Calligraphy Word Style Recognition by KNN Based Feature Library Filtering

Tianjiao Mao[1] Jiangqin Wu[2] Pengcheng Gao Yang Xia Yuan Lin

Abstract. Chinese calligraphy works is a valuable part of the Chinese culture heritage. More and more calligraphy works are digitized, preserved and exhibited in digital library so that people can enjoy the calligraphers' works conveniently. There are five main writing style categories of calligraphy words, namely, seal script, clerical script, standard script, semi-cursive script and cursive script. Users always want to appreciate the style-similar works simultaneously, so it's necessary to classify the words by their writing style. In this study, we proposed a method based on KNN and feature vector filtering. Firstly, extract the SIFT points from the training images, building up a feature library for SIFT feature vectors. Then use a KNN-based method to filter the feature library so that the style-irrelevant feature points can be wiped out. At last we use the filtered feature library to classify the word images by a modified KNN classifier. Experiments show that SIFT feature has better recognition result than that of Gabor feature and GIST feature, but the large amount of feature vectors in the SIFT feature library makes the KNN searching rather slow. To accelerate the recognition speed, Spectral Hashing is used to index the feature library, which makes it faster to classify feature points and gives no side effect on the recognition ratio.

Keywords: Calligraphy words • Writing style recognition • Feature vector filtering • Digital library

1 Introduction

The large amount of Chinese calligraphy works in existence is a valuable part of the Chinese cultural heritage. With the development of the digital library, a lot of

[1] T. Mao
Zhejiang University, Hangzhou, 310007 Zhejiang, China
e-mail: 21121260@zju.edu.cn

[2] J. Wu (✉)
Zhejiang University, Hangzhou, 310007 Zhejiang, China
e-mail: wujq@zju.edu.cn

excellent calligraphy works is digitized and preserved in the digital library. Users always want to enjoy the style-similar works simultaneously and navigate the collection in a style-guide manner, so the recognition of word style is one of the most important problems to be addressed. But there are several challenges for this problem, because calligraphy writing style is human cognition and aesthetics related, there's no explicit indicator to distinguish different calligraphy writing styles.

So far there are two main categories for image representation: global feature descriptor and local feature descriptor. One popular global feature descriptor is Gabor feature. Y. Zhuang and W. Lu proposed Latent Style Model to discover writing styles for calligraphy works [1], in their approach, they use 2D Gabor filter to extract texture feature and verified its discriminability for calligraphy writing style classification, the recognition ratio is more than 56% for each style. They also use Gabor feature to classify the writing style of Chinese words in [2]. It proves that Gabor feature is feasible for discriminate different calligraphy writing styles. Another global feature descriptor that can be used to distinguish different writing styles is GIST descriptor, which was initially proposed in [3]. It has recently shown good results for image search [4]. Y. Lin [5] used GIST descriptor for calligraphy word recognition and achieved good experiment results.

SIFT (Scale-invariant feature transform) descriptor which was published by David Lowe in 1999 [6] is proved to be one of the best local feature descriptors [7], it is widely used for object recognition [8, 9] and content based image retrieval [10]. However, the number of SIFT keypoints in one image often varies from tens to hundreds, matching these feature points brings vast computational cost.

In this study, we use SIFT feature to describe calligraphy words. To avoid the influence of style-irrelevant feature points, we apply a KNN-based feature filtering method to wipe the style-irrelevant feature points out of the feature library, and then use a modified KNN classifier to make the final classification. In order to improve the recognition speed, we test several methods to search the feature library, including k-dimensional tree (k-d tree)[11], Locality Sensitive Hashing (LSH)[12] and Spectral Hashing (SH)[13]. And finally choose Spectral Hashing as our solution due to its best performance in experiment.

2 Calligraphy Style Recognition

In this section, we will discuss how to use KNN classifier to filter the feature points and recognize the calligraphy style of a word image. For global feature descriptors such as Gabor feature or GIST feature, we can apply KNN in the easiest way: use a feature vector to represent the target image, find the nearest neighbors in the feature library according to the Euclid distance and assign the most frequent style occurs in the neighbors to the target image. But for SIFT descriptor, we must add some additional procedure, because each image has a lot of feature points, all

of them have to be classified individually. The final prediction is made by considering the writing style of each feature points. Furthermore, we must notice that not every feature points are capable for style discrimination. How to get rid of them is also an important problem to be considered.

