly sample any regions between the inner and outer bound.

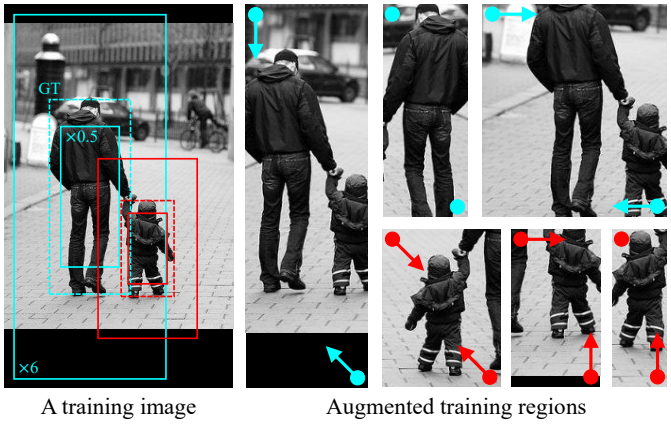


Fig. 3. Examples of generating training regions to learn the attention model. A dashed rectangle is a ground-truth. The box inside a ground-truth is an inner bound which is 2 times smaller than the ground-truth while the outer bound is 6 times larger than that. Any region between these two bounds is randomly sampled with a random horizontal flip. A ground-truth action label at each corner is assigned automatically. The area beyond the image boundary is filled with zeros.

The inner bound is 2 times smaller than the ground-truth (a dashed box) whereas the outer bound is 6 times larger. The area out of an image boundary is filled with zeros. It is important to make the outer bound sufficiently large to take the *large context* into account during training. Each region is horizontally flipped with a probability of 0.5. We then assign a pair of ground-truth action labels at both corners that is determined from a ground-truth bounding-box. We also randomly generate negative regions which are not overlapped with ground-truth bounding-boxes.

Some regions probably include multiple instances as in the most top-right example in Fig. 3. In this case, we simply assign action labels for the biggest instance. These regions are also essential for training. Let us consider a multiple instance detection scenario. If an attention model is trained without these regions, a final detection result from a large initial region probably includes the multiple instances at ones. To make our mechanism always narrow the box down to the biggest instances within the visible area, we must make the outer bound sufficiently large (×6).

Method	Approach	AP (%)
Top-1 result from R-CNN [15]	Bottom-up	79.4
AttentionNet	Top-down	**89.5**

TABLE 1
Single-human detection performances of the bottom-up and top-down approaches on a subset of PASCAL VOC 2007 `test` set, in which each image contains a single human instance.

When we compose a mini-batch for training, we select positive and negative regions in an equal portion. In a batch, each of the 16(=4×4) cases for positive regions occupies a portion of $1/(2\times16)$, and the negative regions occupy the remaining portion of $1/2$. The loss for training is an average of the two log-softmax losses computed independently in TL and BR

$$\ell = \frac{1}{2}\cdot\ell_{\text{softmax}}\left(\mathbf{y}_{\text{TL}}, t_{\text{TL}}\right) + \frac{1}{2}\cdot\ell_{\text{softmax}}\left(\mathbf{y}_{\text{BR}}, t_{\text{BR}}\right)$$
$$\text{s.t.}\quad \ell_{\text{softmax}}\left(\mathbf{y}, t\right) = -y_t + \log\sum_i e^{y_i} \qquad (2)$$

where $\mathbf{y}$ is a 5-dimensional action score vector and t is a ground-truth action label index.

3.3 Top-down VS. Bottom-up

Before we make our detection mechanism detect multiple instances, we verify the effectiveness of our top-down approach, against to the bottom-up approach relying on region proposals [12]. As studied by [43], strong mid-level activations in a deep network come from object parts that is distinctive to other object classes. Since R-CNN based detection depends on the score computed with the activations inside each proposal, the results often focuses on discriminative object *parts* (e.g. face) rather than an entire object (e.g. entire human body).

To analyze this issue, we design an experiment of *single*-human detection over PASCAL VOC 2007 [44]. We train an AlexNet [1] based R-CNN human detector with the code provided by the authors [15]. Also, we train an AlexNet based attention model with the same training data. From VOC 2007 `test` set, we choose images that contain a single human instance to make a sub test set. To highlight the only difference that comes from top-down and bottom-up approaches, we just choose the top-1 region as a detection result for R-CNN. Our detection mechanism begins with the entire image boundary and detects an instance. We measure average precisions (AP) with a standard IoU threshold of 0.5 for positive detection.

Table 1 shows the results. The bottom-up approach shows 79.4% while our top-down approach shows 89.5%. The bottom-up approach shows much lower detection performance due to the weak correlation between a classification score of a proposal and the *entire* human body. As shown in Fig. 4, the maximally scored object proposal of R-CNN is prone to focus on the discriminative faces rather than the entire human bodies. In contrast, our detection mechanism reaches a terminal point starting from a boundary out of a target object.

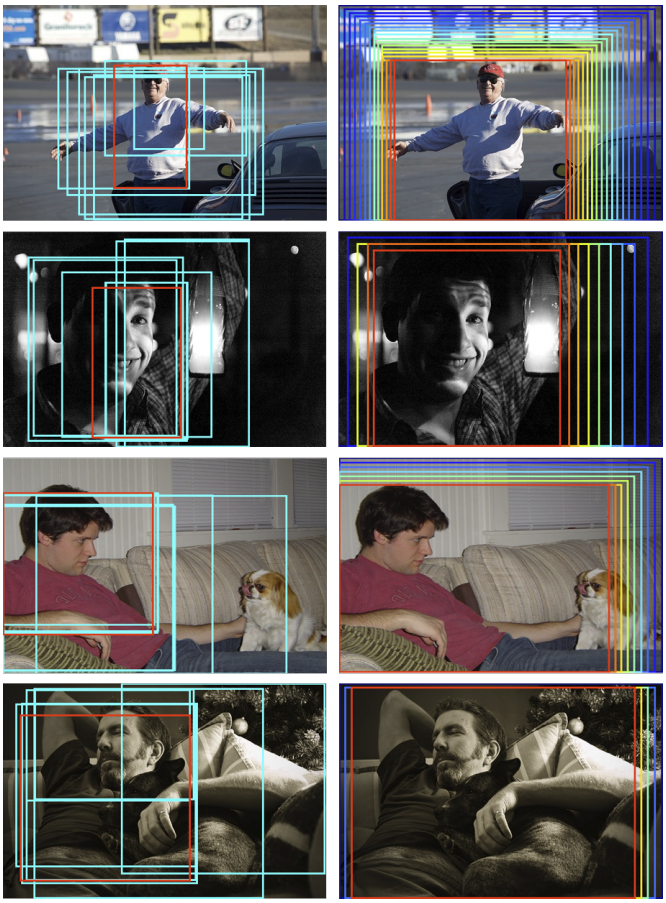


Fig. 4. Qualitative comparison of the bottom-up and top-down approach with a *single*-human detection task. The left column is the bottom-up R-CNN based method and the right column is our top-down method. In the left column, a red bounding-box is the top-1 region and the cyan boxes are top-10 regions.

