Global Image Representation Using Locality-constrained Linear Coding for Large-Scale Image Retrieval

Advisor: Prof. Wen-Hsiao Peng

Student: Yu-Hsing Wu

Institute of Multimedia Engineering

National Chiao Tung Univeristy

1001 Ta-Hsueh Rd., 30010 HsinChu, Taiwan

December 2013

Contents

List of Tables

List of Figures

CHAPTER 1

Research Overview

1.1 Introduction

Large-scale image retrieval has become a hot topic recently due to the need to retrieve similar images (to a query image) from ever-growing image collections. For searching within huge databases, a fast yet efficient approach is the use of global image representation (also known as global descriptor), which usually takes the form of a high-dimensional vector and enables simple similarity measures, such as Cosine Similarity, to be utilized for image comparison.

Bag-of-words (BoW) [12] is a popular model for global image representation. Its process (see Fig. 1.1) usually includes:

1. *extracting* local descriptors, e.g. SIFT, from an image;
2. *encoding* every local descriptor into a vector representation with one non-zero entry indicating to which visual word in a pre-defined codebook it is quantized; and
3. *pooling* the vector representations for different local descriptors into one single vector.

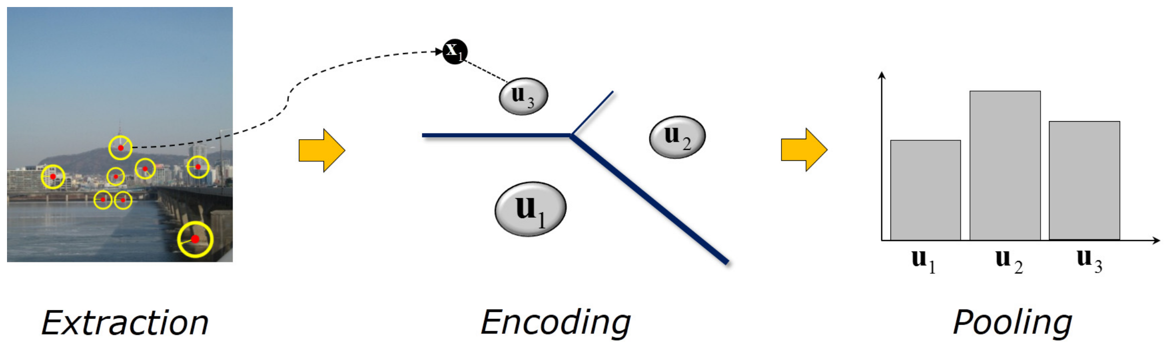


Figure 1.1: Illustration of BoW formation. $\mathbf{x}_i$ denotes local descriptor and $\mathbf{u}_j$ denotes visual word.

The result is essentially a histogram of visual words, characterizing the frequency of their occurrence in an image. Among these steps, the encoding is the most critical step: the resulting vector needs to be discriminative enough to enable reliable retrieval. It was shown however that the nearest-neighbor-based visual word assignment in the current BoW method can yield very different encoding results for local descriptors that are actually similar to each other, a phenomenon often referred to as the ambiguity problem [13].

To alleviate the ambiguity, soft assignment schemes were proposed [10][11], where a local descriptor can be assigned to multiple visual words with proper weighting. In [11] the weighting coefficient to associate with a visual word is determined heuristically based on its distance to the encoding descriptor, while in [10] it is computed as a posterior probability, assuming that the local descriptor follows a Gaussian mixture probability model (GMM) with the means of different components representing the visual words. The latter is a particular case of the more general Fisher vector (FV) representation, which encodes an observed local descriptor by taking the derivatives of its likelihood function with respect to all or some of the model parameters. Interestingly, the FV representation has many important analytical properties, which provide theoretical justification for many other heuristic schemes. Showing promising retrieval performance, it was recently considered by the ISO/IEC MPEG committee for standardization of compact descriptors for visual search [5].

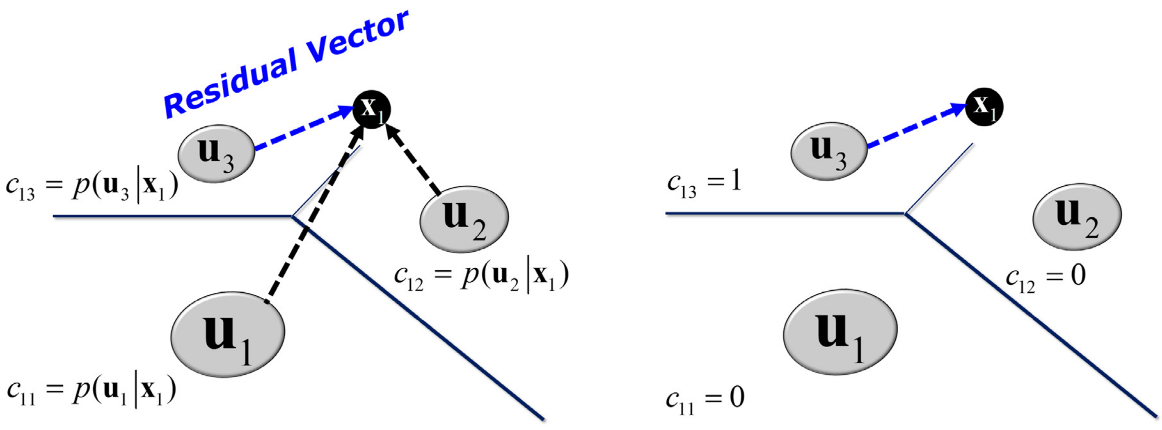


Figure 1.2: The difference of encoding process between FV and VLAD. $\mathbf{c}_{ij}$ denotes the weighting of local descriptor $\mathbf{x}_i$ assigned to visual word $\mathbf{u}_j$.

1.2 Problem Statement

The FV representation however suffers from high computational complexity. In its most popular form [10], the encoding of a local descriptor requires evaluating the posterior probability and computing a normalized residual vector with respect to every visual word (i.e., component). To reduce computations, the vector of locally aggregated descriptors (VLAD) [7] simply obtains the residual vector to the nearest visual word (see Fig. 1.2) and sets its weight to one and the others to zero (just like BoW). Not surprisingly, it inherits the same ambiguity problem from BoW.

