Immediate, scalable object category detection

Yusuf Aytar Andrew Zisserman
Visual Geometry Group, Department of Engineering Science,
University of Oxford

Abstract

The objective of this work is object category detection in large scale image datasets in the manner of Video Google – an object category is specified by a HOG classifier template, and retrieval is immediate at run time.

We make the following three contributions: (i) a new image representation based on mid-level discriminative patches, that is designed to be suited to immediate object category detection and inverted file indexing; (ii) a sparse representation of a HOG classifier using a set of mid-level discriminative classifier patches; and (iii) a fast method for spatial reranking images on their detections.

We evaluate the detection method on the standard PASCAL VOC 2007 dataset, together with a 100K image subset of ImageNet, and demonstrate near state of the art detection performance at low ranks whilst maintaining immediate retrieval speeds. Applications are also demonstrated using an exemplar-SVM for pose matched retrieval.

1. Introduction

Over the last decade there has been considerable progress in immediate large scale object instance retrieval, where the goal is to instantly retrieve images containing a specific object in a large scale image dataset, given a query image of that object [12, 14, 22, 27, 33]. These methods are now appearing in web and mobile applications such as Google Goggles, Kooaba, and Amazon's SnapTell. Similarly, large scale image classification, where the goal is to retrieve images containing an object category, has also seen improvements in both accuracy [6] and descriptors [3, 26]. With the advent of more efficient descriptor compression methods such as Product Quantization [13] or binary codes [29, 35, 37] large scale image search can proceed quickly in memory without requiring (slow) disk access, for example by using efficient approximate nearest neighbor matching or binary operations, respectively.

However, *object category detection*, where the goal is to determine the location of any instances of a category present in an image, has not seen a similar development of immediate large scale retrieval. The complexity of detecting an object category is still linear in the number of images in the dataset. Despite methods of greatly speeding up sliding window search (over all possible scales and positions) on a per image basis [5, 7, 8, 9, 16, 25, 30, 34, 36], the cost of applying these methods at large scale is still extremely high. The objective of this paper is to fill this gap.

To this end, given a HOG classifier template, we investigate an approach that can *immediately* retrieve detections throughout a large scale dataset. Note, the goal is detection rather than image level classification and consequently we operate in a very large space of subwindow candidates. The key idea is to develop a sparse representation of the HOG template in terms of mid-level discriminative patches, that is suitable for inverted file retrieval. This enables the scalable fast retrieval, and precision is improved by re-ranking a short list. We are agnostic about the source of the classifier template; it could be the component of a linear SVM based object category detection [8], or from an exemplar SVM (E-SVM) [19, 31], with the latter more suited to detecting object categories in a similar pose. We can also benefit from recent fast methods of training such classifier templates, such as LDA (Linear Discriminant Analysis) models [10].

A natural question to ask is why another representation is needed when there already exists sparse detectors (e.g. affine-Harris [11, 21]) and descriptors (e.g. SIFT) that are perfectly adequate for large scale immediate object instance retrieval (at least if the objects are lightly textured) using bag-of-visual-word representations [12, 22, 27, 33]. There are two responses: first, this representation does not generalize well over the deformation and intra-class variation required for category (rather than instance) recognition. We demonstrate the validity of this response empirically in section 5.2. Second, and in contrast, the recent development of mid-level discriminative primitives has shown that they *are* suitable for object (and scene) categorization, so they are clear candidates for our task. In this we are inspired by the mid-level sparselet features of [9, 34] and the mid-level discriminative patches used for scene category recognition [2, 15, 23, 24, 32].

2. Architecture Overview

We first briefly overview the architecture of the approach, and then describe the stages in more detail in the following sections. The method follows the framework established for the bag-of-visual-words (BoW) instance retrieval systems [12, 22, 27, 33] with an offline processing stage (that can be computationally expensive), followed by an online search stage using an inverted index and posting-lists that is fast at run time and scalable. The online search involves two steps: an initial ranking of the images using image level vectors alone; followed by reranking of a short list using spatial information.

In our case the offline stage consists of the following steps: (i) build a vocabulary of mid-level discriminative classifier patches (CPs). (ii) represent each image in the dataset by the *maximum response* to each of the CPs, i.e. if there are V CPs then the vector representing each image would be of dimension V if only the maximum response is stored; and (iii) compute an inverted index (or equivalently posting lists) for each CP.

At run time, detection for a given HOG template classifier then proceeds in three stages (illustrated in figures 1 and 2):

1. Represent the HOG classifier template: The classifier is approximated by a set of the CPs.

2. Image retrieval: A 'posting list' is obtained for each CP used to represent the template, and those images occurring in the posting lists are ranked based on their aggregated maximum scores.

3. Rerank a short list: The top K images of the ranked list are reranked based on the spatial consistency of the maximum response CPs in that image and/or the original HOG classifier.

It is worth noting at this point that the reason we store the maximum response of each CP (rather than the count as in BoW) for the image representation is that this provides an upper bound on the score of the (reconstructed) HOG classifier on that image – see section 3.2 below.

3. Approach

This section describes the scalable detection system. The two main aspects are the query representation, and the image representation for the dataset to be searched. We describe how to obtain each of these representations and then their joint use for fast detection.