3.4 Initial Glance

Our attention model provides actions toward a single instance of a visible region. In this section, we introduce an efficient method to extend our detection mechanism to a practical scenario in which an image includes multiple instances. Our solution is to initialize a large box for each instance. We call this initial glance. To this end, we also utilize our attention model, therefore, a separated model is unnecessary. We then can detect an instance from each initial glance, and merge the results into a reduced number of bounding-boxes followed by a final refinement procedure for which we reuse the attention model again.

A required condition for a region to be an initial glance is that the region should contain an entire instance with sufficient surrounding contexts. Let us assume we have an arbitrary region in an image and we feed this region to the attention model. Among 17 possible action combinations, the action prediction of $(\searrow_{\text{TL}}, \nwarrow_{\text{BR}})$ guarantees that the region includes the entire body of an instance with enough margins. In the other predictions, it is possible for a region to be truncating an instance or be a background. Some examples are shown in Fig. 5.

To boost speed and recall of the initial glance mining, we follows the fully convolutional technique presented by

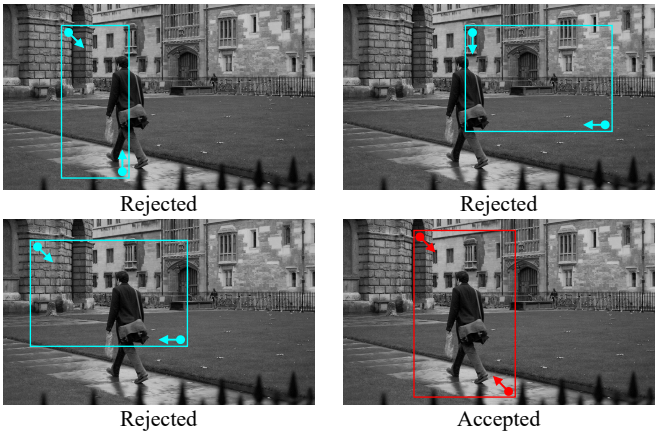


Fig. 5. A condition for a window to be an initial glance. Among muti-scale and muti-aspect ratio windows, we choose only regions that predicted as ($\searrow$,$\nwarrow$) at each corner as initial glances to make sure that the entire object instance is included.

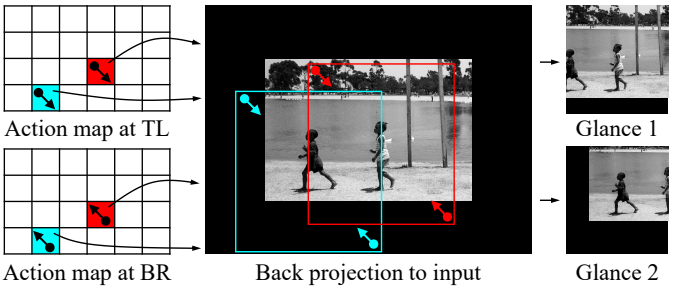


Fig. 6. Efficient initial glance mining from action maps. A large image is fed to the fully convolutional attention model to obtain an action map for each corner. Regions satisfying the condition of ($\searrow_{\text{TL}}, \nwarrow_{\text{BR}}$) becomes initial glances. The glances are given to the attention model again to detect each instance as illustrated in Fig. 2.

[45], [46], [47]. An input of a convolutional network is not limited to a fixed size since a fully connected layer could be replaced with a convolution layer containing 1×1 size filters. We feed K multi-scale multi-aspect ratio images to our fully convolutional attention model and obtain K action maps for each corner. For instance, if our base network is VGG-16 [23], a pixel and its neighbor in an action map correspond to 224×224 size regions with a stride of 32 in an input image. Given these action maps, we choose regions that satisfy ($\searrow_{\text{TL}}, \nwarrow_{\text{BR}}$) condition as initial glances. An example is shown in Fig. 6. The initial glances are then given to the attention model again to detect each instance as the mechanism described in Fig. 2. For some instances located on side of an image, each side of an image dilates with a 112(=224/2) size margin filled with zeros before being fed to the attention model.

Object instances are diverse in aspect ratio and scale. Thus the scales and aspect ratios for an input image are important to successfully mine initial glances in an inference stage. Thus, we introduce a data-driven approach to determine K scales and aspect ratios. Let us assume the minimum input size for an attention model is $(224, 224)$. Also, we have an input image of (w, h) size and its ground-truth bounding-box b of (w_b, h_b) size. If we rescale this input to $\left(w \cdot \frac{224}{w_b}, h \cdot \frac{224}{h_b}\right)$ size, each pixel in an action map corresponds to a region which has a size equal to the ground-truth bounding-box in the original image domain. To make some regions satisfy ($\searrow_{\text{TL}}, \nwarrow_{\text{BR}}$) condition with an enough margin, we can define a margin factor $\alpha > 1$ with which we can rescale the input to $\left(w \cdot \frac{224}{\alpha \cdot w_b}, h \cdot \frac{224}{\alpha \cdot h_b}\right)$ size. We define a scaling factor multiplied to (w, h)

$$\mathbf{s}_b = \left[s_b^w, s_b^h\right] = \left[\frac{224}{\alpha \cdot w_b}, \frac{224}{\alpha \cdot h_b}\right]. \tag{3}$$

Our objective here is to determine K representative scaling factors $\{\mathbf{s}_k \mid k = 1 \cdots K\}$, which will be used for the inference stage, to maximize the chance for mining initial glances.

Given a training image set I^{tr} and their ground-truth bounding-boxes which are $\{w_b, h_b \mid b \in I^{tr}\}$-size, we compute scaling factor samples $\{\mathbf{s}_b \mid b \in I^{tr}\}$ for all the bounding-boxes. K representative scaling factors are then estimated by grouping the samples. We run k-means clustering algorithm over the samples $\{\mathbf{s}_b \mid b \in I^{tr}\}$ in a log-scale space and obtain K centroids $\{\mathbf{s}_k \mid k = 1 \cdots K\}$. In an inference stage, we can rescale a test image to multiple sizes $\left\{w \cdot s_k^w, h \cdot s_k^h \mid k = 1 \cdots K\right\}$ and feed them to the attention model for the initial glance mining. If a rescaled image size is smaller than 224, we pad zeros with equal margins in both sides.

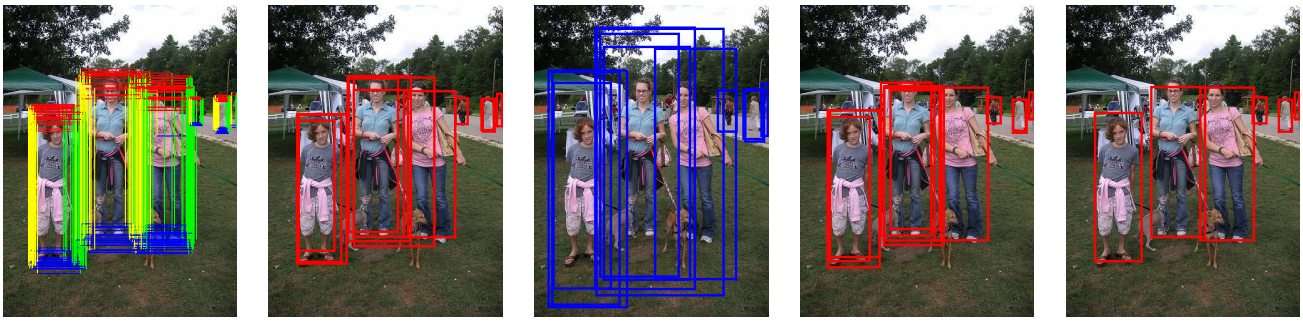
3.5 Initial Detection and Refinement

Each initial glance is fed to the attention model recurrently until this meets ($\bullet_{\text{TL}}, \bullet_{\text{BR}}$) or ($\times_{\text{TL}}, \times_{\text{BR}}$). The first image in Fig. 7 shows a real example of the initial detection. The bounding-boxes are merged to a decreased number by a single-linkage clustering; a group of bounding-boxes satisfying a minimum IoU μ_0 are averaged into one with their scores of Eq. (1).

