Building Hierarchical Representations for Oracle Character and Sketch Recognition

Jun Guo, Changhu Wang, *Senior Member, IEEE*, Edgar Roman-Rangel, Hongyang Chao, *Member, IEEE*, and Yong Rui, *Fellow, IEEE*

***Abstract*— In this paper, we study oracle character recognition and general sketch recognition. First, a data set of oracle characters, which are the oldest hieroglyphs in China yet remain a part of modern Chinese characters, is collected for analysis. Second, typical visual representations in shape- and sketch-related works are evaluated. We analyze the problems suffered when addressing these representations and determine several representation design criteria. Based on the analysis, we propose a novel hierarchical representation that combines a Gabor-related low-level representation and a sparse-encoder-related mid-level representation. Extensive experiments show the effectiveness of the proposed representation in both oracle character recognition and general sketch recognition. The proposed representation is also complementary to convolutional neural network (CNN)-based models. We introduce a solution to combine the proposed representation with CNN-based models, and achieve better performances over both approaches. This solution has beaten humans at recognizing general sketches.**

***Index Terms*— Oracle character recognition, sketch recognition, hierarchical representation.**

I. Introduction

THE STUDY of invaluable historical hieroglyphs has played an important role in archeology and philology. Oracle characters, the oldest hieroglyphs in China, were carved on animal bones or turtle shells for divination purposes during the Bronze Age in ancient China [1], [2]. These hieroglyphs recorded the development of civilization during that period, thereby constituting a foundation for the study of Chinese etymologies and calligraphy as well as providing a glimpse into the history of China. Fig. 1 shows a piece of a turtle shell with characters.

Manuscript received July 25, 2014; revised December 5, 2014, May 13, 2015, and September 29, 2015; accepted October 28, 2015. Date of publication November 11, 2015; date of current version December 3, 2015. The work of J. Guo and H. Chao was supported in part by the Guangdong Natural Science Foundation, China, under Grant S2011020001215, in part by the Guangzhou Science and Technology Program, China, under Grant 201510010165, and in part by the National Natural Science Foundation of China under Grant 61173081. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Sylvain Paris. *(Corresponding author: Changhu Wang.)*

J. Guo and H. Chao are with the School of Data and Computer Science and the SYSU-CMU Shunde International Joint Research Institute, Sun Yat-sen University, Guangzhou 510006, China (e-mail: artanis.protoss@outlook.com; isschhy@mail.sysu.edu.cn).

C. Wang and Y. Rui are with Microsoft Research, Beijing 100080, China (e-mail: chw@microsoft.com; yongrui@microsoft.com).

E. Roman-Rangel is with the Computer Vision and Multimedia Laboratory, University of Geneva, Geneva GE-1227, Switzerland (e-mail: edgar.romanrangel@unige.ch).

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

Digital Object Identifier 10.1109/TIP.2015.2500019

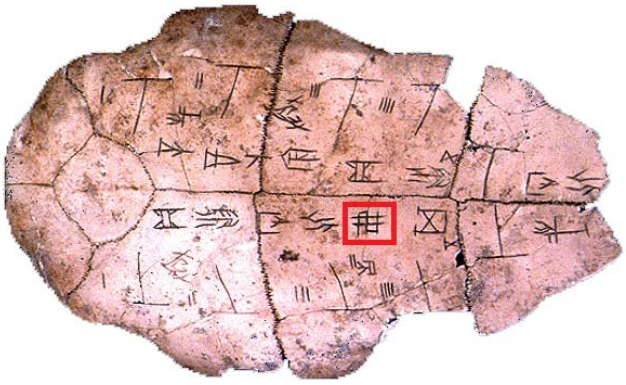


Fig. 1. A turtle shell with multiple oracle instances carved on it. Strokes in the red rectangle form an instance of oracle characters.

Currently, more than four thousand oracle characters have been discovered from hundreds of thousands of oracle bones, of which approximately fifty-five percent have been recognized. Unfortunately, identifying unlabeled or newly collected instances is a very time-consuming task, even for domain experts in archeology or paleography. Such experts usually have to manually scan printed catalogs such as [3] to find similar annotated instances to make further judgments.

However, despite continuous efforts from archaeologists and paleographists since the discovery of oracle bones one century ago, there is little research on oracle character analysis from the perspective of computer vision and no relevant public datasets. Thus, in this work, we attempt to analyze oracle characters via visual representations. On the one hand, we hope to develop an effective representation for encoding oracle characters, based on which automatic recognition / retrieval algorithms and tools can be developed to facilitate the work of archaeologists and paleographists. On the other hand, as hieroglyphs that remain alive in modern Chinese characters, oracle characters can be considered as wonderful and time-tested sketches of ancestors of objects in the real world, as shown in Fig. 2. We hope that the visual analysis of oracle characters can contribute to general sketch-related research. Note that in this paper a general sketch means a hand-drawn sketch of a general object in our daily life, not limited to an oracle character.

Generally, there are four major contributions in this work. First, to perform the above study, we collected 20,000+ legible oracle character images. To the best of our knowledge, this dataset is the largest oracle character dataset in computer vision and multimedia communities.

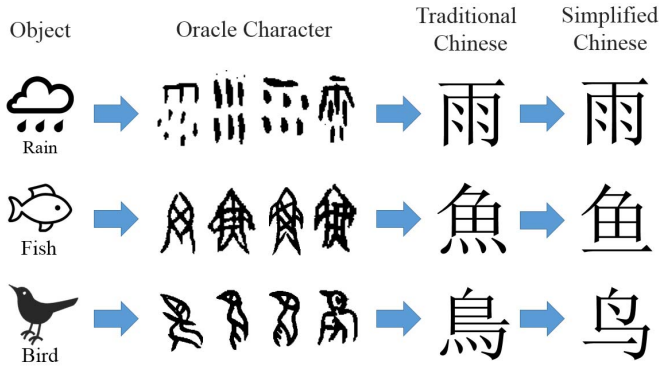


Fig. 2. Objects in the real world and their corresponding oracle characters, traditional Chinese characters, and simplified Chinese characters.

A big challenge for domain research such as oracle character recognition is the scarcity of domain-specific data. Although our dataset is large in the area of oracle recognition, it is still insufficient to directly learn a complex recognition model (see Section VI-D), i.e., domain-specific priors may be required. Thus, as our second contribution, based on the collected dataset, we attempt to look inside oracle characters to figure out important representation design criteria. We hope that this analysis can be generalized to sketch recognition as well. To achieve this goal, eleven popular low-level visual representations for shape / sketch recognition are investigated mainly from four perspectives: performance (classification accuracy), sampling strategy, filters for edge discovery, and scopes of local representations. Moreover, we conduct experiments to investigate the effectiveness of mid-level visual representations by combining the aforementioned low-level representations and the Bag-of-Words [4] framework.

Third, following the above analysis, we carefully develop our representation. In particular, inspired by the representation developed by Eitz *et al.* [5], which obtained good performances in the above experiments, we propose a hierarchical representation consisting of a low-level and a mid-level representation. In contrast to Eitz's work, the low-level representation is built on symmetric Gabor filters and dense sampling to better utilize edge information. Meanwhile, the sparse-encoder-related mid-level representation in conjunction with multiple-kernel-codebook quantization attempt to discover more salient patterns and combine both global structures and local details. Extensive experiments are conducted to evaluate the effects of different factors in the proposed representation as well as the overall comparison with mainstream representations applied to both oracle character recognition and general sketch recognition. Our experiments demonstrate the effectiveness of the proposed representation when applied to these two tasks. This meets our expectation that the analysis on oracle data does benefit general sketch-related research.

Recent works, including [6]–[8], show that, when external data are available, supervised pre-training on an auxiliary task, followed by fine tuning on domain-specific data, may represent an effective paradigm for learning complicated domain-specific models. In this work, we also conduct experiments to verify the usefulness of transfer learning in oracle character / general sketch recognition, and show that, with the help of external data, the proposed representation can also be combined with CNN-based solutions to achieve a large performance improvement in contrast to the case without auxiliary data, even outperforming humans in general sketch recognition. This is considered as the fourth contribution.

The proposed representation may bring three potential applications: 1) helping museum visitors recognize oracle characters using mobile devices, 2) assisting archaeologists and paleographists explore the evolution of Chinese characters, and 3) enabling computers to understand users' drawings and perform sketch-based image search on touch-screen devices.

The remainder of the paper is organized as follows. Sec. II briefly browses related low-level / mid-level representations. Sec. III introduces the collected oracle dataset, followed by a detailed comparison and analysis of existing visual representations in Sec. IV. The proposed representation is presented in Sec. V. Extensive experiments and potential applications are provided in Sec. VI and Sec. VII. We discuss some limitations and conclude the paper in Sec. VIII and Sec. IX.

II. Related Works

A. Low-Level Representations

The lack of color, texture and intensity information limits the choice of algorithms in oracle character recognition. Below, we list some low-level representations used in shape and sketch analysis.

Shape Context (SC) [9] is a local shape representation originally designed for recognizing shapes of simple objects and thus can be used for oracle character analysis. It describes each of N randomly sampled points using the relative positions of the $N-1$ remaining points, which are found in a log-polar grid defined around each sample point. Such N descriptors are then merged as an SC representation. Built on top of SC, Histogram of Orientation Shape Context (HOOSC) [10] was developed to address limitations of SC when applied to complex shapes, such as syllabic Maya hieroglyphs. HOOSC replaces the simple count of the $N-1$ points with a distribution of local orientations for each cell of the grid.

In addition to shape-related representations, such as SC and HOOSC, appearance representations may be applied, because oracle characters can also be regarded as normal gray images. Histograms of Oriented Gradients (HOG) [11] is a popular representation that computes histograms of local edge orientations on a grid of small cells that are uniformly spaced over the entire image and later groups histograms within a block (consisting of adjacent cells) into a bigger block to improve robustness. Similarly, GIST [12] averages the responses obtained by applying Gabor filters to non-overlapping blocks that split the image into well-localized sections to capture the "gist" of an image into a low-dimensional signature vector. Binary Coherent Edge Descriptor (BiCE) [13] does not encode relative gradient magnitudes. Instead, it encodes edge positions, orientations and local linear lengths to achieve robustness to intra-category appearance variation. Moreover, Fixed-Scale-and-Dense-Sampling Scale-Invariant Feature Transform (FD-SIFT) [5], a single-scale version of the dense SIFT representation [14], was proposed

to build slightly offset invariant representations for sketches by computing fixed-scale SIFT descriptors [15] over regularly sampled points. Note that FD-SIFT is equivalent to SHOG [16] when applied to oracle characters / sketches.