2.1 Filtering the Feature Points

After extracting the SIFT feature of the training images, the feature points (each preserved by a feature vector) and its corresponding calligraphy style are stored in a feature library. But not every feature point carries calligraphy style information. For example, in Fig. 1, point a is a typical point in standard script, and point d is a typical point in clerical script, but point b and c are so common that can be found almost in every writing style.

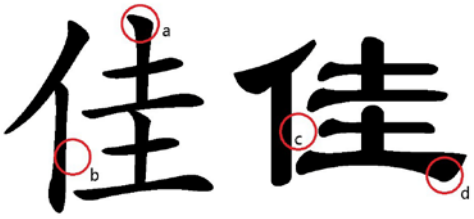

Fig. 1 SIFT points in the same word which means "good" (the left one is written in standard script, the right one is written in clerical script)

To remove the style-irrelevant feature points out of the feature library, we use a KNN-based filtering method. The main idea is that if a feature point is exclusive of its own writing style, the nearest neighbors of it must have the same writing style. So, for each feature point in the feature library, find its k nearest neighbors, if not all the neighbors have the same calligraphy style as it, remove it from the feature library. A pseudocode description for the filtering process is given in Algorithm 1.

Algorithm 1 FeatureFilter(L, k): Filter the feature library to remove the style-irrelevant feature points.
Given: L = The SIFT feature library, every feature vector T_i in the feature library is assigned with a calligraphy style C_i. Given: k = The number of nearest neighbors to search. Output: The filtered feature library. 1. Choose a feature vector T_i in L, get its calligraphy style C_i; 2. Search k nearest neighbor of T_i according to the Euclid distance; 3. If not all the style of the k nearest neighbors are C_i, remove the feature point from the feature library L; 4. Repeat 1~3, until all the feature points in L have been processed.

After the filtering process, a large amount of feature points are removed from the feature library. As a result, the size of the feature library is reduced and the negative effect of style-irrelevant feature points is eliminated.

2.2 Classify the Feature Points

To recognize the calligraphy style of a word image, we must classify all the feature points in the image. In this study, we use a modified KNN algorithm to classify the feature points, which gives a confidence factor f to the classify result. Algorithm 2 describes in pseudocode the procedure of classifying a feature point (represented by a feature vector).

Algorithm 2 Classify(T, L, k): Classify the SIFT feature vector T.
Given: T = The feature vector to be classified. Given: L = The SIFT feature library. Given: k = Number of nearest neighbor to search. Output: C = The predicted calligraphy style of T; f = confidence factor of the prediction. 1. Calculate the Euclid distance between T and all the feature vectors in L; 2. Let P = {k nearest feature vectors of T }; 3. Let c = the most frequent calligraphy style in P; m = count of feature vectors in P whose calligraphy style are c; 4. Let $C = c$; 5. Let $f = m / k$.

In Algorithm 2, we can learn that $f \in [1/N, 1]$, where N is the total count of calligraphy styles. The bigger the confidence factor f is, the more the classify result C is credible. When $f = 1$, it means that all the neighbors of the query point have the same calligraphy style, so the classify result can be very reliable; when $f = 1 / N$, it means that the count of neighbors in every calligraphy style are equal, in this case, the classify result is almost random.

2.3 Predict the Calligraphy Writing Style

Given a word image, we can extract all the feature points and predict their calligraphy writing style according to Algorithm 2. The final prediction must consider the classify results of every feature points. Algorithm 3 gives a pseudocode description on how to predict the calligraphy writing style of a word image.