To refine the result, [10], [15] conduct a post bounding-box regression, which re-localizes the bounding-boxes. This is a linear regression model which maps a feature of the bounding-box to a new one. In our case, we can employ the attention model again for this refinement step. We simply rescale each bounding-box in Fig. 7-(b) to a new region with a rescaling factor of β as shown in Fig. 7-(c). These reinitialized regions are fed to the attention model again and result in new bounding-boxes as shown Fig. 7-(d). This re-detection procedure gives us one more chance to reject false positives as well as fine localization. These bounding-boxes are merged again to final results with an IoU μ_1.

4 Experiments

In this section, we perform human detection task on public datasets to comprehensively verify the strength of our detection mechanism. Among a wide range of object classes, it is beyond question that the class "human" has taken center stage in object detection for decades due to its broad applications. Nonetheless, human detection on uncontrolled natural images is still challenging due to the self-occlusions, diverse poses and clothes.



(a) Initial detections. (b) Initial merge. (c) Rescale. (d) Re-detections. (e) Final merge.

Fig. 7. Intermediate results during our detection procedure. (a) shows initial detection results from the initial glances. The boxes are merged as in (b) with an IoU μ_0. To refine the results, the boxes are rescaled with a factor β as in (c) and shrink again as in (d). The resulting boxes are finally merged with an IoU μ_1.

4.1 Datasets

For human detection task, we choose PASCAL VOC 2007 and 2012 [44] since these are composed of user taken web photos from Flickr so the image condition is completely uncontrolled. A lot of human instances in these sets are severely occluded, truncated and overlapped with diverse pose variations and scales. Each of PASCAL VOC 2007 and 2012 include 10K and 23K images of 20 object classes and equally divide into a `trainval` set and a `test` set. Following the standard protocol used in the previous human detection researches over these sets, we use `trainval` images for training, and report an average precision (AP) on `test` set. For PASCAL VOC 2012, we submit our results to the evaluation server and receive the AP.

4.2 Base Network

We choose VGG-M [48] and VGG-16 [23] as the base networks for the attention model. VGG-M, designed by Chatfield *et al.* [48], is a variant of AlexNet [1] with small modifications. The stride and filter of the first layer are smaller than those of AlexNet but the stride at the second convolution layer is larger. This model is also composed of 8 convolution layers. We adopt this model due to its lower Top-5 error (16.1%) than AlexNet on the ILSVRC classification without significant increase in computation. VGG-16, designed by Simonyan and Zisserman [23], is a much deeper network composed of 16 convolution layers with smaller filter sizes. This single model shows 9.9% Top-5 error on the ILSVRC classification.

The PASCAL VOC series is too small to learn this large model. Thus we initialize the base network with pre-trained weights on ILSVRC CLS-LOC dataset [3] and fine-tune the model for our target task. For each model, we pick out all the pre-trained layers except for the last classification layer and stack our action layers on top of those.

4.3 Parameters

We follow the fine-tuning technique in [43] to train the attention model; an initial learning rate of the pre-trained weights is 0.001 whereas that of the randomly initialized weights is 0.01. When the learning curve is saturated, we decrease the learning rates 0.1-times. For the inference stage, we set the size of each movement action l to 30 pixels for initial detection and 15 pixels for refinement. The number of scaling factors K is 24 with a margin factor α of $\sqrt{6}$. The merging parameters (μ_0, μ_1) are (1.0, 0.5). The rescaling factor β is 3.0.

Methods	Class	VOC'07 Person AP(%)	VOC'12 Person AP(%)
AttentionNet	Single	**71.4**	**72.8**
AttentionNet-Refine	Single	69.9	72.4
Person R-CNN + BB-Regression	Single	59.7	-
Person R-CNN + BB-Regression×2	Single	**59.8**	-
Person R-CNN + BB-Regression×3	Single	59.7	-
DPM [10]	Multi	41.9	-
Poselets [11]	Single	46.9	-
k-Poselets [49]	Single	45.6	-
G-P, HOG-III [50]	Single	55.5	57.0
Poselets (AlexNet) [51]	Single	59.3	58.7
Regression (AlexNet) [13]	Multi	26.2	-
DeepMultiBox (AlexNet) [16]	Multi	37.5	-
R-CNN (AlexNet) [15]	Multi	58.7	57.8
R-CNN + SPP (ZFNet) [21]	Multi	57.6	-
Q-Learning (ZFNet) [27]	Single	45.7	-
Q-Learning (AlexNet) [28]	Multi	-	18.7
Our Multi-class AttentionNet (VGG-M)	Multi	**72.0**	**72.4**

(a) Results with 8-layered networks.

Methods	Class	VOC'07 Person AP(%)	VOC'12 Person AP(%)
AttentionNet	Single	**77.6**	**75.4**
AttentionNet-Refine	Single	77.4	75.3
AttentionNet + Faster R-CNN	Single	81.5	79.9
AttentionNet-Refine + Faster R-CNN	Single	**82.1**	**81.4**
R-CNN (VGG-16) [15]	Multi	64.2	65.2
Fast R-CNN (VGG-16) [17]	Multi	69.0	69.8
Faster R-CNN (VGG-16) [18]	Multi	76.3	77.4
MR-CNN + S-CNN + Loc. (VGG-16) [52]	Multi	74.9	76.4
YOLO (26 layers) [19]	Multi	-	63.5*
SSD (VGG-16) [20]	Multi	76.6	**83.3***
Our Multi-class AttentionNet (VGG-16)	Multi	**80.1**	79.2

*trained on a superset of `trainval'07+test'07+trainval'12`.

(b) Results with networks composed of more than 16 layers.

TABLE 2
Human detection performances on PASCAL VOC 2007/2012 `test` set.

4.4 Results and Analysis

We evaluate our method by the human detection task on PASCAL VOC 2007 and 2012. The results and comparisons with 8-layered networks (e.g. VGG-M [48], AlexNet [1],

ZFNet [53]) are shown in Table 2-(a), and those with networks composed of more than 16 layers (e.g. VGG-16 [23], YOLO [19]) are shown in Table 2-(b). When our attention model is based on VGG-M, our method achieves 71.4% and 72.8% for each dataset without the refinement step of Fig. 7-(c∼e). After the refinement step, marked as "AttentionNet-Refine", we achieve slightly worse performances of 69.9% and 72.4%. When our attention model is equipped with the VGG-16 model, the performances significantly increase to 77.6% and 75.4% for each dataset. After the refinement step, the performances are also slightly decreased to 77.4% and 75.3%. This single class detection has benefit from the refinement step but we observe that this improves the multi-class detection to be presented in Sec. 5.

