FAemb: a function approximation-based embedding method for image retrieval

Thanh-Toan Do Quang D. Tran Ngai-Man Cheung
Singapore University of Technology and Design
{thanhtoan_do, quang_tran, ngaiman_cheung}@sutd.edu.sg

Abstract

The objective of this paper is to design an embedding method mapping local features describing image (e.g. SIFT) to a higher dimensional representation used for image retrieval problem.

By investigating the relationship between the linear approximation of a nonlinear function in high dimensional space and state-of-the-art feature representation used in image retrieval, i.e., VLAD, we first introduce a new approach for the approximation. The embedded vectors resulted by the function approximation process are then aggregated to form a single representation used in the image retrieval framework.

The evaluation shows that our embedding method gives a performance boost over the state of the art in image retrieval, as demonstrated by our experiments on the standard public image retrieval benchmarks.

1. Introduction

The problem of finding a single vector representing a set of local vectors describing an image is an important problem in computer vision. This is because the single representation provides two main benefits. First, it contains the power of local descriptors, such as set of SIFT descriptors [17]. Second, the single represented vectors can be either compared with standard distances used in image retrieval problem or used by robust classification methods such as SVM in classification problem.

There is a wide range of methods for finding a single vector to represent a set of local vectors proposed in the literature such as bag-of-visual-words (BoW) [28], Fisher vector [22], vector of locally aggregated descriptor (VLAD) [12] and its improvements [7, 2], super vector coding [33], vector of locally aggregated tensor (VLAT) [26, 20] which is higher order (tensor) version of VLAD, triangulation embedding (Temb) [14], sparse coding [21], local coordinate coding (LCC) [32], locality-constrained linear coding [29] which is fast version of LCC, local coordinate coding using local tangent (TLCC) [31] which is higher order version of LCC. Among these methods, VLAD [13] and VLAT [20] are well-known embedding methods used in image retrieval problem [13, 20] while TLCC [31] is one of successful embedding methods used in image classification problem.

VLAD is designed for image retrieval problem while TLCC is designed for image classification problem. They also come from different motivations. VLAD's motivation is to characterize the distribution of residual vectors over Voronoi cells learned by a quantizer while TLCC's motivation is to *linearly approximate*[1] a nonlinear function in high dimensional space. Despite above differences, we show that VLAD is actually simplified version of TLCC. This means that we can depart from the idea of linear approximation of function to develop good embedding methods for image retrieval problem.

To find the single representation, all aforementioned methods include two main steps in the processing: embedding and aggregating. The embedding step maps each local descriptor to a high dimensional vector while the aggregating step converts set of mapped high dimensional vectors to a single vector. This paper focuses on the first step. In particular, we develop a new embedding method which can be seen as the generalization of TLCC and VLAT.

In next sections, we first present a brief description of TLCC and show the relationship between TLCC and VLAD. We then present our motivation for designing new embedding method.

1.1. TLCC

TLCC [31] is designed for image classification problem. Its goal is to linearly approximate a smooth nonlinear function $f(\mathbf{x})$, *i.e.* a nonlinear classification function, defined on a high dimensional feature space $\mathbb{R}^d$. TLCC's approach finds an embedding scheme ϕ: $\mathbb{R}^d \to \mathbb{R}^D$ mapping each $\mathbf{x} \in \mathbb{R}^d$ as

$$\mathbf{x} \mapsto \phi(\mathbf{x}) \tag{1}$$

[1]The meaning of "linear approximation" in this paper is that the nonlinear function $f(\mathbf{x})$ defined on $\mathbb{R}^d$ is approximated by a linear function $\mathbf{w}^T\phi(\mathbf{x})$ defined on $\mathbb{R}^D$ where $D > d$.

such that $f(\mathbf{x})$ can be well approximated by a linear function, namely $\mathbf{w}^T\phi(\mathbf{x})$. To solve above problem, TLCC's authors relied on the idea of coordinate coding defined bellow. They showed that with a sufficient selection of coordinate coding, the function $f(\mathbf{x})$ can be linearly approximated.

Definition 1.1 *Coordinate Coding [32]*
A coordinate coding of a point $\mathbf{x} \in \mathbb{R}^d$ is a pair $(\gamma(\mathbf{x}), \mathbf{C})$[2], where $\mathbf{C} = [\mathbf{v}_1, \dots, \mathbf{v}_n] \in \mathbb{R}^{d\times n}$ is a set of n anchor points (bases), and γ is a map of $\mathbf{x} \in \mathbb{R}^d$ to $\gamma(\mathbf{x}) = [\gamma_{\mathbf{v}_1}(\mathbf{x}), \dots, \gamma_{\mathbf{v}_n}(\mathbf{x})]^T \in \mathbb{R}^n$ such that

$$\sum_{j=1}^{n} \gamma_{\mathbf{v}_j}(\mathbf{x}) = 1 \tag{2}$$

It induces the following physical approximation of $\mathbf{x}$ in $\mathbb{R}^d$:

$$\mathbf{x}' = \sum_{j=1}^{n} \gamma_{\mathbf{v}_j}(\mathbf{x})\mathbf{v}_j \tag{3}$$

A good coordinate coding should ensure that $\mathbf{x}'$ closes to $\mathbf{x}$ [3].

Let $(\gamma(\mathbf{x}), \mathbf{C})$ be coordinate coding of $\mathbf{x}$. Under assumption that f is (α, β, ν) Lipschitz smooth, they showed (in lemma 2.2 [31]) that, for all $\mathbf{x} \in R^d$

$$\begin{aligned}&\left| f(\mathbf{x}) - \sum_{j=1}^{n} \gamma_{\mathbf{v}_j}(\mathbf{x}) \left(f(\mathbf{v}_j) + \frac{1}{2}\nabla f(\mathbf{v}_j)^T(\mathbf{x}-\mathbf{v}_j)\right)\right| \\ \leq\ & \frac{1}{2}\alpha \left\|\mathbf{x}-\mathbf{x}'\right\|_2 + \nu \sum_{j=1}^{n} |\gamma_{\mathbf{v}_j}(\mathbf{x})| \left\|\mathbf{x}-\mathbf{v}_j\right\|_2^3 \end{aligned}\tag{4}$$

To ensure a good approximation of $f(\mathbf{x})$, they minimize the RHS of (4). (4) further means that the function $f(\mathbf{x})$ can be linearly approximated by $\mathbf{w}^T\phi(\mathbf{x})$ where $\mathbf{w} = \left[\frac{1}{s}f(\mathbf{v}_j); \frac{1}{2}\nabla f(\mathbf{v}_j)\right]_{j=1}^{n}$ and TLCC embedding $\phi(\mathbf{x})$ defined as

$$\phi(\mathbf{x}) = \left[s\gamma_{\mathbf{v}_j}(x); \gamma_{\mathbf{v}_j}(x)(\mathbf{x}-\mathbf{v}_j)\right]_{j=1}^{n} \in \mathbb{R}^{n(1+d)} \tag{5}$$

where s is a nonnegative constant.