1.3 Contributions

In seeking a better trade-off between complexity and performance, this work applies the locality-constrained linear coding (LLC) [14], a technique proven to be effective for image classification, to encoding local descriptors. As will be seen, its soft assignment nature and locality property ensure that similar local descriptors always have similar encoding results, while the sparseness of the resulting code allows only few weighting coefficients to be computed. Empirical studies were conducted to determine the best form of LLC for image retrieval applications. Specifically, our main contributions in this work include the following:

- We conduct several empirical studies to investigate the effects and benefits of locality property and to adapt the other terms in FV for a better trade-off between performance and complexity.

- We propose a global image representation using LLC-based coding with normalized residual vector.
- We make a few simplifications to FV in hopes of getting a better trade-off between complexity and performance.

Under the test conditions [5] suggested by the MPEG committee, the proposed scheme provides 0.7-3.9% improvements in mean Average Precision (mAP) and reduces almost by half of average encoding time, when compared with the state-of-the-art approaches. Part of our design parallels another independently developed work, Robust Visual Descriptor (RVD) [6]. In addition, the simplified FV not only reduces the encoding time of FV by half, but also improves its performance slightly (1-1.5%).

1.4 Organization of Thesis

The rest of this thesis is organized as follows: Chapter 2 describes the large-scale image retrieval architecture and several state-of-the-art methods to ease the understanding of this work. Chapter 3 presents a series of empirical studies for determining the best form of LLC. Chapter 4 shows performance comparison with other state-of-the-art approaches. Finally, Chapter 5 concludes this thesis.

CHAPTER 2

Background

In this chapter, we describe the large-scale image retrieval architecture and the existing encoding methods for generating global image representation. They will be explained in detail in the following sections.

2.1 Large-Scale Image Retrieval

This architecture is proposed to deal with the large-scale image retrieval task, which is a two-level retrieval pipeline (see Fig. 2.1). It first uses global descriptor to produce a shortlist of similar images in database, and then applies a geometric verification, where local descriptors were matched, to re-ranking the shortlist. We will describe all the components step by step in the following subsections. Note that the ISO/IEC MPEG Compact Descriptors for Visual Search (CDVS) also adopted this architecture.

2.1.1 Compact Descriptor Extraction

This subsection describes how to generate two different kinds of compact descriptors, local descriptor and global descriptor. The procedure includes 1) feature detection; 2)

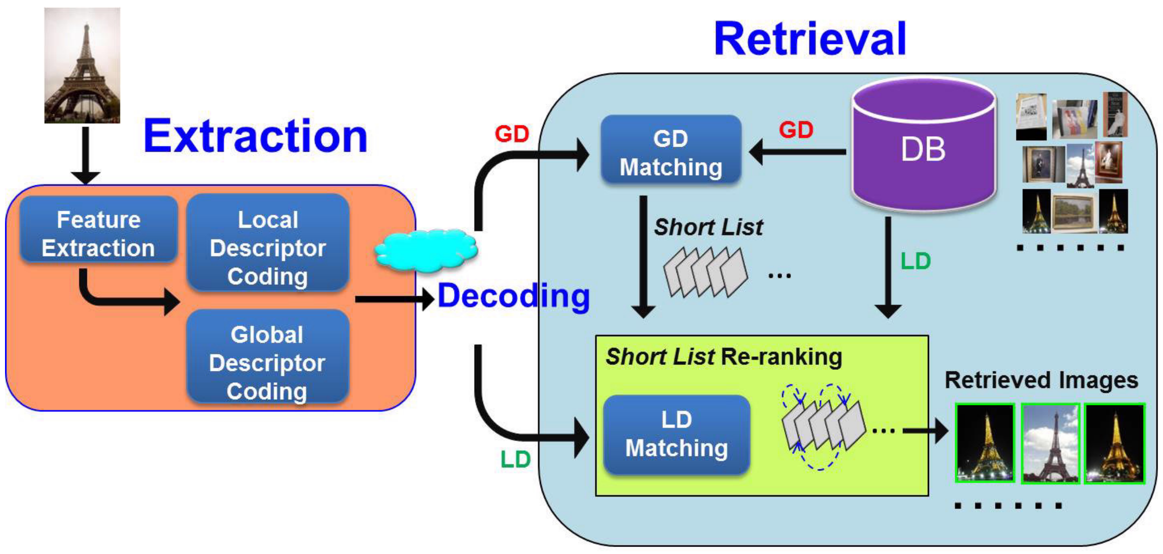


Figure 2.1: Large-scale image retrieval architecture

feature description; 3) feature selection; and 4) feature compression.

For the first two steps, we use scale-invariant feature transform (SIFT) [9], an algorithm in computer vision to detect and describe local features in an image. It has been widely used in image matching problems such as image classification, image retrieval and object detection for its excellent properties. Those properties include invariance to image scaling and rotation, and also partially invariance to illumination and viewpoint change. For the third step, we then select a limited number of SIFT features so that this subset is more possible to be correctly matched.

For the fourth step, before the retrieval procedure, we need to encode those selected SIFT features into compact codes. For local descriptor, each SIFT feature is transformed [5] and scalar quantized to make it a ternary representation. Also, we need to record their coordinates in order to be utilized for geometric verification. As for global descriptor, Principal Component Analysis (PCA) is first employed to reduce the dimension of each raw SIFT feature from 128-dim to 32-dim. It benefits the global image representation in effectively removing redundant information in raw SIFT features. Each 32-dim feature is then encoded into a high dimensional vector. Finally all the vectors are aggregated into a single high dimensional vector. To further compress, we employ a simple sign function to binarize the representation. That is, for each dimension of the representation, we assign the value "1" to any non-negative value, and the value "0" to any negative value.

2.1.2 Retrieval

In large-scale image retrieval, a two-stage retrieval framework was proposed to accelerate the retrieval process. At the first stage, we use the global descriptor to calculate the similarity scores between query image and the dataset images, and get the shortlist from the top of the ranked list. At the second stage, the local descriptors of the query image are matched to the local descriptors of the shortlist images, and the geometric verification is applied for re-ranking the shortlist as a secondary criteria.

In this thesis, we focus on the encoding methods for generating global descriptor, and try to improve the reliability of the shortlist.

