Learning to Rank Bag-of-Word Histograms for Large-scale Object Retrieval

Danfeng Qin
http://www.vision.ee.ethz.ch/~qind/
Yuhua Chen
yuhchen@ee.ethz.ch
Matthieu Guillaumin
http://www.vision.ee.ethz.ch/~mguillau/
Luc Van Gool
http://www.vision.ee.ethz.ch/~vangool/

Computer Vision Laboratory
ETH Zurich
Switzerland

Abstract

Most state-of-the-art object retrieval systems rely on ad-hoc similarities between histograms of quantised local descriptors to find, in their databases, all the images relevant to an image query. In this work, our goal is to replace those similarities with ones that are specifically trained to maximize the retrieval accuracy. We propose to use a simple and very general linear model whose weights directly represent the similarity values. We devise a variant of rank-SVM to learn those weights automatically from training data with fast convergence and we propose techniques to limit the number of parameters of the model and prevent overfitting. Importantly, the flexibility of our model allows us to seamlessly incorporate well-known image retrieval schemes such as burstiness, negative evidence and idf weighting, and still exploit inverted files for efficiency in the large-scale setting. In our experiments, we show that our approach consistently and significantly outperforms the similarities used in several state-of-the-art systems on 4 standard benchmark datasets. In particular, on the Oxford105k dataset, our method outperforms the closest competitor by 6%.

1 Introduction

Retrieving images of a particular query object in a large database of images is an important problem for computer vision with applications in object discovery [10], 3D reconstruction [4], location recognition [25] and mobile visual search [12]. Most recent state-of-the-art large-scale image retrieval systems rely on local features, in particular the SIFT descriptor [17] and its variants. Typically, those local descriptors are aggregated into a histogram-based representation of the image referred to as the Bag-of-Words model (BoW) [28]. BoW models considerably reduce the computational burden and the memory footprint of the systems, because local descriptors are quantised into *visual words*.

For BoW histograms, it is common to use simple similarity functions such as the inner product or cosine similarity [9, 21, 22, 24]. However, such functions are not optimal for modelling the visual similarity between BoW features and thus lead to sub-optimal performance for retrieval [13, 15, 37]. The potential problems are the following: a) The evidence coming from co-missing visual features is under-estimated [13]; b) The similarity between a query image and a database image should not be symmetric [37]; c) Statistical properties of visual words are not taken into account [8, 15, 35].

Even though different methods have been proposed to address each of these problems individually, none provides a satisfying solution to properly account for all of them. Moreover, most authors propose ad-hoc solutions by means of functions controlled by very few parameters. These parameters are then hand-tuned or exhaustively searched on validation/test data to adapt them to each dataset. In this work, we address the problem in a different way, by learning the values of the similarity function directly. Because the number of parameters becomes too large to be set by hand, we learn them using training data.

In the following, we make the following contributions. Firstly, we propose a simple additive approximation of the methods discussed above that leads to our linear model for similarity. We analyze how this model can integrate various statistical properties of the data implicitly. Secondly, in order to learn the parameters of our model on training data, we show how the learning problem can be seen as learning to rank from pairs of images. For this, we optimize a loss function inspired by rank-SVM [16] so as to maximize an approximation of the mean average precision (mAP) of the system. We also show how our model integrates into an efficient inverse file structure and thus how to use it in large-scale retrieval scenarios. In our experiments, we show that our method consistently and significantly outperforms existing similarity measures on four standard image retrieval benchmarks.

This paper is organized as follows. In Sec. 2, we summarize related work. We describe our model and contributions in Sec. 3. In Sec. 4 we present our experimental validation and we draw conclusions in Sec. 5.

2 Related work

The Bag-of-Words (BoW) representation has become the *de facto* standard for large-scale image and object retrieval [21]. In this model, the space $\mathbb{R}^D$ of local features is clustered using k-means into K bins, and an image x, viewed as an unordered set of local descriptors, is represented by $x = [x_i]_{1 \ldots K} \in \mathbb{N}^K$ by counting how many local descriptors of x fall in bin i.

Most recent works in retrieval have focused on improving this model in various ways. The first direction is to improve the different components, such as the local feature representation [23, 27, 31, 36] or the visual codebook [11, 18]. Another is to add more information in the BoW model, such as spatial layout [26, 34], attributes [30] or higher-order statistics [32]. Several works also propose to combine the BoW model with post-processing techniques to further filter retrieval results and obtain state-of-the-art performance. For instance: spatial verification using RANSAC [21], voting based on the Hough transform [14, 26, 34], query expansion [7, 9] or reciprocal nearest neighbours [24, 33].

Our work focuses on an important component of the retrieval system, namely the similarity function used to rank results. Below, we describe in detail the main similarity functions proposed in the literature and used in state-of-the-art methods.

