Pattern Recognition Letters 106 (2018) 33–40

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Bag of Shape Features with a learned pooling function for shape recognition

Wei Shen, Chenting Du, Yuan Jiang, Dan Zeng*, Zhijiang Zhang

Key Laboratory of Specialty Fiber Optics and Optical Access Networks, Shanghai Institute for Advanced Communication and Data Science, School of Communication and Information Engineering, Shanghai University, 200444, China

ARTICLE INFO

Article history:
Received 3 September 2017
Available online 28 February 2018

ABSTRACT

Bag of Shape Features (BoSF), such as Bag of Contour Fragments (BoCF) and Bag of Skeleton-associated Contour Parts (BoSCP), derived from the well-known Bag of Features (BoF), is an effective framework for shape representation. The feature pooling in this framework is a critical step, while either max pooling or average pooling is not a learnable process. In this paper, we aim at learning a pooling function which is adaptive to the input shape features instead. Towards this end, we formulate our pooling function as a weighted sum of max pooling and average pooling, where the weight is expressed by an activation function of the input shape features. To automatically learn this weight, the output of the pooling function is fed into a SVM classifier and they are trained jointly to minimize a shape classification loss. Experimental results on several standard shape datasets demonstrate the effectiveness of the proposed learned pooling function, which can achieve considerable improvements compared with both BoCF and BoSCP.

1. Introduction

Shape plays an important role in object recognition. The objects shown in Fig. 1 have lost their brightness, color and texture information, however we can still recognize their categories by their silhouettes. In other words, shape can be a stable feature representation which is hardly influenced by object color, texture and light conditions. Due to these advantages, shape recognition is a very important part in the field of object recognition for a long time. shape recognition is usually considered as a classification task which aims at predicting the given testing shape's category label as we have trained a classification model based on a set of training shapes as well as labels. The main obstacle in shape recognition is how to form a reliable shape representation which is invariant to local shape deformation while discriminative to different shape classes.

A shape generally can be represented by its contour, a closed curve, or its skeleton, a graph. To form a reliable shape representation, Bag of Shape Features (BoSF), such as Bag of Contour Fragments (BoCF) [36] and Bag of Skeleton-associated Contour Parts (BoSCP) [29], derived from the well-known Bag of Features (BoF) [13,31], converts a shape contour or a skeleton into an informative feature vector, avoiding the contour points matching or skeleton graph matching processes in traditional shape recognition methods [8,20]. The BoSF feature vector is formed by encoding and pooling the local contour fragment features. The pooling function used in BoSF is a fixed max pooling function, but as investigated in [19], learning a pooling function adaptive to input data can benefit performance.

In the current literature, popular pooling functions include max, average, and stochastic pooling. Some more complex pooling operation, such as spatial pyramid pooling, is designed to deal with different resolution images. In this paper, we propose to learn a pooling function which is adaptive to the input shapes features in the BoSF framework. More specifically, we formulate our pooling function as a weighted sum of max-pooling and average-pooling, where the weight is expressed by an activation function of the input shape features. The output of the pooling function is fed into a SVM classifier [12] and the pooling function and the classifier can be trained jointly to minimize a shape classification loss.

Our method is inspired by [19], which learns pooling function in a deep convolutional neural network. The input data of the pooling function are ordered and have a fixed length. While in our problem, the input data of the pooling function are responses on a visual word of a set of shape features, which are unordered and have unfixed lengths. To address this issue, we quantize the responses on each visual word into a fixed number of bins. Then the weight is learned based on the quantization histograms.

The core contribution of the paper is the proposal of the learnable pooling function in the BoSF framework, where we not only

* Corresponding author.
E-mail address: dzeng_shu@outlook.com (D. Zeng).

https://doi.org/10.1016/j.patrec.2018.02.024
0167-8655/

Fig. 1. Human biological vision system is able to recognize these object without any appearance information (brightness, color and texture).

provide an effective way to convert the shape features into a proper input format for the pooling function, but also describe how to learn the pooling function jointly with a shape classifier.

This paper extends our preliminary work [28] by the following contributions: (1) Verifying the effectiveness of the proposed learned pooling function on BoSCP [29]. (2) Discussing some possible alternative designs for the components in our framework. (3) Achieving the state-of-the-art shape classification performance on several standard shape datasets.

The rest of this paper is organized as follows. We review the related works in Section 2. Then, in Section 3 we will introduce the details of our method, i.e., how to learn pooling function in the BoSF framework to recognize shapes. Next, we evaluate the proposed method on several popular shape benchmarks in Section 4. Finally, we will give a conclusion in Section 5.

2. Related work

Shape recognition has been widely studied in the past decade. In early age, most methods aimed at extracting informative and robust shape descriptors. There are two main types of methods, one is contour-based, including curvature scale space (CSS) [23], multi-scale convexity concavity (MCC) [1], triangle area representation (TAR) [2], hierarchical procrustes matching (HPM) [22], shape-tree [14], contour flexibility [37]. Some well-known contour based shape descriptors need to be mentioned. Belongie et al. [8] introduced a shape descriptor named shape context (SC) which describes the relative spatial distribution (distance and orientation) of landmark points sampled on the object contour around feature points. Ling and Jacobs [20] used inner distance to extend shape context to capture articulation. These methods extracted local deformation invariant features at each point, and then match them by using sequence matching, such as Dynamic Time Warping (DTW) [25] and Optimal Subsequence Bijection (OSB) [5]. The other is skeleton-based, among which the shock graph and its variants [21,27] are most popular, which are abstracted from skeletons by designed shape grammar. Bai and Latecki [5] proposed a simple but informative skeleton-based shape descriptor named skeleton paths, which achieves promising shape recognition results. However, shape recognition by these two types of descriptors need cyclic sequence matching or graph matching, which is time consuming.

