Visual vocabulary with a semantic twist

Relja Arandjelović and Andrew Zisserman

Department of Engineering Science, University of Oxford

Abstract. Successful large scale object instance retrieval systems are typically based on accurate matching of local descriptors, such as SIFT. However, these local descriptors are often not sufficiently distinctive to prevent false correspondences, as they only consider the gradient appearance of the local patch, without being able to "see the big picture".
We describe a method, SemanticSIFT, which takes account of local image semantic content (such as grass and sky) in matching, and thereby eliminates many false matches. We show that this enhanced descriptor can be employed in standard large scale inverted file systems with the following benefits: improved precision (as false retrievals are suppressed); an almost two-fold speedup in retrieval speed (as posting lists are shorter on average); and, depending on the target application, a 20% decrease in memory requirements (since unrequired 'semantic' words can be removed). Furthermore, we also introduce a fast, and near state of the art, semantic segmentation algorithm.
Quantitative and qualitative results on standard benchmark datasets (Oxford Buildings 5k and 105k) demonstrate the effectiveness of our approach.

1 Introduction

Large scale specific object retrieval is a well studied topic due to its usefulness in a range of applications, amongst others: geolocalization [1, 2, 3, 4], personal photo search and automatic tagging [5, 6], product and logo recognition [7, 8, 9, 10], video search [11] and 3-D reconstruction [12].

All successful systems rely on matching local image patches using their appearance, usually via matching of local descriptors such as SIFT [13]. The bag-of-visual-words (BoW) framework [11] accomplishes this by quantizing descriptors into visual words and performing efficient retrieval for large scale datasets by using an inverted index. Many improvements to this seminal work have been made by increasing the quality of descriptor matches. These include: higher precision by using large visual vocabularies [7, 14]; storing compact versions of descriptors inside the inverted index [15, 16, 17]; spatial reranking and query expansion [18, 19, 20, 21, 22]; and learning better descriptors (than SIFT) [23, 24].

However, all these methods are built around the core system based on matching local patches/descriptors. In this paper we aim at improving the core system and going beyond blind matching of local patches by also taking account of the semantic content of the query and database images. This addresses one of the

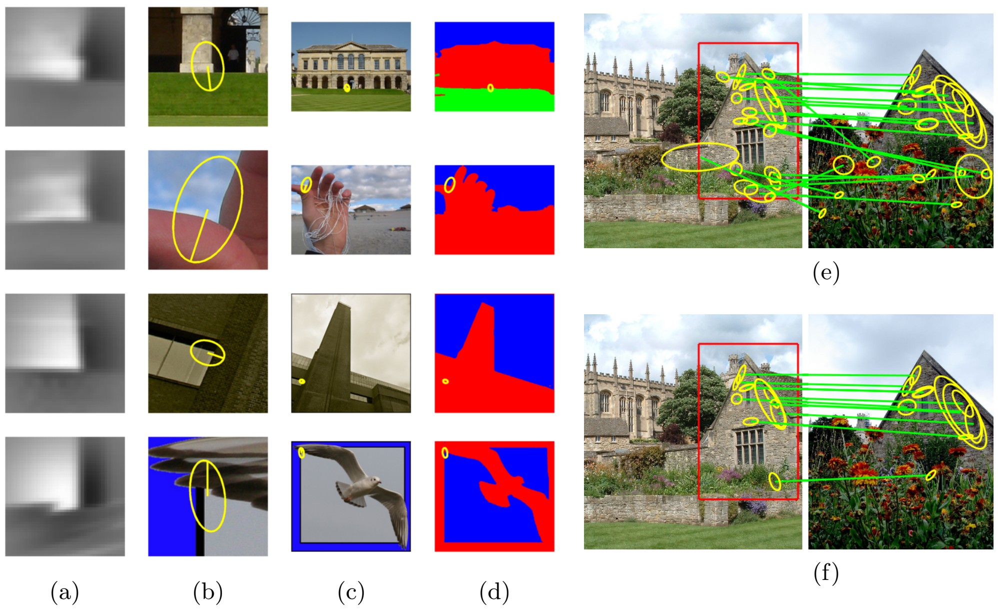


Fig. 1. Patch matching with semantic reasoning. (a) Four normalized image patches extracted from the images in (c); (b) zoom-in around the patches (with standard size and rotation normalization). The four patches are very similar, their descriptors match and it is therefore impossible to filter these false matches using the patches alone. (d) Automatic segmentation of the image into {sky, grass/trees, other} shown as blue, green and red, respectively. The local patches' semantic content provides sufficient information to discard most of these false matches. The four patches (from top to bottom) contain {grass, other}, {sky, other}, {other}, {sky, other}, and consequently only the {sky, other} patches can match in this case. (e,f) Matched patches without and with semantic reasoning, respectively. The number of matches falls from 25 to 12 due to correctly removing correspondences which were falsely established based on patches alone. Note that no spatial consistency check is performed

principal problems of local descriptor matching: that often the patches are not sufficiently discriminative to avoid false matches – see figure 1. The key idea in this work is to use semantic information to filter out such false matches. We term the augmented SIFT descriptor, 'SemanticSIFT'.

The proposed approach can be seen as belonging to an emerging theme in recent computer vision papers of using semantic information to aid classical computer vision tasks, e.g. in stereo correspondence [25], 3D reconstruction [26] and Semantic SLAM [27, 28]. For example, Haene et al. [26] show that 3D reconstruction can be improved by simultaneously reasoning about the classes of objects present in the image, since the objects provide geometric cues about surface normals.

Despite the obvious benefits of using semantic information in various areas of computer vision, semantic segmentation has not yet been applied to large scale object retrieval. Rather, erroneous SIFT matches have been removed using the context of other descriptors and spatial matching [1, 3, 29, 30, 31]. These methods, and the others listed above, are complementary to SemanticSIFT. Note, SemanticSIFT should be distinguished from “semantic retrieval” [32, 33, 34] in CBIR where the goal is to retrieve visual concepts. Here our focus is on *specific* object retrieval, so queries are “find images containing this particular building” rather than “find images of cars in urban environments”. However, as will be seen, in specific object search knowing that a set of pixels represents a building or not is certainly beneficial as false matches to non-building pixels can be discarded.