Weighted cosine similarity. The weighted cosine similarity $s_{\cos}$ is the most common measure for BoW in the literature [9, 21, 22, 24]. It is simply computed as a weighted, normalised inner product between the query q and database images d:

$$s_{\cos}(q,d) = \frac{1}{\|q\|\|d\|} \sum_{i=1}^{K} w_i q_i d_i, \tag{1}$$

where the weights w_i account for the relative importance of visual words. A common approach is to use the squared Inverse Document Frequency (idf) of the database $\mathcal{D}$:

$$\sqrt{w_i} = \text{idf}(i) = \log|\mathcal{D}| - \log|\{d \in \mathcal{D} : d_i > 0\}|. \tag{2}$$

Negative evidence. One problem of the cosine similarity is that only the co-occurrence of visual words is counted as an evidence of similarity while the visual similarity coming from co-missing ones is ignored [13]. Firstly, Eq. (1) accumulates the similarity only over co-occurring visual words. Secondly, the normalization term is also unaltered by absent features. To take this *negative evidence* into account, Jegou *et al*. proposed to transform the original BoW vector by substracting the average vector $\bar{d}$ of $\mathcal{D}$: $x' = x - \beta\bar{d}$, where β is a tuning parameter. With the transformed features q' and d' as input, Eq. (1) now gives a positive contribution for co-missing words. β is tuned by brute force search.

Asymmetric dissimilarity. Another problem is that, in many real world retrieval applications, the (dis-)similarity measurement between a query image and a database image should not be symmetric [37]. In several retrieval scenarios, the query image is restricted to only contain the object of interest, with minimal background. On the other side, database images are unrestricted. Therefore, the presence or absence of features should be weighted differently whether it is in q or d: $-s_{\text{asym}}(q,d) = \sum_{i=1}^{K}(d_i - q_i)^p \mathbb{I}(d_i > q_i) + \lambda \sum_{i=1}^{K}(q_i - d_i)^p \mathbb{I}(q_i > d_i)$, where p is a constant in $\{1,2\}$, $\lambda > 0$ and $\mathbb{I}(\cdot)$ is the indicator function. In other words, a difference between d_i and q_i is penalized by 1 if $d_i > q_i$ and by λ otherwise. Zhu *et al*. [37] proposed, for $p = 1$, to vary λ as a function of the query and a tuning parameter α, using $\lambda = \alpha \frac{\sum_{j=1}^{N}\sum_{i=1}^{K} d_i^j}{\sum_{j=1}^{N}\sum_{i=1}^{K}\min(q_i, d_i^j)} - 1$.

Burstiness weighting. The idf alone is not sufficient to model all the statistical properties of visual words. A missing aspect is that visual words do not appear independently but in bursts [15]. The above metrics typically over-estimate the similarity for visual words with many occurrences. Instead, the penalty for visual word count difference should be attenuated as the raw value grows. This effect is obtained with a sub-linear transformation of the features [15]. Using simple algebra, we can show that the inter-image and intra-image burstiness models are equivalent to using the following similarity:

$$s_{\text{burst}}(q,d) = \sum_i \left(\text{idf}(i)^2 q_i \sqrt[4]{d_i} \Bigg/ \sqrt{\sum_{j=1}^{N} \sqrt{d_i^j}} \right). \tag{3}$$

In summary, all of these similarity measures can be written in a very general form as:

$$s(q,d) = \tau(q)\tau(d) \sum_{i=1}^{K} s_i(q_i, d_i), \tag{4}$$

where the specific choice of τ and s_i help address a specific problem. This approach is not entirely satisfying, as it is challenging to design τ and s_i to account for all of these observations simultaneously and to adapt them to new phenomena to be discovered in the future. Instead, we propose to learn their values directly from training data, as we show below.

3 Learning to rank histograms by similarity

Following previous work [13, 15, 37], we start from Eq. (4) to define the similarity between two BoW histograms. As explained before, authors often motivate their choices of τ and s_i by aiming at the correction of potential shortcomings of previous choices. Instead, we propose to resort to learning and discover the patterns of a good similarity function for image search, automatically from training data. We describe below our model in Sec. 3.1, how we learn its parameters in Sec. 3.2, then devise in Sec. 3.3 simple techniques to improve the robustness of the system by reducing the number of parameters to learn. Finally, we describe in Sec. 3.4 how to integrate our model into an inverted index to allow the use of large-scale databases.

