Coarse-to-Fine Q-Learning for Object Localisation on VHR Images

Dario Zanca and Mason McGill

Feb-Apr 2018

1 Introduction

General motivations and short problem description go here.

Literature review goes here. [6] [3] [1] [2] [5]

In this work we define a coarse-to-fine learning method for fast localization of objects of interest (OOIs) in very high resolution (VHR) images. We learn latent policies in a hierarchy defined by the possible scales (from coarse to fine) at which input can be sampled, to process with high-resolution only certain parts of the input that are good candidates for containing the OOI.

2 Problem definition

2.1 Environment

Given a map

$$\mathcal{M} : \mathcal{R}^2 \to \mathcal{R}^n$$

of pixels, we can make queries

$$q(x, y, z) = [\text{image}]$$

which output a crop of the map of fixed resolution $n \times n$, centered in a certain position (x, y) and zoom z[1].

2.2 Task

We want to localise all the instances of an object of interest (OOI) in $\mathcal{M}$.

[1] In particular, the number z allows us to compute the area corresponding to a single pixel at that zoom level as

$$\frac{A_0}{4^z n^2},$$

where A_0 is the are of [image] at $z = 0$.

2.3 Assumptions

Assumptions about the environment:

- The set of possible zooms is discrete, i.e. $\mathcal{Z} := \{z_1, ..., z_k\}$, where z_1 provides the coarser scale and z_k the finest.
- Assuming the fixed resolution $n \times n$, a query at zoom z_i samples from an area four times bigger then z_{i+1}.

Assumptions about the problem:

- OOI are visible only at the finest scale that correspond to z_k; a model $\mathcal{O}$ for OOI is given (pre-trained).
- $\mathcal{M}$ is a very high resolution (VHR) image.
- Sensing costs a lot. We want to minimize the number of queries performed. On the other hand, we assume cheap processing and infinite memory.
- OOI are sparsely distributed on $\mathcal{M}$; a linear search is not convenient. This implies that there exists at least one zoom level before the finest, in which feature observable at that level are correlated with the presence of OOI.

3 Solution 1: Where to look

3.1 Model

We want to determine the *value* function

$$f_\pi(s, a),$$

that defines the expected return starting from a state $s =$ [image], taking the action a, and thereafter following the policy π. The action a indicates the position (u, v) and the zoom z at which make the next query,

$$a = (u, v, z).$$

More precisely, and omitting the dependence on π,

$$f(s, a) = E\left[r + \gamma \max_{a'} f(s', a')|s, a\right],$$

where r is a reward signal related with the correct localization of non-observed OOIs. In other words, f is a function that quantifies, given the current observation s, *where to look* next in order to maximize the value.

Losing generality for the sake of simplicity, we assume that decisions at each level of zoom z_i are independent. Given the set of function

$$\{f_1, ..., f_{k-1}\} \cup \{f_k \equiv 0\}$$

we define f by cases

$$f(s,a) = \Big\{ f_i(s,a_i), \text{ if } z = z_i,$$

and f_i indicates where to look next with a finer level of zoom z_{i+1}. Moreover, we limit the possible next locations (u,v) in a sparse grid. So that, a_i is chosen from a small set A, with $|A| = m \times m << n \times n$, that depends on the previous observation,

$$a_i \in A = \{...\}^2.$$

3.2 Inference

At each step, we select a location for each zoom level, from the coarsest to the finest. We compute a *fixed saccade activation path*

$$a = (a_1, a_2, ..., a_{k-1}),$$

where,

$$a_i = \arg\max f_i(s_i, \cdot)$$

and, if (u_o, v_0) are the origin's coordinates,

$$s_i = q(u_{i-1}, v_{i-1}, z_i).$$

This path uniquely determines a *final* patch

$$q(u_{k-1}, v_{k-1}, z_k) \subset \mathcal{M}$$

sampled at the finest scale, where the model $\mathcal{O}$ of the object is defined.

Since we are interested in the localisation of non-already-explored OOIs, we add the extra channel $\mathcal{V}$ in input that keeps memory of the locations already visited, so that,

$$f_i : \mathcal{R}^{n^2 \times m^2 \times 2} \to \mathcal{R}.$$

3.3 Training

Since the query-function $q(\cdot)$ is not differentiable, we train the model with reinforce. The environmental action would correspond to the localisation decision, and the reward would reflect if the decision is correct.

