SHAPE RECOGNITION BY BAG OF CONTOUR FRAGMENTS WITH A LEARNED POOLING FUNCTION

Wei Shen, Wenjing Gao, Yuan Jiang, Dan Zeng, and Zhijiang Zhang

Key Laboratory of Specialty Fiber Optics and Optical Access Networks, Shanghai University

ABSTRACT

Bag of Contour Fragments (BoCF), 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 contour fragment 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 contour fragment 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 BoCF.

Index Terms— Shape classification, Bag of Contour Fragments, max pooling, average pooling, learned pooling function

1. INTRODUCTION

Shape plays an important role in object recognition, especially when the object in an image lost its brightness, color and texture information. Shape recognition is the task that aims at predicting which object category an input shape belongs to.

A shape generally can be represented by its contour, a closed curve. 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. Bag of Contour Fragments (BoCF) [1], derived from the well-known Bag-of-Features (BoF) [2, 3], converts a shape contour into an informative feature vector, avoiding the contour point matching process in traditional shape recognition methods [4, 5]. The BoCF feature vector is formed by encoding and pooling the local contour fragment features. The pooling function used in BoCF is a fixed max pooling function, but as investigated in [6], learning a pooling function adaptive to input data can benefits performance.

In this paper, we propose to learn a pooling function which is adaptive to the input contour fragment features in the BoCF 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 contour fragment features. The output of the pooling function is fed into a SVM classifier and the pooling function and the classifier can be trained jointly to minimize a shape classification loss.

Our method is inspired by [6], 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 contour fragment 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 BoCF framework, where we not only provide an effective way to convert the contour fragment features into a proper input format for the pooling function, but also describe how to learn the pooling function jointly with a shape classifier.

2. RELATED WORK

Shape recognition has been widely studied in the past decade. Traditional methods often extract local deformation invariant features, like shape context (SC) [4] and the inner distance shape context (IDSC) [5], at each point on each shape contour, and then match them by using sequence matching, such as Dynamic Time Warping (DTW) [7] and Optimal Subsequence Bijection (OSB) [8]. In order to accommodate more degrees of freedoms of transformations, [9, 10] used partially elliptical features. While to convert a closed curve to a sequence is a non-trivial problem. Normally, sequence matching has to be performed multiple times to obtain an optimal correspondence, which is time consuming.

The Bag of Contour Fragments, proposed by Wang et al. [1], which had proven its effectiveness in shape classification and shape retrieval [11], is an unconventional shape recognition framework. It used Local-constraint linear coding (LLC [12]) 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. Shen *et al.* [13, 14] used this framework to efficiently combine contour and skeleton features for shape recognition. We also use this framework in our method, but the pooling function is learnable and jointly learned with the SVM classifier in our method.

Lee et al. [6] proposed to generalize the pooling function in a Convolutional Neural Network (CNN). 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 [6], 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 Contour Fragments 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 Contour Fragments

In the framework of Bag of Contour Fragments (BoCF) [1], a shape S is represented by a set of meaningful contour fragments $G(S) = \{g_{pq}, p \neq q, p, q \in C(S)\}$, where p, q are two critical points [15] on the contour $C(S)$. For each contour fragment g_{pq}, the shape context (SC) [4] descriptor is used to represent it, which results in a d-dimensional feature vector $\mathbf{x}_{pq} \in R^{d\times 1}$.

To obtain the BoCF representation for a shape, the set of contour fragment features is encoded by a learned codebook, followed by being passed to a pooling function. The codebook is constructed by clustering the contour fragment features extracted from a training shape set, and Locality-constrained Linear Coding (LLC) [12] is used to encode the contour fragment features into shape codes. After such an encoding process, each contour fragment feature $\mathbf{x}_{pq}$ of shape S is encoded into a shape code $\mathbf{c} = (c_j; j = 1, \ldots, K)^T$, where K is the codebook size.