Bag of Words(BoW) has been widely used in image retrieval and classification as well as in 3D shape classification and retrieval tasks [3]. Tabia and Laga [32] extended the standard Bag of Words(BoW) and utilized multiple vocabulary coding for 3D retrieval with Bag of Covariances. Ramesh et al. [24] adopted coding-based frameworks as well as invariant features and contextual in Bag-of-words model for shape classification. Wang et al. [35] constructed middle-level global descriptor with Bag of Features which proved effective for high-level feature learning. Inspired from the well-known Bag of Features [13,31] framework, Wang et al. [36] proposed Bag of Contour Fragments (BoCF). They used Local-constraint linear coding (LLC [34]) to encode local contour fragment features and used max pooling to generate a compact feature vector, which would be then fed into a SVM classifier for shape classification. Since BoCF can convert a shape into a feature vector, using this representation for shape recognition is very efficient. Many researchers followed this coding based framework. Bai et al. [4,7] applied this framework to both 2D and 3D shape retrieval. Shen et al. [30] proposed a skeleton based mid-level representation named Bag of Skeleton Paths (BoSP) and concatenated the BoCF and BoSP for shape recognition. The weights between BoCF and BoSP are automatically learned by a SVM classifier [12]. Shen et al. [29] associates skeletal information with a shape contour on low-level by making full use of the natural correspondence between a contour and its skeleton. We also use this framework in our method, but the pooling function is learnable and jointly learned with the SVM classifier in our method.

The pooling operation has played a central role in many important frameworks, such as convolutional neural networks (CNNs) [18] and deep belief nets (DBN) [15], contributing to invariance to data variation and perturbation. However, pooling operations have been little revised beyond the current primary options of average, max, and stochastic pooling [10,11,38]; this despite indications that e.g. choosing from more than just one type of pooling operation can benefit performance [26]. Lee et al. [19] proposed to generalize the pooling function in a Convolutional Neural Network (CNN). Instead of combining these two pooling functions, they investigated how to combine average pooling and max pooling by a weight learned from the input data of a pooling layer. Our method is inspired from [19], but differs in frameworks (BoF vs CNN) and input data structures (responses of unfixed lengths and orders vs responses of the fixed length and order).

3. Methodology

In this section, we detail the proposed method for shape recognition. First, we briefly review the Bag of Shape Features framework. Then, we introduce the proposed learnable pooling function. Finally, we discuss how to jointly learn a pooling function and a shape classifier.

3.1. Bag of Shape Features framework

Bag of Shape Features (BoSF), such as Bag of Contour Fragments (BoCF) [36] and Bag of Skeleton-associated Contour Parts (BoSCP) [29], derived from the well-known Bag-of-Features (BoF) [13,31], is an effective framework for shape representation. In this framework, shape features, such as shape contour fragments or skeleton parts, are first converted into informative feature vectors, which are then encoded by a learned codebook and passed to a pooling function to form a shape representation.

In Bag of Contour Fragments (BoCF) [36], a shape F is represented by a set of meaningful contour fragments $G_{C(F)} = \{g_{pq}, p \neq q, p, q \in \{1, \ldots, T\}\}$, where p, q are two critical points [16] and T is the number of the critical points. For each contour fragment g_{pq}, the shape context (SC) [8] descriptor is used to represent it, which results in a feature vector $\mathbf{x}_{pq}$.

In Bag of Skeleton-associated Contour Parts (BoSCP) [29], a shape F is represented by its contour $C(F)$ and skeleton $S(F)$. Similar to SC descriptor, a shape part set $G_{C(F)}$ is build. $G_{C(F)} = \{g_{pq}, p \neq q, p, q \in \{1, \ldots, T\}\}$. Associated object thickness value can be computed by each point on the contour $C(F)$ and its correspond points on the skeleton $S(F)$. Given a contour part, we uniformly sample n points on it, then for a given reference contour point r_p, we describe its descriptor by the distribution of relative differences to the n sampled points on Euclidean distance, orientation and associated object thickness value, which is described a coarse histogram h_p. Finally, the SSC descriptors of the reference points on a contour part g_{pq} are concatenated to form the descriptor vector $\mathbf{x}_{pq}$.

After extracting the features in the Bag of Shape Features framework, each shape is represented by the shape context(SC) or skeleton-associated shape context(SSC) descriptors. To

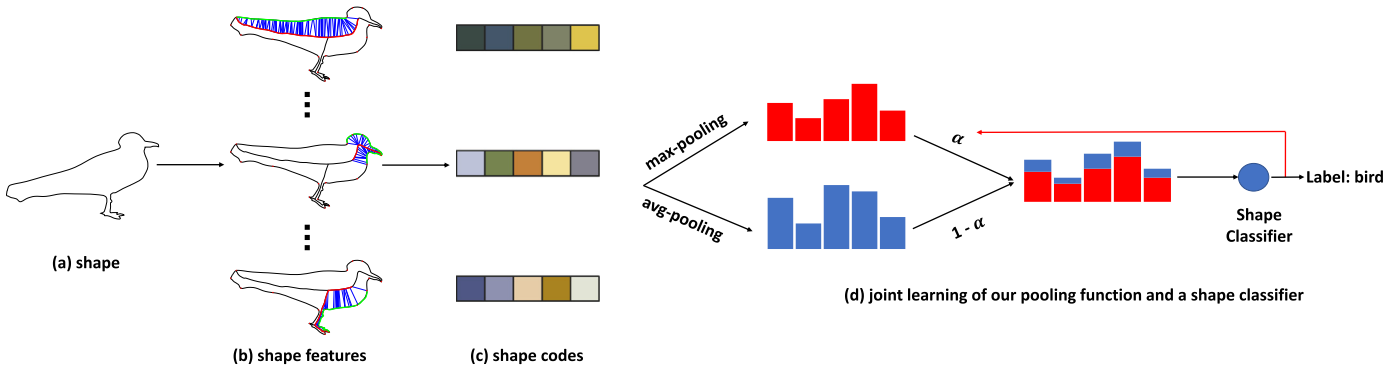

Fig. 2. The pipeline of our method for shape recognition. (a) is an input shape. (b) are the features from the input shape. (c) are the shape codes corresponding to the shape features in (b). In (d), the red histogram (top left), the blue histogram (bottom left) and the mixed histogram (right) stand for the shape representation obtained by max pooling, average pooling and our learnable pooling respectively. Our pooling function can be learned jointly with the classifier (See the red feedback arrow). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