We can reinterpret our method as a regression through iterative classifications, so we compare ours with the detection-by-regression methods such as bounding box regression. To do so, we train a "Person R-CNN" and a bounding box regressor "BB-Regression" by using the official code[1] provided by the R-CNN authors [15]. Only images of "person" class are used as positives while the other images are used as backgrounds, to make the comparison completely fair. The initial detection boxes from R-CNN are given to the bounding box regressor then the boxes are re-localized. This method shows 59.7% as shown in the second block of Table 2-(a). As our method which repeats actions, we also repeat the bounding-box regression in R-CNN, which is noted by "BB-Regression$\times 1, 2$". However, the improvement is negligible: +0.1% and +0.0% for the second and third iterations. Our method beats these approaches with a large margin more than +10% and these results verify the effectiveness of the iterative classifications as a regression method.

There are more detection-by-regression methods [13], [16] in which the network is trained to produce a target object mask [13] or bounding-box coordinates [16] for the purpose of class-agnostic object proposals. Still, our method clearly outperforms these methods as shown in Table 2-(a). Quite recently, [19], [20] estimate a bounding-box for each grid cell of a convolution feature map, so all the outputs are obtained in a single feed-forward while performing in real-time. The performances of these methods are summarized in Table 2-(b). YOLO [19] with 26 convolution layers shows 63.5% which is much lower than ours. SSD [20] based on VGG-16 achieves the state-of-the-art performance of 83.3%, however, this model is trained with a *superset* of `trainval'07+test'07+trainval'12`. These results also verify the benefit of our iterative classifications compared to these regression approaches.

Poselets-based methods [11], [49], [51] are related to ours since these methods are limited to a single object class (e.g. human). Among them, [51] is a deep learning approach which uses an 8-layered network (AlexNet) like ours. However, our method significantly outperforms all these approaches. Through these results, we can expect that our method can be successfully extended to multiple classes, since we do not use a human-specific model to detect humans.

1. https://github.com/rbgirshick/rcnn

R-CNN [15] and its advanced variants [17], [18] have been the most successful detection method so far. These bottom-up methods contrast with our top-down approach. Let us compare our method with R-CNN using 8 layers in Table 2-(a). In VOC 2007, both [15] and [21] show around 58% performance, and our method outperforms them with 71.4% performance. For a fair comparison, we train another R-CNN that only detect the "person" class noted by "Person R-CNN" but the result is similar (59.7%). Our result from the 8-layered network is even significantly better than that of R-CNN with a 16-layered network in Table 2-(b). The performances of VGG-16 based Fast R-CNN [17] and Faster R-CNN [18] are summarized in Table 2-(b). Fast R-CNN shows 69.0% and 69.8% performances while our method beats them with 77.6% and 75.4% performances in VOC 2007 and 2012 respectively. Our method is competitive to Faster R-CNN which shows 76.3% and 77.4% performances.

These two methods, Fast R-CNN and Faster R-CNN, are faster than ours since they are feed-forward while ours is recurrent. Also, they are more advantageous in terms of recall because they only take visible regions into account. In contrast, our method has strong properties driven from scene-level contexts. Thus, we can expect a large performance improvement when we mix these complementary methods. We combine the boxes from Faster R-CNN with our boxes. We rescale our scores, and merge all the boxes and scores with an IoU of 0.5. The results are reported in the second block of Table 2-(b). We achieve the significantly boosted performances; 82.1% from 77.4% in VOC 2007 and 81.4% from 75.3% in VOC 2012. Through this experiment, we expect that our research on the top-down approach can contribute to a future hybrid model taking the advantages from both approaches.

There has been two top-down approaches for object detection [27], [28], which adopt reinforcement Q learning [34] to train their agents with rewards. The performances of these methods are 45.7% in VOC 2007 and 18.7% in VOC 2012 respectively, which are far from the state-of-the-art performance, as shown in Table 2-(a). In contrast, our actions are designed to be optimally chosen at any state so we can train our attention model with a softmax loss, and achieve much superior performances.

We perform extra experiments regarding other behaviors of our detection mechanism and analyze the results in Sec. 6 with a multi-class attention model.

5 Multi-Class Detection Mechanism

In this section, we generalize the attention model that has been specified for an object class to multiple object classes. We first modify the attention model to a multi-class version in Sec. 5.1. We then explain the initial glance mining in Sec. 5.2 followed by the remaining detection procedures in Sec. 5.3. We finally present the training method in Sec. 5.4.

5.1 Attention model

In the class-specific attention model, a pair of action layers is specified for a single object class. To make the model can provide actions regarding multiple classes, we define the action layers for each class and parallelize them on top of a

base network. This model is illustrated in Fig. 8. If we have N classes, the end of the model is composed of N pairs of action layers. Given an input, this model always produces N class-specific action pairs. However, only with these, we can not determine which action pair should be chosen for this input since the object class of the input is unknown. Thus, we define an extra classification layer which recognizes the input object class so that we are able to choose an action pair. The classification layer produces a (N+1)-dimensional vector composed of a background score and the N class scores. Since the classification layer tells us whether an input is background or not, we remove "reject $\times$" actions from all the action layers. Thus, each action layer produces a 4-dimensional vector composed of the 3 movement action scores and a stop action score.

The multi-class detection mechanism with this attention model proceeds as follows. From an input, we obtain classification scores $\mathbf{y}_{\text{cls}}=[y_1 \cdots y_{N+1}]$ and action scores $\{\mathbf{y}_{\text{TL}c}, \mathbf{y}_{\text{BR}c} \mid c = 1 \cdots N\}$. Here the action scores for each class are $\mathbf{y}_{\text{TL}c}=[y^{\rightarrow}_{\text{TL}c}, y^{\searrow}_{\text{TL}c}, y^{\downarrow}_{\text{TL}c}, y^{\bullet}_{\text{TL}c}]$ and $\mathbf{y}_{\text{BR}c}=[y^{\leftarrow}_{\text{BR}c}, y^{\nwarrow}_{\text{BR}c}, y^{\uparrow}_{\text{BR}c}, y^{\bullet}_{\text{BR}c}]$. If the input is predicted as a background class $\hat{c}$=N+1, the input is rejected. If not, we choose the predicted actions of the predicted class $\hat{c}$ for both corners and then take the actions. This procedure is repeated until the action predictions for both corners are $(\bullet_{\text{TL}\hat{c}}, \bullet_{\text{BR}\hat{c}})$, or the input is rejected with $\hat{c}$=N+1.

A detected region is back-projected to a corresponding bounding-box b in the original input image domain. The detection score s^b is also discriminatively defined as

$$
\begin{aligned}
s^b = (1-\gamma)\, s^b_{cls} + \gamma \left(s^b_{\text{TL}} + s^b_{\text{BR}}\right) & \\
s^b_{cls} = y_{\hat{c}} - y_{N+1} & \\
\text{s.t.} \qquad s^b_{\text{TL}} = y^{\bullet}_{\text{TL}\hat{c}} - (y^{\rightarrow}_{\text{TL}\hat{c}} + y^{\searrow}_{\text{TL}\hat{c}} + y^{\downarrow}_{\text{TL}\hat{c}}) & \quad (4)\\
s^b_{\text{BR}} = y^{\bullet}_{\text{BR}\hat{c}} - (y^{\leftarrow}_{\text{BR}\hat{c}} + y^{\nwarrow}_{\text{BR}\hat{c}} + y^{\uparrow}_{\text{BR}\hat{c}}) &
\end{aligned}
$$

where γ is a fusion parameter of the classification and action scores. Here each score y is a value before the softmax normalization.