2.2 Related Works

This section reviews several encoding schemes for generating global image representation. We review five state-of-the-art encoding schemes, including bag-of-words (BoW) [12], Fisher vector (FV) [5], vector of locally aggregated descriptor (VLAD) [7], Robust Visual Descriptor (RVD) [6] and locality-constrained linear coding (LLC) [14]. For notation, we shall use $X = \left\{\mathbf{x}_i | \mathbf{x}_i \in \mathbb{R}^D,\ i = 1, 2, \ldots, N\right\}$ to denote local descriptors (e.g. SIFT) extracted from an image and $U = \left\{\mathbf{u}_j | \mathbf{u}_j \in \mathbb{R}^D,\ j = 1, 2, \ldots, M\right\}$ the visual words learned from k-means clustering.

2.2.1 Bag-of-Words (BoW)

The bag-of-words approach is the state-of-the-art method for the image retrieval that comes from the idea of bag-of-keywords, which is used to retrieve relevant documents. According to the occurrence counts of the keywords in document, each document can have its own histogram representation. Then, two documents can be compared using their histogram representations to judge whether they are similar or not. As for the image, we can view the features from the image as the words in the document, and the visual words as the keywords. That is, the matching algorithm of document search can also be applied to image search.

As shown in the Fig. 1.1, each feature is vector quantized and encoded to a M-dimensional coefficient vector with only one non-zero value, which should be one. In practice, this is done by searching the nearest visual word and this kind of coefficient assignment is so-called hard assignment.

However, hard assignment may suffer from the boundary effect that causes high reconstruction errors, which is called ambiguity problem [13]. To resolve the problem, the intuitive way is to consider not only the nearest visual word but also other visual words in the codebook, this is so-called soft assignment. In this thesis, we roughly divide the existing soft assignment schemes into two frameworks, non-probabilistic (e.g. LLC) and probabilistic (e.g. FV) framework.

2.2.2 Fisher Vector (FV)

The FV representation of a local descriptor is constructed on the basis of a probabilistic framework. It views local descriptors $\{\mathbf{x}_i\}$ extracted from an image as outcomes drawn independently from a parametric probability model $p(\mathbf{x}|\lambda)$ with parameter vector λ. The encoding of an observed local descriptor $\mathbf{x}_i$ is accomplished by taking the derivatives of its log likelihood function $\ln p\left(\mathbf{x}_i|\lambda\right)$ with respect to λ at its initial value, usually obtained through the EM algorithm. To get the final FV representation, a whitening transform is applied to the resulting vector.

As an example, when $p(\mathbf{x}|\lambda)$ is the Gaussian mixture distribution,

$$p(\mathbf{x}|\lambda) = \sum_{j=1}^{M} \pi_j \mathcal{N}\left(\mathbf{x}|\mathbf{u}_j, \mathbf{\Sigma}_j\right), \tag{2.1}$$

$\lambda = \{\pi_j, \mathbf{u}_j, \mathbf{\Sigma}_j,\ j = 1, 2, ..., M\}$, taking derivatives solely with respect to the means $\{\mathbf{u}_j\}$ and applying the transform yields the most popular FV representation for $\mathbf{x}_i$ as the concatenation of all $\mathbf{g}_j$, $j = 1, 2, ..., M$, where

$$\mathbf{g}_j = \frac{1}{\sqrt{\pi_j}} c_{i,j} \mathbf{\Sigma}_j^{-\frac{1}{2}} \left(\mathbf{x}_i - \mathbf{u}_j\right) \tag{2.2}$$

and $c_{i,j} = p\left(j|\mathbf{x}_i\right)$ is the posterior probability.

Interestingly, Equation (2.2) comprises four different terms, each has its own use:

1. the $1/\sqrt{\pi_j}$, which functions similarly to the Inverted Document Frequency (IDF)

for discounting frequent visual words;

2. the posterior probability $c_{i,j}$, which serves as a soft assignment scheme and denotes the weighting coefficient for visual word $\mathbf{u}_j$; and
3. the normalization factor $\mathbf{\Sigma}_j^{-\frac{1}{2}}$ for equalizing the residual vector; and
4. the residual vector $(\mathbf{x}_i - \mu_j)$, which improves discriminative ability.

2.2.3 Vector of Locally Aggregated Descriptors (VLAD)

This encoding scheme can be seen as a simplification of the FV. Just like what is described in BoW, we first learn a codebook of k visual words with k-means clustering. Each local descriptor is associated to its nearest visual word. The idea of the VLAD is to accumulate, for each visual word $\mathbf{u}_j$, the residual vector $\mathbf{x}_i - \mathbf{u}_j$ of the local descriptor $\mathbf{x}_i$ assigned to $\mathbf{u}_j$. The coding complexity can become much simpler by only computing the information with the nearest visual word.

Assuming there are M visual words in the codebook and the local descriptor is an D-dimensional vector, the dimension of the VLAD is $M \times D$. For each local descriptor, we obtain only the residual vector respect to the nearest visual word:

$$g_j = \mathbf{x}_i - \mathbf{u}_j, \tag{2.3}$$

where x_i and u_j respectively denote the i^{th} local descriptor and the j^{th} visual word. Not surprisingly, VLAD inherits the same ambiguity problem from BoW.

2.2.4 Robust Visual Descriptor (RVD)

This scheme was recently discussed in the MPEG CDVS, the motivation of this encoding scheme is the same as VLAD. Each local descriptor $\mathbf{x}_i$ is assigned to the k closest (in ℓ^1 distance sense for complexity reason) visual words with the corresponding ranks and then residual vectors $\mathbf{x}_i - \mathbf{u}_j$ are calculated. Before the aggregation, the residual vectors are first ℓ^1 normalized. It limits the impact of descriptors that are far from visual word (see Fig. 3.3). The representation of each visual word can be formulated as follow:

$$g_j = c_{i,j} \frac{(\mathbf{x}_i - \mu_j)}{\|\mathbf{x}_i - \mu_j\|^1}, \tag{2.4}$$

where $c_{i,j}$ denotes the weighting coefficient of the i^{th} local descriptor respect to the j^{th} visual word. The selection of k depends on number of visual words, stability, reliability and density of local descriptors in an image. In [6] $k = 3$ is used. The weighting coefficients are fixed to 4, 2 and 1.

