Hindawi Publishing Corporation
The Scientific World Journal
Volume 2014, Article ID 632871, 14 pages
http://dx.doi.org/10.1155/2014/632871

Research Article

Context-Aware and Locality-Constrained Coding for Image Categorization

Wenhua Xiao, Bin Wang, Yu Liu, Weidong Bao, and Maojun Zhang

College of Information System and Management, National University of Defense Technology, Changsha, Hunan 410073, China

Correspondence should be addressed to Wenhua Xiao; wenhuaxiao@nudt.edu.cn

Received 22 September 2013; Accepted 22 January 2014; Published 18 March 2014

Academic Editors: J. T. Fernandez-Breis and Z. Yu

Improving the coding strategy for BOF (Bag-of-Features) based feature design has drawn increasing attention in recent image categorization works. However, the ambiguity in coding procedure still impedes its further development. In this paper, we introduce a context-aware and locality-constrained Coding (CALC) approach with context information for describing objects in a discriminative way. It is generally achieved by learning a word-to-word cooccurrence prior to imposing context information over locality-constrained coding. Firstly, the local context of each category is evaluated by learning a word-to-word cooccurrence matrix representing the spatial distribution of local features in neighbor region. Then, the learned cooccurrence matrix is used for measuring the context distance between local features and code words. Finally, a coding strategy simultaneously considers locality in feature space and context space, while introducing the weight of feature is proposed. This novel coding strategy not only semantically preserves the information in coding, but also has the ability to alleviate the noise distortion of each class. Extensive experiments on several available datasets (Scene-15, Caltech101, and Caltech256) are conducted to validate the superiority of our algorithm by comparing it with baselines and recent published methods. Experimental results show that our method significantly improves the performance of baselines and achieves comparable and even better performance with the state of the arts.

1. Introduction

Automatic image categorization has drawn increasing attention of the researchers around the world due to its widespread prospects in various applications (e.g., video surveillance [1], image and video retrieval [2], web content analysis [3], and biometrics [4]). In recent works addressing the image categorization tasks, the BOF based model [5], developed from the BOW (Bag-of-Words) in document analysis [6], is one of the most popular and efficient models in dealing with this problem. BOF based method is often comprised of the following common steps: feature extraction, codebook (or dictionary) designing, feature encoding, and pooling. Given a dataset, firstly, local features are often depicted by descriptors such as SIFT [7]. Secondly, a codebook to span the feature space is often designed by K-means [8], sparse coding [9], K-SVD [10], and others. Thirdly, given feature descriptors and codebook as the input, the output of this step is a coding matrix. In this step, each feature descriptor activates a number of code words and generates a coding vector after features are coded over this codebook. Fourthly, pooling methods (e.g., average pooling [8] and max pooling [11]) are often used to obtain the compact signature of the image. Of all the above four steps, feature coding is the core component, which links feature extraction and feature pooling, and greatly influences image classification in terms of both accuracy and speed [12]. Owning to this key role of coding phase in the pipeline of BOF based method, since the seminal work of [8], improving the coding strategy has drawn increasing attention in recent works.

Coding can be regarded as a procedure assigning few code words with weighted coefficient to represent local features while satisfying some desirable properties. Various coding styles have been proposed in previous literatures [8, 10, 11, 13–17]. And some limitations (e.g., quantization error, nonconsistency, and computational cost) of traditional models have been partially alleviated by those previous works. However, there still *exists an important limitation of BOF* that cannot be solved by previous works. *This limitation is produced* by the features from different classes to depict different objects while

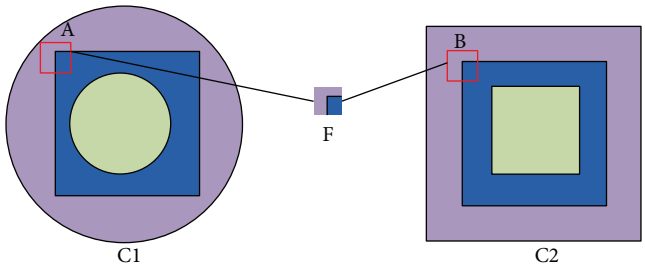


Figure 1: Two ambiguous features A and B are extracted from two different image classes and described with a similar descriptor F.

with similar descriptor. Intuitively, those features should be encoded discriminatively as to preserve their different semantic meaning. However, due to their similar descriptors, they cannot be distinguished with their codes generated by previous coding strategies. This is named *ambiguity problem*. For example, as shown in Figure 1, there are two ambiguous features A and B. A and B, with different semantic meaning, indicate corner patch in two images, respectively. Obviously, A and B should be encoded with different bases to distinguish the two images better. However, due to the reason that they have similar descriptors, they are given similar codes and cannot be distinguished clearly by previous coding strategies such as VQ, SVQ, SC, and LLC.

In this paper, we attempt to further improve the BOF with regard to the above-presented ambiguity problem. The *motivations* of our method are as follows. (1) Scene or object in realistic image has a certain cooccurrence pattern, which determines the difference of each class images, around its neighborhood. So we can use the cooccurrence pattern (context) information to distinguish the ambiguous features to solve the ambiguity problem. (2) Inspired by LLC which enforces locality in feature space and achieves excellent performance, we incorporate the locality in context space into LLC so as to inherit its advantages (analytical solution and real-time coding speed).

In detail, observing the realistic images, it can be easily obtained that each class of scene or object has a certain cooccurrence pattern in its neighborhood. For example, a pan and a stove often appear in their neighborhood in kitchen, and a butt often appears near to a barrel in AK47. This appearance cooccurrence can be considered as the context preserving discriminative information of each class. Even though those descriptors from different classes are similar, their context often appears different because their surroundings often show different appearance cooccurrence pattern. Based on above assumption, if we consider the context information when encoding the feature, the coding result of those similar features with different semantic meaning will be discriminative. Therefore, in this paper, we propose to use this context information to tackle the ambiguity problem. Obviously, how to describe such context information and incorporate it into coding procedure become the main tasks of our approach. For this purpose, firstly, to capture the contextual information, a word-to-word co-occurrence relationship matrix for each class is constructed within local domain of each image. Because the statistical relationship matrix reflects the spatial distribution and the cooccurrence of features in neighbor region of each class, it has the ability to describe partial contextual information. Secondly, the relationship matrix is used to select the optimal bases in context space. Thirdly, combined with the locality factor in feature space, this context factor enhances the LLC [15] model to a novel model called context-aware and locality-constrained coding (CALC). Indeed, CALC can be considered a fineness version of LLC because it locally constrains the coding in both feature space and context space. Here, the expression "*context*" means the surrounding appearance cooccurrence pattern of a local feature. Extensive experiment demonstrated the effectiveness of the proposed method.

The rest of the paper is organized as follows. Section 2 reviews the related feature coding methods. CALC is proposed in Section 3. The details of implementation of CALC are introduced in Section 4. Properties analysis of CALC is presented in Section 5. Then, the experimental results and analysis are shown in Section 6. Finally, conclusion is drawn in Section 7.

2. Related Works

BOF based models are widespread adopted in computer vision and pattern recognition fields. In this section, we concentrate on those related works in the view of image categorization here. Let $\mathbf{X} = \{x_i \in R^D, i \in 1, \ldots, N\}$ be N local descriptors with D-dimension extracted from an image. Given a codebook with M bases $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_M] \in \mathbf{R}^{D\times M}$, $\mathbf{x}_i$ is converted into M-dimensional code denoted as $c_i \in \mathbf{R}^M$ by feature coding methods. Several popular coding methods are as follows.

Vector Quantization (VQ) [8, 19]. In the original BOF model, its coding strategy assigns just a single base to the feature, which is known as VQ (Vector Quantization), or HC (Hard Coding). Each local descriptor is assigned to the nearest visual word:

$$c_{ij} = \begin{cases} 1, & \text{if } j = \underset{j}{\operatorname{argmin}} \left\| \mathbf{x}_i - \mathbf{b}_j \right\|_2^2, \\ 0, & \text{else.} \end{cases} \tag{1}$$

This coding is simple but, as reported in [11], suffers from the reconstruction error due to the reason that it only assigns a single code word to the descriptor.

Soft Vector Quantization (SVQ). To ameliorate the quantization loss of VQ, Gemert et al. [13] proposed SVQ on which a feature is coded across many codebook elements instead of using one:

$$\mathbf{c}_{ij} = \frac{\exp\left(-\beta \left\| \mathbf{x}_i - \mathbf{b}_j \right\|_2^2\right)}{\sum_{l=1}^{M} \exp\left(-\beta \left\| \mathbf{x}_i - \mathbf{b}_l \right\|_2^2\right)}, \tag{2}$$

where β is a parameter controlling how widely the assignment distributes the weight across all the code words. A small β gives a broad distribution, while a large β gives a peaked

distribution, more closely approximating hard assignment. This is further improved by Liu et al. [14], who use localized soft assignment (LSVQ). Their difference is that SVQ encodes the descriptors across all the codebook elements while LSVQ confines the soft assignment to a local neighborhood around the descriptor being coded.