1.2. TLCC as generalization of VLAD

Although TLCC is designed for classification problem and its motivation is different from VLAD, TLCC can be seen as a generalization of VLAD.

If we add following constraint to $\gamma(\mathbf{x})$

$$\left\|\gamma(\mathbf{x})\right\|_0 = 1 \tag{6}$$

then we have $\mathbf{x} \approx \mathbf{x}' = \mathbf{v}_*$. The RHS of (4) becomes

$$\frac{1}{2}\alpha \left\|\mathbf{x}-\mathbf{v}_*\right\|_2 + \nu \left\|\mathbf{x}-\mathbf{v}_*\right\|_2^3 \tag{7}$$

where $\mathbf{v}_*$ is anchor point corresponding to nonzero element of $\gamma(\mathbf{x})$. One of solutions for minimizing (7) under constraints (2) and (6) is K-means algorithm. When K-means is used, we have

$$\mathbf{v}_* = \underset{\mathbf{v}\in\mathbf{C}}{\operatorname{argmin}} \left\|\mathbf{x}-\mathbf{v}\right\|_2 \tag{8}$$

where $\mathbf{C}$ is set of anchor points learned by K-means. Now, considering (5), if we choose $s = 0$ and we remove zero elements attached with s, $\phi(\mathbf{x}) = \left[0, \dots, 0, (\mathbf{x}-\mathbf{v}_*)^T, 0, \dots, 0\right]^T \in \mathbb{R}^{nd}$ will become VLAD.

1.3. Motivation for designing new embedding method

The relationship between TLCC and VLAD means that if we can find $\phi(\mathbf{x})$ such that $f(\mathbf{x})$ can be well linearly approximated ($f(\mathbf{x}) \approx \mathbf{w}^T\phi(\mathbf{x})$), we then can use $\phi(\mathbf{x})$ for image retrieval problem. However, in TLCC's approach, by departing from assumption that f is (α, β, ν) Lipschitz smooth, f is approximated using only its first order approximation at anchor points, *i.e.*, f is approximated as sum of weighted tangents at anchor points. It is not straightforward to use the TLCC framework to have a better approximation, for examples, approximation of f using its second order or higher order approximation at anchor points.

Therefore, in this paper, we propose to use Taylor expansion for function approximation and it is more straightforward to achieve a higher order approximation of f at anchor points by this way. The embedded vectors, resulted by the function approximation process, will be used as new image representations in our image retrieval framework. In following sections, we will note our **F**unction **A**pproximation-based **emb**edding method as **FAemb**.

The remaining of this paper is organized as follows. Section 2 introduces related background. Section 3 introduces FAemb embedding method. Section 4 presents experimental results. Section 5 concludes the paper.

2. Preliminaries

In this section, we review related background preparing for detail presenting of new embedding method in section 3.

Taylor's theorem for high dimensional variables

Definition 2.1 *Multi-index [8]: A multi-index is a d-tuple of nonnegative integers. Multi-indices are generally denoted by α:*

$$\alpha = (\alpha_1, \alpha_2, \dots, \alpha_d)$$

[2]$\mathbf{C}$ is same for all $\mathbf{x}$.

[3]Although the reconstruction error condition for a good coordinate coding, i.e, $\mathbf{x}'$ closes to $\mathbf{x}$, is not explicit mentioned in original definition of coordinate coding, it can be inferred from objective functions of LCC [32] and TLCC [31].

where $(\alpha_j \in \{0, 1, 2, ...\})$. *If* α *is a multi-index, we define*

$$|\alpha| = \alpha_1 + \alpha_2 + \cdots + \alpha_d; \alpha! = \alpha_1!\alpha_2! \dots \alpha_d!$$

$$\mathbf{x}^\alpha = {x_1}^{\alpha_1} {x_2}^{\alpha_2} \dots {x_d}^{\alpha_d}$$

$$\partial^\alpha f(\mathbf{x}) = \frac{\partial^{|\alpha|} f(\mathbf{x})}{\partial^{\alpha_1}(x_1)\partial^{\alpha_2}(x_2)\dots\partial^{\alpha_d}(x_d)}$$

where $\mathbf{x} = (x_1, x_2, \dots x_d)^T \in \mathbb{R}^d$

Theorem 2.2 (Taylor's theorem for high dimensional variables) [8] *Suppose* $f\colon \mathbb{R}^d \to \mathbb{R}$ *of class of* C^{k+1} [4] *on* $\mathbb{R}^d$. *If* $\mathbf{a} \in \mathbb{R}^d$ *and* $\mathbf{a} + \mathbf{h} \in \mathbb{R}^d$, *then*

$$f(\mathbf{a} + \mathbf{h}) = \sum_{|\alpha| \le k} \frac{\partial^\alpha f(\mathbf{a})}{\alpha!} \mathbf{h}^\alpha + R_{\mathbf{a},k}(\mathbf{h}) \tag{9}$$

where $R_{\mathbf{a},k}(\mathbf{h})$ *is Lagrange remainder given by*

$$R_{\mathbf{a},k}(\mathbf{h}) = \sum_{|\alpha|=k+1} \partial^\alpha f(\mathbf{a} + c\mathbf{h}) \frac{\mathbf{h}^\alpha}{\alpha!} \tag{10}$$

for some $c \in (0, 1)$.

Corollary 2.3 *If* f *is of class of* C^{k+1} *on* $\mathbb{R}^d$ *and* $|\partial^\alpha f(\mathbf{x})| \le M$ *for* $\mathbf{x} \in \mathbb{R}^d$ *and* $|\alpha| = k + 1$, *then*

$$|R_{\mathbf{a},k}(\mathbf{h})| \le \frac{M}{(k+1)!} \|\mathbf{h}\|_1^{k+1} \tag{11}$$

The proof of corollary 2.3 is given in [8]

3. Embedding based on function approximation (FAemb)

In this section, we introduce our embedding method. It is inspired from function approximation based on Taylor's theorem represented in previous section.

