Bag of Contour Fragments for Robust Shape Classification

Xinggang Wang[a], Bin Feng[a], Xiang Bai[a], Wenyu Liu[a], Longin Jan Latecki[b]

[a]*Dept. of Electronics and Information Engineering, Huazhong University of Science and Technology, 1037 Luoyu Road, Wuhan, Hubei Province 430074, P.R. China.*
[b]*CIS Dept., Temple University, 1805 N. Broad St. Philadelphia, PA 19122, USA.*

Abstract

Shape representation is a fundamental problem in computer vision. Current approaches to shape representation mainly focus on designing low-level shape descriptors which are robust to rotation, scaling and deformation of shapes. In this paper, we focus on mid-level modeling of shape representation. We develop a new shape representation called Bag of Contour Fragments (BCF) inspired by classical Bag of Words (BoW) model. In BCF, a shape is decomposed into contour fragments each of which is then individually described using a shape descriptor, e.g., the Shape Context descriptor, and encoded into a shape code. Finally, a compact shape representation is built by pooling shape codes in the shape. Shape classification with BCF only requires an efficient linear SVM classifier. In our experiments, we fully study the characteristics of BCF, show that BCF achieves the state-of-the-art performance on several well-known shape benchmarks, and can be applied to real image classification problem.

Keywords: Shape Classification, Shape Representation, Bag of Contour Fragments

1. Introduction

Shape is an intrinsic feature for image understanding, which is stable to illumination and variations in object color and texture. Because of these advantages, shape is widely

Email addresses: xgwang@hust.edu.cn (Xinggang Wang), fengbin@hust.edu.cn (Bin Feng), xbai@hust.edu.cn (Xiang Bai), liuwy@hust.edu.cn (Wenyu Liu), latecki@temple.edu (Longin Jan Latecki)

considered for object recognition. In particular, with the recent advance in contour detection proposed by Arbelaez et al. in [1], shape based object recognition in natural image is becoming more practical and attracts more attention in computer vision community. Main challenges in shape based object recognition include deformation, occlusion and viewpoint variation of objects. Various shape descriptors have been proposed to address these challenges, eg., [2, 3, 4, 5]. Shape based object recognition is usually considered as a classification problem. Given a set of training shapes and category label of each training shape, we need to determine which category a testing shape belongs to. Traditional shape classification methods are usually based on matching shape descriptors from two different shapes: for every training shape, we find correspondences between its shape descriptors and the shape descriptors in the testing shape using matching algorithms, such as Hungarian algorithm, dynamic programming algorithm; then we compute matching costs according to the matching results; finally, we rank training shapes based on the matching costs and classify the testing shape using the nearest neighbor (NN) classifier. This exemplar-based shape classification strategy has been widely used, for example, in [2, 3, 6]. However, it has its own limitations. With few training samples, it is difficult to capture the large intra-class variation using these algorithms. For large training samples, it is extremely time consuming to perform shape matching one-by-one.

Different from exemplar-based shape matching, in this paper, we propose a compact shape representation and handle the large intra-class variation by discriminative learning. Inspired by the huge progress in image classification and representation with Bag-of-Words (BoW) [7, 8], we decompose shape into contour fragments and quantize the contour fragments into shape codes. The contour fragments under different scales contain both local and global shape information which can be encoded utilizing coding strategies for local descriptors [9, 10]. Then, a statistical histogram of shape codes is used to represent each shape and similarity of shapes can be directly computed from these histograms. Matching shapes based on this new shape representation does not explicitly give correspondences between contour fragments. But using a classifier for shape classification is much more efficient than using the typical matching algorithms such as Hungarian, thin plate spline (TPS), dynamic

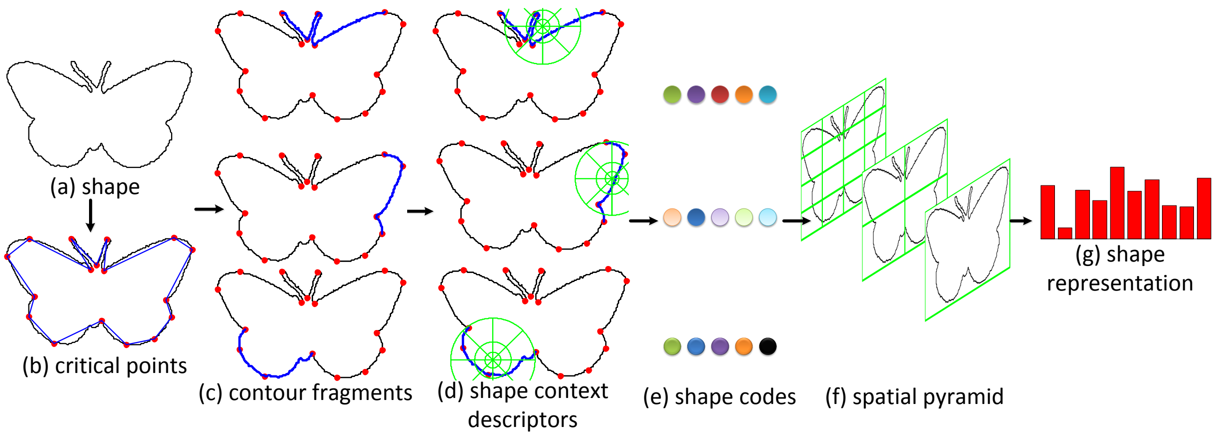


Figure 1: Pipeline of building shape representation using BCF. (a) shows contour of a shape; (b) shows critical points detected using DCE method; (c) shows some contour fragments in blue color; (d) shows that we use shape context [2] to describe each contour fragment; (e) shows shape codes; (f) shows we use 1×1, 2×2, and 4×4 spatial pyramid for max-pooling; (g) shows the histogram for shape representation.

programming, dynamic time warping, etc. In fact, BoW model is a natural solution for finding correspondences between two sets of features and can be used efficiently for recognition tasks. However, it has seldom been successfully applied to shape analysis, since the popular image descriptors such as SIFT [11] and LBP [12] are mainly designed for describing the local texture/appearance variations. These image features are not good at capturing the intrinsic structure in shape. Toward this end, we directly work on shape contour by decomposing it into contour fragments. We name our method Bag of Contour Fragments (BCF), which can not only provide a compact and informative representation, but also achieve state-of-the-art classification performance on several popular shape benchmarks.

Pipeline of building shape representation in BCF is shown in Fig. 1. The outer contour of each shape is decomposed into salient contour fragments using a well-known contour decomposition method named discrete contour evolution (DCE) [13]. Each contour fragment is then described by collecting the shape context features [2] on its reference points, and encoded into shape codes. Finaly, the shape codes are pooled into a compact image representation with spatial pyramid. We utilize the current advances in image classification, such as local-constrained linear coding (LLC) [9] for feature coding and spatial pyramid matching

(SPM) [14] in our BCF shape classification framework. Both LLC and SPM are seldom used in shape analysis. LLC utilizes the locality constraints and encodes each descriptor with its local-coordinate system in a codebook. In practice, it first performs k-nearest-neighbor search to find local-coordinates for feature to be encoded, and then solves a constrained least square fitting problem on the local-coordinates. The state-of-the-art performance on PASCAL VOC [15] image classification has shown effectiveness of LLC. SPM is a simple and computationally efficient extension of the orderless BoW model for image representation. It works by partitioning the image into increasingly fine sub-regions and computing histograms of local features found inside each sub-region. Histograms of different sub-regions are concatenated as final image representation. SPM can capture the spatial information in contour fragments which are useful for shape recognition. BCF naturally utilizes LLC and SPM to improve the accuracy of shape classification.

One of the major difficulties involved in shape classification for many shape-matching based algorithms is to directly match two shapes with large deformation since shapes are only partially similar to each other. BCF can easily solve this problem caused by large shape deformation, and is good at classifying shapes with partial similarity. As each shape contour is divided into contour fragments in BCF, the contour fragments contain partial shape information. After coding, a discriminative classifier such as SVM or Adaboost can be used to select the representative and informative contour parts for each shape category. Fig 4 shows some contour fragments selected by linear SVM in four shape categories in our experiments. We can see that even though contour fragments are parts of the shapes, they are very informative for recognizing shape category. Thus, BCF is able to deal with partial occlusion in shape, especially, in the edge map extracted from real image. Besides, we find that BCF is also robust to noisy contour in our experiments.

In summary, the proposed BCF has several good properties:

1. It provides a very compact shape representation which is a single vector rather than a set of feature vectors used in many other methods.
2. It precisely preserves information of individual shape contour via LLC and spatial

layout of contour fragments in one shape via SPM.

