Square Localization for Efficient and Accurate Object Detection*

Cewu Lu[♯,‡] Yongyi Lu[§] Hao Chen[†] Chi-Keung Tang[§]
[§]The Hong Kong University of Science and Technology [‡]Stanford University
[♯]Shanghai Jiao Tong University [†]The Chinese University of Hong Kong

Abstract

The key contribution of this paper is the compact square object localization, which relaxes the exhaustive sliding window from testing all windows of different combinations of aspect ratios. Square object localization is category scalable. By using a binary search strategy, the number of scales to test is further reduced empirically to only $O(\log(\min\{H, W\}))$ rounds of sliding CNNs, where H and W are respectively the image height and width. In the training phase, square CNN models and object co-presence priors are learned. In the testing phase, sliding CNN models are applied which produces a set of response maps that can be effectively filtered by the learned co-presence prior to output the final bounding boxes for localizing an object. We performed extensive experimental evaluation on the VOC 2007 and 2012 datasets to demonstrate that while efficient, square localization can output precise bounding boxes to improve the final detection result.

1. Introduction

Object detection and recognition has been a long standing problem in computer vision. The advent of state-of-the-art machine learning techniques has made important breakthrough in recent years. Given a testing image, one of the main tasks is to localize (that is, find the coordinates of an image window that contains) a semantic object. Once an image window is given, the remaining step is object recognition, which has been well addressed recently by the state-of-the-art convolutional neural network (CNN) [14] which labels the most likely object categories (or background).

The exhaustive approach is what researchers call "sliding window" testing, where image windows of all possible sizes and aspect ratios at all possible locations are tested. Not surprisingly, the exhaustive sliding window was quickly written off as impractical due to its prohibitive cost, and most researchers resort to compromises and consider significantly fewer image windows. Representative approaches include the effective selective search method [20]. In practice, the exhaustive search can be sidestepped [1] but quantization will be involved in the form of uniform object proposal sampling.

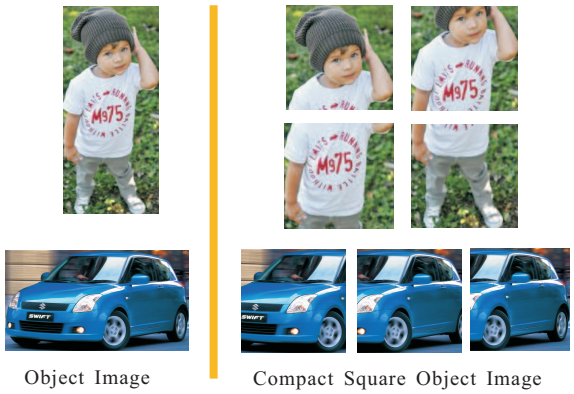

Figure 1. Object images and their compact square images.

Accuracy	CNN model	Human
whole image	87.3%	97.6%
compact square image	84.6%	95.4%

Table 1. Empirical recognition accuracy of the CNN model [14] and human recognition on the VOC 2007 dataset. We ask people to label the category based on compact square object (CSO) images. With more careful parameter tuning of the CNN we believe the above accuracy can be further improved.

While the exhaustive method searches over all possible positions, scales and aspect ratios of image windows, given an $H \times W$ image, the number of aspect ratios and scales combination is $O(HW)$ while the number of scales is only $O(\min(H, W))$. Therefore, if we could relax the aspect ratio requirement, the sliding window would be much more efficient. Interestingly, we observe that both hu-

*This research is supported by the Research Grant Council of the Hong Kong Special Administrative Region under grant number 619711.

[1] For example, a fixed set R of aspects (which can be learned form the data) is used. Also in practice the sampling for the scale is proportional to the object size, to have a constant variation of the object scale, a logarithmic variation in terms of H (the image height) can be used. The time complexity is thus $O(R\log(H))$ which is comparable to our $O(\log(\min\{H, W\}))$.

man vision and state-of-the-art computer vision are very robust in object recognition when presented with what we call a *compact square object* (CSO) image, defined as the largest square region (hence fixed aspect ratio) within a tight bounding box of the object in the image where some examples are given in Figure 1. Note that CSO images do not contain the whole object. Table 1 tabulates our empirical results on the top-1 recognition accuracy based on CSO images created from the VOC 2007 dataset. Specifically, We train a 20 classes plus background CNN model respectively on the whole images and CSO images, and evaluate the accuracies by both computer (CNN model) and human. Both recognition accuracies remain high on CSO images, showing that objects can be recognized by seeing CSO image with high chance.

This observation had inspired us to localize object regions by localizing its CSO images, thus free of aspect ratio testings, where on the other hand concatenating the detected squares will reproduce the entire object. As our experiments will show, our square localization scheme is efficient and accurate as well, outputting precise bounding boxes to increase detection rate in mAP. Computationally, we only need to apply state-of-the-art recognition (e.g. CNN) everywhere in different scales to achieve *compact square localization*. Thanks to current GPU technology, GPU implementation of CNN is extremely fast. As we will show, we only need to implement $O(\min\{H, W\})$ rounds what we call *sliding CNN*. The logarithmic time complexity is achieved by employing a binary search strategy that further cuts down the number of landmark scales to be tested, when the information content along a scale spectrum is largely redundant. Large scale empirical studies confirm that only $O[\log(\min\{H, W\})]$ rounds of sliding CNN testing is run in practice, though the worse complexity is still $O(\min\{H, W\})$ which is still impressive compared with the exhaustive sliding windows. As we will show, our sliding CNN is category scalable. That is, the running time is largely independent of the number of object categories.