Algorithm 3 Predict(P, L, F): Predict the calligraphy writing style of a word image.
Given: P = The input word image. Given: L = The SIFT feature library. Given: F = The minimum confidence factor. Output: C = The prediction calligraphy style of P. 1. Let S=S_1, S_2, … , S_m be all the calligraphy styles in L. 2. Extract SIFT feature of P, get n SIFT feature vectors T = { T_1, T_2, …, T_n}; 3. For each feature vector T_i(i=1,2,…,n), call Algorithm 2, get its prediction style C_i($C_i \in$ S)and confidence factor f_i; 4. For each j = S_1,S_2,…,S_m: Let Count[j] = 0; 5. For each i = 1, 2, …, n: If $f_i > F$, Count[C_i] = Count[C_i] + 1; 6. Let $C = \underset{c}{\text{argmax}}(\text{Count}[c])$.

2.4 Speed Up the Query Process

One shortcoming of KNN is the computational cost during the classify process, because very time when a feature point is to be classified, we need to compute the Euclid distance between the query point and every feature point in the feature library. For a large feature library (in which the number of feature points will be more than a million), the query time can be a serious problem. In this study, we test three methods to improve the query speed, including k-d tree, LSH and SH. Experiment shows that SH has the best performance.

3. Experiment Results and Discussion

In this section, we first introduce the datasets we used for the experiments, and then compare the recognition result using different features. At last we use k-d tree, LSH and SH respectively to speed up the query procedure and give a comparison of the query time they spend.

We use two independent datasets to conduct our experiments. Dataset1 comes from the system font, include five main calligraphy writing style categories (seal script, clerical script, standard script, semi-cursive script and cursive script), each category contains 3755 word images, about 18000 word images in all; Dataset2 comes from CADAL [14], also include five main calligraphy writing style categories, each category contains 1000 word images. 90% of the datasets is used for training and 10% is used for testing.

3.1 Recognition Result for Different Features

Fig. 2 shows the recognition ratio using Gabor feature, GIST feature and SIFT feature respectively.

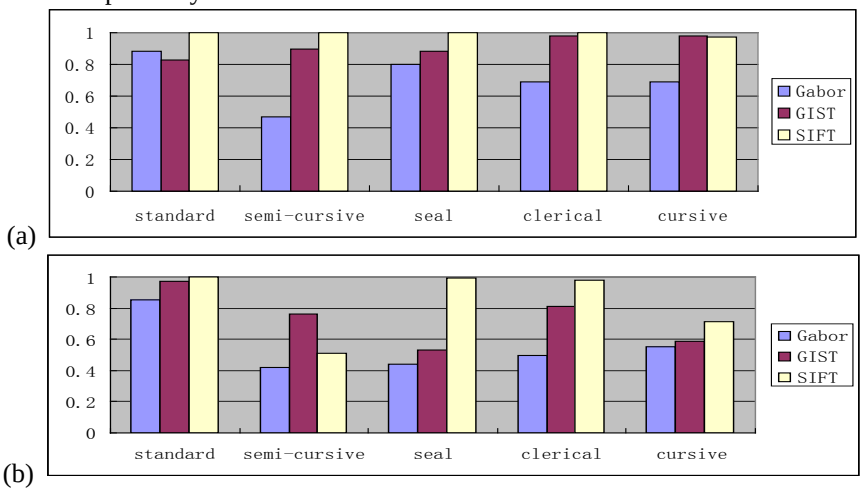

Fig. 2 Comparison of recognition ratio using Gabor feature, GIST feature and SIFT feature, (a) shows the recognition result on Dataset1, (b) shows the recognition result on Dataset2.

In Fig. 2 we can see that SIFT feature outperforms Gabor feature and GIST feature on both datasets. The average recognition ratio is above 99% on Dataset1.Dataset2 is from CADAL digital library, all the word images are written by ancient Chinese calligraphers, so the experiment result on Dataset2 can better reflects the ability to recognize real world data. Table. 1 gives the detailed recognition ratio on Dataset2 using SIFT feature.

Table. 1 Detailed recognition ratio on Dataset2

	Standard	Semi-cursive	Seal	Clerical	Cursive
Standard	1.00	0	0	0	0
Semi-cursive	0.11	0.51	0.01	0.33	0.04
Seal	0	0	0.99	0	0.01
Clerical	0	0.01	0.01	0.98	0
Cursive	0.02	0.10	0.14	0.03	0.71

From Table. 1 We can see that the recognition ratio of standard script, seal script and clerical script is very high, because the writing style of them are quit uniform, but for semi-cursive script and cursive script which is more freestyle, the recognition ratio is much lower, especially semi-cursive script, which is based on standard script and carries some characteristics of cursive script. When semi-

cursive script is taken out of the dataset, the recognition result is much better, see Table. 2.