5.2 Initial Glance

To detect multiple instances in an image, we collect regions called initial glances as presented in Sec. 3.4. A required condition for a region to be an initial glance is that the region should contain an entire object body with sufficient surrounding contexts. Since we recurrently shrink an initial glance to a target object with movement actions, it is important for the initial glance not to be truncating a target object. To this end, given a lot of candidate regions, we choose regions which satisfy the following conditions as initial glances

$$\hat{c} \neq N+1 \quad \text{and} \quad \hat{a}_{\text{TL}\hat{c}} = {}_{\searrow\text{TL}} \quad \text{and} \quad \hat{a}_{\text{BR}\hat{c}} = {}_{\nwarrow\text{BR}} \qquad (5)$$

which indicate that an initial glance should not be a background, and should not be truncating the target object as shown in Fig. 5. Here, $\hat{c}$ is a class prediction from the classification layer, and $(\hat{a}_{\text{TL}\hat{c}}, \hat{a}_{\text{BR}\hat{c}})$ are the action predictions from the action layers of the predicted class $\hat{c}$.

To boost speed and recall of the initial glance mining, we feed multi-scale multi-aspect ratio images to our fully convolutional attention model and pick out the initial glances over the resulting action maps as illustrated in Fig. 6. We also follows the data-driven method to determine the scaling factors which has presented in Sec. 3.4. The scaling factors $\{\mathbf{s}_k \mid k = 1 \cdots K\}$ are estimated from all the ground truth bounding boxes regardless of their classes in a training image set.

5.3 Initial Detection and Refinement

Start from the initial glances, we iteratively shrink the boundaries with the movement actions $(\hat{a}_{\text{TL}\hat{c}}, \hat{a}_{\text{BR}\hat{c}})$ of the predicted class $\hat{c}$. This procedure is repeated until a region is rejected with $\hat{c}$=N+1, or meets the following conditions

$$\hat{c} \neq N+1 \quad \text{and} \quad \hat{a}_{\text{TL}\hat{c}} = \bullet_{\text{TL}} \quad \text{and} \quad \hat{a}_{\text{BR}\hat{c}} = \bullet_{\text{BR}} \qquad (6)$$

which indicate that the final region is a foreground class and terminates with the stop actions at both corners. We back-project the final regions to the corresponding boxes in the original input image domain, and these boxes are then merged with an initial IoU μ_0. As a refinement step, we rescale the initial detection boxes with a rescaling factor β, and shrink them again until they terminate with the stop actions. We finally merge the detection boxes with a final IoU μ_1. Fig. 7 shows real examples for these procedures.

5.4 Training

To make the multi-class attention model recognize the class of an object and proper actions for that object, we train the model with random regions which evenly cover the various object classes, their required actions and backgrounds. Because ground-truths including bounding-boxes and classes are given, we can automatically determine the optimal action label for each corner of a random region. We follows the method in Fig. 3 to generate random regions. When we compose a mini-batch for training, we select the object regions and the background regions with an equal probability as described in Sec. 3.2.

Given a region with a class label t_{cls} and a pair of action labels $(t_{\text{TL}}, t_{\text{BR}})$, we can define three log-softmax losses at a classification layer and the two action layers of the class. Zero-losses are given to the action layers of the other classes. We define a final loss ℓ by combining a classification loss ℓ_{cls} and the two action losses $(\ell_{\text{TL}}, \ell_{\text{BR}})$ such as

$$
\begin{aligned}
\ell = \lambda \cdot \ell_{\text{cls}} + \frac{1-\lambda}{2} \cdot (\ell_{\text{TL}} + \ell_{\text{BR}}) \quad \text{s.t.} & \\
\ell_{\text{cls}} = \ell_{\text{softmax}}\left(\mathbf{y}_{\text{cls}}, t_{\text{cls}}\right) & \\
\ell_{\text{TL}} = \sum_{c=1}^{N} \mathbb{1}(c, t_{\text{cls}}) \cdot \ell_{\text{softmax}}\left(\mathbf{y}_{\text{TL}c}, t_{\text{TL}}\right) & \quad (7)\\
\ell_{\text{BR}} = \sum_{c=1}^{N} \mathbb{1}(c, t_{\text{cls}}) \cdot \ell_{\text{softmax}}\left(\mathbf{y}_{\text{BR}c}, t_{\text{BR}}\right) & \\
\ell_{\text{softmax}}\left(\mathbf{y}, t\right) = -y_t + \log \sum_i e^{y_i} &
\end{aligned}
$$

where λ is a constant from a range $[0, 1]$, and $\mathbb{1}(c, t)$ is 1 if c equals to t but 0 for otherwise. If a region is a background t_{cls}=N+1, the losses from all the action layers become zero.

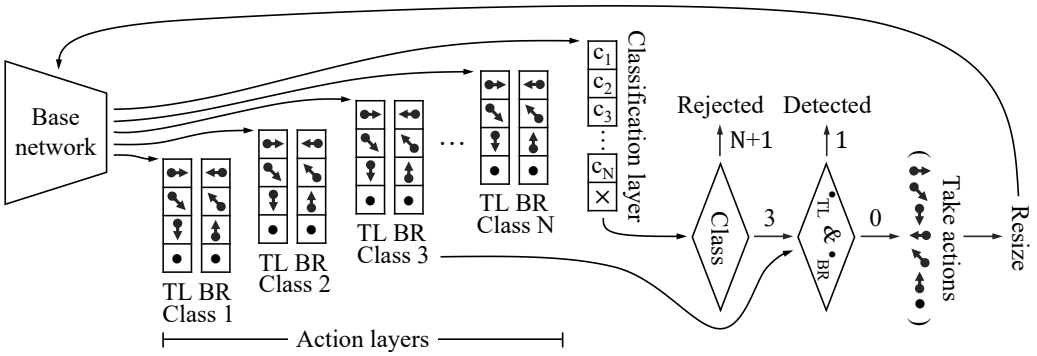


Fig. 8. An attention model extended to multiple object classes. A pair of TL and BR action layers are defined for each class to provide class-specific actions. A classification layer is also defined to recognize the class of an input object.

6 Multi-Class Experiments

In this section, we perform multi-class object detection task on public datasets to comprehensively verify our detection mechanism extended to multiple classes.

6.1 Datasets

For primary evaluation, we choose PASCAL VOC 2007 and 2012 [54] dataset. We follows the standard protocol of these sets as described in Sec. 4.1. Also, we select ILSVRC CLS-LOC dataset [3] to verify our detection capability for large number of classes with large-scale data. This dataset includes images of 1,000 object classes and divide into 1.3M training images, 50K validation images and 100K test images. All the images are annotated with object classes but only a part of the training set, 524K images, contain bounding box annotations. We use these 524K images to train the attention model. To evaluate localization, we submitted the localization results on the `test` set to ILSVRC'15 evaluation server and received the Top-5 LOC error. Top-5 LOC error reflects classification and localization errors at once from top-5 predictions for each image. This metric is defined in Sec. 4.2 of [3].