The reward signal is defined as follow,

$$r(a_i) = \begin{cases} 0, \, i < k-1 \\ \mathcal{O}(s_k), \, i = k-1 \end{cases}$$

[2]Set A is chosen such that by making queries at each possible next location in the successive zoom level produce a tessellation of the current state.

where we assume the object model $\mathcal{O}(\cdot)$ outputs 1 if the OOI is present in the analyzed patch and 0 otherwise[3].
Then,

$$f_i(s_i, a_i) = E\left[r(a_i) + \gamma \max_{a_{i+1}} f_{i+1}(s_{i+1}, a_{i+1}) | s_i, a_i\right],$$

Notice that, when choosing $\gamma = 1$ and not giving any intermediate reward, the problem strictly simplifies to the one of learning $k-1$ *myopic* agents.

We then treat the right-hand side quantity into brackets as target and minimize the MSE function

$$L = \sum_{i=1}^{k-1} \left(\underbrace{r(a_i) + \gamma \max_{a_{i+1}} f_{i+1}(s_{i+1}, a_{i+1})}_{} - f_i(s_i, a_i) \right)^2.$$

3.4 Experiments

In order to produce an experimental verification of the model, we create environments compatible with the assumptions made in the previous sections. Different functions are learned at each zoom level. This results in a hierarchical organization that efficiently locates OOIs on the given map. Performance is evaluated in terms of total reward.

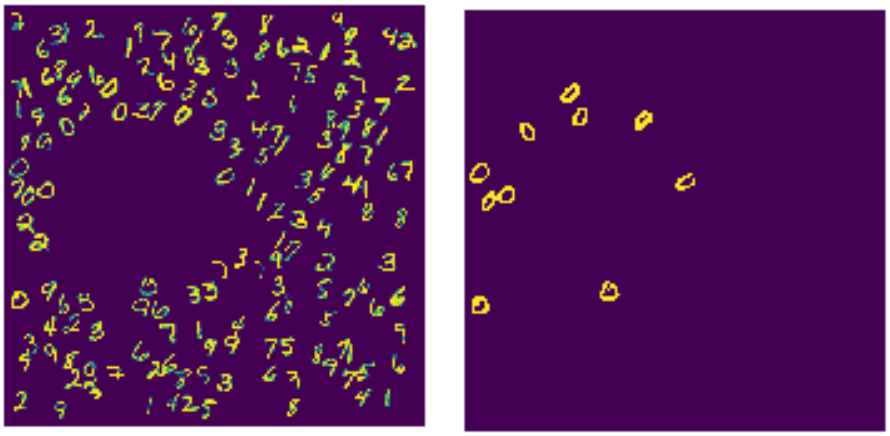
3.4.1 Toy example

Environment. A synthetic environment is defined in which OOIs are zero digits from the Mnist dataset [4], while other digits from the same dataset are used as distractors. Map $\mathcal{M}$, in which digits are distributed, has a much higher resolution ($\geq 2^8$ times) than OOIs. Queries return a fixed resolution patch, 28-by-28. In the first zoom levels, no digits are recognizable. For this reason, environment is generated such that the zeroes distribution creates a pattern that can be observed even at the first zoom level. In particular, the zeros are distributed around an empty square area. An example of this environment is shown in fig. 2. After training for 5000 episodes, value function indicate at each level which tiles guarantee a higher reward. An example is shown in Fig. **??**.

Experimental setup: fixed budget of queries. At each iteration, an activation path through the zoom layers selects a tile in the finest layer. Each observation has a fixed quieries cost, equal to the number of layers[4]. A fixed budget of n_q queries is given for each map. In our experiments we set $n_q = 10$. This correspond to considering an episodic setting of fixed length. Reward is 1 when selecting tiles containing unobserved OOIs, and it is 0 otherwise.

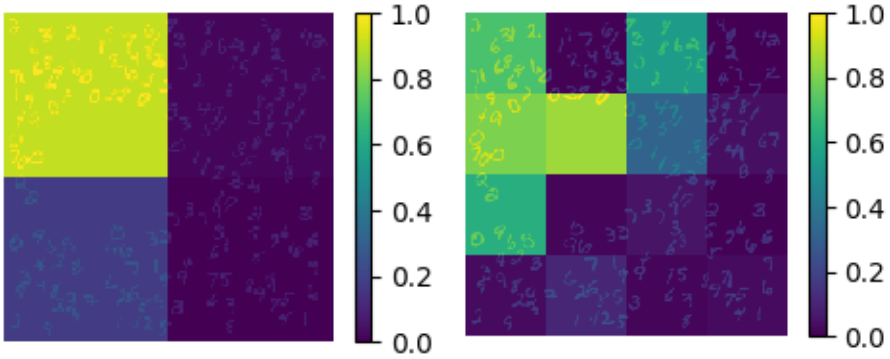
[3] Analogously, we can assume the object model $\mathcal{O}(\cdot)$ outputs the number of OOIs in the analyzed patch

[4] Depending on memory resources, this process would benefit from storing already-requested-patches.