3.1 A linear approximation of histogram similarity

Looking at Eq. (4), we aim at learning the values $s_i(q_i,d_i)$ directly. This is notably impractical, as each q_i and d_i can be arbitrarily large. However, state-of-the-art methods use very large visual codebooks ($K \approx 10^6$) leading to sparse of BoW representations, with few occurrences of any visual word in any given image.[1] As a result, using a *truncated histogram* $\hat{q}_i = \min(q_i,n)$ with $n \in \mathbb{N}^+$ will provide an excellent approximation of the original histogram while limiting the number of possible values of $s_i(\hat{q}_i,\hat{d}_i)$ to $(n+1)^2$.

Additionally, because we learn the values of $s_i(\hat{q}_i,\hat{d}_i)$ directly, these terms can be learned to incorporate a *contribution to the normalisation functions*. This leads to a modified similarity $\hat{s}_i$ and our approximated model becomes additive and writes as:

$$s(q,d) = \tau(q)\tau(d)\sum_{i=1}^{K} s_i(q_i,d_i) \approx \sum_{i=1}^{K} \hat{s}_i(\hat{q}_i,\hat{d}_i), \tag{5}$$

where $\hat{s}_i(j,l)$ for $j,l \in [0,n]$ are the $K\cdot(n+1)^2$ parameters to learn. Notably, this additive approximation allows to rewrite Eq. (5) as a linear combination of indicator functions:

$$\hat{s}_i(\hat{q}_i,\hat{d}_i) = w_{i\hat{q}_i\hat{d}_i} = \sum_{j=0}^{n}\sum_{l=0}^{n} w_{ijl}\mathbb{I}(\hat{q}_i = j)\mathbb{I}(\hat{d}_i = l), \tag{6}$$

where $w_{ijl} = \hat{s}_i(j,l)$. In other words, if we define $\Psi(q,d)$ as the binary vector indexed by (i,j,l) such that $\Psi_{ijl}(q,d) = \mathbb{I}(\hat{q}_i = j)\mathbb{I}(\hat{d}_i = l)$ and define $\mathbf{w} = [w_{ijl}]_{i,j,l}$, then:

$$s(q,d) \approx \mathbf{w}^\top \Psi(q,d). \tag{7}$$

Importantly, Eq. (7) highlights that Ψ acts as a feature encoding for the query-document pair (q,d) in a linear prediction model. Despite its simplicity, this model is very general and flexible, and is able to incorporate many of the properties discussed in Sec. 2, and potentially others, without having to explicitly model them.

To illustrate this, let us first consider the simple case of $n=1$. In such case, the truncated histogram $\hat{q}$ simply encodes the absence or presence of visual words (an encoding often referred to as *max-pooling* or *binary bag-of-words*), and there are only 4 weights to learn per visual word: co-absence $\hat{s}_i(0,0)$, co-occurrence $\hat{s}_i(1,1)$ and either case of mutual exclusion $\hat{s}_i(0,1)$ and $\hat{s}_i(1,0)$. If we learn that $\hat{s}_i(0,0) > \hat{s}_i(0,1)$, then not only have we implicitly learned that co-absence of the visual word i contribute more to the similarity than mutual exclusion (as argued by [13]) but also exactly by which amount. If we learn that $\hat{s}_i(0,1) \neq \hat{s}_i(1,0)$, then this implies that the ideal similarity is indeed asymmetric [37]. Finally, learning all the weights together allows to identify which visual words are more important than others, as indicated by the relative weight of $\hat{s}_i(1,1)$ and $\hat{s}_j(1,1)$. Hence, it automatically models re-weighting schemes such as idf. Finally, when $n>1$, phenomena such as burstiness [15] are also learnt.

[1]As an example, in the UKbench dataset, more than 81% of the visual words occur at most twice in any image and more than 97% of them occur at most 5 times.

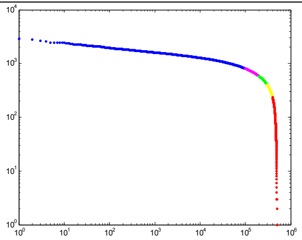

Figure 1: Visual word log-frequency in Oxford105k. Colors illustrate the proposed clustering with 5 groups.

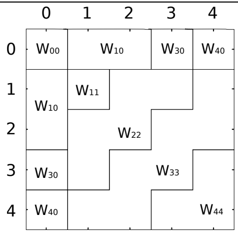

Figure 2: Weight pattern learnt on the UKbench^s data. This pattern was learnt for $N_v = 5$, $N_w = 8$ and enforcing symmetry.