In the following, we describe the SemanticSIFT descriptor and its benefits (section 2). It will be seen that it can reduce the size of posting lists, which leads to a speed up in retrieval times, with no loss in retrieval performance. It also removes many erroneous retrievals, and thereby improves retrieval performance. We also describe a method for fast semantic segmentation that is suitable for large scale and real time systems (section 3).

2 Semantic vocabulary for object retrieval

In this section we develop the SemanticSIFT representation, starting with a semantic vocabulary based on a semantic segmentation of the image into multiple classes. We then describe how this semantic vocabulary is used to match local patches, and its combination with a standard BoW visual vocabulary, to form the hybrid SemanticSIFT vocabulary.

2.1 Semantic vocabulary

Suppose we have a semantic segmentation method that provides a pixel-wise labelling into C semantic classes. High accuracy segmentations are desired and therefore for this work we only focus on the following relatively “easy” classes: sky, flora (i.e. grass, trees and bushes), and “other” (containing everything else, including building, road, human, car, table, sea, etc.); therefore $C = 3$. However, the choice of classes is free and the retrieval method is capable of handling any choice given that the semantic segmentation quality is high.

A local image patch is assigned a “semantic word” based on the semantic class/classes it contains: if the patch contains at least one pixel of a particular semantic class c, then it is deemed to contain class c. There are $K_s = 2^C - 1$ possible semantic words representing all possible combinations of a class appearing or not appearing in the patch (note the “-1” is because a patch has to contain at least one semantic class). For our choice of semantic classes ($\{$sky,flora,other$\}$, $C = 3$), there are $K_s = 7$ semantic words formed from all possible non-empty sub-sets: $\{$sky$\}$, $\{$flora$\}$, $\{$other$\}$, $\{$sky,flora$\}$, $\{$sky,other$\}$, $\{$flora,other$\}$ and $\{$sky,flora,other$\}$.

2.2 Matching patches: SemanticSIFT

As with the standard bag-of-visual-words retrieval methods, two local patches are deemed to match each other if they are assigned to the same word. For visual words this typically means that the relevant local descriptors have been assigned to the same cluster obtained by (approximate) k-means. For the semantic word case, words are naturally defined as the combination of classes appearing in the local patch, and no clustering is required. Figure 1 demonstrates the effectiveness of this approach, where several local patches are assigned to the same visual word as they have almost identical appearance and therefore very similar SIFT descriptors (in fact their Hamming signatures also match – see section 4.1). However, looking at the corresponding images it is clear that the matches are invalid due to the difference in their semantic content. Most of the false matches obtained by using visual words can be eliminated by requiring that the semantic words have to match as well. The procedure therefore increases precision, as will be demonstrated in section 4.2.

2.3 Product vocabulary and speedup

In order to deem two patches to be matching, both their visual words and semantic words have to be the same. This behaviour is identical to defining a hybrid, SemanticSIFT, vocabulary and demanding that the hybrid words match. The SemanticSIFT vocabulary is effectively a product vocabulary of the visual and the semantic one – its size is $K = K_v \times K_s$, where K_v and K_s are the sizes of the visual and semantic vocabularies, respectively. In other words, the SemanticSIFT vocabulary has all combinations of visual and semantic words. If the visual words are v_1, v_2, .., v_{K_v} and semantic words are s_1, s_2, ..., s_{K_s}, then SemanticSIFT words are (v_1, s_1), (v_1, s_2), ..., (v_1, s_{K_s}), (v_2, s_1), (v_2, s_2), ..., (v_2, s_{K_s}), ..., (v_{K_v}, s_{K_s}).

Standard large scale retrieval systems based around the bag-of-words concept use very large visual vocabularies, ranging between 100k to 16M words, making the bag-of-word image representation very sparse. Fast ranking is performed using an inverted index which exploits the BoW sparsity, where images containing a particular visual word are arranged in a posting list. Larger vocabularies cause BoWs to be sparser, which makes average posting list length smaller, and therefore improves the retrieval speed as a smaller number of entries in the inverted index are visited during the scoring. Since the SemanticSIFT vocabulary size is $K_v \times K_s$, i.e. K_s times larger than the baseline vocabulary (where our $K_s = 7$), the average posting list length decreases K_s times making retrieval significantly faster. However, ranking is not actually K_s faster as some semantic words are much rarer than others (e.g. patches which contain all three semantic classes are rarer than patches which contain only "other"). Nevertheless, we experimentally obtain an almost two-fold speedup (see section 4.2).

2.4 Reduction in memory requirements

As discussed earlier, SemanticSIFT increases precision by removing some false matches, while simultaneously improving retrieval speed by visiting shorter posting lists. Both of these improvements come at no storage cost at all as the number of entries in the inverted index doesn't change (the average posting list length decreases as the vocabulary size increases, but the total number of entries remains the same). It is possible to reduce storage requirements, and therefore reduce the RAM consumption of the retrieval system if it is known *a priori* that a particular class of objects is not interesting to the user. For example, it is reasonable to assume that pure sky features are of no use in most retrieval applications as it is unlikely that a user will search for a particular detail in the sky or a particular cloud. It is also likely that features which only contain flora are also not useful or even detrimental as they are often not distinctive enough. Note that we are not proposing to remove all features that contain flora, just the ones that contain only flora or only {flora,sky}, because {flora,other} can indeed be distinctive. For example, a feature from an interface between a building and grass is potentially useful. Note that, apart from reducing storage requirements, removal of features also decreases computational cost as there is a further reduction in the number of visited items in the inverted index.