3.1. Derivation of FAemb

Lemma 3.1 *If* $f\colon \mathbb{R}^d \to \mathbb{R}$ *is of class of* C^{k+1} *on* $\mathbb{R}^d$ *and* $\nabla^k f(\mathbf{x})$ *is Lipschitz continuous with constant* $M > 0$ *and* $(\gamma(\mathbf{x}), \mathbf{C})$ *is coordinate coding of* $\mathbf{x}$, *then*

$$\left| f(\mathbf{x}) - \sum_{j=1}^{n} \gamma_{\mathbf{v}_j}(\mathbf{x}) \sum_{|\alpha| \le k} \frac{\partial^\alpha f(\mathbf{v}_j)}{\alpha!} (\mathbf{x} - \mathbf{v}_j)^\alpha \right| \le \frac{M}{(k+1)!} \sum_{j=1}^{n} |\gamma_{\mathbf{v}_j}(\mathbf{x})| \, \|\mathbf{x} - \mathbf{v}_j\|_1^{k+1} \tag{12}$$

[4]It means that all partial derivatives of f up to (and including) order $k + 1$ exist and continuous.

The proof of Lemma 3.1 is given in Appendix A.1.

If $k = 1$, then (12) becomes

$$\left| f(\mathbf{x}) - \sum_{j=1}^{n} \gamma_{\mathbf{v}_j}(\mathbf{x}) \left(f(\mathbf{v}_j) + \nabla f(\mathbf{v}_j)^T (\mathbf{x} - \mathbf{v}_j) \right) \right| \le \frac{M}{2} \sum_{j=1}^{n} |\gamma_{\mathbf{v}_j}(\mathbf{x})| \, \|\mathbf{x} - \mathbf{v}_j\|_1^2 \tag{13}$$

In the case of $k = 1$, f is approximated as sum of its weighted tangents at anchor points.

If $k = 2$, then (12) becomes

$$\Bigg| f(\mathbf{x}) - \sum_{j=1}^{n} \gamma_{\mathbf{v}_j}(\mathbf{x}) \Big(f(\mathbf{v}_j) + \nabla f(\mathbf{v}_j)^T (\mathbf{x} - \mathbf{v}_j) + \frac{1}{2} \left(V\left(\nabla^2 f(\mathbf{v}_j)\right) \right)^T V\left((\mathbf{x} - \mathbf{v}_j)(\mathbf{x} - \mathbf{v}_j)^T \right) \Big) \Bigg| \le \frac{M}{6} \sum_{j=1}^{n} |\gamma_{\mathbf{v}_j}(\mathbf{x})| \, \|\mathbf{x} - \mathbf{v}_j\|_1^3 \tag{14}$$

where $V(\mathbf{A})$ is vectorization function flattening the matrix $\mathbf{A}$ to a vector by putting its consecutive columns into a column vector. ∇^2 is Hessian matrix.

In the case of $k = 2$, f is approximated as sum of its weighted quadratic approximations at anchor points.

To achieve a good approximation, the coding $(\gamma(\mathbf{x}), \mathbf{C})$ should be selected such that the RHS of (13) and (14) are small enough.

The result derived from (13) is that, with respect to the coding $(\gamma(\mathbf{x}), \mathbf{C})$, a high dimensional nonlinear function $f(\mathbf{x})$ in $\mathbb{R}^d$ can be approximated by linear form $\mathbf{w}^T\phi(\mathbf{x})$ where $\mathbf{w}$ can be defined as $\mathbf{w} = \left[\frac{1}{s} f(\mathbf{v}_j); \nabla f(\mathbf{v}_j)\right]_{j=1}^{n}$ and the embedded vector $\phi(\mathbf{x})$ can be defined as

$$\phi(\mathbf{x}) = \left[s\gamma_{\mathbf{v}_j}(\mathbf{x}); \gamma_{\mathbf{v}_j}(\mathbf{x})(\mathbf{x} - \mathbf{v}_j) \right]_{j=1}^{n} \in \mathbb{R}^{n(1+d)} \tag{15}$$

where s is a nonnegative scaling factor to balance two types of codes.

To make a good approximation of f, in following sections, we put our interest on case where f is approximated by using up to second-order derivatives defined by (14). The result derived from (14) is that the nonlinear function $f(\mathbf{x})$ can be approximated by linear form $\mathbf{w}^T\phi(\mathbf{x})$ where $\mathbf{w}$ can be defined as $\mathbf{w} = \left[\frac{1}{s_1} f(\mathbf{v}_j); \frac{1}{s_2} \nabla f(\mathbf{v}_j); \frac{1}{2} \left(V\left(\nabla^2 f(\mathbf{v}_j) \right) \right) \right]_{j=1}^{n}$ and the embedded vector $\phi(\mathbf{x})$-FAemb can be defined as

$$\phi(\mathbf{x}) = \Big[s_1\gamma_{\mathbf{v}_j}(\mathbf{x}); s_2\gamma_{\mathbf{v}_j}(\mathbf{x})(\mathbf{x} - \mathbf{v}_j); \gamma_{\mathbf{v}_j}(\mathbf{x}) V\left((\mathbf{x} - \mathbf{v}_j)(\mathbf{x} - \mathbf{v}_j)^T \right) \Big]_{j=1}^{n} \in \mathbb{R}^{n(1+d+d^2)} \tag{16}$$

where s_1, s_2 are nonnegative scaling factors to balance three types of codes.

3.2. Learning of coordinate coding

As mentioned in previous section, to get a good approximation of f, the RHS of (14) should be small enough[5]. Furthermore, from definition of coordinate coding 1.1, $(\gamma(\mathbf{x}), \mathbf{C})$ should ensure that the reconstruction error $\|\mathbf{x}' - \mathbf{x}\|_2$ should be small. Putting two above criteria together, we find $(\gamma(\mathbf{x}), \mathbf{C})$ which minimize the following constrained objective function

$$Q(\gamma(\mathbf{x}), \mathbf{C}) = \|\mathbf{x} - \mathbf{C}\gamma(\mathbf{x})\|_2^2 + \mu \sum_{j=1}^{n} |\gamma_{\mathbf{v}_j}(\mathbf{x})| \, \|\mathbf{x} - \mathbf{v}_j\|_1^3$$
$$st. \; \mathbf{1}^T \gamma(\mathbf{x}) = 1 \tag{17}$$

Equivalently, given a set of training samples (descriptors) $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_m] \in \mathbb{R}^{d \times m}$, let γ_{i_j} be coefficient corresponding to base $\mathbf{v}_j$ of sample $\mathbf{x}_i$; $\gamma_i = [\gamma_{i_1}, \ldots, \gamma_{i_n}]^T \in \mathbb{R}^n$ be coefficient vector of sample $\mathbf{x}_i$; $\Gamma = [\gamma_1, \ldots, \gamma_m] \in \mathbb{R}^{n \times m}$. We find $(\Gamma, \mathbf{C})$ which minimize the following constrained objective function

$$Q(\Gamma, \mathbf{C}) = \sum_{i=1}^{m} \left[\|\mathbf{x}_i - \mathbf{C}\gamma_i\|_2^2 + \mu \sum_{j=1}^{n} |\gamma_{i_j}| \, \|\mathbf{x}_i - \mathbf{v}_j\|_1^3 \right]$$
$$st. \; \mathbf{1}^T \gamma_i = 1, i = 1, \ldots, m \tag{18}$$

To minimize (18), we iteratively optimize it by alternatingly optimizing with respect to $\mathbf{C}$ and Γ while holding the other fixed.