3.1. HOG query representation

The query is specified as a HOG classifier template which can be obtained using various methods (e.g. DPMs, E-SVMs, LDAs, etc.), and it is represented using a 'vocabulary' of CPs. The CPs are essentially parts of classifiers that have earlier been trained in a discriminative manner using a DPM on another dataset. The intuition here is that there are naturally repeating classifier patches that can be shared across objects, such as parts of wheels, legs, corner-like patterns of doors and windows, etc. The method of generating the CP vocabulary is described in the implementation details of section 4.

The query HOG template is approximated by another HOG template constructed using a sparse weighted combination of CPs. This procedure is illustrated in figure 1, and the intermediate query representation is referred to as the *reconstructed template (RT)*. In detail, given the query HOG template w, the reconstructed template w^{rt} is obtained from a set of CPs d_j by minimizing a sparse learning objective function:

$$\min_{\alpha_j} ||w - \sum_j \alpha_j u_j||^2 + \gamma||\alpha||_1 \quad st: \alpha_j \geq 0 \tag{1}$$

$$w^{rt} = \sum_j \alpha_j u_j \tag{2}$$

where each u_j consists of a CP d_j located at a position x_j on a w sized zero filled template (as shown in figure 1). The location x_j is determined as the best fit of d_j to the original template w (as measured by sliding d_j over w and selecting the location with maximum normalized dot product). The scalar α_j is the combination weight of the relocated classifier patch d_j, and γ controls the sparsity of the reconstruction. Large values of γ encourage very sparse reconstructions, i.e. a small number of CPs are used. Note that this method differs from the traditional sparse coding reconstruction of image patches by a dictionary of atoms (e.g. [18]) by the necessity to choose both the dictionary element *and* its specific spatial location on the w sized template.

Instead of using all possible u_j's in learning, optimization is performed over a smaller subset that is determined by a similarity (i.e. normalized dot product score) thresholding operation. This simplification allows much faster optimization, and has a negligible effect on the final result since the l_1 norm on α, which induces sparsity, is already forcing most of the unrelated α_j to zero. Note we do not require that the chosen CPs are non-overlapping in the reconstruction w^{rt}. The optimization is performed using SPAMS toolbox [18].

Discussion. There are many other HOG template reconstruction methods that could be explored – for example: dense, or sparse but non-overlapping, or discriminative learning; and others have considered some of these. In particular the sparselet representation [34] is learnt as a vocabulary for reconstruction, whereas [9] learns both the vocabulary and the classifiers simultaneously in a discriminative framework. In future work each of these possibilities can be explored but, as will be seen in the experiments, the simple method we have employed is sufficient here.

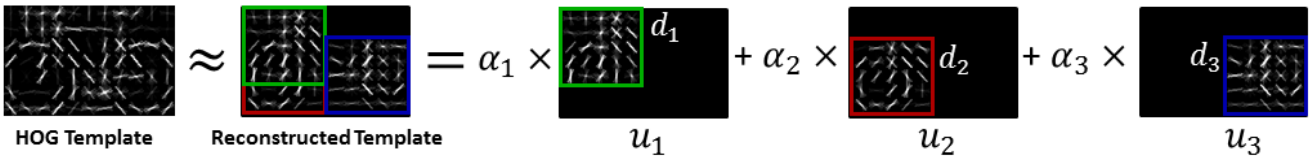

Figure 1. **Query representation.** The query HOG template w is approximated with the reconstructed template $w^{rt} = \sum_j \alpha_j u_j$ which is used as the intermediate query representation. Note, the representation u_j records both the classifier patch (CP) d_j used and also its position relative to the original HOG template.