6.2 Base Network

We also use VGG-M [48] and VGG-16 [23] for the PASCAL VOC series in the same way presented in Sec. 4.2. For ILSVRC CLS-LOC dataset, we use GoogLeNet designed by Szegedy *et al.* [24]. GoogLeNet is also a deep model that has 22 convolution layers but much faster than VGG-16 while showing a comparative performance of 12.9% Top-5 error with a single model. We choose this model to speed up training over the large-scale data. Before training, we pick out all the pre-trained layers of GoogLeNet except for the last classification layer and stack our action layers on top of those.

6.3 Parameters

All the parameters necessary for training and inference are equal to those of the single class detection mechanism, which has been presented in Sec. 4.3, except for the following; we use the number of scaling factors K of 7 and the rescaling factor β of 8.0 for the evaluation with ILSVRC CLS-LOC dataset. The loss fusion parameter λ and the score fusion parameter γ are equally set to 0.5.

6.4 Results and analysis

We evaluate our multi-class detection model with PASCAL VOC 2007 and 2012. The per-class APs(%) of our method at these datasets are listed in Table 3. When the base network of our multi-class attention model is VGG-M, our method achieves 62.3% and 59.8% for each dataset without the refinement step of Fig. 7-(c∼e). After the refinement step, noted by "AttentionNet-Refine", the performances are improved to 63.3% and 60.6%. Different from the single class attention model in Table 2, the refinement step gives a non-negligible benefit coming from re-localizations. Note, our multi-class attention model is reused for the refinement. When the model size increases to the VGG-16 model, performances also significantly increase to 70.1% and 64.4% for each dataset. With the refinement step, the performances slightly increase to 70.7% and 65.6%.

As we have presented in Sec. 3.1, the attention model tells each corner which direction to move. The corner then moves l pixels along that direction. The input region is always resized to 224×224 size and the size of a movement l is constant, so the movement in the original image domain becomes smaller then that of the previous stage, as shown in Fig. 1. Let us take a look at how performance varies with the size of the movement l and how many movements are required to detect an object instance. Table 4 is summarizing the results. The smaller size of movement requires more movements to detect an instance. We expected the accuracy to decrease as the size of movement increases. However, in reality, the accuracy rather increases, and then it starts to decrease later. The reason is because the more iteration, the more chance the attention model has of making false negative decisions $\{\times_{\mathrm{TL}}, \times_{\mathrm{BR}}\}$. The results in both base networks show a similar tendency. When the size of movement is 15 pixels, our method shows the best accuracy of 62.9% and 71.0% in each base network. These results beat Faster R-CNN [18], which show 59.9% with ZFNet and 69.9% with VGG-16, by margins of 3.0% and 1.1%, respectively. However, because the size of 30 pixels requires much less number of movements than that of 15 pixels but shows good performances, we set the size to 30 pixels for the initial detection in all the other experiments.

We can regard our method as a box regression by stacked classifications. We therefore compare ours with the traditional detection-by-regression methods such as [13], [16]

Method	Base net	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motor	person	plant	sheep	sofa	train	tv	mAP (%)
AttentionNet	VGG-M	74.6	71.0	56.8	52.0	47.8	72.1	79.9	67.8	35.9	66.3	49.4	64.5	72.9	70.0	71.4	42.4	64.6	53.5	70.1	62.6	62.3
AttentionNet-Refine	VGG-M	76.5	73.0	57.8	52.3	51.8	71.9	80.0	67.3	37.1	67.8	49.4	65.9	72.3	69.0	72.0	43.2	66.3	53.6	71.9	66.1	**63.3**
AttentionNet	VGG-16	77.4	77.1	70.3	57.4	58.9	76.7	85.5	75.4	45.0	80.1	61.3	76.2	78.3	75.3	78.2	47.7	73.4	63.3	72.5	71.6	70.1
AttentionNet-Refine	VGG-16	79.1	77.6	70.2	58.0	60.0	75.8	85.5	75.9	47.6	79.7	61.6	76.9	78.6	76.0	80.1	47.0	73.9	64.3	74.1	72.5	**70.7**

(a) Our results on PASCAL VOC 2007 `test` set. All the method use `trainval'07` set for training.

Method	Base net	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motor	person	plant	sheep	sofa	train	tv	mAP (%)
AttentionNet	VGG-M	77.7	66.1	61.5	44.8	46.8	70.9	72.0	73.7	32.6	59.4	40.0	71.4	64.8	70.8	70.6	38.5	69.5	38.8	70.0	56.8	59.8
AttentionNet-Refine	VGG-M	77.8	66.4	60.9	44.9	48.5	71.8	72.3	75.3	34.5	59.4	40.4	71.5	65.5	71.3	72.4	38.3	70.9	40.9	70.1	59.3	**60.6**
AttentionNet	VGG-16	77.0	69.4	64.1	51.7	53.9	73.1	76.3	77.0	39.3	69.4	42.2	74.9	72.6	73.5	76.2	46.5	73.2	44.1	73.2	61.1	64.4
AttentionNet-Refine	VGG-16	79.1	68.9	65.5	52.3	55.9	73.5	76.5	79.1	42.3	70.9	42.7	76.8	73.9	73.3	79.2	48.1	74.1	44.9	73.7	62.2	**65.6**

(b) Our results on PASCAL VOC 2012 `test` set. All the method use `trainval'12` set for training.

TABLE 3
Our multi-class detection performances on (a) PASCAL VOC 2007 `test` set and (b) PASCAL VOC 2012 `test` set.

Base net	Motion size l (Pixels)	5	10	15	30	50
VGG-M	Mean number of movements	23.5	12.9	9.3	5.5	4.0
	mAP (%) of initial detection	60.6	62.6	**62.9**	62.3	57.6
VGG-16	Mean number of movements	26.5	14.3	10.2	6.0	4.3
	mAP (%) of initial detection	69.3	70.4	**71.0**	70.1	62.5

TABLE 4
Initial detection performances and mean number of movements per instance as the motion size increases in PASCAL VOC 2007 `test` set.

Methods	VOC'07 mAP(%)	VOC'12 mAP(%)
AttentionNet	62.3	59.8
AttentionNet-Refine	**63.3**	**60.6**
Regression (AlexNet) [13]	30.5	-
DeepMultiBox (AlexNet) [16]	29.2	-
R-CNN (AlexNet) [15]	58.5	**53.3**
R-CNN + SPP (ZFNet) [21]	59.2	-
Fast R-CNN (CaffeNet) [17]	58.4	-
Faster R-CNN (ZFNet) [18]	**59.9**	-
Q-Learning (ZFNet) [27]	46.1	-
Q-Learning (AlexNet) [28]	-	27.0

(a) Comparisons with 8-layered networks.

Methods	VOC'07 mAP(%)	VOC'12 mAP(%)
AttentionNet	70.1	64.4
AttentionNet-Refine	**70.7**	**65.6**
AttentionNet + Faster R-CNN	74.2	71.0
AttentionNet-Refine + Faster R-CNN	**75.8**	**71.5**
R-CNN (VGG-16) [15]	66.0	62.4
Fast R-CNN (VGG-16) [17]	66.9	65.7
Faster R-CNN (VGG-16) [18]	69.9	67.0
YOLO (26 layers) [19]	-	57.9*
SSD (VGG-16) [20]	71.6	**74.9***
MR-CNN + S-CNN + Loc. (VGG-16) [52]	**74.9**	70.7

*trained on a superset of `trainval'07+test'07+trainval'12`.
(b) Comparisons with networks composed of more than 16 layers.

