Deep Reinforcement Learning of Region Proposal Networks for Object Detection

Aleksis Pirinen[1] and Cristian Sminchisescu[1,2]
[1]Department of Mathematics, Faculty of Engineering, Lund University
[2]Institute of Mathematics of the Romanian Academy
{aleksis.pirinen, cristian.sminchisescu}@math.lth.se

Abstract

We propose drl-RPN*, a deep reinforcement learning-based visual recognition model consisting of a sequential region proposal network (RPN) and an object detector. In contrast to typical RPNs, where candidate object regions (RoIs) are selected greedily via class-agnostic NMS, drl-RPN optimizes an objective closer to the final detection task. This is achieved by replacing the greedy RoI selection process with a sequential attention mechanism which is trained via deep reinforcement learning (RL). Our model is capable of accumulating class-specific evidence over time, potentially affecting subsequent proposals and classification scores, and we show that such context integration significantly boosts detection accuracy. Moreover, drl-RPN automatically decides when to stop the search process and has the benefit of being able to jointly learn the parameters of the policy and the detector, both represented as deep networks. Our model can further learn to search over a wide range of exploration-accuracy trade-offs making it possible to specify or adapt the exploration extent at test time. The resulting search trajectories are image- and category-dependent, yet rely only on a single policy over all object categories. Results on the MS COCO and PASCAL VOC challenges show that our approach outperforms established, typical state-of-the-art object detection pipelines.*

1. Introduction

Visual object detection focuses on localizing each instance within a pre-defined set of object categories in an image, most commonly by estimating bounding boxes with associated confidence values. Accuracy on this task has increased dramatically over the last years [11, 14, 42], reaping the benefits of increasingly deep and expressive feature extractors [21, 28, 44, 47]. Several contemporary state-of-the-art detectors [11, 15, 42] follow a two-step process. First bottom-up region proposals are obtained, either from an internal region proposal network (RPN) [42], trained alongside the detection network, or from an external one [2, 6, 39, 48, 52]. In the second step proposals are classified and their localization accuracy may be fine-tuned.

There has recently been an increased interest in active, sequential search methods [5, 9, 17, 19, 22, 23, 27, 30, 33, 35, 37, 50]. This class of approaches, to which our model belongs, seek to only inspect parts of each image sequentially. In this work we aim to make active recognition models more flexible as characterized by *i)* a finely-tuned active search process where decisions of where to look next and when to stop searching are image- and category-dependent; *ii)* context information is aggregated as search proceeds and is used in decision making and to boost detection accuracy; *iii)* the detector and search policy parameters are tightly linked into a single deep RL-based optimization problem where they are estimated jointly; *iv)* the search process can be adapted to a variety of exploration-accuracy trade-offs during inference; and *v)* learning to search is only weakly supervised, as we indicate the model what success means without telling it exactly how to achieve it – there is no apprenticeship learning or trajectory demonstration.

Methodologically we propose *drl-RPN*, a sequential region proposal network combining an RL-based top-down search strategy, implemented as a convolutional gated recurrent unit, and a two-stage bottom-up object detector. Notably, our model is used for class-agnostic proposal generation but leverages class-specific information from earlier time-steps when proposing subsequent regions (RoIs). This context aggregation is also used to increase detection accuracy. Our model offers the flexibility of jointly training the policy and detector, both represented as deep networks, which we perform in alternation in the framework of deep RL. We emphasize that drl-RPN can be used, in principle, in conjunction with any exhaustive two-stage state-of-the-art object detector operating on a set of object proposals, such as Faster R-CNN (Fr R-CNN) [42] or R-FCN [11].

2. Related Work

Among the first to use deep feature extractors for object detection was [43], whereas [14] combined the power of

smaller and more plausible region proposal sets with such deep architectures. This was followed up in [11, 15, 20, 42] with impressive results. There is also a recent trend towards solutions where bounding box and classification predictions are produced in one shot [32,40,41]. Such methods increase detection speed sometimes at the cost of a lower accuracy.

The general detection pipeline above is characterized by its exhaustive, non-sequential nature: even if the set of windows to classify is reduced a-priori, all windows are still classified simultaneously and independently of each other. In contrast, sequential methods for object detection can be in principle designed to accumulate evidence over time to potentially improve accuracy at the given task. Such approaches can coarsely be divided as RL-based [5,9,19,22,23,27,30,35,37] and non-RL-based [17,33,50]. Our drl-RPN model is of the former category.

Orthogonally from us, [23] propose anytime models where a detector can be stopped asynchronously during inference: multi-class models are scheduled sequentially and the order of exhaustively applying sliding window detectors is optimized, potentially without running detectors for some classes. Our drl-RPN is also a multi-class detector but instead avoids searching all image locations. In [5] a class-specific agent observes the entire image, then performs a sequence of bounding box transformations until tightly enclosing the object. Results were improved in [27] where a joint multi-agent Q-learning system [38] is used and sub-agents cooperate to find several objects. In contrast, [35] use policy gradients to train a 'saccade-and-fixate' search policy over pre-computed RoIs that automatically determines when to stop searching. The formulation in [35] is however one-versus-all, not entirely deep, and is primarily designed for single instance detection. On the contrary, the deep model we propose detects multiple instances and categories, circumventing the need to train multiple search policies as in [5,35]. Fast R-CNN [15] is coupled with a tree-structured search strategy in [22] and results exceed or match the basic Fast R-CNN. Differently from us however, [22] manually specify the number of proposals during inference (hence stopping is not automatic but preset) and the detector is not refined jointly with the search policy.

Notable non-RL-based active search approaches include [17, 33, 50]. A soft attention mechanism is learned in [50] where directions for the next step are predicted, akin to a gradual shift in attention; [17] apply a search strategy for partial image exploration guided by statistical relations between regions; and [33] use adjacency and zoom prediction to focus processing on sub-regions likely to contain objects.

3. Two-Step Proposal-based Detection

We now briefly review those standard two-step proposal-based object detection components which will form some of the building blocks of our sequential drl-RPN model. Such detectors take as input an image of size $h_0 \times w_0 \times 3$ and process it through a base network. We use the same VGG-16 base network [44] as in [42]. The base network outputs the *base feature map* with dimension $h \times w \times d$, where h and w depend on h_0 and w_0 and $d = 512$ for VGG-16. The network then separates into two branches: RoI *generation* followed by RoI *pooling* and *classification*.