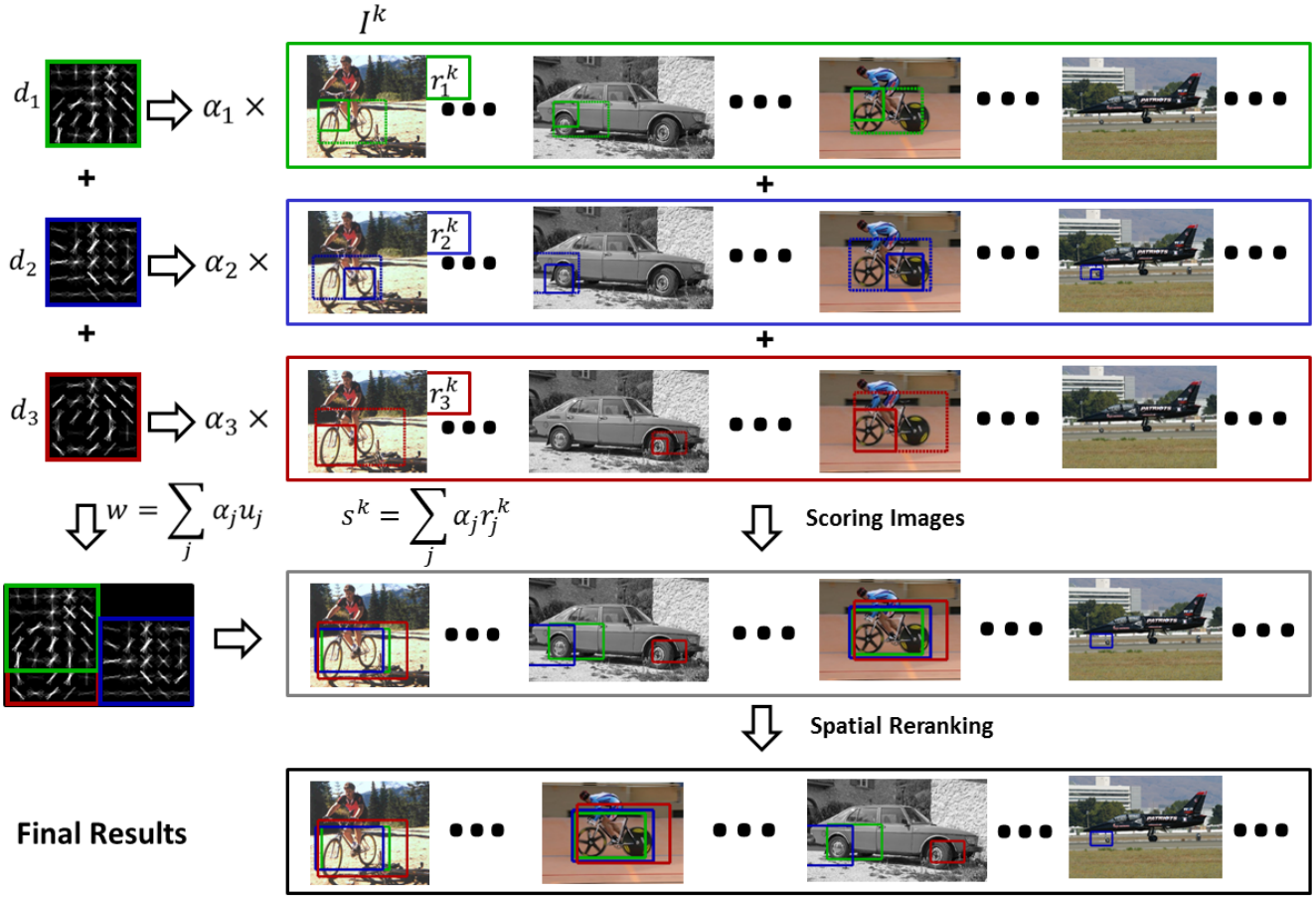

Figure 2. **Detection using a template composed of CPs.** Each CP retrieves a posting list of images, and a score for each image is computed from the weighted maximum response representation. Images are initially ranked on this score, and the top K then reranked using the spatial information associated with each maximum response.

3.2. Image Representation

This section describes the image representation that is used for fast detection. The aim is to define a representation that is suitable for obtaining a good upper bound on the score of w^{rt} on a given image. Each image in the dataset is represented as a V-dimensional vector, r, which contains the maximum scores of each CP evaluated on the image over all possible locations and scales. The location and the scale of the CP detection is also stored so that we can obtain a candidate bounding box as suggested by the specific CP. These candidates will be used later for the spatial reranking.

In detail, let $\Psi(c,I)$ be the maximum response of any classifier template c on the image I over all possible locations and scales. Then the image representation vector r is obtained with components $r_j = \Psi(d_j,I)\ \forall$ CPs d_j. The vector r is V dimensional, where V is the size of the vocabulary of CPs.

Upper bound. For ranking, an image is scored as $\alpha^\top r$, where α is the sparse V-dimensional vector of coefficients of the reconstructed template w^{rt}. This is an *upper bound* on the score of w^{rt} on the image, since

$$\alpha^\top r = \sum_j \alpha_j \Psi(d_j,I) \geq \Psi(w^{rt},I) \tag{3}$$

where the $\geq$ arises because the maximum response of each CP can occur at any position and scale, (i.e. there is no rea-

son why the CPs should 'fire' at the relative positions assigned to them in the w^{rt} template), and the representation r ignores the spatial locations of the CPs detected on the image.

Detection information. The vector r, for each image in the dataset, is sufficient to obtain a ranked list of images, but additional information is necessary in order to evaluate the spatial reranking (described below). First, the location and the scale of the CP detection is also stored so that we can obtain a candidate bounding box (BB) as suggested by the specific CP. Second, instead of storing only the maximum response of each CP, the top R responses (and associated positions) are stored. As will be seen, this information will enable multiple instances of the category to be localized in each image and also improves the localization of each.

3.3. Fast Detection

This section describes how to perform immediate detection using the image and the query representations described in the previous two sections. Given a HOG template as the query the fast detection is performed in three stages:
Query representation. As discussed above in section 3.1 the query HOG template is represented with the reconstructed template $w^{rt} = \sum_j \alpha_j u_j$.
Shortlist retrieval. At this stage a shortlist of images, potentially matching well with the query, are retrieved. The score of the image I^k for the query w^{rt} is computed as $s^k = \alpha^\top r^k$ where the (sparse) reconstruction weights α are obtained from the query representation. As noted in section 3.2, this score is indeed an approximation of the maximum response of w^{rt} evaluated on the image I^k. Then the shortlist is compiled as the selection of top K images sorted by the score s^k. This stage is carried out very efficiently for large scale collections using an inverted file index.
Spatial reranking. After obtaining the shortlist of candidate images, a reranking process is performed which aggregates the scores of candidate bounding boxes (BB) coming from each CP (as determined from its position x_j in the reconstructed template). Each firing of a CP d_j in the image proposes a BB in a manner similar to [4, 20].