Table. 2 Recognition result on Dataset2 without semi-cursive script

Standard	Seal	Clerical	Cursive
1.00	0.99	0.99	0.84

3.2 Query Time for Different Search Method

To improve the time performance of our recognition method, we use k-d tree, LSH and SH respectively to speed up the KNN searching procedure. Table. 3 shows the average query time for one feature point under different size of feature library.

Table. 3 Average query time (in millisecond) for one feature point using different searching method.

method / scale	linear search	k-d tree	LSH	SH
5000	4	12	3	1
10000	7.3	24	5	1.4
20000	14	48	10	2.8
40000	27	91	17	9
80000	53	181	43	19
160000	106	391	105	38
320000	209	601	187	77
640000	418	1190	349	155

Table. 3 shows that SH outperforms any other method we used. We can also see that k-d tree is the slowest one, even slower than linear search, which proves that k-d tree is not suitable for indexing high-dimensional data.

4. Conclusion

In this study, we proposed a method to recognize the writing style of Chinese word images based on KNN and feature library filtering. Firstly, extract the SIFT feature points from the training images, build up a feature library for SIFT feature vectors. Then use a KNN-based method to filter the feature library so that the style-irrelevant feature points can be removed from the feature library. Final pre-

diction is made by applying a modified KNN classifier on the feature library. Experiment shows that SIFT feature has better recognition result than that of Gabor feature or GIST feature. But the large amount of feature vectors in the feature library makes the KNN search rather slow. To accelerate the recognition process, we use Spectral Hashing to index the feature library, which gives better performance than LSH or k-d tree in the experiment. In the future, we will focus on combining global feature descriptor and local feature descriptor together to make a better way to recognize the writing style of Chinese word.

Acknowledgements

This work is supported by National Natural Science Foundation of China (No. 610 70066), the CADAL Project and Research Center, Zhejiang University.

5. References

1. Y. Zhuang, W. Lu, J. Wu. Latent Style Model: Discovering writing styles for calligraphy works. J. Visual Communication and Image Representation 2009, 20(2):84-96.
2. Y. Zhuang, W. Lu, J. Wu. Discovering calligraphy style relationships by supervised learning weighted random walk model. Multimedia Systems. 2009, 15(4):211-242.
3. A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. IJCV, 2001, 42(3):145-175.
4. M. Douze, H. Jegou, H. Singh, L. Amsaleg, C. Schmid. Evaluation of GIST descriptors for web-scale image search. CIVR, 2009.
5. Y. Lin, J. Wu, etc. LSH-Based Large Scale Chinese Calligraphic Character Recognition. JCDL, 2013.
6. David G. Lowe. Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision, 2004, 60(2):91-110.
7. K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. PAMI, 2004.
8. D.G. Lowe. Object Recognition from Local Scale-Invariant Features. *Proc. Seventh Int',l Conf. Computer Vision.* 1999:1150-1157.
9. B. Sirmacek and C. Ünsalan. Urban-area and building detection using SIFT Keypoints and graph theory, *IEEE Trans. Geosci. Remote Sens. 2009*, 47(4):1156–1167.
10. L. Ledwich and S. Williams. Reduced SIFT Features For Image Retrieval and Indoor Localisation. Australasian Conf. on Robotics and Automation, 2004.
11. Finkel R A, Bentley JL. Quad trees-a data structure for retrieval on composite keys. Acta Informatica, 1974, 4(1):1-9.
12. A. Andoni, P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. FOCS, 2006:459-468.
13. Weiss Y, Torralba A B, Fergus R. Spectral Hashing. Preceedings of Advances in Neural Information Processing Systems. Cambridge:MIT Press, 2008:1753-1760.
14. Xiafen Zhang, George Nagy. The CADAL Calligraphic Database. Proc. HIP, 2011.