All the above-mentioned representations are vector-based representations. In contrast, Chamfer Matching (CM) [17], a non-vector-based matching method, measures the similarity of different shapes by calculating their Chamfer distance. Oriented Chamfer Matching (OCM) [18], which is based on Chamfer Matching, utilizes quantized orientations of edge pixels to enhance discrimination. Hitmap (HIT) [19], introduced by Mindfinder [20], is a variant of OCM for the purposes of indexing and improving local robustness and was demonstrated to work well in sketch-based image retrieval [21]. Recently proposed matching methods, including Exemplar SVM (ESVM) [22] and Discriminative Visual Elements (DVE) [23] (well applied in [24]), favor exemplar classification. ESVM learns a linear SVM for each query, taking the query as a single positive example and a large set of randomly selected patches as the negative examples; subsequently, it matches the dataset and the query based on the learned weight vector. DVE works in reverse order – exemplar classifiers are pre-computed from the dataset to avoid the computationally expensive online learning in ESVM and to overcome some partial occlusions or significant deformations. These two methods have produced promising results in image matching.

B. Mid-Level Representations

Although low-level representations can be manually designed, mid-level representations that combine low-level representations are substantially more difficult to be manually explored. Popular mid-level representations are designed using various types of learning procedures.

The most famous mid-level representation is the Bag-of-Words (BoW) framework [4], which learns a representation via k-means quantization. More precisely, this method first constructs a codebook by performing k-means on input vectors and then quantizes these inputs by assigning them to the nearest codeword to build the "one-hot" mid-level representation. Van Gemert *et al.* [25] introduced a variant of BoW by replacing the hard quantization by a soft quantization and achieved some improvements.

In addition to BoW, sparse coding also represents a popular choice for building mid-level representations. Approaches of this type usually use a linear combination of a small number of codewords to approximate inputs, with the combinatorial coefficients being the mid-level representation. Yang *et al.* [26] and methods providing several improvements, such as [27] and [28], obtained state-of-the-art results in certain vision tasks and demonstrated that sparse codes are able to capture salient properties of visual patterns and provide good performance with linear classifiers.

Recently, deep neural networks have become popular in building mid-level representations. Multi-layer Restricted Boltzmann Machines (RBMs) [29], multi-layer sparse auto-encoders [30], [31] and deep convolutional neural networks (CNNs) [32] are driving various advances, especially in recognition. With the success of CNNs in large-scale image classification [33]–[35], it is also appealing to transfer a learned representation via fine tuning to other vision tasks [6]–[8].

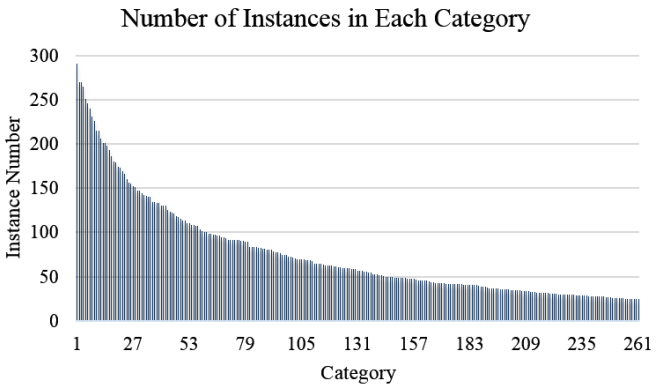


Fig. 3. The distribution of oracle character instances in Oracle-20K. Categories are ordered by number of instances.

III. The Oracle Dataset

As the earliest known Chinese inscriptions, oracle hieroglyphs were carved on animal bones or turtle shells to record divination results of various aspects of ancient China, such as fortune, state power, decision making, weather prediction, trading and warfare, during the late Shang dynasty (1250 BC – 1046 BC) [36].

Following their discovery in 1899, at the excavation site of Xiaotun in the city of Anyang, China, the last capital of the Shang dynasty, many oracle hieroglyphs were gradually unearthed through efforts by civil organizations and governments. Currently, approximately $160,000$ bone and turtle fragments have been found [36], including instances of approximately $4,500$ oracle characters [37]. Unfortunately, deciphering these characters is not an easy task. The real number of oracle characters is still unknown, and only some of their instances have been recognized.

We developed a crawling tool to collect oracle character instances and their labels from a website,[1] wherein there are $31,876$ instances of 952 individual characters obtained from a classical oracle character dictionary [38]. Due to the scarcity of resources, many characters only contain a small number of instances, despite the dictionary that the website relies on being one of the largest oracle character dictionaries in the world. To properly define the recognition task so that stable and comparable experiments can be conducted, we discarded characters with less than 25 instances. As a result, a dataset composed of $20,039$ oracle character instances belonging to 261 categories (unique characters) was obtained. The largest category consists of 291 instances, whereas the smallest categories contain 25. The distribution of instances is shown in Fig. 3. For convenience, we call this dataset Oracle-20K.

Fig. 4 shows example instances from Oracle-20K. We observe that shapes of oracle instances usually have high degrees of inner-class variance. This is because oracle

[1] http://www.chineseetymology.org/

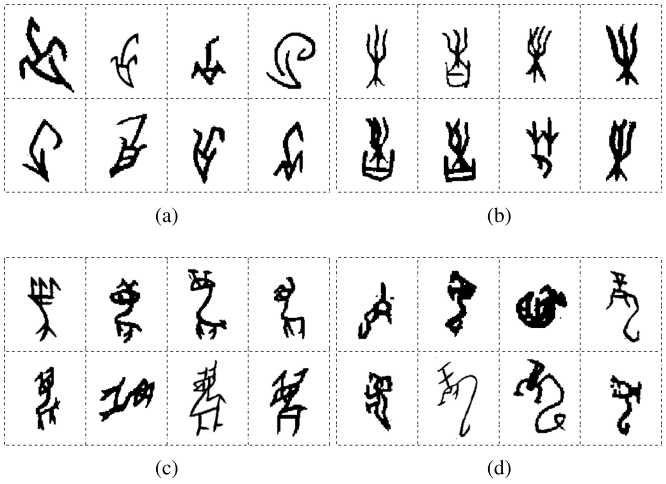


Fig. 4. Example instances in Oracle-20K. Instances from the same category are grouped together. The text below each group is the contemporary meaning of its instances. (a) Danger. (b) Spring. (c) Elk. (d) Dragon.

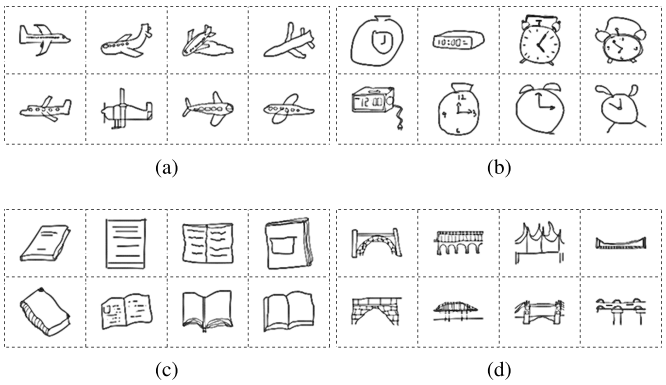


Fig. 5. Example instances in Sketch-20K. Instances from the same category are grouped together. The text below each group is the object label of its instances. (a) Plane. (b) Alarm Clock. (c) Book. (d) Bridge.

bones were produced over several historic periods in multiple regions and were carved by random ancients without much standardization [1]. Moreover, the engraving medium, i.e., animal bones and turtle shells, also imposed surface-related restrictions in terms of size and spatial arrangement.

IV. Evaluation of Visual Representations

This section provides evaluations of the eleven above-mentioned low-level representations and their combinations with the BoW framework. In all experiments, we used our Oracle-20K dataset and a general sketch dataset [5], Sketch-20K, as benchmarks. Sketch-20K was compiled by Eitz *et al.* [5] for sketch recognition and contains 20,000 hand-drawn sketches in 250 object categories, as shown in Fig. 5. This dataset was adopted to 1) improve the credibility of our evaluation and 2) verify the usefulness of oracle character analysis in general sketch-related research.

To achieve scale and translation invariance, we applied a preprocessing procedure to both datasets. First, each image was rescaled so that the longer side would be 224 pixels. Then, it was placed in the center of a blank image with a size of $256 * 256$ to produce a blank border with a width of at least 16 pixels. Such borders can avoid boundary effects that may occur during the execution of certain operations such as filtering. Finally, morphological thinning [39] was applied to ensure equal edge widths for different images.

Following the protocol of [5], the experiments were conducted using three-fold cross-test, and the mean classification accuracy was reported. Since the datasets were partitioned randomly, to obtain reliable results, we ran each experiment under several different partitions and recorded the *worst* mean classification accuracy. Considering the large number of experiments, to reduce computational costs while maintaining acceptable accuracy, we employed k-NN and logistic regression for the classification. k-NN was used due to the presence of non-vector-based representations; this method can simultaneously confirm the results of logistic regression. The performance of logistic regression is typically similar to that of linear SVMs. Hyperparameters of each representation and classifier are selected by cross-validation using the training set.

A. Comparisons of Low-Level Representations

Our evaluation results are presented in Table I. We did not report the classification accuracy of the logistic regression for CM, OCM, HIT, ESVM, and DVE because they only provide distances between dataset instances. Note that the rankings of performance over the two datasets are quite consistent:

1) When logistic regression is available (e.g., in recognition tasks), BiCE outperformed other low-level representations on both datasets, followed by FD-SIFT and HOG. In contrast, GIST, SC and HOOSC performed poorly, mainly suffering from their sampling strategy (which will be analyzed later).
2) Only considering k-NN, OCM and HIT outperformed other low-level representations on the Oracle-20K dataset. They also achieved acceptable performance on Sketch-20K, although DVE obtained the highest accuracy. CM generated almost the worst results because of its insufficiency in distinguishing edges of different orientations. ESVM was mediocre, consistent with [22], which reported unsatisfactory precision when applying ESVM to sketch-based image retrieval. This is reasonable because ESVM attempts to remove background, which is non-existent in oracle characters / sketches; in addition, is unable to address partial occlusions or significant deformations in queries [23].

In summary, these results demonstrate the success of BiCE and FD-SIFT when applied to oracle character / sketch recognition. Note that BiCE can be viewed as a set representation wherein values of 1 indicate the presence of edges; therefore, it can be used in conjunction with min-hash to achieve efficient searching.

In addition, note that retrieval tasks are strongly related to classification tasks using k-NN, and in particular, HIT is capable of quickly retrieving indexed images [19]. Therefore, HIT may be suitable for oracle character / sketch retrieval. DVE is also worth considering if the dataset is not excessively large.