Given this set of BBs, two spatial reranking methods are employed. The first, ReRanking (RR) by *BB aggregation*, is similar to the Hough transform method used in the Implicit Shape Model (ISM) of [17] and to the standard method of greedy non-maximal suppression (NMS) used in object detection. More precisely, if the overlap between two candidate BBs is more than 0.5, then they are aggregated by taking the maximally scoring BB position and summing the scores coming from the two BBs. Starting with the maximum scoring BB, this procedure is repeated until there are no remaining overlapping BBs above the 0.5 overlap threshold. Note that if all the CPs fire with the same relative positioning as they have in the w^{rt}, in other words all the candidate BBs overlap 100%, then this procedure will give the score of the w^{rt} exactly together with the original bounding box. Also note that specifying the overlap threshold as 0.5, rather than a larger value, allows a certain degree of spatial deformation for the detected instance. The second reranking method, *HOG scoring*, is to use the BB to score a detection using the original HOG classifier template w. Note, this operation is very fast (compared to sliding w over the entire image at multiple scales) since the BB specifies the location and scale, so only a single scalar product is required, and this is implemented very efficiently using product quantization (PQ) with a look up table [13, 30, 36]. Note, using PQ for HOG scoring has the additional benefit of a tremendously decreased (128×) memory footprint for the dataset HOG representation.

4. Implementation Details

4.1. Construction of Classifier Patch Vocabulary

The vocabulary is composed of CPs obtained from previously trained classifiers. DPM models are trained using 3 components (6 in total with mirrors) without parts over 1000 classes of the ImageNet 2012 challenge [6] using the provided images. The training set (∼400 positive images per class) is used for training and the quality of the detectors are evaluated using a small validation set of 50 positive images per class and a 1000 randomly selected negative images. The models achieving over 30% AP (329 models) are selected as the source for the CPs – we restrict to these as we wish to use a vocabulary of well trained discriminative patches.

To build the vocabulary, all patches with sizes of 3×3, 4×4, 5×5, 6×6, and 7×7 are extracted from the components of the trained DPMs. For each size, a k-means clustering is performed with $k = 10$K centres. Finally from each cluster the most central patch is selected and is used for the vocabulary (i.e. we do not use the mean of the cluster as this tends to average out details). The outcome is five 10K vocabularies, one for each patch size.

Since the DPMs are trained discriminatively using a large number of training samples, the vocabulary items have a strong 'sense' of foreground/background discrimination, and since they are extracted from semantically meaningful objects, they possess certain spatial semantic properties as well.

4.2. The inverted index and offline processing

For each image in the test dataset, the maximum response of each CP together with the corresponding locations and scale is computed and stored (non-maximum suppression is used while extracting the top R responses for each CP). If there are V CPs and N images, and $R = 1$, then this is equivalent to a $V \times N$ matrix M where each entry m_{ji}

Methods	NC	NR	NW	PR@10	PR@50	PR@100	AP	Time
Original w SW	-	-	0	100.0	98.0	77.0	31.1	$\sim$ 1.3 hrs
Reconstructed w^{rt} SW (6CPs)	6	-	0	90.0	88.0	63.0	19.7	$\sim$ 1.3 hrs
Fast Search (FS)	6	1	0	70.0	58.0	39.0	9.1	0.2 s
FS + Reranking (RR)	6	1	0	80.0	62.0	38.0	9.4	0.3 s
FS + HOG-SC	6	1	1K	90.0	74.0	45.0	12.3	1.2 s
FS + RR + HOG-SC	6	1	1K	90.0	80.0	49.0	13.3	1.3 s
FS + HOG-SC + PQ	6	1	1K	90.0	74.0	42.0	11.4	0.8 s
FS + RR + HOG-SC + PQ	6	1	1K	90.0	74.0	46.0	12.3	0.9 s
FS 6CP + ALL-BB + HOG-SC	6	1	6K	90.0	84.0	61.0	18.6	5.5 s
FS 9CP + ALL-BB + HOG-SC	9	1	9K	100.0	88.0	63.0	19.4	7.8 s
FS 17CP + ALL-BB + HOG-SC	17	1	17K	100.0	86.0	63.0	19.7	12.4 s
FS 6CP + TOP-3 + ALL-BB + HOG-SC	6	3	18K	90.0	86.0	62.0	21.4	14.6 s
FS 9CP + TOP-3 + ALL-BB + HOG-SC	9	3	27K	100.0	88.0	66.0	21.9	23.4 s
FS 6CP + TOP-5 + ALL-BB + HOG-SC	6	5	30K	90.0	86.0	60.0	21.8	29.3 s
FS 9CP + TOP-5 + ALL-BB + HOG-SC	9	5	45K	100.0	86.0	65.0	22.7	36.2 s