Sparse Coding (SC) [11]. Another way to alleviate the quantization loss of VQ is SC which encodes a descriptor by using the coefficients of a linear combination of the code words in $\mathbf{B}$, with a sparsity-promoting l_1 norm:

$$c_i = \underset{c}{\operatorname{argmin}} \left\|\mathbf{x}_i - \mathbf{B}\mathbf{c}\right\|_2^2 + \lambda \|\mathbf{c}\|_1, \quad \lambda \in R, \tag{3}$$

where the first term represents the reconstruction error of $\mathbf{x}_i$ with respect to codebook $\mathbf{B}$. The second term is a sparse constraint regularization on code $\mathbf{c}$, and λ is a regularization factor to balance these terms. Although SC significantly improved its robustness to the problems produced by VQ, its expensive computational demanding and nonconsistent encoding of similar descriptors are the limitations [15].

Locality-Constrained Linear Coding (LLC) [15]. To alleviate the limitations of SC, LLC enforces locality instead of sparsity. LLC uses the following criteria:

$$\begin{aligned} &\mathbf{c}_i = \underset{\mathbf{c}}{\operatorname{argmin}} \left\|\mathbf{x}_i - \mathbf{B}\mathbf{c}\right\|_2^2 + \lambda \|\mathbf{d} \odot \mathbf{c}\|_2^2 \\ &\text{s.t.} \quad \mathbf{1}^T\mathbf{c} = \mathbf{1}, \end{aligned} \tag{4}$$

where the first term is reconstruction error. The second term is the locality constraint regularization on code $\mathbf{c}$, and λ is a regularization factor. In the second term, $\odot$ denotes the element-wise multiplication, and $\mathbf{d} \in R^M$ is the locality adaptor that gives different weight for each base vector proportional to its similarity to the input feature $\mathbf{x}$. Specifically, $\mathbf{d} = \exp(\operatorname{dist}(\mathbf{x}_i, \mathbf{B})/\sigma)$, where $\operatorname{dist}(\mathbf{x}_i, \mathbf{B}) = [\operatorname{dist}(\mathbf{x}_i, \mathbf{b}_1), \ldots, \operatorname{dist}(\mathbf{x}_i, \mathbf{b}_M)]^T$, and $\mathbf{dist}(\mathbf{x}_i, \mathbf{b}_j)$ is the Euclidean distance between $\mathbf{x}_i$ and the jth base $\mathbf{b}_j$. σ is used for adjusting the weight decay speed for the locality adaptor. This coding style is based on the hypothesis that descriptors approximately reside on a lower dimensional manifold in an ambient descriptor space; thus, it alleviates the quantization error while preserving the consistent encoding ability.

Laplacian Sparse Coding (LSC) [16]. Another alternative approach to improve the consistency of SC is LSC coding strategy, which adds a Laplacian matrix to the SC object function and codes all the descriptors simultaneously:

$$\begin{aligned} &\underset{\mathbf{B},\mathbf{C}}{\operatorname{argmin}} \|\mathbf{X} - \mathbf{B}\mathbf{C}\|_2^2 + \lambda \sum_i \left\|c_i\right\|_1 + \beta \operatorname{tr}\left(\mathbf{C}\mathbf{L}\mathbf{C}^T\right) \\ &\text{s.t.} \quad \left\|b_j\right\|^2 \le 1, \quad \forall j, \end{aligned} \tag{5}$$

where $\mathbf{L} = \mathbf{A} - \mathbf{W}$ is the Laplacian matrix obtained from the similarity matrix $\mathbf{W}$ encoding the relationship between local features and $\mathbf{A}_{m,m} = \sum_n \mathbf{W}_{m,n}$. By incorporating the similarity preserving term into the objective of sparse coding, Laplacian sparse coding can alleviate the instability of sparse codes. However, since the Laplacian matrix often has an extremely high dimension, LSC is computationally infeasible.

Table 1: Comparison of previous coding schemes.

Coding scheme	Quantization error	Nonconsistent coding	Computational cost	Ambiguity
VQ [19]	High	Low	Low	High
SVQ [13]	Low	Low	Middle	High
SC [11]	Low	High	High	High
LSC [16]	Low	Low	High	High
LLC [15]	Low	Low	Low	High
LCSR [17]	Low	Low	High	High

Locality-Constrained and Spatially Regularized Coding (LCSR) [17]. A novel coding strategy called LCSR is proposed most recently; unlike the previous works, this approach introduces the spatial information into the coding process and its object function leads to the following optimal assignment configuration:

$$\operatorname{argmin} \sum_p \sum_{i=1}^m \left\|\mathbf{x}_p - \widehat{b}_{p,i}\right\|_2^2 + \beta \sum_{p\sim q} w_{p,q} \sum_{i=1}^m \left\|\widehat{b}_{p,i} - \widehat{b}_{q,i}\right\|_2^2, \tag{6}$$

where $\widehat{\mathbf{B}}_p = \{\mathbf{b}_{p,i}; i = 1, \ldots, m\}$ denotes the set of code words in $\mathbf{B}$ assigned to the local feature $\mathbf{x}_p$, $p \sim q$ indicates the indexes of the spatially neighboring patches under a fixed neighboring system, and $w_{p,q}$ is a local regularization parameter that corresponds to the similarity between local patches $\mathbf{x}_p$ and $\mathbf{x}_q$; the more similar the local patches are, the higher the basis selection operation is regularized. β controls the global regularization. Indeed, this assignment style aims at assigning features to bases of cardinality m within the set of the k-nearest visual words in the codebook while preserving the consistency of the coding regarding the context of the image. Once each local feature is assigned to the optimal bases by solving (6), its response over the selected bases can be obtained using several recent coding strategies (e.g., VQ, SVQ, and LLC). Since it enforces the locality in both the feature space and the spatial domain of the image, as reported in [17], LCSR improves the performance of most of the previous coding schemes when it is integrated into them. However, the object function in (6) is nonconvex and the α-expansion based optimization algorithm is adopted, which lead to computational iteration to get the convergence to a local optimum.

All the aforementioned coding schemes overcome some of the limitations of BOF mentioned in Section 1, and from which we can illustrate the comparison between those coding styles in various aspects. As can be seen in Table 1, none of the coding styles has considered the ambiguous coding problem. In the next section, we propose an efficient and effective method to solve this problem.

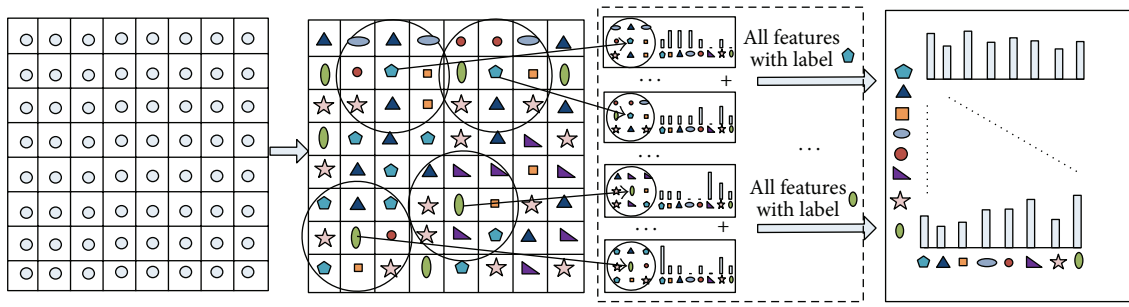


Figure 2: The flowchart of the construction procedure of word-to-word cooccurrence matrix. (a) Extracting features using sift descriptor. (b) Assigning labels to the features using k-means. (c) Constructing word-to-word cooccurrence of each visual word. (d) Concatenating the cooccurrences of all visual words to form the final matrix.

3. Proposed Method

The main components of our method consist of two steps: constructing the word-to-word cooccurrence matrix to describe the local spatial context information and incorporating this context information into coding step. The details of these two aspects are presented as follows.

3.1. Construction of Word-to-Word Cooccurrence Matrix. As mentioned in Section 1, in a specific scene or object, images often share a common or similar cooccurrence pattern in local region. On the level of descriptor, we believe that such pattern was reflected by the cooccurrence relations among the local descriptors in local region of images. In this section, we present a novel and simple way to describe this relationship, the flowchart of the procedure for one image is illustrated in Figure 2, and the final matrix of the specific class is obtained by accumulating the result of each image belonging to this class. The details of the procedure (for one image) are presented as follows.