encode shape parts, we adopt local-constrained linear coding (LLC) [34] scheme, as it has been proved to be efficient and effective for image classification. Codebook construction is usually achieved by unsupervised learning, such as k-means. Given a set of contour parts randomly sampled from all the shapes in a dataset as well as their flipped mirrors, we apply k-means algorithm to cluster them into K clusters and construct a codebook $\mathbf{B} = (b_j; j = 1, \ldots, K)$. LLC additionally incorporates locality constraint, which solves the following constrained least square fitting problem:

$$\min_{\mathbf{c}'_i} ||\mathbf{x}_{pq} - \mathbf{B}'\mathbf{c}'_i||, s.t. \mathbf{1}^T\mathbf{c}'_i = 1, \tag{1}$$

where $\mathbf{B}'$ is the local bases formed by the k nearest neighbors of $\mathbf{x}_{pq}$ and $\mathbf{c}'_i \in \mathbb{R}^k$ is the reconstruction coefficients. The code of $\mathbf{x}_{pq}$ encoded by the codebook $\mathbf{B}$, i.e. $\mathbf{c} \in \mathbb{R}^K$, can be easily converted from $\mathbf{c}'_i$ by setting the corresponding entries of $\mathbf{c}_i$ are equal to $\mathbf{c}'_i$'s and other are zero. After such an encoding process, each feature $\mathbf{x}_{pq}$ of shape F is encoded into a shape code $\mathbf{c}_i = (c_{ij}; j = 1, \ldots, K)^T$, where K is the codebook size. Assuming that there are N_s shape features extracted from the shape F, where N_s is the number of the contour segments in $G_{C(F)}$. After LLC encoding, a set of shape codes $\{\mathbf{c}_i\}_{i=1}^{N_s}$ is obtained, where $\mathbf{c}_i$ is the shape code of the ith shape feature in F.

To form an informative and compact representation $\mathbf{v} = (v_j; j = 1, \ldots, K)^T$ for the shape F, a pooling function is applied to $\{\mathbf{c}_i\}_{i=1}^{N_s}$. The shape codes are pooled into a compact shape feature vector by Spatial Pyramid Matching (SPM). SPM is usually used to incorporate spatial layout information when pooling the image codes. It divides an image into different subregions and each one pooled respectively and has been employed to enhance the performance [33]. Two pooling functions are commonly used. One is max pooling:

$$v_j = f_{\max}(\{c_{ij}\}_{i=1}^{N_s}) = \max_{i \in \{1,\ldots,N_s\}} c_{ij}, \tag{2}$$

the other is average pooling:

$$v_j = f_{\text{avg}}(\{c_{ij}\}_{i=1}^{N_s}) = \frac{1}{N_s}\sum_{i}^{N_s} c_{ij}, \tag{3}$$

3.2. Bag of Shape Features with a learned pooling function

Now we propose our method for shape recognition. We adopt the Bag of Shape Features (BoSF) framework. The pooling function used in BoSF is max pooling, but it is difficult to draw a conclusion that max pooling dominates average pooling. Here, instead of directly using max pooling or average pooling to obtain the final shape representation, we propose to learn a pooling function via combining max and average pooling. We formulate our pooling function as a weighted sum of max and average pooling, where the weight is expressed by an activation function of the input shape features. Thus, our pooling function is adaptive to the input shape codes and can be jointly learned with a shape classifier.

The process of our learnable pooling function is shown as in Fig. 2(d). We will introduce the detail about our learnable pooling function next.

A straightforward way to combine max and average pooling is to sum their results by a weight:

$$v_j = \alpha_j f_{\max}(\{c_{ij}\}_{i=1}^{N_s}) + (1 - \alpha_j) f_{\text{avg}}(\{c_{ij}\}_{i=1}^{N_s}), \tag{4}$$

where α_j is a weight factor. Rather than using a fixed α_j, we would like to learn a data-adaptive α_j. In [19], such a weight is expressed by a nonlinear transformation of the input data. However, the input data should be ordered and have a fixed length. Unfortunately, our input data is a set of shape codes, which are unordered and may have different numbers from one shape to another. To address this issue, we propose to quantize the shape codes corresponding to each visual word into a fixed number of bins. More specifically, given $\{c_{ij}\}_{i=1}^{N_s}$, which represents the shape codes corresponding to the jth visual word in the codebook, our goal is to quantize them into M bins to form a M-dimensional histogram. As we know that each c_{ij} satisfies that $0 \le c_{ij} \le 1$, we divide the interval $(0, 1]$ into M uniform bins, i.e., $(0, 1/M], (1/M, 2/M], \ldots, (1 - 1/M, 1]$. Then we count the number of nonzero values fall in each bin correspond to the jth visual word, which results in a quantization histogram, denoted by $\mathbf{h}_j = (h_{jm}; m = 1, \ldots, M)^T$, where

$$h_{jm} = \#\{c_{ij} \in \text{bin}(m)\}, i \in \{1, \ldots, N_s\}, \tag{5}$$

and

$$\text{bin}(m) = (\frac{1}{M}(m-1), \frac{m}{M}], m \in \{1, \ldots, M\}. \tag{6}$$

By this way, we convert the unordered and unfixed-length shape codes into ordered and fixed-length quantization histograms. Fig. 3 shows an example to quantize two set of shape codes computed from two different shapes. The numbers of shape codes from these two sets are different (N_1 and N_2 respectively), while the formed quantization histograms have the same number of bins ($M = 5$). $\mathbf{h}_j$ is another representation of shape codes $\{c_{ij}\}$, which reflects how strong the shape codes response is on the jth visual word. We can express the α_j in Eq. (4) by $\alpha_j = \sigma(\mathbf{w}_j^T\mathbf{h}_j)$, where $\sigma(\cdot)$ is a sigmoid activation function and $\mathbf{w}_j = (w_{jm}; m = 1, \ldots, M)^T$ is a transformation vector. Now we can rewrite Eq. (4) by

$$v_j = \sigma(\mathbf{w}_j^T\mathbf{h}_j) f_{\max}(\{c_{ij}\}_{i=1}^{N_s}) + [1 - \sigma(\mathbf{w}_j^T\mathbf{h}_j)] f_{\text{avg}}(\{c_{ij}\}_{i=1}^{N_s}). \tag{7}$$

Finally, the shape representation of shape F obtained by our learnable pooling function is: $\mathbf{v}(F) = (v_1, v_2, \ldots, v_K)^T$.

