Using Apache Lucene to Search Vector of Locally Aggregated Descriptors

Giuseppe Amato, Paolo Bolettieri, Fabrizio Falchi, Claudio Gennaro, Lucia Vadicamo
ISTI-CNR, Via G. Moruzzi, 1, 56124, Pisa, Italy
{*giuseppe.amato, paolo.bolettieri, fabrizio.falchi, claudio.gennaro, lucia.vadicamo*}*@isti.cnr.it*

arXiv:1604.05576v1 [cs.CV] 19 Apr 2016

Keywords: Bag of Features, Bag of Words, Local Features, Compact Codes, Image Retrieval, Vector of Locally Aggregated Descriptors

Abstract: Surrogate Text Representation (STR) is a profitable solution to efficient similarity search on metric space using conventional text search engines, such as Apache Lucene. This technique is based on comparing the permutations of some reference objects in place of the original metric distance. However, the Achilles heel of STR approach is the need to reorder the result set of the search according to the metric distance. This forces to use a support database to store the original objects, which requires efficient random I/O on a fast secondary memory (such as flash-based storages). In this paper, we propose to extend the Surrogate Text Representation to specifically address a class of visual metric objects known as Vector of Locally Aggregated Descriptors (VLAD). This approach is based on representing the individual sub-vectors forming the VLAD vector with the STR, providing a finer representation of the vector and enabling us to get rid of the reordering phase. The experiments on a publicly available dataset show that the extended STR outperforms the baseline STR achieving satisfactory performance near to the one obtained with the original VLAD vectors.

1 Introduction

Multimedia information retrieval on a large scale database has to address at the same time both issues related to effectiveness and efficiency. Search results should be pertinent to the submitted queries, and should be obtained quickly, even in presence of very large multimedia archives and simultaneous query load.

Vectors of Locally Aggregated Descriptors (VLAD) (Jégou et al., 2012) were recently proposed as a way of producing compact representation of local visual descriptors, as for instance SIFT (Lowe, 2004), while still retaining high level of accuracy. In fact, experiments, demonstrated that VLAD accuracy is higher than Bag of Words (BoW) (Sivic and Zisserman, 2003). The advantage of BoW representation is that it is very sparse and allows using inverted files to also achieve high efficiency. VLAD representation is not sparse, so general indexing methods for similarity searching (Zezula et al., 2006) must be used, which are typically less efficient than inverted files.

One of the best performing generic methods for similarity searching, is the use of permutation based indexes (Chavez et al., 2008; Amato et al., 2014b). Permutation based indexes rely on the assumption that to objects that are very similar, "see" the space around them in a similar way. This assumption is exploited by representing the objects as the ordering of a fixed set of reference objects (or pivots), according to their distance from the objects themselves. If two objects are very similar, the two corresponding ordering of the reference objects will be similar as well.

However, measuring the similarity between objects using the similarity between permutations is a coarse approximation. In fact, in order to achieve also high accuracy, similarity between permutations is used just to identify an appropriate set of candidates, which is then reordered according to the original similarity function to obtain the final result. This reordering phase, contributes to the overall search cost.

Given that objects are represented as ordering (permutations) of reference objects, permutation based indexes offer the possibility of using inverted files, in every similarity searching problem, where distance functions are metric functions. In fact, (Gennaro et al., 2010) presents an approach where the Lucene text search engines, was used to index and retrieve objects by similarity. The technique is based on an encoding of the permutations by means of a *Surrogate Text Representation* (STR). In this respect, VLAD can be easily indexed using this technique, as

discussed in (Amato et al., 2014a) so that efficient and effective image search engines can be built on top of a standard text search engine.

In this paper, we propose an advancement on this basic techniques, which exploits the internal structure of VLAD. Specifically, the STR technique is applied, independently, to portions of the entire VLAD. This leads, at the same time, to higher efficiency and accuracy without the need of executing the reordering of the set of candidates, which was mentioned above. The final result is obtained by directly using the similarity between the permutations (the textual representation), so saving both time in the searching algorithms, and space, since the original VLAD vectors no longer need to be stored.

The paper is organized as follows. Section 2 makes a survey of the related works. Section 3 provides a brief introduction to the VLAD approach. Section 4 introduces the proposed approach. Section 5 discusses the validation tests. Section 6 concludes.

2 Related Work

In the last two decades, the breakthroughs in the field of image retrieval have been mainly based on the use of the local features. Local features, as SIFT (Lowe, 2004) and SURF (Bay et al., 2006), are visual descriptors of selected interest points of an image. Their use allows one to effectively match local structures between images. However, the costs of comparison of the local features lay some limits on large scale, since each image is represented by typically thousands of local descriptors. Therefore, various methods for the aggregation of local features have been proposed.

One of the most popular aggregation method is the *Bag-of-Word* (BoW), initially proposed in (Sivic and Zisserman, 2003; Csurka et al., 2004) for matching object in videos. BoW uses a *visual vocabulary* to quantize the local descriptors extracted from images; each image is then represented by a histogram of occurrences of visual words. The BoW approach used in computer vision is very similar to the BoW used in natural language processing and information retrieval (Salton and McGill, 1986), thus many text indexing techniques, as inverted files (Witten et al., 1999), have been applied for image search. From the very beginning (Sivic and Zisserman, 2003) words reductions techniques have been used and images have been ranked using the standard *term frequency-inverse document frequency* (tf-idf) (Salton and McGill, 1986) weighting. In order to improve the efficiency of BoW, several approaches for the reduction of visual words have been investigated (Thomee et al., 2010; Amato et al., 2013b). Search results obtained using BoW in CBIR (Content Based Image Retrieval) has also been improved by exploiting additional geometrical information (Philbin et al., 2007; Perd'och et al., 2009; Tolias and Avrithis, 2011; Zhao et al., 2013) and applying re-ranking approaches (Philbin et al., 2007; Jégou et al., 2008; Chum et al., 2007; Tolias and Jégou, 2013). The baseline BoW encoding is affected by the loss of information about the original descriptors due to the quantization process. For example, corresponding descriptors in two images may be assigned to different visual words. To overcome the quantization loss, more accurate representation of the original descriptors and alternative encoding techniques have been used, such as *Hamming Embedding* (Jégou et al., 2008; Jégou et al., 2010), *soft-assignment* (Philbin et al., 2008; Van Gemert et al., 2008; Van Gemert et al., 2010), *multiple assignment* (Jégou et al., 2010; Jégou et al., 2010b), *locality-constrained linear coding* (Wang et al., 2010), *sparse coding* (Yang et al., 2009; Boureau et al., 2010) and the use of *spatial pyramids* (Lazebnik et al., 2006).