2.5 Accounting for segmentation uncertainty

In an ideal case when perfect semantic segmentation of all images is available, SemanticSIFT is guaranteed to improve precision. However, in a real-world scenario the "hard" nature of the matching procedure, i.e. requiring that semantic words have to be identical, could potentially hurt recall. This is particularly evident near the borders between two different semantic classes, where uncertainty in the exact position of the border can lead to noisy semantic word assignments, resulting in false rejection of matching patches.

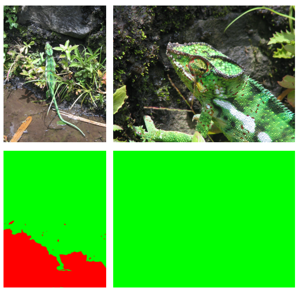
Here we describe a simple extension to SemanticSIFT, called SoftSemanticSIFT, which can take these uncertainties into account. It only assumes that the the underlying semantic segmentation method can output an extra special label, *unknown*, without placing any constraints on the way this information is obtained. For example, one could use the method in section 3 which compares costs across classes, or have an *ad hoc* post-processing step which simply marks pixels close to estimated semantic discontinuities as "unknown".

SemanticSIFT matching, as before, enforces that the underlying patch semantic content has to be identical, but the special "unknown" label is allowed to match any other label(s). For example, {sky,other} are allowed to match {unknown} because the "unknown" pixels could potentially contain "sky" and "other". Note that a patch which contains "unknown" cannot simply match any other patch, as for example {flora,unknown} cannot match {other} as "unknown" can match "other" but the latter patch does not contain "flora".

Impact on speed and memory requirements. The SoftSemanticSIFT is slower than SemanticSIFT because of its smaller rejection rate of visual word matched features, but is also faster than the BoW matching as many patches will remain rejected. The number of semantic words increases to $K_s = 1+2\times 7 = 15$, because an extra {unknown} word is added, and all of the original 7 semantic words can appear in two variants (with or without "unknown"). However, unlike SemanticSIFT, multiple posting lists are visited for a single query feature; for example, for a query {flora} posting lists for {flora}, {flora,unknown} and {unknown} need to be visited (obviously, the search is restricted to the same visual word, as before). Likewise, storage savings with respect to the baseline visual retrieval system are smaller than SemanticSIFT, but still exist.

2.6 Challenges

Some images are very challenging for automatic semantic segmentation, even for the seemingly simple task of finding sky, flora and other. Therefore, it is expected that failures occur. However, not all miss-classifications are catastrophic for the task considered in this paper. As illustrated in figure 2, provided that mistakes are consistent matches will not be lost.


(a) A camouflaged lizard

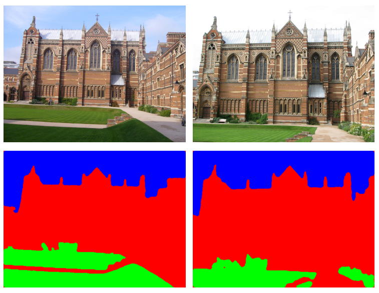

(b) A blue roof building

Fig. 2. Hard cases for semantic segmentation. Two pairs of challenging images are shown in the top row, and the corresponding semantic segmentation is shown in the bottom row. The left pair shows a green lizard which due to camouflage looks like grass, the right pair shows an Oxford college with a sky-colour roof. Even though semantic segmentation fails in these two challenging cases, the retrieval system is not hampered because the miss-segmentation is systematic and repeatable, e.g. the green lizard is always classified as grass and therefore correctly matched features will not be removed

3 Fast semantic segmentation

For the proposed SemanticSIFT method we have two principal requirements for the semantic segmentation: first, *speed*, as we envisage large scale deployment (millions to billions of images) and real time processing of uploaded query images; and second, *accuracy*, as incorrect segmentations may reduce recall. Of course, these two requirements often conflict: currently the methods of [35, 36, 37] produce state of the art semantic segmentations (e.g. on the Stanford background dataset [38]) but take several minutes per image; conversely, existing fast approaches either produce results of insufficient quality, such as Semantic Texton Forest [39] (see supplementary material for examples), or make assumptions which are often violated in real world images, such as the "Tiered Scene" assumption [40] where sky is, if present in the image, forced to be above any other label (this assumption is often invalid in real images, see bottom row of figure 1 or supplementary material).

For this reason we introduce our own semantic segmentation method, dubbed Fast Semantic Segmentation via Soft Segments (FSSS), which is described next. It proceeds in two stages: first, a number of soft segments are defined across the image centred on a regular grid; second, these soft segments are used to provide context when labelling pixels with their semantic class. Note, however, that any available fast and accurate semantic segmentation method could be employed for SemanticSIFT.

3.1 Soft segments

Segmentation methods often employ super-pixels or multiple-segmentations [35, 36, 41] to provide context for pixel labelling and/or to reduce complexity. Here, rather than making a hard decision on such segments, we obtain soft-segments (or soft super-pixels) using the embedding method of [42] together with spatial proximity. The method [42] provides an 8 dimensional embedding for every pixel in an image, such that a small L2 distance between a pair of pixels in this feature space signifies that the two pixels are likely to be a part of the same superpixel. The squared L2 distance between pixels i and j is denoted as $D_s(i, j)$. The method is very fast, producing an embedding for a 500×500 image in 1.7 seconds on a CPU.

A soft-segment takes account of both the similarity in pixel appearances (from $D_s(i, j)$) and also their spatial distance (e.g. a blue car shouldn't be in the same soft-segment as blue sky because they are not close in the image space). Suppose a soft-segment is centred on pixel i (pixel i is the seed), then we define the association of pixel j to this soft-segment by the weight

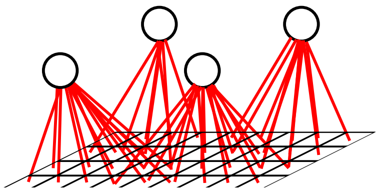
$$w_{i,j} = \exp(-\alpha D_s(i, j) - \beta D_p(i, j)) \tag{1}$$

where $D_p(i, j)$ is the squared L2 distance between normalized pixel locations, and α and β are parameters (to be learnt). Figure 3b shows several examples of soft-segments. A set of soft-segments is obtained for an image by seeding segments on a regular grid (i.e. the i pixels above are chosen on a regular grid).