Albeit the promise, this scheme faces a technical challenge in practice: the response map suffers from clutters, which makes it difficult to accurately delineate the true compact square object region. To address this problem, we propose a novel and effective object co-presence filter to drastically reduce clutter response, thus making amenable the output of the relevant bounding boxes. Our technique gives adequate consideration of the object spatial co-presence prior in a robust statistical manner. Full implementation details of object co-presence for decluttering a response map are available in the appendix section.

Comparison with Part-Based Model At first glance our scheme bears some resemblance to the part-based model [7], but readers may have noticed the significant differences: first, part-based model seek appearance representation in part-level (e.g. wheels in a car); compact square images are still object-level representation. Second, part-based model learns a part location structure model for each category, which makes it not scalable with increasing number of categories [5]. In contrast, our method simply concatenates detected compact square object images with no special model for any category, making our method easily scalable with increasing number of object categories.

2. Related Work

In this section, we focus on object localization which is a core issue in object detection and the main subject of the paper.

The part-based model has been widely adopted [9, 8, 7] to model an object as a composition of various object parts. Dean *et al*. [3] proposed a hash-based approach for efficient part detection, and the method has achieved good results. These methods score regions by considering parts appearance and location distribution. Impressive performance has been achieved thanks to the discriminative and effective hand-crafted features such as HOG. Our method bears some resemblance in their use of sliding window for computing part appearance scores. However, these methods must learn different part patterns for different object categories. In other words, it is not scalable with increasing number of object categories, and thus less amenable to modern large-scale datasets such as the Imagenet [17]. Another difference when compared with the part model method is that it uses a part level representation, whereas our compact square image is very close to an object level representation.

Significant research attempt has been made to address the complexity issue of object-level sliding window testing. Lampert *et al*. [15] proposed a branch-and-bound strategy to avoid testing all sub-windows in image. With recent developments in deep learning, such as the notable work by Krizhevsky *et al*. [14] which significantly improved the state of the arts in object recognition, work on how to extend the success of deep neural networks in object detection has been intensified. In RCNN [10], the localization task is done by selective search [20]. Thousands of objectness bounding boxes are produced. Then, deep CNN classifier is applied on each candidate bounding box to label the object which is time consuming. Instead of applying CNN on each candidate separately, in SPP-Net [12], fixed-length feature on proposals are extracted on the last convolutional layer using max pooling, which accelerates RCNN by a considerable factor. Albeit the compelling detection accuracy, these methods still belong to the pipeline involving pre-computing object proposals, and intractable sliding window problem remains unsolved.

The great success of RCNN has motivated researchers to study various approaches to objectness proposal genera-

tion [21, 2]. One approach is to regress a set of object locations directly using deep learning. The pioneering work is Overfeat [18] which integrated recognition, localization and detection, where the location bounding boxes are provided by a boxes regressor. Recently, a notable work [5] also adopted bounding boxes regression. Comparing to [18], this method can predict a set of class-agnostic bounding boxes by considering both bounding boxes coordinate regression and their likelihood of containing any object of interest. These two considerations are integrated into a unified optimization model in their training phase. Our work goes beyond the above methods by considering object co-presence priors to output reliable bounding boxes and is related to previous work [4] which models context of object layout to post-process detection outputs. The scenario is however more complex for selecting compact square object (CSO) images to form object windows.

In summary, previous works avoid or sidestep the exhaustive sliding windows. In this paper, we propose a novel strategy for localizing compact squares images which enables the CNN to operate in a sliding window manner efficiently. In contrast to most of previous methods, our method is category scalable.

3. Square Localization

In this section, we present our square localization pipeline. We first train *square CNN models* on compact square object images. Given square CNN models, we implement sliding CNN testing to test everywhere in the image to produce response maps at each scale.

3.1. Square CNN Model Learning

In the training phase, unlike previous work that trains the CNN models on the whole object image, we train the models on compact square object images, or CSO images for short, and thus call the trained model *square CNN model*. We describe here how to prepare CSO images for training.

Given a $H \times W$ object image, without loss of generality, we assume $H \geq W$. We require that the sampled CSO images has $1 - \tau$ overlap ($\tau = 0.2$ in our paper) with its neighboring CSO images. In other words, we will uniformly sample $(H - W)/(\tau W)$ CSO images with size $W \times W$ along the vertical direction. Examples of the original object images and their corresponding CSO images are shown in Figure 1.

Let n be number of categories in the training data. We train a $n + 1$ classes ways (square) CNN models on the square images extracted from the training data. The additional way accounts for the background class in the n-object category. We train the models on k ($k = 3$ in paper) different input size, namely, $\{256 \times 256, 128 \times 128, 64 \times 64\}$. Our deep network architecture follows the setting of the AlexNet [14], with the only difference being the size of the input image in the first layer.

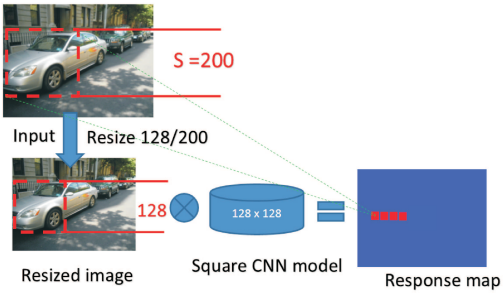

Figure 2. A sliding CNN example. The size of the square region to be tested is $s = 200$. We choose the square CNN model $u = 128$ from $\{256, 128, 64\}$ whose input size is closest to s. Then, we resize the original image with u/s. The 200×200 square window in the original image now becomes 128×128 window in the resized version . Sliding CNN testing is applied on the resized image. A pixel in the response map then corresponds to a 200×200 square window in the original image.