TABLE I

MEAN CLASSIFICATION ACCURACIES ON ORACLE-20K AND SKETCH-20K. REPRESENTATIONS ENDING WITH "BOW" ARE QUANTIZED BY THE BOW FRAMEWORK WITH A KERNEL CODEBOOK [25]. D-GIST IS A VARIANT OF GIST BASED ON DENSE UNIFORM OVERLAPPING SUB-WINDOWS. D-SC AND D-HOOSC ARE VARIANTS OF SC AND HOOSC. THEY EXTRACT DESCRIPTORS ON DENSE UNIFORM SAMPLING POINTS. FS-SIFT IS A VARIANT OF FD-SIFT WHICH BUILDS DESCRIPTORS FROM ORACLE CHARACTER / SKETCH CONTOUR POINTS

Method	Oracle-20K		Sketch-20K	
	k-NN (%)	Logistic Regression (%)	k-NN (%)	Logistic Regression (%)
GIST [12]	69.7	71.2	39.8	42.2
SC [9]	64.0	73.3	34.7	42.5
HOOSC [10]	50.6	72.4	30.1	41.9
HOG [11]	70.4	74.2	39.9	45.8
BiCE [13]	75.1	**78.5**	44.0	**49.6**
FD-SIFT [5]	74.3	77.9	42.6	47.0
CM [17]	57.6	—	27.4	—
OCM [18]	**77.8**	—	43.3	—
HIT [19]	76.7	—	43.4	—
ESVM [22]	70.1	—	39.5	—
DVE [23]	75.4	—	**47.8**	—
SC + BoW(average pooling)	75.6	79.3	39.9	45.5
SC + BoW(max pooling)	60.4	80.4	32.9	49.3
HOOSC + BoW(average pooling)	68.8	77.2	37.5	45.7
HOOSC + BoW(max pooling)	48.7	76.8	27.6	48.3
HOG + BoW(average pooling)	75.2	78.6	43.7	47.6
HOG + BoW(max pooling)	71.6	80.6	35.1	**51.2**
BiCE + BoW(average pooling)	70.4	74.4	40.5	44.1
BiCE + BoW(max pooling)	72.3	75.1	42.1	46.8
FD-SIFT + BoW(average pooling)	**78.5**	**81.6**	**45.3**	50.1
FD-SIFT + BoW(max pooling)	76.9	80.5	42.8	50.3
D-GIST	78.0	78.2	44.7	46.2
D-SC + BoW(average pooling)	76.0	80.1	41.7	45.9
D-SC + BoW(max pooling)	66.0	81.2	39.5	50.0
D-HOOSC + BoW(average pooling)	69.6	77.5	38.3	46.2
D-HOOSC + BoW(max pooling)	59.1	77.1	32.2	49.2
FS-SIFT + BoW(average pooling)	75.7	78.5	36.6	41.8
FS-SIFT + BoW(max pooling)	74.1	77.9	33.3	40.2

B. Effectiveness of Mid-Level Representations

To investigate the effectiveness of mid-level representations in oracle character recognition, we fed local low-level representations, including SC, HOOSC, BiCE and FD-SIFT, into the famous BoW framework with a kernel codebook [25]. We also quantized HOG via BoW, regarding the histogram from each block as a local descriptor. For simplification, the term "+ BoW" appears as a postfix when a low-level representation is quantized by BoW, e.g., SC + BoW means SC quantized by BoW. The results in Table I show that BoW usually not only reduced the representation dimensions but also greatly improved classification performance, thus emphasizing the importance of building mid-level representations. One exception is BiCE – after being processed by BoW, the accuracy dropped a lot. This may be because BiCE is a binary representation, which does not span a continuous space, as required by BoW. We also note the effectiveness of FD-SIFT + BoW on Oracle-20K, although this method performed slightly worse than did HOG + BoW on Sketch-20K.

In addition, our evaluation shows that BoW with max pooling performed significantly better than with average pooling on Sketch-20K, which is consistent with the results of Boureau *et al.* [40]. However, these two pooling approaches performed comparably when applied to oracle character recognition. Such disparity implies unknown differences between oracle characters and natural images (on which Boureau et al. performed their study) / general sketches. Thus, domain-specific studies are required to better understand oracle character data.

C. Sampling Strategy

We have observed the mediocre performance of SC, HOOSC and GIST on both datasets. GIST relies on a set of non-overlapping sub-windows, thus resulting in sparse local descriptions. Moreover, SC and HOOSC sample local patches based on points from oracle character / sketch contours, which might make their local descriptors extracted from those local patches unstable to shape variation. We suspected that such sparse / uncertain sample strategies had negative effects on their performance.

To validate this hypothesis, we extracted GIST representations based on a set of dense uniform overlapping sub-windows and built local descriptors for SC and HOOSC on dense uniform sampling points, resulting in D-GIST, D-SC and D-HOOSC. Moreover, as a negative example, FD-SIFT was twisted to extract descriptors based on the points at contours, leading to FS-SIFT.

The new results are also shown in Table I. For local representations, including D-SC, D-HOOSC and FS-SIFT, we only report the results of their use in conjunction with the BoW framework due to the better obtained performances. As expected, D-GIST, D-SC + BoW and D-HOOSC + BoW outperformed their original versions, namely, GIST, SC + BoW and HOOSC + BoW, respectively, and FS-SIFT + BoW performed much worse than did FD-SIFT + BoW. Therefore, we conclude that, to achieve

good performances in oracle character / sketch recognition, dense uniform sampling is highly recommended.

V. The Proposed Representation

The aforementioned visual representations performed well in their specific domains; however, their performances remain unsatisfactory in oracle character and sketch recognition. We have experimentally studied and observed the superiority of FD-SIFT + BoW. This inspired us to develop a new low-level representation that strongly follows the skeleton of FD-SIFT, upon which we learn a more abstract representation. Based on these studies, we propose a hierarchical representation for oracle character and sketch recognition, which combines a carefully designed low-level representation and an effective mid-level representation.

First, a Gabor-based low-level representation that follows the skeleton of FD-SIFT but that addresses several important issues is introduced.

Next, we rely on a fast feed-forward network to learn sparse codes to build the mid-level representation, mainly following the multi-layer sparse auto-encoder [30], [31].

Finally, we construct a hierarchical representation based on the low-level and mid-level representations. The following are in contrast to the multi-layer (sparse) auto-encoder [30], [31]:

1) Due to restrictions concerning available oracle character/ sketch data, directly learning an auto-encoder on raw image data did not produce good results (see Sec. VI-B). Hence, our auto-encoder is built upon the proposed low-level representation.
2) The proposed hierarchical representation is composed of not only the highest level representation but also representations from lower levels so as to combine global structures and local details.
3) We do not directly rely on pooling approaches to generate the final representation. Instead, to reduce information loss during dimensional reduction resulting from pooling, we first perform quantization with multiple kernel codebooks. We also observe that quantization seems to generate a more robust final representation and thus improves performance (see Sec. VI-C).

In the following sub-sections, we will introduce our low-level and mid-level representations and the strategy used to combine them.

A. Low-Level Representation

We first introduce the skeleton of how to extract the proposed low-level representation, followed by some representation design criteria.

1) The Skeleton of Low-Level Representation: Similar to FD-SIFT, our low-level representation consists of multiple local descriptors. The process used to compute such descriptors is shown in Fig. 6. Given a shape image I as input, we first apply a bank of Gabor filters to extract edge information, resulting in R response images r_i (e.g., Fig. 6b):

$$r_i = I * g(\theta_i), \quad i = 1, 2, \ldots, R, \tag{1}$$

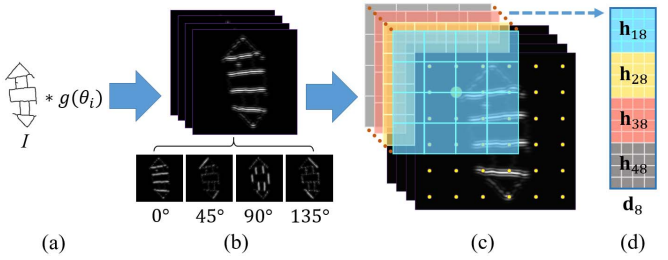


Fig. 6. Illustration of low-level descriptor extraction using 4 orientations and 36 uniformly sampled points. We first convolve the input image I with 4 Gabor filters (a), mapping I to 4 response images (b). These response images are then sampled and a partial descriptor is extracted at each sample point (c). Partial descriptors centered at the same spatial position but originating from different response images are finally concatenated into a complete descriptor (d). In this figure, partial descriptors $\mathbf{h}_{18}, \mathbf{h}_{28}, \mathbf{h}_{38}, \mathbf{h}_{48}$ centered at the 8-th sample point of each response image form the complete descriptor $\mathbf{d}_8$.

where $g(\theta_i)$ is the Gabor filter with orientation θ_i. In addition, a set of P points are selected via the dense uniform sampling (as analyzed in Sec. IV-C) of I, and their positions are mapped to each of the response images r_i so that all response images have the same set of sampled points.

Having both the set of points and the set of response images, a group of local patches centered at each of the sampled points are extracted from each response image. More precisely, centered at the sampled point p_j, a local patch l_{ij} is extracted from the response image r_i (e.g., Fig. 6c).

At this point, the input image I is represented by a set of $R \times P$ local patches. To generate a robust representation for each patch l_{ij}, we describe the patch by binning its Gabor responses into a histogram $\mathbf{h}_{ij}$ of $S \times S$ indexed spatial bins. To smooth the energy changes at bin boundaries, all Gabor responses are linearly interpolated into the neighboring bins.

Then, the corresponding R histograms at each point p_j (i.e., the histograms computed from each response image) are stacked together to generate a local descriptor $\mathbf{d}_j$:

$$\mathbf{d}_j = \left[\mathbf{h}_{1j}^T, \mathbf{h}_{2j}^T, \ldots, \mathbf{h}_{Rj}^T\right]^T. \tag{2}$$

Each local descriptor $\mathbf{d}_j$ is an $S \times S \times R$-dimensional vector, as shown in the example of Fig. 6d. This local descriptor is normalized such that $\|\mathbf{d}_j\|_2 = 1$.

In conclusion, the low-level representation of an image I is a set of local descriptors $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_P]^T$. In this work, the number of densely sampled points P, the orientation number in Gabor filter R, and the spatial binning parameter S are set to $P = 28 \times 28$, $R = 9$, and $S = 4$, respectively.