For learning the coefficients Γ, the optimization problem is equivalent to a regularized least squares problem with linear constraint. This problem can be solved by optimizing over each sample $\mathbf{x}_i$ individually. To find γ_i of each sample $\mathbf{x}_i$, we use Newton's method [4]. The gradient and Hessian of objective function w.r.t. γ_i is given in Appendix A.2.

For learning the bases $\mathbf{C}$, the optimization problem is unconstrained regularized least squares. We use trust-region method [6, 5] to solve this problem [6]. The gradient and Hessian of objective function w.r.t. $\mathbf{C}$ is given in Appendix A.2.

After learning $\mathbf{C}$, given a new descriptor $\mathbf{x}$, we get $\gamma(\mathbf{x})$ by minimizing (17) using learned $\mathbf{C}$. From $\gamma(\mathbf{x})$, we get the embedded vector $\phi(\mathbf{x})$-FAemb by using (16).

3.3. Relationship to other methods

The most related embedding methods to FAemb are TLCC [31] and VLAT [26].

Compare to TLCC [31], our assumption on f in lemma 3.1 is slightly different from assumption of TLCC (lemma 2.2 [31]). Our assumption only needs that $\nabla^k f(\mathbf{x})$ is Lipschitz continuous while TLCC assumes that all $\nabla^j f(\mathbf{x})$ are Lipschitz continuous, $j = 1, \ldots, k$. Our objective function (17) is also different from TLCC (4). We rely on L_1 norm of $(\mathbf{x} - \mathbf{v}_j)$ in the second term while TLCC uses L_2 norm. We solve the constraint on the coefficient γ in our optimization process while TLCC does not. FAemb approximates f using up to its second order derivatives while TLCC approximates f only using its first order derivatives.

FAemb can also be seen as the generalization of VLAT [26]. Similar to the relationship of TLCC and VLAD presented in section 1.2, if we add constraint (6) to $\gamma(\mathbf{x})$ then the objective function (18) will become

$$Q_1(\Gamma, \mathbf{C}) = \sum_{i=1}^{m} \left[\|\mathbf{x}_i - \mathbf{v}_*\|_2^2 + \mu \|\mathbf{x}_i - \mathbf{v}_*\|_1^3 \right]$$
$$st. \; \mathbf{1}^T \gamma_i = 1, i = 1, \ldots, m$$
$$\|\gamma_i\|_0 = 1, i = 1, \ldots, m \tag{19}$$

where $\mathbf{v}_*$ is anchor point corresponding to nonzero element of γ_i.

If we relax L_1 norm in the second term of $Q_1(\Gamma, \mathbf{C})$ into L_2 norm, then we can use K-means algorithm for minimizing (19). After learning $\mathbf{C}$ by using K-means, given an input descriptor $\mathbf{x}$, we have

$$\mathbf{x} \approx \mathbf{v}_* = \underset{\mathbf{v} \in \mathbf{C}}{\operatorname{argmin}} \|\mathbf{x} - \mathbf{v}\|_2 \tag{20}$$

Now, consider (16), if we choose $s_1 = 0, s_2 = 0$ and we remove zero elements attached with them, $\phi(\mathbf{x}) = \left[0, \ldots, 0, \left(V\left((\mathbf{x} - \mathbf{v}_*)(\mathbf{x} - \mathbf{v}_*)^T\right)\right)^T, 0, \ldots, 0\right]^T \in \mathbb{R}^{nd^2}$ will become VLAT.

In practice, to make a fair comparison between FAemb and VLAT, the embedded vectors producing by two methods should have same dimension. To ensure this, we choose s_1, s_2 in (16) equal to 0. It is worth noting that in (16), as matrix $(\mathbf{x} - \mathbf{v}_j)(\mathbf{x} - \mathbf{v}_j)^T$ is symmetric, only the diagonal and upper part are kept while flattening it into vector. The size of VLAT and FAemb is then $\frac{nd(d+1)}{2}$.

3.4. Whitening and aggregating embedded vectors to single vector

3.4.1 Whitening

In [14], authors showed that by applying the whitening processing, the discriminating of embedded vectors can be improved, hence improving the retrieval results.

In particular, given $\phi(\mathbf{x}) \in \mathbb{R}^D$, we achieve whitened embedded vectors $\phi_w(\mathbf{x})$ by

$$\phi_w(\mathbf{x}) = diag\left(\lambda_1^{-\frac{1}{2}}, \ldots, \lambda_D^{-\frac{1}{2}}\right) P^T \phi(\mathbf{x}) \tag{21}$$

where λ_i is ith largest eigenvalue. $P \in \mathbb{R}^{D \times D}$ is matrix formed by the largest eigenvectors associated with the

[5]Because $\frac{M}{6}$ is constant, it can be ignored in the optimization process.

[6]Because the objective function involves L_1 norm, some methods designed for L_1 regularization, i.e, feature-sign search algorithm [16], can be used. However, we find that the Newton's method (for computing Γ) and the trust-region method (for computing $\mathbf{C}$) work well in practice.

largest eigenvalues of the covariance matrix computed from learning embedded vectors $\phi(\mathbf{x})$.

[14] further suggested that by discarding some first components associated with the largest eigenvalues of $\phi_w(\mathbf{x})$, the localization of whitened embedded vectors will be improved. In practice, we also apply this truncation operation. The detail of this truncation operation is presented in section 4.