3. For shape classification it avoids pairwise matching between local shape descriptors and significantly reduces the time cost.
4. It is robust to the shapes with occlusions or parts missing, and can be easily applied to real image classification.

The rest of the paper is organized as follows. We review the related works in Section 2. Then, we introduce the details of our shape representation with BCF in Section 3, including extracting, encoding and pooling contour fragments etc. We evaluate the proposed method on several popular shape benchmarks, illustrate good properties of BCF in applications, and demonstrate its effectiveness in shape classification in various of datasets in Section 4. Finally, we conclude this paper in Section 5.

2. Related Work

Here, we briefly review the recent progress in shape classification. Sun and Super [16] proposed a shape classification framework for recognizing contour shapes using class contour segments as input features with Bayesian classifier. Bai et al. [6] adopted contour segments and skeleton paths as the input features for shape classification with a Gaussian mixture model. Daliri and Torre [17, 18]transformed the contour points into a symbol representation, and then used the edit distance between pair of strings are used for classification with a kernel support vector machine. Wang et al. [19] proposed a tree-union [20] representation as the prototype for each shape category, and performed shape classification is determined by the shape similarity between a test shape and each prototype. Edem and Tari [21] also used a skeletal tree model to represent the prototype of each category, and then used the edit distance between a given shape and each prototype is used as the input feature for a linear SVM. Thus, each prototype in [21] can be considered as a shape codebook. Shape classification by skeleton matching has been studied by [22, 23, 24, 25].

Various shape descriptors have been proposed for shape matching and recognition. There are some region-based methods, such as Zernike moments [26] and generic Fourier descriptor

[27]. Other methods based on contour include curvature scale space (CSS) [4], multi-scale convexity concavity (MCC) [28], triangle area representation (TAR) [5], hierarchical procrustes matching (HPM) [29], shape-tree [30], contour flexibility [31], shape context (SC) [2], inner-distance shape context (IDSC) [3] and so on. In this paper, we only use shape context to describe contour fragments in BCF. Generally speaking, most of these shape descriptors can be adopted as low-level representation in BCF, since each contour fragment can be considered as a shape. We use discrete contour evolution (DCE) [13] for decomposing shape into contour fragments. Other recent shape evolution methods, e.g., [32], can also be adopted in BCF.

Our BCF approach can be considered as a two-layer feature learning framework on shape contours. In the first layer, contour fragment features are encoded into shape codes using local-constrained linear coding (LLC), which is first proposed in [9] for encoding SIFT [11] features in real images. Other feature coding methods include fisher kernel (FK) [10] and kernel codebook encoding (KCB) [33]; we choose LLC for its high efficiency. In the second layer, we use spatial pyramid matching (SPM) for pooling the shape codes. Theory of SPM is given in [34]. SPM is first proposed by Lazebnik et al. for image classification in [14]. Recently deep learning is very popular for feature learning and obtains good results on large-scale image classification [35]. Different from BCF, deep learning approaches have more layers; in [36], for example, a shape model based on deep Boltzmann machine called shape Boltzmann machine (SBM) is proposed; SBM directly works on raw shape pixels and learns probability distributions over object shapes, which is good at shape completion task but hardly works on the challenging Mpeg-7 shape dataset [37] for shape classification. However, BCF learns shape representation over shape contours, which is more robust to deformation of shape and more suitable for shape recognition. Textures feature and BoW model are directly used for shape classification in [38]. BCF utilizes contour fragment as shape feature, which is obviously superior to [38].

The strategy of partitioning shape into contour parts for shape recognition has been adopted by [16, 39, 6]. Unlike these previous works where contour parts are put in an orderless set as shape representation, BCF explores the spatial layout of contour parts and builds a compact shape representation via feature coding and pooling.

3. Bag of Contour Fragments

In this section, given a shape S, we show how to build BCF shape representation $\mathbf{f}(S)$ for S and use $\mathbf{f}(S)$ for shape classification step by step.

3.1. Contour fragments

Contour fragments have been validated as powerful shape features in several previous approaches [16, 6], since they contain both local and global shape information. We adopt contour fragments as basic shape features for learning a shape codebook and building our shape representation. An object boundary can be decomposed into contour fragments in different ways, such as dense sampling and sampling based on curvature like in [16]. Here, we use a more robust technique named discrete contour evolution (DCE) [13] for partitioning the whole object contour into meaningful contour fragments. Let $S(t) = (x(t), y(t))$ be the outer contour of a shape S parameterized by $t \in [0, 1]$. We first apply the DCE to obtain a simplified polygon on S with vertices denoted as

$$\overrightarrow{u} = (u_1, \ldots, u_T),$$

where T denotes number of vertices, which is not previously known but can be automatically computed given a threshold parameter τ. $\overrightarrow{u}$ includes **critical points** on S. Fig. 1(b) shows the critical points extracted by DCE for an input contour S.

Given an object contour S, its **contour fragments** set is denoted by $\mathcal{C}(S)$, which are the segments between every pair of critical points (u_i, u_j). Let c_{ij} denote the contour fragment between u_i and u_j, we have

$$\mathcal{C}(S) = \{c_{ij} = (u_i, u_j), i \neq j, i, j \in [1, \ldots, T]\}. \tag{1}$$

Note that u_i and u_j do not have to be adjacent to each other. Also,

$$S = c_{ij} \cup c_{ji}, \tag{2}$$

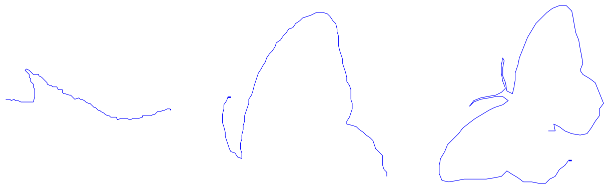


Figure 2: Three exemplar contour fragments containing short-range (left), middle-range (middle) and long-range (right) information in shape.

since one represents a fragment and the other is its counterpart. $\mathcal{C}(S)$ contains very rich information in shape S, since contour fragments between all pairs of critical points are extracted. All the contour fragments extracted from a shape contain multi-scale information, which can be summarized as short-range, middle-range and long-range information as shown in Fig. 2. Therefore, contour fragments are totally different from local descriptors (SIFT, HOG, or LBP, etc.) for image classification, since the local descriptors only contain information of local patches in image. In the rest of this section, we will show how we describe contour fragments and how we select informative contour fragments for shape recognition.

For each contour fragment c_{ij}, we describe it using shape context $\mathbf{x}_{ij} \in \mathbf{R}^{d\times 1}$ where d is the dimension of the feature vector of c_{ij}. As illustrated in Fig. 3, $\mathbf{x}_{ij}$ is computed as follows: we sample 5 reference points on c_{ij} from u_i to u_j equidistantly, and then compute 5 shape context histograms based on the reference points individually. Shape context descriptor for c_{ij} is a concatenation of the 5 shape context histograms.

3.2. Encoding of contour fragments

Encoding contour fragment features $\mathbf{x}_{ij}$ is to map feature vectors of contour fragments into a new space $\mathcal{B}$ spanned by a shape codebook $\mathbf{B}$; in this new space, contour fragments are represented by shape codes $\mathbf{w}_{ij}$.

Many codebook learning methods have been proposed for image representation, including unsupervised methods [9] and supervised method [40, 41] etc. In this paper, we choose k-means [42] as the codebook learning since it is a simple yet stable one. A set of training shape features is randomly selected from all the contour fragment features. We then run k-means algorithm on the selected shape features for clustering. The clustering centers are

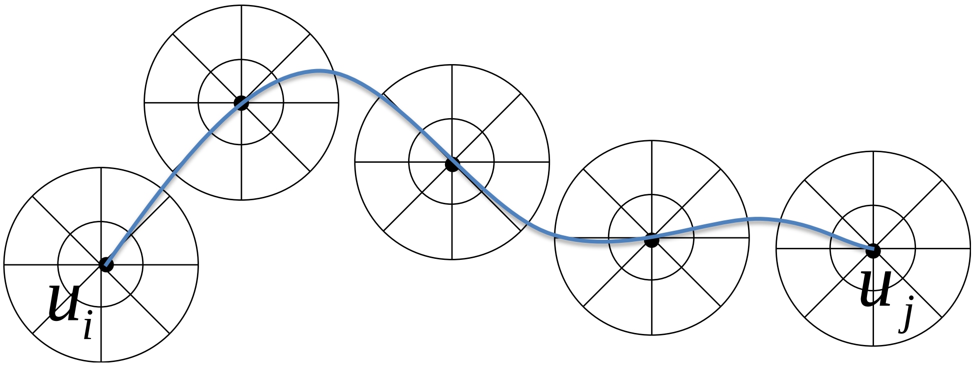


Figure 3: Shape context feature for contour fragment c_{ij}. Note that circles in this figure are plotted for showing the positions to compute shape context which do not stand for the area in which to compute shape context.

used as shape codebook $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_M] \in \mathbf{R}^{d \times M}$, where each column is a clustering center. So the obtained M clustering centers can be approximately considered as M prototypes for describing the whole shape space.