2) Low-Level Representation Design Criteria: Now, we discuss the design criteria for our low-level representation, including the approach to estimating local edge orientations and the scope of the local descriptors.

a) Local Edge Orientation Estimation: Edges are important for recognizing shapes. Estimation based on image gradients and direct detection are among the most frequently used approaches in computer vision to estimate local edge orientations. The former first computes image gradients $(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y})$ by convolving the image with anti-symmetric filters, such as Sobel filters or the first-order derivatives of a Gaussian (dG),

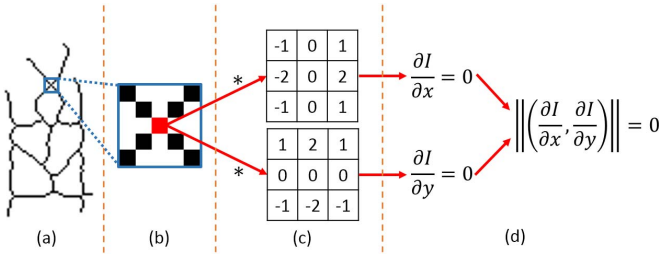


Fig. 7. Illustration of a symmetric pattern filtered by Sobel. For clarity, here, we use a thinned oracle instance (a). We observe that a symmetric pattern (surrounded by a blue rectangle) is present in this instance (b). The red point in this symmetric pattern, when convolved with two 2D Sobel filters (c), will have responses of $\frac{\partial I}{\partial x} = \frac{\partial I}{\partial y} = 0$, leading to zero gradient magnitude (d).

and then estimates local edge orientations by computing the ratio of $\frac{\partial I}{\partial y}$ and $\frac{\partial I}{\partial x}$. The latter directly detects local edge orientations by convolving the image with symmetric filters such as Gabor filters with zero phase, or the second-order derivatives of a Gaussian.

Although both approaches are widely used when addressing natural images, the former approach does not perform well in oracle character recognition because anti-symmetric filters do not effectively address edge junctions. Fig. 7 shows an example. In this figure, when convolved with a 2D Sobel filter, the magnitude of the gradient at the red point is zero, equal to that of a trivial point on a flat background, which is clearly unexpected. Similarly, convolving such a red point with a 1D anti-symmetric filter $[-1, 1]$ does not allow it to be differentiated from a trivial isolated point. Overall, when addressing binary images such as oracle characters, gradient-based approaches are not recommended for extracting edge information. FD-SIFT suffered from this issue in our analysis.

In contrast, symmetric filters are able to capture such patterns. These filters will produce a large response if an edge with the same orientation exists in both the image and the filter. Thus, as discussed in the last section, our low-level representation extracts edge information using a bank of Gabor filters at different orientations. In this work, we fix other parameters, i.e., scale, the wavelength of the sinusoidal factor, and the spatial aspect ratio of the Gabor filters, to 4, 10, and 1, respectively.

b) Scope of Local Descriptors: In contrast to natural images, where most pixels provide relevant information, oracle characters and sketches have a large number of pixels that correspond to empty background. Thus, large patches are required to avoid only capturing trivial local patterns such as small piece of lines, which are less expressive (see Fig. 8). However, this would cause spatial information loss when building descriptors for very large patches, which might lead to low discriminability. Thus, the low-level representation uses local patches covering at least one sixteenth and at most one sixth of the input image because we observed that these values provided good performances (see Sec. VI-A).

B. Mid-Level Representation

The mid-level representation is based on a multi-layer sparse auto-encoder [30], [31]. We first present the construction of the single-layer sparse auto-encoder, followed by the multi-layer version.

1) Single-Layer Sparse Auto-Encoder: Let $\mathbf{X}$ be a set of M input vectors in an N-dimensional vector space, i.e., $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_M]^T \in \mathbb{R}^{M \times N}$. The single-layer auto-encoder starts from an encoder that maps each input vector $\mathbf{x}_i$ to $\mathbf{y}_i$ under a transform $\mathcal{F}$: $\mathbf{y}_i = \mathcal{F}(\mathbf{x}_i)$. $\mathbf{y}_i$ is regarded as the code computed from $\mathbf{x}_i$. This encoder is accompanied by a decoder, which maps $\mathbf{y}_i$ back to the input space under another transform $\mathcal{G}$, thus producing a reconstruction: $\mathbf{r}_i = \mathcal{G}(\mathbf{y}_i)$.

The single-layer auto-encoder is parametrized through its encoder and decoder and attempts to reconstruct the original input as well as possible, i.e., it attempts to make $\mathbf{r}_i \approx \mathbf{x}_i$.

After applying the sparsity regularization to codes, the auto-encoder is forced to learn a sparse representation of the input. Such a sparse auto-encoder must reconstruct each input vector using a compacted representation of the input vector, thus encouraging the auto-encoder to discover unknown common patterns in $\mathbf{X}$. Therefore, with the sparsity regularization, the auto-encoder is able to build more abstract representations for the input.

In this work, both the encoder and decoder are defined as affine transforms followed by a nonlinear mapping:

$$\mathcal{F}(\mathbf{x}) = sig(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1), \tag{3}$$

$$\mathcal{G}(\mathbf{y}) = sig(\mathbf{W}_2\mathbf{y} + \mathbf{b}_2), \tag{4}$$

where $sig(\mathbf{x}) = 1/(1 + e^{-\mathbf{x}})$ is the sigmoid function.

We choose the negative log-likelihood as the measurement of reconstruction error, i.e., the loss arising from the use of $\mathbf{r}_i$ to represent the input vector $\mathbf{x}_i$. The negative log-likelihood between $\mathbf{x}$ and $\mathbf{r}$ is given by the following:

$$\mathcal{L}_{rec}(\mathbf{x}, \mathbf{r}) = -\sum_{k=1}^{N} x_k log(r_k) + (1 - x_k) log(1 - r_k), \tag{5}$$

where x_k and r_k are the k-th components of $\mathbf{x}$ and $\mathbf{r}$, respectively.

For the sparsity constraint, we do not employ the L1 penalty due to its high optimization cost. Instead, we utilize the KL divergence as in [41]:

$$\mathcal{L}_{spa}(\mathbf{y}) = \sum_{k} \rho log(\frac{\rho}{y_k}) + (1 - \rho) log(\frac{1 - \rho}{1 - y_k}), \tag{6}$$

where y_k is the k-th component of $\mathbf{y}$, and ρ is a small number close to 0. The above equation would encourage each component of $\mathbf{y}$ to approach ρ.

Finally, the single-layer sparse auto-encoder is trained to minimize the combination of the reconstruction error and the sparsity constraint over the entire dataset $\mathbf{X}$:

$$\min_{\Theta} \sum_{i=1}^{M} \mathcal{L}_{rec}(\mathbf{x}_i, \mathcal{G}(\mathcal{F}(\mathbf{x}_i))) + \lambda \mathcal{L}_{spa}(\mathcal{F}(\mathbf{x}_i)), \tag{7}$$

where $\Theta = \{\mathbf{W}_1, \mathbf{b}_1, \mathbf{W}_2, \mathbf{b}_2\}$. In this work, we set ρ and λ equal to 0.05 and 0.15, respectively, and minimize Eq. 7 via stochastic gradient descent with a gradually decreasing learning rate. More complex solvers, e.g., L-BFGS, were also attempted, but they produced very similar results.

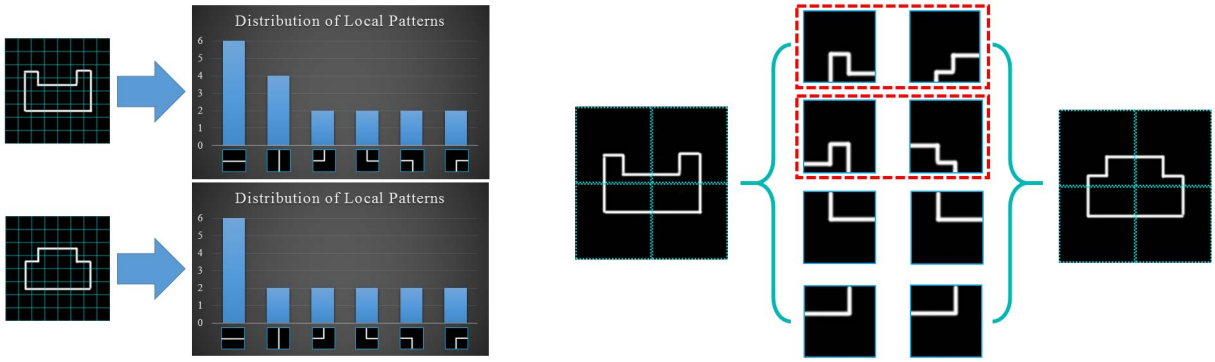


Fig. 8. Illustration of the effects of patch size. For convenience, here, we use non-overlapping patches and two (256 × 256)-pixel ideal instances. Fig. (a) shows that, when small patches are used, two instances cannot be well distinguished because their distributions of local patterns are similar, implying the triviality of patterns in patches. On the contrary, in Fig. (b), large patches cover more complex patterns, leading to a better discrimination because half the local patterns from the two instances are different. (a) Patch Size: 32 × 32. (b) Patch Size: 128 × 128.

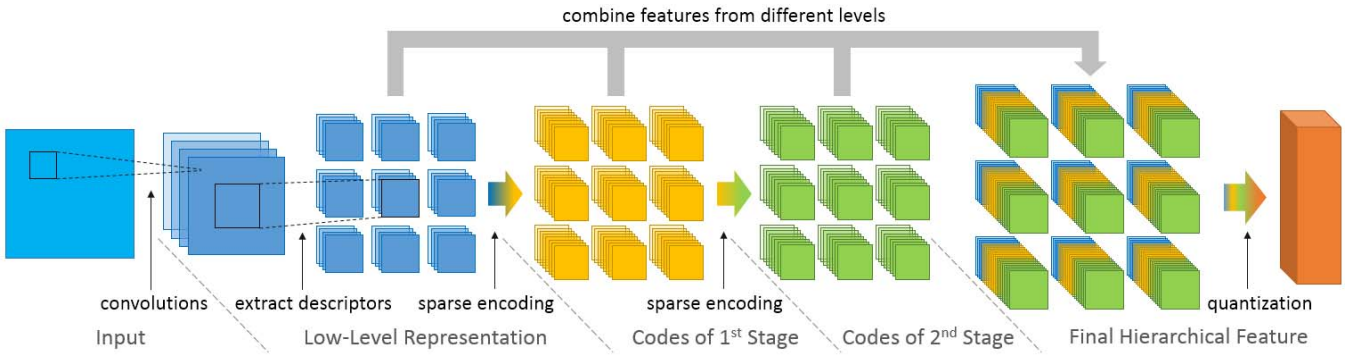


Fig. 9. Illustration of the proposed hierarchical representation. Given an input image, we first convolve it with a bank of Gabor filters and extract local descriptors on Gabor response images to build low-level representations (for clarity, we only show 9 descriptors). These descriptors are encoded by the sparse auto-encoder of the first layer, and the output vectors, i.e., the codes, are fed into the sparse auto-encoder of the second layer. The local descriptors and the output from all sparse auto-encoders are concatenated and quantized to build the final hierarchical representation. For clarity, please see the electronic version.

