Spatial Pooling of Heterogeneous Features for Image Classification

Lingxi Xie, Qi Tian, *Senior Member, IEEE*, Meng Wang, and Bo Zhang

***Abstract*— In image classification tasks, one of the most successful algorithms is the bag-of-features (BoFs) model. Although the BoF model has many advantages, such as simplicity, generality, and scalability, it still suffers from several drawbacks, including the limited semantic description of local descriptors, lack of robust structures upon single visual words, and missing of efficient spatial weighting. To overcome these shortcomings, various techniques have been proposed, such as extracting multiple descriptors, spatial context modeling, and interest region detection. Though they have been proven to improve the BoF model to some extent, there still lacks a coherent scheme to integrate each individual module together. To address the problems above, we propose a novel framework with spatial pooling of complementary features. Our model expands the traditional BoF model on three aspects. First, we propose a new scheme for combining texture and edge-based local features together at the descriptor extraction level. Next, we build geometric visual phrases to model spatial context upon complementary features for midlevel image representation. Finally, based on a smoothed edgemap, a simple and effective spatial weighting scheme is performed to capture the image saliency. We test the proposed framework on several benchmark data sets for image classification. The extensive results show the superior performance of our algorithm over the state-of-the-art methods.**

***Index Terms*— Image classification, BoF model, complementary descriptors, geometric phrases pooling, spatial weighting.**

I. Introduction

IMAGE classification has been playing a crucial role in the computer vision community. It is a basic task towards image understanding, and implies a wide range of applications. The Bag-of-Features (BoF) model is one of the most popular algorithms for image classification. An early version was proposed in [1], and many improved versions are introduced since then [2]–[4]. In essential, the BoF model is a statistics-based model aiming at providing better representation for images. For this purpose, local descriptors such as SIFT [5] are extracted from images, and a codebook is built upon all descriptors, depressing noises and forming a compact visual vocabulary for the dataset. Finally, descriptors are quantized onto the codebook, and visual words are pooled as a statistical histogram for image representation. The output of the BoF model could be applied for various tasks, such as image classification [1] and image retrieval [6].

Despite the great success of the BoF model, there still exist many drawbacks in it. These drawbacks come mainly from the well-known semantic gap [7] between low-level local descriptors and high-level image semantics. Many researchers have noticed that SIFT descriptor suffers from both synonymy and polysemy [8]. The poor description power of local descriptors limits us from learning a discriminative classification model. To overcome the following schemes are widely adopted.

- **Multiple descriptions of local patches.** For single type of descriptors might fail to capture the rich information within local patches, it is reasonable to extract multiple descriptors for compensation. Diversified descriptors represent local patches from different aspects, providing more descriptive and discriminative information. By simply concatenating appearance and shape features, [9] outperforms the models using single descriptors significantly.
- **Mid-level structure connecting low-level features and high-level concepts.** In the BoF model, an image is represented as a large set of visual words. However, there is a big semantic gap between low-level features and high-level concepts [7]. Therefore, many researchers have proposed to use mid-level structures such as macro-descriptors [10] or visual phrases [8], [11], for better image understanding. All of them bridge the semantic gap to some extent.
- **Spatial weighting of images.** Not all regions on a natural image are really useful for classification. Background clutters might bring in noises, which are harmful for training robust models. Therefore, detection of regions-of-interest (ROI) is usually proposed. Successful examples include [9] and [12], which benefit from ROI detection especially on the less aligned image categories such as some small object classes.

To benefit from the advantages from the ideas above, we propose several novel algorithms from different aspects, and also build an integrated flowchart for them to co-operate with

Manuscript received December 2, 2012; revised October 11, 2013 and March 1, 2014; accepted March 3, 2014. Date of publication March 5, 2014; date of current version March 26, 2014. This work was supported by the National Basic Research Program (973 Program) of China under Grants 2013CB329403 and 2012CB316301, the National Natural Science Foundation of China under Grants 61332007, 61273023, 91120011, 61128007, 61272393, and 61322201, the Beijing Natural Science Foundation under Grant 4132046, the Tsinghua University Initiative Scientific Research Program under Grant 20121088071, and the Open Project Program of the NLPR. This work was also supported in part to Dr. Tian by ARO grant W911NF-12-1-0057, Faculty Research Awards by NEC Laboratories of America, and 2012 UTSA START-R Research Award, respectively. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Xilin Chen.

L. Xie and B. Zhang are with the State Key Laboratory of Intelligent Technology and Systems, Tsinghua National Laboratory for Information Science and Technology, and Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, China (e-mail: 198808xc@gmail.com; dcszb@mail.tsinghua.edu.cn).

Q. Tian is with the Department of Computer Science, University of Texas at San Antonio, San Antonio, TX 78249 USA (e-mail: qitian@cs.utsa.edu).

M. Wang is with the Department of Computer Science, Hefei University of Technology, Hefei 230009, China (e-mail: eric.mengwang@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2014.2310117

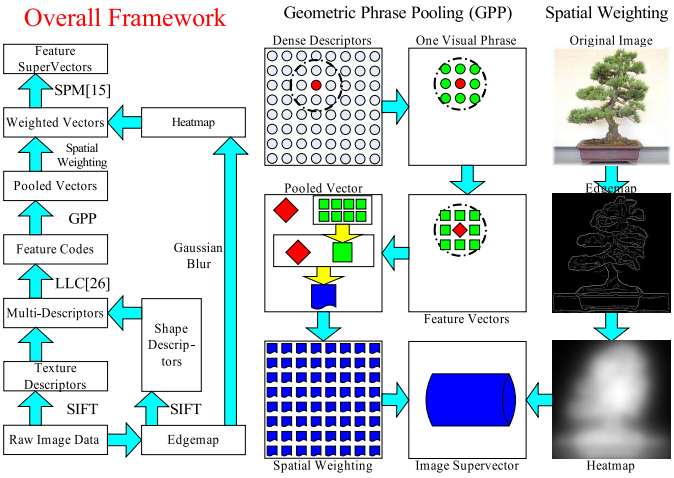

Fig. 1. The proposed model and the key modules. Left: the extended BoF model. Right: Geometric Phrase Pooling (GPP) and spatial weighting.

each other. With the help of boundary operators, we obtain an edgemap (boundary image) from each original image. Upon this, we add three building blocks into the BoF model to improve its description power. First, we simultaneously extract SIFT and Edge-SIFT descriptors and combine them in the generation of the BoF model. Earlier fusion of descriptors makes it easier to discover complementary information from both descriptors. Second, we extract geometric visual phrases upon the low-level visual words as mid-level image representation, and propose a novel pooling algorithm named Geometric Phrase Pooling (GPP) to capture the spatial contexts. Third, we apply a naive Gaussian blur process on the edgemap, obtaining a spatial heatmap for feature weighting on the image plane. Integrating all the techniques produces a much more powerful framework, which outperforms the state-of-the-art systems on various image classification datasets.

The proposed algorithm is illustrated in Fig. 1. Compared with the traditional BoF model, some differences should be marked. First, we compute edgemaps (boundary images) from original images using the Compass Operator [13], an improved version of Canny Operator [14], laying an important foundation of our algorithm. On original images as well as edgemaps, we extract two kinds of descriptors, *i.e.*, SIFT and Edge-SIFT, and mix them as a large set of local features. After codebook is trained and descriptors are quantized onto the visual vocabulary, we build geometric visual phrases as the mid-level representation of our model. Before the traditional max-pooling step, we insert two new steps, *i.e.*, phrase pooling and spatial weighting, for better description of the geometric visual phrases. Finally, we apply SPM [15] for spatial context modeling, and obtain the concatenated supervectors of images.

The remainder of this paper is organized as follows. In Section II, we introduce the traditional Bag-of-Features (BoF) model for image classification. In Section III, IV and V, we introduce the proposed algorithms, *i.e.*, extracting and fusing complementary descriptors, Geometric Phrase Pooling, spatial weighting for interesting region detection, respectively. After experimental results are shown in Section VI, we draw the conclusions in Section VII.

A preliminary version of this paper appeared as [16].

II. The Bag-of-Features Model

The Bag-of-Features (BoF) model is one of the most popular pipelines for image classification. In this section, we build a mathematical notation system for this model.

A. Local Descriptor Extraction

We start with an original **image I** which is a $W \times H$ matrix:

$$\mathbf{I} = \left(a_{ij}\right)_{W \times H} \tag{1}$$

where a_{ij} is the **pixel** on position (i, j).

Due to the limited semantic meaning of raw image pixels, we extract **local descriptors** from small patches on the image plane. There are many works on describing local patches. Among those, SIFT [5] and HOG [17] are probably the most widely used ones. They are both gradient-based histograms extracted on **interest points** of images. Detecting interest points is also a challenging problem. Since many detectors such as DoG [5] or MSER [18] sometimes fail to find semantic and discriminative patches, we use an alternative method by performing dense sampling of local patches, leading to the **Dense-SIFT** or **Dense-HOG** algorithm [19]. When the color information is useful for image understanding, it is also reasonable to calculate color SIFT descriptors, such as RGB-SIFT (calculating SIFT on red, green and blue channels individually), OpponentSIFT, HSV-SIFT, C-SIFT and so on. Among them, the OpponentSIFT descriptor is verified to outperform other ones in most cases [20].

After descriptor extraction, the image **I** could be represented as a set of local descriptors, $\mathcal{M}$:

$$\mathcal{M} = \{(\mathbf{d}_1, \mathbf{l}_1), (\mathbf{d}_2, \mathbf{l}_2), \ldots, (\mathbf{d}_M, \mathbf{l}_M)\} \tag{2}$$

where $\mathbf{d}_m$ and $\mathbf{l}_m$ denote the D-dimensional description vector and the geometric location of the m-th descriptor, respectively. M is the total number of dense descriptors, which could be hundreds or even thousands under dense sampling.

It is verified that SIFT and HOG descriptors are only good at describing texture features. To capture other important properties such as shape and color, it is reasonable to extract other kinds of descriptors. Systems with multiple types of descriptors [9] have been proposed, showing a much better performance over those using single type of descriptors.

B. Quantization for Descriptors

After the descriptors have been extracted, they are often quantized to be compact. For this purpose, we train a **codebook B** using descriptors from the whole dataset. The codebook is a $B \times D$ matrix, or B vectors with dimension D, each of which, *i.e.*, $\mathbf{c}_b$, is called a **codeword**. Most often, the codebook is constructed with K-Means clustering algorithm. Recent years, there are also some works targeting at improving the efficiency and performance of K-Means [21], [22], and building discriminative codebooks for large-scale image applications [23].

Next, the local descriptors are projected onto the codebook for a histogram representation. This process is named **coding**, for we are actually encoding each descriptor into a

sparse vector. Hard quantization strategy presents a descriptor using single codeword, resulting in a large quantization error. In recent years, soft quantization methods have been proposed as alternatives to hard quantization. By projecting a descriptor onto the subspace spanned by a small group of codewords, it produces smaller quantization errors, which is verified more effective in visual representation. Sparse Coding [24], [25] and Locality-constrained Linear Coding [26] techniques are such cases. Given a codebook with B codewords, the quantization vector or **feature vector** for a descriptor $\mathbf{d}_m$ would be a B-dimensional vector $\mathbf{v}_m$. We name $\mathbf{v}_m$ the corresponding **visual word** of **descriptor** $\mathbf{d}_m$.