Recently, other aggregation schemes, such as the *Fisher Vector* (FV) (Perronnin and Dance, 2007; Jaakkola and Haussler, 1998) and the *Vector of Locally Aggregated Descriptors* (VLAD) (Jégou et al., 2010a), have attracted much attention because of their effectiveness in both image classification and large-scale image search. Both FV and VLAD use some statistics about the distribution of the local descriptors in order to transform an incoming set of descriptors into a fixed-size vector representation.

The basic idea of FV is to characterize how a sample of descriptors deviates from an average distribution that is modeled by a parametric generative model. The Gaussian Mixture Model (GMM) (McLachlan and Peel, 2000), estimated on a training set, is typically used as generative model and might be understood as a "probabilistic visual vocabulary".

While BoW counts the occurrences of visual words and so takes in account just 0-order statistics, the VLAD approach, similarly to BoW, uses a visual vocabulary to quantize the local descriptors of an image. The visual vocabulary is learned using a clustering algorithm, as for example the k-means. Compared to BOW, VLAD exploits more aspects of the distribution of the descriptors assigned to a visual word. In fact, VLAD encodes the accumulated difference between the visual words and the associated descriptors, rather than just the number of descriptors assigned to each visual word. As common post-processing step VLAD is power and L2 normalized (Jégou et al., 2012; Perronnin et al., 2010). Furthermore, PCA dimensionality reduction and product quantization have

been applied and several enhancements to the basic VLAD have been proposed (Arandjelovic and Zisserman, 2013; Perronnin et al., 2010; Chen et al., 2011; Delhumeau et al., 2013; Zhao et al., 2013)

In this work, we will focus on VLAD which is very similar to FV. In fact VLAD has been proved to be a simplified non-probabilistic version of FV that performs very similar to FV (Jégou et al., 2012). However, while BoW is a sparse vector of occurrence, VLAD is not. Thus, inverted files cannot be directly applied for indexing and Euclidean Locality-Sensitive Hashing (Datar et al., 2004) is, as far as we know, the only technique tested with VLAD. Many other similarity search indexing techniques (Zezula et al., 2006) could be applied to VLAD. A very promising direction is Permutation-Based Indexing (Chavez et al., 2008; Amato et al., 2014b; Esuli, 2009). In particular the MI-File allows one to use inverted files to perform similarity search with an arbitrary similarity function. Moreover, in (Gennaro et al., 2010; Amato et al., 2011) a Surrogate Text Representation (STR) derived from the MI-File has been proposed. The conversion of the image description in textual form enables us to exploit the off-the-shelf search engine features with a little implementation effort.

In this paper, we extend the STR approach to deal with the VLAD descriptions comparing both effectiveness and efficiency with the STR baseline approach, which has been studied in (Amato et al., 2013a). The experimentation was carried out on the same hardware and software infrastructure using a publicly available INRIA Holidays (Jégou et al., 2008) dataset and comparing the effectiveness with the sequential scan.

3 Vector of Locally Aggregated Descriptors (VLAD)

The VLAD representation was proposed in (Jégou et al., 2010). As for the BoW, a visual vocabulary, here called *codebook*, $\{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{\mathcal{K}}\}$[1] is first learned using a cluster algorithm (e.g. *k*-means). Each local descriptor $\mathbf{x}_t$ is then associated with its nearest visual word (or *codeword*) $NN(\mathbf{x}_t)$ in the codebook. For each codeword the differences between the sub-vectors $\mathbf{x}_t$ assigned to $\boldsymbol{\mu}_i$ are accumulated:

$$\mathbf{v}_i = \sum_{\mathbf{x}_t : NN(\mathbf{x}_t) = i} \mathbf{x}_t - \boldsymbol{\mu}_i$$

The VLAD is the concatenation of the accumulated sub-vectors, i.e. $\mathbf{V} = (\mathbf{v}_1, \ldots, \mathbf{v}_{\mathcal{K}})$. Throughout the paper, we refer to the accumulated sub-vectors $\mathbf{v}_i$ simply as "sub-vectors".

Two normalization are performed: first, a power normalization with power 0.5; second, a L2 normalization. After this process two descriptions can be compared using the inner product.

The observation that descriptors are relatively sparse and very structured suggests performing a principal component analysis (PCA) to reduce the dimensionality of the VLAD. In this work, we decide not to use dimensionality reduction techniques because we will show that our space transformation approach is independent from the original dimensionality of the description. In fact, the STR approach that we propose, transforms the VLAD description in a set of words from a vocabulary that is independent from the original VLAD dimensionality.

4 Surrogate Text Representation for VLAD Vectors

In this paper, we propose to index VLAD using a text encoding that allows using any text retrieval engine to perform image similarity search. As discussed later, we implemented this idea on top of the Lucene text retrieval engine[2].

To this end, we extend the permutation-based approach developed by Chavez et al. (Chavez et al., 2008) to deal with the internal representation of the VLAD vectors. In this section, we first introduce the basic principle of the permutation-based approach and then describe the generalization to VLAD vectors.

4.1 Baseline Permutation-based Approach and Surrogate Text Descriptor

The key idea of the Permutation-based approach relies on the observation that if two objects are near one another, they have a similar view of the objects around them. This means that the orderings (permutations) of the surrounding objects, according to the distances from the two objects, should be similar as well.

Let $\mathcal{D}$ be a domain of objects (features, points, etc.), and $d : \mathcal{D} \times \mathcal{D} \rightarrow \mathbb{R}$ a distance function able to assess the dissimilarity between two objects of $\mathcal{D}$. Let $R \subset \mathcal{D}$, be a set of *m distinct* objects (reference objects), i.e., $R = \{r_1, \ldots, r_m\}$. Given any object $o \in \mathcal{D}$, we denote the vector of rank positions of the reference

[1] Throughout the paper bold letters denote row vectors.