A region proposal network (RPN) is used for generating RoIs, where a d-dimensional feature vector is produced at each spatial location on the base feature map and sent through two class-agnostic layers: box-regression (*reg*) and box-classification (*cls*). To increase object recall several proposals are predicted relative to k anchor boxes (we use the same $k = 9$ anchors as [42]). The last task of the RPN is to reduce the number of RoIs forwarded to RoI pooling and classification. This is performed by class-agnostic NMS based on the objectness scores in *cls*. All RoIs forwarded by the RPN are converted to small spatially fixed-sized feature maps by means of RoI max pooling and are subsequently sent to two fully-connected layers performing class probability and bounding box offset predictions.

4. Sequential Region Proposal Network

We now present the architecture of drl-RPN, consisting of the object detector and the policy $\pi_{\boldsymbol{\theta}}$, see Fig. 1. For the detector we use a publicly available TensorFlow [1] implementation[1] of Fr R-CNN on top of which we implement our drl-RPN model. In principle however, drl-RPN can be integrated with any RPN-based detector, such as [11]. The search policy is based on a convolutional gated recurrent unit (Conv-GRU) which replaces fully-connected components of the GRU [10] with convolutions.

The input to the Conv-GRU at time t is the RL base state volume $\boldsymbol{S}_t$ (see §4.1) and the previous hidden state $\boldsymbol{H}_{t-1}$. The output is a two-channel action volume $\boldsymbol{A}_t$. The spatial extent of all inputs and outputs are $h \times w$. We denote by $*$ the convolution operator and $\odot$ denotes element-wise multiplication. Weights and biases are denoted $\boldsymbol{W}$ and $\boldsymbol{b}$ respectively, and $\sigma[\cdot]$ is the logistic sigmoid function. Below are the equations of our Conv-GRU agent:

$$\boldsymbol{O}_t = \sigma[\boldsymbol{W}_{so} * \boldsymbol{S}_t + \boldsymbol{W}_{ho} * \boldsymbol{H}_{t-1} + \boldsymbol{b}_o] \quad (1)$$

$$\tilde{\boldsymbol{H}}_t = \boldsymbol{W}_{sh} * \boldsymbol{S}_t + \boldsymbol{W}_{hh} * (\boldsymbol{O}_t \odot \boldsymbol{H}_{t-1}) + \boldsymbol{b}_h \quad (2)$$

$$\boldsymbol{Z}_t = \sigma[\boldsymbol{W}_{sz} * \boldsymbol{S}_t + \boldsymbol{W}_{hz} * \boldsymbol{H}_{t-1} + \boldsymbol{b}_z] \quad (3)$$

$$\boldsymbol{H}_t = (1 - \boldsymbol{Z}_t) \odot \boldsymbol{H}_{t-1} + \boldsymbol{Z}_t \odot \tanh[\tilde{\boldsymbol{H}}_t] \quad (4)$$

$$\tilde{\boldsymbol{A}}_t = \mathrm{relu}[\boldsymbol{W}_{h\tilde{a}} * \boldsymbol{H}_t + \boldsymbol{b}_{\tilde{a}}] \quad (5)$$

$$\boldsymbol{A}_t = \tanh[\boldsymbol{W}_{\tilde{a}a} * \tilde{\boldsymbol{A}}_t + \boldsymbol{b}_a] \quad (6)$$

The output $\boldsymbol{A}_t$ of the Conv-GRU corresponds to two possible actions, see §4.1. Let $\boldsymbol{\theta}$ denote *all* parameters of the

[1] https://github.com/smallcorgi/Faster-RCNN_TF

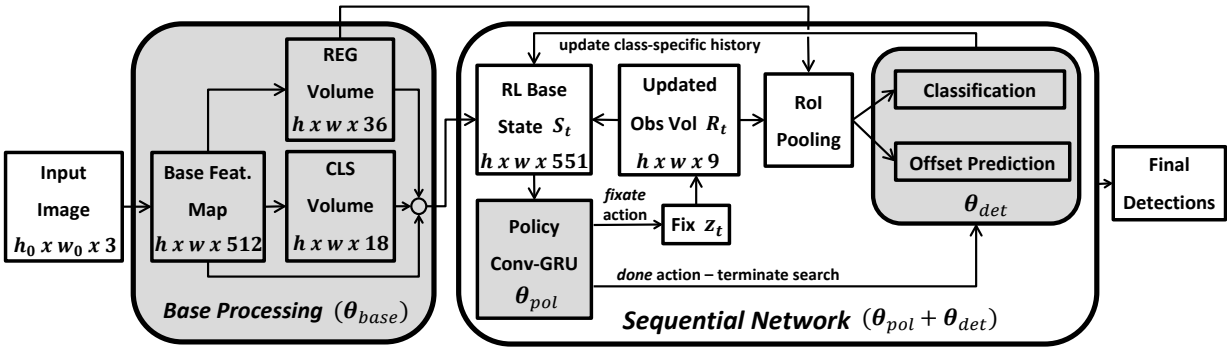

Figure 1: Overview of our proposed drl-RPN model. The base processing is illustrated to the left, showing how the initial state $\boldsymbol{S}_0$ is formed. Each time-step t, the agent decides whether to terminate search based on its stochastic policy $\pi_{\boldsymbol{\theta}}(a_t|s_t)$, c.f. (1) - (8). As long as search has not terminated, a *fixate* action a_t^f is issued and a new location $\boldsymbol{z}_t$ is visited; the RoI observation volume $\boldsymbol{R}_t$ is updated in an area centered at $\boldsymbol{z}_t$. All corresponding RoIs are sent to the RoI pooling module followed by classification and class-specific bounding box offset predictions. The class-specific probability vectors are inserted to the history volume $\boldsymbol{V}_t^4$ which is merged with the RL base state volume $\boldsymbol{S}_t$. Based on the new state, a new action is taken at time-step $t+1$ and the process is repeated until the *done* action a_t^d is issued, then all the selected predictions throughout the trajectory are collected. The trainable parts of the network (including the feature extraction, the classification and regression model, and the policy) are highlighted in gray. See also Fig. 5 for some visualizations of drl-RPN search strategies.

system where drl-RPN is used, which can be decomposed as $\boldsymbol{\theta}_{base}, \boldsymbol{\theta}_{det}$ and $\boldsymbol{\theta}_{pol}$. Here $\boldsymbol{\theta}_{base}$ are the parameters of the base network and the original RPN; $\boldsymbol{\theta}_{det}$ are the parameters of the classifier and bounding box offset predictor; and $\boldsymbol{\theta}_{pol}$ are the search policy parameters, c.f. (1) - (8). The joint training of $\boldsymbol{\theta} = [\boldsymbol{\theta}_{base}; \boldsymbol{\theta}_{det}; \boldsymbol{\theta}_{pol}]$ is described in §5.