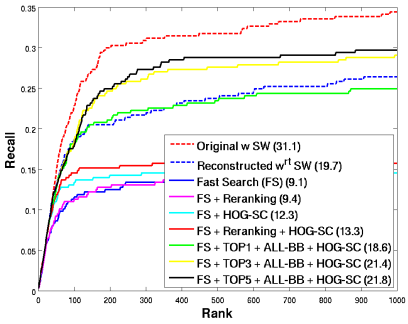

Table 1. **Comparison of sliding window (SW) and fast search (FS) detection performance on VOC07 test for a single template**. The template is a side-facing bicycle component from a DPM. NC: number of CPs used in template reconstruction; NR: number of top responses stored for each CP in each image; NW: number of re-evaluated windows using HOG scoring – this is a good indication of the overall cost. **Compared methods**: (i) exhaustive sliding window search with the original HOG template (Original w SW); (ii) exhaustive sliding window search with the reconstructed template (Reconstructed w^{rt} SW); (iii) fast search (FS) through the use of inverted index files; and (iv) fast search with a number of reranking methods as described in section 3.3: BB aggregation reranking (RR) which aggregates the scores coming from different CPs; and HOG scoring (HOG-SC). "+ PQ" is used for denoting the use of product quantisation during HOG scoring. In "FS + HOG-SC" and "FS + RR + HOG-SC" the HOG scoring is only applied to the best BB in the image; whilst for "ALL-BB + HOG-SC", HOG scoring is applied to all the candidate BBs suggested by the CPs. Note, the number of rescored windows (NW) using the original HOG template is NC $\times$ NR $\times$ 1000 for "ALL-BB'". Timings are for a single core. Recall-rank curves are displayed together with the AP scores. Note that (i) the performance approaches that of the original HOG template, and (ii) the tremendous timing difference between exhaustive and fast search. A 5×5 patch size is used for this experiment.

is the maximum response of the j^{th} CP on the i^{th} image. Since the approximated template is composed from a small number of CPs, obtaining the scores for all images involves computing a weighted sum of a small number of rows of M. In practice the matrix is sparse as only scores above a threshold need be stored.

For the large scale regime, or for $R > 1$, it is more efficient to store an inverted index, where each entry is a posting list for the CP containing: the image identifier, responses to the particular CP in that image, and the location and scale of the associated templates.

The offline process, using our brute-force implementation, takes about a minute per image to compute the responses of 10K dictionary items at all scales and positions, however this can be done very efficiently ($\sim$ 5 sec) using [5, 34].

5. Experiments

The objective of these experiments is to investigate the performance of the retrieval architecture first as the key parameters are varied, and then as the number of images are scaled up. Performance here includes timings (as we are interested in immediate retrieval), but principally in measuring how rankings differ from using the original HOG classifier (or set of HOG classifiers in the case of a DPM). This difference is termed the 'mAP gap' in [28].

The experiments are conducted on two image datasets: (1) the test set of the PASCAL Visual Object Classes 2007 challenge, which will be referred to as VOC07; and (2) the validation sets of the ImageNet Large Scale Visual Recognition Challenge 2011 and 2012 [6], referred to as ImageNet. The number of test images in VOC07 is $\sim$5K, and in ImageNet $\sim$100K. The method is assessed on VOC07 alone, and then on VOC07+ImageNet for large scale detection, where the 100K images of ImageNet act essentially as distractors.

We evaluate the methods based on their object category detection performance in terms of average precision (AP), and precision of the top 10, 50 and 100 ranked detections. Here we follow the VOC practice and software (for example using an overlap threshold of 0.5 between prediction and ground truth for a true positive detection).

5.1. Evaluation on a single component

Initially we evaluate the framework with a single side-facing bicycle component which is obtained from a three component DPM model trained using VOC07 data. The test performance is assessed using all bicycle instances (i.e. not only restricted to side-facing bicycles). We compare the precision and recall as the key parameters of the fast search system are varied, namely: NC, the number of CPs used in template reconstruction (achieved by varying γ); and NR, the number R of top detection responses stored for each CP for obtaining candidate BBs in each image. The reconstruction is performed using a 5×5 vocabulary and the top $K = 1000$ images are reranked in the shortlisting stage. The results are presented in table 1 which also includes plots of recall against rank.

The detection results show that the maximum response representation with spatial reranking does indeed enable immediate detection, with precision at low rank (up to 50) being comparable to exhaustive search with the original detector. The timings are 1.3 seconds compared to 1.3 hours. With fewer CPs the retrieval is faster, however increasing the number of used CPs increases the performance. Re-

PR@10 Results	aero.	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	m.bike	person	plant	sheep	sofa	train	tv	mean
Original SW	100	100	30	70	100	100	100	40	100	80	60	20	100	100	100	70	80	100	90	100	**82.0**
Reconstructed SW	20	100	60	30	40	70	100	0	30	40	10	10	60	90	70	0	20	0	80	70	**45.0**
FS + HOG-SC	70	100	10	20	40	90	90	0	20	90	0	0	90	70	70	10	20	0	70	60	**45.9**
FS + RR + HOG-SC	70	100	30	30	60	90	90	10	40	70	10	10	90	90	70	0	50	0	90	70	**53.5**
Video Google 10K	40	80	0	0	0	20	60	10	0	10	0	10	100	55	60	0	0	10	50	10	**25.8**
Video Google 200K	80	90	0	10	0	10	60	10	0	10	0	10	90	87	50	0	0	10	70	70	**32.9**
PR@50 Results	aero.	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	m.bike	person	plant	sheep	sofa	train	tv	mean
Original SW	64	98	20	42	72	94	100	22	86	66	28	8	94	86	94	42	44	65	82	86	**64.7**
Reconstructed SW	12	98	30	18	28	32	82	6	16	22	2	8	28	54	72	0	14	0	46	46	**30.7**
FS + HOG-SC	40	94	8	14	24	40	88	0	8	35	4	0	54	32	74	6	6	0	24	27	**28.9**
FS + RR + HOG-SC	42	94	14	20	34	62	94	4	16	44	4	2	67	50	76	6	18	0	52	36	**36.8**
Video Google 10K	20	44	0	0	0	6	72	2	0	2	2	6	60	35	16	0	0	2	10	18	**14.8**
Video Google 200K	28	32	0	2	2	14	64	2	0	6	0	2	62	45	40	0	0	2	18	28	**17.4**
PR@100 Results	aero.	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	m.bike	person	plant	sheep	sofa	train	tv	mean
Original SW	51	95	15	27	44	72	98	15	64	51	23	7	85	74	90	25	32	40	70	67	**52.4**
Reconstructed SW	10	80	18	11	19	23	72	6	13	18	1	10	27	39	64	0	14	0	27	42	**24.8**
FS + HOG-SC	21	77	5	9	16	27	84	0	4	24	4	0	32	22	73	5	3	0	16	18	**22.0**
FS + RR + HOG-SC	22	79	7	12	21	36	88	2	12	31	6	1	44	32	73	6	13	0	34	29	**27.5**
Video Google 10K	14	27	2	0	0	5	55	1	0	1	2	4	36	21	13	0	0	1	7	14	**10.2**
Video Google 200K	16	21	0	1	1	8	47	1	1	3	0	1	44	26	23	0	0	1	11	17	**11.1**