With the training data from all classes (e.g., randomly selected 100,000 descriptors from whole datasets), a codebook with size K is firstly built by codebook training method (e.g., k-means or SC). For a specific image class, let the local descriptors $\mathbf{X} = \{x_i \in R^D, i \in 1, \ldots, N_t\}$ from this class be training data; then, the local descriptors are labeled using k-NN. We denote $f_i = \{x_i, l_i, p_i\}$ as the ith local feature, where x_i is the descriptor, l_i (belonging to $1, \ldots, K$) is the corresponding index of code words, and $p_i = \{x_i, y_i\}$ records the pixel location at which feature f_i is centered. Thus, all of features can be clustered into K sets denoted as $S = \{S_1, \ldots, S_K\}$, where $S_i = \{f_1^i, \ldots, f_{N_i}^i\}$ contains the features with the label of i and N_i is the number of features in S_i.

To capture the relations among local features, we define the context domain of feature f_i as follows:

$$C_i = \left\{ f_j \mid \left\{x_j, y_j\right\} \in \Omega\left(p_i\right) \right\}, \tag{7}$$

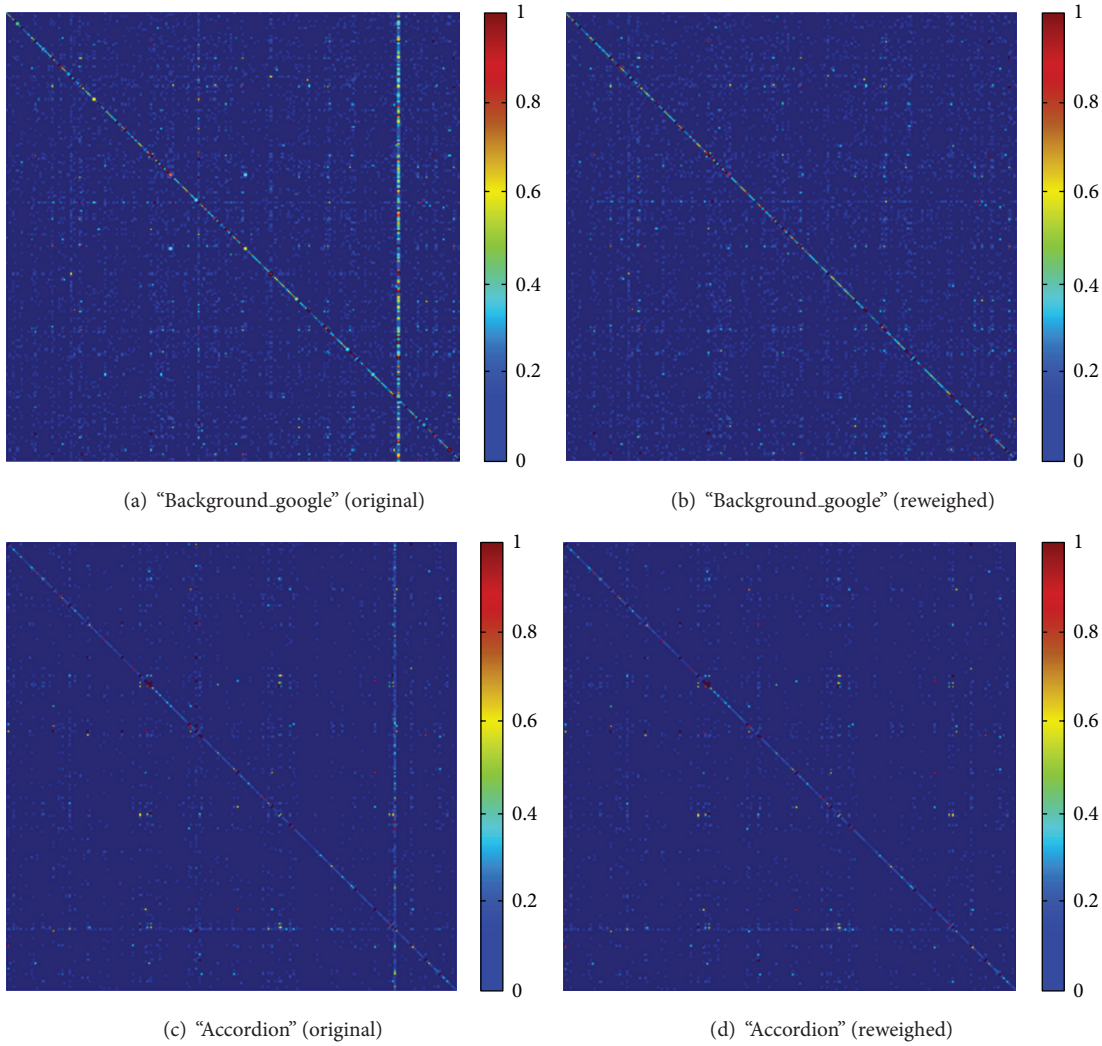
where $\Omega(p_i)$ denotes the local domain of feature f_i, which is represented by a circle with the center of p_i and the radium of r (as shown in Figure 2). Thus, the context domain of f_i contains all features within the boundary of the local area $\Omega(p_i)$. Then, for the jth feature S_{ij} in S_i, a k-dimensional vector $h_{ij} = [v_1^{ij}, \ldots, v_K^{ij}]$ is obtained within context domain C_i, where v_l^{ij} $(l = 1, \ldots, K)$ is the number of features with the label of l within the context domain of S_{ij}. After accumulating the vectors of all features in S_i, a neighbor distribution histogram h_i of the ith code word is obtained:

$$h_i = \sum_{j=1}^{N_i} h_{ij}, \tag{8}$$

where $h_i = [v_1^i, \ldots, v_K^i]$ and v_k^i describe the cooccurrence intensity between the ith code word and the kth code word. If we denote the code words as vertexes and their cooccurrence intensity as their connection weight, this relationship can be shown as a relationship graph. In this paper, we regularized the value of the connection weight to $[0, 1]$. Once we repeat the above procedure over all S_i, $i = 1, \ldots, K$, a relationship matrix of all the code words is constructed. We denote it as $H = [h_1; h_2 \ldots; h_K]$. For distinguishing the relation matrix constructed on test images, we call this matrix generated by trained data as template matrix. As can be seen in Figures 3(a) and 3(c), two distinguished relationship matrixes constructed from category "Background_google" and "accordion" in Caltech101 are illustrated as depth map, which shows the difference of local context between "Background_google" and "accordion." We believe there lies the reason why the context information can solve the ambiguity coding problem. Further, to get more discriminative matrix, code words are reweighed as can be seen in Figures 3(b) and 3(d) (the details of reweighing can be seen in Section 4.1). After we repeat the above procedure over all classes, relationship matrixes for all classes are constructed. We denote it as $\{H^1, H^2 \ldots, H^{\mathrm{nc}}\}$, where nc is the number of classes.

We assume that every image of each class shares the common pattern in its local domain; thus, the relationship matrix that captures the partial pattern of image in local domain can be applied to describing the context information.

3.2. Context-Aware Locality-Constrained Linear Coding. After the context information has been described by the word-to-word cooccurrence matrix, it can be incorporated



(a) "Background_google" (original)

(b) "Background_google" (reweighed)

(c) "Accordion" (original)

(d) "Accordion" (reweighed)

FIGURE 3: The relationship matrixes presented in depth map. "Original" represents the matrix before reweighing, and "reweighed" represents the matrix after reweighing. As can be seen, the lightest column of the original matrix disappeared after reweighing. It is best to magnify this figure for comparison.