[2] http://lucene.apache.org

objects, ordered by increasing distance from o, as $\mathbf{p}(o) = (p_1(o), \ldots, p_m(o))$. For instance, if $p_3(o) = 2$ then r_3 is the 2nd nearest object to o among those in R. The essence of the permutation-based approach is to allow executing similarity searching exploiting distances between permutations in place of original objects' distance. This, as discussed in the following, has the advantage of allowing using a standard text retrieval engine to execute similarity searching.

There are several standard methods for comparing two ordered lists, such as Kendall's tau distance, Spearman Footrule distance, and Spearman Rho distance. In this paper, we concentrate our attention on the latter distance, which is also used in (Chavez et al., 2008). The reason of this choice (explained later on) is tied to the way standard search engines process the similarity between documents and query.

In particular, we exploit a generalization of the Spearman Rho distance that allows us to compare two top-k ranked lists. Top-k list is a particular case of a partial ranked list, which is a list that contains rankings for only a subset of items. For top-k lists, we can use a generalization of the Spearman Rho distance $\tilde{d}(o,q)$, called *location parameter distance* (Fagin et al., 2003), which assigns a rank $k+1$ for all items of the list that have rank greater than k.

In particular, let k be an integer less or equal than m, and $\mathbf{p}^k(o) = (p_1^k(o), \ldots, p_m^k(o))$ the vector defined as follows:

$$p_i^k(o) = \begin{cases} p_i(o) \text{ if } p_i(o) \leq k \\ k+1 \text{ if } p_i(o) > k \end{cases}. \quad (1)$$

Given two top-k ranked lists with $k = k_q$ and $k = k_x$, we define the approximate distance function $\tilde{d}(o,q)$ as follows:

$$\tilde{d}(o,q) = ||\mathbf{p}^{k_x}(o) - \mathbf{p}^{k_q}(q)||_2, \quad (2)$$

where k_q is used for queries and k_x for indexing. The reason for using two different k relies on the fact the performance of the inverted files is optimal when the size of the queries are much smaller than the size of documents. Therefore, we will typically require that $k_q \leq k_x$.

Since, the square root in Eq. (2) is monotonous, it does not affect the ordering (Fagin et al., 2003), so we can safely use $\tilde{d}(o,q)^2$ instead of its square-root counterpart:

$$\begin{aligned} &\tilde{d}(o,q)^2 = \textstyle\sum_{i=1}^m \left(p_i^{k_x}(o) - p_i^{k_q}(q)\right)^2 = \\ &||\mathbf{p}^{k_x}(o)||_2^2 + ||\mathbf{p}^{k_q}(q)||_2^2 - 2\,\mathbf{p}^{k_x}(o) \cdot \mathbf{p}^{k_q}(q) \end{aligned} \quad (3)$$

Figure 1 exemplifies the transformation process. Figure 1a sketches a number of reference objects (black points), objects (white points), and a query object (gray point). Figure 1b shows the encoding of the data objects in the transformed space. We will use this illustration as a running example throughout the remainder of the paper.

So far, we have presented a method for approximating the function d. However, our primary objective is to implement the function $\tilde{d}(o,q)$ in an efficient way by exploiting the built-in cosine similarity measure of standard text-based search engines based on vector space model. For this purpose, we associate each element $r_i \in R$ with a unique key τ_i. The set of keys $\{\tau_1, \ldots, \tau_m\}$ represents our so-called "reference-dictionary". Then, we define a function $t^k(o)$ that returns a space-separated concatenation of zero or more repetitions of τ_i keywords, as follows: $t^k(o) = \bigcup_{i=1}^{m} \bigcup_{j=1}^{k+1-p_i^k(o)} \tau_i$ where, by abuse of notation, we denote the space-separated concatenation of keywords with the union operator $\cup$. The function $t^k(o)$ is used to generate the Surrogate Text Representation for both indexing and querying purposes. k assumes in general the values k_x for indexing and k_q for querying. For instance, consider the case exemplified in Figure 1c, and let us assume τ_1 =A, τ_2 =B, etc. The function t^k with $k_x = 3$ and $k_q = 2$, will generate the following outputs

$$\begin{aligned} t^{k_x}(o_1) &= \text{“E E E B B A”} \\ t^{k_x}(o_2) &= \text{“D D D C C E”} \\ t^{k_q}(q) &= \text{“E E A”} \end{aligned}$$

As can be seen intuitively, strings corresponding to o_1 and q are more similar to those corresponding to o_2 e q, this reflects the behavior of the distance $\tilde{d}$. However, this should not mislead the reader: our proposal is not a heuristic, the distance between the strings corresponds exactly to the distance $\tilde{d}$ between the objects, as we will prove below.

As explained above, the objective now is to force a standard text-based search engine to generate the approximate distance function $\tilde{d}$. How this objective is obtained becomes obvious by the following considerations. A text based search engine will generate a vector representation of STRs generated with $t^{k_x}(o)$ and $t^{k_q}(q)$ containing the number of occurrences of words in texts. This is the case of the simple term-frequency weighting scheme. This means that, if for instance keyword τ_i corresponding to the reference object r_i ($1 \leq i \leq m$) appears n times, the i-th element of the vector will assume the value n, and whenever τ_i does not appear it will be 0. Let $\mathbf{k_x}$ and $\mathbf{k_q}$ be respectively the constant m-dimensional vectors, $(k_x+1, \ldots, k_x+1)$ and $(k_q+1, \ldots, k_q+1)$, then

$$\begin{aligned} \widehat{\mathbf{p}}^{k_x}(o) &= \mathbf{k_x} - \mathbf{p}^{k_x}(o) \\ \widehat{\mathbf{p}}^{k_q}(q) &= \mathbf{k_q} - \mathbf{p}^{k_q}(q) \end{aligned} \quad (4)$$

