Feature Learning for the Image Retrieval Task

Aakanksha Rana, Joaquin Zepeda, Patrick Perez

Technicolor R&I, 975 avenue des Champs Blancs, CS 17616, 35576 Cesson Sevigne, France

Abstract. In this paper we propose a generic framework for the optimization of image feature encoders for image retrieval. Our approach uses a triplet-based objective that compares, for a given query image, the similarity scores of an image with a matching and a non-matching image, penalizing triplets that give a higher score to the non-matching image. We use stochastic gradient descent to address the resulting problem and provide the required gradient expressions for generic encoder parameters, applying the resulting algorithm to learn the power normalization parameters commonly used to condition image features. We also propose a modification to codebook-based feature encoders that consists of weighting the local descriptors as a function of their distance to the assigned codeword before aggregating them as part of the encoding process. Using the VLAD feature encoder, we show experimentally that our proposed optimized power normalization method and local descriptor weighting method yield improvements on a standard dataset.

1 Introduction

Image search methods can be broadly split into two categories. In the first category, *semantic search*, the aim is to retrieve images containing visual concepts. For example, the user might want to find images containing cats. In the second category, *image retrieval*, the search system is given an image of a scene, and the aim is to find all images of the same scene modulo some task-related transformation. Examples of simple transformations include changes in scene illumination, image cropping or scaling. More challenging transformations include drastic changes in background, wide changes in the perspective of the camera, high compression ratios, or picture-of-video-screen artifacts.

Common to both semantic search and image retrieval methods is the need to encode the image into a single, fixed-dimensional feature vector. Many successful image feature encoders have been proposed, and these generally operate on the fixed-dimensional local descriptor vectors extracted from densely [1] or sparsely [2, 3] sampled local regions of the image. The feature encoder aggregates these local descriptors to produce a higher dimension image feature vector. Examples of such feature encoders include the bag-of-words encoder [4], the Fisher encoder [5] and the VLAD encoder [6]. All these encoding methods share common parametric post-processing steps where an element-wise power computation and subsequent l_2 normalization are applied. They also depend on specific models of the data distribution in the local-descripor space. For bag-of-words and VLAD, the model is a codebook obtained using K-means, while the Fisher encoding is based on a Gaussian Mixture Model (GMM). In both cases, the model defining the encoding scheme is built in an unsupervised manner using an optimization objective unrelated to the image search task.

For the case of semantic search, recent work has focused on learning the feature encoder parameters to make it better suited to the task at hand. A natural learning objective to use in this situation is the max-margin objective otherwise used to learn support vector machines. Notably, [7] learned the components of the GMM used in the Fisher encoding by optimizing, relative to the GMM mean and variance parameters, the same objective that produces the linear classifier commonly used to carry out semantic search. Approaches based on deep Convolutional Neural Networks (CNNs) [8, 9] can also be interpreted as feature learning methods, and these now define the new state-of-the art baseline in semantic search. Indeed Sydorov *et al.* discuss how the Fisher encoder can be interpreted as a deep network, since both consist of alternating layers of linear and non-linear operations.

For the image retrieval task, however, the feature learning literature is lacking. One existing proxy approach is to also use the max-margin objective, and hence features encoders that were learned for the semantic search task [10]. Although the search tasks are not the same, this approach indeed results in improved image retrieval results, since both tasks are based on human visual interpretations of similarity. A second approach instead focuses on learning the local descriptor vectors at the input of the feature encoder. The objective used in this is case engineered to enforce matching, based on the learned local descriptors, of small image blocks centered on the same point in 3-D space, but from images taken from different perspectives [11, 12].

One reason why these two approaches circumvent the actual task of image retrieval is the lack of objective functions that are good surrogates for the mean Average Precision (mAP) measure commonly used to evaluate image retrieval systems. Surrogate objectives are necessary because the mAP measure is non-differentiable as it depends on a ranking of the images being searched. The main contribution of this paper is hence to propose a new surrogate objective specifically for the image retrieval task. We show how this objective can be minimized using stochastic gradient descent, and apply the resulting algorithm to select the power-normalization parameters of the VLAD feature encoder. As a second contribution, we also propose a novel method to weight local descriptors for codebook-based image feature encoders that reduces the importance of descriptors too far away from their assigned codeword. We test both contributions indepently and jointly and demonstrate improvements on a standard image retrieval performance.

The remainder of this paper is organized as follows: In the next section we describe standard feature encoding methods, focusing on the VLAD encoding that we use in our experiments. In Section 3 we described the proposed objective and the resulting learning algorithm, and in Section 4 we present the proposed descriptor-weighting method. We present experimental results in Section 5 and concluding remarks in Section 6.