(a) Map $\mathcal{M}$ (b) Labels

Figure 1: Example of environment with ten OOIs distributed close to empty square area.



(a) f_1 (b) f_2

Figure 2: Example of value functions f_1 and f_2 after 5000 episodes of training.

Experiment results. Performance during training are shown in Fig. 3. The total reward is computed by summing up rewards for each action in during the n_q queries exploration. The behavior of the learned functions is illustrated in Fig. 4, 5 and 6, among different stages in training. From the examples, it is clear that reward is not equal to the number of localized OOIs. It might be the case that more instances of the OOI are present in a single tile, or that two or more tiles contain only a part of the same OOI but providing then more rewards. In the case of a more specific task (like counting the instances or marking the exact middle point of every OOI) further processing is required.

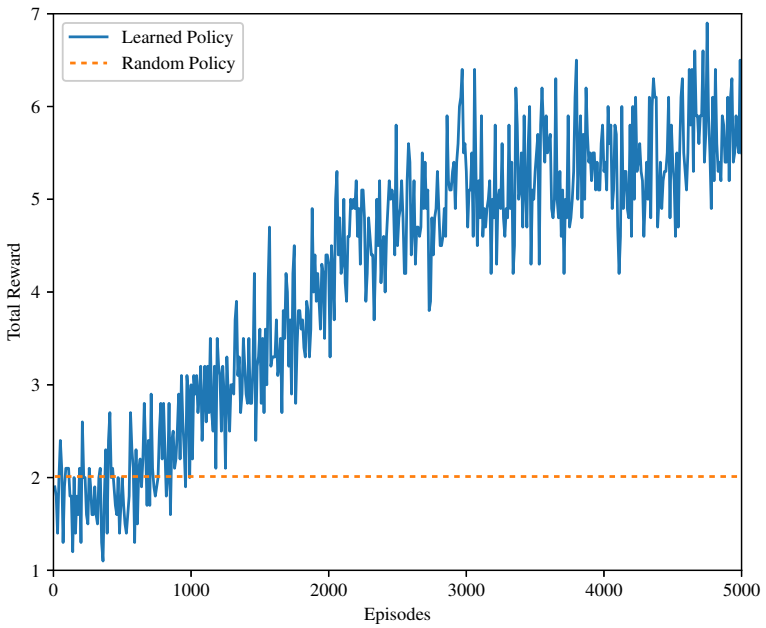


Figure 3: **Total reward during training.** For each episode, a new environment is generated.

3.4.2 Localise swimming pools in Los Angeles

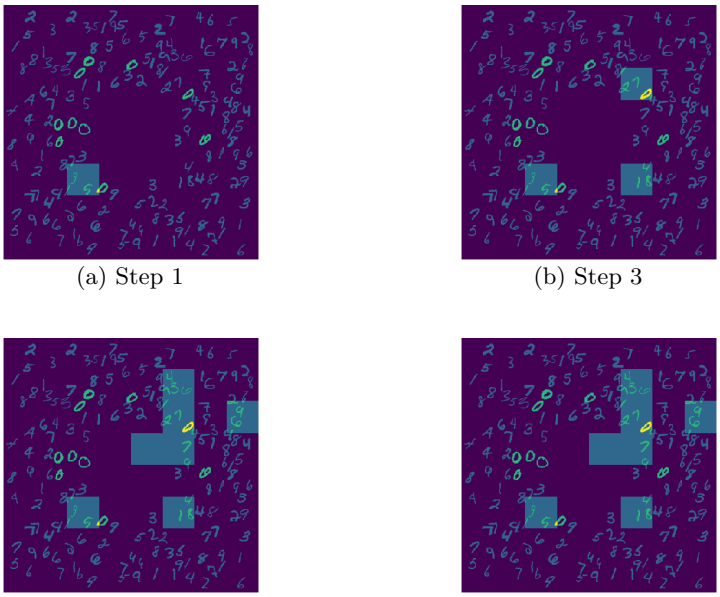
Environment. In the setup offered by the Google maps APIs [5], we consider the problem of locating swimming pools on satellite maps. The pools are only recognizable from a certain level of zoom onwards. However, policies can be learned at the earliest zoom levels to increase the likelihood of locating a swimming pool. For example, swimming pools are unlikely to be found in the middle of the ocean or in the city centre, while in residential areas with trees on the edge of the city they become increasingly easy to find.