4.1. States and Actions

The state at time t is the tuple $s_t = (\boldsymbol{R}_t, \boldsymbol{S}_t, \boldsymbol{H}_t)$, where $\boldsymbol{R}_t \in \{0,1\}^{h\times w\times k}$ is the *RoI observation volume*, $\boldsymbol{S}_t \in \mathbb{R}^{h\times w\times(d+2k+N+1)}$ is the *base state* and $\boldsymbol{H}_t \in \mathbb{R}^{h\times w\times 300}$ is the hidden state of the Conv-GRU. Here N is the number of object categories considered. There are two types of actions, corresponding to one channel each of $\boldsymbol{A}_t$ in (6): a *fixate* action a_t^f and the *done* action a_t^d. The done action is binary, where $a_t^d = 1$ corresponds to terminating search. A *fixate* action $a_t^f = \boldsymbol{z}_t$ is issued if $a_t^d = 0$, where $\boldsymbol{z}_t$ is the $(h \times w)$-plane coordinate of the next fixation. We next define $\boldsymbol{R}_t$ and explain how it relates to *fixate* actions a_t^f, after which we present $\boldsymbol{S}_t$ and explain its connection to $\boldsymbol{R}_t$. Finally, we describe how actions are sampled using our parametric stochastic policy $\pi_{\boldsymbol{\theta}}$.

RoI observation volume $\boldsymbol{R}_t$: drl-RPN maintains a binary volume $\boldsymbol{R}_t$ of size $h \times w \times k$ in which the (i,j,l):th entry is 1 if and only if the corresponding RoI is part of the region proposal set forwarded to the RoI pooling / classification part of the network. We initialize $\boldsymbol{R}_t$ as an all-zeros volume. After a *fixate* action a_t^f, a part of $\boldsymbol{R}_t$ in a neighbourhood of the fixation location $\boldsymbol{z}_t$ is updated. This neighbourhood is a rectangular area centered at $\boldsymbol{z}_t$, in which the side lengths h_{rect} and w_{rect} are a fraction each of h and w, respectively (we set $h_{\text{rect}} = h/4$ and $w_{\text{rect}} = w/4$). We set all entries of $\boldsymbol{R}_t$ inside this rectangle to 1 to indicate that these RoIs have been selected. Note that we here restrict our algorithm to determine at what *spatial* locations to sample RoIs in the $(h \times w)$-plane, so all k anchor candidates are used per spatial location.

Base state volume $\boldsymbol{S}_t$: The state $\boldsymbol{S}_t$ consists of $\boldsymbol{V}_t^1 \in \mathbb{R}^{h\times w\times d}$, $\boldsymbol{V}_t^2, \boldsymbol{V}_t^3 \in \mathbb{R}^{h\times w\times k}$ and $\boldsymbol{V}_t^4 \in \mathbb{R}^{h\times w\times(N+1)}$. We set $\boldsymbol{V}_0^1$ to the base feature map (`conv5_3`) and $\boldsymbol{V}_0^2$ to the objectness layers (in *cls*) of the RPN. The *reg* volume of the RPN is used for $\boldsymbol{V}_t^3$, where $\boldsymbol{V}_0^3$ is set to the magnitude of the $[0,1]$-normalized offsets $[\Delta x_1, \Delta y_1, \Delta x_2, \Delta y_2]$. We use $\boldsymbol{R}_t$ to update these volumes, setting the corresponding locations in $\boldsymbol{V}_t^1, \boldsymbol{V}_t^2$ and $\boldsymbol{V}_t^3$ to -1, meaning those locations have been inspected. The volume $\boldsymbol{V}_t^4$ is a class-specific history of what has been observed so far. We set $\boldsymbol{V}_0^4 = \boldsymbol{0}$. After a fixation the selected RoIs are sent to class-specific predictors. Then *local* class-specific NMS is applied to the classified RoIs to get the most salient information at that location. As we have final bounding box predictions for the surviving RoIs, we map them to certain spatial locations of $\boldsymbol{V}_t^4$. The input image is divided into $L \times L$ bins of size $\approx h_0/L \times w_0/L$ to get a coarse representation of where the agent has looked (we set $L = 3$) by assigning each NMS-survivor to the bin containing its center coordinates. The history $\boldsymbol{V}_t^4$ at these locations is updated with those class probability vectors as a running average.

We use 3×3 convolutional kernels for the base input $\boldsymbol{V}_t^1$ since the effective receptive field is already wide given that we are operating on deep feature maps. For the auxiliary input $\boldsymbol{V}_t^2$ - $\boldsymbol{V}_t^4$ we apply larger 9×9 kernels.

Stochastic policy $\pi_{\boldsymbol{\theta}}(a_t|s_t)$: We now describe how $\boldsymbol{A}_t$ in (6) is used to select actions. Let $\boldsymbol{A}_t^d$ and $\boldsymbol{A}_t^f$ denote the first (*done*) and second (*fixate*) layers of $\boldsymbol{A}_t$, respectively. The done layer $\boldsymbol{A}_t^d$ is bilinearly re-sized to 25×25 and stacked as a vector $\boldsymbol{d}_t \in \mathbb{R}^{625}$. The probability of terminating in state s_t is then given by:

$$\pi_{\boldsymbol{\theta}}(a_t^d = 1|s_t) = \sigma\left[\boldsymbol{w}_d^\top \boldsymbol{d}_t + t\right] \tag{7}$$

where $\boldsymbol{w}_d$ is a trainable weight vector. The fixation layer $\boldsymbol{A}_t^f$ is transformed to a probability map $\hat{\boldsymbol{A}}_t^f$ by applying a spatial softmax to $\boldsymbol{A}_t^f$. The probability of fixating location $\boldsymbol{z}_t = (i, j)$ *given* that the agent did not stop is $\hat{\boldsymbol{A}}_t^f[\boldsymbol{z}_t]$, where $\hat{\boldsymbol{A}}_t^f[\boldsymbol{z}_t]$ is the (i, j):th entry of $\hat{\boldsymbol{A}}_t^f$. The probability of fixating location $\boldsymbol{z}_t$ in state s_t is thus given by:

$$\pi_{\boldsymbol{\theta}}\left(a_t^d = 0, a_t^f = \boldsymbol{z}_t|s_t\right) = \left(1 - \sigma\left[\boldsymbol{w}_d^\top \boldsymbol{d}_t + t\right]\right) \hat{\boldsymbol{A}}_t^f[\boldsymbol{z}_t] \tag{8}$$