C. Feature Pooling

After all the local descriptors are quantized as visual words, we shall aggregate them for global image representation. We call this step **feature pooling**, for we are putting visual words into a pool for statistics. For this purpose, two major pooling strategies are often used. The **max-pooling** strategy calculates the maximal response on each codeword:

$$\mathbf{w} = \max_{1 \le m \le M} \mathbf{v}_m \tag{3}$$

where the notation $\max_m$ denotes the element-wise maximization. Differently, the **average-pooling** strategy calculates the average response:

$$\mathbf{w} = \frac{1}{M} \sum_{m=1}^{M} \mathbf{w}_m \tag{4}$$

Here $\mathbf{w}$, a K-dimensional vector, is named **representation vector** or **feature vector** of the image.

Some researchers have discussed the choice of max-pooling versus average-pooling [10], showing that max-pooling gives more discriminative representation under soft quantization strategies, while average-pooling fits hard quantization better. Recently, various methods have been proposed to integrate both pooling methods to improve their effectiveness. For example, the Geometric ℓ_p-norm Pooling algorithm [12] proposes a generalized ℓ_p-norm pooling strategy and uses a complex optimization to find the best p for each image.

D. Spatial Context Modeling

Spatial context could greatly help us understand the semantics of images [27]. Therefore, various models are proposed for constructing spatial structures. Among those, one of the most successful trials is Spatial Pyramid Matching (SPM) [15], in which images are divided into hierarchical subregions for individual pooling, and the pooled vectors are concatenated as a **super-vector**. Similar method is also generalized onto orientational spaces [28]. For datasets with relatively better alignment such as Caltech101 [29], SPM improves classification accuracy by an impressive margin of 10%. However, it shows little improvement on the less aligned datasets.

Another line for spatial context modeling is to use visual phrases [30], [31]. Compared with visual words, visual phrases are more descriptive and robust [32], therefore produce more semantic features for discriminating similar image samples. It is verified that visual phrases are good at capturing co-occurring patches in image recognition tasks [11]. Spatial coding of visual phrases are also widely used in large-scale Web image search and retrieval systems [27].

E. Classification Models

Before sending the feature vectors into the SVM, feature normalization is considered a crucial data pre-processing step. One of the most popular feature normalization methods is the ℓ_p-norm normalization, in which we divide each feature vector with its length in the ℓ_p space so that all the vectors become ℓ_p-unit-length. In [26], the authors claim that ℓ_1-norm produces much lower classification accuracy than ℓ_2-norm, whereas it is verified in [33] that with a large enough normalization coefficient, the ℓ_1-norm formula would give comparable performance with the ℓ_2-norm. In [33], the authors also discuss several advanced normalization methods for the part-based classification models.

Image classification tasks are usually configured with very long feature vectors and relatively smaller number of images. Therefore, the Support Vector Machine (SVM) is often taken as the default choice of classifier. It is verified that different choices of kernels would severely impact the classification accuracy, and the non-linear kernels such as χ^2 often gives higher performance than the linear inner-product kernel. However the latter one is proved more efficient and scalable [34], therefore is widely adopted in the classification tasks with large number of categories. In some cases, we can also use the Hellinger's kernel, or Bhattacharya's kernel, to produce feature vectors with less values falling in the close neighborhood of 0 [35]. Although it is non-linear, we can still fit it in the linear model with a simple square-root transformation.

F. The Algorithms for Comparison

We mainly compare our model with LLC [26], a recent implementation of the BoF model with single type of descriptors. It serves as an efficient baseline which is easily to be tested on various image collections. The other one is a more complicated system [9], in which various techniques are adopted, including multiple descriptors, detection of regions-of-interest, and so on. We show that our algorithm produces higher classification accuracy using a better designed flowchart.

III. Complementary Descriptors

In this section, we propose a novel idea using complementary descriptors for image classification. First, we introduce a new kind of image descriptors named Edge-SIFT, which are extracted on the edgemap (boundary image) of the original image. Though SIFT and Edge-SIFT descriptors are calculated on different images, they share the same physical meaning, *i.e.*, histogram of gradients, in their corresponding dimension, therefore we could simply mix them on the image space to train the BoF model. We test our model and provide detailed analysis on the effect of using multiple types of descriptors, and also make a short comment on the early fusion strategy to reveal its advantages. Finally, we discuss on the limitations of the proposed descriptor fusion strategy.

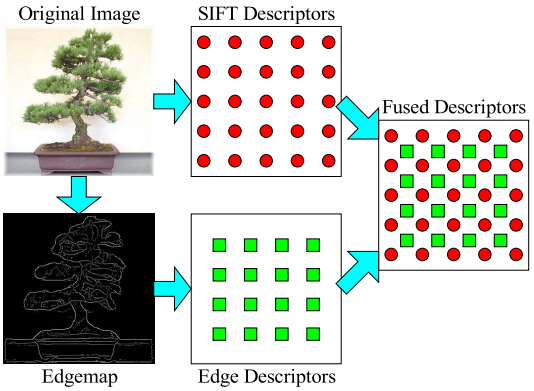

Fig. 2. Fusing two sets of dense descriptors on the image plane.

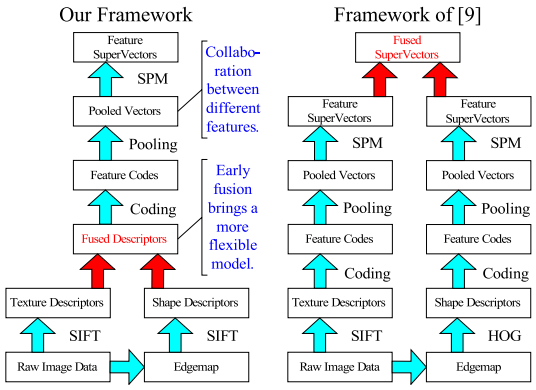

Fig. 3. Difference between our model and [9] (best viewed in color PDF). Red arrows and texts highlight the fusion step in each framework.

A. SIFT and Edge-SIFT Descriptors

For a $W \times H$ image $\mathbf{I}$, we extract dense SIFT [5] descriptors from the image. Denote the set of SIFT descriptors as $\mathcal{M}_{\mathrm{S}}$:

$$\mathcal{M}_{\mathrm{S}} = \left\{ \left(\mathbf{d}_{\mathrm{S}1}, \mathbf{l}_{\mathrm{S}1}\right), \left(\mathbf{d}_{\mathrm{S}2}, \mathbf{l}_{\mathrm{S}2}\right), \ldots, \left(\mathbf{d}_{\mathrm{S}M_{\mathrm{S}}}, \mathbf{l}_{\mathrm{S}M_{\mathrm{S}}}\right) \right\} \tag{5}$$

where the subscript S stands for SIFT, and M_{S} is the number of SIFT patches on the image plane.

As we know, SIFT descriptors are effective on describing texture features, but less effective to capture the shape information. To overcome, we can introduce shape descriptors to help understanding the semantics. Following [9], we apply a boundary detector on image $\mathbf{I}$, producing another $W \times H$ grayscale image $\mathbf{I}_{\mathrm{E}}$:

$$\mathbf{I}_{\mathrm{E}} = \left(e_{ij}\right)_{W \times H} \tag{6}$$

where e_{ij}, a floating value in [0, 1], is the significance quantity of pixel (i, j) located on an edge. We call $\mathbf{I}_{\mathrm{E}}$ the corresponding **edgemap** (boundary image) for the original image $\mathbf{I}$.

We use Compass Operator [13] for boundary detection. Some detected edgemaps are shown in Figs. 2, 4, 5, and 6, respectively. On the edgemaps, texture details of the objects are filtered and the shape features become more clear. Therefore, it is reasonable to extract another set of SIFT descriptors on the edgemap for shape description. We call them **Edge-SIFT descriptors** to differ from the **original SIFT descriptors**. Denote the set of Edge-SIFT descriptors as $\mathcal{M}_{\mathrm{E}}$:

$$\mathcal{M}_{\mathrm{E}} = \left\{ \left(\mathbf{d}_{\mathrm{E}1}, \mathbf{l}_{\mathrm{E}1}\right), \left(\mathbf{d}_{\mathrm{E}2}, \mathbf{l}_{\mathrm{E}2}\right), \ldots, \left(\mathbf{d}_{\mathrm{E}M_{\mathrm{E}}}, \mathbf{l}_{\mathrm{E}M_{\mathrm{E}}}\right) \right\} \tag{7}$$

Similarly, the subscript E stands for Edge-SIFT, and M_{E} is the number of descriptors, which could be different from M_{S} due to the different spatial strides and window sizes used in dense sampling.

It is worth noting that both SIFT and Edge-SIFT descriptors are histograms of gradients, therefore they share the same physical meaning on the corresponding dimensions (histogram bins). We could naturally combine them together to capture both texture and shape features on the images.

B. Fusing Descriptors

Following the basic idea, we simply unite the set of SIFT and Edge-SIFT descriptors on the image plane:

$$\mathcal{M} = \mathcal{M}_{\mathrm{S}} \cup \mathcal{M}_{\mathrm{E}} \tag{8}$$

Here, $\mathcal{M}$ is the union descriptor set for image representation. The number of descriptors in $\mathcal{M}$ is denoted as M, which satisfies $M = M_{\mathrm{S}} + M_{\mathrm{E}}$.

We illustrate the fusion operation on two sets of descriptors in Fig. 2. Here we shall emphasize that the fusion process preserves both the description vector and location of original descriptors. It is shown later that this strategy gives us natural benefits by extracting geometric visual phrases consisting of both texture and shape visual words. Of course, except for the original image and boundary image (edgemap), one can extract more kinds of SIFT descriptors on other images such as saliency map or contour map. To fuse them, we only need to guarantee that all the descriptors share the same physical meaning on the corresponding dimensions, for we are comparing and combining the descriptors dimension-wise at the clustering and quantization steps. Throughout this paper, we only consider two set of SIFT descriptors extracted on the original and boundary images.

Finally, we shall make a short comment to compare our work with [9]. In [9], descriptors are also extracted from original and boundary images respectively. However, two types of descriptors are individually processed through the BoF model, until fusion operation is performed on the pooled representation of both images to form a concatenated supervector. As we could see in Section IV-E, late fusion limits the flexibility of the model, and makes it difficult to construct mid-level structures consisting of both kinds of descriptors. On the contrary, we finish the fusion step much earlier, leaving plenty of room for mid-level structures. We illustrate the difference between the models in Fig. 3, and will further discuss the advantages of our model in Section IV-E, after the visual phrases are introduced and adopted to improve the feature representation in Section IV.