3.2 Labelling

A labelling of the pixels of the image (into the semantic classes) *and* of the set of soft-segments (into the same semantic classes) is obtained by minimizing an energy function. As usual, there are two terms, but in this case the unary is over soft-segments and is low when the label is consistent with the appearance; and the second term is between soft-segments and pixels, and is low when the labelling of the soft-segment is consistent with the labelling of pixels (softly) associated with it. More formally, the energy $E(l, L)$ of a labelling l is defined as follows.

$$E(l, L) = \sum_i \left(\lambda \phi_i (L_i) - \sum_j w_{i,j} \delta (L_i = l_j) \right) \tag{2}$$


(a) Model

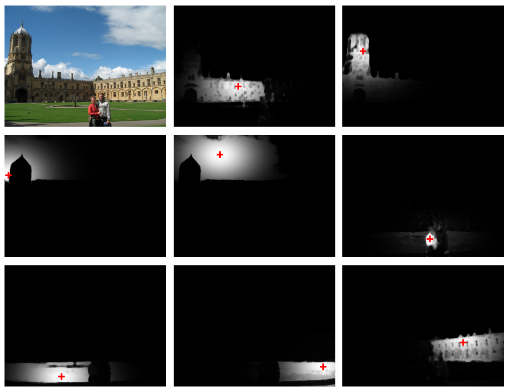

(b) Soft segments

Fig. 3. Graphical model and soft-segments. (a) The graphical model corresponding to the energy (2). Pixels (rectangular grid) are connected to multiple soft-segments (shown as circles), where (b) illustrates the support of the soft-segments. (b) Top left: the original image, all other images show soft-segments as defined in equation (1). Each image shows the "seed" for the soft-segment as a red cross, and the brightness depicts the weight $w_{i,j}$ between the seed pixel i and all pixels j. Note, (i) that the soft-segments are localized and do not cross semantic boundaries, and (ii) the varying effective sizes, e.g. soft-segments containing sky and buildings are large while the one centred on a person is small

where l_j is the label of pixel j, L_i is a latent variable signifying the label of the soft superpixel defined at the grid point i, $\phi_i(L_i)$ is the penalty for superpixel i having the label L_i (i.e. the "unary potential"), $w_{i,j}$ is the reward obtained when soft-segment i and pixel j have the same label, $\delta(L_i = l_j)$ is an indicator function which yields 1 if L_i equals l_j and 0 otherwise, and λ is the relative weighting of the unary and pairwise terms. The graphical model corresponding to this energy is shown in figure 3a. The unary term is described in more detail below.

Efficient inference. From equation (2) and the corresponding graphical model in figure 3a, it is clear that, by design, the pixel labels l are conditionally independent from each other given the soft segment labels L, and vice versa. The energy function is optimized by initializing the segment labels L according to the unary potentials, followed by iterating between fixing L and optimizing for pixel labels l, and fixing l and optimizing for L. The procedure can be interpreted as message passing, where the first set of messages are sent top-down from soft superpixels to the underlying pixels, and the second set of messages are sent bottom-up from pixels to soft superpixels.

The iterations finish when a step does not change any label; in practice the algorithm converges quickly, in 1 to 7 iterations depending on the complexity of the scene. Several examples of the produced semantic segmentations are shown in figure 4 and more are available in the supplementary material.

Segmentation uncertainty. It is sometimes beneficial for a semantic segmentation algorithm to be able to provide an extra "unknown" label when it is uncertain about the true labelling, instead of making a hard and potentially wrong decision; such functionality is exploited by the SoftSemanticSIFT method (section 2.5). A pixel is assigned the "unknown" label if perturbing its optimal label does not change its contribution to the energy (equation (2)) more than a threshold value. The uncertainty estimation does not impact computational efficiency as it is obtained directly during the optimization of the energy function; the supplementary material contains more details.

Implementation details. The features used are a histogram of colours in HSV space, a bag-of-words of dense RootSIFT [18], and a histogram of normalized y pixel locations. These are concatenated into a single feature vector. Each pixel generates one very sparse feature vector (e.g. the histogram of colours component has exactly one non-zero element). The feature vector for a soft-segment L_i is a weighted sum of pixel-wise feature vectors, where the weight for pixel j is $w_{i,j}$, as defined above in (1). We train a multi-class one-vs-all linear SVM on top of soft-segment features, using a Hellinger kernel implemented as an explicit feature map [43]. The soft-segment unary potentials $\phi_i(L_i)$ are then computed directly as negative classifier scores. All implementation details are provided in the supplementary material, i.e. number of soft-segments per image, grid spacing, values of α and β parameters, etc.

The run time is 7 seconds, with a pure MATLAB CPU implementation, on a 500×500 pixel image, including all required preprocessing (i.e. superpixelization, feature extraction, computation of unary and pairwise potentials, etc). Full source code, including inference, training and pre-trained models, is available at [44].

3.3 Segmentation results

Our FSSS method is evaluated on the standard Stanford background dataset [38] which contains 715 320×240 images and 8 semantic categories. This is the most suitable standard benchmark for our task, as others, such as PASCAL VOC, concentrate on objects rather than 'stuff'. As is standard procedure [36, 37, 38], segmentation quality is assessed as the average pixel accuracy across five random splits of the data. Our method achieves competitive performance, 78.0%, while only taking 3.7 seconds end-to-end (i.e. including all feature extraction, soft-segment computation and pixel labelling). The results compare favourably to existing methods which generally use far more complicated features and take longer to compute. For example [38] achieves 76.5% (and uses additional features including the horizon location information and segment shape), and reports that inference takes up to 10 minutes; Tighe and Lazebnik [45] achieve 77.5% with a large collection of features: superpixel shape, GIST, location, texton histograms, SIFT histogram, SIFT histogram on superpixel boundary, colour, context, etc. The best reported accuracy is 81.9% [36], but this method takes minutes per image as it relies on gPb contour detector [46] and superpixelization [47].