into the coding model. Let $\mathbf{X}$ be a set of D-dimensional local descriptors extracted from an image; that is, $\mathbf{X} = [x_1, \ldots, x_N] \in \mathbf{R}^{D\times N}$. Given a codebook with K entries, $\mathbf{B} = [b_1, b_2, \ldots, b_K] \in \mathbf{R}^{D\times K}$, and relationship matrixes for all classes, $[H_1; H_2 \ldots; H_{\text{nc}}]$. Then, we incorporate this context information into coding step by solving the following problem with respect to the template matrix H^p of pth class:

$$\mathbf{c}_i = \arg\min_{\mathbf{c}} \left\| w\mathbf{x}_i - \mathbf{B}\mathbf{c} \right\|_2^2 + \lambda \left\| \frac{\alpha \cdot \mathbf{d}_{f_i} + 1}{\beta \cdot \mathbf{d}_{ci} + 1} \odot \mathbf{c} \right\|_2^2 \quad \text{s.t.} \quad \frac{\mathbf{1}^T \mathbf{c}}{w} = \mathbf{1}, \tag{9}$$

where $\mathbf{c}_i$ is the code for $\mathbf{x}_i$ and $\mathbf{c}_{ij}$ is the jth element of $\mathbf{c}_i$. $\mathbf{d}_{f_i}$ represents the distance between x_i and $\mathbf{B}$ in feature space the same as used in LLC [15]. $\mathbf{d}_{ci}$ indicates the connected weight between x_i and $\mathbf{B}$, which can be considered as the inverse distance between x_i and $\mathbf{B}$ in context space. Particularly,

$$dc_i = \exp\left(\frac{\text{conn}(x_i, B)}{\sigma} \right), \tag{10}$$

where $\text{conn}(x_i, B) = [\text{conn}(x_i, b_1), \ldots, \text{conn}(x_i, b_K)]^T$ and $\text{conn}(x_i, b_j)$ represents the connected weight between x_i and b_j, which is obtained from the template matrix H^p according to the label of feature x_i. And σ is used for adjusting the weight decay speed for the locality adaptor in context space. λ is the regularization parameter controlling the degree of constraint in feature space and context space, α indicates the weight of locality in feature space, and β indicates the weight of locality in context space. Indeed, parameters α and β can be controlled by parameter λ. The reason we introduce these two parameters is to compare the influence of $\mathbf{d}_{f_i}$ and $\mathbf{d}_{ci}$ to

Input: x (feature to be coded), r (context size),
$\mathbf{B} \in \mathbf{R}^{D\times K}$ (dictionary), $\mathbf{H} = \{H^1, \ldots, H^{NC}\}$ (relationship matrixes)
Output: $w \in R^{1\times NC}$

{**Labeling the input feature** x}
(1) $l_x \leftarrow \arg\min_j \|x - \mathbf{B}(:, j)\|^2$,
{**Labeling the feature within the context domain**}
(2) $\Omega(x) \leftarrow$ *find the context domain of* x *with size* r
(3) $f = \{f_1, \ldots, f_N\} \leftarrow$ *find all the features in* $\Omega(x)$
(4) $l = zeros(1, N)$
(5) **for** $i = 1$ **to** N **do**
(6) $\quad l(1, i) \leftarrow \arg\min_j \|f_i - \mathbf{B}(:, j)\|^2$
(7) **end for**
{**calculating the context matching degree**}
(8) $w = zeros(1, NC)$,
(9) **for** $i = 1$ **to** NC **do**
(10) $w(1, i) = \sum_{j=1}^{N} H^i(l_x, l(1, j))/N$
(11) **end for**

Algorithm 1: Context matching degree calculation.

model performance in experiment stage. If the label of x_i is l, then $\text{conn}(x_i, b_j)$ is approximately calculated as follows:

$$\text{conn}\left(x_i, b_j\right) = H_{lj}^p \quad \left(l : \text{the label of } x_i\right). \tag{11}$$

The greater the value of H_{lj}^p is, the closer the relationship between x_i and b_j is represented in class p and the shorter the context distance between x_i and b_j will be because dc_i represents their inverse context distance, and vice versa. As a result, the response coefficient of the corresponding code word b_j is greater. Therefore, from (9), we can see that the distance between x_i and $\mathbf{B}$ in both feature space and context space controlled the response of the code words simultaneously. Thus, those similar features with different context can be encoded discriminatively.

The factor w measures the similarity (context matching degree) between the cooccurrence relationship within the context domain of the feature being coded and the corresponding cooccurrence relationship from the template matrix H^p. The details of its calculation procedure are presented as follows. If x_i is the feature to be coded with the label of l, to calculate the parameter w, firstly, we find the features $\langle f_{i1}, f_{i2}, \ldots, f_{in}\rangle$ within the context domain of x_i and their corresponding label $\langle l_{i1}, l_{i2}, \ldots, l_{in}\rangle$. Then, for each feature in the context domain, we find the value of template matrix H^p at (l, l_{ij}), $j = 1, \ldots, n$. Because the value in matrix H^p represents the strength of the cooccurrence between two code words, the sum of those values with respect to all features can denote the degree of the centering feature fitting its context for the pth image category. Therefore, the corresponding w of x_i over H^p can be calculated as

$$w_i^p = \frac{\sum_{j=1}^{n} H^p\left(l, l_{ij}\right)}{n}. \tag{12}$$

Then, w_i^p is regularized to $[0\ 1]$ by using $w_i^p = \exp(2(w_i^p - \max\_w))$, where $\max\_w$ is the max value of w_i^p ($p = 1, \ldots, \text{nc}$). Obviously, if x_i is extracted from the image in the pth category, the value of w_i^p has a higher probability to be great because its local context is similar to the context of pth image category. Otherwise, the value of w_i^p will be very small due to their dissimilar context. Additionally, from the analytical solution (details can be seen in the appendix) of (9), we can get the conclusion that the greater the value of w_i^p is, the greater the value of coding coefficient is, and vice versa. Therefore, those noisy features will produce coding coefficient with small value because their context often does not match any template context, and they will be discarded in the pooling stage if we use the max pooling strategy to get the final signature. The details of above procedures are summarized in Algorithm 1.

For each H^p, $p = 1, \ldots, \text{nc}$, we encode x_i by the above-presented method; then, we get the coding coefficient $[c_i^1, \ldots, c_i^{\text{nc}}]$, where c_i^m denotes the coding coefficient corresponding to the relationship matrix H^m for the mth image class. Therefore, given an image I with N descriptors $\mathbf{X} = \{x_i \in R^D, i \in 1, \ldots, N\}$, their coding coefficient matrix $c = [c_1^1, \ldots, c_1^{\text{nc}}, c_2^1, \ldots, c_2^{\text{nc}}, \ldots, c_N^1, \ldots, c_N^{\text{nc}}]$ with respect to the relationship matrixes $\{H^1, H^2, \ldots, H^{\text{nc}}\}$ over all classes is obtained. Then, we obtain the final signature by max pooling [11] over matrix c, which is widely used in pattern recognition tasks [16, 17, 20] because it has been proven to be consistent with the properties of the cells in visual cortex [20]. Owning to the function of parameter w, the final signature mainly preserves the coding coefficient value over the class to which the feature really belongs.

4. Implementation

In this section, we present the main details of word-to-word cooccurrence matrix construction, coding coefficient solving, and codebook learning due to their significant influence on the proposed model.

4.1. Discrimination of Word-to-Word Cooccurrence Matrix. The word-to-word cooccurrence matrix plays a key role in our method. As presented in Section 1, the reason why the ambiguity coding problem can be solved lies in that the context of ambiguous features often appears different. Therefore, more attention must be paid to the discrimination property of the relationship matrix. Intuitively, the more discriminative the word-to-word cooccurrence matrixes are, the better the performance of the model is. However, in realistic image, there are often many similar local appearances that exist in every class. For example, in the outdoor scenes, the sky often occupies very large space of the images. As a result, those features extracted from that space will be similar in terms of appearance and context, which degrades the discrimination of the relationship matrixes. As can be seen in Figures 3(a) and 3(c), some columns of the map are very light, which indicates that the corresponding code word appears close to all other words, resulting in that those code words are selected preferentially to encode any feature. To enhance the discrimination of the relationship matrix, the code word reweighing method is adopted. As demonstrated in [21], the purity of each code word is correlated with its discriminative power. To measure the purity of each code word quantitatively, we choose to use the entropy of each visual word's distribution in relationship matrix. The larger the entropy is, the less pure the code word and the smaller weight the code word should be, and vice versa. Let $\{H^1, H^2, \ldots, H^{\mathrm{nc}}\}$ be the relationship matrixes over all classes, and the words relations distribution over all classes is calculated as

$$HD = \sum_{i=1}^{\mathrm{nc}} H^i. \tag{13}$$

Therefore, the relation distribution of the ith word is the ith column of HD. Let e_i represent the entropy of the ith word b_i; then, e_i can be calculated as

$$e_i = -\sum_{j=1}^{\mathrm{nc}} HD_{ji} \ln\left(HD_{ji}\right). \tag{14}$$

The weight of the b_i can be calculated as

$$ww_i = \exp\left(-e_i\right). \tag{15}$$

By using this reweighing method, the word (e.g., the lightest column of the map in Figures 3(a) and 3(c)) with large entropy will be reweighed to near zero. As a result, the discrimination of the relationship matrixes is enhanced. The effectiveness of this method can be seen in Figures 3(b) and 3(d).