*Notation:*We denote scalars, vectors and matrices using, respectively standard, bold, and upper-case bold typeface (e.g., scalar a, vector $\mathbf{a}$ and matrix $\mathbf{A}$). We use $\mathbf{v}_k$ to denote a vector from a sequence $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N$, and v_k to denote the k-th coefficient of vector $\mathbf{v}$. We let $[\mathbf{a}_k]_k$ (respectively, $[a_k]_k$) denotes concatenation of the vectors $\mathbf{a}_k$ (scalars a_k) to form a single column vector. Finally, we use $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$ to denote the Jacobian matrix with (i, j)-th entry $\frac{\partial y_i}{\partial x_j}$.

2 Image Encoding Methods

Image encoders operate on the local descriptors $\mathbf{x} \in \mathbb{R}^d$ extracted from each image. Hence in this work we represent images as a set $I = \{\mathbf{x}_i \in \mathbb{R}^d\}_i$ of local SIFT descriptors extracted densely [1] or with the Hessian affine region detector [3].

One of the earliest image encoding methods proposed was the bag-of-features encoder (BOF) [4]. The BOF encoder is based on a codebook $\{\mathbf{c}_k \in \mathbb{R}^d\}_{k=1}^L$ obtained by applying K-means to all the local descriptors $\mathcal{T} = \bigcup_t I_t$ of a set of training images. Letting $\mathcal{C}_k$ denote the Voronoi cell $\{\mathbf{x} | \mathbf{x} \in \mathbb{R}^d, k = \operatorname{argmin}_j |\mathbf{x} - \mathbf{c}_j|\}$ associated to codeword $\mathbf{c}_k$, the resulting feature vector for image I is

$$\mathbf{r}^B = [\#(\mathcal{C}_k \cap I)]_k, \tag{1}$$

where # yields the number of elements in the set. The Fisher encoder [5] instead relies on a GMM model also trained on $\bigcup_t I_t$. Letting $\beta_i, \mathbf{c}_i, \boldsymbol{\Sigma}_i$ denote, respectively, the i-th GMM component's *i*) prior weight, *ii*) mean vector, and *iii*) covariance matrix (assumed diagonal), the first-order Fisher feature vector is

$$\mathbf{r}^F = \left[\sum_{\mathbf{x} \in I} \frac{p(k|\mathbf{x})}{\sqrt{\beta_i}} \boldsymbol{\Sigma}_k^{-1} (\mathbf{x} - \mathbf{c}_k) \right]_k. \tag{2}$$

A hybrid combination between BOF and Fisher techniques called VLAD has been proposed [13] that offers a good compromise between the Fisher encoders's performance and the BOF encoder's processing complexity: Similarly to the state-of-the art Fisher aggregator, it encodes residuals $\mathbf{x} - \mathbf{c}_k$, but it hard-assigns each local descriptor to a single cell $\mathcal{C}_k$ instead of using a costly soft-max assignment as in (2). In a later work, [6] further proposed incorporating several conditioning steps that improved the performance of the feature encoder. The resulting complete encoding process begins by first aggregating, on a per-cell basis, the l_2 normalized difference of each local descriptor relative the cell's codeword, subsequently rotating the resulting descriptor using the matrix $\boldsymbol{\Phi}_k$ (obtained by PCA on the training descriptors $\mathcal{C}_k \cap \mathcal{T}$):

$$\mathbf{r}_k^V = \boldsymbol{\Phi}_k \sum_{\mathbf{x} \in I \cap \mathcal{C}_k} \frac{\mathbf{x} - \mathbf{c}_k}{|\mathbf{x} - \mathbf{c}_k|} \in \mathbb{R}^d, \tag{3}$$

$$\tag{4}$$

The L sub-vectors thus obtained are then stacked to form a large vector $\mathbf{v}$ that is then power-normalized and l_2 normalized:

$$\mathbf{v} = [\mathbf{r}_k^V]_k \in \mathbb{R}^{dL}, \tag{5}$$

$$\mathbf{p} = [h_{\alpha_j}(v_j)]_j, \tag{6}$$

$$\mathbf{n} = \mathbf{g}(\mathbf{p}). \tag{7}$$

The power normalization function $h_\alpha(x)$ and the l_2 normalization function $\mathbf{n}(\mathbf{v})$ are

$$h_\alpha(x) = \operatorname{sign}(x)|x|^\alpha, \tag{8}$$

$$\mathbf{g}(\mathbf{x}) = \frac{\mathbf{x}}{|\mathbf{x}|_2}. \tag{9}$$

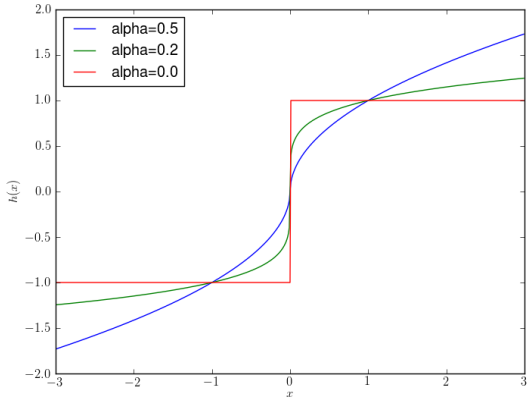

Fig. 1. Plot of $h_\alpha(x)$ for various values of α.

