# Hierarchical Object Detection with Deep Reinforcement Learning

**Míriam Bellver Bueno**
Barcelona Supercomputing Center (BSC)
Barcelona, Catalonia/Spain
miriam.bellver@bsc.es

**Xavier Giró-i-Nieto**
Image Processing Group
Universitat Politècnica de Catalunya (UPC)
Barcelona, Catalonia/Spain
xavier.giro@upc.edu

**Ferran Marqués**
Image Processing Group
Universitat Politècnica de Catalunya (UPC)
Barcelona, Catalonia/Spain
ferran.marques@upc.edu

**Jordi Torres**
Barcelona Supercomputing Center (BSC)
Universitat Politècnica de Catalunya
Barcelona, Catalonia/Spain
jordi.torres@bsc.es

## Abstract

We present a method for performing hierarchical object detection in images guided by a deep reinforcement learning agent. The key idea is to focus on those parts of the image that contain richer information and zoom on them. We train an intelligent agent that, given an image window, is capable of deciding where to focus the attention among five different predefined region candidates (smaller windows). This procedure is iterated providing a hierarchical image analysis.We compare two different candidate proposal strategies to guide the object search: with and without overlap. Moreover, our work compares two different strategies to extract features from a convolutional neural network for each region proposal: a first one that computes new feature maps for each region proposal, and a second one that computes the feature maps for the whole image to later generate crops for each region proposal. Experiments indicate better results for the overlapping candidate proposal strategy and a loss of performance for the cropped image features due to the loss of spatial resolution. We argue that, while this loss seems unavoidable when working with large amounts of object candidates, the much more reduced amount of region proposals generated by our reinforcement learning agent allows considering to extract features for each location without sharing convolutional computation among regions. Source code and models are available at `https://imatge-upc.github.io/detection-2016-nipsws/`.

## 1 Introduction

When we humans look at an image, we always perform a sequential extraction of information in order to understand its content. First, we fix our gaze to the most salient part of the image and, from the extracted information, we guide our look towards another image point, until we have analyzed all its relevant information. This is our natural and instinctive behavior to gather information from our surroundings.

Traditionally in computer vision, images have been analyzed at the local scale following a sliding window scanning, often at different scales. This approach analyses the different image parts independently, without relating them. Just by introducing a hierarchical representation of the image, we can more easily exploit the relationship between regions. We propose to use a top-down scanning

Deep Reinforcement Learning Workshop (NIPS 2016), Barcelona, Catalonia/Spain.

which firstly takes a global view of the image to sequentially focus on the local parts that contain the relevant information (e.g. objects or faces).

Our algorithm is based on an intelligent agent trained by reinforcement learning that is capable of making decisions to detect an object in a still image, similarly to [2]. The agent first analyzes the whole image, and decides in which region of the image to focus among a set of predefined ones. Inspired by [9], our agent can top-down explore a set of five different predefined region candidates: four regions representing the four quadrants plus a central region. Two different strategies have been studied: proposing overlapping or non-overlapping candidates. The agent stops its search when it finds an object. Reinforcement learning is useful for our task because there is no single way of completing it. The agent can explore the hierarchical representation in different ways and still achieve its goal. Then, instead of programming every step that the agent should do, we train it so that it makes decisions under uncertainty to reach its target. Notice that the final goal of object detection is to define a bounding box around the object and that, in our work, these bounding boxes are limited to the predefined regions in the hierarchy.

Most state of the art solutions for object detection analyze large amounts of region proposals. These algorithms need to leverage the bottleneck of describing all these proposals by reusing convolutional feature maps of the whole image. In our work though, as the reinforcement learning agent and the hierarchy allow us to analyze a very reduced number of regions, we can feed each region visited by the agent through a convolutional network to extract its features, allowing us to work with region representations of higher spatial resolution, which are also more informative than those cropped from a feature map of the whole image. To study this trade-off, we have trained and compared two different models based on each of these two principles: the *Image-Zooms* model, which extracts descriptors at each region, and the *Pool45-Crops* model, which reuses feature maps for different regions of the same image.