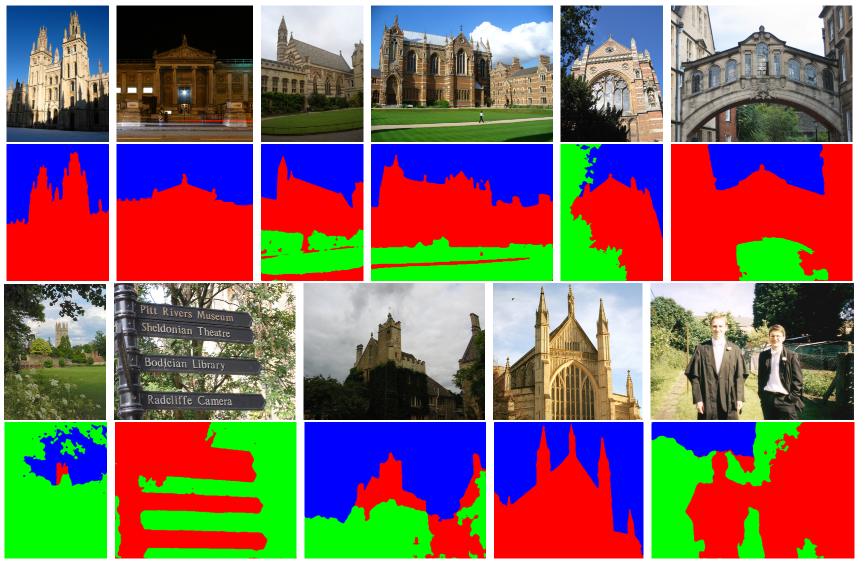


Fig. 4. Example semantic segmentations. Pairs of rows show the original image on the top, and the automatic semantic segmentation on the bottom. The three classes {sky, flora, other} are shown in blue, green and red, respectively. Best viewed in colour. See the supplementary material for more examples

We have demonstrated that FSSS is 'fit for purpose': it has been designed for fast and accurate semantic segmentation, and achieves this whilst being comparable to the state of the art in terms of multi-class semantic segmentation performance.

4 Experimental setup and retrieval results

4.1 Evaluation, datasets and baseline

Retrieval performance is assessed using the standard and publicly available Oxford Buildings benchmark [14]. The basic dataset, Oxford 5k, contains 5062 high-resolution images downloaded from Flickr. Retrieval quality is measured in terms of mean average precision (mAP) over the 55 pre-defined test queries. In order to test larger scale retrieval, the dataset is often expanded with 100k Flickr images acting as distractors, thus forming the Oxford 105k dataset. We follow the common practice [15, 20, 48, 49] of using an independent dataset, Paris 6k [48], for all training (e.g. computation of the visual vocabulary, training for semantic segmentation, etc).

Baseline. We have implemented a baseline retrieval system based on the Hamming Embedding [15] with burstiness normalization [50]. In detail, we extract RootSIFT [18] descriptors from Hessian-Affine interest points [51], and quantize them into 100k visual words. A 64-bit Hamming Embedding [15] signature is stored together with each feature in order to improve feature matching precision. Two features are deemed to match if they are assigned to the same visual word and their Hamming signatures are within a standard threshold of 24 on the Hamming distance [15, 52]. For a given query, a similarity score for a database image is obtained by summing all the Gaussian weighted votes of the image's matching features (a standard parameter value of $\sigma = 16$ is used, as in [50, 52]). Finally, burstiness normalization of [50] is applied as well. The visual vocabulary and Hamming Embedding parameters are all trained on the independent Paris 6k dataset.

The baseline achieves good performance (mAP) on Oxford 5k and Oxford 105k benchmarks (table 1): 70.70% and 61.63%, respectively. Adding spatial reranking [14] improves mAPs to 71.95% and 64.38%.

Semantic Segmentation. We have found that the Stanford background dataset [38], commonly used for semantic segmentation benchmarks, is inappropriate for training a segmentation method working on real-world unconstrained Flickr images. It contains 715 relatively small (320×240) images, with, for example, less than 8 million sky pixels. Furthermore, the set only contains outdoor images shot in daytime. Other datasets, e.g. MSRC [53], suffer from similar drawbacks. We have therefore annotated [54] 360 high resolution images (1024×768), randomly sampled from the Paris 6k [48] and Sculptures 6k [55] datasets, both of which have no images in common with Oxford 5k and 105k datasets. The set (named ParisSculpt360) contains many more labelled pixels: 122, 30 and 43 million for sky, flora and other, respectively, compared to the Stanford background dataset which contains only 55M pixels in total. The images are unconstrained, i.e. there are indoors and outdoors photos, taken at day or night, colour and grayscale, and are not necessarily vertically aligned. The 5-fold average accuracy of FSSS on this dataset is 91.6%.

All SemanticSIFT results and qualitative examples are generated using models trained on this ParisSculpt360 training data.

4.2 Retrieval results

In this section we evaluate the three benefits of using SemanticSIFT versus the baseline, namely: improved retrieval quality, speedup and reduced storage/memory requirements.