3.2 Learning to rank query-document pairs

In this section, we delve into the details of learning the parameters of our similarity function. Let $\mathcal{D} = \{d^1, \ldots, d^{N_\mathcal{D}}\}$ be the database of $N_\mathcal{D}$ images with their BoW representation d^i. Similarly, let $\mathcal{Q} = \{q^1, \ldots, q^{N_\mathcal{Q}}\}$ be a set of $N_\mathcal{Q}$ query images. For each query $q \in \mathcal{Q}$, $\mathcal{D}$ is partitionned in a relevant subset $U_\mathcal{D}(q)$ and an irrelevant one $V_\mathcal{D}(q)$. Ideally, the similarity function s would satisfy the following constraints:

$$\forall q \in Q,\ \forall (u,v) \in U_D(q) \times V_D(q), \quad s(q,u) > 1 + s(q,v), \tag{8}$$

which translate the idea that, for each query, any relevant document should yield a larger similarity than any irrelevant one by a margin of 1. Considering that our model in (7) is linear, we can simply resort to the following rank-SVM [16] formulation to learn the weights that minimize the number of violated constraints (*i.e.*, the mAP of the system):

$$\operatorname*{argmin}_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \sum_{q\in\mathcal{Q}} \sum_{u\in U_\mathcal{D}(q)} \sum_{v\in V_\mathcal{D}(q)} \max\Big(0, 1 - \mathbf{w}^\top\left(\Psi(q,u) - \Psi(q,v)\right)\Big), \tag{9}$$

where λ controls the trade-off between the regularization of $\mathbf{w}$ and the data loss term.

We propose to use a variant of this model to prevent relevant documents that are very far in the retrieval list to incur very large penalties, and normalise query-specific losses as in [6]. This consists in first rewriting Eq. (8) with:

$$\forall q \in \mathcal{Q},\ \forall u \in U_\mathcal{D}(q), \quad s(q,u) > 1 + \max_{v\in V_\mathcal{D}(q)} s(q,v), \tag{10}$$

which leads to the corresponding optimisation problem:

$$\operatorname*{argmin}_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \sum_{q\in\mathcal{Q}} \frac{1}{|U_\mathcal{D}(q)|} \sum_{u\in U_\mathcal{D}(q)} \max\left(0, 1 - \mathbf{w}^\top\Psi(q,u) + \max_{v\in V_\mathcal{D}(q)} \mathbf{w}^\top\Psi(q,v)\right). \tag{11}$$

Both problems are convex, hence can be optimized in the primal using sub-gradient descent. As we will show in our experiments, our proposed formulation in Eq. (11) leads to a faster optimization and learns more robust parameters compared to [16].

3.3 Robust parameter estimation

As it is described above, our model suffers from an important issue. On one side, retrieval systems improve with larger visual vocabularies. On the other side, the number of parameters to estimate grows with K and, the larger the K, the fewer training data we can obtain to estimate the $(n+1)^2$ weights for each visual word. Learning $K(n+1)^2$ weights is simply unreasonable. Below, we propose two solutions to reduce the number of parameters and thus avoid overfitting. First, we group visual words together and learn a single set of weights for each group (Sec. 3.3.1). Second, we reduce the number of weights in each set (Sec. 3.3.2).

3.3.1 Grouping visual words by frequency

Using large vocabularies causes the number of parameters to grow too large. Hence, we propose to identify groups of visual words and share weights among visual words within each group. Similar to previous works that relate visual word frequency with visual word importance (*e.g.*, using idf), we also decide to associate similar frequencies with similar weights. In practice, we resort to a simple unsupervised clustering where we sort the K visual words according to their frequency in the database $\mathcal{D}$ and create N_v groups $\{g_1,\dots,g_{N_v}\}$ such that each group g_k has the same number K/N_v of visual words.

This grouping can be easily incorporated in our optimization problem of Eq. (11) by summing the components of Ψ corresponding to the visual words in each group g_k, leading to $\Psi'(q,d) = [\Psi'_{kjl}(q,d)]_{k,j,l}$ with $\Psi'_{kjl}(q,d) = \sum_{i\in g_k} \Psi_{ijl}(q,d)$. Notably, this grouping is performed only during training to learn the value of w_{kjl} for all k, j, and l. At test time, visual words are kept separate to retain they discriminative power and we use for w_{ijl} the value w_{kjl} of the group g_k containing i. Fig. 1 illustrates the distribution of visual words frequencies in Oxford105k and its clustering.

3.3.2 Grouping weights with patterns

We also propose to share weights within the same visual word group. First, because small variations in x_i may not have enough impact to justify having different weights. But also because as n grows, many specific combinations of visual word counts will become too rare to learn their corresponding weight robustly.

Jointly over all the visual word groups, we propose to cluster the $(n+1)^2$ weights into N_w bins $\{b_1,\dots,b_{N_w}\}$. We resort to the following iterative procedure. We start with $(n+1)^2$ bins, *i.e.* each weight index (j,l) being associated with its own bin b_m. At each subsequent iteration, we merge the 2 adjacent bins[2] whose merging yields the best *mAP* on training data. We iterate until there are only N_w bins left. We refer to the resulting binnings as *weight patterns*, and we illustrate them in Fig. 2. If needed, we can enforce symmetry of the weights.