3.2. Sliding CNN

Now, we want to apply the learned square CNN models on a testing image at different square scales. Recall that each square CNN model requires the input size to be of size $\{256 \times 256, 128 \times 128, 64 \times 64\}$. Let s be the window size under testing. We choose a square CNN model whose input size u ($u = \{256, 128, 64\}$) is closest to s, and resize the testing image with factor u/s. After applying sliding CNN using the chosen square CNN model on the resized image, the output of the square CNN model represents a $s \times s$ square region in the original image. An example is illustrated in Figure 2.

Our square model is applied on the entire resized image with a stride of $u/8$, which is sufficient to cover a very dense set of $u \times u$ square regions. Specifically, given any $u \times u$ square region in the resized image, it has at least $0.88u^2$ ($(1 - \frac{1}{16})^2 u^2$) overlap with one of the tested $u \times u$ square regions. Our GPU implementation is similar to the denseNet [13] which is very fast running at 0.05 second for a million-pixel image on Nvidia's K40 graphics board. The difference is that the denseNet only operates on the first five "convolutional" layers of a network, whereas we operate on the fully-connected layers in a convolutional manner, and use the CNN scores for indicating square regions.

3.3. Scale Selection by Binary Search

We have described above how to use square CNN model given a testing window of arbitrary size. Ideally, we should consider all possible square window scales in the testing image which amounts to $O(\min\{H, W\})$ running time complexity. In practice, we found that the outputs within a range of scales are similar. Therefore, we only test a number of landmark square scales to reduce the computation.

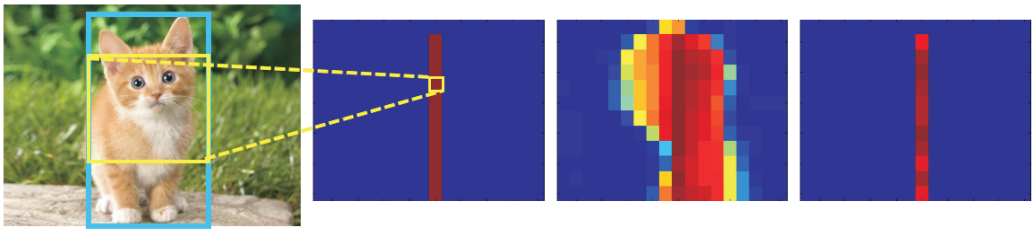

Figure 3. (a) input image, (b) ideal response map; the red pixels are "indicator pixels". An indicator pixel denotes a compact square object image. The entire object can be readily localized (the rectangle in blue) by concatenating indicator pixels, (c) cluttered response, the response smaller than 0.2 is assigned to zero, (d) result filtered using our object co-presence prior.

We adopt a binary search strategy by recursively dividing a given scale range into two halves until the scales within the divided range have very similar output. To this end, we define a similarity measure of two output. Since the output size for two different scales is different, we resize each output to the same size for comparison, where the chosen size is their geometric mean. Then, given two scales L and S, we denote M_L^i and M_S^i as the sliding CNN response output for the i^{th} category. Denoting the respective resize operator as $\varphi_{i,L}(\cdot)$ and $\varphi_{i,S}(\cdot)$, we can write the distance measure as

$$d(S,L) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\Pi_i}\|\varphi_{i,L}(M_L^i) - \varphi_{i,S}(M_S^i)\|_2 \qquad (1)$$

where Π is the normalization factor, i.e., the number of pixels in $\varphi_{i,L}(M_L^i)$. If $d(S,L)$ is sufficiently small, the outputs within scale range $[S,L]$ are similar to each other. In other words, we do not need to test the scales within $[S,L]$ since we already have the respective results at scale S and L.

Pseudo-code 1 summarizes the binary search strategy. Calling $scaleSelection(S_{\min}, L_{\max})$ will evaluate the scale continuum $[S_{\min}, L_{\max}]$, where $L_{\max}$ and $S_{\min}$ are respectively the largest and smallest scales under consideration. In our paper, we set $\varepsilon = 0.03$, $L_{\max} = 500$ and $S_{\min} = 32$. In practice, we found the number of selected scales is $O[\log(L_{\max} - S_{\min})] \leq O[\log(\min\{H,W\})]$. Unlike the denseNet which pre-assigns testing scales and thus may result in missing landmark scales or testing redundant scales, our selection scheme is adaptive to testing data.

Algorithm 1 Pseudo-code of Scale Selection Function

Function $scaleSelection(S, L)$
if $d(S,L) < \varepsilon$ or $L = S + 1$ **then**
  return
else
  selecting scale $M = \sqrt{LS}$
  call $scaleSelection(S, M)$
  call $scaleSelection(M + 1, L)$
end if

With scale selection, we obtain m landmark scales to test. For each scale, the learned square CNN is run as a universal region on the resized image and outputs n classes response maps, where each pixel value indicates likelihood score of a given category in a square region. In total we have $m \times n$ response maps for different scales and categories.