To compute shape codes $\mathbf{x}_{ij}$, a traditional way is to do vector quantization (VQ) like in [14]. VQ only assigns a shape feature $\mathbf{x}_{ij}$ to its nearest neighbor in shape codebook $\mathbf{B}$; it is fast but its quantization error is large. Local-constraint linear coding (LLC) is a very good choice for feature coding proposed in [9], as it is both fast and effective. The LLC method is inspired by the theory of local linear embedding (LLE) [43]. To represent $\mathbf{x}_{ij}$ in the space $\mathcal{B}$ spanned by shape codebook $\mathbf{B}$, LLC uses k nearest neighbours in $\mathbf{B}$ as local bases for $\mathbf{x}_{ij}$ to form a local coordinate system. The k nearest neighbors of $\mathbf{x}_{ij}$ are denoted as $\mathbf{B}_{\pi_{ij}} \in \mathbf{R}^{d \times k}$ where π_{ij} is a set containing the indexes of the k nearest neighbors in $\mathbf{B}$, denoted as $\pi_{ij} = \{\pi_{ij}^1 \dots \pi_{ij}^k\}$. $\mathbf{B}_{\pi_{ij}}$ is a matrix consisting of the $\pi_{ij}^1 \dots \pi_{ij}^k$-th columns of $\mathbf{B}$. Following the assumption in LLE, we expect that $\mathbf{x}_{ij}$ and its nearest neighbors lie on or close to a local linear patch of the manifold. The local geometry of $\mathbf{x}_{ij}$ and $\mathbf{B}_{\pi_{ij}}$ can be characterized by linear coefficients obtained through reconstructing $\mathbf{x}_{ij}$ from $\mathbf{B}_{\pi_{ij}}$. The coefficients $\mathbf{w}_{\pi_{ij}} \in \mathbf{R}^{k \times 1}$ can be obtained by solving the following minimization problem

$$\min_{\mathbf{w}_{\pi_{ij}}} \|\mathbf{x}_{ij} - \mathbf{B}_{\pi_{ij}} \mathbf{w}_{\pi_{ij}}\|^2 \qquad \text{s.t.} \quad \mathbf{1}^T \mathbf{w}_{\pi_{ij}} = 1, \tag{3}$$

where weight vector $\mathbf{w}_{\pi_{ij}}$ summarizes the contributions of local bases to $\mathbf{x}_{ij}$'s reconstruction, which is required to be summed to 1. The minimization problem in (3) is a small-scale least square problem, and its time complexity is $\mathcal{O}(k^2)$. In our experiments, we always set the value of k as 5. We denote code of $\mathbf{x}_{ij}$ as $\mathbf{w}_{ij} \in \mathbf{R}^{d\times 1}$: values of the $\pi_{ij}^1 \dots \pi_{ij}^k$-th entries of $\mathbf{w}_{ij}$ are equal to $\mathbf{w}_{\pi_{ij}}$ and the rest of entries in $\mathbf{w}_{ij}$ are set to zero.

3.3. Max-pooling with spatial pyramid

In this subsection, we build a compact shape representation based on statistics of shape codes $\mathbf{w}_{ij}$. In addition, we utilize spatial pyramid matching (SPM) [14] method to add spatial layout information of contour fragments into our shape representation.

The process of building shape representation is given as follows: First, we divide shape into different regions. Specifically, shape is divided into 1×1, 2×2 and 4×4 regions, as shown in Fig. 1(f); in total, there are 21 regions. Then for each region $Region_r, r \in [1, \dots, 21]$, we do max-pooling. Let $\mathbf{w}^z$ denotes an encoded contour fragment in the position of z in shape (position of a contour fragment is defined as its the median point). Max-pooling works as follows:

$$\mathbf{f}(S, r) = \max(\mathbf{w}^z | z \in Region_r), \tag{4}$$

where the max function works in row-wise, returns a feature vector of $Region_r$, $\mathbf{f}(S, r)$, with the same size as $\mathbf{w}_{ij}$. For each codeword, we take the max value of all shape codes in a region for shape representation, so we called this method as **max-pooling**. Max-pooling procedure is well established by biophysical evidence in visual cortex (V1) [44]. Its correctness empirically verified by many algorithms applied into image classification, such as [9, 45, 10] etc. It also works well with linear classifiers. Final representation $\mathbf{f}(S)$ for shape S is a concatenation of the feature vectors for all regions.

$$\mathbf{f}(S) = [\mathbf{f}(S, 1)^T, \dots, \mathbf{f}(S, 21)^T]^T. \tag{5}$$

It is easy to know the dimension of $\mathbf{f}(S)$ is $21 \times M$.

SPM can encode spatial information among the short-range contour fragments in a coarse-to-fine way. We train a classifier on training shape to decide whether the classifier fires on a coarse level (1×1 region) or a fine level (2×2 and 4×4 regions). More specifically, if the training shapes are well aligned, it contains similar contour fragments in each small grid, so the classifier will fire on a fine level. On the other hand, if the training shapes are rotated in different directions, it contains different contour fragments in a single small grid; but all contour fragments are contained in the coarse level; so the classifier will fire on a coarse level. Thus, SPM is a very flexible strategy.

3.4. Shape classification using linear SVM

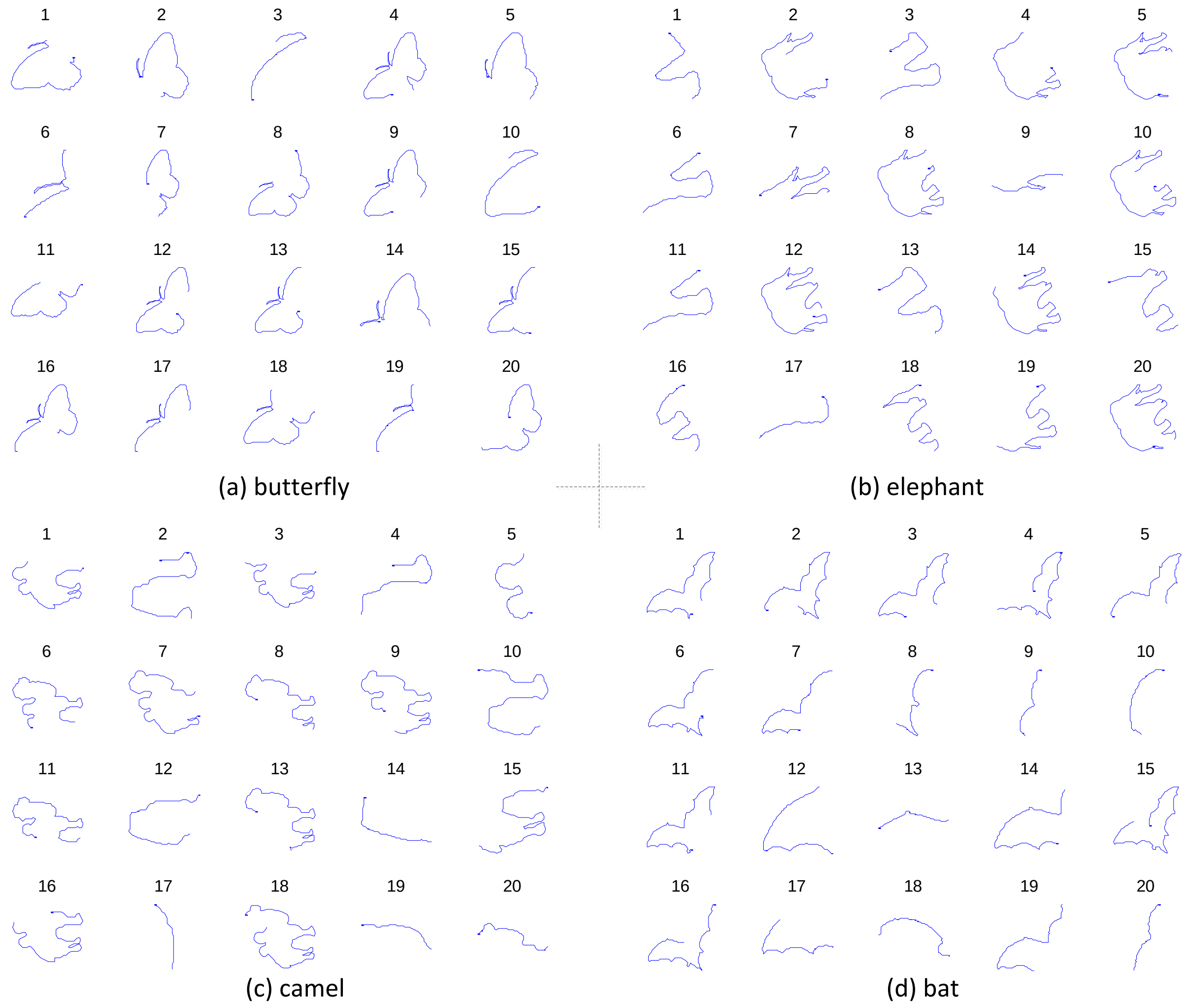
Since our shape representation is a simple vector, we directly adopt SVM for shape classification. For multi-class SVM, we use the formulation proposed by Crammer and Singer in [46]. Given a set of training shapes $\{\mathbf{f}_i\}$ with labels $\{y_i \in [1, \ldots, N]\}$ where N is the number of shape classes. Crammer and Singer's multi-class SVM can be used to solve the following optimization problem:

$$\min_{\omega_1,\ldots,\omega_N} \quad \sum_{n=1}^{N} \parallel \omega_n \parallel^2 + \lambda \sum_i \max(0, 1 + \omega_{r_i}^T \mathbf{f}_i - \omega_{y_i}^T \mathbf{f}_i), \tag{6}$$

where $r_i = \arg\max_{n\in[1,\ldots,N],n\neq y_i} \omega_n^T \mathbf{f}_i$. In Eq. (6), the left part is a regularization term; the right part is multi-class hinge-loss; parameter λ controls the relative weight of the regularization term. To solve (6), we use the off-shelf SVM solver, LibLinear developed by Lin et al. [47]. In the testing stage, shape label is predicted by

$$\widehat{y} = \arg\max_{n\in[1,\ldots,N]} \omega_n^T \mathbf{f}. \tag{7}$$

Learning with SVM is a process of selecting support vectors, during which certain contour fragments important for recognition are selected in every shape. In Fig. 4, we show some examples. For a shape, we find top 20 values in all $\mathbf{f}_i \cdot \omega_{y_i}$. We then find the contour fragments that contribute to the top 20 entries in $\mathbf{f}_i$; in other words, we find the contour fragments which have maximal code value in the top 20 entries in $\mathbf{f}_i$. As shown in Fig. 4, the selected contour fragments by coding (with LLC), max-pooling and SVM are meaningful. There are a few trivial fragments in top 20, such as, the 17th and 19th in (c), because the values of their codes are large as it is easier for them to be precisely encoded. But their corresponding value in ω_{y_i} is small.



(a) butterfly

(b) elephant

(c) camel

(d) bat

Figure 4: Top 20 contour fragments which contribute the most for recognition in a butterfly shape (a), an elephant shape (b), a camel shape (c), and a bat shape (d).

Time consuming kernels, such as the RBF kernel and intersection kernel etc., can further improve the shape classification performance. But for faster speed, we use linear SVM.

4. Experiments

In this section we test our method for shape classification on a variety of shape datasets and compare results of our method with the state-of-the-art shape classification approaches, available for those datasets in the literature. We also study the robustness of our method. First of all, we give implementation details as following. [1]

4.1. Implementation details

Extracting contour fragments. We use DCE to extract about 400 contour fragments per shape; max curvature τ of DCE is set to 0.5. When computing shape context for contour fragment, we have 5 reference points given in Section 3.1, and set number of bins of shape context to be 60 (10 for dividing angle space and 6 for dividing radius space). Thus, dimension of our shape context descriptor for a contour fragment is 300. Besides, positions of contour fragments in shape are also recorded which are used for pooling with spatial pyramid.

Learning shape codebook. Standard k-means clustering is adopted for training the codebook. The number of the contour segments collected from the dataset could be enormous. As a result, the codebook training can be very time consuming and computationally expensive. Therefore, we randomly select 1000 images and for each image only 300 shape context features are picked for training the codebook. The number of clustering centers is set to 1500 if it is not specified. In addition, we will study the performance of BCF with different number of clustering centers.

Coding, pooling and classification. In the coding scheme, the approximated LLC with 5 nearest neighbors is adopted. When pooling, a shape is divided into 1×1, 2×2, and 4×4,

[1] MATLAB code of these experiments is available at `https://bitbucket.org/xinggangw/bcf`

in total 21 regions. The final feature vector for shape representation is normalized by its ℓ_2 norm. For shape classification, a fast off-shelf linear SVM toolbox, LibLinear [48], is used.

Datesets. We evaluate our BCF method on shape classification benchmark dataset which is the MPEG-7 dataset [37], and use BCF for 70 classes animal classification on Animal dataset [6], for leaf classification on Swedish Leaf dataset [3], and for multi-view object classification on ETH-80 image dataset [49]. In the rest of this section, we give experimental results and analysis.

4.2. MPEG-7 dataset

The MPEG-7 dataset is widely used for shape analysis in the field of computer vision. It has 1400 silhouette images divided into 70 classes with high shape variability. Each class has 20 different shapes (see Fig. 5 for some typical images). We use two strategies for evaluating shape classification performance: (1) *half training*, we randomly select 10 shapes in each class for training and use the rest shapes for testing in each round; this procedure is repeated for 10 times; average classification accuracy and standard derivation of classification accuracies are reported; (2) *leave-one-out*, for each shape, we use all shapes except the current one for training and use the current one for testing; average classification accuracy is reported.

BCF is compared with other shape classification methods in Table 1. In [16, 6] contour fragments are used for shape classification. BCF outperforms [16, 6] by over 6% when using half of shapes for training. The superior performance may be attributed to the fact that the discriminative learning via SVM in our approach can maximize the margins between different shape categories and find very informative contour fragments for each category. In [16, 6], however, all contour fragments have equal weights. Preconditions of discriminative learning with SVM are that BCF provides a compact shape representation and LLC precisely preserves information of contour fragment. In [50, 17, 18], shape is described based on the symbolic representation. BCF achieves the state-of-the-art performance when using leave one out for testing which is the same as the result in the most recent work in [18].

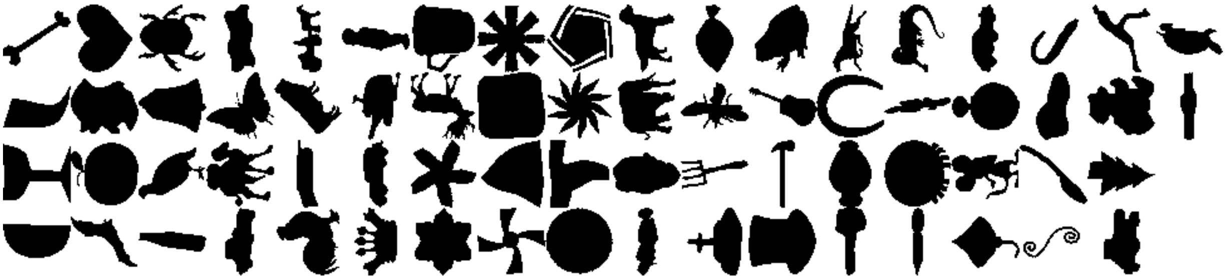


Figure 5: Typical shapes from the Mpeg-7 dataset [37]. One image from each class.

Table 1: Classification accuracy comparison on Mpeg-7 dataset [37]

	Classification accuracy	
Algorithm	Half training	Leave one out
Class segment set [16]	90.9%	97.93%
Contour segments [6]	91.1%	-
Skeleton paths [6]	86.7%	-
ICS [6]	96.6%	-
Polygonal multi-resolution [51]	-	97.57%
String of symbols [50]	-	97.36%
Robust symbolic [17]	-	98.57%
Kernel-edit distance [18]	-	**98.93**%
BCF	**97.16±0.79**%	**98.93**%

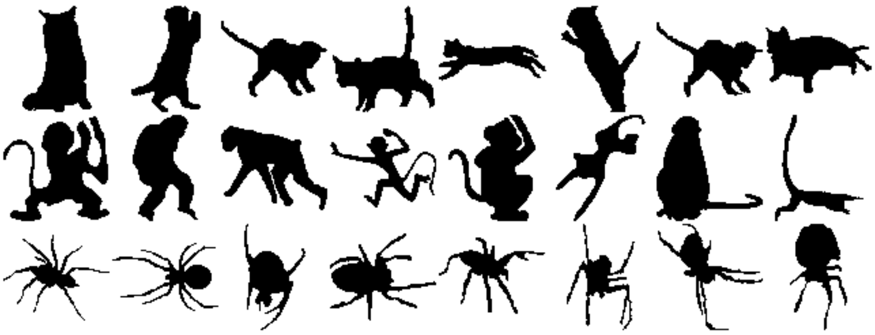


Figure 6: Some shapes form Animal dataset [6]. The first, second and last row shows 8 shapes from Cat (1st row), Monkey (2nd row) and Spider (3rd row) class from this dataset respectively.