4.2. Efficiency of Coefficient Solving. Unlike some coding strategies (e.g., SC, LSC, and LCSR) that need computational iteration to get the optimal coding coefficient, CALC has an analytical solution because its object function is convex. From (9), the analytical solution of CALC can be derived by

$$c_i = w\left(1^T \Psi 1\right)^{-1} (\Psi)^{-1} 1, \tag{16}$$

where $\Psi = 2(Q + \lambda\, \mathrm{diag}(\mathbf{d}_i)^2)$, $Q = (x_i \mathbf{1}^T - \mathbf{B})^T (x_i \mathbf{1}^T - \mathbf{B})$, and

$$d_i = \frac{\alpha \cdot \mathbf{d}_{f_i} + \mathbf{1}}{\beta \cdot \mathbf{d}_{ci} + \mathbf{1}}. \tag{17}$$

The details of its derivation procedure are in the appendix at the end of this paper. In implementation, to guarantee the low reconstruction error and computational complexity, we adopt the similar approximation strategy as used in [15]. Firstly, we select k ($k \ll K$) nearest basis of x_i in feature space as candidate basis in advance, and then x_i is encoded over these k bases using the proposed model. Indeed, this strategy forms a smaller codebook $\widetilde{\mathbf{B}}$ with the size of $D \times k$, and then features are coded over it, which further improve the coding speed to a real-time level because the size of $\widetilde{\mathbf{B}}$ is much smaller than $\mathbf{B}$. For a 255×255 image with 31×31 descriptors extracted, less than 0.5 second to solve their coefficient using a CPU with a frequency of 2.7 GHz is only spent.

4.3. Optimization of Codebook Training. In Section 3, we assume the codebook is given. A simple way to generate codebook is to use clustering based methods such as k-means [8]. As demonstrated in [15], the codebook generated by this kind of method is not optimal because clustering based method is a common approach and it does not consider the specific criteria (e.g., feature space locality and context space locality) of the current model. In this section, we train a more optimal codebook and analyze the algorithm of constructing codebook in detail. According to the codebook learning method presented in [18], the specific codebook learning model for CALC can be rewritten as

$$\begin{aligned} &\min_{\mathbf{C},\mathbf{B}} \quad \sum_{i=1}^{N} \left(\left\| x_i - \mathbf{B} c_i \right\|_2^2 + \lambda \left\| \frac{\alpha \cdot \mathbf{d}_{f_i} + \mathbf{1}}{\beta \cdot \mathbf{d}_{ci} + \mathbf{1}} \odot \mathbf{c}_i \right\|_2^2 \right) \\ &\text{s.t.} \quad 1^T c_i = 1, \quad \forall i. \end{aligned} \tag{18}$$

It must be noted that this codebook optimization formulation is different from the formulation in LLC [15]. The original LLC imposes the norm-bounded constraint $\|b_i\| \leq 1$ in its codebook learning formulation, while in (18), this constraint is dropped. As demonstrated in [18], the benefits of dropping the norm-bounded constraint in (18) are twofold. First, we are able to obtain a codebook $\mathbf{B}$ which better fits the local data structure and favors classification. Second, closed-form solutions can be derived for both codebook update and sparse coding stages when solving (18), and thus faster convergence can be expected.

As suggested in [18], it can be solved using block coordinate descent or nonlinear Gauss-Seidel methods [22] to iteratively optimize $\mathbf{C}(\mathbf{B})$ based on existing $\mathbf{B}(\mathbf{C})$. We adopt the same steps of the codebook training method as in literature [18]. In the sparse coding stage (when $\mathbf{B}$ is fixed), the analytical solution of $\mathbf{C}$ exists and is unique as derived in (16). As for the codebook update stage (when $\mathbf{C}$ is fixed), we have the closed-form solution for $\mathbf{B}$ by setting its partial derivatives of $F(\mathbf{B})$ to zeroes (see the details in [18]). Theoretically, such an iterative procedure will converge to stationary points [22]. The details of optimization procedure

input: $\mathbf{X} \in \mathbf{R}^{D\times N}, \mathbf{X}_s \in \mathbf{R}^{D\times N_s}$
output: $\mathbf{B}$
(1) {**Codebook initialization stage**}
(2) $\{s_1, \ldots, s_K\} \leftarrow$ clustering the randomly sampled data $\mathbf{X}_s$ into k sets by k-means
(3) $B_{\text{init}} \leftarrow zeros(D, K)$
(4) **for** $i = 1$ **to** K **do**
(5) $\quad B_{\text{init}}(:, i) \leftarrow \sum_{j=1}^{N_i} s_{ij}/N_i, N_i \leftarrow size(s_i, 2)$
(6) **end for**
(7) $B \leftarrow B_{\text{init}}$
(8) **while (not stopping criterion)**
{**sparse coding stage**}
(9) **for** $i = 1$ **to** N **do**
(10) $\quad$ **Solving** (18) **with B fixed. The solution** $c_i = (1^T\Psi 1)^{-1}(\Psi)^{-1}1$. **when** $w = 1$. **The calculation of** Ψ **can be seen in Section 4.2**
(11) **end for**
{**codebook updating stage**}
(12) **Solving** (18) **with C fixed. The analytical solution can be referred in [18].**
(13) **End while**

Algorithm 2: Codebook optimization.

are presented in Algorithm 2, where $\mathbf{X}_s$ is the data randomly selected from the whole data, the initial codebook B_{init} is the average of each cluster, and the stopping criterion is that the objective function in (18) is no longer decreasing.

5. Analysis of CALC

Evolution. It is noted that this coding scheme degenerates into two particular cases when controlling the parameters w, α, and β. (1) When $w = 1$ and $\alpha = 1$, $\beta = 0$, it just considers the locality in feature space and CALC degenerates into the case of LLC scheme. (2) When $w = 1$ and $\alpha = 0$, $\beta = 1$, it degenerates into the case just considering the locality in the context space.

Advantages. Compared with the previous works such as VQ, SVQ, SC, LSC, and LLC, some advantages of CALC coding scheme can be presented here.

(1) *Avoiding Coding Ambiguity.* CALC encodes the feature locally in both feature space and context space. The locality in feature space guarantees the reconstruction precision while in context space guarantees the semantic coding. Thus, coding ambiguity problem is originally handled in this paper by incorporating the context information into coding procedure; meanwhile, the reconstruction precision is also guaranteed.

(2) *Noise Removing Ability.* In every image, there are lots of descriptors (e.g., the descriptors extracted from the clustering background in American flag as shown in first row of Figure 4) that are not only nonsense but also harmful for describing the image; we regard such descriptors as noise. By introducing the parameter w into CALC, the coefficient of noise is very small because its context does not often match any template context (corresponding w is small). As a result, the noise will not make any contribution to the final signature of the image with the max pooling operation. Its noise removing ability is also demonstrated by our experiment. And the experiment result can be seen in Figure 4.

(3) *Fast Computational Speed Prospects.* Due to the convexity of its object function, CALC inherits the unique advantage, an analytical solution for the object function, from the LLC coding strategy. Furthermore, unlike LSC that encodes all features simultaneously when considering their relationship, CALC encodes those features independently while preserving their relationship. These advantages lead to a real-time speed prospect under the MapReduce framework [23] in cloud computing even though dealing with massive amount of images, which make significant sense for its realistic application.

Additionally, it must be noted that CALC is different from LCSR in terms of using context information. Although context information has been used in LCSR, our method is different from LCSR in the following two aspects: (1) the main motivation of considering the context information is different. In LCSR, coding consistency in terms of local spatial domain is their purpose, while, in this paper, we aim at making the coding semantically discriminative. (2) The context description style is also different. In LCSR, the similarity of spatially neighboring patches under a fixed neighboring system is measured to describe the context, while, in this work, a word-to-word cooccurrence matrix is learnt for every class.

Figure 4: The noise removing functions of w. We select some categories from Caltech101 and Caltech256 for experiment. Local descriptors are densely extracted with step size of 8 pixels. And the blue circles represent the descriptors preserved under condition $w > 0.4$. As can be seen, most of the background descriptors are removed while the common features of each class are preserved. You may magnify this figure for details.

6. Experiment and Analysis

In this section, we conduct experiments on three widely used image datasets Scene15 [19], Caltech101 [24], and Caltech256 [25] to evaluate the proposed method. On these datasets, with the common pipeline as adopted in [10, 12, 14, 16, 17, 20, 24], we evaluate the proposed method through the following aspects. First, the effectiveness of context consideration is evaluated by comparing with LLC because CALC is an enhancement of LLC. Second, the parameters' (including context size r, dictionary size d, and parameters α, β, w) selection of the proposed method is analyzed on Caltech101. Third, we compare the performance of CALC with the state of the arts on all three datasets.