C. Experiments and Discussion

Now, we test our model on the Caltech101 dataset [29]. Besides the clutter category, the dataset contains 101 different classes of objects, covering a wide range of animals, plants and man-made tools. We inherit the baseline system [26], and use different sets of descriptors, *i.e.*, SIFT descriptors, Edge-SIFT descriptors, and fused set of descriptors as local features. We report the classification accuracy (30 training images per

TABLE I

CALTECH101 RESULTS ON THE FUSED SET OF DESCRIPTORS AND ON THE SINGLE SET (SIFT OR EDGE-SIFT) OF DESCRIPTORS. THE NUMBERS IN PARENTHESES ARE THE SPATIAL STRIDES AND WINDOW SIZES FOR DENSE SAMPLING, RESPECTIVELY

Desc #1	Desc #2	Only #1	Only #2	Fused
SIFT(7,7)	SIFT(6,12)	74.41%	72.69%	75.14%
SIFT(7,7)	Edge(6,12)	74.41%	73.08%	78.75%
SIFT(7,7)	SIFT(7,12)	74.41%	73.77%	75.75%
SIFT(7,7)	Edge(7,12)	74.41%	72.89%	**78.94%**
SIFT(7,7)	SIFT(8,12)	74.41%	73.60%	75.32%
SIFT(7,7)	Edge(8,12)	74.41%	72.93%	78.92%

TABLE II

COMPARISON OF TWO KINDS OF DESCRIPTORS ON THE CALTECH101 DATASET. THE UPPER AND LOWER PARTS LIST THE CATEGORIES BEST CLASSIFIED WITH SIFT AND EDGE-SIFT DESCRIPTORS, RESPECTIVELY

Category Name	SIFT	Edge	Difference
wild cat	57.50%	30.00%	27.50%
water lily	65.71%	38.57%	27.14%
crocodile	33.00%	13.00%	20.00%
ferry	89.46%	70.81%	18.65%
hedgehog	82.50%	65.83%	16.67%
anchor	44.17%	73.33%	29.17%
butterfly	50.16%	72.62%	22.46%
wrench	57.78%	77.78%	20.00%
pyramid	71.48%	87.04%	15.56%
saxophone	75.00%	90.00%	15.00%

category, averaged on 10 individual runs) to compare the performance of different sets of descriptors.

Table I shows our results on different sets of descriptors. The best classification accuracy is achieved when we fuse different kinds of descriptors, *i.e.*, SIFT and Edge-SIFT, while we observe significantly lower accuracy when the same kind of descriptors, *i.e.*, both SIFT, are merged. Therefore, the compensation between two kinds of descriptors is clear.

To give a better intuition to this finding, we return to systems using single set of descriptors. We choose parameters from the best post-fusion performance in Table I, *i.e.*, the spatial strides of SIFT and Edge-SIFT are both 7 pixels, while the window sizes of local patches are chosen to be 7 and 12 pixels, respectively. To compare different sets of descriptors, we evaluate the category-wise classification accuracies in both models, and list the top-5 categories with the largest accuracy variations in Table II, and the top-3 with sample images in Fig. 4.

It is clear that different types of objects are better described using different types of descriptors. For objects with less deformation such as manmade tools, the better strategy is to ignore their texture details and pay more attention on the boundary image. Therefore, Edge-SIFT descriptors give a better description on these objects. However, there are also a number of objects in which texture features are more discriminative than shape features, such as animals and plants, in which it is better to preserve texture details in the original images and use SIFT descriptors.

Since the SIFT and Edge-SIFT descriptors are complementary for local feature description, it is reasonable to preserve

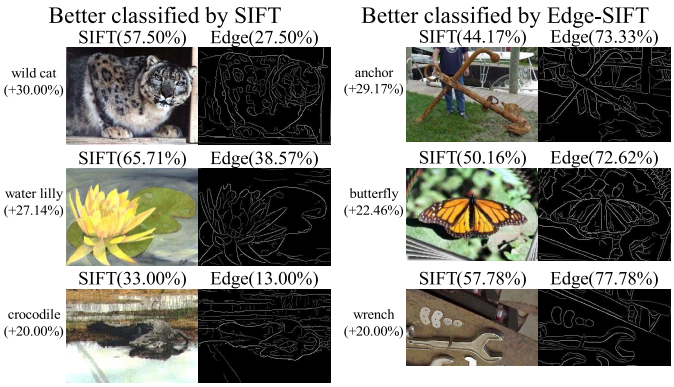

Fig. 4. Comparison of two kinds of descriptors. Classification accuracies with single set of descriptors are listed above the sample images.

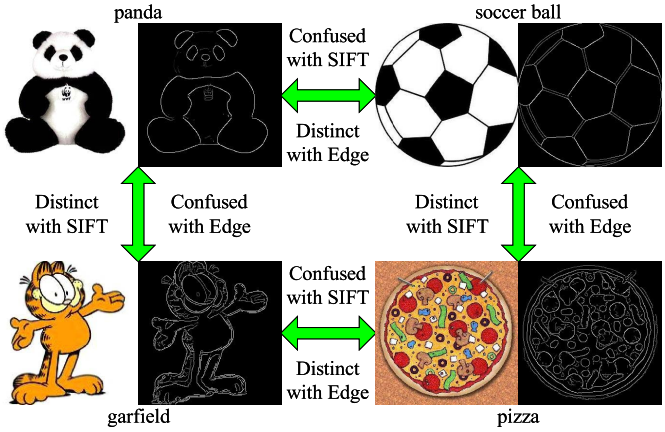

Fig. 5. Fused descriptors help to distinguish confusing categories.

both of them for a more robust image representation. Fig. 5 shows an example of four categories.

For example, both *panda* and *football* images contain black-and-white patches, but the edges detected on *panda* images are more likely to be curves while straight line segments on *football* images. Therefore, Edge-SIFT would be more discriminative when classifying these two categories. Similar situations are also observed between other pairs shown in the image. To summarize, it is difficult to distinguish the four concepts using any single kind of descriptors, only when we extract both kinds of descriptors makes it easier to capture the discriminative features for categorization.

D. Limitation

Although the proposed fusion algorithm gives superior performances on the Caltech101 dataset, it does not mean that we could simply transplant the method onto any other cases. To illustrate this, we test our model on another dataset containing 17 categories of flowers [36], and summarize the classification results with different sets of descriptors in Table III. We observe that best classification results are produced using SIFT descriptors alone, both using Edge-SIFT descriptors alone and fusing complementary descriptors cause the classification accuracy to drop dramatically.

To explain, we return to the classification algorithms with single type of descriptors, and list the category-wise accuracies in Table IV. We can easily find that SIFT descriptors work

TABLE III
OXFORD FLOWER-17 RESULTS ON THE FUSED SET OF DESCRIPTORS AND ON THE SINGLE SET (SIFT OR EDGE-SIFT) OF DESCRIPTORS. THE NUMBERS IN PARENTHESES ARE THE SPATIAL STRIDES AND WINDOW SIZES FOR DENSE SAMPLING, RESPECTIVELY

Desc #1	Desc #2	Only #1	Only #2	Fused
SIFT(8,16)	Edge(10,16)	69.51%	27.23%	65.48%
SIFT(8,16)	Edge(12,16)	69.51%	28.66%	64.90%
SIFT(8,16)	Edge(16,16)	69.51%	27.43%	66.72%
SIFT(12,16)	Edge(10,16)	69.52%	27.23%	63.34%
SIFT(12,16)	Edge(12,16)	69.52%	28.66%	63.55%
SIFT(12,16)	Edge(16,16)	69.52%	27.43%	65.56%

TABLE IV
COMPARISON OF TWO KINDS OF DESCRIPTORS ON THE OXFORD FLOWER-17 DATASET. SIFT DESCRIPTORS WORK BETTER ON ALL THE CATEGORIES

Category	SIFT	Edge	Category	SIFT	Edge
001	**66.27%**	16.13%	002	**59.07%**	22.53%
003	**58.80%**	22.40%	004	**68.80%**	10.93%
005	**43.73%**	18.53%	006	**68.40%**	48.13%
007	**81.07%**	23.20%	008	**26.53%**	7.47%
009	**80.40%**	44.40%	010	**95.33%**	50.53%
011	**93.73%**	22.27%	012	**44.53%**	22.53%
013	**83.60%**	52.80%	014	**61.07%**	15.73%
015	**87.33%**	22.53%	016	**84.53%**	43.60%
017	**78.67%**	43.47%			

better on all the 17 flower categories, which means that Edge-SIFT descriptors do not provide very useful compensation for flower description. This is different with what we observe in Caltech101 (see Table II), where some categories are better described using SIFT and some others better using Edge-SIFT. We further show some examples of flower categories in Fig. 6. Since color information is much more important than shape in distinguishing the flowers, it is not instructive to fuse both kinds of descriptors so that low-quality features water down the high-quality ones, decreasing the robustness of image representation and producing worse classification results.

Of course we shall admit that adding new kinds of features should help for image understanding, given the added features provide better classification results than random guess. However it is worth noting that in the image classification problem the number of training samples is often limited, therefore we need to extract compact feature vectors so as to prevent the over-fitting on low-quality feature dimensions. In general, before we fuse two kinds of descriptors for image classification, we can first conduct experiments with single type of descriptors and category-wisely compare the classification accuracies. If one type of descriptors significantly outperform the other set on almost all the categories, it should imply that fusing both descriptors might be harmful for providing discriminative features.

As an example, we turn to fine-grained image classification tasks which are very popular in recent years. Such datasets, such as the Caltech-UCSD Birds-200-2011 dataset [43], often contain a large number of objects with very similar semantics, say, 200 species of birds. The large inter-class similarity makes it difficult for both (texture and shape) descriptors to work efficiently. Since many previous works [38]–[40] have shown

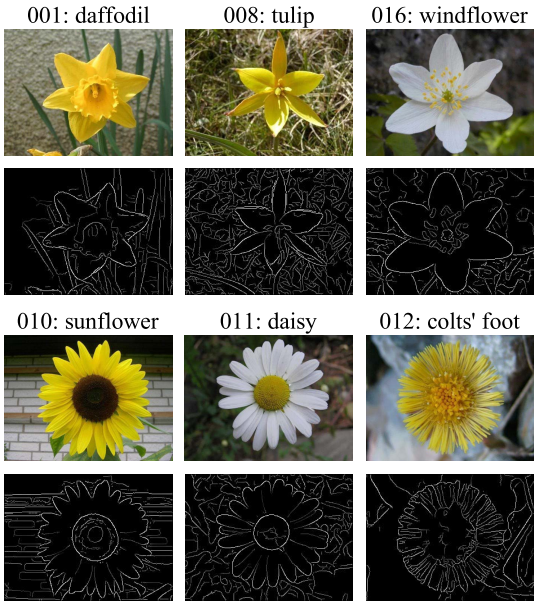

Fig. 6. In the Oxford Flower-17 dataset, images are discriminative with SIFT descriptors, but confusing with Edge-SIFT ones, in which the important texture details are filtered out. Top and Bottom rows show two groups of sample images.

that objects such as birds and flowers are more discriminative with texture and color features than shape ones, it is reasonable to preserve only one set of color-SIFT descriptors on such image collections.

To summarize, the motivation of fusing multiple descriptors come from their compensation in describing different objects. If the precondition does not hold, the fusion operation might introduce noises into the model and produce poor results. To judge this, it is helpful to test individual descriptors for classification and compare their category-wise accuracies.