3.4. Bounding Boxes Extraction

The next step is to extract bounding boxes given the response maps, which turns out to be a non-trivial task. Figure 3 (c) shows a response map produced by sliding CNN with a large number clutters.

To understand the cause of clutter, let us first consider what constitutes an ideal response map. For simplicity and without losing generality, a single response map is considered for a given object category at a given scale. Define "indicator pixel" to be the center pixel of a (ground-truth) compact square object image, as shown in Figure 3 (b). An ideal response map is one with a high impulse at an indicator pixel and zero responses at other pixels. By concatenating high impulse responses horizontally and vertically, the object bounding boxes in the original image (blue rectangle in Figure 3 (a)) can be readily extracted.

However, in practice, we often find pixels around indicator pixels to have high responses as well, since an object image with a minor shift in location can also produce a large response by the CNN model or any reasonable recognition technique. These cluttered responses pose challenges to the extraction of precise bounding boxes.

To reduce clutters, in this paper, we introduce the *object co-presence prior*: given an indicator pixel, its neighborhood pixels cannot be an indicator pixel for another object (with the same category), based on the assumption that the respective indicator pixels for two detectable objects cannot be in extreme close proximity (Figure 4 illustrates an example). This is similar to non-maximal suppression (NMS), while standard NMS technique cannot be applied since each indicator pixel represents a square image window. Full implementation details of object co-presence for decluttering a response map are available in the appendix section.

In practice two detectable objects with same category may be highly overlapping but this is quite rare. In the

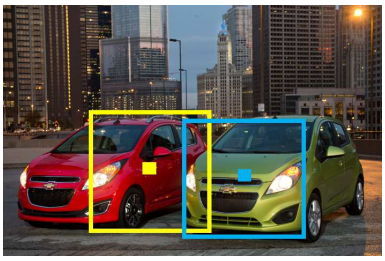

Figure 4. Two indicator pixels for two detectable objects should not be in extreme close proximity. The yellow and blue points are two indicator pixels for the respective square regions.

ILSVRC 2014 detection dataset, only 0.00034% of the object pairs have an overlap larger than 90%. In this paper, we conduct statistical analysis on a large volume of data to derive the object co-presence priors. Given a clutter response map (see Figure 3(c)), the remaining pixels after our co-presence filtering represent a compact square square object region (see (d)). We concatenate these indicating pixels vertically (or horizontally) to produce the final bounding rectangle to produce the entire object. In Figure 3, the bounding rectangle of the cat is the union of square regions indicated by red pixels (responses larger than 0.2) in (d).

4. Experiments

Our method is highly scalable in the number of categories in testing. That is, the computation overhead is arguably negligible when the number of object categories increases. In testing square CNN models, the computation overhead for each additional category only occurs in the last layer of CNN which is extremely small. Furthermore, the number of object categories exhibiting in the same image is limited, so most of the response maps are zero matrices which can be quickly discarded. In practice, the number of response maps need to be processed is nearly constant, independent of the number of object categories.

We first describe our experimental settings and then evaluate our method using the standard Pascal VOC 2007 and 2012 datasets. Then we analyze our localization performance. Since our contribution is primarily on localization, we will focus on our localization results rather than overall object detection results.

4.1. Experiment Settings

We generate $m \times n$ response maps for different scales and categories. We reject the response maps whose maximum response is smaller than 0.5. Typically, we only need to process about 10 to 20 response maps for each testing image, where the output bounding boxes are used for localizing an object. Since our focus is localization, for the recognition part, we use an established CNN recognizer to predict the category label and score, instead of checking the corresponding response map, since our square CNN model is trained on CSO images rather than whole object images. In detail, the regions under the output bounding boxes are first resized to 256×256, followed by applying another CNN model learned from whole images (also resized into 256×256). The reason of using another CNN is as follows: although in our square framework we can output the object score directly upon concatenating the full object box, we found a 2–3mAP difference in comparison with assigning score using a trained CNN geared to whole object images. Note that the number of output boxes is small after our localization and thus the extra time is minor for applying a whole-object CNN classifier. We will report the mAP performance of both square CNN and an additional pass of applying whole-object CNN.

4.2. Pascal VOC

The Pascal Visual Object Classes (VOC) Challenge [6] is one of the most common benchmarks for evaluating object detection performance. The labeled objects belong to one of the 20 object categories occurring in complex scenes with different scales and aspect ratios.

Training methodology. Since the data size of VOC is not large compared with more recent big datasets, we first train three square CNN models with different input sizes$\{256^2, 128^2, 64^2\}$ on the ILSVRC 2014 object detection dataset which has 200 object categories. Then, three square CNN models with 21 classes ways for VOC 2007 (2012) are fine-tuned on them respectively.