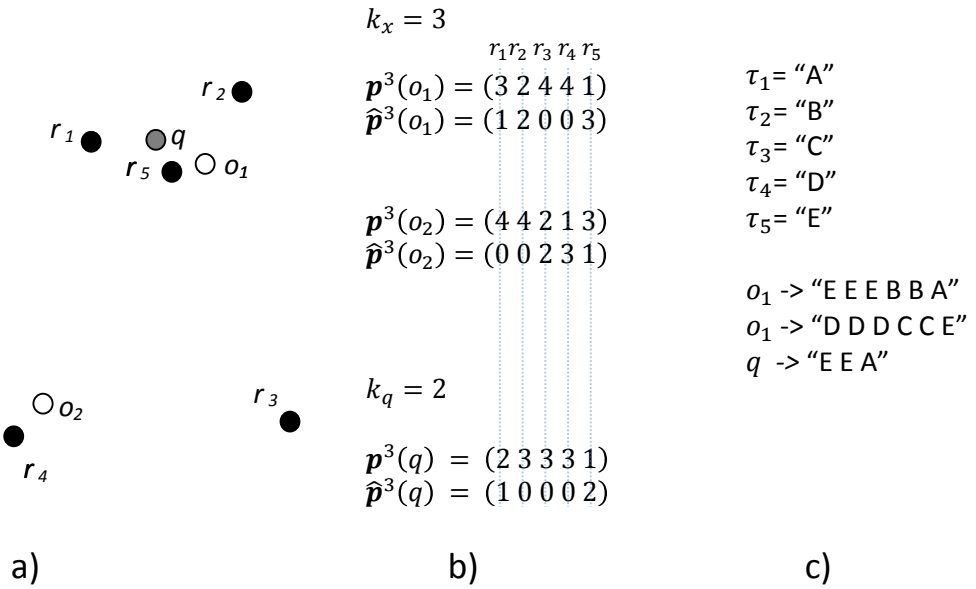

Figure 1: Example of perspective based space transformation. a) Black points are reference objects; white points are data objects; the grey point is a query. b) Encoding of the data objects in the transformed space. c) Encoding of the data objects in textual form.

It is easy to see that the vectors corresponding to $t^{k_x}(o)$ and $t^{k_q}(q)$, are the same of $\widehat{\mathbf{p}}^{k_x}(o)$ and $\widehat{\mathbf{p}}^{k_q}(q)$, respectively.

The cosine similarity is typically adopted to determine the similarity of the query vector and a vector in the database of the search engine, and it is defined as:

$$sim_{cos}(o,q) = \frac{\widehat{\mathbf{p}}^{k_x}(o)\cdot\widehat{\mathbf{p}}^{k_q}(q)}{\|\widehat{\mathbf{p}}^{k_x}(o)\| \ \ \|\widehat{\mathbf{p}}^{k_q}(q)\|} \propto \widehat{\mathbf{p}}^{k_x}(o)\cdot\widehat{\mathbf{p}}^{k_q}(q). \tag{5}$$

It is worth noting that $\widehat{\mathbf{p}}^k$ is a permutation of the m-dimensional vector $(1,2,\ldots,k,0,\ldots,0)$, thus its norm equals $\sqrt{k(k+1)(2k+1)/6}$. Since k_x and k_q are constants, the norms of vectors $\widehat{\mathbf{p}}^{k_x}$ and $\widehat{\mathbf{p}}^{k_q}$ are constants too, therefore can be neglected during the cosine evaluation (they do not affect the final ranking of the search result).

What we are now to show is that sim_{cos} can be used as a function for evaluating a similarity of two objects in place of the distance $\tilde{d}$ and it possible to prove that the first one is a order reversing monotonic transformation of the second one (they are equivalent for practical aspects). This means that if we use $\tilde{d}(o,q)$ and we take the first k nearest objects from a dataset $X \subset \mathcal{D}$ (i.e, from the shortest distance to the highest) we obtain exactly the same objects in the same order if we use $sim_{cos}(o,q)$ and take the first k similar objects (i.e., from the greater values to the smaller ones).

By substituting Eq. (4) into Eq. (5), we obtain:

$$sim_{cos}(o,q) \propto (\mathbf{k_x} - \mathbf{p}^{k_x}(o))\cdot(\mathbf{k_q} - \mathbf{p}^{k_q}(q)) =$$
$$= \mathbf{k_x}\cdot\mathbf{k_q} - \mathbf{k_x}\cdot\mathbf{p}^{k_q}(q) - \mathbf{k_q}\cdot\mathbf{p}^{k_x}(o) + \mathbf{p}^{k_x}(o)\cdot\mathbf{p}^{k_q}(q) \tag{6}$$

since $\mathbf{p}^{k_x}(o)$ ($\mathbf{p}^{k_q}(q)$) include all integers numbers from 1 to k_x (k_q) and the remaining assumes k_x+1 (k_q+1) values, the scalar product $\mathbf{k_x}\cdot\mathbf{p}^{k_q}(q)$ ($\mathbf{k_q}\cdot\mathbf{p}^{k_x}(o)$) is constant. We can substitute the first three member in Eq. (6) with a constant $L(m,k_x,k_q)$, which depends only on m, k_x, and k_q as follows:

$$sim_{cos}(o,q) \propto L(m,k_x,k_q) + \mathbf{p}^{k_x}(o)\cdot\mathbf{p}^{k_q}(q). \tag{7}$$

Finally, combining Eq. (7) with Eq. (3), we obtain:

$$sim_{cos}(o,q) \propto L(m,k_x,k_q) + \tfrac{1}{2}||\mathbf{p}^{k_x}(o)||_2^2 + \\ + \tfrac{1}{2}||\mathbf{p}^{k_q}(q)||_2^2 - \tfrac{1}{2}\tilde{d}(o,q)^2. \tag{8}$$

Since $||\mathbf{p}^{k_x}(o)||$ and $||\mathbf{p}^{k_q}(q)||$ depend only on the constants m, k_x, and k_q, the Eq. (8) proves that $sim_{cos}(o,q)$ is a monotonic transformation of $\tilde{d}(o,q)^2$ in the form $sim_{cos} = \alpha - \beta\tilde{d}^2$.

To summarize, given a distance function d, we were able to determine an approximate distance function $\tilde{d}$, which we transformed in a similarity measure. We proved that this similarity measure can be obtained using the STR and that it is equivalent from the point of view of the result ranking to $\tilde{d}$.

Note, however, that searching using directly the distance from permutations suffers of low precision. To improve effectiveness, (Amato et al., 2014b) proposes to reorder the results set according to original distance function d. Suppose we are searching for the k most similar (nearest neighbors) descriptors to the query. The quality of the approximation is improved by reordering, using the original distance function d, the first c ($c \geq k$) descriptors from the approximate result set at the cost of c additional distance computations.