The power normalization function (8) is widely used as a post-processing stage for image features [14, 6, 1, 15]. This post-processing stage is meant to mitigate (respectively, enhance) the contribution of the larger (smaller) coefficients in the vector (*cf.*, Fig. 1). Combining power normalization with the PCA rotation matrices $\boldsymbol{\Phi}_k$ was shown in [6] to yield very good results. In all the approaches using power normalization, the α_j are kept constant for all entries in the vector, $\alpha_j = \alpha, \forall j$. This restriction comes from the fact that α is chosen empricially (often to $\alpha = 0.5$ or $\alpha = 0.2$), and choosing different values for each α_j is difficult. In section 3 we remove this difficulty by applying our proposed feature learning method to the optimization of the α_j.

3 Feature learning for image retrieval

Feature learning has been pursued in the context of image classification [7] or for learning local descriptors akin to parametric variants of the SIFT descriptor [11, 12]. Learning features specifically for the image retrieval task, however, has not been pursued previously. In this section we propose an approach to do so, and apply it to the optimization of the parameters of the VLAD feature encoding method described in Section 2.

The main difficulty in learning for the image retrieval task lies in the non-smoothness and non-differentiablity of the standard performance measures used in this context. These measures are all based on *recall* and *precision* computed over a ground-truth dataset containing known groups of matching images [16, **?**]: A given query image is used to obtain a ranking $(i_k \in \{1,\dots,N\})_k$ of the N images in the dataset (for example, by an ascending sort of their feature distances relative to the query feature). Given the ground-truth matches $\mathcal{M} = \{i_{k_j}\}_j$ for the query, the recall and precision at rank k are computed using the first k ranked images $\mathcal{F}_k = \{i_1,\dots,i_k\}$ as follows (where # denotes

set cardinality):

$$r(k) = \frac{\#(\mathcal{F}_k \cap \mathcal{M})}{\#\mathcal{M}}, \tag{10}$$

$$p(k) = \frac{\#(\mathcal{F}_k \cap \mathcal{M})}{k}. \tag{11}$$

The *average precision* is then the area under the curve obtained by plotting $p(k)$ versus $r(k)$ for a single query image. A common performance measure is the mean, over all images in the dataset, of the average precision. This mean Average Precision (mAP) measure, and all measures based on recall and precision, are non-differentiable, and it is hence difficult to use them in an optimization framework, motivating the need for an adequate surrogate objective.

3.1 Proposed objective

We assume that we are given a set of N training images and that for each image i, we are also given labels $\mathcal{M}_i \subset \{1,\ldots,N\}$ of images that are a match to image i and labels $\mathcal{N}_i \subset \{1,\ldots,N\}$ of images that do not match image i. We assume that some feature encoding scheme has been chosen that is parametrized by a vector $\boldsymbol{\theta}$ and that produces feature vectors $\mathbf{n}_i(\boldsymbol{\theta})$. Our aim is to define a procedure to select good values for the parameters $\boldsymbol{\theta}$ by minimizing the following objective:

$$f(\boldsymbol{\theta}) = \frac{1}{M} \sum_{i,j \in \mathcal{M}_i, k \in \mathcal{N}_i} \phi(\mathbf{n}_i(\boldsymbol{\theta}), \mathbf{n}_j(\boldsymbol{\theta}), \mathbf{n}_k(\boldsymbol{\theta})), \tag{12}$$

where M is the total number of terms in the triple summation and

$$\phi(\eta, \mathbf{a}, \mathbf{b}) = \max(0, \varepsilon - (\eta^{\mathrm{T}}(\mathbf{a} - \mathbf{b}))). \tag{13}$$

The parameter ε enforces some small, non-zero margin that can be held constant (*e.g.*, $\varepsilon = 1e-2$) or increased gradually during the optimization (*e.g.*, between 0 and $1e-1$).

An objective based on image triplets similarly to (12) has been previously used in metric learning [17], where the aim is commonly to learn a matrix $\mathbf{W}$ used to compute distances between two given feature vectors $\mathbf{n}_i$ and $\mathbf{n}_j$ using $(\mathbf{n}_i - \mathbf{n}_j)^{\mathrm{T}}\mathbf{W}(\mathbf{n}_i - \mathbf{n}_j)$. Our aim is instead to learn the parameters $\boldsymbol{\theta}$ that define the encoding process. In this work in particular we learn the power normalization parameters α_j in (6).