Assuming that there are N_s contour fragments extracted from the shape S, 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 i-th contour fragment in S. To form an informative and compact representation $\mathbf{v} = (v_j; j = 1, \ldots, K)^T$ for the shape S, a pooling function is applied to $\{\mathbf{c}_i\}_{i=1}^{N_s}$. Two pooling functions are commonly used. One is max pooling:

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

the other is average pooling:

$$v_j = f_{\text{avg}}(\{c_{ij}\}_{i=1}^{N_s}) = \frac{1}{N_s}\sum_{i}^{N_s} c_{ij}, \tag{2}$$

3.2. Bag of Contour Fragments with A Learned Pooling Function

Now we propose our method for shape recognition. We adopt the Bag of Contour Fragments (BoCF) framework. The pooling function used in BoCF 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 contour fragment 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.1(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:

$$v_j = \alpha_j f_{\text{max}}(\{c_{ij}\}_{i=1}^{N_s}) + (1-\alpha_j) f_{\text{avg}}(\{c_{ij}\}_{i=1}^{N_s}), \tag{3}$$

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 [6], 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 j-th 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 j-th visual word, which results in a quantization histogram, denoted by $\mathbf{h}_j = (h_{jm}; m = 1, \ldots, M)^T$, where

$$h_{jm} = \#\{c_{ij} \in \text{bin}(m)\}, i \in \{1, \ldots, N_s\}, \tag{4}$$

and

$$\text{bin}(m) = (\frac{1}{M}(m-1), \frac{m}{M}], m \in \{1, \ldots, M\}. \tag{5}$$

By this way, we convert the unordered and unfixed-length shape codes into ordered and fixed-length quantization histograms. Fig. 2 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

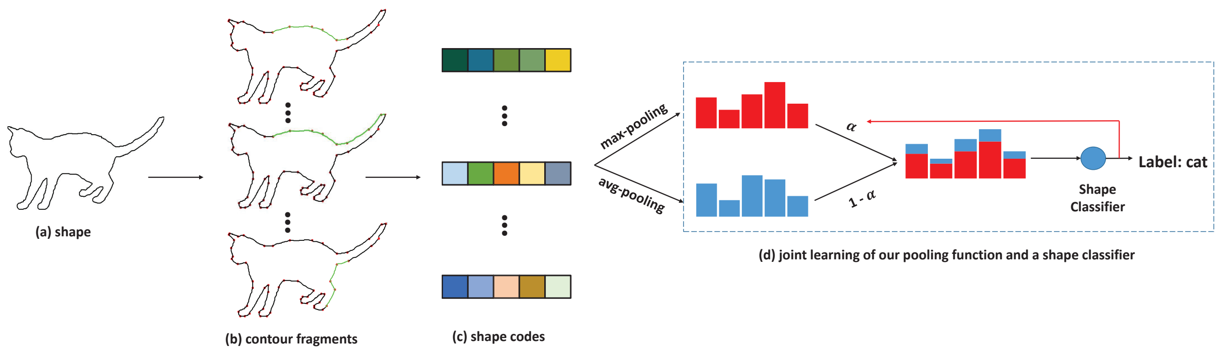

Fig. 1. The pipeline of our method for shape recognition. (a) is an input shape. (b) are the contour fragments from the input shape. (c) are the shape codes corresponding to the contour fragments in (c). In (d), the red histogram (top left), the blue histogram (bottom left) and the mixed histogram (right) 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).

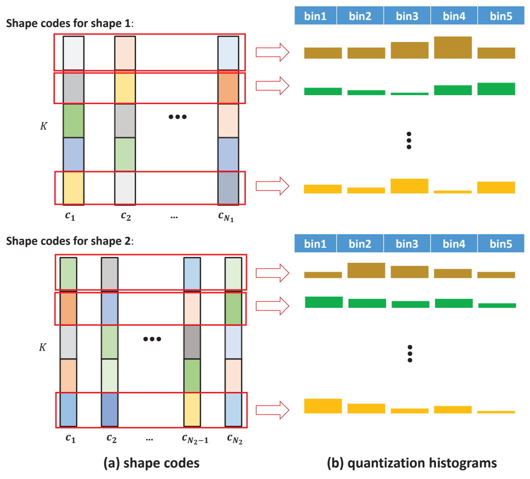

Fig. 2. Shape code quantization. (a) The shape codes computed from two different shapes, whose numbers are N_1 and N_2 respectively. (b) The quantization histograms of the shape codes in (a). Each row shows the shape codes and their quantization histograms corresponding to one visual word. The shape codes are quantized into $M = 5$ bins uniformly in the interval $(0, 1]$.