Detection Improvement. We extract bounding boxes for all the testing images. The image regions under the output bounding boxes are resized to 256×256. To make the comparison fair, since we focus on localization, we use the same recognition CNN models provided in the RCNN website. Thus, any recognition improvement is directly due to our better localization performance. Tables 2 and 3 show the detection AP for different object categories using different localization strategies under the RCNN framework on the VOC 2007 and VOC 2012 datasets respectively. The annotation of testing set of VOC 2012 is not available so we submit our results to the VOC evaluation server. [2]

As we can see, our localization has made improvement in comparison to standard selective search [20] and edge box [21]) given the same CNN recognition model. Note that while [5] provides a localization scheme their resulting bounding boxes are unavailable. Given the same recognition model, better mAP indicates better localization. Sample results are shown in Figure 5. Note that in comparison with standard RCNN which produces a few thousands candidate boxes, we produce only 100 to 300 bounding boxes per image. On the other hand, we are interested in the per-

[2]VOC 2012 testing set results using our square localization: `http://host.robots.ox.ac.uk:8080/anonymous/CUY6QS.html`

VOC 2007 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
R-CNN (SS)	64.2	69.7	50.0	41.9	32.0	62.6	71.0	60.7	32.7	58.5	46.5	56.1	60.6	66.8	54.2	31.5	52.8	48.9	57.9	64.7	54.2
R-CNN BB (SS)	68.1	72.8	**56.8**	43.0	36.8	66.3	74.2	67.6	34.4	63.5	54.5	61.2	69.1	68.6	58.7	33.4	62.9	51.1	62.5	64.8	58.5
R-CNN (EB)	63.7	64.8	50.3	42.2	31.4	59.4	68.4	61.3	31.2	54.7	42.6	58.6	63.1	64.2	56.3	30.2	53.1	45.2	58.6	66.2	53.2
Ours	**73.1**	**76.1**	55.8	**45.6**	**40.6**	**71.4**	**80.0**	**69.4**	**39.1**	**67.4**	**57.6**	**64.7**	**72.5**	**72.3**	**62.6**	**35.7**	**64.6**	**56.2**	**65.6**	**69.7**	**62.0**

Table 2. Per-class average precision score on PASCAL VOC 2007 dataset. We fix the recognition part and replace the localization part. 'SS' [20] and 'EB' [21] respectively denote selective search and edge box. 'BB' means the bounding boxes regression proposed in [20], which can be regarded as a localization method.

VOC 2012 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
R-CNN (SS)	68.1	63.8	46.1	29.4	27.9	56.6	57.0	65.9	26.5	48.7	39.5	66.2	57.3	65.4	53.2	26.2	54.5	38.1	50.6	51.6	49.6
R-CNN BB (SS)	71.8	65.8	52.0	34.1	32.6	59.6	60.0	69.8	27.6	52.0	**41.7**	69.6	61.3	68.3	57.8	29.6	57.8	40.9	59.3	54.1	53.3
R-CNN (EB)	68.3	63.8	46.5	29.6	27.8	56.6	54.5	63.3	26.2	45.3	37.8	66.4	57.4	64.3	50.5	24.7	52.8	39.3	50.4	52.8	48.9
Ours	**76.8**	**71.2**	**61.2**	**45.1**	**35.9**	**62.5**	**60.9**	**75.5**	**31.3**	**58.3**	39.4	**73.8**	**68.6**	**73.1**	**60.9**	**33.1**	**59.1**	**41.0**	**66.3**	**57.6**	**57.7**

Table 3. Per-class average precision score on PASCAL VOC 2012 dataset. We fix the recognition part and replace the localization part. 'SS' [20] and 'EB' [21] respectively denote selective search and edge box. 'BB' denotes the bounding boxes regression proposed in [20], which can be regarded as a localization method.

VOC 2012 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
NUS_NIN_c2000 [16]	80.2	73.8	61.9	43.7	**43.0**	**70.3**	**67.6**	80.7	**41.9**	**69.7**	**51.7**	78.2	75.2	76.9	**65.1**	**38.6**	**68.3**	**58.0**	68.7	**63.3**	**63.8**
R-CNN VGG [10]	76.8	70.9	56.6	37.5	36.9	62.9	63.6	81.1	35.7	64.3	43.9	80.4	71.6	74.0	60.0	30.8	63.4	52.0	63.5	58.7	59.2
Feature Edit [19]	74.6	69.1	54.4	39.1	33.1	65.2	62.7	69.7	30.8	56.0	44.6	70.0	64.4	71.1	60.2	33.3	61.3	46.4	61.7	57.8	56.3
SDS [11]	69.7	58.4	48.5	28.3	28.8	61.3	57.5	70.8	24.1	50.7	35.9	64.9	59.1	65.8	57.1	26.0	58.8	38.6	58.9	50.7	50.7
Ours	**80.8**	**74.3**	**64.6**	**46.4**	40.3	65.5	64.6	**84.9**	38.6	68.2	40.2	**82.0**	**77.6**	**79.2**	62.6	35.7	65.7	51.9	**69.7**	59.3	62.7

Table 4. Comparison with state of the arts on PASCAL VOC 2012 test dataset. We simply combine our localization with VGG network [1] without using extra data and state-of-the-art results.

number of boxes	10	20	30	40	50	60	70	80	90	100
selective search [20]	0.02	0.03	0.06	0.09	0.10	0.14	0.17	0.20	0.25	0.27
Edge box [21]	0.01	0.03	0.07	0.12	0.14	0.17	0.20	0.25	0.30	0.34
[5] (max-center + 2 scale)	**0.53**	**0.62**	**0.64**	**0.68**	**0.70**	0.71	0.72	0.73	0.75	0.75
[5] (max-center + 1 scales)	0.52	0.60	0.63	0.64	0.67	0.68	0.70	0.71	0.72	0.73
[5] (one shot)	0.47	0.52	0.53	0.53	0.54	0.54	0.55	0.55	0.56	0.57
ours	0.37	0.41	0.54	0.60	0.64	**0.73**	**0.76**	**0.80**	**0.81**	**0.82**

Table 5. Detection rate of class object vs number of bounding boxes per image suggested by [5].

formance without applying a whole-object CNN, so we also compute the mAP of scoring an object image by averaging selected square region indicator scores in the response map. The resulting mAP in VOC 2007 is 59.3 or a 2.7 mAP drop.