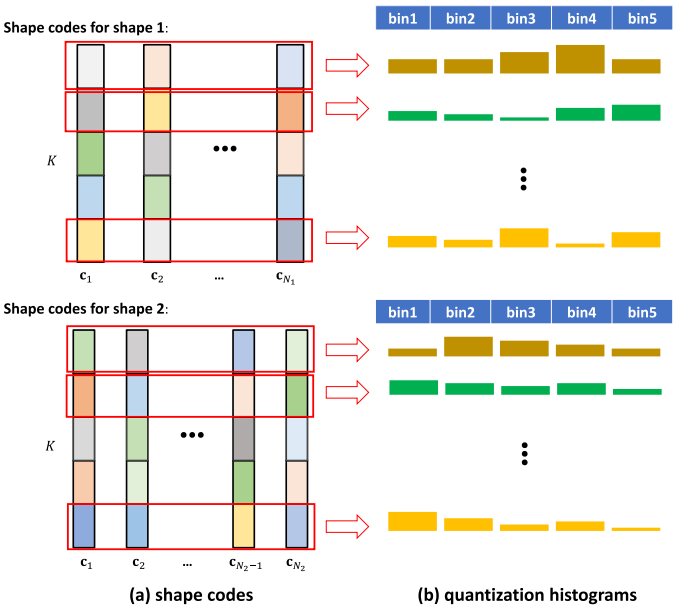

Fig. 3. Shape code quantization. (a) The shape codes computed from two different shapes, whose numbers are N_1 and N_2 respectively. (b) The quantization histograms of the shape codes in (a). Each row shows the shape codes and their quantization histograms corresponding to one visual word. The shape codes are quantized into $M = 5$ bins uniformly in the interval (0, 1].

3.3. Joint learning of a pooling function and a classifier

Since $\mathbf{h}_j$ is directly computed from the input of our pooling function, i.e., $\{c_{ij}\}_{i=1}^{N_s}$, α_j is adaptive to input data . So the transformation vector $\mathbf{w}_j$ can be learned from the data. Feeding the output of our pooling function into a classifier, e.g., SVM [12], the transformation vector $\mathbf{w}_j$ and the classifier can be learned jointly to minimize a shape classification loss.

Given a training set $\{\mathbf{v}_s, y_s\}_{s=1}^{N}$ consisting of N shapes from L classes, where $\mathbf{v}_s$ is the shape representation of the sth shape, $y_s \in \{1, 2, \ldots, L\}$ is the class label of the sth shape. Then we train a multi-class linear SVM classifier as follows:

$$\mathcal{L} = \min_{\mathbf{z}_1,\ldots,\mathbf{z}_L} \sum_{l=1}^{L} \|\mathbf{z}_l\|^2 + \beta \sum_{s=1}^{N} \max(0, 1 + \mathbf{z}_{l_s}^T \mathbf{v}_s - \mathbf{z}_{y_s}^T \mathbf{v}_s), \tag{8}$$

where$\mathcal{L}$ is the loss function of the multi-class SVM classifier, $l_s = \arg\max_{l\in\{1,2,\ldots,L\}, l\neq y_s} \mathbf{z}_l^T \mathbf{v}_s$, $\mathbf{z}_l$ is the lth dimension parameter of SVM to be learned and β is a hyper parameter to control the relative weight between the regulation term (the left part) and the multi-class hinge-loss term (the right part).

Stochastic gradient descent is used to minimize the loss $\mathcal{L}$. We can compute the gradient with respect to $\mathbf{w}_j$ by:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}_j^T} = \frac{\partial \mathcal{L}}{\partial \mathbf{v}_s^T} \frac{\partial \mathbf{v}_s}{\partial \mathbf{w}_j^T}, \tag{9}$$

where

$$\frac{\partial \mathcal{L}}{\partial \mathbf{v}_s^T} = \begin{cases} 0 & 1 + \mathbf{z}_{l_s}^T \mathbf{v}_s - \mathbf{z}_{y_s}^T \mathbf{v}_s \leq 0 \\ \mathbf{z}_{l_s}^T - \mathbf{z}_{y_s}^T & 1 + \mathbf{z}_{l_s}^T \mathbf{v}_s - \mathbf{z}_{y_s}^T \mathbf{v}_s > 0, \end{cases} \tag{10}$$

and

$$\frac{\partial \mathbf{v}_s}{\partial \mathbf{w}_j^T} = \sigma(\mathbf{w}_j^T \mathbf{h}_j)(1 - \sigma(\mathbf{w}_j^T \mathbf{h}_j)) \cdot (f_{\max}(\{c_{ij}\}_{i=1}^{N_s}) - f_{\text{avg}}(\{c_{ij}\}_{i=1}^{N_s}))\mathbf{diag}(\mathbf{h}_j), \tag{11}$$

where $\mathbf{diag}(\mathbf{h}_j)$ is a diagonal matrix whose diagonal entries are the elements in $\mathbf{h}_j$. For a testing shape, its shape representation obtained by our pooling function is $\mathbf{v}_t$, then its label can be predicted by: $\hat{y} = \arg\max_{l\in\{1,2,\ldots,L\}} \mathbf{z}_l^T \mathbf{v}_t$.

4. Experimental results

In this section, we evaluate the proposed learned pooling function in Bag of Shape Features frameworks on several shape benchmarks in comparison to the state-of-the-arts. We name our methods as BoCF-LP and BoSCP-LP, which refer to BoCF and BoSCP with a learned pooling function, respectively. We also investigate the effects of important parameters introduced in our method on classification accuracy.