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 j-th visual word. We can express the α_j in Eqn. 3 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 Eqn. 3 by

$$v_j = \sigma(\mathbf{w}_j^T \mathbf{h}_j) f_{\max}(\{c_{ij}\}_{i=1}^{N_s}) + [1 - \sigma(\mathbf{w}_j^T \mathbf{h}_j)] f_{\text{avg}}(\{c_{ij}\}_{i=1}^{N_s}). \tag{6}$$

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

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 [16], 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 s-th shape, $y_s \in \{1, 2, \ldots, L\}$ is the class label of the s-th shape. Then we train a multi-class linear SVM classifier as follows:

$$\mathcal{L} = \min_{\mathbf{z}_1, \ldots, \mathbf{z}_L} \sum_{l=1}^{L} \|\mathbf{z}_l\|^2 + \beta \sum_{s=1}^{N} \max(0, 1 + \mathbf{z}_{l_s}^T \mathbf{v}_s - \mathbf{z}_{y_s}^T \mathbf{v}_s), \tag{7}$$

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 l-th 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:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}_j^T} = \frac{\partial \mathcal{L}}{\partial \mathbf{v}_s^T} \frac{\partial \mathbf{v}_s}{\partial \mathbf{w}_j^T}, \tag{8}$$

where

$$\frac{\partial \mathcal{L}}{\partial \mathbf{v}_s^T} = \begin{cases} 0 & 1 + \mathbf{z}_{l_s}^T \mathbf{v}_s - \mathbf{z}_{y_s}^T \mathbf{v}_s \leq 0 \\ \mathbf{z}_{l_s}^T - \mathbf{z}_{y_s}^T & 1 + \mathbf{z}_{l_s}^T \mathbf{v}_s - \mathbf{z}_{y_s}^T \mathbf{v}_s > 0, \end{cases} \tag{9}$$

Algorithm	Classification accuracy
Skeleton Paths [17]	67.90%
Contour Segments [17]	71.70%
ICS [17]	78.40%
Bioinformatic [19]	83.7%
BoCF [1]	$83.40 \pm 1.3\%$
Ours	$\mathbf{86.3 \pm 0.2\%}$

Table 1. Classification accuracy comparison on Animal dataset [17]

Algorithm	Classification accuracy
Skeleton Paths [17]	86.70%
Contour Segments [17]	90.90%
Bioinformatic [19]	96.10%
ICS [17]	96.60%
BoCF [1]	97.16 ± 0.79 %
Ours	$\mathbf{98.22 \pm 0.2}$ **%**

Table 2. Classification accuracy comparison on MPEG-7 dataset [18]

and

$$\frac{\partial \mathbf{v}_s}{\partial \mathbf{w}_j^T} = \sigma(\mathbf{w}_j^T \mathbf{h}_j)(1 - \sigma(\mathbf{w}_j^T \mathbf{h}_j)) \cdot$$
$$(f_{\max}(\{c_{ij}\}_{i=1}^{N_s}) - f_{\text{avg}}(\{c_{ij}\}_{i=1}^{N_s}))\mathbf{diag}(\mathbf{h}_j), \qquad (10)$$

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. RESULTS AND DISCUSSIONS

We evaluate our method on several shape classification benchmark datasets, including the Animal dataset [17] and the MPEG-7 dataset [18]. 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.

4.1. Animal Dataset

We first test our method on the Animal dataset which is introduced in [17]. This dataset contains 2000 shapes consisting of 20 kinds of animals. Following the previous methods [1], we randomly choose 50 shapes per class for training and leave the rest 50 shapes for testing. The comparison between our method and other competitors is demonstrated in Table 1.

As shown in Table 1, the proposed method significantly outperforms the state-of-the-art method, which proves that the learned pooling function is more effective.

4.2. MPEG-7 Dataset

Then we evaluate our method on the MPEG-7 dataset [18], 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. Average classification accuracy and standard derivation of classification accuracies are reported in Table 2.

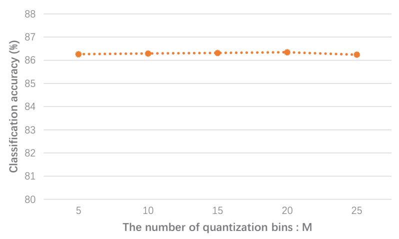

Fig. 3. Classification accuracies on Animal dataset [17] by varying the number of quantization bins.

As shown in Table 2, our method achieves the best performance on the MPEG-7 dataset. BoCF [1] has already obtained a good result, while our learned pooling function still leads an improvement.