Lesion Study. We also conduct a lesion study on the VOC 2007 dataset. First, we replace the binary scale search by uniform scale sampling. Specifically, we set the smallest and largest scale to be 32 and 500 respectively, and uniformly sample Z scales from $[32, 500]$, where Z is the number of scales used in our binary scale search. The result is 57.1 mAP (i.e., with 4.9 mAP drop). Second, we replace the square detection system with a sliding window search run for $K = 14$ different discrete aspect ratios which cover ratio $\frac{1}{n}$ and $\frac{n}{1}, n = 1, \ldots, 7$. We obtained 54.8 mAP (i.e., with 7.2 mAP drop) while the computation cost goes up 14 times higher than our proposed scheme. Third, we replace co-occurrence prior filtering with a fixed overlap criterion. That is, the prior map is unique distribution. The final result is 57.3 mAP (i.e., with 4.7 mAP drop).

Comparison with State-of-the-Art Method. We also compare our results with other state-of-the-art methods on VOC 2012 test set and Table 4 tabulates the results. Our performance is comparable to other state-of-the-arts, and close to the current best NUS_NIN_c2000 which uses 2000 categories of ImageNet for pre-training while we only use the ILSVRC data [3]

4.3. Localization Evaluation

In this subsection we analyze the performance of our localization in isolation according to [5] using the Pascal detection criterion. Two localization methods are compared, namely, selective search and edge-box (with objectness score). Since our work is a supervised method without objectness prediction, so we only produce 100 to 300 bounding boxes per image only which helps to reduce false positives.

Table 5 tabulates the number of detected objects with respect to the number of proposals, which shows that we can achieve better localization results under different reporting number requirements. When only 100 boxes are used, we can achieve 82% localization. Our results significantly outperforms the results of selective search and edge boxes which are unsupervised methods. We also found that our square localization performs better than [5] which is a supervised method.

[3] VOC 2012 testing set results using our square localization with VGG model:`http://host.robots.ox.ac.uk:8080/anonymous/8CIIOF.html`. Note that our main focus is localization. Replacing CaffeNet with more sophisticated VGG model [1], our localization can help to achieve state-of-the-art result: our mAP increases to 62.7% which surpasses RCNN VGG by 3.5%.

Figure 5. Sample detection results on the VOC 2012 dataset.

IoU	0.5	0.6	0.7	0.8	0.9
[20] (equal num)	0.51	0.46	0.32	0.24	0.12
[21] (equal num)	0.53	0.47	0.36	0.22	0.06
[20] (1000)	0.89	0.82	0.77	0.54	0.31
[21] (1000)	0.92	0.88	0.79	0.50	0.13
ours	0.91	0.83	0.73	0.48	0.19

Table 6. Recall over IoU comparison with selective search [20] and Edge box [21] in the VOC 2007 dataset. Here, "equal num" means using the same number of proposals used in our method. "1000" is the increased number of proposals used.

Category	5	10	15	20
First 6 layers	0.713	0.711	0.715	0.714
7^{th} layer	0.001	0.001	0.002	0.002
Co-P filtering	0.024	0.024	0.025	0.025
whole-object CNN	0.110	0.109	0.110	0.110
Total	0.872	0.869	0.877	0.876

Table 7. Time performance of individual stage of our square localization under different number of object categories. The reported time performance is the average time running on the VOC 2007 testing dataset using a K40 GPU. Here "whole-object CNN" refers to testing the whole object image using trained object CNN.

Table 6 reports the recall over overlap. From the table, our method significantly outperforms [21, 20] at all IoUs given the same number of proposals.

4.4. Time Complexity and Scalability

The main computation for the sliding CNN occurs in the first 6 layers inference. The inference in the last layer and object co-presence filtering are inexpensive when compared with the main computation. Table 7 summarizes the time cost of each part in VOC 2007 testing data. We found that the running time of the overall system is nearly constant with respect to the number of object categories, since the main computation time of first 6 layer remains relatively constant. Only the last layer is related to the number of categories, and its computation cost is low. For the co-presence prior, its time performance depends on the number of response maps, i.e., the number object categories present in an image, which is scene dependent. Recall that weak response maps are discarded and typically only a small number of object categories are present in an image.

We also empirically found that the number of selected scales is $O(\log(\min\{H, W\}))$. We selected 2000 images from the VOC 2007 dataset and resized them into different scales. Figure 6 plots the mean number of selected scales which grows logarithmically. This is because as the size increases the context information does not significantly increase. To corroborate this, we enumerate all the logarithmic image sizes, and compute the correlation coefficient with their output selected number of scales. We found that the correlation coefficient is 0.9442, that is, the selected number of scales is linear with the logarithmic image size, or in other words, the running time complexity is $O(\log(\min\{H, W\}))$.

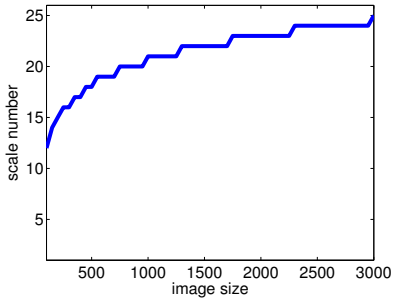

Figure 6. Number of selected scales grows logarithmically as the image size increases.