IV. GEOMETRIC PHRASE POOLING

In the previous section, we present a simple idea using complementary local features for image classification. However, it is worth noting that different kinds of descriptors are individually encoded in the BoF model, which limits us from constructing efficient mid-level structures to capture the feature contexts in local groups. In this section, we shall introduce a simple local structure named geometric visual phrase, and propose a novel encoding algorithm, Geometric Phrase Pooling (GPP), with intuitive explanations. GPP could be considered as a mid-level image representation model connecting low-level features and high-level concepts. We illustrate our algorithm in the middle column of Fig. 1.

A. The GPP Algorithm

We start with Equation (8), and rewrite it as the same form of Equation (2):

$$\mathcal{M} = \{(\mathbf{d}_1, \mathbf{l}_1), (\mathbf{d}_2, \mathbf{l}_2), \ldots, (\mathbf{d}_M, \mathbf{l}_M)\} \tag{9}$$

To begin with, we give a new definition of geometric visual phrases. Following [8], we consider a visual phrase as a set of visual words which are close to each other. As local patches

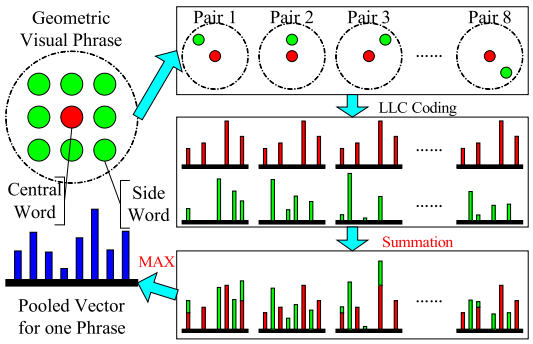

Fig. 7. Illustration of the Geometric Phrase Pooling (GPP) algorithm. We group the central word and each side word as a pair and sum their feature vectors together. At last, maximization is performed on the whole phrase.

on images are not organized so well as textual words, we ignore their order and regard a phrase as a disordered set of visual words. Following this basic idea, we propose a simple algorithm. Define a positive integer K, $K \ll M$, we search for the K nearest neighbors (by ℓ_2 distance) on the image plane and construct a word group for each descriptor $(\mathbf{d}_m, \mathbf{l}_m)$, $m = 1, 2, \ldots, M$:

$$\mathcal{G}_m = \left\{ \left(\mathbf{d}_{m,0}, \mathbf{l}_{m,0}\right), \ldots, \left(\mathbf{d}_{m,K}, \mathbf{l}_{m,K}\right) \right\} \tag{10}$$

Here, $\mathcal{G}_m$ is the m-th **geometric visual phrase**. The **central word** of $\mathcal{G}_m$ is defined as the zeroth descriptor $\left(\mathbf{d}_{m,0}, \mathbf{l}_{m,0}\right)$, which is simply $(\mathbf{d}_m, \mathbf{l}_m)$ itself. The **location** of phrase $\mathcal{G}_m$ is defined by $\mathbf{l}_{m,0}$. Other K descriptors are called **side words**. K is the **order** of $\mathcal{G}_m$, which contains $K + 1$ words.

Suppose that we have trained a codebook $\mathbf{B}$ with B codewords. For a phrase $\mathcal{G}_m$, we compute LLC [26] encoding for each of its words. LLC is a sparse coding scheme. Given $\mathbf{B}$ and number of bases r (most often $r \ll B$), it produces feature vectors with at most r nonzero elements among totally B dimensions. For $\mathcal{G}_m$, there will be $K+1$ sparse feature vectors, one for each visual word. Denote $\mathbf{v}_{m,k}$ as the feature vector of k-th word in $\mathcal{G}_m$.

Now, the **Geometric Phrase Pooling** (GPP) algorithm is very easy to implement. We calculate a B-dimensioanl feature vector $\mathbf{w}_m$ for each visual phrase $\mathcal{G}_m$, $m = 1, 2, \ldots, M$:

$$\mathbf{w}_m = \max_{1 \le k \le K} \left\{ \mathbf{v}_{m,0} + \mathbf{v}_{m,k} \right\} \tag{11}$$

$$= \mathbf{v}_{m,0} + \max_{1 \le k \le K} \mathbf{v}_{m,k} \tag{12}$$

where the notation $\max_k$ denotes element-wise maximization on K vectors with B dimensions. Equation (11) is the core equation of GPP, while (12) is an equivalent version for easier implementation. We illustrate the simple working mechanism of (11) in Fig. 7.

It is worth emphasizing that GPP is an extra module between the coding and pooling steps of the BoF model. After GPP, we still need to perform max-pooling over all the visual phrases, instead of visual words, to obtain a feature vector $\mathbf{w}$ to represent the whole image:

$$\mathbf{w} = \max_{1 \le m \le M} \mathbf{w}_m \tag{13}$$

B. Deep Insights for GPP

The formulation of GPP (11) is very easy to implement. However, the intuition behind the simple algorithm is not that straightforward. Here, we clarify the advantages we obtain to give a better understanding of GPP.

First of all, let's go back to Equations (11) and (13). Simple derivation gives:

$$\mathbf{w} = \max_{1 \le m \le M} \mathbf{w}_m \tag{14}$$

$$= \max_{1 \le m \le M} \left\{ \max_{1 \le k \le K} \left\{ \mathbf{v}_{m,0} + \mathbf{v}_{m,k} \right\} \right\} \tag{15}$$

$$= \max_{1 \le m_1, m_2 \le M, m_1 \diamond m_2} \left\{ \mathbf{v}_{m_1} + \mathbf{v}_{m_2} \right\} \tag{16}$$

where $m_1 \diamond m_2$ means that $\mathbf{v}_{m_1}$ and $\mathbf{v}_{m_2}$ are adjacent words, *i.e.*, when one of them is taken as central word, the other is one of the side words. Hence, $\mathbf{w}$ is the maximization over summations of all adjacent visual word pairs, *i.e.*, disordered pair (m_1, m_2) satisfying $m_1 \diamond m_2$. Define $\mathbf{v}_{m_1} + \mathbf{v}_{m_2}$ as the **contribution to GPP** from the word pair (m_1, m_2).

Now, recall the formulation of max-pooling and rewrite it into an equivalent though redundant version:

$$\mathbf{w} = \max_{1 \le m \le M} \mathbf{v}_m \tag{17}$$

$$= \max_{1 \le m \le M} \left\{ \max_{1 \le k \le K} \left\{ \max\left\{ \mathbf{v}_{m,0}, \mathbf{v}_{m,k} \right\} \right\} \right\} \tag{18}$$

$$= \max_{1 \le m_1, m_2 \le M, m_1 \diamond m_2} \left\{ \max\left\{ \mathbf{v}_{m_1}, \mathbf{v}_{m_2} \right\} \right\} \tag{19}$$

Naturally, $\max\left\{ \mathbf{v}_{m_1}, \mathbf{v}_{m_2} \right\}$ is defined as the **contribution to max-pooling** from word pair (m_1, m_2). Obviously, the contribution term is the only difference between (16) and (19).

For an adjacent word pair (m_1, m_2), consider its contributions to GPP and max-pooling. For simplicity, we denote feature vectors for them as $\mathbf{v}_1$ and $\mathbf{v}_2$, respectively. If $\mathbf{v}_1$ and $\mathbf{v}_2$ have no nonzero dimensions in common, we have $\max\{\mathbf{v}_1, \mathbf{v}_2\} = \mathbf{v}_1 + \mathbf{v}_2$. which means that the word pair contributes equally to max-pooling and GPP. However, if there are common nonzero dimensions in $\mathbf{v}_1$ and $\mathbf{v}_2$, things will be different: this word pair would contribute more to GPP by assigning a larger value on the overlapping dimensions. Our intuition is illustrated in Fig. 8. In the following part of this paper, we say the word pairs with common nonzero dimensions **really** contribute to GPP.

From the analysis above, we have learned that only word pairs with common nonzero dimensions would **really** contribute to GPP. In our framework, we perform descriptor quantization using LLC [26], a locality-sensitive coding scheme. Two descriptors are encoded by intersected sets of codewords only if they are close in the feature space. Therefore, GPP latently selects those word pairs both similar and adjacent, and enhance their common nonzero responses.

There is an acknowledged observation that on natural images, local patches with the same semantics are more likely to be correlated, *i.e.*, visually similar. Therefore for a geometric visual phrase consisting of one central word and K side words, the correlation between central word and side words will be high (visually similar) if this geometric visual phrase is located

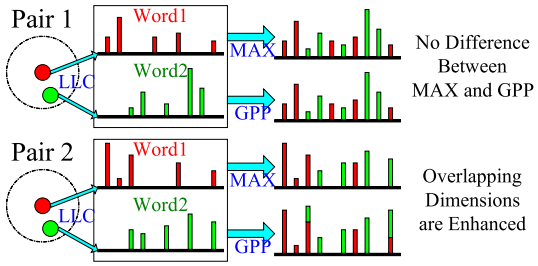

Fig. 8. Intuition of GPP (best viewed in color PDF). Cases of non-overlapping word pair and partial overlapping word pair are presented respectively.

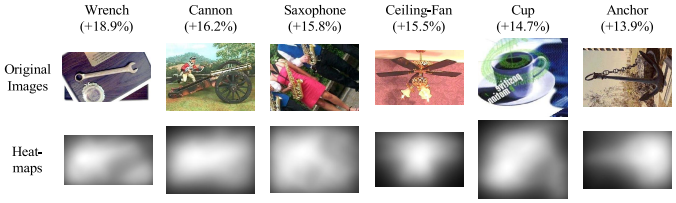

Fig. 9. Heatmaps generated from the percentages of useful pairs of visual words on the image plane. Numbers in parentheses are accuracy gains on those categories, from LLC to GPP.

on a semantic patch. This high correlation turns to be the common nonzero dimensions of the visual words. Therefore, GPP enhances the responses of such highly correlated patches, or visual word groups, in a latent way.

We conduct an interesting experiment to verify our statements. For each visual phrase, we count the percentage of side words with common nonzero dimensions with the central word. This could be considered as one kind of statistics to estimate the enhancement of visual phrases. Smoothing it gives us a heatmap illustrating the visual correlations on the corresponding patches. Fig. 9 shows some examples from the Caltech101 dataset. We could find that the significantly enhanced visual responses are mainly located on the semantic regions. By assigning larger feature values on these regions, GPP produces more discriminative representations than LLC encoding.

In summary, we can conclude that GPP is indeed a superior mid-level encoding algorithm with a clear intuition, convincing improvements, and a very easy implementation.

C. Enhancing GPP

In this section, we boost the performance of GPP by proposing three simple ideas which are very easy to implement. The improvements are summarized in Fig. 10.

1) Extracting Longer Phrases: In Section IV-B, we observe that word pairs with similar appearance and geometric location would **really** contribute to GPP. To search for more such pairs, it is reasonable to increase K, the order of geometric visual phrases. However, a longer visual phrase could also contain more irrelevant visual word pairs, which is harmful for the robustness. In practise, we don't set K automatically, but choose the best parameter after testing a wide range from 5 to 30. Testing results are plotted in Fig. 11(b). Throughout the rest part of this paper, we use $K = 20$ for experiments.