3.2 Optimization strategy

Stochastic Gradient Descent (SGD) is a well-established, robust optimization method offering advantages when computational time or memory space is the bottleneck [18], and this is the approach we take to optimize (12). Given the parameter estimate $\boldsymbol{\theta}_t$ at iteration t, SGD substitutes the gradient for the objective

$$\left.\frac{\partial f}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}_t} = \frac{1}{M} \sum_{i,j \in \mathcal{M}_i, k \in \mathcal{N}_i} \left.\frac{\partial \phi(\mathbf{n}_i, \mathbf{n}_j, \mathbf{n}_k)}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}_t} \tag{14}$$

by an estimate from a single (i,j,k)-triplet drawn at random at time t,

$$\nabla\phi_{i_t j_t k_t}(\boldsymbol{\theta}_t) \triangleq \left.\frac{\partial\phi(\mathbf{n}_{i_t},\mathbf{n}_{j_t},\mathbf{n}_{k_t})}{\partial\boldsymbol{\theta}}\right|_{\boldsymbol{\theta}_t}. \tag{15}$$

The resulting SGD update rule is

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \gamma_t \cdot \nabla\phi_{i_t j_t k_t}(\boldsymbol{\theta}_t) \tag{16}$$

where γ_t is a learning rate that can be made to decay with t, *e.g.*, $\gamma_t = \gamma/t$, and the parameter γ can be set by cross-validation. SGD is guaranteed to converge to a local minimum under mild decay conditions on γ_t [18].

When the power normalization and l_2 normalization post-processing stages in (6) and (7) are used, the gradient (15) required in (16) can be computed using the chain rule as follows, where we use the notation $\frac{\partial\mathbf{n}}{\partial\mathbf{p}_i} = \left.\frac{\partial\mathbf{n}}{\partial\mathbf{p}}\right|_{\mathbf{p}_i}$:

$$\begin{aligned}\nabla\phi_{i,j,k}(\boldsymbol{\theta}) \triangleq & \left.\frac{\partial\phi}{\partial\eta}\right|_{\mathbf{n}_i} \cdot \frac{\partial\mathbf{n}}{\partial\mathbf{p}_i} \cdot \frac{\partial\mathbf{p}(I_i)}{\partial\boldsymbol{\theta}} \\ & + \left.\frac{\partial\phi}{\partial\mathbf{a}}\right|_{\mathbf{n}_j} \cdot \frac{\partial\mathbf{n}}{\partial\mathbf{p}_j} \cdot \frac{\partial\mathbf{p}(I_j)}{\partial\boldsymbol{\theta}} \\ & + \left.\frac{\partial\phi}{\partial\mathbf{b}}\right|_{\mathbf{n}_k} \cdot \frac{\partial\mathbf{n}}{\partial\mathbf{p}_k} \cdot \frac{\partial\mathbf{p}(I_k)}{\partial\boldsymbol{\theta}}.\end{aligned} \tag{17}$$

The partial Jacobians in the above expression are given below, where we use sub-gradients for those expressions relying on the non-differentiable hinge loss:

$$\frac{\partial\boldsymbol{\phi}}{\partial\boldsymbol{\eta}} = \begin{cases} 0, & \text{if } (\eta^{\mathrm{T}}(\mathbf{a}-\mathbf{b})) \geq \varepsilon \\ (\mathbf{b}-\mathbf{a})^{\mathrm{T}}, & \text{otherwise} \end{cases}, \tag{18}$$

$$\frac{\partial\boldsymbol{\phi}}{\partial\mathbf{b}} = -\frac{\partial\boldsymbol{\phi}}{\partial\mathbf{a}} = \begin{cases} 0, & \text{if } (\eta^{\mathrm{T}}(\mathbf{a}-\mathbf{b})) \geq \varepsilon \\ \boldsymbol{\eta}^{\mathrm{T}}, & \text{otherwise} \end{cases}, \tag{19}$$

$$\frac{\partial\mathbf{n}}{\partial\mathbf{p}} = |\mathbf{p}|_2^{-1}\left(\mathbf{I}-\mathbf{n}\mathbf{n}^{\mathrm{T}}\right). \tag{20}$$

The above expressions are generic and can be used for any parameter $\boldsymbol{\theta}$ of the feature encoder that one wishes to specialize. In this work we learn the power normalization coefficients α_j in (6) and hence $\boldsymbol{\theta} = \boldsymbol{\alpha}$, and the required Jacobian is

$$\frac{\partial\mathbf{p}}{\partial\boldsymbol{\alpha}} = \mathrm{diag}\left([\log(|v_i|).|v_i|^{\alpha_i}]_i\right). \tag{21}$$

4 Local-descriptor pruning

In this section we propose a local-descriptor pruning method applicable to feature encoding methods like BOF, VLAD and Fisher that are based on stacking sub-vectors $\mathbf{r}_k$,

where each sub-vector is computed from the local descriptors assigned to a cell $\mathcal{C}_k$. The proposed approach shares some similaritites with [19, 20].