6.1. Experiment Setting. Unless indicated otherwise, in all the experiments we conducted, common experiment setting is adopted as follows to ensure consistency. For all datasets, images are first resized to keep the maximum size of height and width no more than 300 pixels. Dense SIFT features [7] are extracted from all datasets from a single scale of 16×16 patches with the step size of 8 pixels. Fairly, codebooks, using both the method in Algorithm 2 and k-means [8], are trained on a randomly selected subset of SIFTs ($\sim 10^5$ SIFTs) belonging to the training dataset. The relationship matrixes are learned using 30, 60, and 100 images randomly selected from each category of Caltech101, Caltech256, and Scene15, respectively. The candidate basis size k is set to 10. For obtaining the final signature of the images, the max pooling [11] method is adopted and the SPM [19] strategy with three levels $(1 \times 1, 2 \times 2, 4 \times 4)$ is used. The linear SVM package [26] is used for the classification task because it showed good performance when combining with the max pooling method [11]. Following the standard experimental setting, we use randomly selected 30, 30, and 100 images per class for training while leaving the remaining for test on datasets Caltech101, Caltech256, and Scene15, respectively. All the experiments are conducted under 10 times repetition, and the average accuracy of each class is finally reported.

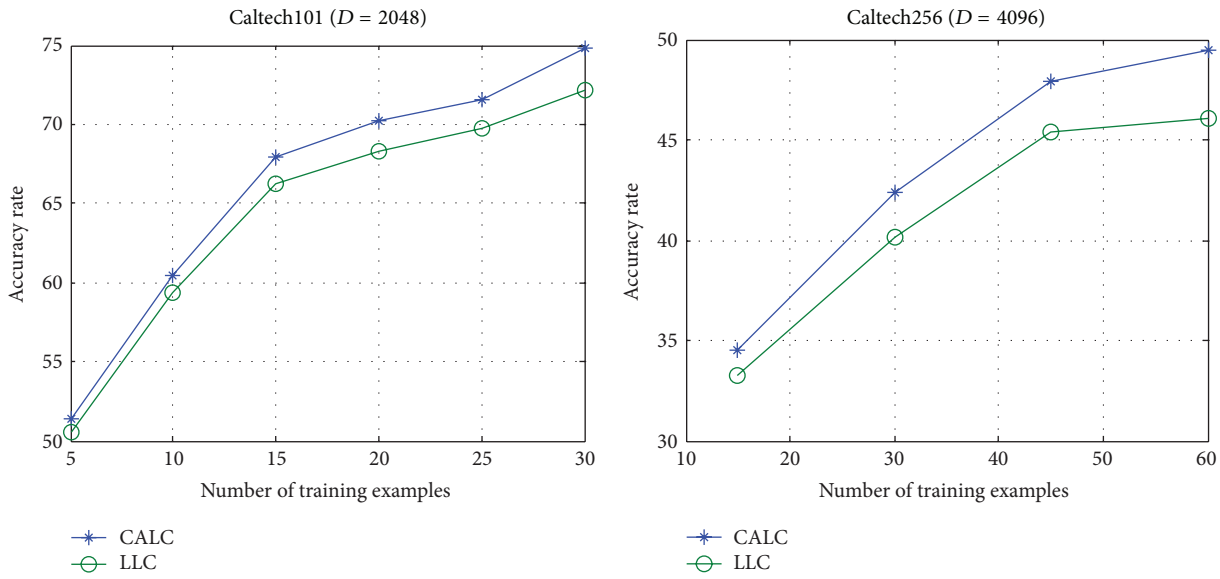


Figure 5: Comparison between CALC and LLC under different training examples.

6.2. Datasets

Scene15. This dataset contains 4485 images fallen into 15 scene categories; the number of images of each class varies from 200 to 400. Scenes are captured from the environments varying from indoor to outdoor.

Caltech101. This dataset has 101 object categories and one background category, each containing from 31 to 800 images. In contrast to the two previous datasets, containing scene images, the current task rather deals with object recognition.

Caltech256. Caltech256 contains 29,780 images belonging to 256 categories besides a background class in which none of the images belongs to these 256 categories. Comparing with Caltech101 in which objects are often in the center of images, the intraclass variances in Caltech256 are much bigger (including object location and viewpoint). As a result, object recognition task is very challenging in this dataset.

6.3. The Effectiveness of Considering Context. To evaluate the effectiveness of context factor, we compare our method with LLC by reimplementing LLC [15] based on the codes provided by its authors. This comparison experiment is conducted on Caltech101, Caltech256 with such different settings: different number of training images per category under the two different coding schemes. For fair comparison, the codebooks are constructed by using k-means. In this experiment, we select the optimal parameters ($\alpha = 0.6, \beta = 0.4, r = 17$) for CALC as analyzed in Section 6.4. As can be seen in Figure 5, our CALC outperforms LLC regardless of the size of the training images on Caltech101 and Caltech256. Hence, the effectiveness of considering context factor is demonstrated.

Furthermore, we analyze the detailed classification rate improvement on top 10 misclassified categories in Caltech101 when using LLC. From Figure 6, we see that the classification rate of our method improves significantly (the highest improvements achieve 12% on "anchor" and "platypus") to LLC on the majority of the categories even on those confused categories such as "lobster," "crab," and "crayfish." We believe that this significant improvement may be due to the context consideration. These confused categories are similar in details, and they are easy to be misclassified by using LLC because it has no ability to solve the ambiguity problem, while by using CALC, this misclassification will be alleviated because the coding ambiguity is solved by considering context.

6.4. Parameter Analysis. On Caltech101, we studied the influence of the parameters α, β, and w and the size of local context domain r on our algorithm. For comparison, we restricted $\alpha + \beta = 1$. The codebook in this experiment is learned by Algorithm 2 as presented in Section 4.1. In this experiment, two versions of w ($w = 1, w = w$) are evaluated due to its significant importance to CALC, where $w = w$ means that the value of w is calculated using (12) in Section 3.

As can be seen in Figure 7, when $w = w$, the main trend of the performance increases first and decreases then with the increase of β (with the descending of α). It implies that locality in both feature space and context space is important to the performance of coding. As presented, locality in feature space guarantees the reconstruction precision while in context space guarantees the semantic coding. The reason why the performance decreases as β further increases after achieving the top point may lie in that reconstruction error is not guaranteed when discarding the locality in feature space. It is worth noting that the best performance (75.84%) of CALC with this version improves the best result (74.9%) in Figure 4 about 1%, which is due to their difference styles in codebook construction; the former uses k-means and the latter uses Algorithm 2.

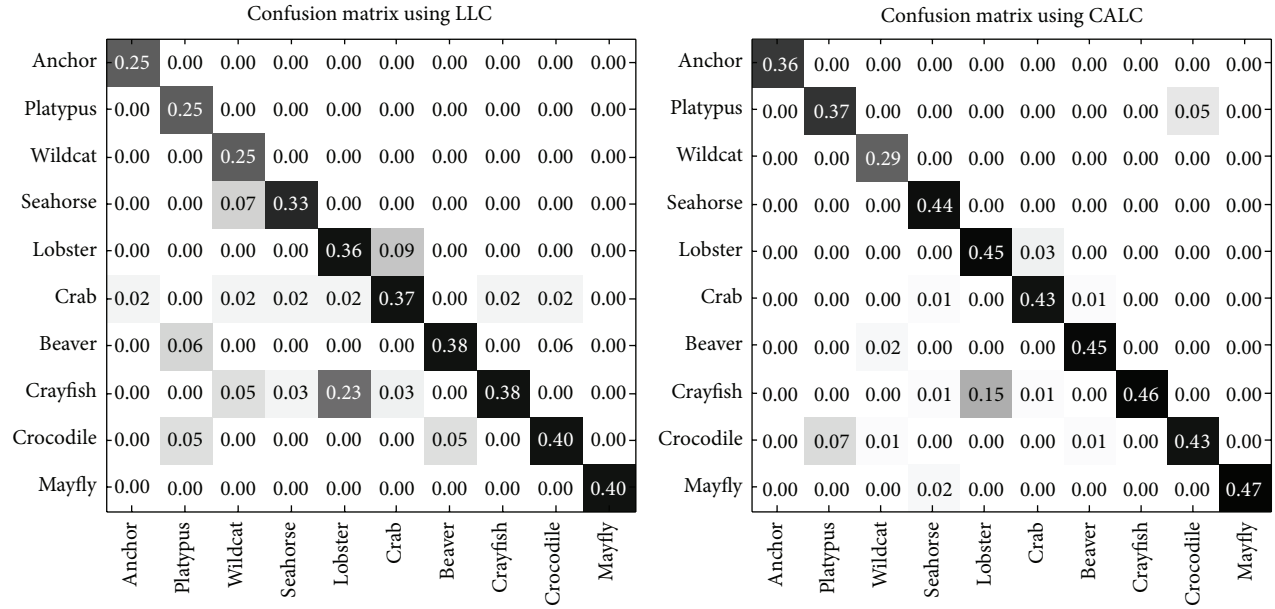


FIGURE 6: The details comparison of top 10 misclassified objects. This subconfusion matrix is extracted from the whole confusion matrix with size of 102 × 102.