Table 2. **Performances of the complete DPM detector (3 components) on the VOC07 test set**. Note that fast search approaches the performance of the original sliding window, though with significantly smaller search times, and clearly outperforms Video Google.

call can be increased at higher ranks by storing more CP responses and BBs per image, and a very respectable AP of 22% can be reached (compared to the original 31%) but at the increased cost of around 30s for the retrieval. Note though that these timings are evaluated using a single core, and the HOG rescoring operations can easily be paralellized and obtained much faster with a multiple core implementation.

Approaches such as cascade detection [8], branch and bound [16], and product quantization [13, 30, 36] also improves the detection speed per image, but scale linearly with the number of images as opposed to the near constant time complexity of fast search ($\sim$ 1sec).

5.2. Fast detection using a DPM

In the previous section the experiments were conducted on a single template. Here we evaluate a DPM models with multiple components (templates) over all the VOC classes. The DPM models are trained on VOC07 train/val using 3 components (6 with mirrors) but excluding parts. For the fast search implementation, each component of the DPM model is reconstructed and used to retrieve a ranked list of detections; these lists are then aggregated using the weightings and NMS from the DPM model. The performances are reported in table 2 for a CP size of 5×5.

As a related baseline we also adapt a Video Google (VG) [33] like BoW approach to object category detection, using an implementation with sparse affine-covariant detections and SIFT descriptors as described in [1]. In VG category detection all the training BBs are used as queries and the returned list of detections (together with their matching scores) are merged by a simple ranking based on scores (the score is the number of inliers to the estimated affine transformation between the query and target BB). This ranking method is recommended in [1] for merging the returns from multiple queries. VG is performed with two different vocabulary sizes, $10K$ and $200K$. There exist 332 queries per class on average and each query takes about 100ms, hence the timing per class is around $\sim$ 30 seconds. The timings for fast search is around 6 seconds per class since there are 6 components.

PR@10 Results	3x3	4x4	5x5	6x6	7x7
Reconstructed SW	57.5	52.0	45.0	43.0	49.0
FS + HOG-SC	23.5	44.0	45.9	53.3	48.0
FS + RR + HOG-SC	34.5	53.7	53.5	59.9	53.5
PR@50 Results	3x3	4x4	5x5	6x6	7x7
Reconstructed SW	43.2	35.7	30.7	28.8	34.2
FS + HOG-SC	13.1	24.2	28.9	33.8	35.2
FS + RR + HOG-SC	20.9	32.1	36.8	39.7	38.7
PR@100 Results	3x3	4x4	5x5	6x6	7x7
Reconstructed SW	34.1	29.4	24.8	21.7	27.2
FS + HOG-SC	8.7	17.5	22.0	24.5	25.9
FS + RR + HOG-SC	14.6	22.8	27.5	29.9	29.4

Table 3. **DPM detector performance as a function of CP size.** As CP size increases, the performance of fast search gradually increases, reaching a limit at 6×6. For Video Google and Original SW results, please refer to table 2

From the mean (average over class) performance it can be seen that that FS with spatial reranking: (i) outperforms the DPM using reconstructed templates, (ii) approaches the level of the original DPM models (see the performances of bike and car classes in table 2), and (iii) outperforms VG substantially (by a factor of two at rank 100). Interestingly, VG has a similar performance to FS for some classes (e.g. horse and motorbike) but fails completely for several others (e.g. cow).

5.3. Effect of CP size

As in the previous section, we evaluate the FS version of the DPM detector over all 20 classes of VOC07, and report the mean performance in table 3 as the the CP size varies from 3×3 to 7×7. As would be expected, the quality of reconstruction generally decreases with increase in CP size (since smaller CPs can better match the original template), and this is reflected in the decrease in reconstructed template (SW) results. However, as CP size increases, the CPs become more semantically meaningful and discriminative, and this improves the quality of the fast search results (less false positives). Consequently, at around a size of 6×6 FS methods obtains better performance than reconstructed template SW (as noted above, FS with HOG scoring using original HOG template can exceed the performance of the

Figure 3. Comparison of top 10 detections of the side-facing bicycle component over 5K and 105K test sets.