3.4.2 Aggregating

Let $\mathcal{X} = \{\mathbf{x}\}$ be set of local descriptors describing the image. Sum-pooling [15] and max-pooling [30, 3] are two common methods for aggregating set of whitened embedded vectors $\phi_w(\mathbf{x})$ of the image to a single vector. Sum-pooling lacks discriminability because the aggregated vector is more influenced by frequently-occurring uninformative descriptors than rarely-occurring informative ones. Max-pooling equalizes the influence of frequent and rare descriptors. However, classical max-pooling approaches can only be applied to BoW or sparse coding features. Recently, [14] introduced a new aggregating method named *democratic aggregation* applied to image retrieval problem. This method bears similarity to generalized max-pooling [19] applied to image classification problem. Democratic aggregation can be applied to general features such as VLAD, Temb, Fisher vector. [14] showed that democratic aggregation achieves better performance than sum-pooling. The main idea of democratic aggregation is to find a weight for each $\phi_w(\mathbf{x})$ such that $\forall \mathbf{x}_i \in \mathcal{X}$

$$\lambda_i \left(\phi_w(\mathbf{x}_i)\right)^T \sum_{\mathbf{x}_j \in \mathcal{X}} \lambda_j \phi_w(\mathbf{x}_j) = 1 \tag{22}$$

Generally, the process to produce the single vector from set of local descriptors describing the image is as follows. First, we map each $\mathbf{x} \in \mathcal{X} \to \phi(\mathbf{x})$ and whitening $\phi(\mathbf{x})$, producing $\phi_w(\mathbf{x})$. We then use democratic aggregation to aggregate vectors $\phi_w(\mathbf{x})$ to the single vector ψ by

$$\psi(\mathcal{X}) = \sum_{\mathbf{x}_i \in \mathcal{X}} \lambda_i \phi_w(\mathbf{x}_i) \tag{23}$$

4. Experiments

This section presents results of our FAemb embedding method. In section 4.3, we compare FAemb to other three methods: VLAD [13], Temb [14] and VLAT [26]. We reimplement VLAD and VLAT in our framework. For Temb, we use the source code provided by [14]. To make a fair comparison, the whitening and the aggregating presented in section 3.4 are applied for all four embedding methods. As suggestion in [14], for Temb and VLAD methods, we discard d first components of $\phi_w(\mathbf{x})$. The final dimension of $\phi_w(\mathbf{x})$ is therefore $D = (n-1) \times d$. For VLAT and FAemb methods, we discard $\frac{d\times(d+1)}{2}$ first components of $\phi_w(\mathbf{x})$. The final dimension of $\phi_w(\mathbf{x})$ is therefor $D = \frac{(n-1)d(d+1)}{2}$.

In section 4.4, we compare our framework with image retrieval benchmarks.

The value of μ in (18) is selected by empirical experiments and is fixed to 10^{-3} for all FAemb results reported bellow.

4.1. Dataset and evaluation protocol

INRIA holidays [11] consists of 1491 high resolution images containing personal holiday photos with 500 queries. The search quality is measured by mean average precision (mAP) over 500 queries, with the query removed from the ranked list. As standardly done in the literature, for all the learning stages, we use the independent dataset Flickr60k provided with Holidays.

Oxford buildings (Oxford5k) [24] consists of 5062 images of buildings and 55 query images corresponding to 11 distinct buildings in Oxford. The search quality is measured by mAP computed over the 55 queries. Images are annotated as either relevant, not relevant, or junk, which indicates that it is unclear whether a user would consider the image as relevant or not. We follow same configuration in [7, 14, 12] where the junk images are removed from the ranking before computing the mAP. As standardly done in the literature, for all the learning stages, we use the Paris6k dataset [25].

4.2. Implementation notes

Local descriptors are detected using the Hessian-affine detector [18] and described by the SIFT local descriptor [17]. We used RootSIFT variant [1] in all our experiments.

For VLAT and FAemb, at beginning, all SIFT descriptors are reduced from 128 to 45 dimensions using PCA. This makes the dimension of VLAT and FAemb comparable to dimension of compared embedding methods.

Power-law normalization. The problem of burtiness visual elements is first introduced in [10]: numerous descriptors almost similar within the same image. This phenomenon strongly affects the measure of similarity between two images. To reduce the effect of burtiness, we similarly do as previous works [12, 14]: applying power-law normalization [23] to the final image representation ψ and subsequently L_2 normalize it. The applying of power-law normalization to each component a of ψ is done by $a := |a|^\alpha sign(a)$, where $0 \leq \alpha \leq 1$ is a constant. To ensure a fair comparison, for each embedding method, we run experiments with $\alpha = \{1, 0.9, ..., 0.1, 0\}$ and report the best mAP.

Table 1. The comparison between the implementation of VLAD and VLAT in this paper and their improved versions [13, 20] on Holidays dataset. D is final dimension of aggregated vectors. Reference results are obtained from corresponding papers.

method	D	mAP
VLAD [13]	16,384	58.7
VLAD (this paper)	8,064	67.4
VLAD (this paper)	16,256	68.3
VLAT$_{\text{improved}}$ [20]	9,000	70.0
VLAT (this paper)	7,245	70.9
VLAT (this paper)	15,525	72.7

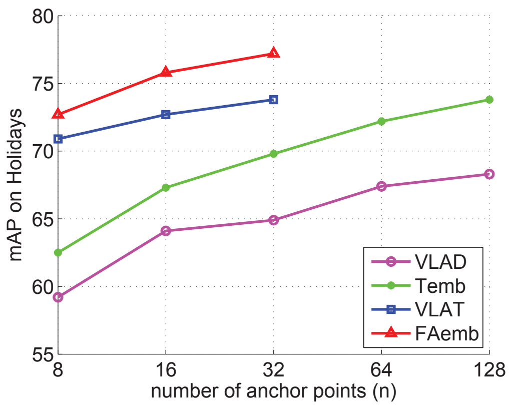

Figure 1. Impact of number of anchor points on the Holidays dataset for different embedding methods: VLAD, Temb, VLAT and our FAemb. Given n, the dimension of VLAD and Temb is $128 \times (n-1)$; the dimension of VLAT and FAemb is $\frac{45\times46}{2} \times (n-1)$.

4.3. Impact of parameters and comparison of methods

It is worth noting that even with a lower dimension, the implementation of VLAD and VLAT in our framework (RootSIFT descriptors, VLAD/VLAT embedding, whitening, democratic aggregation and power-law normalization) achieves better retrieval results than their improved versions reported by the authors [13, 20]. The comparison on Holidays dataset is shown in Table 1.

Impact of parameters: the main parameter here is number of anchor points n. The analysis for this parameter is shown in Figure 1 and Figure 2 for Holidays and Oxford5k datasets, respectively. We can see that the mAP increases with the increasing of n for all four methods. For Temb, VLAT and FAemb, the improvement tends to be smaller for larger n. For VLAT and FAemb, when $n > 32$, the improvement in mAP is not worth the computation overhead.