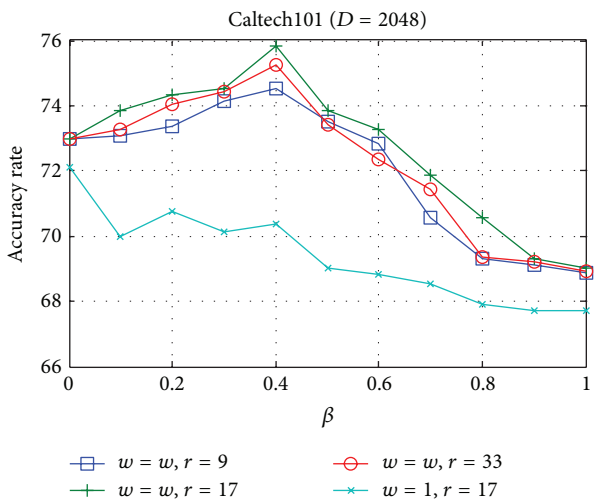


FIGURE 7: Performance of CALC under different parameters. We set $\alpha + \beta = 1$ in this experiment.

When $w = 1$, the performance of the CALC decreases with the increase of β. Note that in this case CALC degenerates into LLC when $\beta = 0$, and the performance of CALC with this version is even worse than LLC. This implies that w is very important to the performance of CALC and only with it the context consideration can be effective. We believe this may be because the max pooling in CALC is conducted over the coding coefficients corresponding to all the relationship matrixes, which leads to a confusion signature of image when $w = 1$. As a result, the performance is degraded. It also worth noting that when $\beta = 0$, the version of $w = w$ outperforms the version of $w = 1$. Indeed, when $\beta = 0$, $w = w$, the CALC enhanced the LLC ($\beta = 0$, $w = 1$) through introducing a noise removing parameter w. This result also demonstrated the noise removing function of w.

Analytically, the context size r is also very important to the CALC model. When it is too small (e.g., r is smaller than the step size of feature extraction), the relationship matrix will be a diagonal matrix, which means there is no relationship among code words but self-to-self relationship. On the contrary, when it is too great (e.g., r is greater than the size of image), each row of the relationship matrix will be the statistic histogram of the whole image; thus, all rows are the same. These two extreme cases do not satisfy the discrimination property of relationship matrix, so r must be set to a balanced value so as to satisfy the discrimination property. On Caltech101, its optimized value is 17 as shown in Figure 7. Generally, as an empiric from the experiments on different datasets, it can be set to a value able to contain 2~3 neighboring features.

The influence of codebook size on proposed model is also studied. The codebook in this experiment is constructed using K-means. As can be seen in Figure 8, the CALC outperforms LLC regardless of the codebook size on Caltech101 and Caltech256. Additionally, with the increase of codebook size, the accuracy rate of our method improves slightly when the codebook size is enough (e.g., greater than 1024 on Caltech101), which is different from LLC in which the performance is sensitive to the codebook size. We believe this may be due to that, in LLC, the bigger the codebook is, the high probability the similar feature is encoded discriminatively, while in CALC, a small codebook is enough to encode the similar feature discriminatively due to its context consideration. As a result, with a smaller codebook size, our method achieves a comparable result with LLC using a far more bigger codebook size (e.g., similar performance with codebook size 512 on CALC and 2048 on LLC on Caltech101).

Table 2: The comparison result with several coding styles on Caltech101 (training examples with size of 30), Caltech256 (training examples with sizes of 30, 60), and Scene15 (training examples with size of 100). Up the bold line are the results from the corresponding literature; below the line are the results from our implementation. And the two versions of CALC implemented are dictionary trained using k-means and Algorithm 2.

Unit: %	Cal. 101 (# 30)	Cal. 256 (# 30)	Cal. 256 (# 60)	Scene15 (# 100)
VQ [19]	64.60 ± 0.80	NA	NA	81.40 ± 0.50
SC [11]	73.20 ± 0.54	34.02 ± 0.35	40.14 ± 0.91	80.28 ± 0.93
LSC [16]	NA	35.74 ± 0.10	40.32 ± 0.32	**89.78 ± 0.40**
LLC [15]	73.44 ± NA	**41.19 ± NA**	**47.68 ± NA**	NA
LSVQ [14]	**74.21 ± 0.81**	NA	NA	82.70 ± 0.39
LCSR [17]	73.23 ± 0.81	NA	NA	87.23 ± 1.14
LLC [ours]	72.32 ± 0.91	40.32 ± 0.26	46.56 ± 0.78	81.73 ± 0.75
LSVQ [ours]	72.58 ± 0.79	38.51 ± 0.42	43.10 ± 0.11	**83.08 ± 0.56**
CALC (k-means)	**74.90 ± 0.44**	**42.37 ± 0.38**	**49.45 ± 0.67**	81.89 ± 0.54
CALC (learned)	**75.84 ± 0.56**	**43.12 ± 0.62**	**51.44 ± 0.92**	82.53 ± 0.81

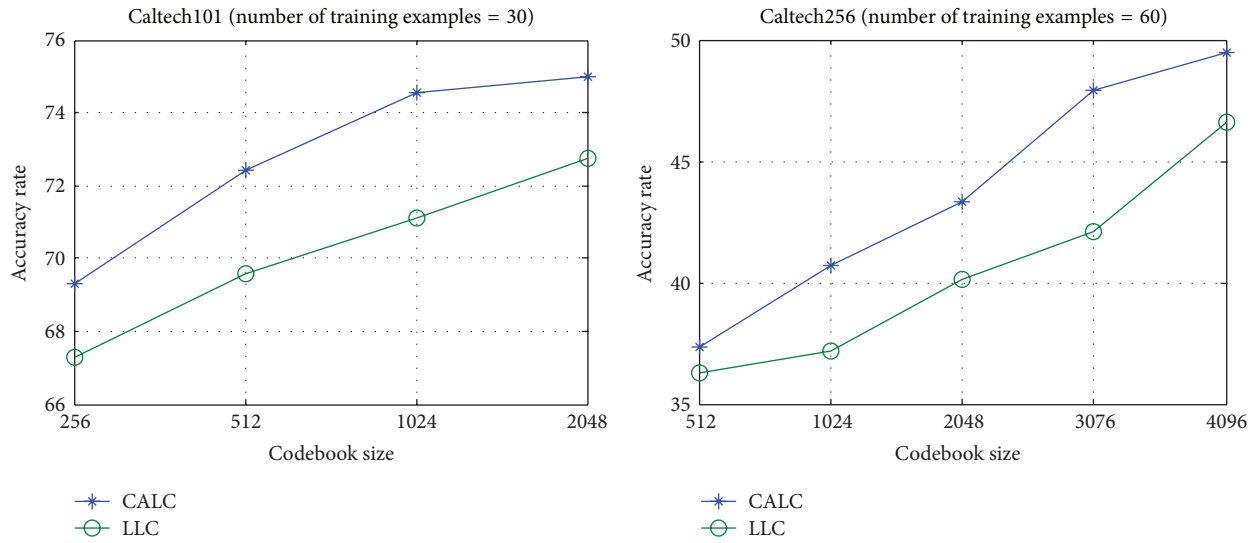


Figure 8: Performance comparison of CALC and LLC under different codebook sizes.

6.5. Comparison with the State of the Arts. In this subsection, we compare our method with several published methods on three datasets. Our comparison mainly focuses on the following two strategies: LSVQ and LLC, because these schemes are representatives of the state of the arts. We have to mention that the results of those schemes in the literatures [14, 15] are produced under different settings. For instance, LLC extracts multiscale feature every 8 pixels, a mix-order max-pooling operation is applied by LSVQ, the size of codebook varies from each other, and so forth. For fair comparison, we reimplement those methods using the same setting with our method. Meanwhile, comparing with other implementations provides a reference to evaluate the performance of our method. In this experiment, according to the parameter analysis result in Section 6.4, the same parameter setting $\alpha = 0.6$, $\beta = 0.4$, $r = 17$ is adopted on all datasets. Following the setup of LLC, we train the dictionary with sizes of 1024, 2048, and 4096 on datasets Scene15, Caltech101, and Caltech256, respectively, using k-means. Additionally, to evaluate the effectiveness of dictionary optimization algorithm, the performance of CALC using dictionary learned by Algorithm 2 is also presented. As can be seen in Table 2, our algorithm outperforms majority of the methods on three datasets under our experiment setting. The performance utilizing learned dictionary improves K-means about 1%; therefore, the algorithm proposed in Algorithm 2 is effective. In detail, the accuracy rate of our implementation version of LLC is slightly worse than the published result in [15]. We think it may be due to the difference in feature extraction. Single scale level is adopted in our implementation while three scales are used in original LLC. It also must be noted that our implementation version of LSVQ achieves a higher accuracy than the original version on Scene15; the reason may lie in that our version's codebook size is larger than that of the original version. Additionally, our method outperforms all of the methods listed on Caltech101 and Caltech256 while performs not so perfectly on Scene15. In terms of this aspect, our method is more suitable for object recognition rather

than scenes image classification. We believe this may be due to the fact that object often shares more similar context than scenes image within local domain. It also can be seen that the accuracy rate of our approach is far lower than LSVQ on Scene15. Nevertheless, on this dataset, our method is more computationally fast and it improves LLC and obtains comparable accuracy rate with the most of the listed methods.