TABLE 5
Object detection performances on PASCAL VOC 2007/2012 `test` set.

of which Table 5-(a) lists performances. The network of [13] estimates a object box mask for each sliding window and shows 30.0% performance. The network of [16] directly produces box coordinates of class-agnostic object proposals and shows 29.2% performance. Our method beats these approaches with a large margin more than +30%. YOLO [19] is a recent detection-by-regression method in which a 26-layered network estimates a bounding-box for each grid cell of a convolutional feature map. As listed in Table 5-(b), this method shows a much lower performance of 57.9% than our 65.6% in VOC 2012 even if this is trained with a *superset* composed of `trainval'07+test'07+trainval'12`. SSD [20] is a variant of YOLO but it operates on a multiple scale feature maps, and shows the state-of-the-art performances of 71.6% and 74.9% in each dataset. Our performance of 70.1% is slightly worse than that of SSD in VOC 2007, and SSD outperforms ours with a large margin in VOC 2012 but SSD is also trained with the same *superset* used in YOLO.

R-CNN [15] and its advanced variants such as Fast R-CNN [17] and Faster R-CNN [18] are typical bottom-up approaches. Let us compare our top-down approach with them. As shown in Table 5, the performances of the original R-CNN are significantly worse than ours in general. Our method also beats Fast and Faster R-CNNs in VOC 2007. In VOC 2012, the performance of our method is similar to that of Fast RCNN and lower than that of Faster R-CNN. Our top-down method demonstrated comparable performances compared to these state-of-the-art bottom-up method.

Bottom-up and top-down approaches are complementary to each other. A bottom-up method is feed-forward and efficient. Also, detection from proposals is more advantageous in recall. In contrast, our top-down method is recursive so slower than that but our high-level action cascade is more advantageous in the scene-level context so results in less false positives. Thus, we can expect a large performance improvement when we mix the two complementary approaches. We rescale our scores, and merge all the boxes and their scores with an IoU of 0.5. As shown in the second block of Table 5-(b), the results are 75.8% in VOC 2007 and 71.5% in VOC 2012 with gains of +5.1% and +5.9%. These results clearly demonstrate the potential of mixing the top-down and bottom-up methods.

Similar to ours, [27], [28] also use an agent providing actions but they depend on the reinforcement Q-learning [34] to train that because selecting an optimal action given a state is ambiguous. In VOC 2007, [27] shows 46.1% which is much worse than our 63.3%. Also, in VOC 2012, [28] shows 27.0% while ours is 60.6%. Both of these methods are successful top-down methods using reinforcement learning

Methods	IoU=0.5 mAP(%)	IoU=0.7 mAP(%)	mAP(%) drop
AttentionNet	70.1	49.7	−20.4
AttentionNet (l=15)	**71.0**	52.3	−18.7
AttentionNet-Refine	70.7	**53.1**	**−17.6**
R-CNN NoBBReg (VGG-16) [15]	60.6	30.8	−29.8
R-CNN (VGG-16) [15]	66.0	35.2	−30.8
R-CNN + Bayesian (VGG-16) [55]	68.4	43.7	−24.7
Faster R-CNN (VGG-16) [18]	69.9	46.0	**−23.9**
MR-CNN + S-CNN + Loc. (VGG-16) [52]	**74.9**	**48.4**	−26.5

TABLE 6
Object detection performances with different IoU thresholds (standard 0.5 and strict 0.7) for positive detection in PASCAL VOC 2007 `test` set. The base network of all the methods is VGG-16.

Year	Loc. methods	Base net for loc.	Model depth (#conv)	Loc. net ensemble	Top-5 LOC err (Top-5 CLS err)
2013	Regression [14]	OverFeat	4	Yes	0.2988 (0.1568)
2014	Regression [24]	GoogLeNet	22	Yes	0.2644 (0.1483)
2014	Regression [23]	VGG-16	16	Yes	0.2532 (0.0741)
2015	Faster R-CNN [4]	ResNet	**152**	Yes	**0.0902** (**0.0357**)
2015	AttentionNet	GoogLeNet	22	**No**	0.1473 (0.0792)

TABLE 7
Performance comparison of 1,000-class object localization on ILSVRC CLS-LOC `test` set.

for detection, but the reinforcement method has a high variance in the gradient of the expected reward so their performances are far from the state-of-the-art methods. In contrast, our top-down mechanism first demonstrates competitive performances compared to the recent bottom-up methods.

Our detection method has another strength driven from the iterative action classifications. Compared to the bottom-up methods, our box localization is more accurate because it combines several weak actions to estimate a final detection box. Table 6 summarizes the performance comparisons with different IoU thresholds in PASCAL VOC 2007. In our method without the refinement step, the performance drop is -20.4 as the IoU threshold increases from the typical 0.5 to a strict 0.7. When we conduct the refinement step, we observe a smaller performance drop of -17.6 because we have one more chance to correct mis-localizations such as the traditional bounding-box regression R-CNN does. The performance drop of all the other methods is much larger than that of ours, so our method significantly beats all those when IoU threshold is 0.7.

We finally present the localization performance, measured by Top-5 LOC error [3], from a large-scale experiment with ILSVRC CLS-LOC dataset. The comparison between our result[2] and the existing winning methods is summarized in Table 7. All entries being compared with ours are top methods in the ILSVRC localization task. Also, all these are using the bottom-up approaches for localization. [14] is the winning method in 2013, and [23], [24] are the first and second places in 2014, respectively. These three methods localize object by the bounding-box regression over dense sliding windows. Compared with these methods, our method shows much lower error of 0.1473 with a large margin of more than 0.1. More recently, He *et al.* [4] won this challenge in 2015 with Faster R-CNN [18] based on the very deep residual network composed of 152 layers. This method shows a small localization error of 0.0902 which is lower than ours. However, this performance gap mainly comes from the classification performance since the localization error is including the classification error as well. The classification error of the residual network is 0.0357, which is already 2-times smaller than our classification error of 0.0792. Also all the other methods in Table 7 ensemble multiple localization networks but our localization only depends on *a single multi-class attention model*.

2. http://image-net.org/challenges/ilsvrc+mscoco2015

7 Conclusions

In this paper, we have proposed a novel method for object detection. We adopted a well-studied classification technique for object detection and presented a weak attention model to get high-level actions from that to get closer to a target object. Since we actively explorer an exact bounding-box of a target object in a top-down approach, we do not suffer from the quality of initial object proposals and also take the scene-level context into consideration.

Through this study, we have an important observation that our top-down approach is complementary to the previous state-of-the-art method using a bottom-up approach, therefore combining the two approaches boosts the performance of object detection. Thus, we believe the research on top-down approaches and combining them with the bottom-up methods will likely contribute to the next direction for object detection.

Acknowledgments

This work was supported by the Technology Innovation Program (No. 10048320), funded by Korea government (MOTIE). This work was also supported by the National Research Foundation of Korea (No. 2010-0028680), funded by Korea government (MSIP).

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[2] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[3] 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.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[5] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *arXiv preprint arXiv:1512.00567*, 2015.

[6] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1. IEEE, 2001, pp. I–511.