Unlike the case for low-dimensional sub-spaces, the cells $\mathcal{C}_k$ in high-dimensional local-descriptors spaces are almost always unbounded, meaning that they have infinite volume.[1] Yet only a part of this volume is informative visually. This suggests removing those descriptors that are too far away from the cell center $\mathbf{c}_k$ when constructing the sub-vectors $\mathbf{r}_k$ in (1), (2) and (3). This can be done by restricting the summations in (1), (2) and (3) only to those vectors $\mathbf{x}$ that *i*) are in the cell $\mathcal{C}_k$ and *ii*) satisfy the following distance-to-$\mathbf{c}_k$ condition:

$$(\mathbf{x}-\mathbf{c}_k)^{\mathrm{T}}\mathbf{M}_k^{-1}(\mathbf{x}-\mathbf{c}_k) \leq \gamma\sigma_k^2. \tag{22}$$

Here γ is determined experimentally by cross-validation and the parameter σ_k is the empirical variance of the distance in (22) computed over those descriptors from the training set that are in the cell. The matrix $\mathbf{M}_k$ can be either

anisotropic: the empirical covariance matrix computed from $\mathcal{T} \cap \mathcal{C}_k$;
axes-aligned: the same as the anisotropic $\mathbf{M}_k$, but with all elements outside the diagonal set to zero;
isotropic: a diagonal matrix $\sigma_k^2\mathbf{I}$ with σ_k^2 equal to the mean diagonal value of the axes-aligned $\mathbf{M}_k$.

While the anisotropic variant offers the most geometrical modelling flexibility, it also drastically increases the computational cost. The isotropic variant, on the other hand, enjoys practically null computational overhead, but also the least modelling flexibility. The axes-aligned variant offers a compromise between the two approaches.

4.1 Soft-weight extension

The prunning carried out by (22) can be implemented by means of $1/0$ weights

$$w_k(\mathbf{x}) = [\![(\mathbf{x}-\mathbf{c}_k)^{\mathrm{T}}\mathbf{M}_k^{-1}(\mathbf{x}-\mathbf{c}_k) \leq \gamma\sigma_k^2]\!] \tag{23}$$

applied to the summation terms in (1), (2) and (3). For example, for (3) the weights would be used as follows:

$$\mathbf{r}_k^V = \boldsymbol{\Phi}_k^{\mathrm{T}} \sum_{\mathbf{x}\in I\cap\mathcal{C}_k} w_k(\mathbf{x}) \frac{\mathbf{x}-\mathbf{c}_k}{|\mathbf{x}-\mathbf{c}_k|} \in \mathbb{R}^d. \tag{24}$$

A simple extension of the hard-pruning approach corresponding to (23) consists of instead using *exponential weights*

$$w_k(\mathbf{x}) = \exp\left(-\frac{\omega}{\sigma_k^2}(\mathbf{x}-\mathbf{c}_k)^{\mathrm{T}}\mathbf{M}_k^{-1}(\mathbf{x}-\mathbf{c}_k)\right), \tag{25}$$

[1] Although l_2 normalization commonly applied to local descriptors limits the effective volume of each cell, one should note that l_2 normalization amounts to a reduction of dimensionality by one dimension, and that l_2-normalized data is still high-dimensional. Yet the question still remains on whether pruning mechanisms other than those proposed herein exist that better take into account the constraints on the data layout.

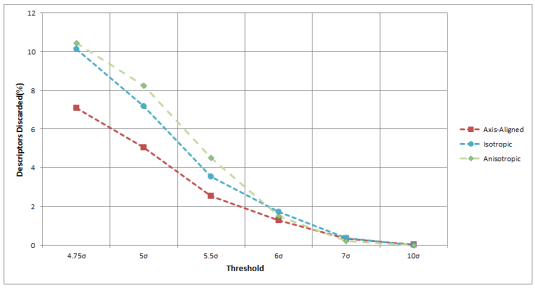

Fig. 2. Percentage of pruned descriptors by anisotropic axes aligned pruning, isotropic pruning, and anisotropic pruning. Holidays dataset with Hessian-Affine SIFT.