Comparison of methods: we find that the following observations are consistent on both Holidays and Oxford5k datasets.

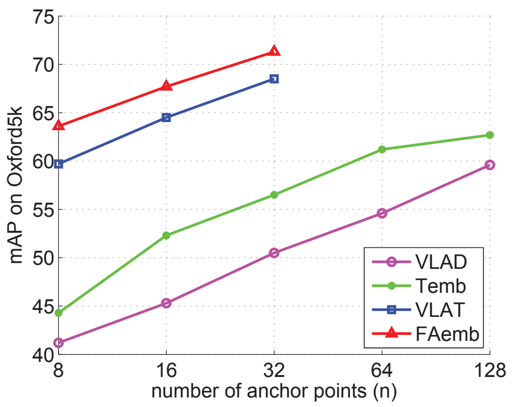

Figure 2. Impact of number of anchor points on the Oxford5k dataset for different embedding methods: VLAD, Temb, VLAT and our FAemb. Given n, the dimension of VLAD and Temb is $128 \times (n-1)$; the dimension of VLAT and FAemb is $\frac{45\times46}{2} \times (n-1)$.

- For same n, FAemb and VLAT have same dimension. However, FAemb improves the mAP over VLAT by a fair margin. When $n = 8$, the improvement is **+1.8%** and **+3.9%** on Holidays and Oxford5k, respectively. When $n = 16, 32$, the improvement is about **+3%** on both datasets.
- When the dimension is comparable, FAemb significantly improves the mAP over VLAD and Temb. For examples, comparing FAemb at $(n = 16, D = 15,525)$ with VLAD/Temb at $(n = 128, D = 16,256)$, the gain of FAemb over VLAD/Temb is **+7.5%/+2%** on Holidays and **+8.1%/+5%** on Oxford5k.

4.4. Comparison with the state of the art

In this section, we compare our framework with benchmarks having similar representation, *i.e.*, they represent an image by a single vector. The main differences between compared frameworks are shown in Table 2. Excepting VLAT$_{\text{improved}}$ [20], other compared methods and ours consist of power-law normalization step and use Euclidean distance when comparing the aggregated vectors. VLAT$_{\text{improved}}$ [20] doesn't have power-law normalization and it uses Mahalanobis distance when comparing the aggregated vectors. VLAD$_{\text{LCS}}$ [7] and VLAT$_{\text{improved}}$ [20] don't have whitening step on final embedded vector ($\phi(\mathbf{x})$) but they first apply PCA on Voronoi cells separately. The sub-embedded vectors on Voronoi cells are then concatenated to form final embedded vector.

Our framework, by itself, outperforms the state of the art by introducing new effective embedding method, and by combining most of effective ingredients.

Table 2. The difference between compared frameworks. The frameworks are named by embedding methods used. RSIFT means RootSIFT. *Do whitening* means if whitening is applied on embedded vectors.

Frame work	Local desc.	Do whitening?	Aggr. method
BoW [13]	SIFT	No	Sum
VLAD [13]	SIFT	No	Sum
Fisher [13]	SIFT	No	Sum
VLAD_{LCS} [7]	RSIFT	No	Sum
$\text{VLAD}_{\text{intra}}$ [2]	RSIFT	No	Sum
$\text{VLAT}_{\text{improved}}$ [20]	SIFT	No	Sum
Temb [14]	RSIFT	Yes	Democratic
Ours (FAemb)	RSIFT	Yes	Democratic

Table 3. Comparison with the state of the art on Holidays and Oxford5k datasets. The frameworks are named by embedding methods used. n is number of anchor points. D is dimension of embedded vectors. Reference results are obtained from corresponding papers.

Frame work	n	D	mAP Hol.	mAP Ox5k
VLAD [13]	256	16,384	58.7	-
Fisher [13]	256	16,384	62.5	-
VLAD_{LCS} [7]	64	8,192	65.8	51.7
$\text{VLAD}_{\text{intra}}$ [2]	64	8,192	56.5	44.8
$\text{VLAD}_{\text{intra}}$ [2]	256	32,536	65.3	55.8
$\text{VLAT}_{\text{improved}}$ [20]	64	9,000	70.0	-
Temb [14]	64	8,064	72.2	61.2
Temb [14]	128	16,256	73.8	62.7
Our framework				
FAemb	8	7,245	72.7	63.6
FAemb	16	15,525	**75.8**	**67.7**

Table 3 shows that our framework outperforms the compared frameworks by a large margin on both datasets. The gain over recent improved VLAD [2] having a high (32,536) dimension is **+10.5%** on Holidays and **+11.9%** on Oxford5k. Comparing with $\text{VLAT}_{\text{improved}}$ [20] which is the latest version of VLAT, the gain is **+5.8%** on Holidays. Even with a lower dimension, we ($D = 7,245$) outperform $\text{VLAT}_{\text{improved}}$ ($D = 9,000$) **+2.7%**. Comparing with the latest embedding method (Temb) [14], we also achieve a gain **+2%** on Holidays and **+5%** on Oxford5k.

In Temb embedding [14], to suppress the influence of co-occurrences descriptors that corrupts the similarity measure [9], they applied (before power-law normalization) rotation postprocessing introduced in [27] on aggregated vectors. For instance, they rotate data with a PCA rotation matrix learned on aggregated vectors from learning set. This rotation postprocessing is a complementary operation and it boosts the performance. In this section, we also show results when this operation is applied on our FAemb. To make a fair comparison with results of Temb [14], we use the same number of learning images as [14]. They are 10k images from Flickr60k for Holidays and 6k images from Paris6k for Oxford5k. The results with the applying of this rotation are noted as +RN, and shown in Table 4.

Table 4. Comparison between Temb [14] and FAemb on Holidays and Oxford5k datasets when rotation postprocessing is applied on the aggregated vector. n is number of anchor points. D is dimension of embedded vectors. Reference results are obtained from corresponding paper.

Method	n	D	mAP Hol.	mAP Ox5k
Temb + RN [14]	64	8,064	77.1	67.5
Temb + RN [14]	128	16,256	76.8	66.5
FAemb + RN	8	7,245	76.2	66.7
FAemb + RN	16	15,525	**78.7**	**70.9**

We can see that the applying of the rotation normalization to Temb and FAemb gives a large improvement in performance. The mAP of FAemb+RN at $D = 7,245$ is slightly lower than Temb+RN at $D = 8,064$ on both datasets. However, we note a larger variance: the best results of FAemb+RN are higher than the best results of Temb+RN, especially on Oxford5k. For instance, the gain is **+1.6%** on Holidays and **+3.4%** on Oxford5k.