2.2.5 Locality-constrained Linear Coding (LLC)

The traditional sparse coding (SC) approach works well for image classification, which encourages the global representation to be sparse (sparseness). Yu et al. [16] empirically observed that the nonzero coefficients are often assigned to bases nearby to the encoded local descriptor (locality) and theoretically pointed out that under certain assumptions locality is more essential than sparsity, so they proposed a modification to SC, called local coordinate coding (LCC). However, SC and LCC both require high computational complexity for solving an optimization problem.

Recently, Wang *et al.* [14] proposed an method called LLC, which can be seen as a fast implementation of LCC. Originating from sparse representation [15] and local coordinate coding [16] for image classification, the LLC departs from the probabilistic framework to encode a local descriptor into a high-dimensional vector by linear coding. The encoding acts by computing the least-squares solution $\widetilde{\mathbf{c}}_i$ to approximate an encoding descriptor $\mathbf{x}_i$ using a selected set of visual words from U as basis vectors; that is,

$$\min_{\widetilde{\mathbf{c}}_i} \|\mathbf{x}_i - \mathbf{U}_i\widetilde{\mathbf{c}}_i\|^2 \text{ s.t. } \mathbf{1}^T\widetilde{\mathbf{c}}_i = 1, \tag{2.5}$$

where the columns of $\mathbf{U}_i$ consist of the selected visual words and the unit gain constraint $\mathbf{1}^T\widetilde{\mathbf{c}}_i = 1$ is to meet the shift-invariant requirement. Note that the complete encoded representation of $\mathbf{x}_i$ is a vector $\mathbf{c}_i$ of size $M \times 1$, whose elements denote the weighting coefficients to associate with visual words in U and can be recovered from $\widetilde{\mathbf{c}}_i$ (of size $k \times 1, k << M$) by copying its elements to the entries that correspond to the selected visual words and setting the others to zero. To preserve the sparseness of the resulting code, $\mathbf{U}_i$ contains only few of the visual words from the over-complete dictionary U (in our scheme, $k = 3$); moreover, to meet the locality constraint, they have to be spatially close to the encoding descriptor $\mathbf{x}_i$–i.e., only the first few closest visual words are considered.

From the perspective of FV, LLC is different in that it is without residual vector, residual normalization and IDF terms. Moreover, the coefficient computation method is in least square sense and it explicitly encourages the weighting coefficients to be *local*, which is more probable for similar local descriptors to be represented by similar bases, and is beneficial for computation complexity. Although reporting excellent results in image classification, it has not yet applied in image retrieval.

To summarize, the sparseness property of LLC brings computational advantages since only few weighting coefficients need to be computed, and meanwhile, the locality constraint helps avoid the ambiguity problem.

CHAPTER 3

Global Image Representation Using Locality-constrained Linear Coding

Motivated by the performance benefits of FV and the computational advantages of LLC, we try to combine their merits to form a better global descriptor. Based on the form of FV, we first replace the sophisticated computation of the posterior probability $c_{i,j}$ with LLC. Then, we shall adapt the other terms in FV to such change, so that the performance of FV can be maintained or even improved while the computational complexity is being reduced. To this end, we conduct a series of empirical studies

1. to determine a proper sparsity value for the LLC code,
2. to evaluate the performance difference between probabilistic and non-probabilistic representations,
3. to understand the respective benefits of the residual vector, IDF factor and normalization factor, and whether further simplification to these factors is possible, and finally
4. to see how binarizing the resulting vector to achieve compression may impact the performance.

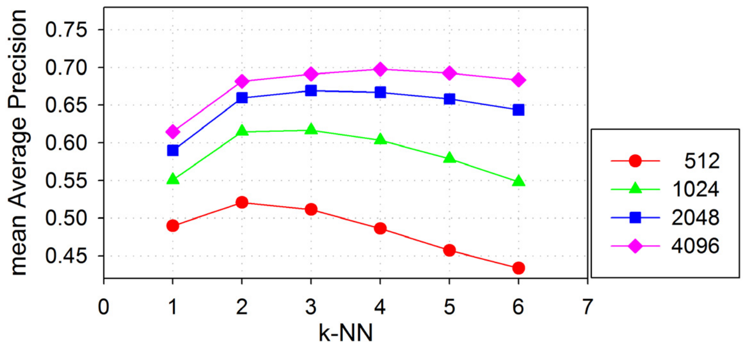


Figure 3.1: Performance (mAP) of 100K experiment on k-NN selection of LLC.

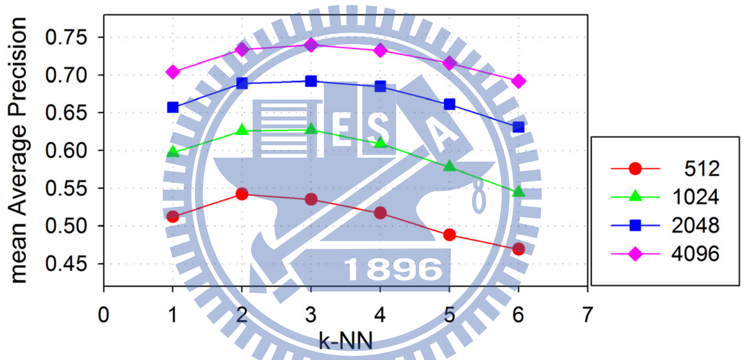


Figure 3.2: Performance (mAP) of 100K experiment on k-NN selection of LLC with residual vector.

All the experiments follow the test conditions specified in Chapter 4, except that distractor images reduced to 100K.

3.1 k-Nearest-Neighbors

Our first experiment aims to choose a proper sparsity value k for LLC, i.e., the number of nearest visual words to be used in Equation (2.5). For the present experiment, a local descriptor is simply encoded into its LLC code.

Fig. 3.1 and Fig. 3.2 show the retrieval performance against the value of k for various codebook sizes. As can be seen, selecting too few or too many visual words is detrimental to performance, regardless of codebook size. Generally, the best performance is achieved when $k = 3$. The result remains even when the other factors are incorporated. We thus choose k to be 3 for all the following tests.