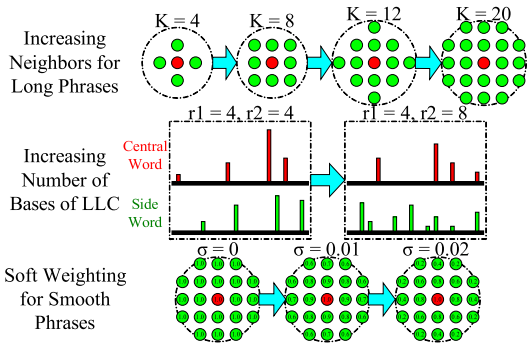

Fig. 10. Illustration of 3 different improvements for GPP (best viewed in color PDF). Above: increasing K for longer phrases. Middle: enlarging the number of bases in the LLC encoding for more overlapping nonzero dimensions. Below: soft weighting on geometric visual phrases, where floating numbers are the soft weights on the visual words.

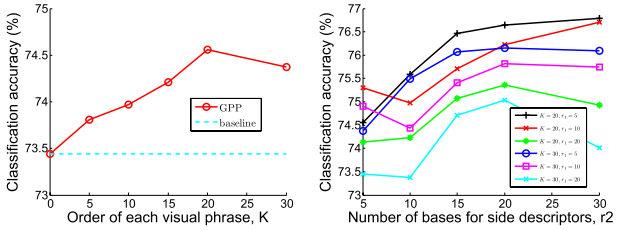

Fig. 11. (a) Impact of the order of visual phrases, K. The baseline performance of LLC, 73.44%, is plotted in a dashed line. (b) The numbers of bases of LLC will also impact on the accuracies. Differences in K and r_1 are presented with different polylines.

2) Larger Number of Bases for Encoding: As LLC [26] is a sparse encoding algorithm, the number of bases r is much smaller than the visual codebook size B. In this case, histogram representations of adjacent visual words could hardly overlap, therefore the percentage of word pairs that **really** contribute to GPP is small. Increasing r produces more useful word pairs, but also damages the locality of LLC. The selection of r is a tradeoff between description and robustness.

The central word is the most significant component in a visual phrase, therefore its robustness is more important than side words. For this, we use a larger r for the side words and yet a smaller r for the central one. To clarify, we denote r_1 and r_2 as the numbers of bases for encoding central words and side words, respectively, and test several combinations of r_1 and r_2 as shown in Fig. 11(a). Following the plotted results, we use $r_1 = 5$ and $r_2 = 30$ in the later experiments.

3) Soft Weighting for Smooth Phrases: By intuition, if the distance between visual words is large, the relationship between them is loose. Therefore, we apply an exponential decay on the side words by assigning lower weights onto the distant ones. In precise, for a side word $(\mathbf{d}_{m,k}, \mathbf{l}_{m,k})$ with the center word $(\mathbf{d}_{m,0}, \mathbf{l}_{m,0})$, we penalize it with a soft weight s_k defined as:

$$s_k = \exp\{-\sigma_w \times \|\mathbf{l}_0 - \mathbf{l}_k\|_2\} \tag{20}$$

where σ_w is the smoothing parameter on side words, and $\|\cdot\|_2$ is the Euclidean distance. Now, the original GPP formulation

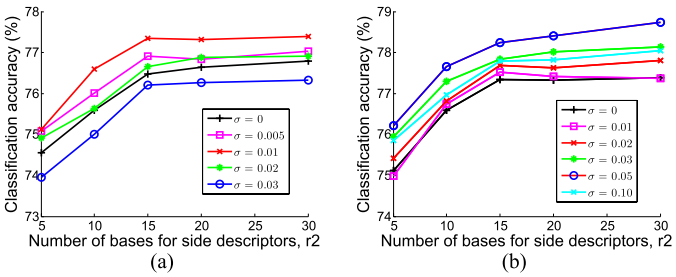

Fig. 12. (a) Impact of the smoothing parameter, σ_w (σ in plotting for short). We have plotted results of $\sigma_w = 0$ for comparison. (b) Impact of the smoothing parameter, σ_w (σ in plotting for short). $\sigma_e = 0$ means uniform spatial weighting, and produces the same curve as the best one in Fig. 12(a).

(12) becomes smoother:

$$\mathbf{w}_m = \mathbf{v}_{m,0} + \max_{1 \le k \le K} s_k \times \mathbf{v}_{m,k} \tag{21}$$

To choose a proper smoothing parameter, we test different choices of σ_w in Fig. 12(a). The best parameter in practise, $\sigma_w = 0.01$, is selected for later experiments.

In conclusion, we generalize the GPP algorithm by introducing several adjustable parameters, *i.e.*, the order of phrases K, numbers of encoding bases r_1 and r_2, and smoothing parameter σ_w. We can summarize from Fig. 11(a)–12(a) that GPP is not very sensitive to each individual parameter, since the maximal accuracy difference between the best and worst parameters is relatively small, but we do obtain a better classification model when we use a relatively better set of parameters. With the parameter set we find, *i.e.*, $K = 20$, $r_1 = 5$, $r_2 = 30$ and $\sigma_w = 0.01$, the classification accuracy, 77.39%, is much higher than the baseline performance 73.44%, which verifies that GPP is helpful to provide more discriminative image representation. Although the parameters are tuned on the Caltech101 dataset, we will show in Section VI that these parameters also produce good performance on a variety of image collections.

D. Time Complexity and Sparsity

Here, we test the time complexity and the feature sparsity of GPP to show its simplicity and efficiency. To make comparison, we implement LLC and GPP with different parameters. We construct different sizes of codebooks, and record the average time used in coding and pooling for each image, as well as the average percentage of nonzero dimensions in the feature super-vector $\mathbf{w}$.

Results are plotted in Figs. 13(a) and (b). Owe to the simplicity, the proposed algorithm is very efficient to carry out. In average, the GPP module requires no more than 0.4 extra seconds (about twice of original time cost) on a single image, which is much better than those complicated algorithms such as GLP [12] which requires much longer. On the other hand, Fig. 13(b) reveals that GPP generates much denser feature vectors for image representation, especially in the scenarios with longer phrases and larger number of encoding bases on side words. A comment needs to be made here. In recent years, people have been debating that sparse feature vectors are more efficient in image classification than dense features [25].

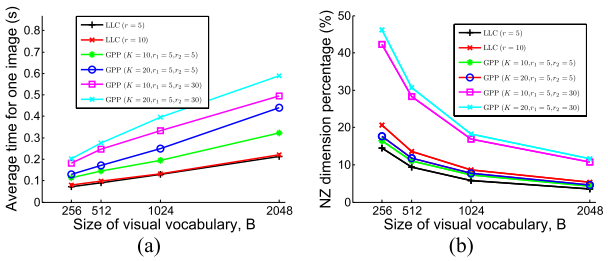

Fig. 13. (a) Average time required for each image. We have used 4×2.0GHz CPUs for testing. (b) Average percentage of nonzero dimensions in the pooled and concatenated super-vectors.

TABLE V
CALTECH101 CLASSIFICATION RESULTS ON THE MODELS WITH DIFFERENT FUSION AND POOLING STRATEGIES

	Late Fusion [9]	Early Fusion (Ours)
LLC [26]	79.12%	78.94%
GPP (Ours)	79.48%	**81.36%**

However, our experiments give a neutral argument on this topic: denser features do not mean to provide worse results, but we need a proper structure to organize the feature space. GPP actually provides a natural solution to encoding more information into the feature vectors of the same length.

E. Early Fusion vs. Late Fusion

Finally, we continue the discussion in Section III-C on early fusion versus late fusion. On the Caltech101 dataset, we implement the basic BoF model (LLC) as well as the GPP algorithm (GPP) in the context of early fusion and late fusion strategies, and list the classification results in Table V. From the table, we could observe the advantage of early fusion over late fusion. When the geometric visual phrases are not introduced, early fusion strategy does not provide higher accuracy than late fusion. However, when we extract geometric visual phrases on the early-fused descriptors, it captures spatial contexts containing both texture and shape features as shown in Fig. 14. This results in a notable improvement on the early fusion system.

V. SPATIAL WEIGHTING

Traditional BoF model uses all the extracted local descriptors for image representation, however some of them might not fall on the objects we really want to recognize. Since such descriptors often introduce noises into the model, it is reasonable to filter them for better image understanding. This is equivalent to learning a spatial weighting, a saliency map, or simply a heatmap on the image plane. In this paper, we follow the observation that higher contrast regions provide stronger stimuli to vision [41], and propose a simple spatial weighting strategy through a Gaussian blur process on the boundary images.

First of all, we calculate an edgemap (boundary image) for the original image $\mathbf{I}$ of size $W \times H$. Following (6), the edgemap $\mathbf{I}_{\mathrm{E}}$ is another $W \times H$ matrix in which the elements represent the

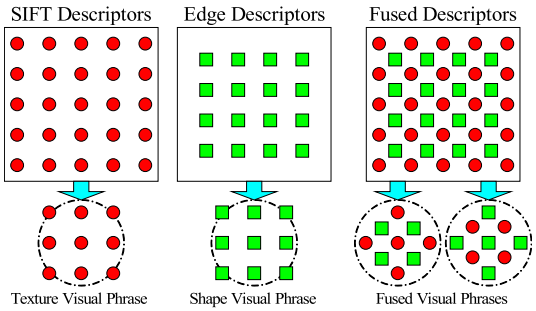

Fig. 14. Applying GPP on the fused descriptors helps to generate more discriminative visual phrases.

intensity of edge responses. We thereafter calculate a $W \times H$ **weighting matrix W**:

$$\mathbf{W} = \left(w_{ij}\right)_{W \times H} \tag{22}$$

Here, w_{ij} is the spatial weight at position (i, j), which is accumulated from the decayed edge responses:

$$w_{ij} = \sum_{i', j'} e_{i'j'} \times \exp\left\{-\sigma_e \left\|(i, j) - \left(i', j'\right)\right\|_2\right\} \tag{23}$$

Here, coordinate $\left(i', j'\right)$ are enumerated on the whole image, σ_e is the smoothing parameter on edge responses, and $\|\cdot\|_2$ is the Euclidean distance. As σ_e goes up, there shall be smaller weights accumulated on the faraway pixels.

With spatial weights, the max-pooling formulation (13) becomes:

$$\mathbf{w} = \max_{1 \le m \le M} \{w_m \times \mathbf{w}_m\} \tag{24}$$

where w_m is the weight at $\mathbf{l}_m$, the central pixel of $\mathcal{G}_m$.

To evaluate the proposed spatial weighting scheme as well as the smoothing parameter σ_e, we again test the classification accuracy on the Caltech101 dataset using the best parameters learned from the previous section. We set the smoothing parameter σ_e as 0, 0.01, 0.02, 0.03, 0.05 and 0.10, respectively, and plot the classification results with different settings in Fig. 12(b). The parameter which gives the best performance, *i.e.*, $\sigma_e = 0.05$, will be used in later experiments.

To give a better visualization on the effect of smoothing parameter σ_e, we plot the corresponding weighting matrices $\mathbf{W}$ as heatmaps in Fig. 15 for comparison. From the gradually changing heatmaps, the impact of σ_e becomes very clear. When σ_e is small, the heatmap of spatial weighting is similar to a uniform distribution on the image plane. As the smoothing parameter goes up, the spatial weights become more concentrated around the boundary responses. If the parameter becomes too large, *e.g.*, $\sigma_e = 0.20$, the obtained heatmap is very similar to the edgemap. This forces the model discard texture details, which is harmful for recognizing some cases such as animals and plants.