Preliminary analysis. A pre-trained model of swimming pool is needed in order to evaluate performance of the defined scheme in this task.

References

[1] Andrew G Barto and Sridhar Mahadevan. Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems*, 13(4):341–379, 2003.

[5] https://cloud.google.com/maps-platform/



(a) Step 1 (b) Step 3

(c) Step 7 (d) Step 10

Figure 4: **After 1000 episodes of training.** Zeros are slightly highlighted for visualization purposes only. The light blue boxes mark the tiles selected until each step.

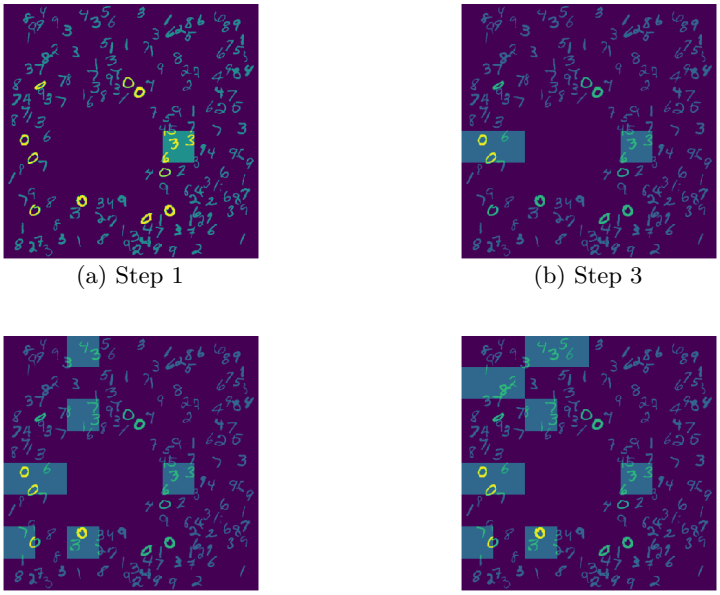
[2] X. Chen, S. Xiang, C. L. Liu, and C. H. Pan. Vehicle detection in satellite images by hybrid deep convolutional neural networks. *IEEE Geoscience and Remote Sensing Letters*, 11(10):1797–1801, Oct 2014.

[3] Thomas G Dietterich. The maxq method for hierarchical reinforcement learning. In *ICML*, volume 98, pages 118–126. Citeseer, 1998.

[4] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[5] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2003.

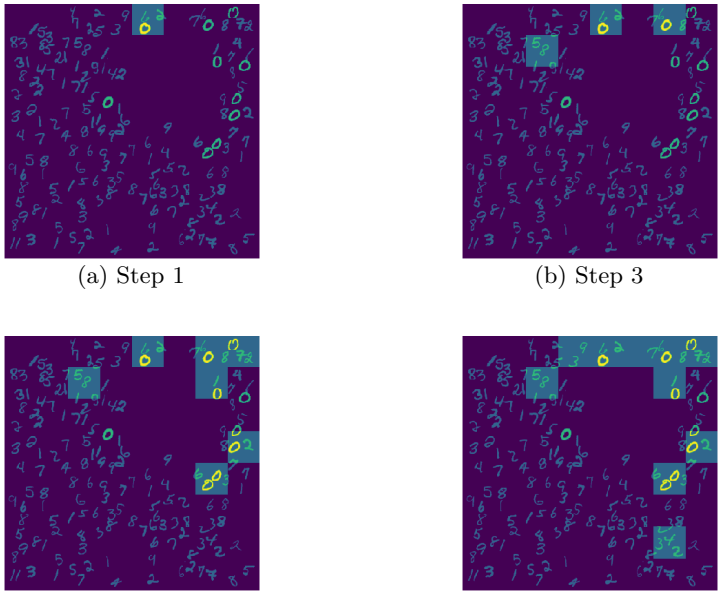
[6] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–511–I–518 vol.1, 2001.



(a) Step 1 (b) Step 3

(c) Step 7 (d) Step 10

Figure 5: **After 3000 episodes of training.** Zeros are slightly highlighted for visualization purposes only. The light blue boxes mark the tiles selected until each step.



(a) Step 1 (b) Step 3

(c) Step 7 (d) Step 10

Figure 6: **After 5000 episodes of training.** Zeros are slightly highlighted for visualization purposes only. The light blue boxes mark the tiles selected until each step.