4.2. Contextual Class Probability Adjustment

Typical detection pipelines classify all regions simultaneously and independently of each another, which is simple and efficient but limits information exchange between regions. We argue for an alternative where the search process and the classification of candidate proposals are unified into a single system, creating synergies between both tasks. We already explained how classified regions are used to guide the search process and now augment drl-RPN to use context accumulation also to perform a posterior update of its detection probabilities based on the search trajectory.[2]

The augmented model uses a summary of all object instances discovered during search to refine the final class probability scores for these detections. For this we use the history aggregation described in §4.1. Given the up to L^2 history vectors, we stack them as an $L^2(N+1)$-dimensional vector $\boldsymbol{x}_{hist}$ and represent non-observed regions by zeros in $\boldsymbol{x}_{hist}$. The final classification layer softmax$(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})$ is replaced with softmax$[\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b} + \boldsymbol{f}_{hist}(\boldsymbol{x}_{hist})]$ to account for the search trajectory. We use a one-layer activation $\boldsymbol{f}_{hist}(\boldsymbol{x}_{hist}) = \tanh(\boldsymbol{W}_{hist}\boldsymbol{x}_{hist} + \boldsymbol{b}_{hist})$.

5. Training

Training the full model (detector and policy) proceeds in alternation. Recall that we distinguish between three sets of parameters: $\boldsymbol{\theta} = [\boldsymbol{\theta}_{base}; \boldsymbol{\theta}_{det}; \boldsymbol{\theta}_{pol}]$, where $[\boldsymbol{\theta}_{base}; \boldsymbol{\theta}_{det}]$ are the parameters of the original Fr R-CNN. We use a pre-trained network[3] as initialization of $[\boldsymbol{\theta}_{base}; \boldsymbol{\theta}_{det}]$. Xavier initialization [16] is used for the search policy parameters $\boldsymbol{\theta}_{pol}$. We next explain how to learn $\boldsymbol{\theta}_{pol}$ via deep RL; the joint training of the full system is described in §5.3.

5.1. Reward Signal

There are two criteria which the agent should balance. First, the chosen RoIs should yield high object instance overlap and second, the number of RoIs should be as low as possible to reduce the number of false positives and to maintain a manageable processing time.

Fixate action reward: To balance the above trade-off we give a small negative reward $-\beta$ for each *fixate* action (we set $\beta = 0.075$), but the agent also receives a positive reward for fixations yielding increased intersection-over-union (IoU) with any ground-truth instances $\boldsymbol{g}_i$ for the current image. For each object instance $\boldsymbol{g}_i$ we keep track of the so-far maximum IoU-yielding[4] RoIs selected by the agent at previous time-steps $0, \ldots, t-1$. Let this be denoted IoU^i and note that $IoU^i = 0$ at $t = 0$. When $t \geq 1$ we compute the maximum IoU for all ground-truth instances $\boldsymbol{g}_i$ given by RoIs from that particular time-step, denoted IoU_t^i, and check if $IoU_t^i > IoU^i \geq \tau$, where we set $\tau = 0.5$ in accordance with the positive threshold for PASCAL VOC. For each ground-truth $\boldsymbol{g}_i$ satisfying this condition we give the positive reward $(IoU_t^i - IoU^i)/IoU_{max}^i$ after which we set $IoU^i = IoU_t^i$. Here IoU_{max}^i is the maximum IoU that $\boldsymbol{g}_i$ has with *any* of all hwk possible regions. Hence the fixation reward r_t^f at time t is given by

$$r_t^f = -\beta + \sum_i \mathbb{1}\left[\boldsymbol{g}_i : IoU_t^i > IoU^i \geq \tau\right] \frac{IoU_t^i - IoU^i}{IoU_{max}^i} \tag{9}$$

Done action reward: Upon termination the agent receives a final reward reflecting the quality of the search trajectory:

$$r_t^d = \sum_i \mathbb{1}\left[\boldsymbol{g}_i : IoU_{max}^i \geq \tau\right] \frac{IoU^i - IoU_{max}^i}{IoU_{max}^i} \tag{10}$$

Here IoU^i is the maximum IoU-yielding RoI (with instance $\boldsymbol{g}_i$) selected by the agent in the entire trajectory. Note that (10) evaluates to zero if all $\boldsymbol{g}_i$ are maximally covered and otherwise becomes increasingly negative depending on how severely the ground-truths were missed.

[2] We update detections *after* terminating search which gives also early detections an opportunity of being adjusted. In principle however, one could update detections as search proceeds based only on past detections.

[3] Using the same settings as in [42], including an additional anchor scale of 64^2 pixels when training on MS COCO.

[4] This refers to IoU *after* class-specific bounding box adjustments to ensure that the objective lies as close as possible to the final detection task.

5.1.1 Separation of Rewards

Although drl-RPN is a single-agent system taking one action per time-step via the policy in (7) - (8), it may be viewed as consisting of two subagents `agt_d` and `agt_f` with some shared and some individual parameters. The agent `agt_d`, governed by (7), decides whether to keep searching, whereas `agt_f` is governed by (8) and controls where to look *given* that `agt_d` has not terminated search. We argue that `agt_d` should not necessarily be rewarded based on the performance of `agt_f`. For example, early in training `agt_f` may choose poor fixation locations, thus missing the objects. In a standard reward assignment both `agt_f` and `agt_d` receive negative reward based on the behaviour of `agt_f`. However, only `agt_f` should be penalized in this situation as it alone is responsible for not localizing objects despite the opportunity given by `agt_d`.

Instead of giving the actual fixation reward r_t^f in (9) to `agt_d` we define an optimistic corresponding reward as

$$\tilde{r}_t^f = -\beta + \max_{IoU \geq \tau} \frac{IoU - IoU^i}{IoU^i_{max}} \tag{11}$$

The reward (11) reflects the maximum increase of IoU of *one* single ground-truth instance $\boldsymbol{g}_i$ attainable by *any* fixate action. Note that (11) may not always be optimistic; the true fixation reward (9) can be higher in images with several objects (by covering multiple instances in one fixation). Early in training however, (11) is often higher than (9). Therefore we give $\max(r_t^f, \tilde{r}_t^f)$ as fixation reward to `agt_d`.

This *separation of rewards* between `agt_d` and `agt_f` helped drl-RPN find a reasonable termination policy; it otherwise tended to stop the search process too early. Separation of rewards does not increase computational cost and is easy to implement, making it a simple adjustment for improving learning efficiency. It is applicable in any RL problem where actions have similar hierarchical dependencies.

5.1.2 Adaptive Exploration-Accuracy Trade-Off

So far we have described drl-RPN with a fixed exploration penalty β in training, c.f. (9). After training the exploration extent is hard-coded into the policy parameters. By treating β as an input we can instead obtain a *goal-agnostic* agent whose exploration extent may be specified at test time. Goal-agnostic agents have also been proposed in different contexts by contemporary work; see e.g. [12, 51].