The first contribution of our work is the introduction of a hierarchical representation to top-down (zoom-in) guide our agent through the image. We explore how the design of the hierarchy affects the detection performance and the amount of visited regions. The second contribution is the study between extracting features for each region instead of reusing feature maps for several locations. We show the gain of the region-specific features for our scheme, and argue that the computational overhead is minor thanks to the very reduced amount of regions considered by the agent.

## 2 Related Work

Reinforcement learning is a powerful tool that has been used in a wide range of applications. The most impressive results are those from DeepMind [11], who have been able to train an agent that plays Atari 2600 video games by observing only their screen pixels, achieving even superhuman performance. Also they trained a computer that won the Go competition to a professional player for the first time [16]. More specifically to traditional computer vision tasks, reinforcement learning has been applied to learn spatial glimpse policies for image classification [10, 1], for captioning [20] or for activity recognition [21]. It has also been applied for object detection in images [2], casting a Markov Decision Process, as our approach does.

The traditional solutions for object detection are based on region proposals, such as Selective Search [19], CPMC [3] or MCG [13] and other methods based on sliding windows such as EdgeBoxes [23]. The extraction of such proposals was independent of the classifier that would score and select which regions compose the final detection. These methods are computationally expensive because rely on a large number of object proposals. Then the first trends based on Convolutional Neural Networks appeared, such as Fast R-CNN [6], that already studied how to share convolutional computation among locations, as they identified that the extraction of features for the hypothesized objects was the bottleneck for object detection.

More recent proposals such as Faster R-CNN [15] have achieved efficient and fast object detection by obtaining cost-free region proposals sharing full-image convolutional features with the detection network. Directly predicting bounding boxes from an image is a difficult task, and for this reason, approaches such as Faster R-CNN rely on a number of reference boxes called anchors, that facilitate the task of predicting accurate bounding boxes by regressing these initial reference boxes. One key of our approach is the refinement of bounding box predictions through the different actions selected by the reinforcement learning agent. Besides Faster R-CNN or other approaches such as YoLo

[14] or MultiBox [4] based on anchors, there are other works that are based on the refinement of predictions. Yoo et al. [22] propose the AttentionNet. They cast an object detection problem as an iterative classification problem. AttentionNet predicts a number of weak directions pointing to the target object so that a final accurate bounding box is obtained. The state-of-the-art in object detection is the Single Shot MultiBox Detector (SSD) [8], which works with a number of default boxes of different aspect ratios and scales per each feature map location, and also adjusts them to a better match to the object shape.

Another approach that supports this idea is the Active Object Localization method proposed by Caicedo and Lazebnik [2]. Their method trains an intelligent agent using deep reinforcement learning that is able to deform bounding boxes sequentially until they fit the target bounding box. Each action that the agent does to the bounding box can change its aspect ratio, scale or position. Our main difference to this approach is that we add a fixed hierarchical representation that forces a top-down search, so that each action zooms onto the predicted region of interest.

How to benefit from super-resolution has also been studied by other works. In the paper of Lu et al. [9] a model is trained to determine if it is required to further divide the current observed region because there are still small objects on it, and in this case, each subregion is analyzed independently. Their approach could also be seen as hierarchical, but in this case they analyze each subregion when the zoom prediction is positive, whereas we just analyze the subregion selected by the reinforcement learning agent. On contrast to their proposal, we analyze fewer regions and then we can afford extracting high-quality descriptors for each of them, instead of sharing convolutional features.

## 3 Hierarchical Object Detection Model

In this work we define the object detection problem as the sequential decision process of a goal-oriented agent interacting with a visual environment that is our image. At each time step the agent should decide in which region of the image to focus its attention so that it can find objects in a few steps. We cast the problem as a Markov Decision Process, that provides a framework to model decision making when outcomes are partly uncertain.

### 3.1 MDP formulation

In order to understand the models for the object detection task that we have developed, we first define how the Markov Decision Process is parameterized.