Retrieval quality. Table 1 shows the retrieval performance of the baseline, SemanticSIFT and SoftSemanticSIFT. The baseline achieves the mAP of 70.70% and 61.63% on the Oxford 5k and 105k benchmarks, respectively. Our SemanticSIFT method improves precision increasing the mAP to 70.82% and 62.28%. However, it struggles to bring an improvement after spatial reranking, because segmentation errors reduce recall. SoftSemanticSIFT, which is robust to some uncertainties in the automatic semantic segmentation, gives better results, 71.17% and 62.34%, and does outperform both the baseline and SemanticSIFT

Table 1. Retrieval performance (mAP). All methods use exactly the same features, 100k visual vocabulary and Hamming signatures with all training performed on the Paris6k dataset (section 4.1). "+Spat." denotes spatial reranking [14]. SoftSemanticSIFT clearly outperforms the baseline and SemanticSIFT methods

Method	Oxford 5k		Oxford 105k	
		+spat.		+spat.
Baseline	0.7070	0.7195	0.6163	0.6438
SemanticSIFT	0.7082	0.7196	0.6228	0.6434
SoftSemanticSIFT	**0.7117**	**0.7238**	**0.6234**	**0.6487**

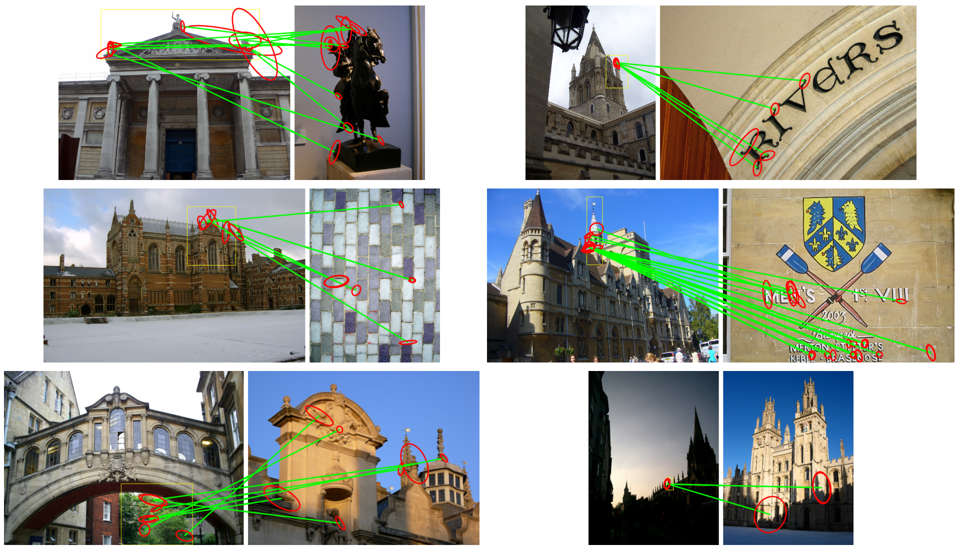


Fig. 5. Filtered false matches. Six pairs of images showing patch correspondences based on baseline descriptor-only based matching (i.e. matching visual word and Hamming signatures). Patches are shown as red ellipses, matching patches are connected with green lines. All displayed matches are correctly filtered out using SemanticSIFT because of the semantic content mismatch. Note, for example, the very challenging pair in the bottom-right, where patches corresponding to spires match patches showing shadows of spires; SemanticSIFT correctly discards these as the former contain {sky,other} while the latter only contain {other}

after spatial reranking as well. We further evaluate the statistical significance of the obtained improvements by employing five different visual vocabularies (corresponding to different random initializations of the approximate k-means algorithm). In all five cases SoftSemanticSIFT outperforms the baseline, for Ox105k the relative improvement is +1.2% on average, the minimal being +1%.

Thus, even a small number of semantic classes leads to an improvement in retrieval mAP performance. It is to be expected that as more classes are introduced this improvement will increase. Figure 5 shows a selection of examples where the use of SemanticSIFT removes false matches (arising from the visual words and Hamming signatures alone).

Table 2. Speedup and reduction in memory requirements. "Speedup" shows the reduction in the number of posting list entries traversed for the 55 pre-defined queries in the Oxford Buildings benchmark, achieved due to using a semantic vocabulary. Relative reduction in memory requirements (index size) is achieved by removing {flora} and {sky,flora} features, while "total speedup" denotes the speedup achieved after the index reduction

Method	Oxford 5k			Oxford 105k		
	speedup	memory savings	total speedup	speedup	memory savings	total speedup
SemanticSIFT	31.6%	13.2%	31.9%	41.1%	19.6%	41.5%
SoftSemanticSIFT	23.1%	9.4%	23.3%	30.7%	14.5%	31.0%

Speedup. As discussed in section 2.3, SemanticSIFT provides a significant speedup due to the reduction in the average posting list length compared to the baseline. The retrieval speed directly depends on the number of posting list entries that are traversed during the ranking stage, which is also equal to the number of Hamming distance computations. Therefore, the appropriate and accurate measure of speedup is the reduction in the average number of traversed posting list entries when using SemanticSIFT compared to the baseline. The speedup (table 2), is 31.6% and 41.1% for the Oxford 5k and Oxford 105k tests, respectively. For the SoftSemanticSIFT case, where more posting list entries are traversed due to handling uncertainty of semantic segmentation, the speedup is smaller but still large: 23.1% and 30.7% for the two tests, respectively.

The above speedup measurements are based on the predefined 55 queries in the Oxford Buildings benchmarks. Another way of assessing retrieval speed across inverted indexes with varying properties is to compare the expected number of inverted index entries the system has to process for an average query [56]. In the supplementary material we give the details of the computation for the case of SemanticSIFT. The result is that the expected speedup over the baseline for an average query is 40.7% and 50.4%, for the Oxford 5k and Oxford 105k tests, respectively.

One alternate way of improving retrieval speed is to increase the visual vocabulary (as this also reduces the length of posting lists on average). However, this leads to increased quantization errors and the retrieval performance suffers: with a 700k visual vocabulary, equal to the size of SemanticSIFT's product vocabulary, the baseline only achieves mAP of 54.9% on Oxford 5k, compared to 70.7% obtained with a 100k vocabulary. In contrast, SoftSemanticSIFT gets 71.2% while preserving the same speedup.