4.2 Blockwise Permutation-based Approach

The idea described so far uses a textual/permutation representation of the object as whole, however, in our particular scenario, we can exploit the fact that VLAD vector is the result of concatenation of sub-vectors. In short, we apply and compare the textual/permutation representation for each sub-vector $\mathbf{v}_i$ of the whole VLAD, independently. We refer to this approach as Blockwise Permutation-based approach.

As we will see, this approach has the advantage of providing a finer representation of objects, in terms of permutations, so that no reordering is needed to guarantee the quality of the search result.

In order to decrease the complexity of the approach and since sub-vectors $\mathbf{v}_i$ are homogeneous, we use the same set of reference objects $R = \{r_1, \ldots, r_m\}$ to represent them as permutations taken at random from the dataset of VLAD vectors. Let $\mathbf{v}_i$ be the i-st sub-vector of a VLAD sub-vector $\mathbf{V}$, we denote by $\mathbf{p}^{k_x}(\mathbf{v}_i)$ the corresponding permutation vector. Given two VLAD vectors $\mathbf{V} = (\mathbf{v}_1, \ldots, \mathbf{v}_\mathcal{K})$ and $\mathbf{W} = (\mathbf{w}_1, \ldots, \mathbf{w}_\mathcal{K})$, and their corresponding concatenated permutation vectors $\mathbf{O} = (\mathbf{p}^{k_x}(\mathbf{v}_1), \ldots, \mathbf{p}^{k_x}(\mathbf{v}_\mathcal{K}))$ and $\mathbf{Q} = (\mathbf{p}^{k_q}(w_1), \ldots, \mathbf{p}^{k_q}(w_\mathcal{K}))$, we generalize the Spearman Rho distance for two vectors $\mathbf{V}$ and $\mathbf{W}$ as follows:

$$\begin{aligned} &\tilde{d}(\mathbf{V},\mathbf{W})^2 = \textstyle\sum_{i=1}^{\mathcal{K}} \tilde{d}(\mathbf{v}_i,\mathbf{w}_i)^2 = \\ &\textstyle\sum_{i=1}^{\mathcal{K}} ||\mathbf{p}^{k_x}(\mathbf{v}_i) - \mathbf{p}^{k_q}(\mathbf{w}_i)||_2^2 = ||\mathbf{O}-\mathbf{Q}||_2^2 \end{aligned} \tag{9}$$

This generalization has the advantage of being faster to compute since it treats the concatenated permutation vector as a whole. Moreover, it does not require square roots and it can be evaluated using the cosine. Defining in the same way as above:

$$\begin{aligned} \widehat{\mathbf{p}}^{k_x}(\mathbf{v}_i) &= \mathbf{k_x} - \mathbf{p}^{k_x}(\mathbf{v}_i) \\ \widehat{\mathbf{p}}^{k_q}(\mathbf{w}_i) &= \mathbf{k_q} - \mathbf{p}^{k_q}(\mathbf{w}_i). \end{aligned} \tag{10}$$

By a similar procedure shown above, it is possible to prove that also in this case $sim_{cos}(\mathbf{V},\mathbf{W}) \propto \alpha - \beta \tilde{d}^2(\mathbf{V},\mathbf{W})$ holds.

In order to correctly match the transformed blockwise vectors, we need to extended the reference dictionary to distinguish the key produced from sub-vectors $\mathbf{v}_i$ with different subscript i. There for a set m of reference objects, and $\mathcal{K}$ element in the VLAD codebook, we employ dictionary including a set of $m \times \mathcal{K}$ keys $\tau_{i,j}$ ($1 \leq i \leq m$, $1 \leq j \leq \mathcal{K}$).

For example, we associate, say, the set of keys A_1, B_1,... to the sub-vector $\mathbf{v}_1$, A_2, B_2,... to the sub-vector $\mathbf{v}_2$, and so on.

4.3 Dealing with VLAD Ambiguities

One of the well-known problems of VLAD happens when no local descriptor is assigned to a codeword (Peng et al., 2014). A simple approach to this problem is produce a sub-vector of all zeros ($\mathbf{v}_i = \mathbf{0}$) but this has the disadvantage to be ambiguous since it is identical to the case in which the mean of the local descriptors assigned to a codeword is equal to the codeword itself.

Moreover, as pointed out by (Spyromitros-Xioufis et al., 2014), given two images and the corresponding VLAD vectors $\mathbf{V}$ and $\mathbf{W}$, and assuming that $\mathbf{v}_i = \mathbf{0}$, the contribution of codeword $\boldsymbol{\mu}_i$ to the cosine similarity of $\mathbf{V}$ and $\mathbf{W}$ will be the same when either $\mathbf{w}_i = \mathbf{0}$ or $\mathbf{w}_i \neq \mathbf{0}$. Therefore, this under-estimates the importance of jointly zero components, which gives some limited yet important evidence on visual similarity (Jégou and Chum, 2012). In (Jégou and Chum, 2012), this problem was treated by measuring the cosine between vectors $\mathbf{V}$ and $\mathbf{W}$ at different point from the origin.

This technique, however, did not lead to significant improvement of our experiments. To tackle this problem, we simply get rid of the sub-vectors $\mathbf{v}_i = \mathbf{0}$ and omit to transform them in text. Mathematically, this means that we assume $\widehat{\mathbf{p}}^{k_x}(\mathbf{0}) = \mathbf{0}$.

5 Experiments

5.1 Setup

INRIA Holidays (Jégou et al., 2010a; Jégou et al., 2012) is a collection of 1,491 holiday images. The authors selected 500 queries and for each of them a list of positive results. As in (Jégou et al., 2009; Jégou et al., 2010; Jégou et al., 2012), to evaluate the approaches on a large scale, we merged the Holidays dataset with the Flickr1M collection[3]. SIFT features

[3] http://press.liacs.nl/mirflickr/

have been extracted by Jegou et al. for both the Holidays and the Flickr1M datasets[4].

For representing the images using the VLAD approach, we selected 64 reference features using *k-means* over a subset of the Flickr1M dataset.