**State** The state is composed by the descriptor of the current region and a memory vector. The type of descriptor defines the two models we compare in our work: the *Image-Zooms* model and the *Pool45-Crops* model. These two variations are explained in detail in Section 3.3. The memory vector of the state captures the last 4 actions that the agent has already performed in the search for an object. As the agent is learning a refinement of a bounding box, a memory vector that encodes the state of this refinement procedure is useful to stabilize the search trajectories. We encode the past 4 actions in a one-shot vector. As there are 6 different actions presented in the following section, the memory vector has 24 dimensions. This type of memory vector was also used in [2].

**Actions** There are two types of possible actions: *movement actions* that imply a change in the current observed region, and the *terminal* action to indicate that the object is found and that the search has ended. One particularity of our system is that each movement action can only transfer the attention top-down between regions from a predefined hierarchy. A hierarchy is built by defining five subregions over each observed bounding box: four quarters distributed as $2x2$ over the box and a central overlapping region. We have explored two variations of this basic $2x2$ scheme: a first one with *non-overlapped* quarters (see Figure 1), and a second one with *overlapped* quarters (see Figure 2), being the size of a subregion 3/4 of its ancestor. Then, there are five movement actions, each one associated to one of the yellow regions. If, on the other hand, the terminal action is selected, there is no movement and the final region is the one marked with blue.

**Reward** The reward functions used are the ones proposed by Caicedo and Lazebnik [2]. The reward function for the movement actions can be seen in Equation 1 and the reward function for the terminal action in Equation 2. Given a certain state *s*, a reward is given to those actions that move

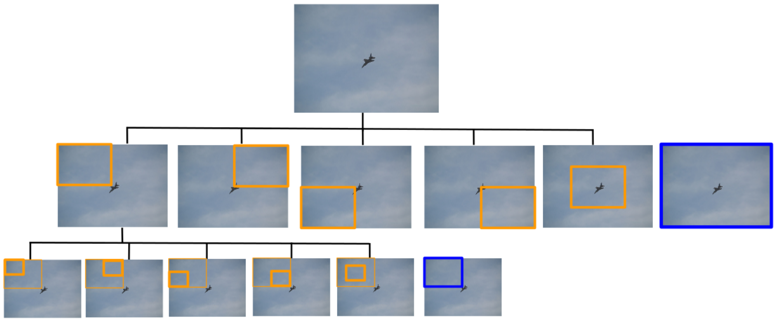

Figure 1: Image hierarchy of three levels with non-overlapped quarters

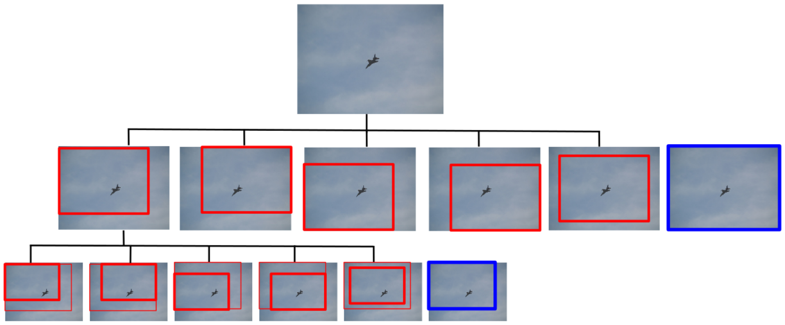

Figure 2: Image hierarchy of three levels with overlapped quarters

towards a region *b'* with a greater Intersection Over Union (IoU) with the ground truth $g$ than the region $b$ considered at the previous step. Otherwise, the actions are penalized. For the trigger action, the reward is positive if the Intersection Over Union of the actual region $b$ with the ground truth is greater than a certain threshold $\tau$, and negative otherwise. We consider $\tau$ = 0.5 , because it is the threshold for which a detection is considered positive, and $\eta$ is 3, as in [2].

$$R_m(s, s') = sign(IoU(b', g) - IoU(b, g)) \tag{1}$$

$$R_t(s, s') = \begin{cases} +\eta & if\ IoU(b, g) \geq \tau \\ -\eta & otherwise \end{cases} \tag{2}$$