Memory savings. As discussed in section 2.4, features can be removed from the inverted index based on their semantic content if it is known *a priori* that they are not useful for the application in question. When searching for buildings (which belong to the "other" class), as in the Oxford Buildings tests, it is clear that flora features are not useful. Therefore, the database features assigned to semantic words {flora} or {flora,sky} are removed, whilst the semantic word {flora,other} is kept as it can still be useful. With these changes the retrieval quality (mAP) remains virtually unchanged for both Oxford 5k and Oxford 105k tests. The memory/storage saving on the other hand is significant (table 2): 13.2% and 19.6% of features are removed from the SemanticSIFT index for Oxford 5k and 105k, respectively. Furthermore, there is an additional slight improvement in speed due to the decreased number of posting lists that have to be traversed for query images which contain the removed features. The total speedup for the two datasets with respect to the baseline is 31.9% and 41.5%, compared to the 31.6% and 41.1% achieved without feature removal. The speedup is minor because the 55 predefined queries for the Oxford Buildings benchmarks don't contain many flora features.

Colour. One might argue that the SemanticSIFT's increase in retrieval performance is purely due to the use of colour (used indirectly for semantic segmentation), which the baseline method does not use. However, OpponentSIFT [57], the state-of-the-art colour SIFT variant, actually performs slightly worse than SIFT (the mAP decreases by 0.7%) on Oxford 5k. This proves that the improvement from SemanticSIFT is not due to the use of colour, but to taking proper account of the semantic classes when matching.

5 Conclusions and future work

We have presented a method, SemanticSIFT, which improves the standard large scale specific object retrieval by leveraging semantic information to efficiently filter out some falsely matched descriptors, thereby increasing precision and improving retrieval performance. Furthermore, there is a "win-win-win" situation: the gain in recognition accuracy is obtained simultaneously with a nearly two-fold speedup, due to visiting shorter posting lists, and a 20% decrease in storage (RAM) requirements. The method can be used as an improvement to any standard retrieval systems based on matching local patches – a 'plug in' to boost speed and retrieval performance.

Semantic reasoning for object retrieval is a very promising idea that opens many directions for future work, which are out of scope and length limitations of this paper. With future improvements of semantic segmentation methods (which will surely happen over time), one can hope to take more classes into consideration, e.g. people/faces (successful face detection already exists), buildings, cars, roads, flowers, etc. A finer scale reasoning can be used too – for the "buildings" class example one could remove falsely matched features between a window and a door. Successful fine-grained distinctions within a class would be very useful as well.

Another interesting direction is to employ the semantic labels as a form of automatic supervision which could be used in descriptor learning. Automatic generation of training data for descriptor learning in the form of matching and non-matching image patches using object retrieval has been done in [58, 59], but unsupervised discovery of hard negatives has always been a problem. SemanticSIFT can provide the needed automatic supervision.

Acknowledgement. We are grateful for financial support from ERC grant VisRec no. 228180 and a Royal Society Wolfson Research Merit Award.

References

[1] Cummins, M., Newman, P.: Highly scalable appearance-only SLAM - FAB-MAP 2.0. In: RSS. (2009)

[2] Schindler, G., Brown, M., Szeliski, R.: City-scale location recognition. In: Proc. CVPR. (2007)

[3] Knopp, J., Sivic, J., Pajdla, T.: Avoiding confusing features in place recognition. In: Proc. ECCV. (2010)

[4] Torii, A., Sivic, J., Pajdla, T., Okutomi, M.: Visual place recognition with repetitive structures. In: Proc. CVPR. (2013)

[5] Quack, T., Leibe, B., Van Gool, L.: World-scale mining of objects and events from community photo collections. In: Proc. CIVR. (2008)

[6] Gammeter, S., Bossard, L., Quack, T., Van Gool, L.: I know what you did last summer: object-level auto-annotation of holiday snaps. In: Proc. ICCV. (2009)

[7] Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree. In: Proc. CVPR. (2006) 2161–2168

[8] Shen, X., Lin, Z., Brandt, J., Wu, Y.: Mobile product image search by automatic query object extraction. In: Proc. CVPR. (2012)

[9] Romberg, S., Lienhart, R.: Bundle min-hashing for logo recognition. In: ACM ICMR. (2013)

[10] Google Goggles: (http://www.google.com/mobile/goggles)

[11] Sivic, J., Zisserman, A.: Efficient visual search of videos cast as text retrieval. IEEE PAMI **31** (2009) 591–606

[12] Agarwal, S., Snavely, N., Simon, I., Seitz, S.M., Szeliski, R.: Building Rome in a day. In: Proc. ICCV. (2009)

[13] Lowe, D.: Distinctive image features from scale-invariant keypoints. IJCV **60** (2004) 91–110

[14] Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: Proc. CVPR. (2007)

[15] Jégou, H., Douze, M., Schmid, C.: Hamming embedding and weak geometric consistency for large scale image search. In: Proc. ECCV. (2008) 304–317

[16] Jégou, H., Douze, M., Schmid, C.: Exploiting descriptor distances for precise image search. Technical report, INRIA (2011)

[17] Aly, M., Munich, M., Perona, P.: Compactkdt: Compact signatures for accurate large scale object recognition. In: IEEE Workshop on Applications of Computer Vision. (2012)

[18] Arandjelović, R., Zisserman, A.: Three things everyone should know to improve object retrieval. In: Proc. CVPR. (2012)

[19] Chum, O., Philbin, J., Sivic, J., Isard, M., Zisserman, A.: Total recall: Automatic query expansion with a generative feature model for object retrieval. In: Proc. ICCV. (2007)

[20] Chum, O., Mikulik, A., Perďoch, M., Matas, J.: Total recall II: Query expansion revisited. In: Proc. CVPR. (2011)

[21] Shen, X., Lin, Z., Brandt, J., Avidan, S., Wu, Y.: Object retrieval and localization with spatially-constrained similarity measure and k-NN reranking. In: Proc. CVPR. (2012)