Importantly, although the visual word groups share the same weight patterns, they still have their own specific values for the weights themselves. That is, the number of parameters of the model is $N_v \times N_w$. The weight patterns are incorporated in the training algorithm in the same way as the visual word grouping, as described in Sec. 3.3.1.

As a short remark, weight patterns can be viewed as a generalization of truncated histograms, as the latter already correspond to weight sharing for histogram values beyond n (*c.f.* Sec. 3.1). In a way, we are here searching for an optimal projection of $\mathbb{N}^2$ to $[1,N_w]$ such that the corresponding similarity values maximize the retrieval performance of the system.

3.4 Integrating in an inverted index

An important aspect of object retrieval system is its ability to scale gracefully with the number of images in the database and the size of the visual codebook. Using Eq. (7) to this end is not practical. For a given query, it involves building the joint feature encoding Ψ with each document in the database and then applying a potentially large vector-matrix multiplication. It turns out that, using simple algebra, we can rewrite $s(q,d) = \sum_i w_{iq_id_i}$ the following way:

$$s(q,d) = \sum_i w_{i0d_i} + \sum_{\substack{i \text{ s.t.}\\ q_i>0}} \left(w_{iq_i0} - w_{i00}\right) + \sum_{\substack{i \text{ s.t.}\\ q_i>0\\ d_i>0}} \left(w_{iq_id_i} - w_{iq_i0} - w_{i0d_i} + w_{i00}\right) = s(0,d) + \bar{s}(q,0) + \sum_{\substack{i \text{ s.t.}\\ q_i>0\\ d_i>0}} \overline{w}_{iq_id_i}, \quad (12)$$

[2]Two bins b_s and b_t are adjacent if they contain indices $(j,l) \in b_s$ and $(j',l') \in b_t$ such that $|j-j'|+|l-l'| \leq 1$.

where $s(0,d) = \sum_i w_{i0d_i}$ represents the similarity with the empty query and can be pre-computed offline for the database images, $\bar{s}(q,0) = \sum_{i \text{ s.t. } q_i>0} (w_{iq_i0} - w_{i00})$ depends only on the query (thus can be ignored for ranking the database images) and finally $\overline{w}_{ijl} = w_{ijl} - w_{ij0} - w_{i0l} + w_{i00}$ are the only terms that need to be explicitly computed for a query. Since they sum only over visual words that are present in both the query and the document, Eq. (12) has exactly the form needed to benefit from using an inverted file structure [29].

At query time, the pre-computed vector with elements $S_d = s(0,d)$ is loaded, then for each visual word i occurring in q, the inverted file structure is used to add $\overline{w}_{iq_id_i}$ to S_d for documents that also contain i. Finally, the documents are sorted by $S_d = s(q,d)$.[3]

4 Experiments

In this section, we present our experiments on four standard benchmark data sets for retrieval. We first present them below in Sec. 4.1. We study the different parameters of our approach in Sec. 4.2 and then compare it to the state of the art in Sec. 4.3.

4.1 Dataset, features, evaluation protocol, and implementation details

We evaluated our method on the standard Oxford5k [3, 21], Oxford105k [21], University of Kentucky (UKbench) [2, 19] and the INRIA Holidays [1, 14] benchmark datasets.
Oxford5k consists of 5,062 images from Flickr with Oxford landmarks. It comes with ground-truth for 55 query images with 5 query images corresponding to a same landmark.
Oxford105k contains Oxford5k and 100,000 distractor images. The distractor images are downloaded from Flickr with the same resolution as the original Oxford5k images.
UKbench consists of 10,200 images of 2,550 objects, *i.e.* there are 4 images per object.
Holidays contains 500 image groups and 1,491 images in total. Each group represents a distinct scene or object. The first image of each group is the query image and the correct retrieval results are the other images of the group.

On those datasets, we computed RootSIFT [5] descriptors on improved Hessian Affine interest points [20]. Using approximate k-means [21], we trained a visual vocabulary of 500,000 visual words for each of Oxford5k, UKbench and Holidays.

As our method requires data to learn its parameters, we generate training and test splits of the datasets by randomly dividing them into two halves. During the process, we ensure that relevant queries and documents remain in the same split. Hence, we guarantee that the system will not see at test time any image of any object used to learn its parameters. To prevent results from depending on the actual random choice of split, we generate 10 splits for each dataset and report the average performance. We refer to this protocol as Oxford5k^s, Oxford105k^s, Holidays^s and UKbench^s, resp. To measure performance, we use mean average precision (*mAP*), except for UKbench where we use the top-4 score [19].