An adjustment is performed between equations (5) - (6), where a constant β-valued feature map is appended to $\tilde{\boldsymbol{A}}_t$. In training we define a set of β-values the model is exposed to and for each trajectory we randomly sample a β from this set. In testing we simply specify β, which does not necessarily have to be from the set of β-values seen in training.

5.2. Objective Function

To learn the policy parameters we maximize the expected cumulative reward on the training set, given by $J(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{s}\sim\pi_{\boldsymbol{\theta}}}\left[\sum_{t=1}^{|\boldsymbol{s}|} r_t\right]$, where $\boldsymbol{s}$ represents a trajectory of states and actions, sampled by running the model from the initial state s_0 (c.f. §4.1). A sample-based approximation to the gradient [46] of the objective function $J(\boldsymbol{\theta})$ is obtained using REINFORCE [49]. We use 50 search trajectories to approximate the true gradient, forming one batch in our gradient update (one image per batch), and update the policy parameters via backpropagation using Adam [25]. To increase sample efficiency we use the return normalization in [19], where cumulative rewards for each episode are normalized to mean 0 and variance 1 over the batch. The maximum trajectory length is set to 12.

5.3. Joint Training of Policy and Detector

As we use one image per batch it is straightforward to also tune the detector parameters $[\boldsymbol{\theta}_{base}; \boldsymbol{\theta}_{det}]$. Once the policy parameters $\boldsymbol{\theta}_{pol}$ have been updated for an image (with $[\boldsymbol{\theta}_{base}; \boldsymbol{\theta}_{det}]$ frozen[5]) we fix $\boldsymbol{\theta}_{pol}$ and produce one more search trajectory for that image. The RoIs selected by drl-RPN during this trajectory are used as RoIs in Fr R-CNN instead of RoIs from the standard RPN, but otherwise the detector is updated as in [42]. Once the full drl-RPN model has been trained it is simple to also learn (refine) the parameters of the posterior class probability predictor in §4.2. Specifically, we jointly train $\boldsymbol{W}$, $\boldsymbol{W}_{hist}$, $\boldsymbol{b}$ and $\boldsymbol{b}_{hist}$ as for the original Fr R-CNN model, except that drl-RPN is used for generating RoIs. The remaining parameters are kept frozen at this stage, although it is possible to alternate.

6. Experiments

We now compare our proposed drl-RPN[6] to Fr R-CNN[7] on the MS COCO [31] and PASCAL VOC [13] detection challenges. We report results mainly for models trained with a fixed exploration penalty $\beta = 0.075$; results for the goal-agnostic model presented in §5.1.2 are found in §6.3.

For PASCAL VOC we repeat the alternating training in §5.3 for 70k iterations on VOC 07+12 train-val.[8] The learning rate for $\boldsymbol{\theta}_{pol}$ is initially `2e-5` (`4e-6` after 50k iterations) and $\boldsymbol{\theta}_{det}$ has corresponding learning rates `2.5e-4` and `2.5e-5`. We use the same settings for MS COCO (trained on COCO 2014 train-val) but alternate for 350k iterations and update the learning rate after 250k iterations.

We compare drl-RPN to Fr R-CNN using the standard

[5] We keep $\boldsymbol{\theta}_{base}$ frozen throughout as tuning the base network did not increase performance.

[6] Unless otherwise specified we refer by drl-RPN to the model using the posterior class probability adjustments introduced in §4.2.

[7] We report results obtained for the implementation we used, which are often higher than in [42]; this was achieved by training for more iterations.

[8] For VOC 2012 we include the 2007 test set in training, as typical.

model	settings	test-dev mAP@.5	test-dev mAP@.75	test-dev mAP@[.5, .95]	test-dev mAR@[.5, .95]	test-std mAP@.5	test-std mAP@.75	test-std mAP@[.5, .95]	test-std mAR@[.5, .95]	voc12-test mAP@.5	voc07-test mAP@.5
RPN	**default**	42.7	21.4	22.3	32.3	42.7	21.1	22.3	32.3	73.0	75.6
drl-RPN	**ads**	43.3 40.9%, 8.1	23.0 40.9%, 8.1	23.4 40.9%, 8.1	32.9 40.9%, 8.1	43.3 40.7%, 8.0	23.0 40.7%, 8.0	23.4 40.7%, 8.0	32.9 40.7%, 8.0	74.1 37.7%, 7.1	**76.4** 39.9%, 7.6
	12-fix	**43.6** 51.7%, 12	**23.1** 51.7%, 12	**23.5** 51.7%, 12	33.3 51.7%, 12	**43.6** 51.6%, 12	**23.1** 51.6%, 12	**23.5** 51.6%, 12	33.3 51.6%, 12	**74.2** 50.4%, 12.0	**76.4** 51.1%, 12.0
	ads, np	43.2 40.9%, 8.1	22.0 40.9%, 8.1	22.8 40.9%, 8.1	33.1 40.9%, 8.1	43.0 40.7%, 8.0	21.9 40.7%, 8.0	22.7 40.7%, 8.0	33.2 40.7%, 8.0	73.7 37.7%, 7.1	76.1 39.9%, 7.6
	12-fix, np	43.4 51.7%, 12	22.2 51.7%, 12	23.0 51.7%, 12	**33.5** 51.7%, 12	43.3 51.6%, 12	22.0 51.6%, 12	22.8 51.6%, 12	**33.5** 51.6%, 12	74.0 50.4%, 12.0	76.0 51.1%, 12.0
	ads, nh	43.1 39.0%, 7.5	21.8 39.0%, 7.5	22.6 39.0%, 7.5	33.0 39.0%, 7.5	42.9 38.9%, 7.5	21.7 38.9%, 7.5	22.5 38.9%, 7.5	33.2 38.9%, 7.5	73.6 34.7%, 6.4	75.7 37.0%, 7.0

Table 1: Detection results on the MS COCO 2015 test sets, as well as the PASCAL VOC 2012 and 2007 test sets (two right-most columns). The first row of each drl-RPN modification shows the detection performance (mAP or mAR) and the second row shows average exploration (% of forwarded RoIs) and average number of fixations per image.