5. Conclusion and Future Work

The key technical contribution of our paper is square object localization which relieves the exhaustive sliding windows from testing all windows of different combinations of aspect ratios. Further reduction in the number of scale testing is achieved by our new binary search strategy. Square object localization is category scalable. We performed extensive experimental evaluation on two datasets, focusing our evaluation on the last part to demonstrate how our localization method can effectively improve the final detection results.

A. Clutter Suppression

Different categories have different co-presence priors. In the following we describe the co-presence prior for one category where extension to all categories is straightforward. Let $\mathbf{M}$ and $\mathbf{R}$ be respectively the response map and the de-cluttered map, and let $\mathbf{M}(\mathbf{x})$ and $\mathbf{R}(\mathbf{x})$ be respec-

tively the intensity of $\mathbf{M}$ and $\mathbf{R}$ at pixel $\mathbf{x}$. Given a pixel $\mathbf{x}$, we compare its neighborhood region pixels at scale s $\Omega(\mathbf{x}, s)$. We concatenate high responses in $\mathbf{R}(\mathbf{x})$ to finally produce a rectangle object region, since most object image windows are rectangles. We implement the concatenation scheme along the horizontal and vertical direction respectively. Here, we only discuss the vertical case, that is, producing bounding boxes $H > W$. The horizontal case is similar.

Design of Clutter Suppression Factor. We suppress the response at $\mathbf{x}$ if the following two conditions hold: (1) pixels $\mathbf{y} \in \Omega(\mathbf{x}, s)$ have a higher response (an object is likely at $\mathbf{y}$); (2) the likelihood that if $\mathbf{y}$ is an indicator pixel, then $\mathbf{x}$ cannot be an indicator pixel for another object is very high, according to the statistics of the training data. We denote this likelihood as $\gamma(\mathbf{x}|\mathbf{y}, s)$ which ranges in $[0, 1]$. If pixel $\mathbf{y}$ does not have an indicator pixel of another object in its neighborhood, $\gamma(\mathbf{x}|\mathbf{y}, s) = 1$.

Mathematically, if there exists a pixel $\mathbf{y}$, $\gamma(\mathbf{x}|\mathbf{y}, s)\mathbf{M}(\mathbf{y})$ (that is, both conditions hold) is larger than $\mathbf{M}(\mathbf{x})$, we will suppress the response at $\mathbf{x}$. Therefore, we design the clutter suppression factor $h(\mathbf{M}, \mathbf{x})$ as

$$\max\left(\tanh(\beta[\mathbf{M}(\mathbf{x}) - \max_{\mathbf{y}\in\Omega(\mathbf{x},s)}\{\gamma(\mathbf{x}|\mathbf{y}, s)\mathbf{M}(\mathbf{y})\}]), 0\right) \quad (2)$$

where β is a non-linearity strength parameter. We will discuss how to set β shortly. If $\mathbf{M}(\mathbf{x})$ is larger than $\max_{\mathbf{y}\in\Omega(\mathbf{x},s)}\{\gamma(\mathbf{x}|\mathbf{y}, s)\mathbf{M}(\mathbf{y})\}$, $h(\mathbf{M}, \mathbf{x}, \beta)$ is close to 1, and thus we do not suppress the intensity at $\mathbf{x}$.

Object Co-presence Prior. We now detail the design of $\gamma(\mathbf{x}|\mathbf{y}, s)$ by conducting statistical analysis on a large volume of data using the ILSVRC 2014 dataset. First, we implement the domain $\mathbf{y} \in \Omega(\mathbf{x}, s)$, whose elements $(\mathbf{x}, \mathbf{y})$ are indicator pixels of the same category at the similar scale (relative different smaller than 0.05). Let $\mathbf{v}(\mathbf{x}, \mathbf{y}, s) = \frac{\mathbf{x}-\mathbf{y}}{s}$ be the scale normalized difference vector of two indicator pixels at locations $\mathbf{x}$ and $\mathbf{y}$. We consider pixel pairs $(\mathbf{x}, \mathbf{y})$ where both components of $\mathbf{v}$ are within $[-1, 1]$. That is, the two square regions corresponding to the two indicator pixels are overlapping. Let $\{\mathbf{v}_1, \ldots, \mathbf{v}_N\} \in [-1, 1] \times [-1, 1]$ be the set of $\mathbf{v}$'s for the given category.

We uniformly partition the space $[-1, 1] \times [-1, 1]$ into 32×32 sub-regions, thus producing 1024 bins. Given a difference vector $\mathbf{v}$ we cast it into its corresponding bin indexed using the function $bin[\mathbf{v}]$. Therefore, we can compute the distribution of difference vectors in bin k as $P(k) = \frac{1}{N}\sum_{i=1}^{N} U(bin[\mathbf{v}_i] = k)$ where $U(bin[\mathbf{v}_i] = k)$ outputs 1 if $\mathbf{v}_i$ falls in bin k; otherwise it returns 0. Now, we design our object co-presence prior is:

$$\gamma(\mathbf{x}|\mathbf{y}, s) = \exp\left(-\mu P(bin[\frac{\mathbf{x}-\mathbf{y}}{s}])\right) u(\mathbf{x}|\mathbf{y}, s/2) \quad (3)$$

where μ is a contrast non-linear parameter, we will discuss how to determine in follows. If pixel $\mathbf{y}$ never sees an indicator pixel of another object in its neighborhood, we set $\gamma(\mathbf{x}|\mathbf{y}, s) = 1$. $u(\mathbf{x}|\mathbf{y}, s/2)$ is an indication function that outputs 1 if pixels $\mathbf{y}$ and $\mathbf{x}$ are in the same vertical line and their distance is smaller than $s/2$; otherwise returns 0. Thanks to this indication function, we do not suppress the vertical neighborhoods for better vertical concatenation.