For our approach, three parameters have to be set. The histogram truncation n is set by ensuring that less than 10% of the visual words occur more than n times in any image. This results in $n = 2$ for Oxford and $n = 4$ for Holidays and UKbench. For N_v and N_w, we resort to cross-validation. For all choices of N_v between 1 and 10, we consider all possible numbers of weight patterns. Those are limited since the truncation leaves only few weights. We then select the combination of N_v and N_w that maximizes cross-validation accuracy.

[3]Actually, $[s(0,d)]_{d\in\mathcal{D}}$ can also be sorted offline and an adaptive sort used to just update the ranking.

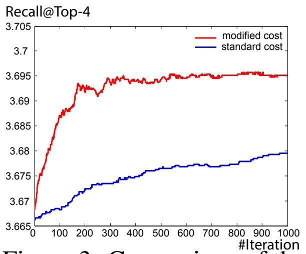

Figure 3: Comparison of the performance of learning-to-rank formulations.

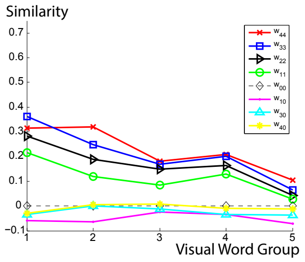

Figure 4: Weights learnt for UKbench (pattern shown in Fig. 2) for each of the $N_v = 5$ word group.

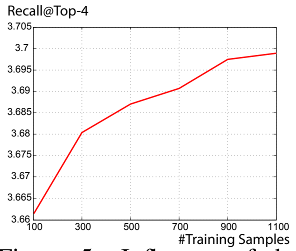

Figure 5: Influence of the number of training samples on the top-4 score.

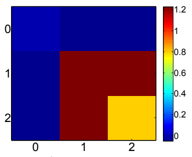

Figure 6: Learnt weight pattern for Oxford105k^s ($N_v = 5, N_w = 5$)

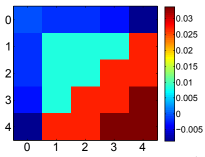

Figure 7: Learnt weight pattern for Holidays^s ($N_v = 2, N_w = 6$, sym.)

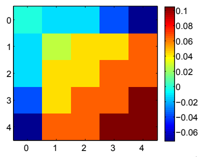

Figure 8: Learnt weight pattern for UKbench^s ($N_v = 5, N_w = 8$, sym.)

4.2 Qualitative analysis of our approach

In this section, we study the following four components of our approach: the choice of optimization problem (Sec. 3.2), the grouping of visual words and patterns (Sec. 3.3) and the size of the training set. For this analysis, we use the UKbench dataset.

Problem formulation. We start by comparing the influence of problem formulation on the performance of the system. Fig. 3 shows the top-4 score on the test set as a function of the training iterations for the standard rank-SVM cost [16], Eq. (9), and our modified cost, Eq. (11). As one can see, the performance of our proposed formulation converges faster (in about 200 iterations) and yields better accuracy.

Learnt weights. Fig. 4 shows the weights learnt for each of the $N_v = 5$ visual word group, using the pattern shown in Fig. 2, and normalized such that $w_{00} = 0$. We make three main observations. First, in each group, the weights on the diagonal (*i.e.*, w_{11} to w_{44}) are larger than w_{00} while most of the weights corresponding to mutual exclusion are smaller. This shows that co-missing evidence is indeed learnt. Second, those diagonal weights also show a diminishing return as the word frequency increases. This is consistent with burstiness modelling. Third, when comparing the different visual word groups, the corresponding weights tend to decrease as we move to more frequent visual words. This is a property of idf weighting.

Number of training samples. Fig. 5 shows the top-4 score with weights learnt from varying numbers of training samples. As we see, the performance of our model increases with more training data, and does not reach a plateau. This highlights the flexibility of our model and the interest of creating larger training datasets in the future to further improve performance.

Weight patterns. Finally, Figures 6, 7 and 8 show the weight patterns that we learnt for Oxford, Holidays and UKBench datasets, respectively. Notably, the weights for Oxford are not enforced to be symmetric, as we observe $w_{01} = -0.0412$ and $w_{10} = -0.0493$.

	Oxford5k	Oxford105k	Holidays	UKbench
Cosine Similarity [9]	0.769	0.679	**0.848**	3.42
Burstiness Weighting [15]	0.777	0.703	0.847	**3.46**
Negative Evidence [13]	0.771	0.629	0.836	3.35
Adaptive Asymmetric Similarity [37]	**0.793**	**0.719**	0.783	3.30
Comparable literature	0.795[27]	0.723[20]	0.793[15]	3.50[24]

Table 1: Comparison of ad-hoc similarities on the original versions of the datasets (mAP or top-4 score depending on the dataset). In bold we show the best results: they are comparable with state-of-the-art results available in the literature for comparable approaches.