All experiments were conducted on a Intel Core i7 CPU, 2.67 GHz with 12.0 GB of RAM a 2TB 7200 RPM HD for the Lucene index. We used Lucene v4.7 running on Java 6 64 bit.

The quality of the retrieved images is typically evaluated by means of precision and recall measures. As in many other papers (Jégou et al., 2009; Perronnin et al., 2010; Jégou et al., 2012), we combined this information by means of the mean Average Precision (mAP), which represents the area below the precision and recall curve.

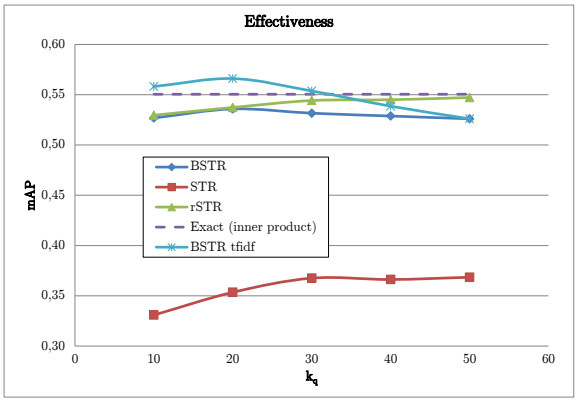

Figure 2: Effectiveness (mAP) of the various approach for the INRIA Holidays dataset, using $k_x = 50$ for STR, rSTR, BSTR, and BSTR tfidf (higher values mean better reults).

5.2 Results

In a first experimental analysis, we compared the performance of blockwise approach versus the baseline approach (with and without reordering) that threats the VLAD vectors as whole-objects, which was studied in (Amato et al., 2014a). In this latter approach, as explained Section 4, since the performance was low, we had to reorder the best results using the actual distance between the VLAD descriptors. With this experiment, we want to show that with the blockwise approach this phase is no longer necessary, and the search is only based on the result provided by text-search engine Lucene. For the baseline approach, we used m =4,000 reference objects while for blockwise, 20,000. In both cases, we set $k_x = 50$, which, we recall, is the number of closest reference objects used during indexing.

Figure 2 shows this comparison in terms of mAP. We refer to baseline approach as STR, the baseline approach with reordering as rSTR, and to blockwise approach as BSTR. For the rSTR approach, we reordered the first 1,000 objects of the results set. The horizontal line at the top represents the performance obtained matching the original VLAD descriptors with the inner product, performing a sequential scan of the dataset, which exhibits a mAP of 0.55. The graph in the middle shows the mAP of our approach (BSTR) versus the baseline approach without reordering (STR) and with reordering (rSTR). The graphs show also how the performance changes varying k_q (the number of closest reference objects for the query) from 10 to 50.

An interesting by-product of the experiment is that, we obtain a little improvement of the mAP for the BSTR approach when the number of reference objects used for the query is 20.

A quite intuitive way of generalizing the idea of reducing the size of the query is to exploit the knowledge of the *tf*idf* (i.e., term frequency * inverse document frequency) statistic of the BSTR textual representation. Instead of simply reducing the k_q of the query, i.e., the top-k_q element nearest to the query, we can retain the elements that exhibit greater values of *tf*idf* starting from the document generated with $k_q = 50$ and eliminate the others. Therefore, we take, for instance, the first 40 elements that have best *tf*idf*, the first 30 elements, and so on. Figure 2 shows the performance of this approach, with the name 'BSTR tfidf'. It is interesting to note that we had not only an important improvement of the mAP for increasing reduction of the queries but also that this approach outperforms the performance of the inner product on the original VLAD dataset.

In order to ascertain the soundness of the proposed approach, we tested it on the larger and challenger Flickr1M dataset.

The results are shown in Figure 3. We can see that BSTR tfidf is still the winner in terms of mAP. However, in this case all the techniques exhibit lower performance with respect the inner product on the original VLAD dataset. The latter test is performed as a sequential scan of the entire dataset obtaining a mAP of 0.34. The results presented in this figure also show the performance of the approach called BSTR tfidf[2], which consists in applying the reduction of the blockwise textual representation using *tf*idf* also for the indexed document (in addition to the queries), setting $k_x = k_q$ for all the experiments. The mAPs values in this case are slightly lower than BSTR tfidf, however, as we are going to see in the next experiment there is a great advance in terms of space occupation.

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

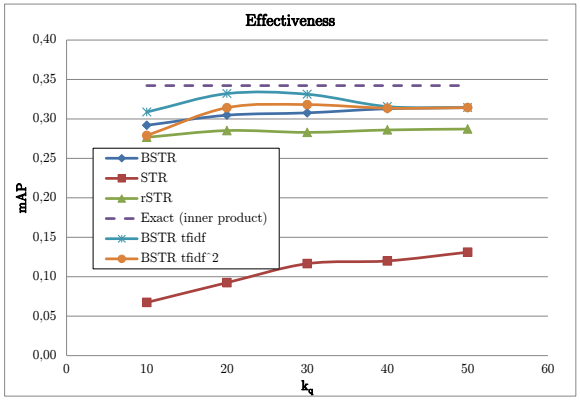

Figure 3: Effectiveness (mAP) of the various approach for the INRIA Holidays + Flickr1M dataset, using $k_x = 50$ for STR, rSTR, BSTR, and BSTR tfidf. While for BSTR tfidf^2, we set $k_x = k_q$ (higher values mean better reults).

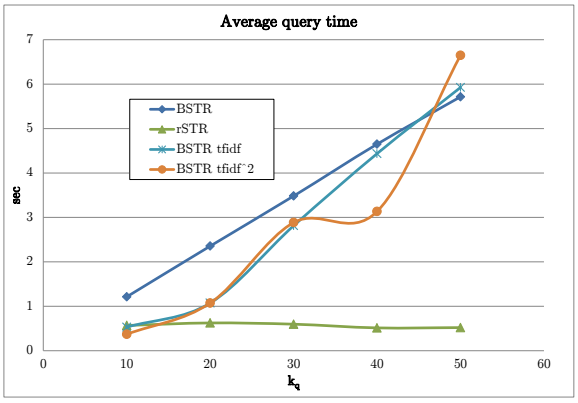

Figure 4: Average time per query in seconds of the various approaches for the INRIA Holidays + Flickr1M dataset, using $k_x = 50$ for rSTR, BSTR, and BSTR tfidf. While for BSTR tfidf^2, we set $k_x = k_q$ (higher values mean worse performance).