Table 2: Classification accuracy comparison on Animal dataset [6]

Algorithm	Classification accuracy
Class segment set [16]	69.7%
IDSC [3]	73.6%
Bag of SIFT [38]	74.9%
Contour segments [6]	71.7%
Skeleton paths [6]	67.9%
ICS [6]	78.4%
BCF	**83.40±1.30%**

4.3. Animal dataset

The animal dataset was introduced in [6], it contains 2000 shapes describing 20 kinds of animals, including horse, rabbit, monkey, etc. Each category have 100 animals images. Some of shapes form the two most difficult classes (Cat and Monkey) and the easiest class (Spider) in this dataset are shown in Fig. 6. The dataset has much more intra-class variability since the same kind of animals may have various gestures. We use 50 shapes randomly selected per class for training and the rest of shapes for testing. We run experiments for 10 times and average classification accuracy of our method is compared with that of other methods in Table 2.

As shown in Table 2, the proposed BCF method obtains a classification accuracy of

Table 3: Detailed classification accuracy on Animal dataset [6]

Method	Bird	Butterfly	Cat	Cow	Crocodile	Deer	Dog	Dolphin	Duck	Elephant
CS[6]	76%	89%	39%	70%	54%	69%	69%	87%	83%	95%
ICS[6]	76%	**93%**	48%	**80%**	66%	79%	75%	**89%**	**89%**	**97%**
BCF	**87.6%**	92.2%	**73.8%**	77.4%	**76.8%**	**90.4%**	**82.6%**	**89.0%**	87.0%	95.2%
Method	Fish	Fly-bird	Hen	Horse	Leopard	Monkey	Rabbit	Rat	Spider	Tortoise
CS[6]	70%	57%	89%	96%	56%	21%	81%	52%	98%	81%
ICS[6]	74%	65%	94%	**97%**	65%	33%	**87%**	**84%**	**100%**	90%
BCF	**79.8%**	**72.0%**	**94.2%**	95.4%	**66.4%**	**58.4%**	85.8%	70.6%	99.2%	**93.6%**

83.40% which significantly outperforms the classical shape descriptor, inner distance shape context [3], and the previous state-of-the-art method [6] which integrates contour segments and skeleton paths for shape classification. Average classification accuracy for each of the 20 classes in Animal dataset are reported in Table 3. BCF dramatically improves classification accuracy in Cat and Monkey classes. This shows that BCF can capture the intra-class partial similarity within the highly deformed objects from Animal dataset. Bag of SIFT method [38] directly uses texture feature for shape classification, obtains a classification accuracy of 74.9% which is much lower than BCF's accuracy. This shows that our contour fragment feature is more suitable for shape classification than SIFT.

4.4. Swedish Leaf dataset

In this subsection, we use BCF for leaf image recognition on the Swedish Leaf Dataset [52]. The Swedish leaf dataset comes from a leaf classification project at Linköping University and Swedish Museum of Natural History. The dataset contains isolated leaves from 15 different Swedish tree species, with 75 leaves per species. Some typical binary shapes of leaf images are shown in Fig. 7. Note that some species are indistinguishable to the untrained eye, e.g. the 1st, 3rd, 9th, 11th and 15th species. We follow the experimental setting in [3]. In each species, 25 shapes are randomly selected for training and the rest of shapes are used for testing. We run training and testing for 10 times and report the average and standard deviation of the classification accuracies. We compare classification accuracy of our method with other pure shape-based recognition methods in Table 4. The methods

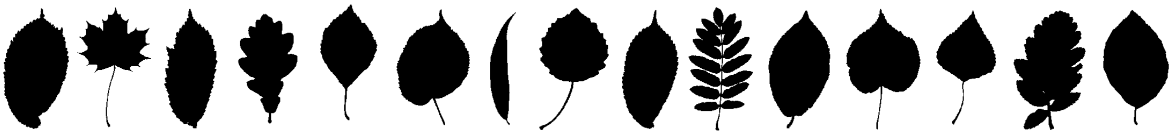


Figure 7: Typical shape of images from the Swedish leaf dataset [52]. One image from each species.

Table 4: Classification accuracy comparison on Swedish leaf dataset [52].

Algorithm	Classification accuracy
Moment+Area+Curvature [52]	82%
Fourier [3]	89.6%
SC+DP [3]	88.12%
IDSC+DP [3]	94.13%
MDM [53]	93.60%
IDSC+Morphological strategy [54]	94.80%
Robust symbolic [17]	95.47%
Shape-tree [30]	96.28%
BCF	**96.56±0.67**%

compared include a preliminary work [52] using some simple features like moments, area and curvature etc, the Fourier descriptor, the shape context with dynamic programming (SC+DP), the inner distance shape context with dynamic programming (IDSC+DP), the multi-scale matrix distance matrix [53], the morphological strategy method in [54], a robust symbolic representation method [17] and the shape-tree method in [30]. BCF obtains the state-of-the-art performance among these methods.

4.5. ETH-80 dateset

The ETH-80 datastet [49] contains 80 3-D high resolution objects (Fig. 8) from eight categories. For each object, there are 41 color images from different viewpoints. So the dataset contains 3280 images in total. Segmentation masks of all images are provided to evaluate shape-based object recognition approaches with this dataset. The test mode of this dataset is leave-one-object-out cross-validation [49]. Specifically, in each round images

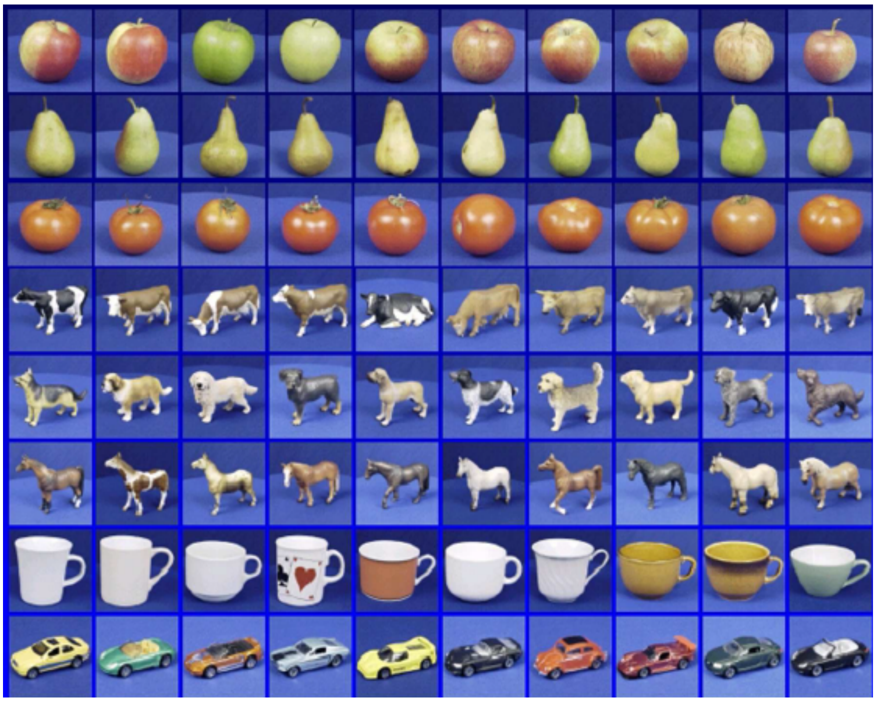


Figure 8: 80 3-D objects from ETH-80 image set. Each row shows one category.

from 79 objects are used for training and images from the remaining one object are used for testing. We compare the average classification accuracy of BCF to many other previous approaches in Table 5. BCF gets a classification accuracy of 91.49% which outperforms pervious state-of-the-art approach in [18].