4.3 Comparison to the state of the art

In this section we compare our approach with ad-hoc similarities used in state-of-the-art methods and then report performance on the datasets with train-test splits.

Similarities in the state of the art. For this experiment, we have implemented several similarities used in state-of-the-art approaches (*c.f*. Sec. 2): a) Cosine similarity [9]; b) Burstiness weighting [15]; c) Negative evidence [13]; and d) Adaptive asymmetric similarity [37]. Tab. 1 shows their performances on the original datasets using the same system (as described in Sec. 4.1) for fair comparison. We observe in Tab. 1 that the best similarity to use depends on the dataset. The asymmetric assumption fits Oxford well, whereas Cosine and Burstiness are better similarities for Holidays and UKbench. Importantly, these numbers are comparable with similar state-of-the-art systems in the literature (*e.g*., single-assignment BoW on Hessian Affine points, without post-processing), so we can confidently use them on our train-test splits as strong competitors.

Evaluation on train-test splits. We can now compare the same state-of-the-art similarities to our approach on the train-test splits of the benchmark datasets. Tab. 2 summarizes the performances over 10 such splits as described in Sec. 4.1. As each split is smaller than the original database – thus making each search task slightly easier –, the reported performances are slightly increased. Still, the relative ranking of the ad-hoc similarities [9, 13, 15, 37] is consistent with Tab. 1 on the original datasets.

From Tab. 2, we make the following observations. Clearly, our method consistently and significantly improves over all other similarities on all datasets. Over the 40 total runs (10 splits, 4 datasets), our method does not achieve the best performance for only 1 run. This is true even when the number of training samples is small: with only about 30 queries per split for Oxford5k and Oxford105k, we still obtain $+3\%$ and $+6\%$ improvement, resp., over the best competitor [37]. We also show improvements on Holiday^s ($+1\%$) and UKbench^s ($+0.16$, *i.e.* $+4\%$, relatively). This shows that our model is very general and very flexible, and can adapt to the specificities of each dataset, and that our contributions in Sec. 3.3 indeed allow to learn robust weights and prevent overfitting.

These results are also very promising considering that our approach has the potential to exploit more training data (*c.f*. Fig. 5) and that the procedures for identifying visual word groups and weight patterns have room for improvement. Despite these limitations, we report state-of-the-art results for 4 different datasets without the burden of hand-crafting similarity functions to adapt to the specificities of the data. Moreover, our system can seamlessly benefit from improvements in interest point detection, local descriptors, visual vocabulary learning and post-processing techniques such as geometrical verification. Such techniques typically bring improvements in the order of $+10\%$ [27].

	Oxford5k^s	Oxford105k^s	Holidays^s	UKbench^s
Cosine Similarity [9]	0.819 (0)	0.725 (0)	0.862 (0)	3.51 (0)
Burstiness Weighting [15]	0.826 (0)	0.748 (0)	0.858 (0)	3.54 (0)
Negative Evidence [13]	0.830 (0)	0.684 (0)	0.848 (0)	3.44 (0)
Adaptive Asymmetric Similarity [37]	0.839 (1)	0.758 (0)	0.795 (0)	3.38 (0)
Learnt Histogram Similarity (this paper)	**0.870 (9)**	**0.816 (10)**	**0.871 (10)**	**3.70 (10)**

Table 2: Comparison to alternative similarities. We report the average performance over the 10 splits of the data (mAP or top-4 score depending on the dataset) and in parenthesis the number of runs where the method is the best. In bold is the best result for each dataset.

5 Conclusion

In this paper, we have presented a novel framework to directly learn the visual similarity that maximizes the accuracy of an object retrieval system. Our model is very flexible and allows us to seamlessly integrate statistical properties of BoW histograms without modelling them explicitly. In our experiments, we have shown the superiority of our similarities compared to those used in state-of-the-art approaches.

Acknowledgements

We gratefully acknowledge the support of Swiss CTI project 15769.1 PFES-ES.

References

[1] http://lear.inrialpes.fr/~jegou/data.php.

[2] http://www.vis.uky.edu/~stewe/ukbench/.

[3] http://www.robots.ox.ac.uk/~vgg/data/oxbuildings/index.html.

[4] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building rome in a day. *Communications of the ACM*, 54(10):105–112, 2011.

[5] Relja Arandjelovic and Andrew Zisserman. Three things everyone should know to improve object retrieval. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2911–2918. IEEE, 2012.

[6] Yunbo Cao, Jun Xu, Tie-Yan Liu, Hang Li, Yalou Huang, and Hsiao-Wuen Hon. Adapting ranking svm to document retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 186–193. ACM, 2006.