		PR@10	PR@50	PR@100	TIME
VOC07 (5K)	**FS**	70.0	58.0	39.0	123 ms
	FS + RR	80.0	62.0	38.0	238 ms
ImageNet + VOC07	**FS**	50.0	36.0	39.0	121 ms
	FS + RR	60.0	42.0	40.0	244 ms

Table 4. **Detection performance of the side-facing bicycle component evaluated at large-scale (∼105K).** Neither the timings nor the performances are affected much by the additional ∼100K distractors.

reconstructed template SW).

5.4. Large scale detection

In this experiment we extend the VOC07 test set by adding distractors from ImageNet validation sets. In the combined test set we have ∼105K images. Since we don't have reliable ground truth for the ImageNet set, we perform manual evaluation.

We evaluated the side-facing bicycle component on VOC07 alone and on the combined set. The results are shown in table 4 and figure 3. We observe that the quality of the detections at low ranks is not affected by the distractors coming from ImageNet, and the timings are similar even though we move from a 5K to a 105K set, i.e. scaling up to large databases does not affect the speed or accuracy of the system – our original objective.

Note that the drop in performance with the ImageNet distractors at lower ranks is due to tandems (see figure 3) – two of these appear in the top 10, and are not counted as bicycles hence marked as false retrievals. If we count them as bicycles as well, the performance would be much higher.

5.5. Fast Detection using E-SVMs

Finally, we use the fast search for templates obtained from E-SVM models for VOC07, and compare the performance to the original E-SVM template and to Video Google (where for VG the query is the image BB that is used to determine the E-SVM). For the quantitative results, E-SVMs are trained for 50 BBs randomly selected for each of the 20 classes. The truncated and the difficult BBs are not used for training since they are not good representatives of the class. Each E-SVM model is evaluated individually for the category that it belongs to.

Many of the original E-SVMs (*using sliding window search*) do not return any positive images within the top 10 retrievals. Therefore, for a more sound evaluation, we use only the 192 queries where either the original E-SVM or the reconstructed one retrieves at least three instances of the target class within the top 10 using sliding window search. Table 5 reports the mean (class average) performances of the E-SVM queries. Again, the FS with reranking performance at low ranks is quite good – though not at the same level as for the DPM trained templates of table 2. This is because the E-SVM templates cannot be represented so well by the CPs since E-SVMs are not such 'pure' detectors. Again, the fast search methods perform significantly better than Video Google, with similar timings. Figure 4 shows qualitatively the retrieved images for a sample of E-SVMs.

	PR@10	PR@50	PR@100	TIME
Original SW	46.8	24.9	17.0	∼ 1.3 hrs
Reconstructed SW	31.7	21.1	15.8	∼ 1.3 hrs
Fast Search (FS)	15.4	8.7	6.4	0.2 s
FS + RR	15.5	8.7	6.5	0.3 s
FS + RR + HOG-SC	24.5	13.0	9.0	1.1 s
Video Google 10K	8.8	4.5	3.0	0.1 s
Video Google 200K	4.0	0.8	0.4	0.1 s

Table 5. **Mean performance comparison of sliding window (SW) and fast search (FS) methods on VOC07 test set using 192 E-SVMs trained from randomly selected samples from the 20 classes of VOC07 training set.** Note the tremendous timing difference between sliding window methods (which employs exhaustive search) and fast search methods.

6. Summary and discussion

Traditionally object category detection is performed in a sliding window fashion, which is a very costly procedure for large scale datasets. We have presented a fast, scalable, detection framework which obtains near state of the art low rank results immediately. It is now possible, by learning the HOG classifier using an E-SVM or LDA (see figure 4), to go from specifying a visual query in an image, to retrieving object category detections over a large scale dataset in just a matter of seconds on a single core. Furthermore, recall can be increased to quite satisfactory levels at a cost of greater retrieval times, though these costs can be ameliorated by simple multi-core parallelism of the spatial reranking.

Acknowledgements. We are grateful to Relja Arandjelović for providing the Video Google results [1, 33], and for insightful discussions with Derek Hoiem. This work was supported by ERC grant VisRec no. 228180 and EU Project AXES ICT–269980.

References

[1] R. Arandjelović and A. Zisserman. Multiple queries for large scale specific object retrieval. In *Proc. BMVC.*, 2012. 6, 7

[2] Y. Aytar and A. Zisserman. Enhancing exemplar svms using part level transfer regularization. In *Proc. BMVC.*, 2012. 1

[3] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman.

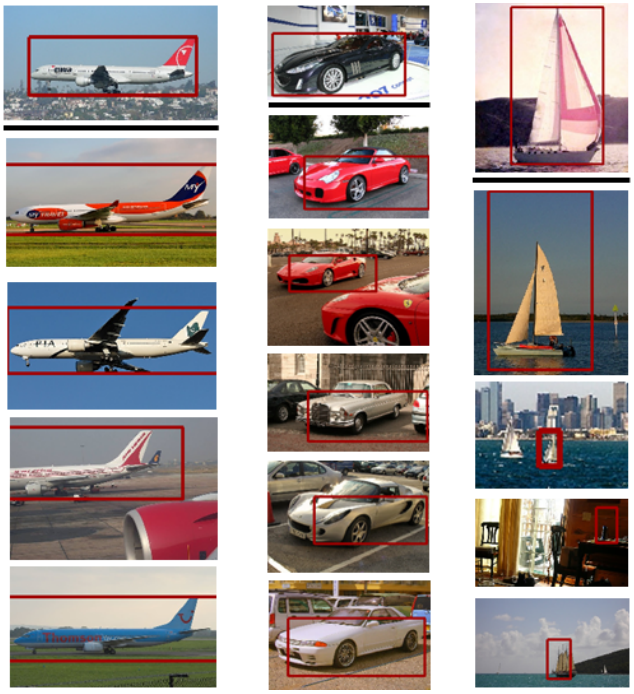

Figure 4. **Visualization of detection results for a few E-SVM queries.** Note the pose and the type similarity between the query and the retrieved results. These results are retrieved in less than a second. The top images in each column is the query image.