4.6. Robustness to noise

In the above experiments, the shapes are quite smooth in these datasets. To evaluate the performance of our descriptor under noisy conditions, we add Gaussian noise to shape boundaries and carry out image classification using BCF. We use the whole Mpeg-7 dataset [37] as the original shape boundaries. Noise is added by perturbing all pixels on each shape contour in both x- and y-coordinates by values drawn from a Gaussian random variable with zero mean and standard derivation σ. As the parameter σ increases, we add increasing Gaussian noise to the shape boundaries. Fig. 9 shows an example of shape boundaries with increasing Gaussian noise. We report the classification accuracies using half for training and leaving one out for testing as noise ratio σ varying from 0 to 1 in Fig. 10. Classification accuracy

Table 5: Classification accuracy comparison on ETH-80 dataset [49].

Algorithm	Classification accuracy
Color histogram [49]	64.86%
PCA gray [49]	82.99%
PCA masks [49]	83.41%
SC+DP [49]	86.40%
IDSC+DP [3]	88.11%
IDSC+Morphological strategy [54]	88.04%
Height function [55]	88.72%
Robust symbolic [17]	90.28%
Kernel-edit [18]	91.33%
BCF	**91.49%**

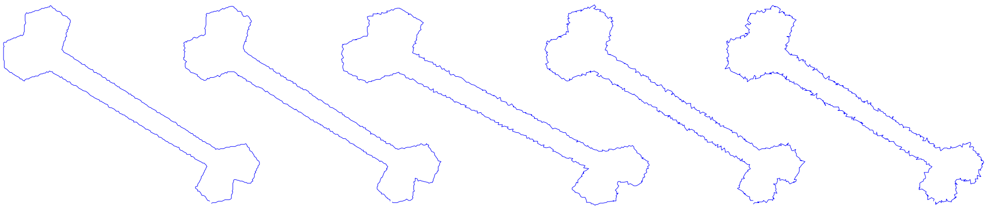


Figure 9: An example of shape boundaries with increasing Gaussian noise.

using half training drops about 4% when σ increases from 0 to 1, which shows that BCF is robust to noise. This is due to the fact that both DCE and the shape context are robust to noise.

4.7. The effect of codebook size

In this experiment, we do shape classification using shape codebooks of different sizes on the full Mpeg-7 dataset. Shape classification accuracies of BCF using codebooks of different sizes are reported in Fig. 11. Generally, shape classification accuracy improves as the size of codebook increases, but gets saturated when codebook size increases to 1500.

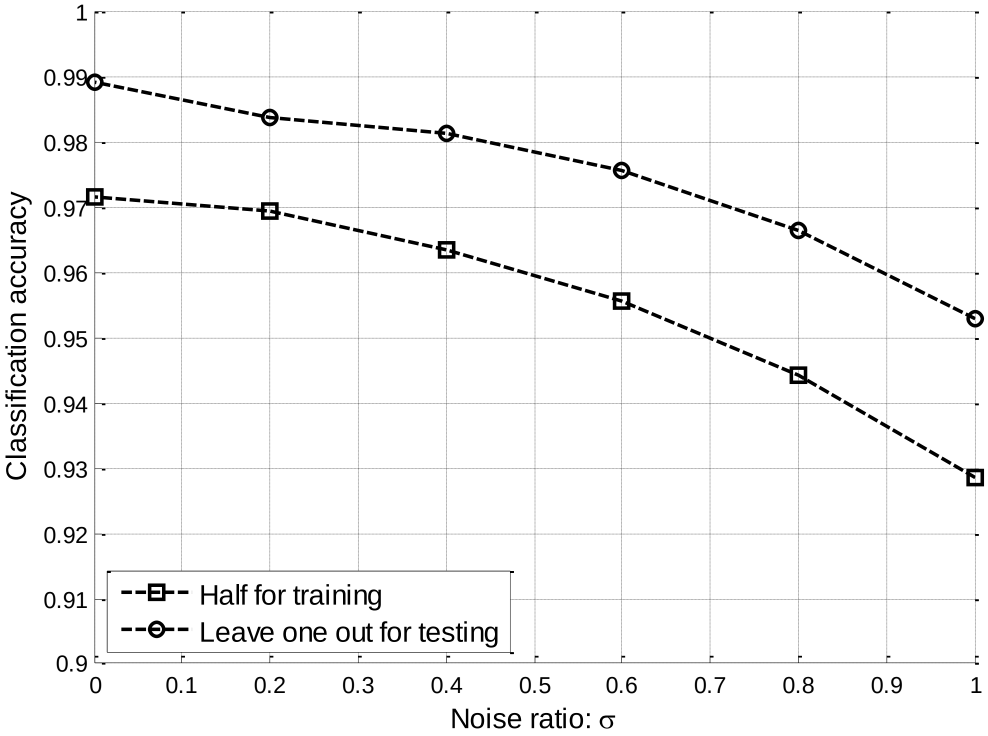


Figure 10: Classification accuracies on Mpeg-7 dataset are reported as noise ratio σ varying from 0 to 1 under both half for training setting and leave one out setting.

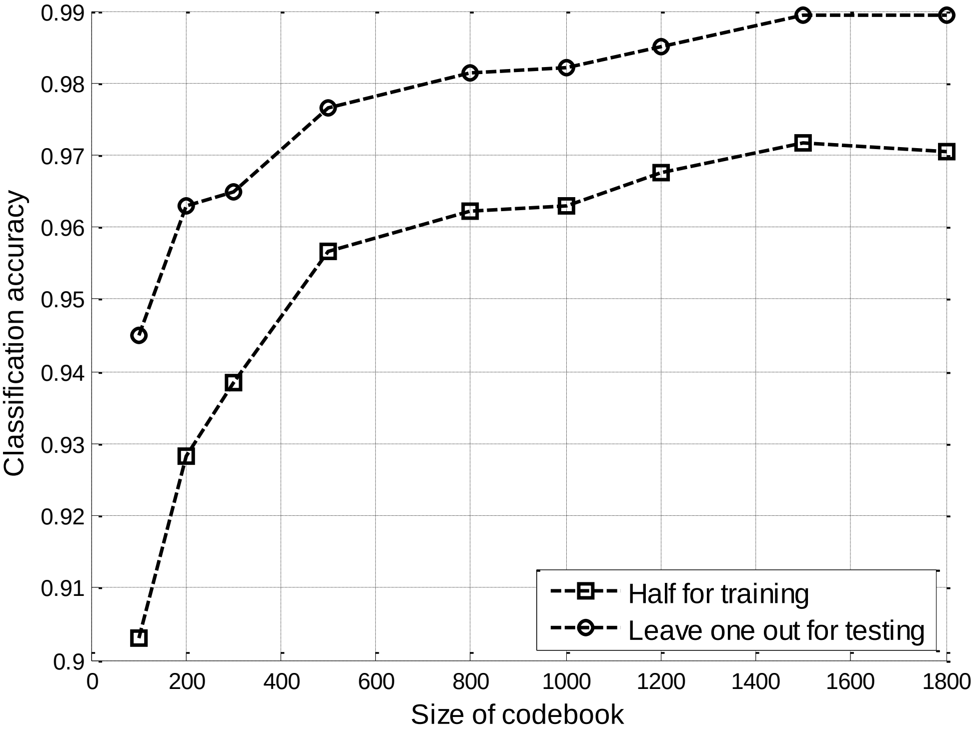


Figure 11: Classification accuracies of using half for training and leaving one out for testing are reported as size of codebook changing from 100 to 1800.

Table 6: Classification accuracy before and after shape codebook exchanging

	Mpeg-7 dataset	Animal dataset
original	97.16±0.79%	83.40±1.30%
codebook exchanging	95.55±0.55%	82.40±1.07%

4.8. Generalization ability of shape codebook

In this experiment, we investigate the generalization ability of the shape codebook learned by k-means. As the space of contour fragments of shapes is much smaller than the space of local features of natural images, e.g., SIFT and HOG, we investigate whether it is possible to learn a universal codebook of contour fragments for shape classification. Therefore, we use the codebook learned from Mepg-7 dataset for descriptor coding, building shape representation and performing shape classification on Animal dataset. We also use the codebook learned from Animal dataset for descriptor coding, building shape representation and performing shape classification on the Mepg-7 dataset. The sizes of both codebooks are 1500. Except the codebook, all other experimental settings are the same. We call this experiment "codebook exchanging". Shape classification results (half shapes for training for both datasets) are shown in Table 6. The results show that there is only about one percentage drops in classification accuracy after codebook exchanging on both datasets. These results show that the generalization ability of our shape codebook is very good. The reason why codebook exchanging can work is that different datasets share lots of common contour fragments. For example, the legs of a horse are very similar to the ones of a dog, and the leaf of an apple may be very similar to the wing of a bat. The success of codebook exchanging implies that we may use a universal shape codebook for all codebook-based shape recognition system.