4.1. Experimental setup

In order to forming shape feature descriptor vector, we concatenate the descriptors which is computed on 5 reference points. For each reference point, the bins for computing the features includes 5 Euclidean distance bins, 12 orientation bins and 5 object thickness difference bins, totally 300 bins. As a consequence, the dimension of a descriptor vector for each shape is 1500. When learning the codebook, the number of cluster centers (codebook size) is set to 2500 by default. To encode a shape part, we adopt LLC with 5 nearest neighbors. When using Spatial Pyramid Matching(SPM) pooling, a shape is divided into 1 !'Á 1, 2 !'Á 2 and 4 !'Á 4, in total 21 regions. The weight between the regularization term and the multi-class hinge-loss term in the multi-class linear SVM formulation is set to 10. The number of quantization bins is 10 ($M = 10$). For BoSCP, the number of object thickness different bins for computing SSC is 5 ($N_{td} = 5$). Also, we will discuss the reason of choosing these parameters.

All the experiments were carried out on a workstation (3.1 GHz 32-core CPU, 128G RAM and Ubuntu14.04 64-bit OS). It takes about 25 ms to compute the descriptor for one shape part, and about 1s to encode the feature vector for one shape. The process of feature computation and codebook learning takes nearly 8 hours. The learned pooling function and shape classification are trained end-to-end which costs 80 min. The testing process for one shape takes 15 ms.

We evaluate our method on several shape classification benchmark datasets, including the Animal dataset [6] and the MPEG-7 dataset [17]. To avoid the biases caused by randomness, the process of training and testing is repeated for 10 times. Average classification accuracy is reported to evaluate the performance of different shape classification methods. In each round, we randomly select half of shapes in each class to train and use the rest shapes to evaluate for every dataset.

4.2. Animal dataset

We first test our method on the Animal dataset which is introduced in [6]. This dataset contains 2000 shapes consisting of 20 kinds of animals. It is the most challenging shape dataset due to the large intra-class variations caused by view point change and various gestures of animals (as shown in Fig. 4). Following the previous methods [36], we randomly choose 50 shapes per class for training and leave the rest 50 shapes for testing. The comparison between our methods and other competitors is demonstrated in Table 1.

As shown in Table 1, our methods BoCF-LP and BoSCP-LP which use learned pooling function achieves outperforms results compared to the origin BoCF and BoSCP, which proves that the learned pooling function is more effective.

Fig. 4. Shapes of two classes from Animal dataset [6]. The first row belongs to label cat while the second one belongs to leopards. Some of these shapes have similar gestures which makes the shape recognition more difficult.

Table 1
Classification accuracy comparison on Animal dataset [6].

Algorithm	Classification accuracy %
Skeleton Paths [6]	67.90
Contour Segments [6]	71.70
IDSC [20]	73.60
ICS [6]	78.40
BoCF [36]	83.40 ± 1.30
Bioinformatic [9]	83.70
Shape Vocabulary [7]	84.30 ± 1.01
BoCF+BoSP [30]	85.50 ± 0.88
Contextual BOW model [24]	86.00
BoCF-LP (ours)	86.30 ± 0.20
BoSCP [29]	89.04 ± 0.95
BoSCP-LP (ours)	**89.77 ±0.65**

Fig. 5. Typical shapes of some classes from MPEG-7 dataset [17].

4.3. MPEG-7 dataset

Then we evaluate our method on the MPEG-7 dataset [17], which is the most well-known dataset for shape analysis in the field of computer vision. 1400 images of the dataset are divided into 70 classes with high shape variability, where there are 20 different shapes in each class. We show some shapes in Fig. 5. Average classification accuracy and standard derivation of classification accuracies are reported in Table 2.

Table 2
Classification accuracy comparison on MPEG-7 dataset [17].

Algorithm	Classification accuracy%
Skeleton Paths [6]	86.70
Contour Segments [6]	90.90
Bioinformatic [9]	96.10
ICS [6]	96.60
BoCF [36]	97.16 ± 0.79
BoCF+BoSP [30]	98.35 ± 0.63
BoCF-LP (Ours)	98.22 ± 0.20
BoSCP [29]	98.41 ± 0.63
BoSCP-LP (Ours)	**98.72 ±0.42**

As shown in Table 2, our method BoCF-LP achieves 98.22% compared to BoCF by over 1.1% on the MPEG-7 dataset. However, compared to BoSCP, BoSCP-LP achieves few improvements, which proves that the SSC feature descriptor is very good for shape classification. Also, the improvement on this dataset is not as significant as the one on the Animal dataset, the reason is the accuracies of the state-of-the-arts on this dataset have already approached to 100.

4.4. Parameter discussion

In this section, we discuss the effects of four parameters on shape classification accuracy.

4.4.1. The number of quantization bins

We first discuss when the number of the bins (M) used for quantization changes, how the classification accuracy is influenced on the Animal dataset [6]. As shown in Fig. 6, the shape classification accuracy changes slightly when the number of quantization bins varies. This experiment shows that the classification accuracy is not sensitive to the number of quantization bins. As a result, the number of quantization bins is 10.

4.4.2. The number of object thickness different bins for computing SSC

In the BoSCP framework, when forming the SSC descriptors, the number of object thickness different bins is a key component to affect the performance. As shown in Fig. 7, our method achieves