The devil is in the details: an evaluation of recent feature encoding methods. In *Proc. BMVC.*, 2011. 1

[4] O. Chum and A. Zisserman. An exemplar model for learning object classes. In *Proc. CVPR*, 2007. 4

[5] T. Dean, M. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, and J. Yagnik. Fast, accurate detection of 100,000 object classes on a single machine. In *Proc. CVPR*, 2013. 1, 5

[6] J. Deng, A. Berg, S. Satheesh, H. Su, and F. F. Khosla, A. Li. Imagenet large scale visual recognition challenge 2012 (ILSVRC2012), 2012. 1, 4, 5

[7] C. Dubout and F. Fleuret. Exact acceleration of linear object detectors. In *Proc. ECCV*, 2012. 1

[8] P. Felzenszwalb, R. Girshick, and D. McAllester. Cascade object detection with deformable part models. In *Proc. CVPR*, 2010. 1, 6

[9] R. Girshick, H. Song, and T. Darrell. Discriminatively activated sparselets. In *Proc. ICML*, 2013. 1, 2

[10] B. Hariharan, J. Malik, and D. Ramanan. Discriminative decorrelation for clustering and classification. In *Proc. ECCV*, 2012. 1

[11] C. G. Harris and M. Stephens. A combined corner and edge detector. In *Proc. Alvey Vision Conf.*, 1988. 1

[12] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *Proc. ECCV*, 2008. 1, 2

[13] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE PAMI*, 2011. 1, 4, 6

[14] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *Proc. CVPR*, 2010. 1

[15] M. Juneja, A. Vedaldi, C. V. Jawahar, and A. Zisserman. Blocks that shout: Distinctive parts for scene classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 1

[16] I. Kokkinos. Rapid deformable object detection using dual-tree branch-and-bound. In *NIPS*, 2011. 1, 6

[17] B. Leibe and B. Schiele. Interleaved object categorization and segmentation. In *BMVC03*, volume 2, 2003. 4

[18] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *Proc. ICCV*, 2009. 2

[19] T. Malisiewicz, A. Gupta, and A. A. Efros. Ensemble of exemplar-SVMs for object detection and beyond. In *Proc. ICCV*, 2011. 1

[20] M. Marszalek and C. Schmid. Spatial weighting for bag-of-features. In *Proc. CVPR*, 2006. 4

[21] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *IJCV*, 2005. 1

[22] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Proc. CVPR*, 2006. 1, 2

[23] M. Pandey and S. Lazebnik. Scene recognition and weakly supervised object localization with deformable part-based models. In *Proc. ICCV*, 2011. 1

[24] S. Parizi, J. Oberlin, and P. Felzenszwalb. Reconfigurable models for scene recognition. In *Proc. CVPR*, 2012. 1

[25] M. Pedersoli, A. Vedaldi, and J. Gonzalez. A coarse-to-fine approach for fast deformable object detection. In *Proc. CVPR*, 2011. 1

[26] F. Perronnin, Y. Liu, J. Sanchez, and H. Poirier. Large-scale image retrieval with compressed fisher vectors. In *Proc. CVPR*, 2010. 1

[27] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proc. CVPR*, 2007. 1, 2

[28] J. Philbin, M. Isard, J. Sivic, and A. Zisserman. Descriptor learning for efficient retrieval. In *Proc. ECCV*, 2010. 5

[29] M. Raginsky and S. Lazebnik. Locality sensitive binary codes from shift-invariant kernels. In *NIPS*, 2009. 1

[30] M. Sadeghi and D. Forsyth. Fast template evaluation with vector quantization. In *NIPS*, 2013. 1, 4, 6

[31] A. Shrivastava, T. Malisiewicz, A. Gupta, and A. A. Efros. Data-driven visual similarity for cross-domain image matching. *ACM Trans. Graph.*, 2011. 1

[32] S. Singh, A. Gupta, and A. Efros. Unsupervised discovery of mid-level discriminative patches. In *Proc. ECCV*, 2012. 1

[33] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proc. ICCV*, 2003. 1, 2, 6, 7

[34] H. Song, S. Zickler, T. Althoff, R. Girshick, M. Fritz, C. Geyer, P. Felzenszwalb, and T. Darrell. Sparselet models for efficient multiclass object detection. In *Proc. ECCV*, 2012. 1, 2, 5

[35] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: a large dataset for non-parametric object and scene recognition. *IEEE PAMI*, 2008. 1

[36] A. Vedaldi and A. Zisserman. Sparse kernel approximations for efficient classification and detection. In *Proc. CVPR*, 2012. 1, 4, 6

[37] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *NIPS*, 2008. 1