In order to assess which approach is most promising, we have also evaluated the efficiency in terms of space and time overhead. Figure 4 shows the average time for a query for the proposed approaches. The rSTR approach considers also the time for reordering the result set, however, its average time is obtained using a solid state disk (SSD) disk in which the original VLAD vectors are available for the reordering. The SSD is necessary to guarantee fast random I/O, while using a standard disk the seek time would affect the query time of more than one order of magnitude.

Figure 5 presents the index occupation expressed in GB. The rSTR approach occupies 16.8 GB on the disk, including the overhead for the storage of the VLAD vectors used for the reordering of the results. The BSTR tfidf^2 solution has great impact of the space occupation: just for a reduction of the 20% of the documents (i.e., from $k_x = 50$ to $k_x = 40$) we get a reduction of the 80% for the inverted file.

Considering all the alternatives seen so far, an optimal choice could be BSTR tfidf^2 with $k_x = k_q = 20$, which is efficient in term of both time and space overheads and still maintains satisfactory mAP.

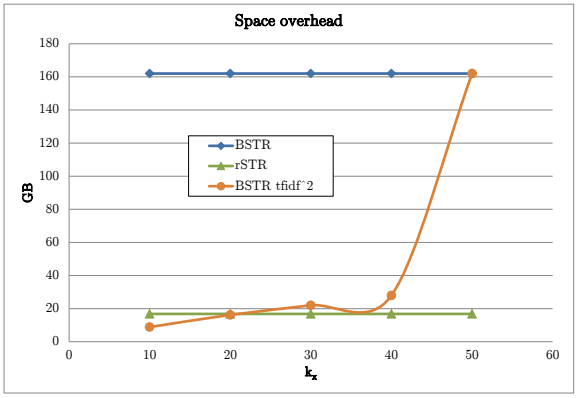

Figure 5: Space occupation of the index for the different type of solutions, using the same value of $k_x = 50$ for BSTR and rSTR, and varing k_x for BSTR tfidf^2. Note that for the rSTR, we consider also the overhead for the storage of the VLAD vectors used for the reordering of the results (higher values mean greater occupations).

6 Conclusions and Future Work

In this work, we proposed a 'blockwise' extension of surrogate text representation, which is in principle applicable not only to VLAD but also to any other vector or compound metric objects. The main advantage of this approach is the elimination for the need of the reordering phase. Using the same hardware and text search engine (i.e., Lucene), we were able to compare with the state-of-the-art baseline STR approach exploiting the reordering phase.

The experimental evaluation on the blockwise extension revealed very promising performance in terms of mAP and response time. However, the drawback of it resides in the expansion of the number of terms in the textual representation of the VLADs. This produces an inverted index that, using Lucene, is one order of magnitude greater than the baseline STR. To alleviate this problem, we propose to shrink the index reducing the document, as we did for the query, by eliminating the terms associated with a low value of *tf*idf* weight. This approach is very effective but has the disadvantage that need a double indexing phase or at least a pre-analysis of the dataset in order to calculate the *tf*idf* weight of the terms. Future work will investigate this aspect in more detail.

ACKNOWLEDGEMENTS

This work was partially supported by EAGLE, Europeana network of Ancient Greek and Latin Epigraphy, co-founded by the European Commision, CIP-ICT-PSP.2012.2.1 - Europeana and creativity, Grant Agreement n. 325122.

REFERENCES

Amato, G., Bolettieri, P., Falchi, F., and Gennaro, C. (2013a). Large scale image retrieval using vector of locally aggregated descriptors. In Brisaboa, N., Pedreira, O., and Zezula, P., editors, *Similarity Search and Applications*, volume 8199 of *Lecture Notes in Computer Science*, pages 245–256. Springer Berlin Heidelberg.

Amato, G., Bolettieri, P., Falchi, F., Gennaro, C., and Rabitti, F. (2011). Combining local and global visual feature similarity using a text search engine. In *Content-Based Multimedia Indexing (CBMI), 2011 9th International Workshop on*, pages 49 –54.

Amato, G., Falchi, F., and Gennaro, C. (2013b). On reducing the number of visual words in the bag-of-features representation. In *VISAPP 2013 - Proceedings of the International Conference on Computer Vision Theory and Applications*, volume 1, pages 657–662.

Amato, G., Falchi, F., Gennaro, C., and Bolettieri, P. (2014a). Indexing vectors of locally aggregated descriptors using inverted files. In *Proceedings of International Conference on Multimedia Retrieval*, ICMR '14, pages 439:439–439:442.

Amato, G., Gennaro, C., and Savino, P. (2014b). MI-File: using inverted files for scalable approximate similarity search. *Multimedia Tools and Applications*, 71(3):1333–1362.

Arandjelovic, R. and Zisserman, A. (2013). All about VLAD. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1578–1585.

Bay, H., Tuytelaars, T., and Van Gool, L. (2006). SURF: Speeded Up Robust Features. In Leonardis, A., Bischof, H., and Pinz, A., editors, *Computer Vision - ECCV 2006*, volume 3951 of *Lecture Notes in Computer Science*, pages 404–417. Springer Berlin Heidelberg.

Boureau, Y.-L., Bach, F., LeCun, Y., and Ponce, J. (2010). Learning mid-level features for recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2559–2566.

Chavez, G., Figueroa, K., and Navarro, G. (2008). Effective proximity retrieval by ordering permutations. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(9):1647 –1658.

Chen, D., Tsai, S., Chandrasekhar, V., Takacs, G., Chen, H., Vedantham, R., Grzeszczuk, R., and Girod, B. (2011). Residual enhanced visual vectors for on-device image matching. In *Signals, Systems and Computers (ASILOMAR), 2011 Conference Record of the Forty Fifth Asilomar Conference on*, pages 850–854.

Chum, O., Philbin, J., Sivic, J., Isard, M., and Zisserman, A. (2007). 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.

Csurka, G., Dance, C., Fan, L., Willamowski, J., and Bray, C. (2004). Visual categorization with bags of keypoints. *Workshop on statistical learning in computer vision, ECCV*, 1(1-22):1–2.