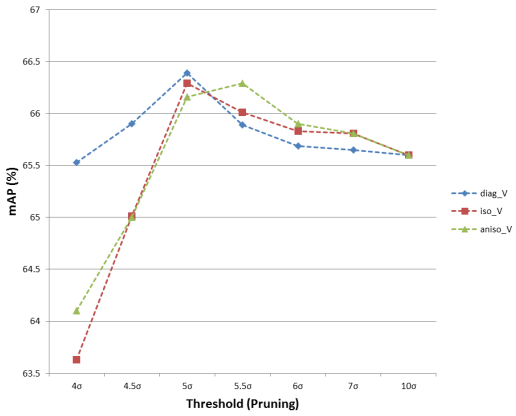

Fig. 3. Impact of Mahalanobis-metric based descriptor pruning on image retrieval performance when using anisotropic axes-aligned pruning (blue), isotropic pruning (red), and anisotropic pruning (green). Holidays dataset with Hessian-Affine SIFT.

where the parameter ω is set experimentally, or *inverse weights*

$$w_k(\mathbf{x}) = \frac{\sigma_k^2}{(\mathbf{x}-\mathbf{c}_k)^{\mathrm{T}}\mathbf{M}_k^{-1}(\mathbf{x}-\mathbf{c}_k)}. \tag{26}$$

5 Results

Setup: We use SIFT descriptors extracted from local regions computed with the Hessian-affine detector [3] or from a dense-grid using three block sizes (16, 24, 32) with a step size of 3 pixels [1]. When using the Hessian affine detector, we use the RootSIFT variant following [14]. As a training set, we use the Flickr60K dataset [16] composed of 60,000 images extracted randomly from Flickr. This data set is used to learn the codebook,

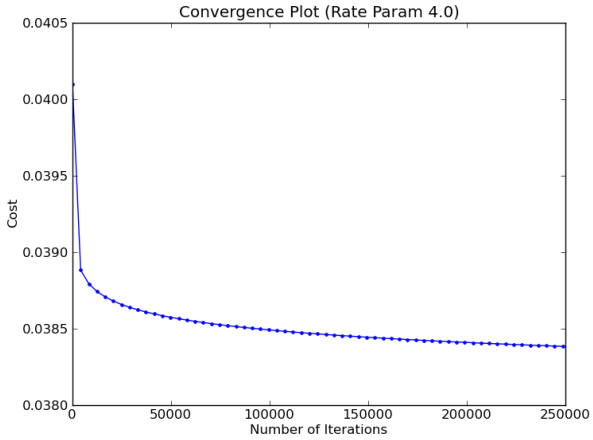

Fig. 4. Convergence plot for the α_j learning procedure.

rotation matrices, per-cluster pruning thresholds and covariance matrices for the computation of the Mahalanobis metrics used for pruning of local descriptors. For testing, we use the INRIA Holidays dataset [16] which contains 1491 high resolution personal photos of 500 locations or objects, where common locations/objects define matching images. The search quality in all the experiments is measured using mAP (mean average precision) using the code provided by the authors [16]. All the experiments have been carried out using the VLAD image encoder and a codebook of size 64 following [6].

Evaluation of pruning methods: In Table 1, we evaluate the pruning approaches discussed in Section 4. Each variant is specified by a choice of weight type (hard, exponential or inverse), metric type (isotropic, anisotropic or axes-aligned), and local feature (dense or Hessian affine). The best result overall is obtained using axes-aligned exponential weighting (74.28% and 67.02% for dense and Hessian affine detections, respectively). The choice of the weighting parameter for exponential pruning is empirically set to $\omega = 1.55$. For completeness, we provide plots, for the case of hard-pruning, depicting the percentage of local descriptors removed (Fig. 2) and the resulting mAP score (Fig. 3) as a function of $\sqrt{\gamma}\sigma_k$. The values plotted in Fig. 2 are averaged over all cells $\mathcal{C}_k$.

Evaluation of $\boldsymbol{\alpha}$ learning: In Fig. 4, we provide a plot of the cost in (12) as a function of the number of SGD iterations (16) using a dataset of $M = 8,000$ image triplets. The cost drops from 0.0401 to less than 0.0385. The resulting mAP is given in Table 2, where we present results both for the case where α is learned with and without exponential weighting of the local descriptors. The combined effect of exponential weighting and α learning is a gain of 1.86 mAP points.

Descriptors	mAP (%)				
	baseline	Weights	Iso	Aniso	Ax-align
Hessian Affine	65.60	hard	66.29	66.29	66.40
		inverse	66.40	66.39	66.55
		exponential	66.45	66.40	**67.02**
Dense	72.71	hard	73.34	73.37	73.56
		inverse	73.45	73.45	73.60
		exponential	73.69	73.61	**74.28**

Table 1. Summary of feature pruning results for all combinations of detectors-dense or Hessian-affine, metrics - isotropic (Iso), anisotropic (Aniso), and axes-alinged (Ax-align) and weighting schemes - hard, exponential and inverse. Underlines indicate best-in-row and bold best overall. The baseline results are for the system in [6].

Baseline	Exp. weighting only	Learned α_j only	Exp. weighting and learn α_j
72.71	74.28	74.30	**74.57**

Table 2. Summary of best results (with dense detection) when using *(i)* only exponential weighting, *(ii)* learned α_j parameters without exponential weighting, and *(iii)* combined exponential weighting and learning of the α_j paraemters. The baseline results are for the system in [6].

