Free-hand Sketch Recognition by Multi-Kernel Feature Learning

Yi Li[a,*], Timothy Hospedales[a], Yi-Zhe Song[a], Shaogang Gong[a]

[a] *Computer Vision Lab, School of Electronic Engineering and Computer Science, Queen Mary University of London, Mile End Road, London E1 4NS*

Abstract

Free-hand sketch recognition has become increasingly popular due to the recent expansion of portable touchscreen devices. However, the problem is non-trivial due to the complexity of internal structures that leads to intra-class variations, coupled with the sparsity in visual cues that results in inter-class ambiguities. In order to address the structural complexity, a novel structured representation for sketches is proposed to capture the holistic structure of a sketch. Moreover, to overcome the visual cue sparsity problem and therefore achieve state-of-the-art recognition performance, we propose a Multiple Kernel Learning (MKL) framework for sketch recognition, fusing several features common to sketches. We evaluate the performance of all the proposed techniques on the most diverse sketch dataset to date [1], and offer detailed and systematic analyses of the performance of different features and representations, including a breakdown by sketch-super-category. Finally, we investigate the use of attributes as a high-level feature for sketches and show how this complements low-level features for improving recognition performance under the MKL framework, and consequently explore novel applications such as attribute-based retrieval.

Keywords: sketch recognition, star graph, ensemble matching, multiple kernel learning, attributes

*Corresponding author

Email addresses: `yi.li@qmul.ac.uk` (Yi Li), `t.hospedales@qmul.ac.uk` (Timothy Hospedales), `yizhe.song@qmul.ac.uk` (Yi-Zhe Song), `s.gong@qmul.ac.uk` (Shaogang Gong)

1. Introduction

Throughout human civilization, sketch has been used as a basic form of communication. Examples of human sketches from ancient times can still be found in pre-historic cave art and pictograms nowadays. With the rapid emergence of portable touchscreen devices, sketches became much easier to obtain and are often a few finger sweeps away. This movement consequently led to an ever growing interest in free-hand sketch analysis from the computer vision community, where researchers investigated the feasibility of utilizing sketches in many novel tasks such as automatic sketch recognition [1] and sketch-based image retrieval (SBIR) [2, 3, 4, 5]. It has even been argued that sketches are more expressive than raw text when retrieving images [6, 3], and are able to capture visual memory of natural scenes [7].

Nevertheless, the task of automatically recognizing free-hand human sketches remains nontrivial, mainly due to the relatively large intra-class variations and inter-class ambiguities as opposed to images and other forms of sketches traditionally studied (e.g., CAD (Computer-Aided Design) drawings [8, 9]). More specifically: (i) sketches generally capture complex structures in abstract forms, a characteristic that is more evident in free-hand sketches where the depicting process is heavily unconstrained in terms of style and drawing ability; (ii) sketches, unlike conventional images, are naturally sparse in visual cues (e.g., without color and texture), this consequently makes applications of traditional image-oriented algorithms nontrivial. These unique properties of free-hand sketches ultimately render traditional shape/contour matching techniques inapplicable [10, 11]. Figure 1 offers a visual comparison of inputs used for shape matching (Figure 1(a)) and human free-hand sketches (Figure 1(b)). As can be seen, sketches are generally abstract, lack visual cues and it is often subtle internal structure differences that disambiguate one category from another. Meanwhile, differences between users in choices of abstraction and detail results in large intra-class variations.

Prior work on sketches typically addresses the feature sparsity challenge by

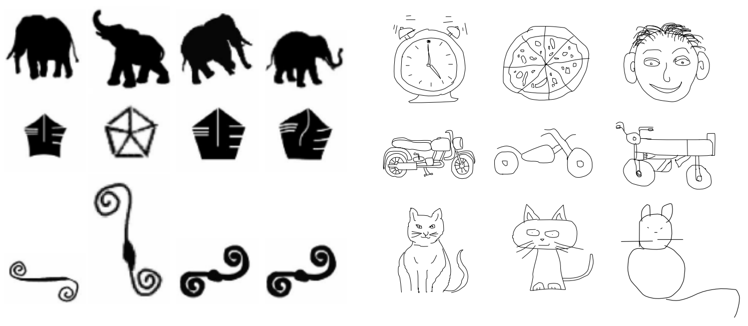


Figure 1: (a) Typical inputs of shape matching are generally silhouettes with quite simple internal structures [11]: three rows each corresponding to one object category; (b) Human sketches generally have more complicated internal structures. Human sketches often exhibit similar silhouettes, however it is the internal structures that make them different, e.g., the alarm clock, the pizza, and the face shown in the first row in Figure 1(b), and since people have different drawing styles, abstraction level and completeness of internal structures for the same object categories tend to differ, see motorbike and cat.

densely sampling on a grid [1, 4] or along the edges [2], or utilizing larger patches [1, 12]. And most sketch recognition [1] or SBIR methods [2, 3, 4, 5], uniformly employ a BoF (Bag-of-features) representation, in which holistic structure information is lost. Techniques taking account of structural information are commonly found in the image domain. Notable ones include for example, spatial pyramid matching [13] and spatial BoF [14]. However, these methods either use a rigid grid scheme [13] or presume that a dominant direction (either linear or circular) exists for the arrangement of the image features. But due to the abstractness, deformation and large variations of sketches, those schemes are not flexible enough to capture sketch structure. To the best of our knowledge, only a few methods [6, 15] employ a hierarchical topology graph to encode holistic structure of sketches. However, they are limited to working with CAD and clip-art [1] drawings which are clean and topologically separable, whereas sketch segmentation itself is an open problem under active research [16].