[22] Qin, D., Gammeter, S., Bossard, L., Quack, T., Van Gool, L.: Hello neighbor: accurate object retrieval with k-reciprocal nearest neighbors. In: Proc. CVPR. (2011)
[23] Simonyan, K., Vedaldi, A., Zisserman, A.: Descriptor learning using convex optimisation. In: Proc. ECCV. (2012)
[24] Winder, S., Hua, G., Brown, M.: Picking the best daisy. In: Proc. CVPR. (2009) 178–185
[25] Ladicky, L., Sturgess, P., Russell, C., Sengupta, S., Bastanlar, Y., Clocksin, W., Torr, P.H.S.: Joint optimisation for object class segmentation and dense stereo reconstruction. IJCV (2012)
[26] Haene, C., Zach, C., Cohen, A., Angst, R., Pollefeys, M.: Joint 3D scene reconstruction and class segmentation. In: Proc. CVPR. (2013)
[27] Castle, R.O., Klein, G., Murray, D.W.: Combining monoSLAM with object recognition for scene augmentation using a wearable camera. Image and Vision Computing (2010)
[28] Civera, J., Gálvez-López, D., Riazuelo, L., Tardós, J.D., Montiel, J.M.M.: Towards semantic SLAM using a monocular camera. In: IEEE Intelligent Robots and Systems (IROS). (2011)
[29] Turcot, T., Lowe, D.G.: Better matching with fewer features: The selection of useful features in large database recognition problems. In: ICCV Workshop on Emergent Issues in Large Amounts of Visual Data (WS-LAVD). (2009)
[30] Wu, Z., Ke, Q., Isard, M., Sun, J.: Bundling features for large scale partial-duplicate web image search. In: Proc. CVPR. (2009)
[31] Fernando, B., Tuytelaars, T.: Mining multiple queries for image retrieval: On-the-fly learning of an object-specific mid-level representation. In: Proc. ICCV. (2013)
[32] Wang, J.Z., Li, J., Wiederhold, G.: SIMPLIcity: Semantics-sensitive integrated matching for picture libraries. IEEE PAMI (2001)
[33] Duygulu, P., Barnard, K., de Freitas, J.F.G., Forsyth, D.A.: Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In: Proc. ECCV. (2002)
[34] Li, J., Wang, J.Z.: Automatic linguistic indexing of pictures by a statistical modeling approach. IEEE PAMI (2003)
[35] Munoz, D., Bagnell, J.A., Hebert, M.: Stacked hierarchical labeling. In: Proc. ECCV. (2010)
[36] Lempitsky, V., Vedaldi, A., Zisserman, A.: A pylon model for semantic segmentation. In: NIPS. (2011)
[37] Ladicky, L., Russell, C., Kohli, P., Torr, P.H.S.: Associative hierarchical random fields. IEEE PAMI (2013)
[38] Gould, S., Fulton, R., Koller, D.: Decomposing a scene into geometric and semantically consistent regions. In: Proc. ICCV. (2009)
[39] Shotton, J., Johnson, M., Cipolla, R.: Semantic texton forests for image categorization and segmentation. In: Proc. CVPR. (2008)
[40] Felzenszwalb, P.F., Veksler, O.: Tiered scene labelling with dynamic programming. In: Proc. CVPR. (2010)

[41] Russell, B.C., Efros, A.A., Sivic, J., Freeman, W.T., Zisserman, A.: Using multiple segmentations to discover objects and their extent in image collections. In: Proc. CVPR. (2006)
[42] Leordeanu, M., Sukthankar, R., Sminchisescu, C.: Efficient closed-form solution to generalized boundary detection. In: Proc. ECCV. (2012)
[43] Vedaldi, A., Zisserman, A.: Efficient additive kernels via explicit feature maps. In: Proc. CVPR. (2010)
[44] Arandjelović, R., Zisserman, A.: Fast semantic segmentation code. http://www.robots.ox.ac.uk/∼vgg/software/fast_semantic_segmentation (2014)
[45] Tighe, J., Lazebnik, S.: SuperParsing: Scalable nonparametric image parsing with superpixels. In: Proc. ECCV. (2010)
[46] Maire, M., Arbelaez, P., Fowlkes, C., Malik, J.: Using contours to detect and localize junctions in natural images. In: Proc. CVPR. (2008)
[47] Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: From contours to regions: An empirical evaluation. In: Proc. CVPR. (2009)
[48] Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Lost in quantization: Improving particular object retrieval in large scale image databases. In: Proc. CVPR. (2008)
[49] Jégou, H., Chum, O.: Negative evidences and co-occurrences in image retrieval: the benefit of PCA and whitening. In: Proc. ECCV. (2012)
[50] Jégou, H., Douze, M., Schmid, C.: On the burstiness of visual elements. In: Proc. CVPR. (2009)
[51] Mikolajczyk, K., Schmid, C.: Scale & affine invariant interest point detectors. IJCV **1** (2004) 63–86
[52] Tolias, G., Jégou, H.: Local visual query expansion: Exploiting an image collection to refine local descriptors. Technical Report RR-8325, INRIA (2013)
[53] Shotton, J., Winn, J., Rother, C., Criminisi, A.: *TextonBoost*: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In: Proc. ECCV. (2006) 1–15
[54] Arandjelović, R., Zisserman, A.: Parissculpt360 annotations. http://www.robots.ox.ac.uk/∼vgg/data/data-various.html (2014)
[55] Arandjelović, R., Zisserman, A.: Smooth object retrieval using a bag of boundaries. In: Proc. ICCV. (2011)
[56] Stewenius, H., Gunderson, S.H., Pilet, J.: Size matters: exhaustive geometric verification for image retrieval. In: Proc. ECCV. (2012)
[57] van de Sande, K.E.A., Gevers, T., Snoek, C.G.M.: Evaluating color descriptors for object and scene recognition. IEEE PAMI **32** (2010) 1582–1596
[58] Philbin, J., Isard, M., Sivic, J., Zisserman, A.: Descriptor learning for efficient retrieval. In: Proc. ECCV. (2010)
[59] Simonyan, K., Vedaldi, A., Zisserman, A.: Learning local feature descriptors using convex optimisation. IEEE PAMI (2014)