4.9. Image classification on Caltech 101 dataset

The Caltech 101 dataset contains 9144 images in 101 object classes including animals, vehicles, flowers, etc, with significant variance in shape, color and texture, and a background class. The number of images per category varies from 31 to 800. We follow the common

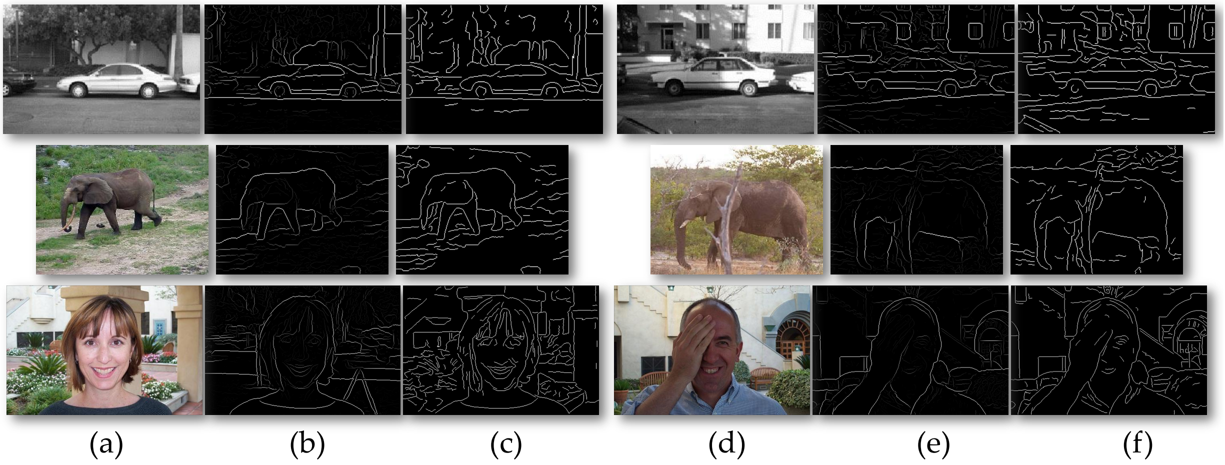


Figure 12: Example images in Caltech 101 dataset in (a) and (d) together with their gPB edge maps [1] in (b) and (e) and binary shapes obtained by post processing in (c) and (f).

experiment setup for Caltech 101, training on 30 images per class and testing on the rest, and measure the performance using average accuracy over the 102 classes.

The color/gray images in Caltech 101 dataset are different from the binary shapes we tested in the previous experiments. Now we show how to use the proposed BCF approach to build an image representation for a color/gray image. Given a color/gray image, we first compute its edge map using the gPB algorithm in [1] (some of the edge maps are shown in Figure 12(b) and (e)), and set all the pixels on the edge map with their values larger than 0.1*255 as edge pixels. Then, the edge-linking algorithm in [56] is applied on the binary edge image to retrieve a set of contours shown in Figure 12(c) and (f). Finally, steps (b)-(g) in Fig. 1 are taken to build image representation. Similar to shape classification, we use linear SVM for image classification.

Comparison with SIFT-based method. We directly compare our contour fragment feature in BCF with dense SIFT feature in [9] using the same coding method (LLC), the same pooling method (SPM), the codebooks of the same size (1024) in Table 7, and the same classifier (linear SVM). The results of LLC [9] and RBC [57] are obtained by running the source code released the authors. The performance of BCF with SPM is 54.5%, which is worse than 71.7% of the SIFT feature with LLC and SPM. Contour fragment feature performs worse

Table 7: Classification accuracy on Caltech 101 dataset

Methods		Average accuracy (%)
SVM-KNN [59]		66.2±0.4
SLRR [60]		73.6
LSGC [61]		75.1
LLC [9]		71.7±0.8
RBC [57]		75.6±0.8
Shape Context [57]		3.0±0.7
BCF	level 1×1	23.9±0.8
	level 2×1	40.9±0.7
	level 3×3	49.8±0.7
	level 4×4	51.7±1.2
	pyramid	54.5±1.5
	pyramid+LLC	75.4±0.8
	pyramid+RBC	77.8±1.0

for two reasons: (1) some object contours (e.g., the outline of car in Fig. 12) and some object parts (e.g., the noses of person in Fig. 12) are missing in the edge maps; even though the edge maps are obtained by the state-of-the-art edge detector; (2) contextual information, such as, the ground in car image and the grass and tree in the elephant image, cannot be captured by our contour fragment feature. All this information is useful for recognition and can be capture by SIFT feature. Although, BCF performs worse than SIFT, we show BCF and SIFT feature are complementary to each other in Table 7. LLC [9] and RBC [57] are two SIFT based approaches; by combining BCF with them using the simple LP-β method in [58], the average image classification accuracy can be improved by 3.7% and 2.2%, respectively.

Comparison with previous shape-based method. We implemented shape context [44] method for image classification by setting 16 reference points in binary edge image resulting 960-dimensional feature vector. Then we use a linear SVM for image classification based on the shape context feature vector. The average image classification accuracy of shape context feature is only 3%. Both shape context and BCF are pure shape-based method. By diving contour into fragments and encoding their shape context features, BCF can obtain an average classification accuracy of 54.5% which is a significant improvement. It means that BCF is more robust to occlusions/edge-broken in real image than the previous shape descriptor.

The effectiveness of spatial pyramid. In Table 7, we show that from level 1×1 to 4×4 the accuracy of BCF improves from 23.9% to 51.7%; by combining the four levels, the accuracy of BCF (denoted as "pyramid") is 54.5%. This shows that SPM is effective for BCF in image classification.

We also quote some results from very recent literatures, e.g., [60, 61], and a classical method called SVM-KNN in [59] in Table 7. In summary, we give comprehensive studies of BCF for real image classification and show good performance by combining BCF with SIFT-based methods, which is better than the most recent results in [60, 61].

5. Conclusions

In this paper, we present a novel shape representation called BCF for shape classification. To the best of our knowledge, this is the first paper that introduces the idea of BoW together with LLC and SPM for shape representation. Since BCF is a part-based model, it is intrinsically robust to occlusion and deformation of shape. In the experiments, we have extensively tested the performance of BCF; all these experimental results on shape benchmarks show that BCF is able to achieve the state-of-the-art performance; moreover, we have tested BCF for image classification on the real image dataset and stress it can dramatically outperform the other shape-based method and is complementary to the texture descriptor. In the future, we will study how to use BCF for object recognition in real image; for example, on edge map extracted from real image, BCF can either do object recognition by combining with sliding window method, or provide shape cue for other off-shelf object detectors.

Acknowledgment

We would like to thank the anonymous reviewer for their helpful suggestions. This work was supported by National Natural Science Foundation of China (NSFC) Grants 61222308 and 61173120, the Program for New Century Excellent Talents in University in China, National Science Foundation (NSF) Grants OIA-1027897 and IIS-1302164, and Fundamental Research Funds for the Central Universities (HUST 2013TS115). We thank Mr. Ho Simon Wang for assisting to improve the language of this paper.

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, J. Malik, Contour detection and hierarchical image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 33 (2011) 898–916.

[2] S. Belongie, J. Malik, J. Puzicha, Shape matching and object recognition using shape contexts, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (2002) 509–522.

[3] H. Ling, D. Jacobs, Shape classification using the inner-distance, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (2007) 286–299.

[4] F. Mokhtarian, S. Abbasi, J. Kittler, et al., Efficient and robust retrieval by shape content through curvature scale space, Series on Software Engineering and Knowledge Engineering 8 (1997) 51–58.

[5] N. Alajlan, I. El Rube, M. Kamel, G. Freeman, Shape retrieval using triangle-area representation and dynamic space warping, Pattern Recognition 40 (2007) 1911–1920.

[6] X. Bai, W. Liu, Z. Tu, Integrating contour and skeleton for shape classification, in: International Conference on Computer Vision Workshops (ICCV Workshops), 2009, IEEE, pp. 360–367.

[7] J. Sivic, A. Zisserman, Video google: A text retrieval approach to object matching in videos, International Conference on Computer Vision, 2003 (2003) 1470–1477.

[8] G. Csurka, C. Dance, L. Fan, J. Willamowski, C. Bray, Visual categorization with bags of keypoints, Workshop on Statistical Learning in Computer Vision, ECCV, 2004 (2004).

[9] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, Y. Gong, Locality-constrained linear coding for image classification, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010, IEEE, pp. 3360–3367.