3.2 Coefficient Computation

This study compares the performance between probabilistic and non-probabilistic encoding methods, including FV, LLC, and BoW, for setting $c_{i,j}$. As in the previous study, we represent an encoded local descriptor $\mathbf{x}_i$ using solely the $c_{i,j}, j = 1, 2, ..., M$ factors (i.e., $\mathbf{g}_j = c_{i,j}$). Recall that in FV, $c_{i,j}$ denotes the posterior probability $p(j|\mathbf{x}_i)$ of the j-th visual word given $\mathbf{x}_i$, whereas in both LLC and BoW, it represents its weighting coefficient.

From the results in Sec ♯1 of Table 3.1, we see that with the same codebook size (i.e., the same representation length), the probabilistic method from FV (denoted as PP) outperforms the non-probabilistic ones, BoW and LLC. In addition, the soft assignment scheme (LLC) is superior to the hard assignment one (BoW). Thus, the computational advantages of LLC and BoW come at the cost of inferior performance.

3.3 Residual Vector

This study evaluates the benefit of residual vector $(\mathbf{x}_i - \mathbf{u}_j)$ when it is additionally incorporated as part of the final representation (i.e., $\mathbf{g}_j = c_{i,j}\,(\mathbf{x}_i - \mathbf{u}_j)$). From comparing the results in Sec ♯1 and Sec ♯2 of Table 3.1, we see that including residual vector always gives better performance at the same representation length. Also, its use leads to more significant mAP improvement for BoW and LLC (8% and 5%, respectively) than for PP (0.6%, on average). Interestingly, the configuration (LLC + Residual Vector) performs very close to (PP + Residual Vector), suggesting that a simpler coefficient coding may be used with the inclusion of residual vector. Finally, it should be aware that in order to have the same representation length, the codebook size M for methods in Sec ♯2 is smaller than that for their counterparts in Sec ♯1, due to the inclusion of residual vector, which has a size of D.

Table 3.1: Performance (mAP) of 100K non-binarized experiment on coefficient computation, residual vector, IDF and normalization effect in different encoding methods

				Representation Length ($M \times D$)			
				Non-Binarized			
Sec	**Encoded Information**		**Method**	**512**	**1024**	**2048**	**4096**
♯1	Coeff.		BoW	0.486	0.560	0.601	0.625
			PP	0.552	0.645	0.700	0.735
			LLC	0.511	0.617	0.670	0.691
♯2	Coeff. + Res.		BoW	0.512	0.597	0.657	0.704
			PP	0.556	0.642	0.706	0.755
			LLC	0.535	0.627	0.692	0.740
♯3	Coeff. + Res. + IDF		BoW	0.501	0.588	0.610	0.673
			PP	0.550	0.637	0.702	0.750
			LLC	0.531	0.623	0.698	0.743
♯4	Coeff. + Res. + Norm.	(Variance)	BoW	0.513	0.594	0.656	0.700
			PP	0.557	0.647	0.702	0.752
			LLC	0.539	0.631	0.696	0.738
		(ℓ^1)	BoW	0.536	0.616	0.675	0.716
			PP	0.573	0.658	0.715	0.757
			LLC	0.553	0.643	0.701	0.743
		(ℓ^2)	BoW	0.538	0.618	0.677	0.718
			PP	0.576	0.660	0.715	0.758
			LLC	0.556	0.646	0.703	0.747

Note 1 - M is the number of visual words in the codebook and D is the size of encoded information.
Note 2 - In Sec ♯1, $M = 512, 1024, 2048, 4096$, $D = 1$; In Sec ♯2,♯3 and ♯4, $M = 16, 32, 64, 128$ $D = 32$.
Note 3 - The unit of non-binarized representation is 32 bits.

3.4 Inverted Document Frequency (IDF)

The factor $1/\sqrt{\pi_j}$ in FV representation is believed to exert an effect similar to IDF, which helps discount the frequent visual words under certain assumptions. However, comparing the results in Sec ♯3 of Table 3.1 with those in Sec ♯2, where this IDF term is ignored, shows that the extra gain it brings seems rather minor. We shall thus exclude it from our approach.

3.5 Residual Vector Normalization

So far, we have seen that including residual vector $(\mathbf{x}_i - \mathbf{u}_j)$ can bring performance advantages. In FV representation, this residual vector is further normalized by $\mathbf{\Sigma}_j^{-\frac{1}{2}}$, the covariance matrix of the j-th component, so that the variability along each of its dimensions can be equalized in a statistical sense. Here we wish to understand whether there can be significant benefit from performing such normalization (i.e., $\mathbf{g}_j = c_{i,j}\mathbf{\Sigma}_j^{-\frac{1}{2}}(\mathbf{x}_i - \mathbf{u}_j)$). Additionally, we compare its performance with two ad-hoc normalization schemes based on the ℓ^1 and ℓ^2 norm of the residual vector (i.e.,

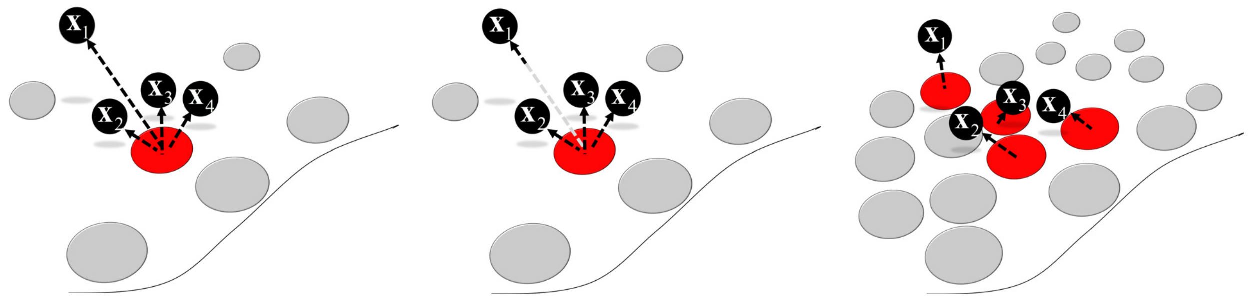


Figure 3.3: Illustration of the effect of normalization on residual vector. Local descriptors (Gray) denotes $\mathbf{x}_i$ and visual words (Transparent) denotes $\mathbf{u}_j$ and the visual word selected as local basis is in red. (a) Without normalization, final representation would be dominated by the residual vector of $\mathbf{x}_1$. (b) With normalization, the effect disappears as we now consider only the direction of the residual vector, in this case in the sense of ℓ^2. (c) Without normalization, the effect would also decrease when larger codebook size is used because of high density visual word used.