### 3.2 Q-learning

The reward of the agent depending on the chosen action $a$ at state $s$ is governed by a function *Q(s,a)*, that can be estimated with Q-learning. Based on *Q(s,a)*, the agent will choose the action that is associated to the highest reward. Q-learning iteratively updates the action-selection policy using the Bellman equation 3, where $s$ and $a$ are the current state and action correspondingly, $r$ is the immediate reward and $max_a Q(s', a')$ represents the future reward. Finally $\gamma$ represents the discount factor. In our work, we approximate the Q-function by a Deep Q-network trained with Reinforcement Learning [11].

$$Q(s, a) = r + \gamma max_a Q(s', a') \tag{3}$$

### 3.3 Model

In our work we study two different approaches for visual feature extraction, which are used to train a Deep Q-Network. Figure 3 depicts the two variations with the common reinforcement learning network.

We compare two models to extract the visual features that define the state of our agent: the *Image-Zooms* model and the *Pool45-Crops* model. For the *Image-Zooms* model, each region is resized to 224x224 and its visual descriptors correspond to the feature maps from Pool5 layer of VGG-16 [17]. For the *Pool45-Crops* model, the image at full-resolution is forwarded into VGG-16 [17] through Pool5 layer. As Girshick [6], we reuse the feature maps extracted from the whole image for all the regions of interest (ROI) by pooling them (ROI pooling). As in SSD [8], we choose which feature map to use depending on the scale of the region of interest. In our case, we only work with the Pool4 and Pool5 layers, that are the two last pooling layers from VGG-16. Once we have a certain region of interest from the hierarchy, we decide which feature map to use by comparing the scale of the ROI and the scale of the feature map. For large objects, the algorithm will select the deeper feature map, whereas for smaller objects a shallower feature map is more adequate.

The two models for feature extraction result into a feature map of 7x7 which is fed to the common block of the architecture. The region descriptor and the memory vector are the input of the Deep Q-network that consists of two fully connected layers of 1024 neurons each. Each fully connected layer is followed by a ReLU [12] activation function and is trained with dropout [18]. Finally the output layer corresponds to the possible actions of the agent, six in our case.

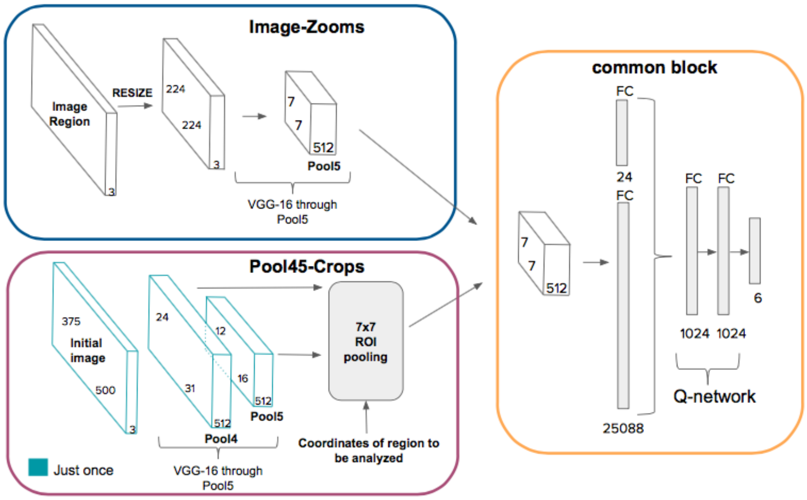

Figure 3: Hierarchical Object Detection Models

### 3.4 Training

In this section we will explain the particularities that we chose to train the Q-network.

**Exploration-Exploitation** To train the deep Q-network with reinforcement learning we use an $\epsilon$-greedy policy, that starts with $\epsilon$=1 and decreases until $\epsilon$=0.1 in steps of 0.1. Then, we start with random actions, and at each epoch the agent takes decisions relying more on the already learnt policy. Actually, in order to help the agent to learn the terminal action, which in random could be difficult to learn, we force it each time the current region has a IoU > 0.5. With this approach we can accelerate the training. Notice that we always do exploration, so we do not get stuck into a local minimum.