[10] F. Perronnin, J. Sánchez, T. Mensink, Improving the fisher kernel for large-scale image classification, in: Europe Conference on Computer Vision, 2010, Springer, 2010, pp. 143–156.

[11] D. Lowe, Distinctive image features from scale-invariant keypoints, International journal of computer vision 60 (2004) 91–110.

[12] T. Ahonen, A. Hadid, M. Pietikainen, Face description with local binary patterns: Application to face recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (2006) 2037–2041.

[13] L. Latecki, R. Lakämper, Convexity rule for shape decomposition based on discrete contour evolution, Computer Vision and Image Understanding 73 (1999) 441–454.

[14] S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, in: IEEE Conference on Computer Vision and Pattern Recognition, 2006, volume 2, IEEE, pp. 2169–2178.

[15] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, A. Zisserman, The pascal visual object classes (voc) challenge, International Journal of Computer Vision 88 (2010) 303–338.

[16] K. Sun, B. Super, Classification of contour shapes using class segment sets, in: IEEE Conference on Computer Vision and Pattern Recognition, 2005, volume 2, IEEE, pp. 727–733.

[17] M. Daliri, V. Torre, Robust symbolic representation for shape recognition and retrieval, Pattern Recognition 41 (2008) 1782–1798.

[18] M. Daliri, V. Torre, Shape recognition based on kernel-edit distance, Computer Vision and Image Understanding 114 (2010) 1097–1103.

[19] B. Wang, W. Shen, W. Liu, Y. X, B. X, Shape classification using tree-unions, in: International Conference on Pattern Recognition, 2010, IEEE, pp. 983–986.

[20] A. Torsello, E. Hancock, Learning shape-classes using a mixture of tree-unions, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (2006) 954–967.

[21] A. Erdem, S. Tari, A similarity-based approach for shape classification using aslan skeletons, Pattern Recognition Letters 31 (2010) 2024–2032.

[22] K. Siddiqi, A. Shokoufandeh, S. Dickinson, S. Zucker, Shock graphs and shape matching, International Journal of Computer Vision 35 (1999) 13–32.

[23] A. Torsello, E. Hancock, A skeletal measure of 2d shape similarity, Computer Vision and Image Understanding 95 (2004) 1–29.

[24] X. Bai, L. Latecki, Path similarity skeleton graph matching, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (2008) 1282–1292.

[25] E. Baseski, A. Erdem, S. Tari, Dissimilarity between two skeletal trees in a context, Pattern Recognition 42 (2009) 370–385.

[26] W. Kim, Y. Kim, A region-based shape descriptor using zernike moments, Signal Processing: Image Communication 16 (2000) 95–102.

[27] D. Zhang, G. Lu, Generic fourier descriptor for shape-based image retrieval, in: IEEE International Conference on Multimedia and Expo, volume 1, IEEE, pp. 425–428.

[28] T. Adamek, N. O'Connor, A multiscale representation method for nonrigid shapes with a single closed

contour, IEEE Transactions on Circuits and Systems for Video Technology 14 (2004) 742–753.

[29] G. McNeill, S. Vijayakumar, Hierarchical procrustes matching for shape retrieval, in: IEEE Conference on Computer Vision and Pattern Recognition, 2006, volume 1, IEEE, pp. 885–894.

[30] P. Felzenszwalb, J. Schwartz, Hierarchical matching of deformable shapes, in: IEEE Conference on Computer Vision and Pattern Recognition, 2007, IEEE, pp. 1–8.

[31] C. Xu, J. Liu, X. Tang, 2d shape matching by contour flexibility, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (2009) 180–186.

[32] S. Lewin, X. Jiang, A. Clausing, Perceptually motivated shape evolution with shape-preserving property, Pattern Recognition Letters 31 (2010) 447–453.

[33] J. Philbin, O. Chum, M. Isard, J. Sivic, A. Zisserman, Lost in quantization: Improving particular object retrieval in large scale image databases, in: IEEE Conference on Computer Vision and Pattern Recognition, 2008, IEEE, pp. 1–8.

[34] K. Grauman, T. Darrell, The pyramid match kernel: Discriminative classification with sets of image features, in: International Conference on Computer Vision, 2005, volume 2, IEEE, pp. 1458–1465.

[35] A. Krizhevsky, I. Sutskever, G. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1106–1114.

[36] S. Eslami, N. Heess, J. Winn, The shape boltzmann machine: a strong model of object shape, in: IEEE Conference on Computer Vision and Pattern Recognition, 2012, IEEE, pp. 406–413.

[37] L. Latecki, R. Lakamper, T. Eckhardt, Shape descriptors for non-rigid shapes with a single closed contour, in: IEEE Conference on Computer Vision and Pattern Recognition, 2000, volume 1, IEEE, pp. 424–429.

[38] K.-L. Lim, H. Galoogahi, Shape classification using local and global features, in: Pacific-Rim Symposium on Image and Video Technology, 2010, pp. 115–119.

[39] X. Bai, X. Yang, L. Latecki, Detection and recognition of contour parts based on shape similarity, Pattern Recognition 41 (2008) 2189–2199.

[40] F. Moosmann, E. Nowak, F. Jurie, Randomized clustering forests for image classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (2008) 1632–1646.

[41] J. Yang, K. Yu, T. Huang, Supervised translation-invariant sparse coding, in: IEEE Conference on Computer Vision and Pattern Recognition, 2010, IEEE, pp. 3517–3524.

[42] R. Duda, P. Hart, D. Stork, Pattern classification and scene analysis (1995).

[43] S. Roweis, L. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (2000) 2323–2326.

[44] T. Serre, L. Wolf, T. Poggio, Object recognition with features inspired by visual cortex, in: IEEE Conference on Computer Vision and Pattern Recognition, 2005, volume 2, IEEE, pp. 994–1000.

[45] J. Yang, K. Yu, Y. Gong, T. Huang, Linear spatial pyramid matching using sparse coding for image classification, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, IEEE, pp. 1794–1801.

[46] K. Crammer, Y. Singer, On the algorithmic implementation of multiclass kernel-based vector machines, The Journal of Machine Learning Research 2 (2002) 265–292.

[47] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, C.-J. Lin, LIBLINEAR: A library for large linear classification, Journal of Machine Learning Research 9 (2008) 1871–1874.

[48] R. Fan, K. Chang, C. Hsieh, X. Wang, C. Lin, Liblinear: A library for large linear classification, The Journal of Machine Learning Research 9 (2008) 1871–1874.

[49] B. Leibe, B. Schiele, Analyzing appearance and contour based methods for object categorization, in: IEEE Conference on Computer Vision and Pattern Recognition, 2003, volume 2, IEEE, pp. II–409.

[50] M. Daliri, V. Torre, Shape recognition and retrieval using string of symbols, in: International Conference on Machine Learning and Applications, 2006, IEEE, pp. 101–108.

[51] E. Attalla, P. Siy, Robust shape similarity retrieval based on contour segmentation polygonal multiresolution and elastic matching, Pattern Recognition 38 (2005) 2229–2241.

[52] O. Söderkvist, Computer vision classification of leaves from swedish trees, Ph.D. thesis, Linköping, 2001.

[53] R. Hu, W. Jia, H. Ling, D. Huang, Multiscale distance matrix for fast plant leaf recognition, IEEE transactions on image processing (2012).

[54] R. Hu, W. Jia, Y. Zhao, J. Gui, Perceptually motivated morphological strategies for shape retrieval, Pattern Recognition (2012).

[55] J. Wang, X. Bai, X. You, W. Liu, L. Latecki, Shape matching and classification using height functions, Pattern Recognition Letters (2011).

[56] P. D. Kovesi, MATLAB and Octave functions for computer vision and image processing, Centre for Exploration Targeting, School of Earth and Environment, The University of Western Australia, 2013. Available from: <http://www.csse.uwa.edu.au/∼pk/research/matlabfns/>.

[57] X. Wang, X. Bai, W. Liu, L. J. Latecki, Feature context for image classification and object detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2011, IEEE, pp. 961–968.

[58] P. Gehler, S. Nowozin, On feature combination for multiclass object classification, in: International Conference on Computer Vision, 2009, IEEE, pp. 221–228.

[59] H. Zhang, A. C. Berg, M. Maire, J. Malik, Svm-knn: Discriminative nearest neighbour classification for visual category recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2006, volume 2, IEEE, pp. 2126–2136.

[60] Y. Zhang, Z. Jiang, L. S. Davis, Learning structured low-rank representations for image classification,

in: IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 676 – 683.

[61] A. Shaban, H. Rabiee, M. Farajtabar, M. Ghazvininejad, From local similarity to global coding; an application to image classification, in: IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 2794 – 2801.