5. Conclusion

By departing from the goal of linear approximation of a nonlinear function in high dimensional space, this paper proposes a new powerful embedding method for image retrieval problem. The proposed embedding method-FAemb can be seen as the generalization of several well-known embedding methods such as VLAD, TLCC, VLAT. The new presentation compares favorably with state-of-the-art embedding methods for image retrieval, such as VLAD, VLAT, Fisher kernel, Temb, even with a shorter presentation.

A. Appendix

A.1. Proof of Lemma 3.1

Because $\nabla^k f(\mathbf{x})$ is Lipschitz continuous with constant $M > 0$, we have $\left\|\nabla^{k+1} f(\mathbf{x})\right\|_2 \leq M$. So for $|\alpha| = k+1$, we have $|\partial^\alpha f(\mathbf{x})| \leq \left\|\nabla^{k+1} f(\mathbf{x})\right\|_2 \leq M$.

We have

$$\begin{aligned}
&\left| f(\mathbf{x}) - \sum_{j=1}^{n} \gamma_{\mathbf{v}_j}(\mathbf{x}) \sum_{|\alpha| \le k} \frac{\partial^\alpha f(\mathbf{v}_j)}{\alpha!} (\mathbf{x} - \mathbf{v}_j)^\alpha \right| \\
= &\left| \sum_{j=1}^{n} \gamma_{\mathbf{v}_j}(\mathbf{x}) \left(f(\mathbf{x}) - \sum_{|\alpha| \le k} \frac{\partial^\alpha f(\mathbf{v}_j)}{\alpha!} (\mathbf{x} - \mathbf{v}_j)^\alpha \right) \right| \\
\le &\sum_{j=1}^{n} \left| \gamma_{\mathbf{v}_j}(\mathbf{x}) \left(f(\mathbf{x}) - \sum_{|\alpha| \le k} \frac{\partial^\alpha f(\mathbf{v}_j)}{\alpha!} (\mathbf{x} - \mathbf{v}_j)^\alpha \right) \right| \\
= &\sum_{j=1}^{n} |\gamma_{\mathbf{v}_j}(\mathbf{x})| \, |R_{\mathbf{v}_j,k}(\mathbf{x} - \mathbf{v}_j)| \\
\le &\frac{M}{(k+1)!} \sum_{j=1}^{n} |\gamma_{\mathbf{v}_j}(\mathbf{x})| \, \|\mathbf{x} - \mathbf{v}_j\|_1^{k+1} .
\end{aligned}$$

where the last inequation comes from corollary 2.3.

A.2. Gradient and Hessian w.r.t. $\gamma_i, \mathbf{C}$ of objective function $Q(\Gamma, \mathbf{C})$ [7]

We have the objective function

$$Q(\Gamma, \mathbf{C}) = \sum_{i=1}^{m} \left[\|\mathbf{x}_i - \mathbf{C}\gamma_i\|_2^2 + \mu \sum_{j=1}^{n} |\gamma_{i_j}| \, \|\mathbf{x}_i - \mathbf{v}_j\|_1^3 \right]$$

A.2.1 Gradient and Hessian w.r.t. γ_i

Let $\mathbf{a} = \left[\|\mathbf{x}_i - \mathbf{v}_1\|_1^3, \|\mathbf{x}_i - \mathbf{v}_2\|_1^3, \dots, \|\mathbf{x}_i - \mathbf{v}_n\|_1^3 \right]^T$, we have

$$\nabla Q(\gamma_i) = 2\mathbf{C}^T(\mathbf{C}\gamma_i - \mathbf{x}_i) + \mu \, sign(\gamma_i) \odot \mathbf{a} \quad (24)$$
$$\nabla^2 Q(\gamma_i) = 2\mathbf{C}^T\mathbf{C} \quad (25)$$

where $sign(\gamma_i) = [sign(\gamma_{i_1}), sign(\gamma_{i_2}), \dots, sign(\gamma_{i_n})]^T$ and $\odot$ denotes Hadamard product.

A.2.2 Derivative and Hessian w.r.t. C

Let $R = \sum_{i=1}^{m} \|\mathbf{x}_i - \mathbf{C}\gamma_i\|_2^2 = \|\mathbf{X} - \mathbf{C}\Gamma\|_2^2$, we have

$$\nabla R(\mathbf{C}) = 2(\mathbf{C}\Gamma - \mathbf{X})\Gamma^T \quad (26)$$

Let $L = \sum_{j=1}^{n} \sum_{i=1}^{m} |\gamma_{i_j}| \, \|\mathbf{x}_i - \mathbf{v}_j\|_1^3$ and let $\mathbf{d}_j = \nabla L(\mathbf{v}_j) = 3\sum_{i=1}^{m} |\gamma_{i_j}| \, \|\mathbf{v}_j - \mathbf{x}_i\|_1^2 \, sign(\mathbf{v}_j - \mathbf{x}_i)$, we have

$$\nabla L(\mathbf{C}) = [\mathbf{d}_1, \dots, \mathbf{d}_j, \dots, \mathbf{d}_n] \quad (27)$$

[7] Theoretically, the partial derivatives $\partial(Q)/\partial\gamma_{i_k}$ and $\partial(Q)/\partial\mathbf{v}_{j_k}$ do not exist at some points. We found, however, the Newton's method and the trust-region method with the provided derivatives work well in practice.

Finally, we get

$$\nabla Q(\mathbf{C}) = \nabla R(\mathbf{C}) + \mu \nabla L(\mathbf{C}) \quad (28)$$

Let $u_j = \sum_{i=1}^{m} \gamma_{i_j}^2$: sum of square of coefficients corresponding to base $\mathbf{v}_j$ of all data points $\mathbf{x}$; let $\mathbf{A}_{jj} = 2u_j\mathbf{I}_{d\times d} \in \mathbb{R}^{d\times d}, j = 1, \dots, n$, we have

$$\nabla^2 R(\mathbf{C}) = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{0}_{d\times d} & \dots & \mathbf{0}_{d\times d} \\ \mathbf{0}_{d\times d} & \mathbf{A}_{22} & \dots & \mathbf{0}_{d\times d} \\ \vdots & \vdots & \ddots & \mathbf{0}_{d\times d} \\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} & \dots & \mathbf{A}_{nn} \end{pmatrix} \quad (29)$$

where $\mathbf{0}_{d\times d}$ is matrix having size of $d \times d$ and zero elements.