Datar, M., Immorlica, N., Indyk, P., and Mirrokni, V. S. (2004). Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry*, SCG '04, pages 253–262.

Delhumeau, J., Gosselin, P.-H., Jégou, H., and Pérez, P. (2013). Revisiting the VLAD image representation. In *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, pages 653–656.

Esuli, A. (2009). MiPai: Using the PP-Index to Build an Efficient and Scalable Similarity Search System. In *Proceedings of the 2009 Second International Workshop on Similarity Search and Applications*, SISAP '09, pages 146–148.

Fagin, R., Kumar, R., and Sivakumar, D. (2003). Comparing top-k lists. *SIAM J. of Discrete Math.*, 17(1):134–160.

Gennaro, C., Amato, G., Bolettieri, P., and Savino, P. (2010). An approach to content-based image retrieval based on the lucene search engine library. In Lalmas, M., Jose, J., Rauber, A., Sebastiani, F., and Frommholz, I., editors, *Research and Advanced Technology for Digital Libraries*, volume 6273 of *Lecture Notes in Computer Science*, pages 55–66. Springer Berlin Heidelberg.

Jaakkola, T. and Haussler, D. (1998). Exploiting generative models in discriminative classifiers. In *In Advances in Neural Information Processing Systems 11*, pages 487–493.

Jégou, H. and Chum, O. (2012). Negative evidences and co-occurences in image retrieval: The benefit of pca and whitening. In Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., and Schmid, C., editors, *Computer Vision–ECCV 2012*, volume 7573 of *Lecture Notes in Computer Science*, pages 774–787. Springer.

Jégou, H., Douze, M., and Schmid, C. (2008). Hamming embedding and weak geometric consistency for large scale image search. In Forsyth, D., Torr, P., and Zisserman, A., editors, *Computer Vision – ECCV 2008*, volume 5302 of *Lecture Notes in Computer Science*, pages 304–317. Springer Berlin Heidelberg.

Jégou, H., Douze, M., and Schmid, C. (2009). Packing bag-of-features. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2357 –2364.

Jégou, H., Douze, M., and Schmid, C. (2010). Improving bag-of-features for large scale image search. *International Journal of Computer Vision*, 87:316–336.

Jégou, H., Douze, M., Schmid, C., and Pérez, P. (2010a). Aggregating local descriptors into a compact image representation. In *IEEE Conference on Computer Vision & Pattern Recognition*, pages 3304–3311.

Jégou, H., Perronnin, F., Douze, M., Sànchez, J., Pérez, P., and Schmid, C. (2012). Aggregating local image descriptors into compact codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1704–1716.

Jégou, H., Schmid, C., Harzallah, H., and Verbeek, J. (2010b). Accurate image search using the contextual dissimilarity measure. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(1):2–11.

Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2.

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

McLachlan, G. and Peel, D. (2000). *Finite Mixture Models*. Wiley series in probability and statistics. Wiley.

Peng, X., Wang, L., Qiao, Y., and Peng, Q. (2014). Boosting vlad with supervised dictionary learning and high-order statistics. In Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T., editors, *Computer Vision - ECCV 2014*, volume 8691 of *Lecture Notes in Computer Science*, pages 660–674. Springer International Publishing.

Perd'och, M., Chum, O., and Matas, J. (2009). Efficient representation of local geometry for large scale object retrieval. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 9–16.

Perronnin, F. and Dance, C. (2007). Fisher kernels on visual vocabularies for image categorization. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8.

Perronnin, F., Liu, Y., Sanchez, J., and Poirier, H. (2010). Large-scale image retrieval with compressed fisher vectors. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3384 –3391.

Philbin, J., Chum, O., Isard, M., Sivic, J., and Zisserman, A. (2007). Object retrieval with large vocabularies and fast spatial matching. In *Computer Vision and Pattern Recognition, 2007. CVPR 2007. IEEE Conference on*, pages 1–8.

Philbin, J., Chum, O., Isard, M., Sivic, J., and Zisserman, A. (2008). 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.

Salton, G. and McGill, M. J. (1986). *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA.

Sivic, J. and Zisserman, A. (2003). Video google: A text retrieval approach to object matching in videos. In *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ICCV '03, pages 1470–1477.

Spyromitros-Xioufis, E., Papadopoulos, S., Kompatsiaris, I. Y., Tsoumakas, G., and Vlahavas, I. (2014). A comprehensive study over vlad and product quantization in large-scale image retrieval. *Multimedia, IEEE Transactions on*, 16(6):1713–1728.

Thomee, B., Bakker, E. M., and Lew, M. S. (2010). TOP-SURF: A visual words toolkit. In *Proceedings of the International Conference on Multimedia*, MM '10, pages 1473–1476.

Tolias, G. and Avrithis, Y. (2011). Speeded-up, relaxed spatial matching. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1653–1660.

Tolias, G. and Jégou, H. (2013). Local visual query expansion: Exploiting an image collection to refine local descriptors. Research Report RR-8325, INRIA.

Van Gemert, J., Veenman, C., Smeulders, A., and Geusebroek, J.-M. (2010). Visual word ambiguity. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(7):1271–1283.

Van Gemert, J. C., Geusebroek, J.-M., Veenman, C. J., and Smeulders, A. W. (2008). Kernel codebooks for scene categorization. In Forsyth, D., Torr, P., and Zisserman, A., editors, *Computer Vision - ECCV 2008*, volume 5304 of *Lecture Notes in Computer Science*, pages 696–709. Springer Berlin Heidelberg.

Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., and Gong, Y. (2010). Locality-constrained linear coding for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3360–3367.

Witten, I. H., Moffat, A., and Bell, T. C. (1999). *Managing gigabytes: compressing and indexing documents and images*. Multimedia Information and Systems Series. Morgan Kaufmann Publishers.

Yang, J., Yu, K., Gong, Y., and Huang, T. (2009). Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1794–1801.

Zezula, P., Amato, G., Dohnal, V., and Batko, M. (2006). *Similarity Search: The Metric Space Approach*, volume 32 of *Advances in Database Systems*. Springer-Verlag.

Zhao, W.-L., Jégou, H., and Gravier, G. (2013). Oriented pooling for dense and non-dense rotation-invariant features. In *BMVC - 24th British Machine Vision Conference*.