**Learning trajectories** One fact that we detected while training was that we should not impose which object of the image to look first. At each time step, the agent will focus on the object in the current region with the highest overlap with its ground-truth. This way, it is possible then that the target object changes during the top-down exploration.

**Training parameters** The weights for the Deep Q-network were initialized from a normal distribution. For learning, we used Adam optimizer [7] with a learning rate of 1e-6 to avoid that the gradients explode. We trained each model for 50 epochs.

**Experience Replay** As we have seen previously, Bellman Equation 3 learns from transitions formed by *(s, a, r, s')*, which can also be called experiences. Consecutive experiences in our algorithm are very correlated and this could lead to inefficient and unstable learning, a traditional problem in Q-learning. One solution to make the algorithm converge is collecting experiences and storing them in a replay memory. Random minibatches from this replay memory are used to train the network. We used an experience replay of 1,000 experiences and a batch size of 100.

**Discount factor** To perform well in the long-run, the future rewards should also be taken into account and not only the most immediate ones. To do this, we use the discounted reward from Bellman Equation 3 with a value of $\gamma$ = 0.90. We set the gamma high because we are interested in balancing the immediate and future rewards.

# 4 Experiments

Our experiments on object detection have used images and annotations from the PASCAL VOC dataset [5]. We trained our system on the trainval sets of 2007 and 2012, and tested it on the test set of 2007. We performed all the experiments for just one class, the *aeroplane* category, and only considering pictures with the target class category. This experiment allows us to study the behavior of our agent and estimate the amount of regions that must be analyzed to detect an object.

## 4.1 Qualitative results

We present some qualitative results in Figure 4 to show how our agent behaves on test images. These results are obtained with the *Image-Zooms* model with overlapped regions, as this is the one that yields best results, as argued in the following sections. We observed that for most images, the model successfully zooms towards the object and completes the task in a few steps. As seen in the second, third and fourth rows, with just two or three steps, the agent selects the bounding box around the object. The agent also performs accurately when there are small instances of objects, as seen in the first and last rows.

## 4.2 Precision-Recall curves

We will analyze the precision and recall curves for different trained models, considering that an object is correctly detected when the Intersection over Union (IoU) of its bounding box compared to the ground truth is over 0.5, as defined by the Pascal VOC challenge [5].

The Precision-Recall curves are generated by ranking all regions analyzed by the agent. The sorting is based on the reward estimated by the sixth neuron of the Q-network, which corresponds to the action of considering the region as terminal.

**Upper bound and random baselines** Our results firstly include baseline and upper-bound references for a better analysis. As a baseline we have programmed an agent that chooses random actions and detection scores at each time step. As an upper-bound, we have exploited the ground truth annotations to manually guide our agent towards the region with the greatest IoU. The result of these random baselines and upper-bounds for hierarchy type can be seen in Figure 5. It is also important to notice that the best upper-bound option does not even achieve a a recall of 0.5. This poor performance is because more than half of the ground truth objects do not fit with the considered region proposals, so they cannot be detected in our framework.

**Overlapped and non-overlapped regions** The results obtained with the upper-bound and baseline methods provide enough information to compare the overlapped and non-overlapped schemes. The overlapped regions scheme is the one that provides higher precision and recall values, both for the upper-bound and the random models. This superiority of the overlapped case can be explained by the slower reduction of spatial scale with respect to the non-overlapped model: as bounding box regions