2) Multi-Layer Sparse Auto-Encoder: The construction of a multi-layer sparse auto-encoder is rather straightforward: each layer is separately trained using the outputs from the previous layer. More precisely, the multi-layer auto-encoder is built in a stacked fashion, therein consisting of multiple single-layer sparse auto-encoders. Each auto-encoder takes the representation learned from the previous layer (or the low-level representation for the first auto-encoder) as input.

C. Final Hierarchical Representation

The final representation is first constructed by combining low-level and mid-level representations and is then quantized using multiple kernel codebooks. Fig. 9 shows the entire pipeline.

1) Combining Multi-Level Representations: Given a (training) dataset, we first build low-level representations for all images and subsequently construct a multi-layer sparse auto-encoder over these low-level representations. In this way, an image I is first transformed into a representation consisting of a set of local descriptors $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_{28*28}]^T$. Then, these local descriptors are encoded by the trained multi-layer sparse auto-encoder layer by layer. In this work, we use a two-layer, 160-dimensional sparse auto-encoder; a descriptor $\mathbf{d}_i$ is first encoded by the auto-encoder of the first layer as $\mathbf{y^{(1)}}_i \in \mathbb{R}^{160}$ and then encoded by the second layer as $\mathbf{y^{(2)}}_i \in \mathbb{R}^{160}$.

Note that higher level representations tend to model global shapes and structures. Thus, we also branch outputs from lower levels into our hierarchical representation to capture local details. More precisely, we build the raw hierarchical representation by concatenating the low-level representation and outputs from sparse auto-encoders of all layers:

$$\mathbf{F} = [\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_{28*28}]^T, \tag{8}$$

where

$$\mathbf{f}_i = [\mathbf{d}_i^T, \mathbf{y^{(1)}}_i^T, \mathbf{y^{(2)}}_i^T]^T. \tag{9}$$

Such a raw representation is a high-dimensional representation. Thus, we feed it into the BoW framework with multiple kernel codebooks, which perform quantization and produce the final representation.

2) Multiple Kernel Codebooks: Again, let $\mathbf{X}$ be a set of M vectors in an N-dimensional vector space, i.e., $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_M]^T \in \mathbb{R}^{M \times N}$. The BoW framework trains over $\mathbf{X}$ to obtain a codebook $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_C]^T \in \mathbb{R}^{C \times N}$, where C is the number of visual words, and then encodes each

vector $\mathbf{x}_i$ as $\mathbf{v}_i$. A raw representation consisting of multiple vectors is finally quantized via average / maximum pooling on codes of these vectors.

For the BoW framework with a kernel codebook [25], $\mathbf{v}_i$ is calculated with a Gaussian kernel:

$$\mathbf{v}_i = [v_{i1}, v_{i2}, \ldots, v_{iC}]^T, \tag{10}$$

where

$$v_{ij} = \frac{exp(-\|\mathbf{x}_i - \mathbf{t}_j\|_2/2\sigma^2)}{\sum_{\mathbf{t}_k} exp(-\|\mathbf{x}_i - \mathbf{t}_k\|_2/2\sigma^2)}. \tag{11}$$

Note that after applying the BoW framework, each input vector $\mathbf{x}_i$ is reconstructed as $\mathbf{T}\mathbf{v}_i$, resulting in a residue $\mathbf{e}_i = \mathbf{x}_i - \mathbf{T}\mathbf{v}_i$. Inspired by the work of [42] and [43], we refine the BoW quantization by re-coding the residues of all input vectors, i.e., by reapplying BoW (with a kernel codebook) to the residues. Thus, an improved reconstruction of $\mathbf{x}_i$ is the concatenation of its code and the code of its residue. The quantization of a raw representation is correspondingly updated.

In this work, we only re-code residues once to balance classification accuracy with computational cost.

VI. Experiments

In Sec. IV, we presented the experimental results of several low-level visual representations and their combinations with BoW applied to oracle and sketch recognition. In this section we evaluate the proposed hierarchical representation as well as the effects of certain key factors in this representation. The experimental setup is similar to that in Sec. IV, i.e., each experiment is run several times and the *worst* mean classification accuracy of three-fold cross-test is reported. If not noted specifically, logistic regression is employed to ensure computational efficiency. We do not consider k-NN here because all representations involved in this section are vector-based representations; in this case, k-NN is always dominated by logistic regression.

A. Low-Level Representation

We first compare our low-level representation (LL) with FD-SIFT and HOG due to their outstanding performances when used in conjunction with BoW (see Sec. IV). BiCE is skipped because it could not match up with FD-SIFT + BoW or HOG + BoW, although it performs best in the absence of BoW. We also design three variants of LL by replacing Gabor filters with anti-symmetric filters, i.e., Sobel (LL(T1)), $[-1, 1]$ (LL(T2)) and $[-1, 0, 1]$ (LL(T3)), for comparison.

As shown in Table II, the proposed low-level representation significantly outperforms FD-SIFT, HOG, and other visual representations mentioned in Sec. IV. Moreover, after replacing Gabor filters with anti-symmetric filters, the performance of LL drops a lot. This result verifies the discussion about edge orientation estimation in Sec. V-A2.

Cross-validation shows that the optimal patch size for LL is one eighth of an input image. We vary the patch size to other scales, the results of which are shown in the last four rows of Table II. We can see that the performance of LL gradually decreases whenever the patch size increases or decreases. This result confirms the importance of using a large but suitable scope of descriptors as discussed in Sec. V-A2.

TABLE II

Performance of the Proposed Low-Level Representation and Other Visual Representations. "avg" and "max" Denote Average and Max Pooling. The Numbers After "LL" Illustrate the Size (the Percentage of an Image) of Local Patches. Numbers Followed by '*' Are Estimated by Cross-Validation. If Such Numbers Are Not Presented, the Optimal Size (Determined via Cross-Validation) Is Used

Method	Oracle-20K		Sketch-20K	
	avg (%)	max (%)	avg (%)	max (%)
FD-SIFT + BoW	81.6	80.5	50.1	50.3
HOG + BoW	78.6	80.6	47.6	51.2
LL[$\frac{1}{8}$*] + BoW	83.7	**84.1**	52.7	**56.9**
LL(T1) + BoW	81.1	79.6	50.3	50.4
LL(T2) + BoW	77.6	76.1	49.4	49.9
LL(T3) + BoW	78.1	76.4	50.5	50.6
LL[$\frac{1}{40}$] + BoW	80.7	81.3	49.9	53.7
LL[$\frac{1}{16}$] + BoW	82.4	83.2	52.3	55.2
LL[$\frac{1}{6}$] + BoW	83.5	84.0	52.5	56.1
LL[$\frac{1}{4}$] + BoW	81.8	82.5	51.3	54.5

B. Mid-Level and Final Representation

The proposed hierarchical representation is an extension of the multi-layer sparse auto-encoder, which is however different from the original version in the following three ways (see Sec. V-B): 1) the mid-level representation is based on the proposed low-level representation rather than the raw image, 2) the final representation includes information from both low-level and mid-level representations, and 3) multiple kernel codebooks are used for quantization.

Here, we evaluate the first two factors, leaving the last one for the next sub-section. We denote the multi-layer sparse auto-encoder with N layers as MA(N). MA(N)-Raw / MA(N)-LL represents the variants of MA(N) that take raw image data / our low-level representation as the initial input. In contrast to MA(N)-LL, which only employs outputs from the final layer for quantization, MA(N)-LL-All combines the low-level representation and outputs from each layer of the mid-level representation. To save space, we do not differentiate between average pooling and max pooling, i.e., only the best results are reported.

The first four rows of Table III clearly show the improvements obtained using the following:

1) Building mid-level representations upon manually designed low-level representations;
2) Branching outputs from lower levels into the final representation.

The first improvement is obtained by comparing MA(2)-Raw + BoW or MA(3)-Raw + BoW with MA(2)-LL + BoW. Note that MA(3)-Raw + BoW can be viewed as a variant of MA(2)-LL + BoW, which adds an extra layer at the bottom to learn low-level representations from raw image data, thereby playing a similar role as the manually designed low-level representation. The inferior performance of MA(3)-Raw + BoW demonstrates the

TABLE III

PERFORMANCE OF MID-LEVEL AND FINAL HIERARCHICAL REPRESENTATIONS. THE NUMBERS FOLLOWING "RAW" OR "LL" ARE THE SPARSITY LEVEL λS, AND THE NUMBERS FOLLOWING "BOW" INDICATE THE SIZES OF CODEBOOKS C. NUMBERS FOLLOWED BY '*' ARE ESTIMATED BY CROSS-VALIDATION. OPTIMAL PARAMETERS (DETERMINED BY CROSS-VALIDATION) WERE USED IF NOT SPECIFIED

Method	Oracle-20K	Sketch-20K
MA(2)-Raw[$\lambda = 1.0*$] + BoW	72.3	50.7
MA(3)-Raw[$\lambda = 1.6*$] + BoW	69.9	51.2
MA(2)-LL[$\lambda = 1.5*$] + BoW	85.3	57.5
MA(2)-LL-All + BoW[$C = 1500*$]	86.2	58.2
MA(2)-Raw[$\lambda = 0.0$] + BoW	71.0	49.2
MA(2)-Raw[$\lambda = 2.0$] + BoW	71.6	50.1
MA(3)-Raw[$\lambda = 0.0$] + BoW	67.8	48.4
MA(3)-Raw[$\lambda = 3.2$] + BoW	69.5	51.1
MA(2)-LL[$\lambda = 0.0$] + BoW	84.4	57.3
MA(2)-LL[$\lambda = 3.0$] + BoW	83.9	56.8
MA(2)-LL-All	80.5	53.1
MA(3)-LL-All	80.6	54.7
MA(2)-LL-All + BoW[$C = 3000$]	86.1	58.0
MA(2)-LL-All + BoW[$C = (1500, 1500)*$]	**87.1**	**59.4**

necessity of manually designed low-level representations when domain-specific data are scarce.

The second improvement could be observed after comparing MA(2)-LL + BoW with MA(2)-LL-All + BoW. Such an improvement, although not significant, illustrates the effect of combining local details with global structures.

In addition, we also investigate the influence of the sparsity level λ in Rows 5-10 of Table III. Compared with the results in Rows 1-3 of Table III, it can be observed that the sparsity term moderately affects the accuracy and tends to have a greater impact when the number of layers increases. In addition, for very small λ, MA seems to learn an identity transform and fails to discover salient patterns, whereas an excessively large λ leads to substantial information loss. In both cases, the performances are degraded.