Let $\mathbf{B}_{jj} = \nabla^2 L(\mathbf{v}_j) \in \mathbb{R}^{d\times d}$ be Hessian of L w.r.t. base $\mathbf{v}_j, j = 1, \dots, n$, we have

$$\mathbf{B}_{jj} = \begin{pmatrix} \frac{\partial^2 L}{\partial\mathbf{v}_{j_1}\partial\mathbf{v}_{j_1}} & \frac{\partial^2 L}{\partial\mathbf{v}_{j_1}\partial\mathbf{v}_{j_2}} & \dots & \frac{\partial^2 L}{\partial\mathbf{v}_{j_1}\partial\mathbf{v}_{j_d}} \\ \frac{\partial^2 L}{\partial\mathbf{v}_{j_2}\partial\mathbf{v}_{j_1}} & \frac{\partial^2 L}{\partial\mathbf{v}_{j_2}\partial\mathbf{v}_{j_2}} & \dots & \frac{\partial^2 L}{\partial\mathbf{v}_{j_2}\partial\mathbf{v}_{j_d}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 L}{\partial\mathbf{v}_{j_d}\partial\mathbf{v}_{j_1}} & \frac{\partial^2 L}{\partial\mathbf{v}_{j_d}\partial\mathbf{v}_{j_2}} & \dots & \frac{\partial^2 L}{\partial\mathbf{v}_{j_d}\partial\mathbf{v}_{j_d}} \end{pmatrix} \quad (30)$$

For $k = 1, \dots, d$; $h = 1, \dots, d$; if $k = h$ then

$$\frac{\partial^2 L}{\partial\mathbf{v}_{j_k}\partial\mathbf{v}_{j_h}} = 6\sum_{i=1}^{m} |\gamma_{i_j}| \, \|\mathbf{v}_j - \mathbf{x}_i\|_1 \left(sign(\mathbf{v}_{j_k} - \mathbf{x}_{i_k})\right)^2$$

If $k \neq h$ then

$$\frac{\partial^2 L}{\partial\mathbf{v}_{j_k}\partial\mathbf{v}_{j_h}} = 6\sum_{i=1}^{m} |\gamma_{i_j}| \, \|\mathbf{v}_j - \mathbf{x}_i\|_1 \, sign(\mathbf{v}_{j_k} - \mathbf{x}_{i_k}) sign(\mathbf{v}_{j_h} - \mathbf{x}_{i_h})$$

We have

$$\nabla^2 L(\mathbf{C}) = \begin{pmatrix} \mathbf{B}_{11} & \mathbf{0}_{d\times d} & \dots & \mathbf{0}_{d\times d} \\ \mathbf{0}_{d\times d} & \mathbf{B}_{22} & \dots & \mathbf{0}_{d\times d} \\ \vdots & \vdots & \ddots & \mathbf{0}_{d\times d} \\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} & \dots & \mathbf{B}_{nn} \end{pmatrix} \quad (31)$$

Finally, we get

$$\nabla^2 Q(\mathbf{C}) = \nabla^2 R(\mathbf{C}) + \mu \nabla^2 L(\mathbf{C}) \quad (32)$$

References

[1] R. Arandjelovic and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, 2012.

[2] R. Arandjelovic and A. Zisserman. All about VLAD. In *CVPR*, 2013.

[3] Y. Boureau, J. Ponce, and Y. LeCun. A theoretical analysis of feature pooling in visual recognition. In *ICML*, 2010.

[4] S. Boyd and L. Vandenberghe. *Convex optimization, p. 528*. Cambridge university press, 2004.

[5] T. F. Coleman and Y. Li. On the convergence of interior-reflective newton methods for nonlinear minimization subject to bounds. *Mathematical programming*, pages 189–224, 1994.

[6] T. F. Coleman and Y. Li. An interior trust region approach for nonlinear minimization subject to bounds. *SIAM Journal on optimization*, pages 418–445, 1996.

[7] J. Delhumeau, P. H. Gosselin, H. Jégou, and P. Pérez. Revisiting the VLAD image representation. In *MM*, 2013.

[8] G. B. Folland. *Advanced Calculus*. Prentice Hall, 1st edition, 2002.

[9] H. Jégou and O. Chum. Negative evidences and co-occurences in image retrieval: The benefit of PCA and whitening. In *ECCV*, 2012.

[10] H. Jégou, M. Douze, and C. Schmid. On the burstiness of visual elements. In *CVPR*, 2009.

[11] H. Jégou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. *IJCV*, pages 316–336, 2010.

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

[13] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid. Aggregating local images descriptors into compact codes. *PAMI*, 2012.

[14] H. Jégou and A. Zisserman. Triangulation embedding and democratic aggregation for image search. In *CVPR*, 2014.

[15] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[16] H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In *NIPS*, 2006.

[17] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, pages 91–110, 2004.

[18] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *IJCV*, pages 63–86, 2004.

[19] N. Murray and F. Perronnin. Generalized max pooling. In *CVPR*, 2014.

[20] R. Negrel, D. Picard, and P. H. Gosselin. Web-scale image retrieval using compact tensor aggregation of visual descriptors. *IEEE Transactions on Multimedia*, 2013.

[21] B. A. Olshausen and D. J. Fieldt. Sparse coding with an overcomplete basis set: a strategy employed by v1. *Vision Research*, pages 3311–3325, 1997.

[22] F. Perronnin and C. R. Dance. Fisher kernels on visual vocabularies for image categorization. In *CVPR*, 2007.

[23] F. Perronnin, J. Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *ECCV*, 2010.

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

[25] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *CVPR*, 2008.

[26] D. Picard and P. H. Gosselin. Improving image similarity with vectors of locally aggregated tensors. In *ICIP*, 2011.

[27] B. Safadi and G. Quénot. Descriptor optimization for multimedia indexing and retrieval. In *CBMI*, 2013.

[28] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003.

[29] J. Wang, J. Yang, K. Yu, F. Lv, T. S. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *CVPR*, 2010.

[30] J. Yang, K. Yu, Y. Gong, and T. S. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *CVPR*, 2009.

[31] K. Yu and T. Zhang. Improved local coordinate coding using local tangents. In *ICML*, 2010.

[32] K. Yu, T. Zhang, and Y. Gong. Nonlinear learning using local coordinate coding. In *NIPS*, 2009.

[33] X. Zhou, K. Yu, T. Zhang, and T. S. Huang. Image classification using super-vector coding of local image descriptors. In *ECCV*, 2010.