$\mathbf{g}_j = c_{i,j}\,(\mathbf{x}_i - \mathbf{u}_j)\,/\,\|\mathbf{x}_i - \mathbf{u}_j\|^p$, $p = 1$ or 2). This is motivated by the intention to de-emphasize, during the pooling process, the contribution of (outlier) local descriptors that are spatially far away from visual words and that therefore have a large residual vector. Fig. 3.3 (a) and (b) show an illustration of such effect.

From Table 3.1, the methods in Sec ♯2 perform almost identically to their counterparts with covariance normalization in Sec ♯4, which suggests that such sophisticated normalization may not be as effective as expected. But, somewhat surprisingly, using simpler ℓ^1 or ℓ^2 normalization gives 2% gains on average for smaller codebook sizes, although the benefits diminish when the codebook size is increased. As illustrated in Fig. 3.3 (c), with larger codebook size used, influence on this effect is decreasing due to the increasing density of visual words used.

3.6 Global Descriptor Binarization

Sign binarization is a common technique for further compressing the pooling result from local descriptors' representations. It quantizes a non-negative real value to "1" and negative one to "0". As an example, binarizing a real-valued pooling result with residual vector size $D = 32$ and codebook size $M = 128$ leads to great reduction in global

Table 3.2: Performance (mAP) of 100K binarized experiment on coefficient computation, residual vector, IDF and normalization effect in different encoding methods

				Representation Length ($M \times D$)			
				Binarized			
Sec	**Encoded Information**		**Method**	**512**	**1024**	**2048**	**4096**
$\sharp 1$	Coeff.		BoW	$\times$	$\times$	$\times$	$\times$
			PP	$\times$	$\times$	$\times$	$\times$
			LLC	$\times$	$\times$	$\times$	$\times$
$\sharp 2$	Coeff. + Res.		BoW	0.387	0.510	0.622	0.710
			PP	0.444	0.567	0.660	0.719
			LLC	0.433	0.571	0.669	0.728
$\sharp 3$	Coeff. + Res. + IDF		BoW	0.387	0.510	0.622	0.710
			PP	0.444	0.567	0.660	0.719
			LLC	0.433	0.571	0.669	0.728
$\sharp 4$	Coeff. + Res. + Norm.	(Variance)	BoW	0.387	0.510	0.622	0.709
			PP	0.445	0.566	0.660	0.719
			LLC	0.433	0.570	0.670	0.734
		(ℓ^1)	BoW	0.407	0.534	0.641	0.718
			PP	0.460	0.581	0.668	0.721
			LLC	0.456	0.593	0.683	0.735
		(ℓ^2)	BoW	0.407	0.538	0.641	0.719
			PP	0.463	0.583	0.666	0.721
			LLC	0.458	0.593	0.680	0.736

Note 1 - M is the number of visual words in the codebook and D is the size of encoded information.
Note 2 - In Sec $\sharp 1$, $M = 512, 1024, 2048, 4096$, $D = 1$; In Sec $\sharp 2$,$\sharp 3$ and $\sharp 4$, $M = 16, 32, 64, 128$ $D = 32$.
Note 3 - The unit of binarized representation is 1 bit.

descriptor's size from 16K bytes to 512 bytes. The computation of the similarity score is the same as that in CDVS test model [5]. For the matching of the global descriptors $\mathbf{g}^X$ and $\mathbf{g}^Y$, given image X and Y, the similarity score $S_{X,Y}$ can be calculated quickly by using bitwise comparison to compute Hamming distances, as follows

$$S_{X,Y} = \frac{\sum_{j=1}^{M} b_j^X b_j^Y \left(32 - 2\mathrm{Ha}\left(\mathrm{sign}(\mathbf{g}_j^X), \mathrm{sign}(\mathbf{g}_j^Y)\right)\right)}{\sqrt{128 \sum_{j=1}^{M} b_j^X} \times \sqrt{128 \sum_{j=1}^{M} b_j^X}}, \tag{3.1}$$

where $\mathrm{sign}(\mathbf{g}_j^X)$ denotes the binarized vector of $\mathbf{g}_j^X$ and $\mathrm{Ha}(.,.)$ denotes the Hamming distance between two binarized vectors. We have $b_j^X = 1$ if there is at least one local descriptor assigned to the j^{th} visual word; otherwise, $b_j^X = 0$. The similarity score of j^{th} visual word between two binarized vectors will be taken into account, when their own b_j^X and b_j^Y are equal to one. Intuitively, binarization comes at the expense of poorer performance. This last study is aimed at investigating how sensitive the performance of different representations is to sign binarization.

In Table 3.2 presents the performance when the real-valued pooling representations shown in Table 3.1 are sign binarized. As expected, binarization results in performance decline in almost all cases and to a large degree, especially when a small codebook size is in use. Its effect, however, is less obvious for larger codebook sizes. Remarkably, representations with probabilistic-based $c_{i,j}$ computation are relatively more sensitive to binarization, in which case, they perform worse than LLC-based schemes. This may be attributed to that the binarized signals for components with small $c_{i,j}$ values can be very noisy, which cause the $b_j^X = 1, \forall j = 1, ..., M$. The sparse constraint of LLC helps avoid this problem. A side experiment confirms our finding by showing that the former can benefit from quantizing to zero the small $c_{i,j}$ values in FV representation, which implies some visual words will not be considered, $b_j^X = 0$, as in the Equation (3.1).

To conclude, we shall adopt a binarized global descriptor with

$$\mathbf{g}_j = c_{i,j} \frac{(\mathbf{x}_i - \mathbf{u}_j)}{\|\mathbf{x}_i - \mathbf{u}_j\|^2}, j = 1, 2, ..., M, \tag{3.2}$$

where $c_{i,j}$ is obtained by LLC.

CHAPTER 4

Experimental Results

This chapter compares the retrieval performance of the proposed scheme with state-of-the-art global representations, including VLAD, FV, and RVD. We also compared their average encoding time (where the time for extracting local descriptors is excluded) for forming a global descriptor, so as to give a rough indication of their relative computational complexity. Experiments are conducted based on the datasets from the MPEG CDVS [4], including (1) mixed text/graphics, (2) paintings, (3) frames from video clips, (4) landmarks/buildings, (5) objects/scenes, and 1M distractor images from Flickr.