C. BoW With Multiple Kernel Codebooks

We construct the final representation by quantizing outputs from MA(2)-LL-All using BoW with multiple kernel codebooks. Notice that BoW works by first learning the codebooks and then coding the outputs using pooling. To obtain a fair comparison, we also employ a variant by adding an additional layer on top of MA(2)-LL-All and then performing pooling on this layer, i.e., pooling on the outputs of MA(3)-LL-All. The additional layer serves as a replacement for BoW.

Let us compare MA(2)-LL-All + BoW (Row 4) with MA(2)-LL-All (Row 11) or MA(3)-LL-All (Row 12) in Table III. The results show that the representation quantized by BoW with a kernel codebook outperforms those directly originating from MA(2)-LL-All or MA(3)-LL-All. This phenomenon has resulted in an emphasis on discovering better coding schema and is consistent with the conclusion of Coates *et al.* [44].

We have shown that, with a 1500-word codebook, MA(2)-LL-All + BoW (Row 4) performed quite well. By conducting two additional experiments, i.e., MA(2)-LL-All-BoW using a 3000-word codebook / two 1500-word codebooks (one for quantizing the outputs of MA and the other for quantizing the residues), we also determined that, with multiple kernel codebooks, BoW (Row 14) performed better than did the single kernel codebook version (Row 4 & Row 13). More precisely, with two 1500-word codebooks (Row 14) BoW achieved a higher accuracy than when using one 1500-word codebook (Row 4) or even one 3000-word codebook (Row 13). This result proves that our multiple kernel codebooks can reduce information losses by learning codebooks on residues, thus increasing classification accuracy.

TABLE IV

COMPARISON WITH THE STATE OF THE ART (NO EXTERNAL DATA)

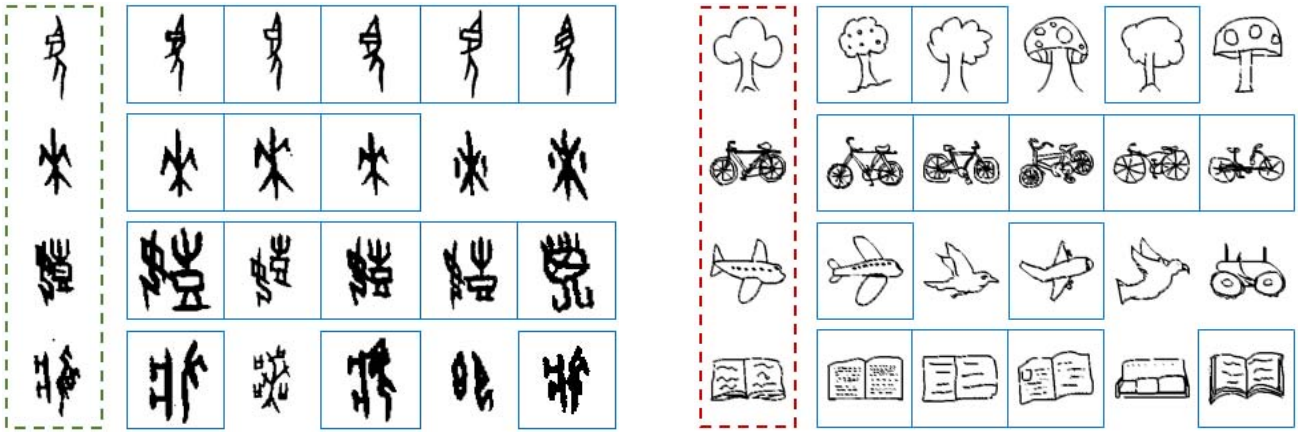
Method	Oracle-20K	Sketch-20K
Eitz	82.4	56.0
ScSPM-SIFT	82.0	53.3
ScSPM-LL	84.1	56.5
LeNet	79.9	54.3
TaoNet	81.7	55.9
Ours	**89.2**	**62.3**

D. Comparison With the State of the Art (No External Data)

To the best of our knowledge, the state of the art in general sketch recognition on Sketch-20K can achieve an accuracy of 56.0% by feeding FD-SIFT + BoW into a kernel SVM [5]. Moreover, our evaluation in Sec. IV confirmed the effectiveness of FD-SIFT + BoW on Oracle-20K. To further investigate our hierarchical representation, we also implemented other typical works in natural image recognition and shape analysis. We compared our representation with the following:

1) Eitz *et al.* [5] – the aforementioned state of the art in general sketch recognition.
2) ScSPM [26] – mid-level representations built by sparsely encoding SIFT representations. We also introduce a variant by building upon our proposed low-level representation. These two ScSPMs are denoted as ScSPM-SIFT and ScSPM-LL, and their outputs are classified by a linear SVM because we do not observe better results when a kernel SVM was applied.
3) LeNet [32] – a small convolutional neural network applied in digit recognition. We directly train the network on Oracle-20K / Sketch-20K and classify the output using an SVM with an RBF kernel (although not shown, using a linear SVM decreases the accuracies by approximately 0.5% on both datasets).
4) TaoNet [45] – applied in alphabet recognition. This method is similar to LeNet; however, the two convolutional layers are slightly larger, therein consisting of 96 8-by-8 and 256 2-by-2 filters. We also directly train this network and classify its output using an SVM with an RBF kernel (a linear SVM slightly decreases the performances by approximately 1% on both datasets).
5) Our method – feeding MA(2)-LL-ALL + BoW (1500 + 1500) into an SVM with a χ^2 kernel.

Row 2 of Table IV shows the unsatisfactory performances of the original ScSPM (ScSPM-SIFT). Replacing SIFT with our proposed low-level representation (ScSPM-LL) produces



(a) (b)

Fig. 10. The top five similar instances of four randomly selected queries for Oracle-20K and Sketch-20K using the proposed representation. The first column shows the query images. Relevant instances are enclosed by blue rectangles. (a) Oracle-20K. (b) Sketch-20K.

a large improvement and exceeds the state of the art, which again verifies the effectiveness of our low-level representation. Moreover, when processing a given test image, our approach does not need to solve an optimization problem and thus is much faster than ScSPM.

Due to the restriction on available domain data, only small CNNs can be directly trained without producing serious overfitting. However, it is difficult for small networks to detangle useful properties from input. As shown in Rows 4-5 of Table IV, directly applying LeNet or TaoNet to oracle character or sketch recognition does not seem to produce very good results. This phenomenon again encourages domain-specific studies when external data are inaccessible.

E. Comparison With the State of the Art (With External Data)

Recently it is believed that a representation learned from one dataset can capture underlying common factors, which may benefit tasks in other domains. Thus, when external data are available, reusing representations trained on large-scale external data may facilitate domain research wherein data are scarce such as research on oracle character recognition.

Here, we pre-train three complicated convolutional neural networks called AlexNet [33], VGGNet [34] and GoogLeNet [35] on ImageNet [46], a large dataset containing millions of natural images, and then fine tune the learned networks on Oracle-20K / Sketch-20K. Inspired by the success of ensemble [35], we also train 5 independent GoogLeNets differing in sampling methodologies and the order of input images. Their predictions are aggregated for testing (GoogLeNet-Ensemble). In addition, we attempt to combine the fine-tuned representation (i.e., outputs of the last but one fully connected layer) with our designed hierarchical representation by simply concatenating them (AlexNet + Ours, VGGNet + Ours, and GoogLeNet + Ours). The classification results based on logistic regression are reported in Table V. Our network implementations were based on Caffe [47].

TABLE V
COMPARISON WITH THE STATE OF THE ART (WITH EXTERNAL DATA)

Method	Oracle-20K	Sketch-20K
AlexNet	92.4	74.2
VGGNet	92.7	75.8
GoogLeNet	92.5	75.9
GoogLeNet-Ensemble	92.8	76.3
AlexNet + Ours	94.1	76.5
VGGNet + Ours	**94.2**	77.5
GoogleNet + Ours	**94.2**	**77.6**
Humans [5]	—	73.1

It is appealing that the recognition performance can be greatly increased using external data. Without significant manual exploration, fine-tuned AlexNet achieves an accuracy of greater than 92% in oracle character recognition and even outperforms human beings in sketch recognition, thereby demonstrating the power of transfer learning. More complicated networks, i.e., VGGNet and GoogLeNet generate better results on both datasets. The ensemble of 5 GoogLeNets also slightly improves the accuracy, but at the cost of much longer training / testing time (5 times compared with a single GoogLeNet). Nevertheless, incorporating the proposed representation with AlexNet, VGGNet or GoogLeNet can further improve the obtained performance by a moderate margin. This shows that our domain-specific work may complement models learned on external data.

F. Similar Oracle / Sketch Search

Fig. 10 shows the top five similar (measured by the proposed representation) instances of four randomly selected queries for each database. Relevant instances are enclosed in blue rectangles. We observe that, in the top results, most wrongly matched instances have shapes that are quite similar to the queries.

G. Oracle Character Analysis

In this section, we conduct an unsupervised intra-category analysis on Oracle-20K using the proposed representation. To reduce intra-category variance, we rotate the oracle characters according to their principle orientations in advance.

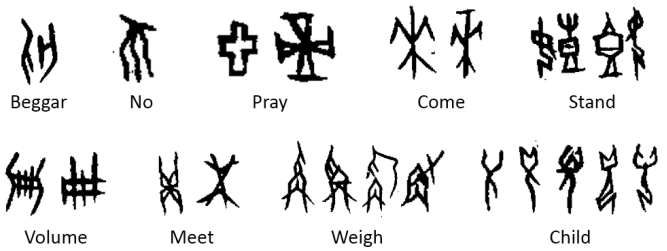


Fig. 11. Iconic representations of example categories of Oracle-20K. The text below each category is its contemporary meaning.

TABLE VI
THE CLUSTER NUMBERS FOR 5 SELECTED ORACLE CHARACTER CATEGORIES AND THE AVERAGE FOR ALL 105 CATEGORIES HAVING AGE INFORMATION. THE RESULTS ARE LISTED IN ORDER OF HISTORICAL PERIOD (FROM EARLIEST TO LATEST). THE NUMBERS OUTSIDE THE PARENTHESES ORIGINATE FROM MEAN-SHIFT CLUSTERING; THE NUMBERS INSIDE THE PARENTHESES WERE MANUALLY COUNTED

Period	Category					Overall
	Mid	Emperor	Sacrifice	Bare	If	Average
Zi	2(2)	4(5)	1(1)	2(2)	1(1)	2.1(2.0)
Bin	4(4)	4(4)	3(2)	5(5)	2(1)	3.4(3.0)
Li	4(4)	3(4)	2(2)	8(8)	1(2)	3.2(3.4)
Chu	1(1)	2(1)	2(1)	3(4)	1(1)	1.5(1.8)
He	1(1)	1(1)	1(1)	1(2)	2(2)	1.2(1.4)
Huang	1(1)	1(1)	2(2)	2(2)	1(1)	1.2(1.3)

1) Intra-Category Distribution: We are interested in the distribution of oracle characters in the proposed representation space. As mentioned in Sec. III, there is large diversity among oracle characters within the same category for historical reasons, implying that there might be more than one cluster within each category. To determine the cluster numbers, we separately apply adaptive mean-shift [48] to hierarchical representations of each category. The average number of clusters within each category is 2.6 (with our manual clustering result being 3.2), indicating that most categories distribute over several distinct clusters in the representation space.