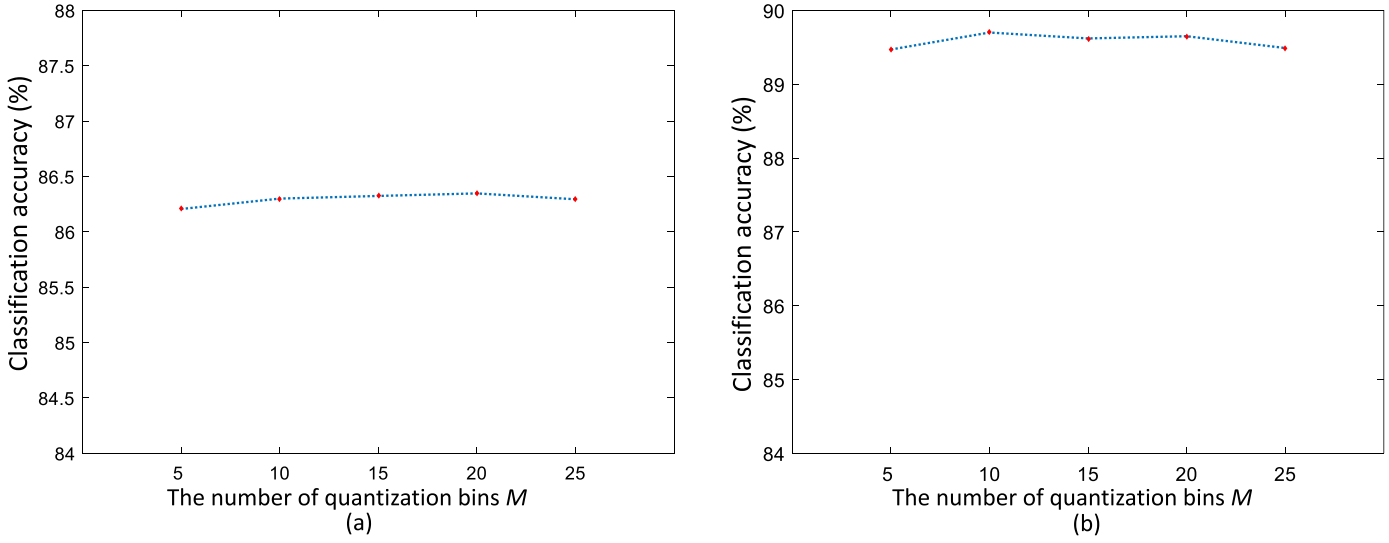

Fig. 6. Classification accuracies on Animal dataset [6] by varying the number of quantization bins. (a) shows the result of BoCF-LP. (b) shows the result of BoSCP-LP.

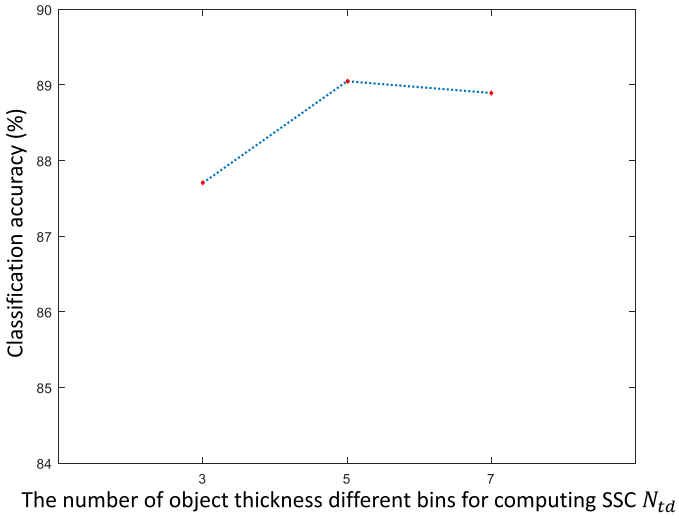

Fig. 7. Classification accuracies of BoSCP-LP on Animal dataset [6] by varying the number of object thickness different bins for computing SSC.

the best performance when N_{td} is set to 5. SSC with small N_{td} can only give a coarse representation of the thickness information, while losing most of the information a skeleton provides. Although $N_{td} = 7$ leads to a result close to the best one, it will result in significant increase in SSC descriptor computation, codebook learning and feature encoding. As a result, we choose $N_{td} = 5$ to be the best trade-off between accuracy and efficiency.

4.4.3. Codebook size

In our experiment, we adopt codebook sizes, including 500, 1000, 1500, 2000, 2500 and 3000, to classify shapes on the Animal dataset. As shown in Fig. 8, as the codebook size increases, shape classification accuracy improves generally. However, when codebook size is too big, it will predict the shapes in the same category to be different categories. As a consequence, the codebook size in our experiment is 2500.

4.4.4. The number of reference points

We also show how performance changes by varying the number of reference points when computing our shape feature descriptor in Fig. 9. With the increase of the number of reference points, the classification accuracy is improved. However, using more reference points leads to a significantly time consuming shape feature computation process. To balance the performance and computational cost, we choose 5 reference points.

4.5. Method design discussion

In this section, we discuss some possible alternative designs for the components in our framework.

4.5.1. Feature division

In our framework, we adopt SPM to divide the extracted local features. SPM can encode spatial information among the short-range contour fragments in a coarse-to-fine way. If we remove SPM from our framework, the classification accuracy drops to 87.87% on the Animal dataset.

One shortcoming of SPM is it is less effective against rotation and translation. Instead of SPM, we can use other feature division approaches, such as the one proposed in [7] which proposed to divide the extracted local features according to other properties, instead of the spatial locations of the features. We test this method on the Animal dataset, which achieves a classification accuracy of 88.73%. This result shows that with a more variation robust feature division approach, our framework can achieves a better performance.

4.5.2. The joint learning strategy

As mentioned in Section 3.3, we learn the pooling function and the classifier jointly. Alternatively, we can learn them in a stepwise manner: We have tried to first learn the pooling function with the softmax loss, and then used the learned features to train a SVM classifier. But, this stepwise learning strategy results in a classification accuracy of 86.79% (BoSCP) on the Animal dataset, which is worse than our joint learning strategy.

4.5.3. Input-dependent pooling function

Since we formulate our pooling function as a weighted sum of max and average pooling functions, where the weights are expressed by an activation function of the input shape features, our pooling function is input-dependent. To verify the effectiveness of the input-dependent pooling function, we compare it with an input-independent pooling function: We