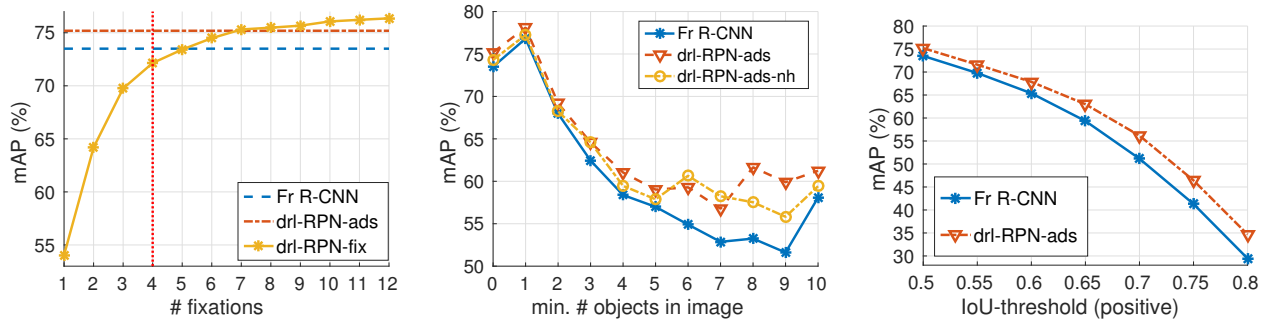

Figure 2: Ablation results on the PASCAL VOC 2007 test set. *Left:* Using a constant, preset number of fixations per image requires almost twice as many fixations per image to reach the same detection accuracy as the adaptively stopping model. *Mid:* The mAP of drl-RPN compared to Fr R-CNN is relatively higher in more crowded scenes and the class-specific history appears more useful in such scenes. *Right:* The relative performance of drl-RPN compared to Fr R-CNN generally increases with increased IoU-threshold (c.f. Fig. 4).

RPN and also investigate some variants of drl-RPN. Specifically, we compare to a model using the class-specific history only to guide the search process but not for posterior class probability adjustments (np); to a model completely void of a class-specific history (nh); and to a model enforcing 12 fixations per image (12-fix). Adaptive stopping models are denoted (ads).[9] For the various drl-RPN models we also show the average fraction of RoIs forwarded for class-specific predictions (called *exploration*, reported in %) and the average number of fixations per image.

6.1. Results on MS COCO

Results on MS COCO 2015 test-std and test-dev are shown in Table 1, together with PASCAL VOC 2007 and 2012 results for these models. On MS COCO the mAP of drl-RPN is 1.1 higher than for Fr R-CNN. Comparing with the ads-np and ads-nh models, the posterior class probability adjustments yield mAP boosts of 0.7 and 0.9, respectively. Enforcing 12 fixations marginally improves mAP by 0.1, while significantly increasing exploration by 25%. Also, drl-RPN increases mean average recall (mAR) by 0.6. As for PASCAL VOC, drl-RPN beats Fr R-CNN by 1.1 and 0.8 mAP on VOC 2012 and 2007, respectively. The class-specific history yields 0.5 and 0.7 mAP boosts respectively on VOC 2012 and 2007. Enforcing 12 fixations leads to negligible mAP improvements.

Overall, drl-RPN consistently outperforms the baseline Fr R-CNN model. We also see that the class-specific history with posterior adjustments yields significantly improved accuracy and that the adaptive stopping condition provides a drastic reduction in average exploration, yet matches the mAP of the corresponding 12-fixation policy.

6.2. Results on PASCAL VOC

Table 2 shows results on PASCAL VOC 2007 and 2012. To show the effect of joint policy-detector training we also present Fr R-CNN results using the drl-RPN tuned detector parameters (drl-RPN det). For VOC 2007 drl-RPN-ads achieves 1.7 mAP above Fr R-CNN. By enforcing 12 fixations drl-RPN more significantly outperforms the Fr R-CNN baseline by 2.9 mAP; c.f. Fig. 2 (left). Moreover, both the ads- and 12-fix drl-RPN models achieve signifi-

[9] Adaptive stopping drl-RPN models are used if not otherwise specified.

model	settings	mAP - 2007	mAP - 2012
RPN	default	73.5	70.4
	drl-RPN det	73.6	70.6
	all RoIs	74.2	70.7
drl-RPN	ads — 22.9%, 4.0	75.2	70.8
	12-fix — 40.3%, 12.0	**76.4**	**72.2**
	ads, np — 22.9%, 4.0	74.5	70.4
	12-fix, np — 41.7%, 12.0	75.5	71.8
	ads, nh — 22.1%, 3.9	74.3	70.1

Table 2: Detection results on the PASCAL VOC 2007 and 2012 test sets. We also show drl-RPN's average exploration and average number of fixations per image.

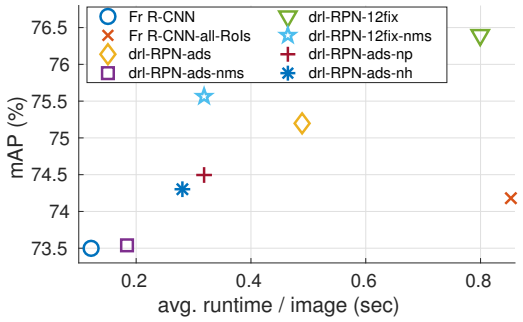

Figure 3: mAP vs. runtime for evaluated models on the PASCAL VOC 2007 test set. Fr R-CNN, while fast, offers very limited tuning of the speed-accuracy trade-off, whereas drl-RPN can be adapted to a wide range of requirements on accuracy or speed. See also Fig. 6.

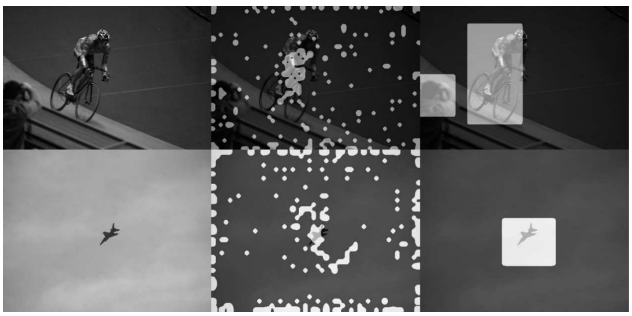

Figure 4: The drl-RPN attention (right) is more object-centric and less scattered over the image compared to the standard RPN (mid), resulting in fewer false positives.

cantly higher mAP compared to an exhaustive variant of Fr R-CNN which forwards all RoIs (without class-agnostic NMS), so increasing mAP is *not* merely a matter of detecting more RoIs. The Fr R-CNN results change negligibly when replacing the class-specific detector parameters to those of the tuned drl-RPN detector.[10] Hence, unsurprisingly, *it is crucial to perform detector tuning jointly with the policy learning*. Moreover, the class-specific history yields considerably better results (see also §6.3 and Fig. 2 (mid)). Similar results apply to VOC 2012. The adaptive stopping drl-RPN-ads beats Fr R-CNN by 0.4 mAP; it also surpasses the exhaustive "all RoIs" variant. At 12 fixations drl-RPN significantly outperforms Fr R-CNN by 1.8 mAP.