To visualize such categories, we identify iconic characters that can be viewed as good representatives of a category based on the clustering results. The oracle character with minimum Euclidean distance to the cluster center is regarded as the iconic representative of that cluster. Several examples are shown in Fig. 11.

2) Analysis Over Historic Periods: The existence of multiple clusters within a category has aroused our interest in analyzing the intra-class variability over time in detail. In the field of archeology, ages of oracle characters are arranged into multiple periods such as Bin, Chu, and Huang [49]. Unfortunately, only a few characters (a total of 105 categories) in Oracle-20K were collected along with their age information. We apply the previous clustering algorithm to these characters and provide various results in Table VI. To confirm these results, we also manually count the cluster numbers and present them in the table.

First, we can observe that the first historical period (Zi) contains a moderate number of clusters for selected and overall characters, whereas the second and third periods (Bin and Li) exhibit the highest variabilities. Moreover, the distributions of the latest three periods (Chu, He and Huang) are quite concentrated. This analysis might suggest that, for a small set of our dataset, the visual layout underwent a standardization process across time (i.e., the average cluster number decreased over time). This initial result will be validated with more data in the future.

In addition, we observe that the mean-shift-generated results are quite similar to the counts mainly obtained by humans, implying the good separability of our representation. Note that not every character distributes across all historical periods, and the sets of iconic representatives for different periods are highly overlapped. Thus, the average number of clusters within each category is much smaller than the summation of the average cluster numbers of each period. We should also note that unsupervised clustering is somewhat subjective and that different people may have different opinions. Nevertheless, when a dataset is large and manual clustering is impossible, our representation is worth considering.

VII. POTENTIAL APPLICATIONS

In this section, we introduce several potential applications based on the proposed representation.

A. Recognizing Oracle Characters in Photos

Museum visitors are often amazed by these invaluable historical hieroglyphs. However, it is quite a pity that non-professional visitors are typically unable to understand such attractive ancient characters carved on cultural objects. In an attempt to solve this problem, a mobile App can be designed to detect and classify oracle characters in real time. When a visitor takes a photo of a historical hieroglyph, the following procedure is run:

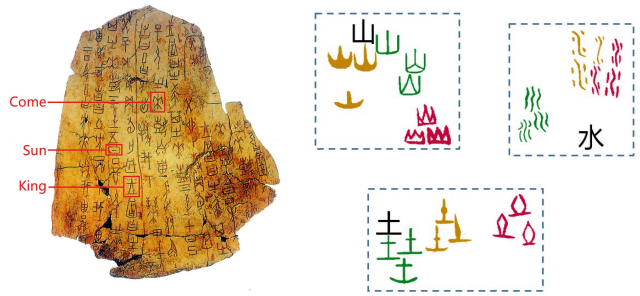
1) Use sliding windows of adaptive sides to detect oracle characters in the photo;
2) Extract the hierarchical representation for each detected oracle character;
3) Classify the character using a logistic regression classifier;
4) Display the predicted labels.

The entire procedure is fully feed-forward and does not need to solve any optimization problem on usage. Hence, the App can run very fast; see Fig. 12(a) for an example.

B. Exploring the Evolution of Chinese Characters

Chinese characters have experienced a long evolution. Started from oracle characters, Chinese characters evolved into bronze characters, seal characters, etc. sequentially. Currently, traditional Chinese characters and simplified Chinese characters are widely used in China.

However, discovering relations between characters from different historical periods remains challenging because many historical records have been lost over time. Inspired by the visual similarity between characters, we extend our hierarchical representation to address characters of all types. Given two



(a) (b)

Fig. 12. Applications enabled by the proposed representation. (a) Recognize oracle characters in a photo. For clarity, we only show three recognized characters. (b) Similarity of oracle characters (red), bronze characters (yellow), seal characters (green), and a traditional/simplified Chinese character (black). The Euclidean distance between two characters measures their similarity (the closer two characters are, the more similar they are). We bound characters with the same meaning using blue rectangles. The three groups are 'mountain', 'water', and 'soil'. Note that the high similarity between characters indicates an evolutionary relation.

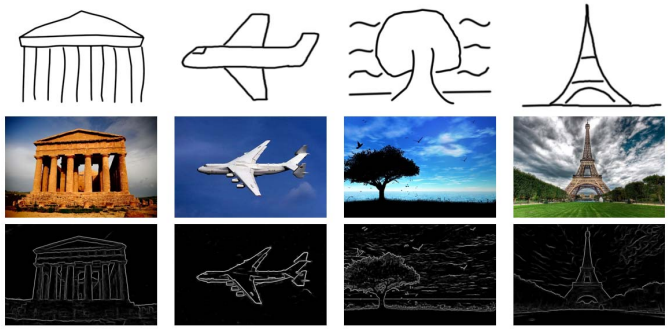


Fig. 13. Illustration of sketch-based image search. Each column contains a sketch query, the most similar image in the dataset, and the corresponding edge map.

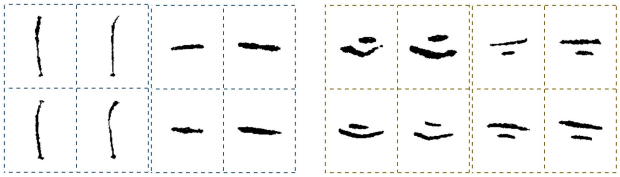
characters from different historical periods, if the distance between their representations is sufficiently small, this produces a hypothesis that one is likely to be evolved from the other. We hope such computational similarity can assist archaeologists; see Fig. 12(b) for an example.

C. Sketch-Based Image Search

We have demonstrated the effectiveness of the proposed representation in general sketch recognition. This inspires us to think about another potential application, i.e., sketch-based image retrieval [16], [19], as shown in Fig. 13.

A typical sketch-based image retrieval method consists of first extracting edge maps from images, followed by performing a sketch query to edge map matching supported by a well-designed indexing strategy [16], [19]. As shown in Fig. 13, the edge maps of top results are quite similar to users' sketch queries. For example, in the first column of Fig. 13, given a sketch query (Row 1), the system outputted an image of a temple containing several vertical pillars and a tent-like roof (Row 2), the edge map (Row 3) of which having much in common with the query. Hence, the edge maps can be considered as an intermediary to bridge the gap between natural images and binary sketches.

Based on the above concept, in the offline stage, the edge map [50] of each image in the database can be encoded by the proposed representation. In the online stage, users can draw a sketch of the expected images as the query. Then, the system extracts its hierarchical representation, compares with database images, and returns the most similar ones to users. Fig. 13 shows some preliminary results from a small dataset composed of thousands of images downloaded from the Internet. For larger datasets, an indexing technology must be leveraged to speed up the process. This will be our future work.



(a) Must (b) One (c) Up (d) Down

Fig. 14. Example instances that mainly differ in orientations: instances belonging to (b) appear similar to the 90-degree rotated versions of instances belonging to (a), and instances belonging to (d) seem to be the reflected versions of instances belonging to (c).

VIII. Discussions and Limitations

The representation proposed in this paper has produced promising results in not only oracle character recognition but also general sketch recognition. However, this method still suffers from certain limitations, which we will discuss below.

A. Other Representations

Our representation is built upon the famous bag-of-words framework, which does not encode any spatial layout information. To make use of such information, we have experimented with spatial pyramid matching [51] developed for natural image analysis but did not observe an obvious improvement. Moreover, our representation is currently based on the early work of sparse auto-encoders. We have also tried later works, such as contractive auto-encoders [52], but no attractive results were produced. Nevertheless, we have shown that, with a few twists, multi-layer sparse encoders could perform well when applied to oracle and sketch recognition. Thus, we believe that our hierarchical representation can be enhanced if we can find a better method of encoding spatial information. However, such representations might require priors from oracle characters, similar hieroglyphs or sketches, which would be different from those explored from natural images.

B. Rotation Invariance

In Sec. III, we demonstrated that certain oracle characters are similar to the rotated / reflected versions of other characters in the same category (see Fig. 4); thus, it seems that the classification accuracy can be increased if we can properly address rotation variance. Unfortunately, the performance of our method slightly decreased after rotating oracle characters relative to their principal orientations. This may be because some oracle characters belonging to different categories mainly differ in orientations (see Fig. 14). Rotating such characters to a specific orientation would harm the

discriminability. We will conduct additional experiments on this issue in the future.

IX. Conclusion and Future Work

In this work, computer vision techniques were leveraged to analyze oracle characters. Due to the unavailability of public oracle character data, we generated the Oracle-20K dataset, which is hoped to be helpful to the community in the future. Several popular shape- / sketch-related visual representations were studied and analyzed, based on which a novel hierarchical representation was proposed. We conducted extensive analysis on factors that might influence the proposed representation and compared it with related works. The experimental results demonstrated the superiority of the proposed representation over state-of-the-art representations on not only Oracle-20K but also Sketch-20K. This result verified the generalizability of the proposed representation on shape- / sketch-related works. We also showed that, when auxiliary data are available, the proposed representation can also be combined with CNN-based solutions to achieve a large performance improvement. In addition, we performed an unsupervised analysis on Oracle-20K and found that oracle characters tend to have less visual variability in subsequent periods. This trend will be validated on more data in the future.

We only explored mainstream low-level representations and certain simple mid-level representations; we leave the study of sophisticated representations and classification models as future work. We also would like to evaluate the proposed representation on more shape- / sketch-related problems.

References

[1] R. K. Flad, "Divination and power: A multi-regional view of the development of oracle bone divination in early China," *Current Anthropol.*, vol. 49, no. 3, pp. 403–437, 2008.

[2] D. N. Keightley, "Graphs, words, and meanings: Three reference works for Shang oracle-bone studies, with an excursus on the religious role of the day or sun," *J. Amer. Oriental Soc.*, vol. 117, no. 3, pp. 507–524, Jul./Sep. 1997.

[3] Y. X. Sui, *General Annotations of the Oracle-Bone Inscription in the Yin Dynasty Ruins*. Beijing, China: Zhonghua Book Company, 2011.