In our experiments, a subset of detected SIFT features are selected to generate the global descriptor (we select up to 300 SIFT features per image), and then SIFT features are power normalized and ℓ^2 normalized. PCA is employed to reduce the dimensionality of SIFT features from 128-dim to 32-dim. We perform PCA transform matrix training over 2 million randomly selected SIFT features, extracted from the independent datasets, including INRIA Holiday [1], Oxford Building [2] and Pasadena Building [3]. Other implementation details include:

1. visual words are obtained from k-means clustering for all tested schemes;
2. model parameters for FV are estimated with sample variance and sample mixture weight according to the visual words from k-means;

Table 4.1: Performance (mAP) of 1M non-binarized experiment on different encoding methods with $M = 128$ and $D = 32$

	Dataset						Encoding
Method	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**Avg.**	**Time**
VLAD [7]	0.665	0.679	0.724	0.543	0.585	0.639	2.5 ms
FV [10]	0.723	0.715	0.767	0.579	0.666	0.696	12.6 ms
RVD [6]	0.691	0.722	0.744	0.564	0.633	0.672	3.0 ms
Ours	0.721	0.746	0.766	0.580	0.660	0.698	3.2 ms
Simplified FV	0.730	0.764	0.780	0.575	0.657	0.706	6.7 ms

Note - Encoding time: encoding plus pooling time.

Table 4.2: Performance (mAP) of 1M binarized experiment on different encoding methods with $M = 128$ and $D = 32$

	Dataset						Encoding
Method	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**Avg.**	**Time**
VLAD [7]	0.666	0.752	0.782	0.513	0.566	0.654	2.5 ms
FV [10]	0.680	0.776	0.795	0.509	0.589	0.668	12.6 ms
RVD [6]	0.700	0.790	0.795	0.536	0.633	0.686	3.0 ms
Ours	0.704	0.805	0.809	0.537	0.632	0.693	3.2 ms
Simplified FV	0.704	0.786	0.806	0.519	0.603	0.683	6.7 ms

Note - Encoding time: encoding plus pooling time.

3. power-normalization [8] is applied to the pooling result; and
4. Cosine Similarity is used to ranking the images.

As usual, the performance is measured by mean Average Precision (mAP) according to the top 500 images. All the experiments were conducted on an Intel i7 server with 32G memory and 1TB HDD.

4.1 Large-Scale Experiment

From Table 4.1, with real-valued, non-binarized global representation, our scheme performs comparably to FV, and outperforms VLAD and RVD by 5.9% and 2.6%, respectively. Of interest is that both VLAD and RVD somehow benefit slightly from sign binarization while the others suffer from it, see Table 4.2. Nevertheless, the proposed scheme still performs best even with binarization. As to complexity, the FV has the largest average encoding time. RVD is slightly faster than ours due to the use of power-of-two weighting factors (i.e., $c_{i,j}$).

In addition, motivated by our studies in Chapter 3, we made a few simplifications to FV in hopes of getting a better trade-off between complexity and performance. The

Algorithm 3.1 Codebook Learning

```
input:$\mathbf{B}_{init} \in \mathbb{R}^{D \times M}$,$\mathbf{X} \in \mathbb{R}^{D \times N}$,K
output:B
 1:$\mathbf{B} \leftarrow \mathbf{B}_{init}$
 2:for i =1 to N do
 3:   $\mathbf{d} \leftarrow 1 \times M$ zero vector
      {find K nearest neighbors}
 4:   for $j = 1 to M$ do
 5:      $d_j \leftarrow \|\mathbf{x}_i - \mathbf{b}_j\|^2$
 6:   end for
 7:   $id \leftarrow \{j | \mathbf{b}_j \in k^{th}$ NN of $\mathbf{x}_i, \forall k = 1...K\}$
 8:   $\mathbf{B}_i \leftarrow \mathbf{B}(:, id)$
      {coding}
 9:   $\widetilde{\mathbf{c}}_i \leftarrow \arg\min_c \|\mathbf{x}_i - \mathbf{B}_i \widetilde{\mathbf{c}}_i\|^2, s.t. \mathbf{1}^T \widetilde{\mathbf{c}}_i = 1$
      {update basis}
10:   $\triangle \mathbf{B}_i \leftarrow -2\widetilde{\mathbf{c}}_i^T (\mathbf{x}_i - \mathbf{B}_i \widetilde{\mathbf{c}}_i), \mu = \sqrt{1/i}$
11:   $\mathbf{B}_i \leftarrow \mathbf{B}_i - \mu \triangle \mathbf{B}_i / |\widetilde{\mathbf{c}}_i|_2$
12:   $\mathbf{B}(:, id) \leftarrow \mathbf{B}_i$
13:end for
```

changes we made involve:

1. imposing locality-constraint with $k = 10$, that is, $c_{i,j}$ is computed and properly normalized with respect to the first 10 nearest visual words only;
2. using ℓ^2 normalization; and
3. removing IDF.

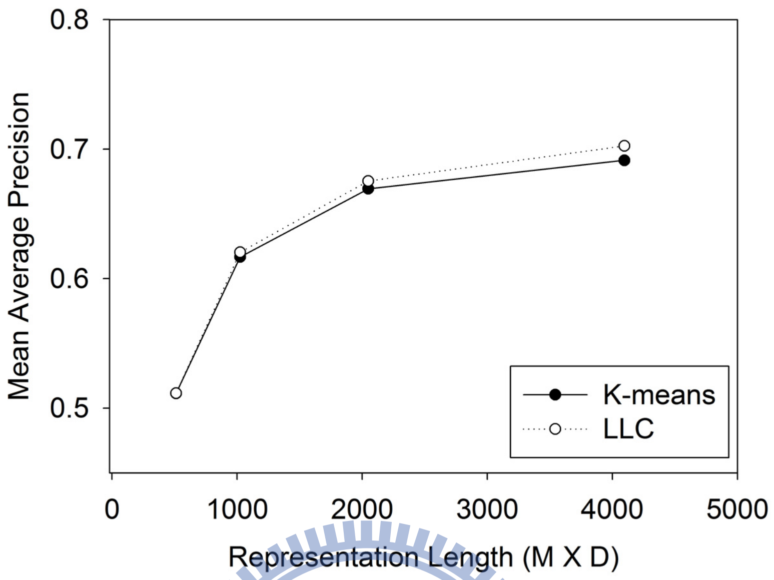
As can be seen, these changes not only reduce the encoding time of FV by half, but also improve its performance slightly (1-1.5%).