Comparing the VOC and COCO results, search trajectories for VOC are about 50% shorter on average. This is not surprising given that COCO scenes are significantly more crowded and complex; indeed, this further shows the benefit of an adaptive search with automatic stopping condition.

In Fig. 2 (left) we show results on VOC 2007 when enforcing exactly n fixations per image for $n = 1, \ldots, 12$. The mAP increases with the number of fixations and surpasses drl-RPN-ads for $n \geq 7$ and Fr R-CNN for $n \geq 5$. Drawn is also a vertical line corresponding to the mean number of fixations of drl-RPN-ads (4.0). Comparing to the model with a preset number of fixations clearly shows the benefit of the automatic stopping (3.0 mAP difference).

6.3. Ablation Studies

To further investigate our model we evaluate drl-RPN and Fr R-CNN in a few settings on the VOC 2007 test set. Some visualizations of drl-RPN search strategies and final detections are shown in Fig. 5.

Runtime and mAP comparisons: Fig. 3 shows mAP and runtime comparisons[11] between various models evaluated in this work. Our drl-RPN model outperforms Fr R-CNN in detection accuracy but not in speed. This is mainly because drl-RPN forwards a larger set of RoIs.[12] The sequential processing (based on the Conv-GRU described in §4) also adds an overhead of about 13 ms per fixation. Applying class-agnostic NMS to gate the drl-RPN proposals yields runtimes closer to that of Fr R-CNN while still improving mAP. Also, drl-RPN outperforms the exhaustive Fr R-CNN variant in both speed and accuracy.

mAP vs. number of objects per image: Comparing drl-RPN-ads to drl-RPN-ads-nh in Fig. 2 (mid) shows that class-specific context aggregation gets increasingly useful in crowded scenes which is quite expected (exceptions for 6, 7 objects). Also, drl-RPN-ads outperforms Fr R-CNN at all object counts and the improvement gets more pronounced in more crowded scenes.

mAP vs. IoU-threshold: Fig. 2 (right) shows that the relative performance of drl-RPN increases with box IoU-threshold τ, despite using the standard $\tau = 0.5$ during training. Comparing the COCO-style mAP scores (mAP@[.5, .95]), drl-RPN even more significantly out-

[10] We also tried drl-RPN without detector tuning, causing an mAP drop of 2.0, so joint policy-detector tuning is crucial.

[11] Runtimes reported using a Titan X GPU.

[12] The set of RoIs is however much more spatially compact, c.f. Fig. 4.

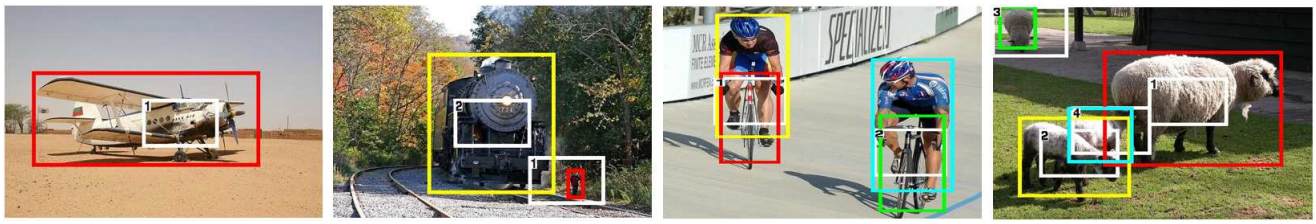

Figure 5: Upscaled fixation areas in white (c.f. $\boldsymbol{R}_t$ in §4.1) generated by drl-RPN and detection boxes (colored) for a few PASCAL VOC 2007 test images. We also show the time-step each area was observed. The sizes of the fixation areas are *not* related to the sizes of the selected RoIs; they simply determine where RoIs are being forwarded for class-specific predictions.

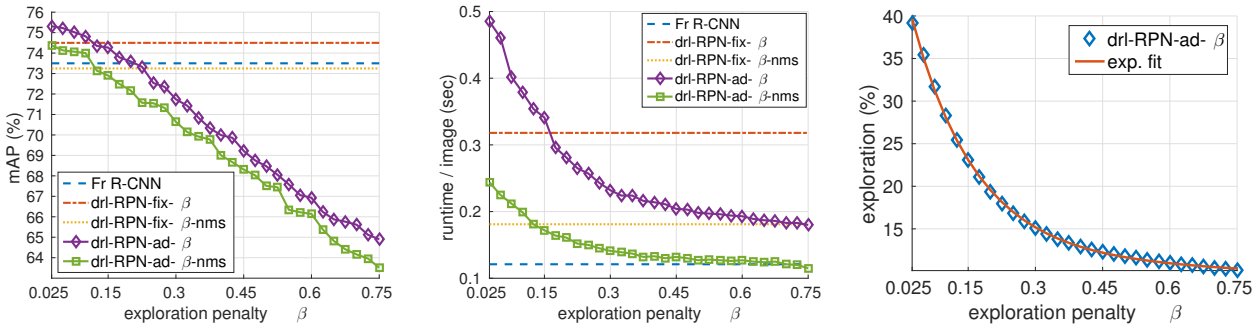

Figure 6: Investigation of exploration-accuracy trade-off on the PASCAL VOC 2007 test set. *Left:* For small β the goal-agnostic agents outperform the fixed-β counterparts as well as Fr R-CNN, while mAP expectedly decreases as β increases. *Mid:* Average runtime also decreases with increased β; at $\beta = 0.15$ (twice the β used for fixed-β models) the goal-agnostic models become faster than the fixed-β counterparts. *Right:* Exploration vs. β with a fitted two-term exponential $ae^{-b\beta} + ce^{-d\beta}$. The accurate functional fit allows for specifying exploration extent at test time.

performs Fr R-CNN with 44.3 against 41.3 mAP. See also the attention comparison in Fig. 4 showing where (spatially) RoIs are forwarded for class-specific predictions. For drl-RPN this corresponds to the upscaled fixation areas (c.f. $\boldsymbol{R}_t$ in §4.1). For the standard RPN we locate where the survivors of the class-agnostic NMS end up spatially and upsample those locations to match the image size.