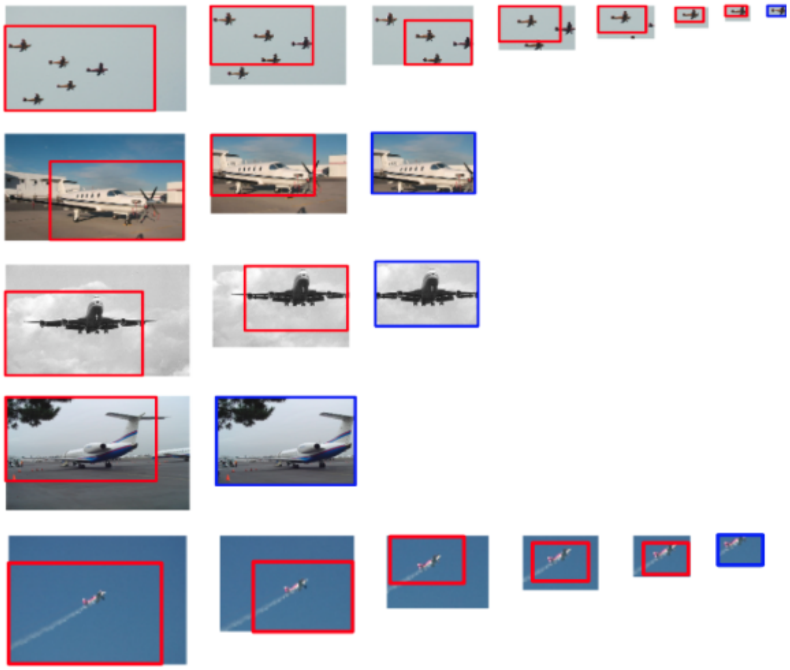

Figure 4: Visualizations of searches for objects

are larger due to the overlap, their division in equal-size subregions also generate larger subregions. This also implies that the agent will require more steps to reach a lower resolution, but this finer top-down exploration is shown as beneficial in our experiments as the chances of missing an object during the descent are also lower.

**Model comparison** The *Image-Zooms* model and the *Pool45-Crops* model are compared in Figure 6. Results clearly indicate that the *Image-Zooms* model performs better than the *Pool45-Crops* model. We hypothesize that this loss of performance is due to the loss of resolution resulting from the ROI-pooling over Pool4 or Pool5 layers. While in the *Image-Zooms* model the 7x7 feature maps of Pool5 have been computed directly from a zoom over the image, in the *Pool45-Crops* model the region crops over Pool4 or Pool5 could be smaller than 7x7. While these cases would be upsampled to the 7x7x512 input tensor to the deep Q-Net, the source feature maps of the region would be of lower resolution than their counterparts in the *Image-Zoom* model.

**Models at different epochs** We study the training of our *Image-Zooms* model by plotting the Precision-Recall curves at different epochs in Figure 7. As expected, we observe how the performance of the model improves with the epochs.

### 4.3 Number of regions analyzed per object

An histogram of the amount of regions analyzed by our agent is shown in 8. We observe that the major part of objects are already found with a single step, which means that the object occupies the major part of the image. With less than 3 steps we can almost approximate all objects we can detect.

## 5 Conclusions

This paper has presented a deep reinforcement learning solution for object detection. Our solution is characterized by a top-down exploration of a hierarchy of regions guided by an intelligent agent.

Our experiments indicate that objects can be detected with very few proposals from an appropriate hierarchy, but that working with a predefined set of regions clearly limits the recall. A possible