4.2 Codebook Learning

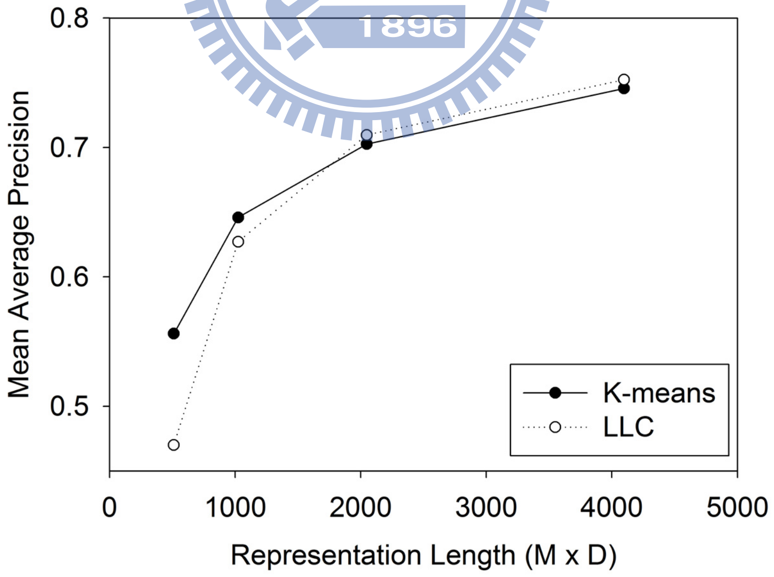
To provide more comprehensive analysis of the proposed LLC-based encoding method, we further evaluated its performance with respect to codebook learning. A simple way to generate the codebook is to use clustering based method such as k-means algorithm. To further improve the performance, we use the LLC coding criteria to learn the codebook, which is similar to the algorithm proposed by [14].

To elaborate, we first use a codebook trained by k-means clustering to initialize $\mathbf{B}$. Then we loop through all the training descriptors to update $\mathbf{B}$ incrementally. In each iteration, we take in a single local descriptor to select the local bases $\mathbf{B}_i$ and solve Equation (2.5) to obtain the corresponding coefficients. The obtained coefficients are then used to update the codebook in a gradient descent fashion. The learning procedure is illustrated in Alg. 3.1.

We compared the retrieval performance using codebooks trained by k-means algorithm and by our proposed Alg. 3.1. In Fig. 4.1, both coding schemes can obtain 0.6-1.2% improvement over the codebook by k-means for larger codebook size. From the experiments, the LLC optimization can improve the performance when the codebook size is large enough. Though little gain is obtained, it indicates further study on it.



(a) LLC (D = 1)



(b) Our Approach (D = 32)

Figure 4.1: Performance (mAP) of 100K experiment on codebook learning.

CHAPTER 5

Conclusions

With a thorough analysis of the interaction among locality property and other terms in FV, we conduct a series of studies and the final result is telling that you can preserve high performance for just using a simple global representation.

In our work, we demonstrate that the sparseness and locality from LLC bring several advantages, where locality property ensures that similar local descriptors always have similar encoding results, while the sparseness of the resulting code allows only few weighting coefficients to be computed. According to the properties, this thesis proposed two global image representations.

1. We propose a global image representation using LLC-based coding with normalized residual vector. We use least-square solution to obtain the coefficients and apply ℓ^2 normalization on the residual vector. LLC provides an analytical solution which makes the coefficient computation much simpler and faster.
2. We make a few simplifications to FV, which reduce a lot of floating point operations during the encoding process. By applying locality property, the encoded information only have a few non-zero values. It makes the pooling step much faster.

Under the test conditions [5] suggested by the MPEG committee, the proposed schemes not only benefit in mAP performance, but also offer complexity advantages when compared with other similar works. From our analysis, we can see that sparseness and locality can bring the advantages for image retrieval.

Bibliography

[1] "Holidays dataset," *http://lear.inrialpes.fr/people/jegou/data.php.*

[2] "Oxford building dataset," *http://www.robots.ox.ac.uk/ vgg/data/oxbuildings/.*

[3] "Pasadena building dataset," *http://www.vision.caltech.edu/archive.html.*

[4] "Evaluation Framework for Compact Descriptors for Visual Search," *ISO/IEC JTC1/SC29/WG11 MPEG 98th meeting, N12202*, July 2011.

[5] M. Bober, G. Cordara, S. Paschalakis, K. Iwamoto, G. Francini, and V. Chandrasekhar, "Test Model 4: Compact Descriptors for Visual Search," *ISO/IEC JTC1/SC29/WG11 MPEG 102th meeting, W13145*, October 2012.

[6] M. Bober, S. Husain, S. Paschalakis, and K. Wnukowicz, "Improvements to tm6 with a robust visual descriptor – proposal from university of surrey and visual atoms," *ISO/IEC JTC1/SC29/WG11 MPEG 105th meeting, M30311*, July 2013.

[7] H. Jegou, M. Douze, C. Schmid, and P. Perez., "Aggregating local descriptors into a compact image representation," *In Proc. CVPR*, 2010.

[8] H. Jgou, M. Douze, and C. Schmid, "On the burstiness of visual elements," *In Proc. CVPR*, 2009.

[9] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, 2004.

[10] F. Perronnin, Y. Liu, J. Sanchez, and H. Poirier, “Large-scale image retrieval with compressed Fisher vectors,” *In Proc. CVPR*, 2010.

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

[12] J. Sivic and A. Zisserman, “Video Google: A Text Retrieval Approach to Object Matching in Videos,” *In Proc. ICCV*, 2003.

[13] J. van Gemert, C. Veenman, A. Smeulders, and J.-M. Geusebroek, “Visual Word Ambiguity,” *In IEEE Trans. Pattern Analysis and Machine Intelligence (TPAMI)*, 2010.

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

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

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