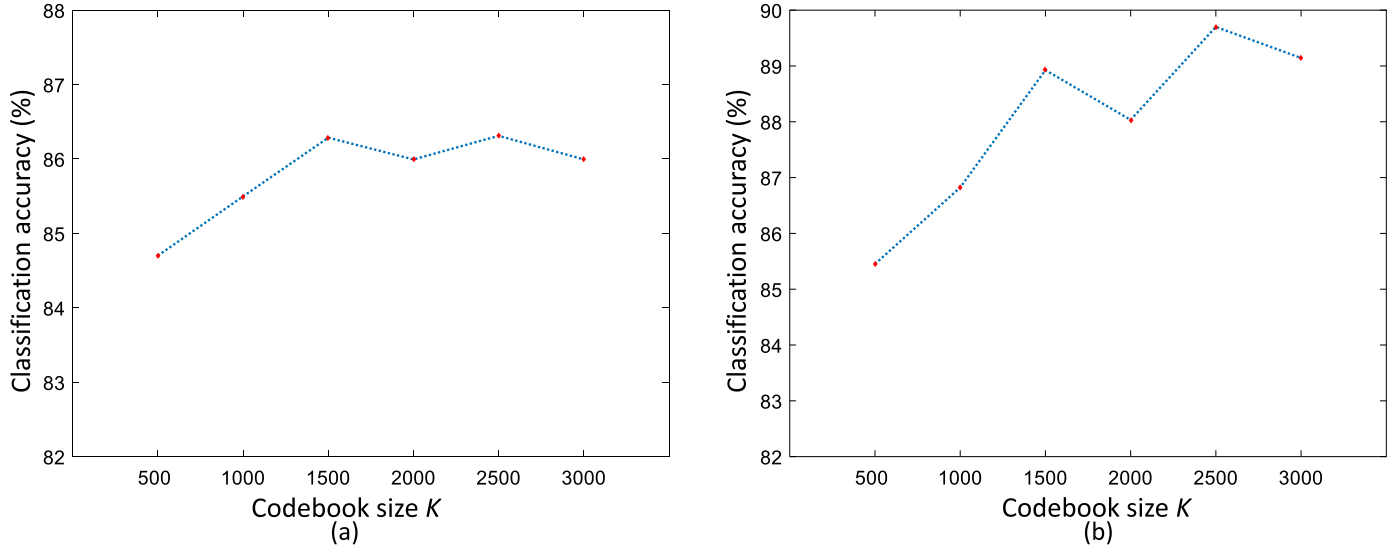

Fig. 8. Classification accuracies on Animal dataset [6] by varying the codebook size. (a) shows the result of BoCF-LP. (b) shows the result of BoSCP-LP.

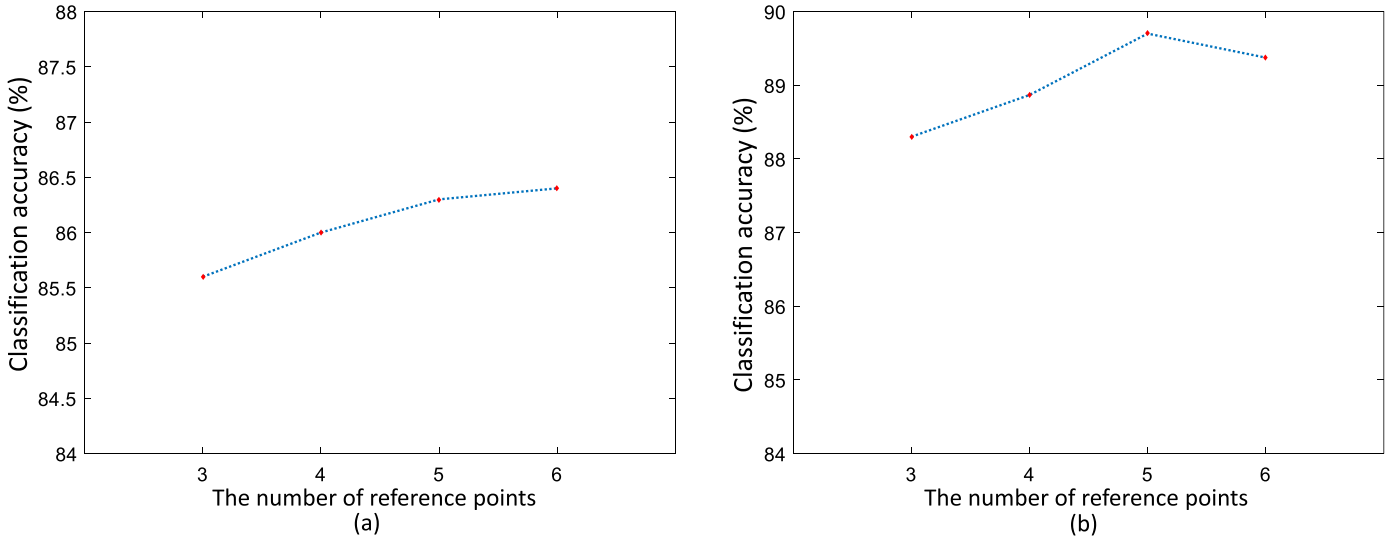

Fig. 9. Classification accuracies on Animal dataset [6] by varying the reference points. (a) shows the result of BoCF-LP. (b) shows the result of BoSCP-LP.

set $v_j = \sigma(w_j) f_{\max}(\{c_{ij}\}_{i=1}^{N_s}) + [1 - \sigma(w_j)] f_{\text{avg}}(\{c_{ij}\}_{i=1}^{N_s})$. This input-independent pooling function leads to a classification accuracy of 87.25% (BoSCP) on the Animal dataset, which is worse than our input-dependent learning pooling function.

5. Conclusion

In this paper, we propose a learnable pooling function which is adaptive to the input shape features. Bag of Shape Features (BoSF), such as Bag of Contour Fragments (BoCF) and Bag of Skeleton-associated Contour Parts (BoSCP), derived from the well-known Bag of Features (BoF), is an effective framework for shape representation. The proposed pooling function is a weighted sum of max pooling and average pooling, and the weights can be jointly learned with a shape classifier by gradient descent. The experimental results on two standard shape datasets demonstrate the effectiveness of the proposed learned pooling function.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China No. 61672336 and in part by “Chen Guang” project supported by Shanghai Municipal Education Commission and Shanghai Education Development Foundation No. 15CG43.