4.3. The numbers of quantization bins

Then we will discuss when the the number of the bins (M) used for quantization changes, how the classification accuracy is influenced on the Animal dataset [17]. As shown in Fig. 3, 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.

5. CONCLUSION

In this paper, we proposed a learnable pooling function which is adaptive to the input contour fragment features. 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.

Acknowledgement. This work was supported in part by the National Natural Science Foundation of China under Grants 61672336 and 61303095, and in part by Chen Guang project supported by Shanghai Municipal Education Commission and Shanghai Education Development Foundation under Grant 15CG43.

6. REFERENCES

[1] Xinggang Wang, Bin Feng, Xiang Bai, Wenyu Liu, and Longin Jan Latecki, "Bag of contour fragments for robust shape classification," *Pattern Recognition*, vol. 47, no. 6, pp. 2116–2125, 2014.

[2] Josef Sivic and Andrew Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*. IEEE, 2003, pp. 1470–1477.

[3] Gabriella Csurka, Christopher R Dance, Lixin Fan, Jutta Willamowski, and Cdric Bray, "Visual categorization with bags of keypoints," *Workshop on Statistical Learning in Computer Vision Eccv*, pp. 1–22, 2004.

[4] Serge Belongie, Jitendra Malik, and Jan Puzicha, "Shape matching and object recognition using shape contexts," *IEEE transactions on pattern analysis and machine intelligence*, vol. 24, no. 4, pp. 509–522, 2002.

[5] Haibin Ling and David W Jacobs, "Shape classification using the inner-distance," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 2, pp. 286–299, 2007.

[6] Chen-Yu Lee, Patrick W Gallagher, and Zhuowen Tu, "Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree," in *International Conference on Artificial Intelligence and Statistics*, 2016.

[7] Hiroaki Sakoe and Seibi Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE transactions on acoustics, speech, and signal processing*, vol. 26, no. 1, pp. 43–49, 1978.

[8] Xiang Bai and Longin Jan Latecki, "Path similarity skeleton graph matching," *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 7, pp. 1282–1292, 2008.

[9] Panu Srestasathiern and Alper Yilmaz, "Planar shape representation and matching under projective transformation," *Computer Vision and Image Understanding*, vol. 115, no. 11, pp. 1525–1535, 2011.

[10] Qi Jia, Xin Fan, Yu Liu, Haojie Li, Zhongxuan Luo, and He Guo, "Hierarchical projective invariant contexts for shape recognition," *Pattern Recognition*, vol. 52, pp. 358–374, 2016.

[11] Xiang Bai, Cong Rao, and Xinggang Wang, "Shape vocabulary: A robust and efficient shape representation for shape matching," *IEEE Transactions on Image Processing*, vol. 23, no. 9, pp. 3935–3949, 2014.

[12] Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong, "Locality-constrained linear coding for image classification," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 3360–3367.

[13] Wei Shen, Xinggang Wang, Cong Yao, and Xiang Bai, "Shape recognition by combining contour and skeleton into a mid-level representation," in *Chinese Conference on Pattern Recognition*. Springer, 2014, pp. 391–400.

[14] Wei Shen, Yuan Jiang, Wenjing Gao, Dan Zeng, and X-inggang Wang, "Shape recognition by bag of skeleton-associated contour parts," *Pattern Recognition Letters*, vol. 83, pp. 321–329, 2016.

[15] Longin Jan Latecki and Rolf Lakämper, "Convexity rule for shape decomposition based on discrete contour evolution," *Computer Vision and Image Understanding*, vol. 73, no. 3, pp. 441–454, 1999.

[16] Koby Crammer and Yoram Singer, "On the algorithmic implementation of multiclass kernel-based vector machines," *Journal of machine learning research*, vol. 2, no. Dec, pp. 265–292, 2001.

[17] Xiang Bai, Wenyu Liu, and Zhuowen Tu, "Integrating contour and skeleton for shape classification," in *IEEE Workshop on NORDIA*, 2009.

[18] Longin Jan Latecki, Rolf Lakamper, and 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, vol. 1, pp. 424–429.

[19] Manuele Bicego and Pietro Lovato, "A bioinformatics approach to 2d shape classification," *Computer Vision and Image Understanding*, vol. 145, pp. 59–69, 2016.