Object Co-presence Filtering. Now, we consider the filtered response map

$$\mathbf{R}(\mathbf{x}) \approx \mathbf{M}(\mathbf{x})h(\mathbf{M}, \mathbf{x}), \quad (4)$$

and concatenate high responses in $\mathbf{R}(\mathbf{x})$ to finally produce a rectangle object region, since most object image windows are rectangles. We implement the concatenation scheme along the vertical direction. We define a concatenation prior: $W(\mathbf{R}) = \sum_x |\nabla_v \mathbf{R}(\mathbf{x})|^2$ where ∇_v is the first order gradient operator along the vertical direction. By minimizing this equation zero vertical gradient is encouraged, that is, two vertical neighborhood pixels should have the same intensity. So, we have the following optimization function:

$$\min_{\mathbf{R}}\left(\sum_{\mathbf{x}}(\mathbf{R}(\mathbf{x}) - \mathbf{M}(\mathbf{x})h(\mathbf{M}, \mathbf{x}, \beta))^2 + \lambda W(\mathbf{R})\right) \quad (5)$$

where the first term serves to satisfy Eq. 4, the second term encourages vertical neighbors to have similar intensity, and λ is a balance parameter, we will discuss how to determine in follows. We rewrite Eq. 5 into its matrix form $\min_{\mathbf{R}} \|\mathbf{R} - G(\mathbf{M})\|_2^2 + \lambda\|\mathbf{D}_v\mathbf{R}\|_2^2\}$ where pixel $\mathbf{x}$ in $G(\mathbf{M})$ is $\mathbf{M}(\mathbf{x})h(\mathbf{M}, \mathbf{x})$ and $\mathbf{D}_v$ is a vertical gradient matrix with $\|\mathbf{D}_v\mathbf{R}\|_2^2 = W(\mathbf{R})$. This is a quadratic optimization problem which can be solved by seeking the zero-root on the differentiated objective function $\mathbf{R} - G(\mathbf{M}) + \lambda\mathbf{D}_v^T\mathbf{D}_v\mathbf{R} = 0$. Finally, we solve the above equation in the frequency domain using Fast Fourier Transform,

$$\mathbf{R} = \mathcal{IFFT}\left(\frac{\mathcal{FFT}[G(\mathbf{M})]}{\mathbf{1} + \lambda\mathcal{FFT}[\mathbf{D}_v]^* \cdot \mathcal{FFT}[\mathbf{D}_v]}\right) \quad (6)$$

where $\mathcal{IFFT}$ and $\mathcal{FFT}$ are Fast Fourier Transform and Inverse Fast Fourier Transform, $\mathbf{1}$ is a matrix whose elements are all one, and $\cdot$ is the pixel-wise multiplication operator. Here, $\mathcal{FFT}[\mathbf{D}_v]$ is the vertical gradient matrix. Thus, given $\mathbf{M}$ we can produce a de-cluttered map by filtering $\mathbf{M}$ using Eq. 6 which incorporates the object co-presence prior. An example is shown in Figure 3 (d). We pick up the pixels whose response after the above filtering is larger than 0.2, and then concatenate them vertically (or horizontally) to produce the final object bounding box.

Parameters. We have two non-linear parameters β and μ. They are determined by exhaustive search to achieve the highest mAP (mean average precision) in the validation data set, which are fixed in the testing phase.

References

[1] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *BMVC*, 2014.

[2] M.-M. Cheng, Z. Zhang, W.-Y. Lin, and P. Torr. Bing: Binarized normed gradients for objectness estimation at 300fps. In *CVPR*, 2014.

[3] T. Dean, M. A. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, and J. Yagnik. Fast, accurate detection of 100,000 object classes on a single machine. In *CVPR*, 2013.

[4] C. Desai, D. Ramanan, and C. C. Fowlkes. Discriminative models for multi-class object layout. *IJCV*, 95(1):1–12, 2011.

[5] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov. Scalable object detection using deep neural networks. In *CVPR*, 2014.

[6] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, 2010.

[7] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008.

[8] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *PAMI*, 32(9):1627–1645, 2010.

[9] M. A. Fischler and R. A. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, 22(1):67–92, 1973.

[10] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[11] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *ECCV*, 2014.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, 2014.

[13] F. Iandola, M. Moskewicz, S. Karayev, R. Girshick, T. Darrell, and K. Keutzer. Densenet: Implementing efficient convnet descriptor pyramids. *arXiv preprint arXiv:1404.1869*, 2014.

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

[15] C. H. Lampert, M. B. Blaschko, and T. Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *CVPR*, 2008.

[16] M. Lin, Q. Chen, and S. Yan. Network in network. In *ICLR*, 2014.

[17] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2014.

[18] 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.

[19] Z. Shen and X. Xue. Do more dropouts in pool5 feature maps for better object detection. *arXiv preprint arXiv:1409.6911*, 2014.

[20] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *IJCV*, 104(2):154–171, 2013.

[21] L. Zitnick and P. Dollar. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014.