References

[1] T. Adamek, N.E. O'Connor, A multiscale representation method for nonrigid shapes with a single closed contour, IEEE Trans. Circuits Syst. Video Technol. 14 (2004) 742–753.
[2] N. Alajlan, I. El Rube, M.S. Kamel, G. Freeman, Shape retrieval using triangle-area representation and dynamic space warping, Pattern Recognit. 40 (2007) 1911–1920.
[3] S. Bai, X. Bai, Sparse contextual activation for efficient visual re-ranking, IEEE Trans. Image Process. 25 (2016) 1056–1069.
[4] X. Bai, S. Bai, Z. Zhu, L.J. Latecki, 3D shape matching via two layer coding, IEEE Trans Pattern Anal. Mach. Intell. 37 (2015) 2361–2373.
[5] X. Bai, L.J. Latecki, Path similarity skeleton graph matching, IEEE Trans. Pattern Anal. Mach. Intell. 30 (2008) 1282–1292.
[6] X. Bai, W. Liu, Z. Tu, Integrating contour and skeleton for shape classification, in: Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on, IEEE, 2009, pp. 360–367.
[7] X. Bai, C. Rao, X. Wang, Shape vocabulary: a robust and efficient shape representation for shape matching, IEEE Trans. Image Process. 23 (2014) 3935–3949.
[8] S. Belongie, J. Malik, J. Puzicha, Shape matching and object recognition using shape contexts, IEEE Trans. Pattern Anal. Mach. Intell. 24 (2002) 509–522.
[9] M. Bicego, P. Lovato, A bioinformatics approach to 2d shape classification, Comput. Vision Image Understand. 145 (2016) 59–69.
[10] Y.L. Boureau, N. Le Roux, F. Bach, J. Ponce, Y. LeCun, Ask the locals: multi-way local pooling for image recognition, in: Computer Vision (ICCV), 2011 IEEE International Conference on, IEEE, 2011, pp. 2651–2658.
[11] Y.L. Boureau, J. Ponce, Y. LeCun, A theoretical analysis of feature pooling in visual recognition, in: Proceedings of the 27th International Conference on Machine Learning (ICML-10), 2010, pp. 111–118.
[12] K. Crammer, Y. Singer, On the algorithmic implementation of multiclass kernel-based vector machines, J. Mach. Learn. Res. 2 (2001) 265–292.
[13] G. Csurka, C. Dance, L. Fan, J. Willamowski, C. Bray, Visual categorization with bags of keypoints, in: Workshop on Statistical Learning in Computer Vision, ECCV, Prague, 2004, pp. 1–2.
[14] P.F. Felzenszwalb, J.D. Schwartz, Hierarchical matching of deformable shapes, in: Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, IEEE, 2007, pp. 1–8.
[15] G.E. Hinton, S. Osindero, Y.W. Teh, A fast learning algorithm for deep belief nets, Neural Comput. 18 (2006) 1527–1554.
[16] L.J. Latecki, R. Lakämper, Convexity rule for shape decomposition based on discrete contour evolution, Comput. Vis. Image Understand. 73 (1999) 441–454.
[17] L.J. Latecki, R. Lakamper, T. Eckhardt, Shape descriptors for non-rigid shapes with a single closed contour, in: Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on, IEEE, 2000, pp. 424–429.
[18] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural Comput. 1 (1989) 541–551.
[19] C.Y. Lee, P.W. Gallagher, Z. Tu, Generalizing pooling functions in convolutional neural networks: mixed, gated, and tree, in: Artificial Intelligence and Statistics, 2016, pp. 464–472.
[20] H. Ling, D.W. Jacobs, Shape classification using the inner-distance, IEEE Trans. Pattern Anal. Mach. Intell. 29 (2007) 286–299.
[21] D. Macrini, S. Dickinson, D. Fleet, K. Siddiqi, Object categorization using bone graphs, Comput. Vision Image Understand. 115 (2011) 1187–1206.
[22] G. McNeill, S. Vijayakumar, Hierarchical procrustes matching for shape retrieval, in: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, IEEE, 2006, pp. 885–894.
[23] F. Mokhtarian, S. Abbasi, J. Kittler, Efficient and robust retrieval by shape content, Image Databases Multi-Media Search 8 (1998) 51.
[24] B. Ramesh, C. Xiang, T.H. Lee, Shape classification using invariant features and contextual information in the bag-of-words model, Pattern Recognit. 48 (2015) 894–906.
[25] H. Sakoe, S. Chiba, Dynamic programming algorithm optimization for spoken word recognition, IEEE Trans. Acoust. 26 (1978) 43–49.
[26] D. Scherer, A. Müller, S. Behnke, Evaluation of pooling operations in convolutional architectures for object recognition, Artificial Neural Netw.–ICANN 2010 (2010) 92–101.
[27] T.B. Sebastian, P.N. Klein, B.B. Kimia, Recognition of shapes by editing their shock graphs, IEEE Trans. Pattern Anal. Mach. Intell. 26 (2004) 550–571.
[28] W. Shen, W. Gao, Y. Jiang, D. Zeng, Z. Zhang, Shape recognition by bag of contour fragments with a learned pooling function, in: International Conference on Image Processing, 2017, Proceedings. IEEE Conference on, 2017.
[29] W. Shen, Y. Jiang, W. Gao, D. Zeng, X. Wang, Shape recognition by bag of skeleton-associated contour parts, Pattern Recognit. Lett. 83 (2016) 321–329.
[30] W. Shen, X. Wang, C. Yao, X. Bai, Shape recognition by combining contour and skeleton into a mid-level representation, in: Chinese Conference on Pattern Recognition, Springer, 2014, pp. 391–400.

[31] J. Sivic, A. Zisserman, Video google: a text retrieval approach to object matching in videos, null, IEEE (2003) 1470.
[32] H. Tabia, H. Laga, Multiple vocabulary coding for 3d shape retrieval using bag of covariances, Pattern Recognit. Lett. (2017).
[33] P. Tang, X. Wang, Z. Huang, X. Bai, W. Liu, Deep patch learning for weakly supervised object classification and discovery, Pattern Recognit. (2017).
[34] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, Y. Gong, Locality-constrained linear coding for image classification, in: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, IEEE, 2010, pp. 3360–3367.
[35] M. Wang, J. Xie, F. Zhu, Y. Fang, Linear discrimination dictionary learning for shape descriptors, Pattern Recognit. Lett. 83 (2016) 349–356.
[36] X. Wang, B. Feng, X. Bai, W. Liu, L.J. Latecki, Bag of contour fragments for robust shape classification, Pattern Recognit. 47 (2014) 2116–2125.
[37] C. Xu, J. Liu, X. Tang, 2D shape matching by contour flexibility, IEEE Trans. Pattern Anal. Mach. Intell. 31 (2009) 180–186.
[38] Zeiler M.D., Fergus R., Stochastic pooling for regularization of deep convolutional neural networks, 2013, arXiv:1301.3557.