In Fig. 5 and Fig. 5 we provide two examples of top-ten ranked results for two different query images using our proposed modifications. We also provide examples of query images that resulted in improved (Fig. 5) and worsened (Fig. 5) ranking.

6 Conclusions

In this paper we proposed learning the power normalization parameters commonly applied to image feature encoders using an image-triplet-based objective that penalizes erroneous ranking in the image retrieval task. The proposed feature learning approach is applicable to other parameters of the feature encoder. We also propose, for the case of codebook-based feature encoders, weighting local descriptors based on their distance from the assigned codeword. We evaluate both methods experimentally and show that they provide improved results on a standard dataset.

Acknowledgement. This work was partially supported by the FP7 European integrated project AXES.

References

1. Chatfield, K., Lempitsky, V., Vedaldi, A., Zisserman, A.: The devil is in the details: an evaluation of recent feature encoding methods. In: Procedings of the British Machine Vision Conference, British Machine Vision Association (2011) 76.1–76.12
2. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision **60** (2004) 91–110

Fig. 5. Top-ten ranked results. *Top:* Baseline. *Bottom:* With exponential weighting / α_j learning.

3. Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, a., Matas, J., Schaffalitzky, F., Kadir, T., Gool, L.V.: A Comparison of Affine Region Detectors. International Journal of Computer Vision **65** (2005) 43–72
4. Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. International Conference on Computer Vision (2003) 2–9
5. Perronnin, F., Sánchez, J., Mensink, T.: Improving the fisher kernel for large-scale image classification. European Conference on Computer Vision (2010) 143–156
6. Delhumeau, J., Gosselin, P.H., Jégou, H., Pérez, P.: Revisiting the VLAD image representation. Proceedings of ACM International Conference on Multimedia **21** (2013) 653–656
7. Sydorov, V., Sakurada, M., Lampert, C.: Deep Fisher KernelsEnd to End Learning of the Fisher Kernel GMM Parameters. In: Computer Vision and Pattern Recognition. (2014)
8. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet Classification with Deep Convolutional Neural Networks. In: Proceedings of Neural Information Processing Systems. (2012) 1–9
9. Oquab, M., Bottou, L.: Learning and Transferring Mid-Level Image Representations using Convolutional Neural Networks. (2013)
10. Mensink, T., Verbeek, J., Perronnin, F., Csurka, G.: Metric Learning for Large Scale Image Classification : Generalizing to New Classes at Near-Zero Cost. Pattern Analysis and Machine Intelligence (2012)
11. Brown, M., Hua, G., Winder, S.: Discriminative learning of local image descriptors. IEEE Transactions on Pattern Analysis and Machine Intelligence **33** (2011) 43–57
12. Simonyan, K., Vedaldi, A., Zisserman, A.: Descriptor learning using convex optimisation. Proceedings of European Conference on Computer Vision (2012) 1–14

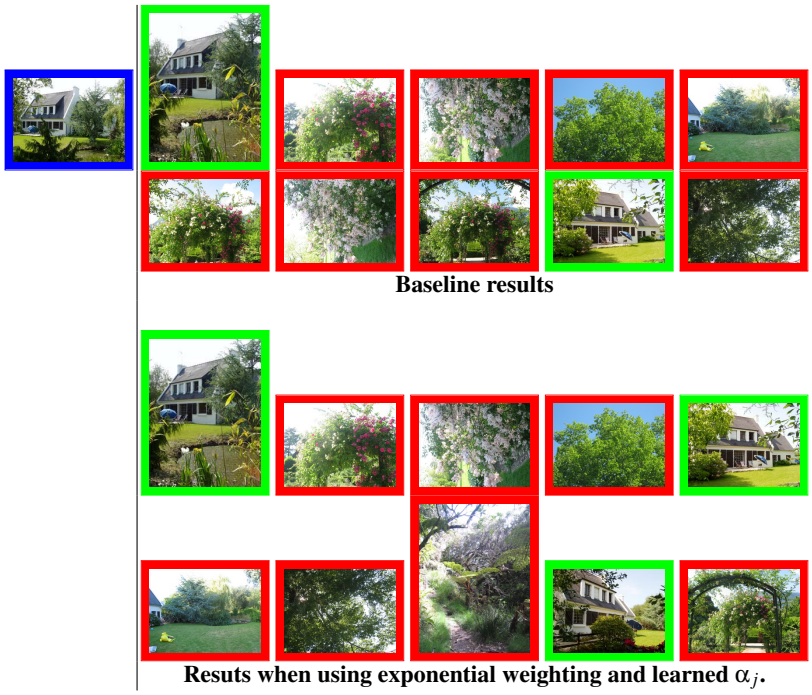

Fig. 6. Top-ten ranked results. *Top:* Baseline. *Bottom:* With exponential weighting / α learning.