Exploration-accuracy trade-off: Fig. 6 shows results[13] for the goal-agnostic extension of drl-RPN accepting the exploration penalty β as input (c.f. §5.1.2), evaluated for the set $\{0.025, 0.050, \ldots, 0.750\}$ of β-values used in training. We also compare to a model using class-agnostic NMS to gate the drl-RPN proposals. With this straightforward extension we obtain models which can be adjusted to a wide range of speed-accuracy trade-offs.

7. Conclusions

We have presented drl-RPN, a sequential deep reinforcement learning model of 'where to look next' for visual object detection, which automatically determines when to terminate the search process. The model produces image- and category-dependent search trajectories, yet it features a single policy over all object categories. All the (deep) parameters – including the fixation policy, stopping conditions, and object classifiers – can be trained jointly and experiments show that such joint refinement improves detection accuracy. Overall, drl-RPN achieves results superior to exhaustive, typical state-of-the-art methods and is particularly accurate in applications demanding higher IoU-thresholds for positive detections.

Results showing the advantages of a class-specific memory and context-aggregation within drl-RPN have also been presented. This offers a mechanism to incrementally accumulate evidence at earlier visited image regions and detections to guide the search process and boost detection accuracy. As expected, such a mechanism leads to even more dramatic improvements in more crowded scenes. Finally, we have shown that drl-RPN can learn a wide variety of exploration-accuracy trade-offs which makes it possible to specify the exploration extent at test time.

Acknowledgments: This work was supported by the European Research Council Consolidator grant SEED, CNCS-UEFISCDI PN-III-P4-ID-PCE-2016-0535, the EU Horizon 2020 Grant DE-ENIGMA, and SSF.

[13] We here use the model without posterior class probability adjustments.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[2] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 328–335, 2014.

[3] L. Bazzani, d. N. Freitas, H. Larochelle, and V. Muriono. Learning attentional policies for tracking and recognition in video with deep networks. In *International Conference on Machine Learning*, 2011.

[4] N. J. Butko and J. R. Movellan. Infomax control of eye movements. *IEEE Transactions on Autonomous Mental Development*, 2(2):91–107, 2010.

[5] J. Caicedo and S. Lazebnik. Active object localization with deep reinforcement learning. In *IEEE International Conference on Computer Vision*, 2015.

[6] J. Carreira and C. Sminchisescu. CPMC: Automatic Object Segmentation Using Constrained Parametric Min-Cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

[7] X. Chen and A. Gupta. An implementation of faster rcnn with study for region sampling. *arXiv preprint arXiv:1702.02138*, 2017.

[8] X. Chen and A. Gupta. Spatial memory for context reasoning in object detection. *arXiv preprint arXiv:1704.04224*, 2017.

[9] X. S. Chen, H. He, and L. S. Davis. Object detection in 20 questions. In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, pages 1–9. IEEE, 2016.

[10] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[11] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks. *arXiv preprint arXiv:1605.06409*, 2016.

[12] A. Dosovitskiy and V. Koltun. Learning to act by predicting the future. *arXiv preprint arXiv:1611.01779*, 2016.

[13] M. Everingham, L. V. Gool, C. K. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/.

[14] R. Girschick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2014.

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

[16] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*, volume 9, pages 249–256, 2010.

[17] A. Gonzalez-Garcia, A. Vezhnevets, and V. Ferrari. An active search strategy for efficient object class detection. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2015.

[18] B. Goodrich and I. Arel. Reinforcement learning based visual attention with application to face detection. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2012.

[19] K. Hara, M.-Y. Liu, O. Tuzel, and A.-M. Farahmand. Attentional network for visual object detection. *arXiv preprint arXiv:1702.01478*, 2017.

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

[21] 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*, pages 770–778, 2016.

[22] Z. Jie, X. Liang, J. Feng, X. Jin, W. Lu, and S. Yan. Tree-structured reinforcement learning for sequential object localization. In *Advances in Neural Information Processing Systems*, pages 127–135, 2016.

[23] S. Karayev, T. Baumgartner, M. Fritz, and T. Darrell. Timely object recognition. In *Advances in Neural Information Processing Systems*, 2012.

[24] S. Karayev, M. Fritz, and T. Darrell. Anytime recognition of objects and scenes. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2014.

[25] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[26] T. Kong, F. Sun, A. Yao, H. Liu, M. Lu, and Y. Chen. Ron: Reverse connection with objectness prior networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 2, 2017.

[27] X. Kong, B. Xin, Y. Wang, and G. Hua. Collaborative deep reinforcement learning for joint object search. *arXiv preprint arXiv:1702.05573*, 2017.

[28] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

[29] H. Larochelle and G. E. Hinton. Learning to combine foveal glimpses with a third-order boltzmann machine. In *Advances in Neural Information Processing Systems*, 2009.

[30] Z. Li, Y. Yang, X. Liu, S. Wen, and W. Xu. Dynamic computational time for visual attention. *arXiv preprint arXiv:1703.10332*, 2017.

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

[32] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. Reed. Ssd: Single shot multibox detector. *arXiv preprint arXiv:1512.02325*, 2015.

[33] Y. Lu, T. Javidi, and S. Lazebnik. Adaptive object detection using adjacency and zoom prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2351–2359, 2016.

[34] G. F. Lucas Paletta and C. Seifert. Q-learning of sequential attention for visual object recognition from informative local descriptors. In *International Conference on Machine Learning*, 2005.

[35] S. Mathe, A. Pirinen, and C. Sminchisescu. Reinforcement learning for visual object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2894–2902, 2016.

[36] S. Mathe and C. Sminchisescu. Action from still image dataset and inverse optimal control to learn task specific visual scanpaths. In *Advances in Neural Information Processing Systems*, 2013.

[37] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu. Recurrent models of visual attention. *arXiv*, 2014.

[38] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. In *NIPS Deep Learning Workshop*, 2013.

[39] P. Rantalankila, J. Kannala, and E. Rahtu. Generating object segmentation proposals using global and local search. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2014.

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

[41] J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. *arXiv preprint arXiv:1612.08242*, 2016.

[42] 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*, pages 91–99, 2015.

[43] 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.

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

[45] S. Singh, D. Hoiem, and D. Forsyth. Learning a sequential search for landmarks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3422–3430, 2015.

[46] R. S. Sutton and A. G. Barto. *Reinforcement Learning*. MIT Press, 1998.

[47] 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*, pages 1–9, 2015.

[48] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104, 2013.

[49] R. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 1992.

[50] D. Yoo, S. Park, K. Paeng, J.-Y. Lee, and I. S. Kweon. Action-driven object detection with top-down visual attentions. *arXiv preprint arXiv:1612.06704*, 2016.

[51] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 3357–3364. IEEE, 2017.

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