The heatmaps generated by GPP (Fig. 9) and edgemap (Fig. 15) look similar. The difference lies in that, GPP enhances those local regions with co-occurrence of similar features, whereas edgemap gives higher weight onto the regions with strong edge response. Both of them provide useful information for image classification.

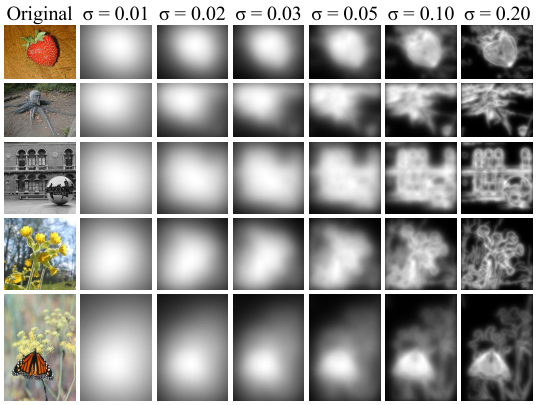

Fig. 15. The gradually changing heatmaps for the original images (left column). When the smoothing parameter is higher, the spatial weights are more concentrated. The sample images are selected from: strawberry (Caltech101), octopus (Caltech101), MIT-inside-city (Scene-15), 014-cowship (Oxford Flower-17) and monarch-open (Butterfly-7), respectively. All the selected categories benefit from the spatial weighting scheme in classification accuracies.

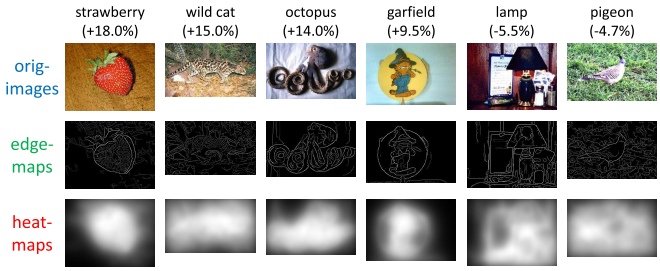

Fig. 16. Edgemaps and weighting heatmaps of images from those categories with most accuracy gains (first four) and drops (last two) after spatial weighting. Numbers in parentheses are the differences in accuracy.

It is worth noting that detecting the saliency regions on the image is itself an open problem in computer vision. Certainly, our algorithm could not completely solve the problem, but we provide a simple and efficient algorithm which provides useful information for image classification. To illustrate this, we calculate the classification accuracies by category, and list the most increased and decreased ones in Fig. 16. We can see that our algorithm works well on the situations with relatively simple background clutters, but could also harm the classification accuracy in the categories with poor saliency detection results. Since the proposed algorithm produces larger accuracy gains than drops, the averaged classification accuracy is boosted.

Finally let us consider the time complexity of the proposed spatial weighting algorithm. It is easy to note that (23) requires a complete enumeration on every pairs of pixels, therefore is very computational expensive, *i.e.*, takes more than 30 seconds on a single-core CPU for a 300×300 image. To accelerate, we adopt an approximation by skipping the accumulation of the pairs with Euclidean distances larger than 50 pixels. Under the best smoothing parameter, *i.e.*, $\sigma_e = 0.05$, the maximal ignored coefficient could be $\exp\{-0.05 \times 50\} \approx 0.08$, which is relatively small. With the reasonable approximation, our algorithm only requires less than 0.5 second on a single image, which is very efficient in practise considering the spatial weighting is computed only once.

VI. Experimental Results

In this section, we show the experimental results on several publicly available image classification datasets. To compare our method with other works, we inherit the same settings from the state-of-the-art algorithms, and adopt descriptor fusion, GPP, and spatial weighting with the best settings learned from the previous sections.

- **Boundary detection.** We use the Compass Operator [13] for boundary detection. The radius parameter σ is fixed as 4 as proposed in the same literature.
- **Image descriptors.** We use the VLFeat [42] library to extract dense SIFT descriptors. The spatial stride and window size are discussed individually for each dataset.
- **Codebook construction.** We use K-Means for clustering. The codebook size is 4096 for Caltech256, 8192 for Pascal VOC 2007, and 2048 for others. The number of descriptors for clustering does not exceed 2 million.
- **Coding and phrase pooling.** We use LLC [26] for local feature coding and apply GPP with the best parameters: $K = 20$, $\sigma_w = 0.01$, $r_1 = 5$ and $r_2 = 30$.
- **Spatial weighting.** We take $\sigma_e = 0.05$ for the edge-based spatial weighting scheme.
- **Spatial Pyramid and normalization.** We apply a 3-layer $(1 \times 1 + 2 \times 2 + 4 \times 4)$ SPM for enhancing the global spatial context. After that, an ℓ_2-norm normalization is performed to produce comparable feature vectors.
- **SVM for classification.** We use LibLINEAR [34], a recent scalable SVM implementation for training and testing. For the Pascal VOC retrieval task, we rank the testing images according to their confident scores.
- **Accuracy evaluation.** For the Pascal VOC 2007 Challenge, we use the standard benchmark [**?**]. On other datasets, we select fixed numbers of images for training the classification model, and test it on the remaining images to calculate the average classification accuracy over all the categories. We repeat the random selection 10 times and report the averaged results.

A. The Caltech101 Dataset

The Caltech101 dataset [29] contains 9144 images of 102 classes, including a background category. There exists significant deformation among different objects from the same category. Sample images are listed in Fig. 17.

The spatial stride and window size for SIFT descriptors are 7 and 7, while for Edge-SIFT are 7 and 12. We use 5, 10, 15, 20, 30 images per category for training, and others for testing. Results are concluded in Table VI. In all cases, GPP outperforms LLC by more than 9%, and even more than 10% in the scenarios of fewer training samples.

B. The Caltech256 Dataset

The Caltech256 dataset [44] contains 30607 images of 257 classes, including a clutter category. It is a expansion of Caltech101, and also much more challenging, for there are severe intra-class variations and inter-class similarity, and objects are less aligned. Sample images are shown in Fig. 18.

TABLE VI
Classification Results on the Caltech101 Dataset

# training	5	10	15	20	30
Lazebnik [15]	–	–	56.4	–	64.6
Yang [25]	–	–	67.0	–	73.2
Wang [26]	51.15	59.77	65.43	67.74	73.44
Boureau [10]	–	–	–	–	75.7
Bosch [9]	–	–	–	–	81.3
Ours	**61.90** ±0.54	**71.75** ±0.60	**76.03** ±0.63	**78.53** ±0.39	**82.45** ±0.59

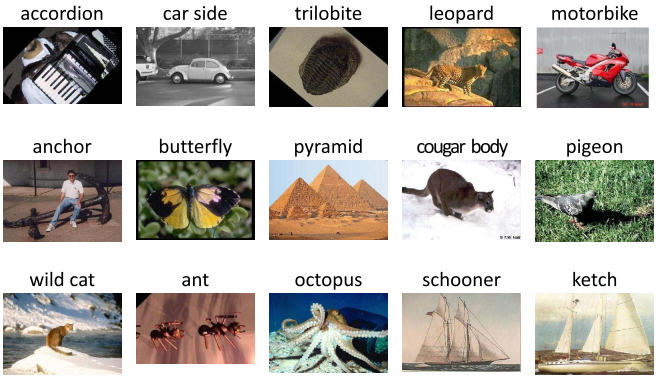

Fig. 17. Sample images from the Caltech101 dataset. Upper: categories on which our algorithm reports 100% accuracies (30 trains). Middle: categories on which our algorithm most outperforms LLC [26]. Bottom: categories on which our algorithm produces poor results. Images on the first and second rows are correctly classified, while others are not.

Fig. 18. Sample images from the Caltech256 dataset. Upper: categories on which accuracies are 90% or higher (30 trains). Middle: images with small objects. Bottom: images with many objects. Images on the first row are correctly classified, while others are not.

The spatial stride and window size for SIFT descriptors are 5 and 5, while for Edge-SIFT are 8 and 8. We use 5, 15, 30, 45, 60 images per category for training, and others for testing. Results are concluded in Table VII. Again, the classification results of GPP shows advantages over the results of LLC [26].

C. The Pascal VOC 2007 Dataset

As a competition set, the Pascal VOC 2007 dataset [?] contains 9963 images and 20 kinds of objects. From sample images listed in Fig. 19, we could find it a challenging dataset, for the significant varying of the appearances, scales, numbers and locations of the objects. The task is to train an individual retrieval model in the training set for each of the 20 objects, and use it to find other images in the testing set containing

TABLE VII
CLASSIFICATION RESULTS ON THE CALTECH256 DATASET

# training	5	15	30	45	60
Griffin [44]	–	28.3	34.1	–	–
Yang [25]	–	27.73	34.02	37.46	40.14
Gao [45]	–	29.77	35.67	38.61	40.30
Wang [26]	–	34.36	41.19	45.31	47.68
Bosch [9]	–	–	44.0	–	–
Ours	**26.12**	**36.35**	**45.07**	**48.02**	**50.33**
	±0.21	±0.31	±0.24	±0.25	±0.18

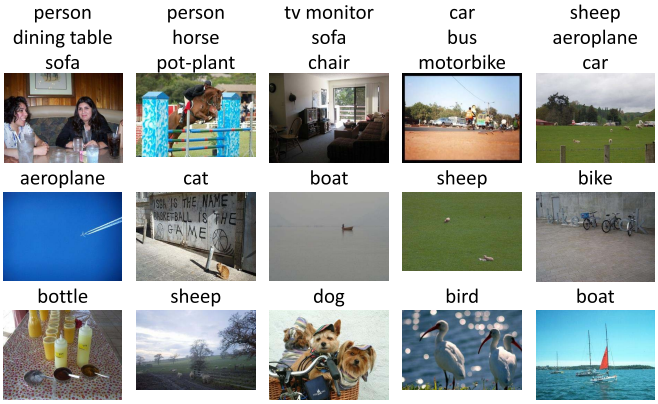

Fig. 19. Sample images from the Pascal VOC 2007 dataset. Upper: images with objects from multiple categories. Middle: images with small objects. Bottom: images with more than one objects from the same category. All the objects on the bottom two rows are not retrieved (wrongly classified) by our algorithm.

TABLE VIII
RESULTS LISTED BY CATEGORY OF THE PASCAL VOC 2007 DATASET

name	[26]	Ours	improv.	name	[26]	Ours	improv.
aeropl	67.47	72.29	**4.82**	bicycl	55.29	56.33	**1.04**
bird	40.68	45.41	**4.72**	boat	58.56	61.26	**2.71**
bottle	21.19	26.24	**5.05**	bus	44.10	53.77	**9.68**
car	69.43	73.56	**4.13**	cat	46.73	52.18	**5.44**
chair	51.50	54.19	**2.70**	cow	31.21	40.78	**9.58**
dining	35.06	47.40	**12.33**	dog	39.00	41.58	**2.57**
horse	72.41	74.38	**1.98**	motorb	53.98	57.52	**3.55**
person	79.18	83.02	**3.84**	potted	18.77	26.03	**7.80**
sheep	33.14	37.51	**4.37**	sofa	44.73	52.30	**7.57**
train	66.59	69.51	**2.92**	tvmoni	40.96	47.50	**6.53**
average	48.50 [a]	53.64 [a]	**5.14**				

[a] The average precision presented here is not the same as the reported ones, *i.e.*, 49.13 and 53.89, for the reported ones (higher) are averaged on the best accuracies on all the categories among different codebooks, and the ones presented here (a bit lower) is obtained from a single testing process.

the same object. We use the standard benchmark provided by Pascal Challenge, which calculates the mean Average Precision (mAP) measure for each object individually.