[7] Ondrej Chum, James Philbin, Josef Sivic, Michael Isard, and Andrew Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[8] Ondrej Chum, James Philbin, and Andrew Zisserman. Near duplicate image detection: min-hash and tf-idf weighting. In *BMVC*, 2008.

[9] Ondrej Chum, Andrej Mikulik, Michal Perdoch, and Jiri Matas. Total recall ii: Query expansion revisited. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 889–896. IEEE, 2011.

[10] Stephan Gammeter, Lukas Bossard, Till Quack, and Luc Van Gool. I know what you did last summer: object-level auto-annotation of holiday snaps. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 614–621. IEEE, 2009.

[11] Ke Gao, Yongdong Zhang, Ping Luo, Wei Zhang, Junhai Xia, and Shouxun Lin. Visual stem mapping and geometric tense coding for augmented visual vocabulary. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3234–3241. IEEE, 2012.

[12] Bernd Girod, Vijay Chandrasekhar, David M Chen, Ngai-Man Cheung, Radek Grzeszczuk, Yuriy Reznik, Gabriel Takacs, Sam S Tsai, and Ramakrishna Vedantham. Mobile visual search. *Signal Processing Magazine, IEEE*, 28(4):61–76, 2011.

[13] Hervé Jégou and Ondřej Chum. Negative evidences and co-occurences in image retrieval: The benefit of pca and whitening. In *Computer Vision–ECCV 2012*, pages 774–787. Springer, 2012.

[14] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *ECCV08*, 2008.

[15] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. On the burstiness of visual elements. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1169–1176. IEEE, 2009.

[16] Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142. ACM, 2002.

[17] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[18] Andrej Mikulík, Michal Perdoch, Ondřej Chum, and Jiří Matas. Learning a fine vocabulary. In *Computer Vision–ECCV 2010*, pages 1–14. Springer, 2010.

[19] David Nistér and Henrik Stewénius. Scalable recognition with a vocabulary tree. In *CVPR*, 2006.

[20] Michal Perdoch, Ondrej Chum, and Jiri Matas. Efficient representation of local geometry for large scale object retrieval. In *Computer Vision and Pattern Recognition (CVPR) Conference*, 2009.

[21] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[22] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[23] James Philbin, Michael Isard, Josef Sivic, and Andrew Zisserman. Descriptor learning for efficient retrieval. In *Computer Vision–ECCV 2010*, pages 677–691. Springer, 2010.

[24] Danfeng Qin, Stephan Gammeter, Lukas Bossard, Till Quack, and Luc Van Gool. Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 777–784. IEEE, 2011.

[25] Grant Schindler, Matthew Brown, and Richard Szeliski. City-scale location recognition. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–7. IEEE, 2007.

[26] Xiaohui Shen, Zhe Lin, Jonathan Brandt, Shai Avidan, and Ying Wu. Object retrieval and localization with spatially-constrained similarity measure and k-nn re-ranking. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3013–3020. IEEE, 2012.

[27] K. Simonyan, A. Vedaldi, and A. Zisserman. Learning local feature descriptors using convex optimisation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014.

[28] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1470–1477. IEEE, 2003.

[29] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1470–1477. IEEE, 2003.

[30] Yu Su and Frédéric Jurie. Visual word disambiguation by semantic contexts. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 311–318. IEEE, 2011.

[31] Christian Wengert, Matthijs Douze, and Hervé Jégou. Bag-of-colors for improved image search. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 1437–1440. ACM, 2011.

[32] Zhong Wu, Qifa Ke, Michael Isard, and Jian Sun. Bundling features for large scale partial-duplicate web image search. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 25–32. IEEE, 2009.

[33] Shaoting Zhang, Ming Yang, Timothee Cour, Kai Yu, and Dimitris N Metaxas. Query specific fusion for image retrieval. In *Computer Vision–ECCV 2012*, pages 660–673. Springer, 2012.

[34] Yimeng Zhang, Zhaoyin Jia, and Tsuhan Chen. Image retrieval with geometry-preserving visual phrases. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 809–816. IEEE, 2011.

[35] Liang Zheng, Shengjin Wang, Ziqiong Liu, and Qi Tian. Lp-norm idf for large scale image search. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1626–1633. IEEE, 2013.

[36] Liang Zheng, Shengjin Wang, Ziqiong Liu, and Qi Tian. Packing and padding: Coupled multi-index for accurate image retrieval. *arXiv preprint arXiv:1402.2681*, 2014.

[37] Cai-Zhi Zhu, Hervé Jégou, and Shin'ichi Satoh. Query-adaptive asymmetrical dissimilarities for visual object retrieval. In *ICCV-International Conference on Computer Vision*, 2013.