[7] R. Lienhart and J. Maydt, "An extended set of haar-like features for rapid object detection," in *Image Processing. 2002. Proceedings. 2002 International Conference on*, vol. 1. IEEE, 2002, pp. I–900.

[8] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman, "Multiple kernels for object detection," in *2009 IEEE 12th international conference on computer vision*. IEEE, 2009, pp. 606–613.

[9] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: A benchmark," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 304–311.

[10] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 9, pp. 1627–1645, 2010.

[11] L. Bourdev, S. Maji, T. Brox, and J. Malik, "Detecting people using mutually consistent poselet activations," in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 168–181.

[12] 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.

[13] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," in *Advances in Neural Information Processing Systems*, 2013, pp. 2553–2561.

[14] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv preprint arXiv:1312.6229*, 2013.

[15] 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*, 2014, pp. 580–587.

[16] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, "Scalable object detection using deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2147–2154.

[17] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, 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," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[19] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[20] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. Reed, "Ssd: Single shot multibox detector," in *European Conference on Computer Vision*, 2016.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 37, no. 9, pp. 1904–1916, 2015.

[22] W. Ouyang, P. Luo, X. Zeng, S. Qiu, Y. Tian, H. Li, S. Yang, Z. Wang, Y. Xiong, C. Qian *et al.*, "Deepid-net: multi-stage and deformable deep convolutional neural networks for object detection," *arXiv preprint arXiv:1409.3505*, 2014.

[23] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[24] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[25] A. Gonzalez-Garcia, A. Vezhnevets, and V. Ferrari, "An active search strategy for efficient object class detection," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2015, pp. 3022–3031.

[26] D. Yoo, S. Park, J.-Y. Lee, A. S. Paek, and I. So Kweon, "Attentionnet: Aggregating weak directions for accurate object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2659–2667.

[27] J. C. Caicedo and S. Lazebnik, "Active object localization with deep reinforcement learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2488–2496.

[28] S. Mathe, A. Pirinen, and C. Sminchisescu, "Reinforcement learning for visual object detection," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[29] C. Gu, J. J. Lim, P. Arbeláez, and J. Malik, "Recognition using regions," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 1030–1037.

[30] M.-M. Cheng, Z. Zhang, W.-Y. Lin, and P. Torr, "Bing: Binarized normed gradients for objectness estimation at 300fps," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3286–3293.

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

[32] J. Hosang, R. Benenson, P. Dollár, and B. Schiele, "What makes for effective detection proposals?" *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 4, pp. 814–830, 2016.

[33] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 248–255.

[34] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998, vol. 1, no. 1.

[35] J. Ba, V. Mnih, and K. Kavukcuoglu, "Multiple object recognition with visual attention," *arXiv preprint arXiv:1412.7755*, 2014.

[36] V. Mnih, N. Heess, A. Graves *et al.*, "Recurrent models of visual attention," in *Advances in Neural Information Processing Systems*, 2014, pp. 2204–2212.

[37] K. Gregor, I. Danihelka, A. Graves, D. Rezende, and D. Wierstra, "Draw: A recurrent neural network for image generation," in *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 2015, pp. 1462–1471.

[38] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 2017–2025.

[39] T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang, "The application of two-level attention models in deep convolutional neural network for fine-grained image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 842–850.

[40] C. Cao, X. Liu, Y. Yang, Y. Yu, J. Wang, Z. Wang, Y. Huang, L. Wang, C. Huang, W. Xu *et al.*, "Look and think twice: Capturing top-down visual attention with feedback convolutional neural networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2956–2964.

[41] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 2015, pp. 2048–2057.

[42] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville, "Describing videos by exploiting temporal structure," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4507–4515.

[43] P. Agrawal, R. Girshick, and J. Malik, "Analyzing the performance of multilayer neural networks for object recognition," in *Computer Vision–ECCV 2014*. Springer, 2014, pp. 329–344.

[44] M. Everingham, L. Van Gool, C. K. I. 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.

[45] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler, "Joint training of a convolutional network and a graphical model for human pose estimation," in *Advances in neural information processing systems*, 2014, pp. 1799–1807.

[46] D. Yoo, S. Park, J.-Y. Lee, and I. Kweon, "Multi-scale pyramid pooling for deep convolutional representation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2015, pp. 71–80.

[47] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[48] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *British Machine Vision Conference*, 2014.

[49] G. Gkioxari, B. Hariharan, R. Girshick, and J. Malik, "Using k-poselets for detecting people and localizing their keypoints," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3582–3589.

[50] Y. Jiang and J. Ma, "Combination features and models for human detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 240–248.

[51] L. Bourdev, F. Yang, and R. Fergus, "Deep poselets for human detection," *arXiv preprint arXiv:1407.0717*, 2014.

[52] S. Gidaris and N. Komodakis, "Object detection via a multi-region and semantic segmentation-aware cnn model," in *Proceedings of the*

IEEE International Conference on Computer Vision, 2015, pp. 1134–1142.

[53] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer vision–ECCV 2014*. Springer, 2014, pp. 818–833.

[54] 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.

[55] Y. Zhang, K. Sohn, R. Villegas, G. Pan, and H. Lee, "Improving object detection with deep convolutional networks via bayesian optimization and structured prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 249–258.

Donggeun Yoo received BS degree in 2011 and MS degree in 2013 in School of Electrical Engineering, KAIST, South Korea. In present, he is a PhD student in the same school and department. His research interests include representation learning, learning with large-scale data, unsupervised learning and visual recognition. He is a student member of the IEEE.

Sunggyun Park received BS degree in 2010 and MS degree in 2012 in Department of Industrial and Systems Engineering, KAIST, South Korea. In present, he is a PhD student in the same school and department, and also a co-founder and a research scientist at Lunit Inc, South Korea. His research interests include stochastic process for product pricing and machine learning for problems, which are related to operations management and computer vision. He is a student member of the IEEE.

Kyunghyun Paeng received BS degree in 2011 in School of Electrical Engineering, KAIST, South Korea. Currently, he is a PhD student in the same school and department, and also a co-founder and a research scientist at Lunit Inc, South Korea. His research interests include 3D computer vision, visual recognition, weakly supervised learning, and medical image analysis with deep learning.

Joon-Young Lee received BS degree in electrical and electronic engineering from Yonsei University, South Korea, in 2008, and MS degree and PhD degree in 2009 and 2015 respectively, in School of Electrical Engineering from KAIST, South Korea. He is currently working in Adobe Research, San Jose, CA. His research interests include photometric methods in computer vision, image enhancement, computational photography, and deep learning for video analysis. He received the Samsung HumanTech Paper Award and the Qualcomm Innovation Award. He is a member of the IEEE.

In So Kweon received BS and MS degrees in mechanical design and production engineering from Seoul National University, Korea, in 1981 and 1983, respectively, and PhD degree in robotics from the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 1990. He was with Toshiba R&D Center, Japan, and joined the Department of Automation and Design Engineering, KAIST, Korea in 1992, where he is currently a professor with the Department of Electrical Engineering. He received the best student paper runner-up award at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 09). His research interests are in camera and 3-D sensor fusion, color modeling and analysis, visual tracking, and visual SLAM. He was the program co-chair for the Asian Conference on Computer Vision (ACCV 07) and was the general chair for the ACCV 12. He is also on the editorial board of the International Journal of Computer Vision. He is a member of the IEEE and the KROS.