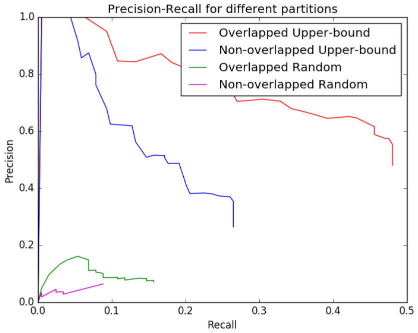

Figure 5: Baselines and upper-bounds

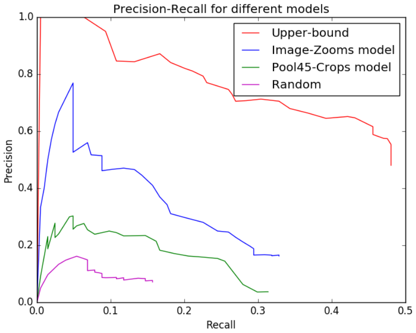

Figure 6: Comparison of the two models

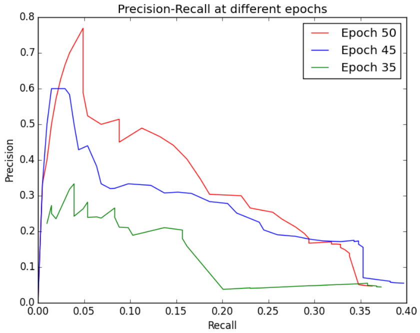

Figure 7: Image-Zooms at different epochs

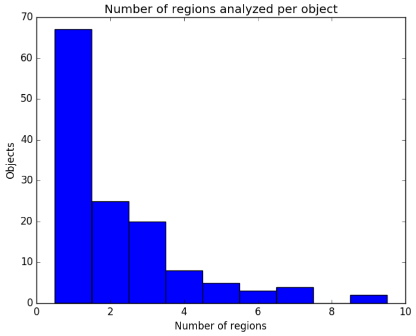

Figure 8: Regions analyzed per object

solution to this problem would be refining the approximate detections provided by the agent with a regressor, as in [15].

Finally, our results indicate the limitations of cropping region features from the convolutional layers, especially when considering small objects. We suggest that, given the much smaller amount of region proposals considered by our reinforcement learning agent, feeding each region through the network is a solution that should be also considered.

The presented work is publicly available for reproducibility and extension at `https://imatge-upc.github.io/detection-2016-nipsws/`.

## Acknowledgments

This work has been developed in the framework of the project BigGraph TEC2013-43935-R, funded by the Spanish Ministerio de Economia y Competitividad and the European Regional Development Fund (ERDF). This work has been supported by the grant SEV2015-0493 of the Severo Ochoa Program awarded by Spanish Government, project TIN2015-65316 by the Spanish Ministry of Science and Innovation contracts 2014-SGR-1051 by Generalitat de Catalunya. The Image Processing Group at the UPC is a SGR14 Consolidated Research Group recognized and sponsored by the Catalan Government (Generalitat de Catalunya) through its AGAUR office. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the GeForce GTX Titan Z used in this work and the support of BSC/UPC NVIDIA GPU Center of Excellence.

## References

[1] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. Multiple object recognition with visual attention. *arXiv preprint arXiv:1412.7755*, 2014.

[2] Juan C Caicedo and Svetlana Lazebnik. Active object localization with deep reinforcement learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2488–2496, 2015.

[3] Joao Carreira and Cristian Sminchisescu. Cpmc: Automatic object segmentation using constrained parametric min-cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(7):1312–1328, 2012.

[4] Dumitru Erhan, Christian Szegedy, Alexander Toshev, and Dragomir Anguelov. Scalable object detection using deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2147–2154, 2014.

[5] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[6] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015.

[7] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[8] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, and Scott Reed. Ssd: Single shot multibox detector. *arXiv preprint arXiv:1512.02325*, 2015.

[9] Yongxi Lu, Tara Javidi, and Svetlana Lazebnik. Adaptive object detection using adjacency and zoom prediction. *arXiv preprint arXiv:1512.07711*, 2015.

[10] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems*, pages 2204–2212, 2014.

[11] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[12] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

[13] Jordi Pont-Tuset, Pablo Arbeláez, Jonathan T. Barron, Ferran Marques, and Jitendra Malik. Multiscale combinatorial grouping for image segmentation and object proposal generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2016.

[14] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *arXiv preprint arXiv:1506.02640*, 2015.

[15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[16] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

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

[18] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1): 1929–1958, 2014.

[19] Koen EA Van de Sande, Jasper RR Uijlings, Theo Gevers, and Arnold WM Smeulders. Segmentation as selective search for object recognition. In *2011 International Conference on Computer Vision*, pages 1879–1886. IEEE, 2011.

[20] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*, 2(3):5, 2015.

[21] Serena Yeung, Olga Russakovsky, Greg Mori, and Li Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. *arXiv preprint arXiv:1511.06984*, 2015.

[22] Donggeun Yoo, Sunggyun Park, Joon-Young Lee, Anthony S Paek, and In So Kweon. Attentionnet: Aggregating weak directions for accurate object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2659–2667, 2015.

[23] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *European Conference on Computer Vision*, pages 391–405. Springer, 2014.