[4] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *Proc. 9th IEEE Int. Conf. Comput. Vis.*, Oct. 2003, pp. 1470–1477.

[5] M. Eitz, J. Hays, and M. Alexa, "How do humans sketch objects?" *ACM Trans. Graph.*, vol. 31, no. 4, p. 44, Jul. 2012.

[6] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 487–495.

[7] E. J. Crowley and A. Zisserman, "In search of art," in *Proc. Workshop Comput. Vis. Art Anal., ECCV*, 2014, pp. 54–70.

[8] S. Karayev *et al.* (2013). "Recognizing image style." [Online]. Available: http://arxiv.org/abs/1311.3715

[9] S. Belongie, J. Malik, and J. Puzicha, "Matching shapes," in *Proc. 8th IEEE Int. Conf. Comput. Vis. (ICCV)*, vol. 1. Jul. 2001, pp. 454–461.

[10] E. Roman-Rangel, C. Pallan, J.-M. Odobez, and D. Gatica-Perez, "Analyzing ancient maya glyph collections with contextual shape descriptors," *Int. J. Comput. Vis.*, vol. 94, no. 1, pp. 101–117, Aug. 2011.

[11] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1. Jun. 2005, pp. 886–893.

[12] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *Int. J. Comput. Vis.*, vol. 42, no. 3, pp. 145–175, May 2001.

[13] C. L. Zitnick, "Binary coherent edge descriptors," in *Computer Vision—ECCV*. New York, NY, USA: Springer-Verlag, 2010, pp. 170–182.

[14] E. Nowak, F. Jurie, and B. Triggs, "Sampling strategies for bag-of-features image classification," in *Computer Vision—ECCV*. New York, NY, USA: Springer-Verlag, 2006, pp. 490–503.

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

[16] M. Eitz, K. Hildebrand, T. Boubekeur, and M. Alexa, "Sketch-based image retrieval: Benchmark and bag-of-features descriptors," *IEEE Trans. Vis. Comput. Graphics*, vol. 17, no. 11, pp. 1624–1636, Nov. 2011.

[17] G. Borgefors, "Hierarchical chamfer matching: A parametric edge matching algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 10, no. 6, pp. 849–865, Nov. 1988.

[18] B. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla, "Model-based hand tracking using a hierarchical Bayesian filter," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 9, pp. 1372–1384, Sep. 2006.

[19] Y. Cao, C. Wang, L. Zhang, and L. Zhang, "Edgel index for large-scale sketch-based image search," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2011, pp. 761–768.

[20] Y. Cao, H. Wang, C. Wang, Z. Li, L. Zhang, and L. Zhang, "MindFinder: Interactive sketch-based image search on millions of images," in *Proc. Int. Conf. Multimedia*, 2010, pp. 1605–1608.

[21] X. Sun, C. Wang, C. Xu, and L. Zhang, "Indexing billions of images for sketch-based retrieval," in *Proc. 21st ACM Int. Conf. Multimedia*, 2013, pp. 233–242.

[22] A. Shrivastava, T. Malisiewicz, A. Gupta, and A. A. Efros, "Data-driven visual similarity for cross-domain image matching," *ACM Trans. Graph.*, vol. 30, no. 6, p. 154, Dec. 2011.

[23] M. Aubry, B. C. Russell, and J. Sivic, "Painting-to-3D model alignment via discriminative visual elements," *ACM Trans. Graph.*, vol. 33, no. 2, p. 14, Mar. 2014.

[24] E. J. Crowley and A. Zisserman, "The state of the art: Object retrieval in paintings using discriminative regions," in *Proc. BMVC*, 2014, pp. 1–12.

[25] J. C. van Gemert, J.-M. Geusebroek, C. J. Veenman, and A. W. M. Smeulders, "Kernel codebooks for scene categorization," in *Computer Vision—ECCV*. New York, NY, USA: Springer-Verlag, 2008, pp. 696–709.

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

[27] J. Yang, K. Yu, and T. Huang, "Supervised translation-invariant sparse coding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2010, pp. 3517–3524.

[28] J. Yang, Z. Wang, Z. Lin, S. Cohen, and T. Huang, "Coupled dictionary training for image super-resolution," *IEEE Trans. Image Process.*, vol. 21, no. 8, pp. 3467–3478, Aug. 2012.

[29] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, May 2006.

[30] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 19. 2007, p. 153.

[31] C. Poultney, S. Chopra, and Y. LeCun, "Efficient learning of sparse representations with an energy-based model," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 1137–1144.

[32] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[33] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[34] K. Simonyan and A. Zisserman. (2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: http://arxiv.org/abs/1409.1556

[35] C. Szegedy *et al.* (2014). "Going deeper with convolutions." [Online]. Available: http://arxiv.org/abs/1409.4842

[36] Z. Xu, K. K. C. Yau, and F. R. Stephenson, "Astronomical records on the Shang dynasty oracle bones," *Archaeoastronomy*, vol. 20, no. 14, pp. 61–72, 1989.

[37] C. Bazerman, *Handbook of Research on Writing: History, Society, School, Individual, Text*. Evanston, IL, USA: Routledge, 2010.

[38] X. Jin, *Xu Jia Gu Wen Bian*. Emerald Lake Hills, CA, USA: Yee Wen Publishing Co, 1993.

[39] L. Lam, S.-W. Lee, and C. Y. Suen, "Thinning methodologies—A comprehensive survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 9, pp. 869–885, Sep. 1992.

[40] Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce, "Learning mid-level features for recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2010, pp. 2559–2566.
[41] J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, Q. V. Le, and A. Y. Ng, "On optimization methods for deep learning," in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 265–272.
[42] H. Jegou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 1, pp. 117–128, Jan. 2011.
[43] H. Jégou, R. Tavenard, M. Douze, and L. Amsaleg, "Searching in one billion vectors: Re-rank with source coding," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2011, pp. 861–864.
[44] A. Coates, A. Y. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2011, pp. 215–223.
[45] T. Wang, D. J. Wu, A. Coates, and A. Y. Ng, "End-to-end text recognition with convolutional neural networks," in *Proc. 21st Int. Conf. Pattern Recognit.*, Nov. 2012, pp. 3304–3308.
[46] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2009, pp. 248–255.
[47] Y. Jia *et al.* (2014). "Caffe: Convolutional architecture for fast feature embedding." [Online]. Available: http://arxiv.org/abs/1408.5093
[48] B. Georgescu, I. Shimshoni, and P. Meer, "Mean shift based clustering in high dimensions: A texture classification example," in *Proc. 9th IEEE Int. Conf. Comput. Vis.*, Oct. 2003, pp. 456–463.
[49] X. Li, "Lun 'Fu Hao' Mu De Nian Dai Ji You Guan Wen Ti," *Cultural Relics*, vol. 11, pp. 32–37, 1977.
[50] P. Dollár and C. L. Zitnick, "Structured forests for fast edge detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2013, pp. 1841–1848.
[51] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2. 2006, pp. 2169–2178.
[52] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, "Contractive auto-encoders: Explicit invariance during feature extraction," in *Proc. 28th Int. Conf. Mach. Learn. (ICML)*, 2011, pp. 833–840.

Jun Guo received the B.S. degree from the School of Software, Sun Yat-sen University, Guangzhou, China, in 2013, where he is currently pursuing the Ph.D. degree with the School of Data and Computer Science. His research interests include computer vision and machine learning.

Changhu Wang (S'07–M'10–SM'14) received the B.E. and Ph.D. degrees from the University of Science and Technology of China, Hefei, China, in 2004 and 2009, respectively. Since 2009, he has been with Microsoft Research Asia, Beijing, China, where he is currently a Lead Researcher with the Multimedia Search and Mining Group. His research interests include multimedia retrieval and computer vision.

Edgar Roman-Rangel received the Ph.D. degree from the Ecole Polytechnique Federale de Lausanne, in 2013. He is currently a Post-Doctoral Researcher affiliated to the CVMLab with the University of Geneva, Switzerland. He is also a Visiting Researcher with the National Institute of Anthropology and History of Mexico, in the Tepalcatl project funded by the Swiss National Science Foundation. He likes working in real world application in areas of cultural heritage and digital arts. His research interests include computer vision, machine learning, and cross-modal information retrieval.He has served as part of the Technical Committee and Program Committee for several international conferences.

Hongyang Chao (M'06) received the B.S. and Ph.D. degrees in computational mathematics from Sun Yet-sen University, Guangzhou, China. In 1988, she joined the Department of Computer Science, Sun Yet-sen University, where she was initially an Assistant Professor and later became an Associate Professor. She is currently a Full Professor with the School of Data and Computer Science. She has published extensively in the area of image/video processing and holds 3 U.S. patents and four Chinese patents in the related area. Her current research interests include image and video processing, image and video compression, massive multimedia data analysis, and content-based image (video) retrieval.

Yong Rui (SM'04–F'10) received the B.E. degree from Southeast University, Dhaka, Bangladesh, in 1991, the M.S. degree from Tsinghua University, Beijing, China, in 1994, and the Ph.D. degree from the University of Illinois at Urbana–Champaign, Champaign, IL, USA, in 1999. He holds a Microsoft Leadership Training Certificate from the Wharton Business School, University of Pennsylvania, Philadelphia, PA, USA. He is currently an Assistant Managing Director of Microsoft Research Asia, Beijing.

He holds 56 issued U.S. and international patents. He has authored or co-authored 16 books and book chapters, and over 100 referred journal and conference papers. He is a fellow of IAPR and SPIE, and a Distinguished Scientist of ACM. He is an Executive Member of ACM SIGMM, and the Founding Chair of its China Chapter. He is the Editor-in-Chief of the *IEEE Multimedia Magazine*, an Associate Editor of the ACM TRANSACTIONS ON MULTIMEDIA COMPUTING, COMMUNICATIONS, AND APPLICATIONS, a Founding Editor of the *International Journal of Multimedia Information Retrieval*, and a Founding Associate Editor of the IEEE ACCESS. He was an Associate Editor of the IEEE TRANSACTIONS ON MULTIMEDIA (2004–2008), the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (2006–2010), the *ACM/Springer Multimedia Systems Journal* (2004–2006), and the *International Journal of Multimedia Tools and Applications* (2004–2006). He is an Organizing Committee Member and a Program Committee Member of numerous conferences, including ACM MM, ACM ICMR, the IEEE ICME, SPIE ITCom, and ICPR. He was the General Co-Chair of ACM MM in 2009 and 2014, ICMR in 2006 and 2012, and ICIMCS in 2010, and the Program Co-Chair of ACM MM in 2006, PCM in 2006, and ICME in 2009. He is/was on the Steering Committees of ACM MM, ICMR, ICME, and PCM.