The spatial stride and window size for SIFT descriptors are 6 and 4, while for Edge-SIFT are 7 and 12. Individual accuracy on each object and the averaged score are listed in Table VIII. We can find that our algorithm outperforms the LLC [26] in each of the 20 individual task, and report a 53.89% average accuracy, winning LLC (49.13%) remarkably by 4.76%.

D. The Butterfly-7 Dataset

The Butterfly-7 dataset [46] contains 619 images of 7 different species of butterflies. In each category, we can find

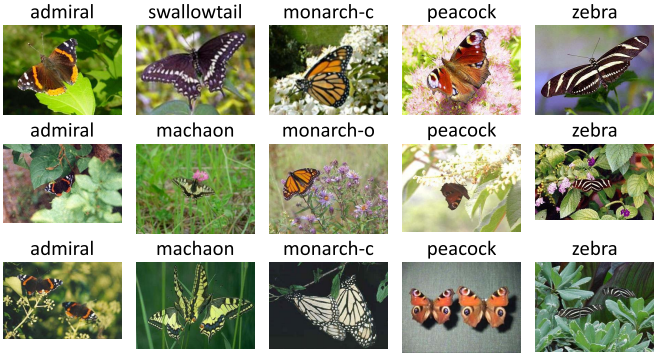

Fig. 20. Samples from the Butterfly-7 dataset. Upper: normal cases. Middle: difficult cases for object detection. Bottom: cases with multiple objects. Images on the first row are correctly classified, while others are not.

TABLE IX
RESULTS ON THE BUTTERFLY-7 DATASET

# training	1	5	10	20	26
Lazebnik [46]	–	–	–	–	90.4% [a]
Larlus [47]	–	–	–	–	90.61% [b]
Wang [26]	54.41	77.98	83.38	86.33	87.54
Ours	**64.25**	**80.19**	**85.30**	**88.93**	**90.83**
	±6.46	±1.72	±1.97	±1.00	±1.34

[a] They used a complex part-based model, which is much more computationally expensive than our algorithm.

[b] This is the accuracy on the best run, not averaged. The best run among our 10 individual tests gives a 93.31% accuracy.

a number of challenging samples, including small objects and multiple objects. Example images are listed in Fig. 20.

In this dataset, we use the OpponentSIFT descriptors [20] with spatial stride of 7 and window size of 12. Due to the discussion in Section III-D, we do not use Edge-SIFT descriptors in this case. We use 1, 5, 10, 20, 26 (the number specified in [46]) images per category for training, and others for testing. Experimental results are shown in Table IX.

E. The Oxford Flower-17 Dataset

The Oxford Flower-17 dataset [36] contains 17 classes of flowers with 80 images per category. The flowers suffer from large variations of scale, viewpoint angle and illumination. Example images could be found in Fig. 21.

In this dataset, we use the OpponentSIFT descriptors [20] with spatial stride of 12 and window size of 16. Due to the discussion in Section III-D, we do not use Edge-SIFT descriptors in this case. We use 5, 10, 20, 30, 60 images per category for training, and others for testing. Experimental results are shown in Table X.

F. The Scene-15 Dataset

The Scene-15 dataset [15] contains 15 scenes and 4485 images. All the instances are grayscale images collected from outdoor environments. It is one of the most widely used dataset for scene understanding tasks, in which we need to discriminate various categories of scenes listed in Fig. 22.

The spatial stride and window size for SIFT descriptors are 5 and 5, while for Edge-SIFT are 8 and 8. We use 10, 20, 30, 50, 100 images per category for training, and others for testing. Results are shown in Table XI.

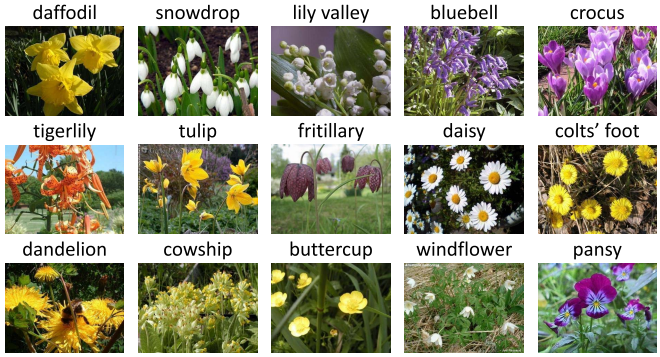

Fig. 21. Wrongly classified samples from the Oxford Flower-17 dataset. Many images from the dataset contain more than one objects of interest.

TABLE X
RESULTS ON THE OXFORD FLOWER-17 DATASET

# training	5	10	20	30	60
Nilsback [36]	–	–	–	–	81.3% [a]
Gehler [48]	–	–	–	–	85.5% [a]
Wang [26]	69.52	76.98	82.43	84.94	88.24
Ours	**72.08**	**79.39**	**84.47**	**86.94**	**91.56**
	±1.60	±0.87	±1.16	±0.67	±1.61

[a] They used a fixed data split, and a different performance measure with us. Under the specified setting, the reported accuracy is 91.43%, which is much higher than theirs.

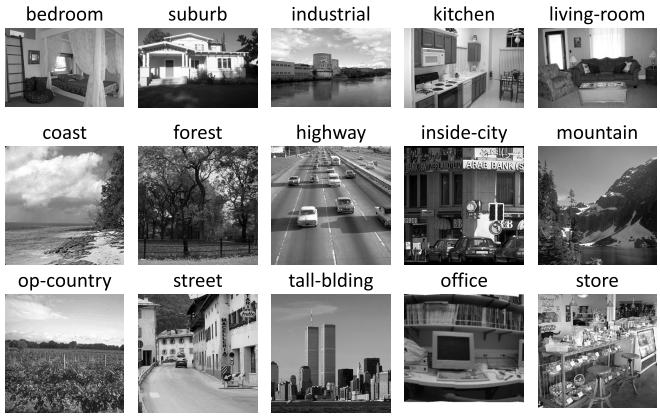

Fig. 22. Sample images from the Scene-15 dataset, one for each category.

TABLE XI
RESULTS ON THE SCENE-15 DATASET

# training	10	20	30	50	100
Lazebnik [15]	–	–	–	–	81.4
Li [49]	–	–	–	–	80.9
Gao [45]	–	–	–	–	83.68
Wang [26]	66.97	72.44	75.78	78.84	82.34
Ours	**70.67**	**76.12**	**78.74**	**81.72**	**85.13**
	±0.46	±0.73	±0.88	±0.48	±0.72

G. The MIT Indoor-67 Dataset

The MIT Indoor-67 dataset [50] contains 67 indoor scenes and 15620 images. It is a challenging dataset for indoor scene recognition. Sample images are listed in Fig. 23.

The spatial stride and window size for SIFT descriptors are 6 and 4. We do not use Edge-SIFT descriptors since the layouts of indoor scenes are often similar. We use 5, 10, 20, 40, and 80 images per category for training, and others for testing. Results are shown in Table XII.

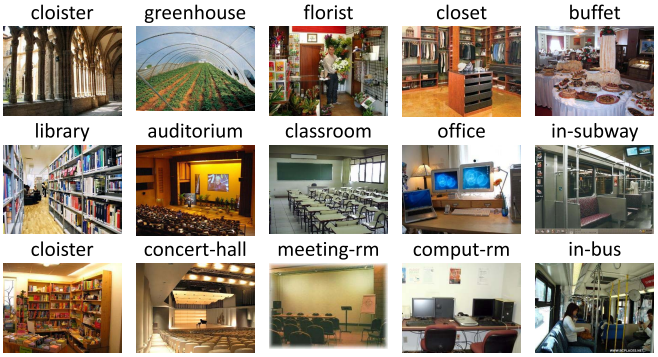

Fig. 23. Sample images from the MIT Indoor-67 dataset. Upper: categories on which accuracies are 80% or higher (80 trains). Middle and bottom: category pairs (in column) with very high semantic similarities. Images on the first row are correctly classified, while others are wrongly classified to the corresponding confusing classes.

TABLE XII
RESULTS ON THE MIT INDOOR-67 DATASET

# training	5	10	20	40	80
Quattoni [50]	–	–	–	–	26.0
Li [49]	–	–	–	–	37.6
Bo [51]	–	–	–	–	41.8
Wang [26]	19.31	25.61	31.34	37.01	43.10
Ours	**21.11**	**27.64**	**34.22**	**40.56**	**46.38**
	±0.49	±0.69	±0.45	±0.60	±0.75

H. Discussion

It is shown above that the proposed model with 3 new modules, *i.e.*, extracting complementary descriptors, Geometric Phrase Pooling (GPP), and edge-based spatial weighting, does produce superior performance over the traditional BoF model. Here, we provide some extra experimental results on the Caltech101 dataset to observe the effect of each module. We use the case with 30 training images per category, in which the overall accuracy gain is about 9%.

- **Extracting complementary descriptors** improves the classification accuracy by about 5% in this case. We shall emphasize that we can not only find compensation between texture (SIFT) and shape (Edge-SIFT) descriptors, but also in the case when we extract dense descriptors with different spatial strides and window scales. The latter technique is also widely used in the community [12], [26].
- **Geometric Phrase Pooling** improves the classification accuracy by about 4%. GPP provides an effective way of spatial context modeling on the local feature groups. With little extra time complexity, it produces significant improvement on all the classification tasks.
- **Edge-based spatial weighting** improves the classification accuracy by about 2%. We observe both accuracy gain and drop on different categories, but fortunately the averaged accuracy is boosted, showing the effectiveness of this simple trick.

In conclusion, all the proposed modules could be adopted individually to boost the classification performance. Even though the overall accuracy gain is less than the sum of individual improvements due to the marginal effect, we can verify that the proposed modules could co-operate with each other to produce a more powerful BoF model.

VII. Conclusion and Future Work

In this paper, we propose a novel framework based on the traditional BoF model for image classification. We add three new modules into the BoF model to enhance its description power. First, we extract SIFT and Edge-SIFT descriptors from the original image and the corresponding edgemap respectively, and then fuse them directly into a large set of descriptors. Experiments have revealed good compensation property of SIFT and Edge-SIFT descriptors. Moreover, fusing them at very early stage gives us more opportunities for spatial context modeling. Second, we propose a novel pooling strategy named Geometric Phrase Pooling (GPP). By extracting geometric visual phrases upon complementary visual words, it is possible to construct mid-level structures containing both texture and shape features, providing more robust intermediate representation. Third, based on boundary detection, we propose a simple and effective spatial weighting scheme to detect regions-of-interest. Integrating all the above modules coherently, we obtain a very powerful model that outperforms the state-of-the-art algorithms on various image classifications tasks.