13. Jegou, H., Douze, M., Schmid, C., Perez, P.: Aggregating local descriptors into a compact image representation. Proceedings of Computer Vision and Pattern Recognition (2010) 3304–3311
14. Arandjelovic, R., Zisserman, A.: Three things everyone should know to improve object retrieval. Proceedings of Computer Vision and Pattern Recognition (2012)
15. Jegou, H., Perronnin, F., Douze, M., Jorge, S., Patrick, P., Schmid, C.: Aggregating local image descriptors into compact codes. IEEE Transactions on Pattern Analysis and Machine Intelligence (2011) 1–12
16. Jegou, H.: INRIA Holidays dataset (2014)
17. Chechik, G., Shalit, U.: Large Scale Online Learning of Image Similarity through Ranking. Journal of Machine Learning Research (2010) 1–26
18. Bottou, L.: Stochastic gradient descent tricks. In Montavon, G., Orr, G., Müller, K.R., eds.: Neural Networks Tricks of the Trade. 2 edn. Springer (2012)
19. Avila, S., Thome, N., Cord, M., Valle, E., de A Araujo, A.: Bossa: Extended bow formalism for image classification. In: Image Processing (ICIP), 2011 18th IEEE International Conference on. (2011) 2909–2912
20. Avila, S., Thome, N., Cord, M., Valle, E., De A. AraúJo, A.: Pooling in image representation: The visual codeword point of view. Comput. Vis. Image Underst. **117** (2013) 453–465

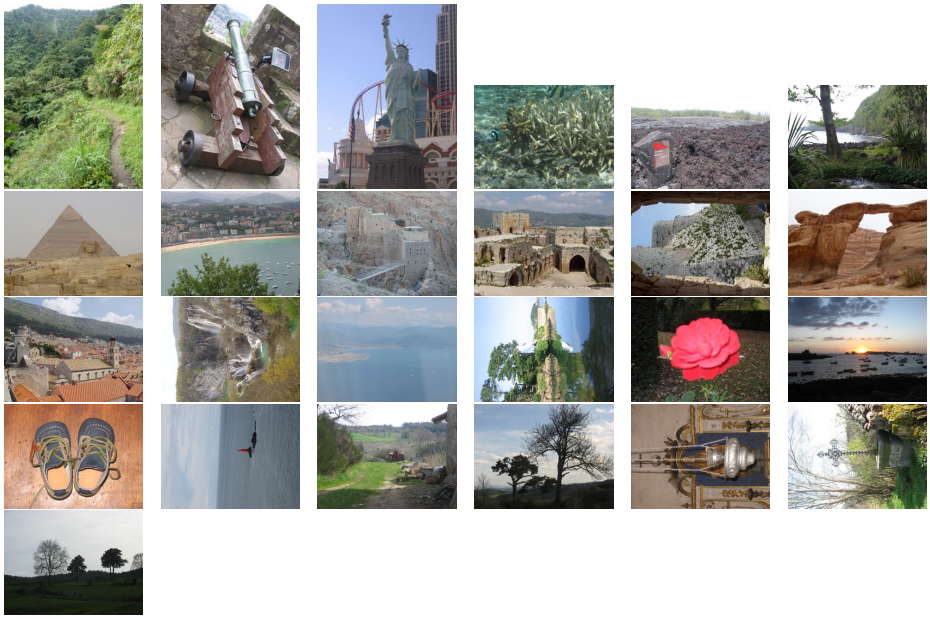

Fig. 7. Query images that result in improved ranking when using α_j learning with exponential weighting.

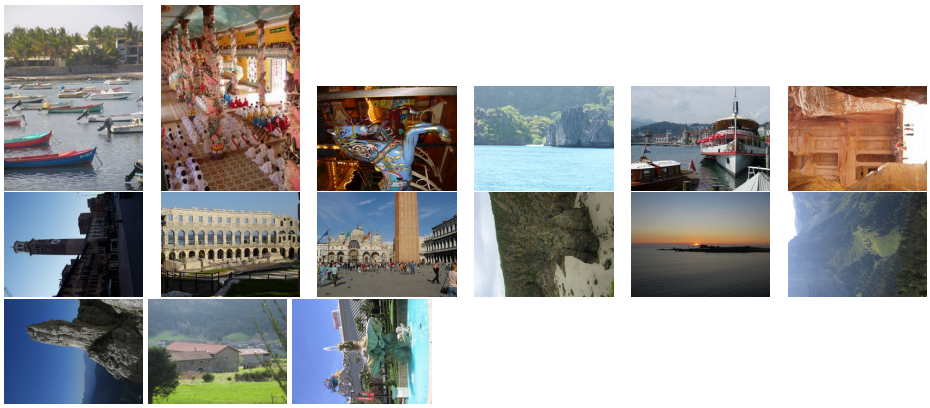

Fig. 8. Query images that result in degraded ranking when using α_j learning with exponential weighting.