7. Conclusion

To alleviate the ambiguity problem in coding, a novel improvement version of BOF named CALC with employing the context information is introduced in this paper. Since the context information describes the objects on the whole view, the proposed coding approach helps to alleviate the ambiguity problem and make the coding semantic at some degree. Furthermore, by introducing the feature weight parameter into the novel coding model, CALC has the ability to overcome the distortion problem produced by noisy feature. Experiment on several common used datasets demonstrated the effectiveness of the proposed method. Compared with the traditional strategies, this approach outperforms majority of the published methods in both Caltech101 and Caltech256. Furthermore, it inherits the unique advantage, an analytical solution, of LLC model, which leads to a real application prospect of this method. The experiment results also show that this method is more suitable for object recognition than scenes classification owning to the fact that object shares more common pattern in local domain than scenes share. Our future works will focus on the following aspects: (1) seeking a more robust context description method for both object and scenes image; (2) applying context information to object tracking task; and (3) conducting extensive experiment on other datasets.

Appendix

The Derivation Procedure of (16)

To determine the solution of c_i, we consider the Lagrange function $L(c_i, \eta)$, which is defined as

$$L(c_i, \eta) = \left\| w\mathbf{x}_i - \mathbf{B}\mathbf{c}_i \right\|_2^2 + \lambda \left\| \frac{\alpha \cdot \mathbf{d}_{f_i} + \mathbf{1}}{\beta \cdot \mathbf{d}_{ci} + \mathbf{1}} \odot \mathbf{c}_i \right\|_2^2 + \eta \left(\frac{1^T c_i}{w} - 1 \right). \tag{A.1}$$

Denote $(\alpha \cdot \mathbf{d}_{f_i} + \mathbf{1})/(\beta \cdot \mathbf{d}_{ci} + \mathbf{1})$ as $\mathbf{d}_i$ and considering the constraint $1^T c_i / w = 1$, the above formula can be derived as

$$L(c_i, \eta) = \left\| \left(w x_i \frac{\mathbf{1}^T}{w} - \mathbf{B} \right) c_i \right\|_2^2 + \lambda \left\| \mathbf{d}_i \odot \mathbf{c}_i \right\|_2^2 + \eta \left(\frac{\mathbf{1}^T c_i}{w} - 1 \right), \tag{A.2}$$

which can be reformed as

$$L(c_i, \eta) = c_i^T Q c_i + \lambda c_i^T \operatorname{diag}(\mathbf{d}_i)^2 c_i + \eta \left(\frac{1^T c_i}{w} - 1 \right), \tag{A.3}$$

where $Q = (x_i 1^T - \mathbf{B})^T (x_i 1^T - \mathbf{B})$, $\operatorname{diag}(\mathbf{d}_i)$ is a diagonal matrix whose nonzero elements are the entries of $\mathbf{d}_i$.

Let $\partial L(c_i, \eta)/\partial c_i = 0$; we have

$$\Psi c_i + \frac{\eta}{w} \mathbf{1} = 0, \tag{A.4}$$

where $\Psi = 2(Q + \lambda \operatorname{diag}(\mathbf{d}_i)^2)$. Once we premultiply (A.4) by $1^T \Psi^{-1}$, we obtain

$$\begin{aligned} &\mathbf{1}^T \Psi^{-1} \Psi c_i + \mathbf{1}^T \Psi^{-1} \frac{\eta}{w} \mathbf{1} \\ &= \mathbf{1}^T c_i + \mathbf{1}^T \Psi^{-1} \frac{\eta}{w} \mathbf{1} \\ &= w + \mathbf{1}^T \Psi^{-1} \frac{\eta}{w} \mathbf{1} = 0. \end{aligned} \tag{A.5}$$

So $\eta = -w^2 (1^T \Psi^{-1} 1)^{-1}$; submitting n into (A.4) gives the analytical solution

$$c_i = w \left(1^T \Psi 1 \right)^{-1} (\Psi)^{-1} 1. \tag{A.6}$$

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work is jointly supported by the National Natural Science Foundation (NSFC) of China under Projects nos. 61175006 and 61271438.

References

[1] R. Collins, A. Lipton, T. Kanade et al., "A system for video surveillance and monitoring," Tech. Rep. CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University, Pittsburgh, Pa, USA, 2000.

[2] A. Vailaya, M. A. T. Figueiredo, A. K. Jain, and H.-J. Zhang, "Image classification for content-based indexing," *IEEE Transactions on Image Processing*, vol. 10, no. 1, pp. 117–130, 2001.

[3] R. Kosala and H. Blockeel, "Web mining research: a survey," *ACM SIGKDD Explorations Newsletter*, vol. 2, no. 1, pp. 1–15, 2000.

[4] A. K. Jain, A. Ross, and S. Prabhakar, "An introduction to biometric recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 1, pp. 4–20, 2004.

[5] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *Proceedings of the 8th European Conference on Computer Vision: Workshop on Statistical Learning in Computer Vision (ECCV '04)*, pp. 1–22, Prague, Czech Republic, 2004.

[6] T. Joachims, "Text categorization with support vector machines: learning with many relevant features," in *Proceedings of the 10th European Conference on Machine Learning (ECML '98)*, Chemnitz, Germany, 1998.

[7] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[8] J. Sivic and A. Zisserman, "Video google: a text retrieval approach to object matching in videos," in *Proceedings of the 9th International Conference on Computer Vision (ICCV '03)*, pp. 1470–1477, Nice, France, October 2003.

[9] B. A. Olshausen and D. J. Field, "Sparse coding with an over complete basis set: a strategy employed by V1?" *Vision Research*, vol. 37, no. 23, pp. 3311–3325, 1997.

[10] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[11] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classfication," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '09)*, pp. 1794–1801, Miami, Fla, USA, 2009.

[12] Y. Huang, Z. Wu, L. Wang, and T. Tan, "Feature coding in image classiffication: a comprehensive study," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013.

[13] J. C. Gemert, J. Geusebroek, C. J. Veenman, and A. W. M. Smeulders, "Kernel codebooks for scene categorization," in *Proceedings of the 10th European Conference on Computer Vision (ECCV '08)*, pp. 696–709, Marseille, France, October 2008.

[14] L. Liu, L. Wang, and X. Liu, "In defense of soft-assignment coding," in *Proceedings of the 13th IEEE International Conference on Computer Vision (ICCV '11)*, pp. 2486–2493, Barcelona, Spain, November 2011.

[15] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '10)*, pp. 3360–3367, San Francisco, Calif, USA, June 2010.

[16] S. Gao, I. W. Tsang, and L. Chia, "Laplacian sparse coding, hypergraph laplacian sparse coding, and applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 92–104, 2013.

[17] A. Shabou and H. L. Borgne, "Locality-constrained and spatially regularized coding for scene categorization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, pp. 3618–3625, Providence, RI, USA, June 2012.

[18] C. Wei, Y. Chao, Y. Yeh, and Y. F. Wang, "Locality-sensitive dictionary learning for sparse representation based classification," *Pattern Recognition*, vol. 46, no. 5, pp. 1277–1287, 2013.

[19] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: spatial pyramid matching for recognizing natural scene categories," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '06)*, pp. 2169–2178, New York, NY, USA, June 2006.

[20] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio, "Robust object recognition with cortex-like mechanisms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 3, pp. 411–426, 2007.

[21] C. Zhang, J. Liu, J. Wang et al., "Image classiffication using spatial pyramid coding and visual word reweighting," in *Proceedings of the 10th Asian Conference on Computer Vision (ACCV '10)*, vol. 6494 of *Lecture Notes in Computer Science*, pp. 131–140, Queenstown, New Zealand, 2010.

[22] D. Bertsekas, *Nonlinear Programming*, Belmont, Mass, USA, Athena Scientific, 1999.

[23] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[24] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories," *Computer Vision and Image Understanding*, vol. 106, no. 1, pp. 59–70, 2007.

[25] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," Tech. Rep., California Institute of Technology, Pasadena, Calif, USA, 2007.

[26] http://www.csie.ntu.edu.tw/~cjlin/liblinear/.