Despite the excellent accuracy gain we have obtained, there are still some open problems in our framework. It is verified that objects are better described by complementary features such as texture and shape. However, calculating SIFT descriptors on the boundary images directly is not the best way for shape description. Although there exist a number of efficient shape descriptors such as shape context [52], it still remains to integrate both texture and shape features in an early fusion strategy. Geometric Phrase Pooling (GPP) is an efficient algorithm to capture spatial contexts, and it is reasonable to consider both scale and orientation of local descriptors for a more sophisticated spatial modeling. As to the naive spatial weighting scheme based on boundary detection, due to the lack of semantic consideration, it sometimes harms the classification accuracy (see Fig. 16). Saliency detection is an intuitive and instructive way of finding useful regions for image discrimination. We will investigate these problems in the future towards a more powerful classification model.

References

[1] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual Categorization with Bags of Keypoints," in *Proc. Workshop Statist. Learn. Comput. Vis., Eur. Con. Comput. Vis.*, 2004, pp. 1–22.

[2] M. Marszalek and C. Schmid, "Spatial weighting for bag-of-features," in *Proc. Comput. Vis. Pattern Recognit.*, 2006, pp. 2118–2125.

[3] E. Nowak, F. Jurie, and B. Triggs, "Sampling strategies for bag-of-features image classification," in *Proc. Comput. Vis. Pattern Recognit.*, 2006, pp. 490–503.

[4] Y. Cao, C. Wang, Z. Li, L. Zhang, and L. Zhang, "Spatial-bag-of-features," in *Proc. Comput. Vis. Pattern Recognit.*, 2010, pp. 3352–3359.

[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[6] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *Proc. Comput. Vis. Pattern Recognit.*, 2006, pp. 2161–2168.

[7] Y. Lu, L. Zhang, J. Liu, and Q. Tian, "Constructing lexica of high-level concepts with small semantic gap," *IEEE Trans. Multimedia*, vol. 12, no. 4, pp. 288–299, Jul. 2010.

[8] J. Yuan, Y. Wu, and M. Yang, "Discovery of collocation patterns: From visual words to visual phrases," in *Proc. Comput. Vis. Pattern Recognit.*, 2007, pp. 1–8.

[9] A. Bosch, A. Zisserman, and X. Muoz, "Image classification using random forests and ferns," in *Proc. 11th Comput. Vis. Pattern Recognit.*, 2007, pp. 1–8.

[10] Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce, "Learning mid-level features for recognition," in *Proc. Comput. Vis. Pattern Recognit.*, 2010, pp. 2559–2566.

[11] Y. Zhang, Z. Jia, and T. Chen, "Image retrieval with geometry-preserving visual phrases," in *Proc. Comput. Vis. Pattern Recognit.*, 2011, pp. 809–816.

[12] J. Feng, B. Ni, Q. Tian, and S. Yan, "Geometric ℓp-norm feature pooling for image classification," in *Proc. Comput. Vis. Pattern Recognit.*, 2011, pp. 2609–2704.

[13] M. Ruzon and C. Tomasi, "Color edge detection with the compass operator," in *Proc. Comput. Vis. Pattern Recognit.*, 1999, pp. 160–166.

[14] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, no. 6, pp. 679–698, Jun. 1986.

[15] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Proc. Comput. Vis. Pattern Recognit.*, 2006, pp. 2169–2178.

[16] L. Xie, Q. Tian, and B. Zhang, "Spatial pooling of heterogeneous features for image applications," in *Proc. 20th ACM Multimedia*, 2012, pp. 539–548.

[17] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. Comput. Vis. Pattern Recognit.*, 2005, pp. 886–893.

[18] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image Vis. Comput.*, vol. 22, no. 10, pp. 761–767, 2004.

[19] A. Bosch, A. Zisserman, and X. Munoz, "Scene classification via pLSA," in *Proc. Int. Conf. Comput. Vis.*, 2006, pp. 517–530.

[20] K. Van De Sande, T. Gevers, and C. Snoek, "Evaluating color descriptors for object and scene recognition," *IEEE Tran. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1582–1596, Sep. 2010.

[21] D. Li, L. Yang, X. Hua, and H. Zhang, "Large-scale robust visual codebook construction," in *Proc. ACM Multimedia*, 2010, pp. 1183–1186.

[22] Q. Tian, S. Zhang, W. Zhou, R. Ji, B. Ni, and N. Sebe, "Building descriptive and discriminative visual codebook for large-scale image applications," *Multimedia Tools Appl.*, vol. 51, no. 2, pp. 441–477, 2011.

[23] S. Zhang, Q. Huang, G. Hua, S. Jiang, W. Gao, and Q. Tian, "Building contextual visual vocabulary for large-scale image applications," in *Proc. Int. Conf. Multimedia*, 2010, pp. 501–510.

[24] H. Lee, A. Battle, R. Raina, and A. Ng, "Efficient sparse coding algorithms," in *Proc. Neural Inf. Process. Syst.*, 2007, pp. 801–805.

[25] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Proc. Comput. Vis. Pattern Recognit.*, 2009, pp. 1794–1801.

[26] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *Proc. Comput. Vis. Pattern Recognit.*, 2010, pp. 3360–3367.

[27] W. Zhou, Y. Lu, H. Li, Y. Song, and Q. Tian, "Spatial coding for large scale partial-duplicate web image search," in *Proc. Int. Conf. Multimedia*, 2010, pp. 511–520.

[28] L. Xie, J. Wang, B. Zhang, and Q. Tian, "Orientational pyramid matching for recognizing indoor scenes," in *Proc. Comput. Vis. Pattern Recognit.*, 2014, pp. 1–4.

[29] 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," *Comput. Vis. Image Understand.*, vol. 106, no. 1, pp. 59–70, 2007.

[30] D. Liu, G. Hua, P. Viola, and T. Chen, "Integrated feature selection and higher-order spatial feature extraction for object categorization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.

[31] S. Zhang, Q. Tian, G. Hua, Q. Huang, and S. Li, "Descriptive visual words and visual phrases for image applications," in *Proc. 17th ACM Int. Conf. Multimedia*, 2009, pp. 75–84.

[32] J. Yuan, M. Yang, and Y. Wu, "Mining discriminative co-occurrence patterns for visual recognition," in *Proc. Comput. Vis. Pattern Recognit.*, Jun. 2011, pp. 2777–2784.

[33] L. Xie, Q. Tian, and B. Zhang, "Feature normalization for part-based image classification," in *Proc. Int. Conf. Image Process.*, 2013, pp. 1–3.
[34] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *J. Mach. Learn. Res.*, 2008, pp. 1817–1874.
[35] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 143–156.
[36] M. Nilsback and A. Zisserman, "A visual vocabulary for flower classification," in *Proc. Comput. Vis. Pattern Recognit.*, 2006, pp. 1447–1454.
[37] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-UCSD birds-200-2011 dataset," California Inst. Technol., Pasadena, CA, USA, Tech. Rep. CNS-TR-2011-001, 2011.
[38] L. Xie, Q. Tian, R. Hong, S. Yan, and B. Zhang, "Hierarchical part matching for fine-grained visual categorization," in *Proc. Int. Conf. Comput. Vis.*, 2013, pp. 1–8.
[39] Y. Chai, V. Lempitsky, and A. Zisserman, "Symbiotic segmentation and part localization for fine-grained categorization," in *Proc. Int. Conf. Comput. Vis.*, 2013, pp. 1–10.
[40] E. Gavves, B. Fernando, C. Snoek, A. Smeulders, and T. Tuytelaars, "Fine-grained categorization by alignments," in *Proc. Int. Conf. Comput. Vis.*, 2013, pp. 1–10.
[41] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *Pattern Anal. Mach. Intell.*, vol. 20, no. 11, pp. 1254–1259, 1998.
[42] A. Vedaldi and B. Fulkerson, "VLFeat: An open and portable library of computer vision algorithms," in *Proc. Multimedia*, 2010, pp. 1469–1472.
[43] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes challenge 2007 (VOC2007) results," 2011.
[44] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," California Inst. Technol., Pasadena, CA, USA, Tech. Rep. CNS-TR-2007-001, 2007.
[45] S. Gao, I. Tsang, and L. Chia, "Kernel sparse representation for image classification and face recognition," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 1–14.
[46] S. Lazebnik, C. Schmid, and J. Ponce, "Semi-local affine parts for object recognition," in *Proc. Brit. Mach. Vis. Conf.*, 2004, pp. 959–968.
[47] D. Larlus and F. Jurie, "Latent mixture vocabularies for object categorization and segmentation," *Image Vis. Comput.*, vol. 27, no. 5, pp. 523–534, 2009.
[48] P. Gehler and S. Nowozin, "On feature combination for multiclass object classification," in *Proc. 12th Int. Conf. Comput. Vis.*, 2009, pp. 221–228.
[49] L. Li, H. Su, E. Xing, and L. Fei-Fei, "Object bank: A high-level image representation for scene classification and semantic feature sparsification," *Neuron Inf. Process. Syst.*, 2010, pp. 1–3.
[50] A. Quattoni and A. Torralba, "Recognizing indoor scenes," in *Proc. Comput. Vis. Pattern Recognit.*, 2009, pp. 413–420.
[51] L. Bo, X. Ren, and D. Fox, "Hierarchical matching pursuit for image classification: Architecture and fast algorithms," *Neural Inf. Process. Syst.*, vol. 1, no. 2, pp. 1–6, 2011.
[52] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 4, pp. 509–522, Apr. 2002.

Lingxi Xie received the B.E. degree from the Department of Computer Science and Technology, Tsinghua University, China, in 2010, where he is currently pursuing the Ph.D. degree. He was a Research Intern with Microsoft Research Asia from 2013 to 2014. His research interests include computer vision, multimedia information retrieval, and machine learning.

Qi Tian (M'96–SM'03) received the B.E. degree in electronic engineering from Tsinghua University, China, in 1992, the M.S. degree in electrical and computer engineering from Drexel University in 1996, and the Ph.D. degree in electrical and computer engineering from the University of Illinois, Urbana-Champaign, in 2002. He is currently a Professor with the Department of Computer Science, University of Texas at San Antonio. He took a one-year Faculty leave with Microsoft Research Asia from 2008 to 2009. His research interests include multimedia information retrieval and computer vision.

Meng Wang is a Professor with the Hefei University of Technology, China. He received the B.E. and Ph.D. degrees in the Special Class for the Gifted Young from the Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei, China, respectively. His current research interests include multimedia content analysis, search, mining, recommendation, and large-scale computing.

Bo Zhang received the B.E. degree from the Department of Automatic Control, Tsinghua University, China, in 1958. From 1980 to 1982, he visited the University of Illinois at Urbana-Champaign, USA, as a Scholar. He is currently a Professor with the Department of Computer Science and Technology, Tsinghua University. His research interests include artificial intelligence, machine learning, pattern recognition, knowledge engineering, intelligent robotics, and intelligent control. He was a recipient of the Prize of European Artificial Intelligence in 1984, the 1st-Class of Science and Technology Progress Prize three times in 1987, 1993, and 1998, and the 3rd-Class Prize two times in 1995 and 1999. He is a member of the Chinese Academy of Sciences.