To address the complexity of internal structures, we propose a mid-level representation to capture the holistic structure of sketches. More specifically, we employ a star graph to encode both local features and holistic structure of a sketch and exploit ensemble matching as a similarity measure. A standard star graph, also known as an ensemble, has an arbitrarily assigned center with all the graph nodes connected to it by edges. The nodes represent feature points in the image and store the corresponding feature descriptors. A star graph encodes both direction and distance of each node to the center in the edges' weights. Detailed comparisons of different features and representations are performed on support vector machine (SVM) classifiers, and the results clearly show the advantage of the proposed star graph representation.

Furthermore, although different features or representations have different levels of performance, we argue that all features contain some potentially complementary information, at least for some classes, and should ideally be used together. We therefore address the cue sparsity problem via Multiple Kernel Learning (MKL), aiming at fully utilizing the discriminative power of all features and eliminating the both bias imposed by any single feature, as well as the design challenge of selecting the 'best' feature for an application. Our experiment on the human free-hand sketch dataset with the most categories to date [1] confirms state-of-the-art performance of MKL on sketch recognition over approaches [1, 17] employing BoF representation of popular features including Histogram of Oriented Gradient (HOG) [18], Self-Similarity (SSIM) [19] and Daisy [20], and the star graph representation constructed on HOG feature. Somewhat more subtly, but equally importantly, the same strategy addresses the open design challenge of deciding which similarity metric [5] to use in a given application.

The dataset [1] we evaluate on has as many as 250 categories. In order to show how different representations benefit certain categories more clearly, we introduce the concept of super-categories, which is defined as a superset of basic categories that share a higher-level semantic property (e.g., animal, plant). We found that although the star graph is generally best, different representations tend to favour different super-categories. By using all the features together, MKL obtains the best performance overall on all super-categories.

An interesting finding from the super-category analysis is that the confusions inside super-categories are much bigger than those between super-categories. This is especially true for large super-categories such as animal and vehicle. It is hypothesized that higher-level semantic properties shared amongst categories (e.g., spots on the body of a giraffe or butterfly) could help to remove ambiguity within a super-category – a hypothesis that was found to be successful in the image domain [21, 22, 23]. Inspired by [21], this work performs a preliminary study on how sketch attributes can benefit sketch recognition by constructing an attribute kernel within the MKL framework. The experiment is carried out on the animal super-category with classic animal attributes from [21] as well as additional attributes obtained from WordNet [24]. Experimental results show attributes to be effective in improving recognition performance inside super-categories.

Finally, going beyond simple recognition/retrieval of sketch categories, we show how the high-level semantic nature of attribute features can be used to enable novel applications. We demonstrate attribute-based retrieval (query by description rather than category; e.g., stripy), and joint category-attribute retrieval (find a long-leg ant, etc). The attribute classifiers can be further used to offer semantic-level rankings to sketches, for instance telling which zebra is stripier.

Our preliminary work [17] introduced the star graph: a spatially structured representation to model the structural complexity of sketches. By further combining with a category filtering step, it significantly improved the state-of-the-art sketch recognition performance [1]. This paper extends that work to further improve the sketch recognition performance and demonstrate some new interesting applications on sketches. More specifically: (i) we propose a MKL model to utilize multiple features, representations and similarity metrics, including star graph, to address the visual cue sparsity problem and surpass prior state-of-the-art performance; (ii) we demonstrate for the first time sketches attributes, and their value for both sketch recognition and enabling new retrieval applications.

2. Related work

Towards free-hand sketches There are several categories of sketches each possessing different sophistication levels and characteristics, they include CAD drawings [8, 9], artistic sketches (clip-art drawings) [25, 26], and free-hand human sketches [2, 3, 4]. CAD drawings are generated by designers using professional software, where standard building blocks are used to construct more sophisticated entities. As a result, CAD drawings often show clear topological structure, and the sophistication level varies from simple combinations of basic shapes to photo-realistic. Artistic sketches are another kind of sketch specifically produced by skilled artists. They tend to closely resemble the appearance of objects and exhibit low level of abstraction. Compared with artistic sketches, free-hand sketches refer to those drawn by non-artists using touch sensitive electronic devices, and are often highly abstract and exhibit large deformations. Shape matching is another related research topic [10, 11], yet significant differences exist compared to sketches. Most of them work on enclosed 2D outline contours of objects without internal structural details and/or with more consistent shape characteristics extracted from object images. On the other hand, free-hand sketches are relatively free (less regularized) with internal feature details of a sketch being important for discriminating sketches of different objects (Figure 1).

This research focuses on the free-hand sketches by using the largest free-hand sketch dataset to date [1], where inter-class ambiguities and intra-class variations commonly exist. It makes recognition on this dataset very challenging: the human recognition rate on this dataset is only 73.1% [1].

Local Features for Sketch Recognition Many local features commonly used in the image domain have been investigated for sketches. Eitz et al. [4] offer a detailed comparison of many popular features including Shape Context (SC), Spark feature, Histogram of Oriented Gradients (HOG) and sketched HOG (SHOG) by evaluating them on a SBIR system where BoF representation is employed. The outcome is that HOG based features generally outperform others, and the performance is sensitive to patch size and codebook size of the BoF representation. Very recently, a similar evaluation was carried out by Hu and Collomosse [5], in which several local features including gradient field HOG (GF-HOG), multi-resolution HOG (MR-HOG), Scale Invariant Feature Transform (SIFT), Self-Similarity (SSIM), Shape Context (SC) and Structure Tensor (ST) are investigated on a BoF based SBIR system. HOG based features again outperform other features. This is intuitive because HOG is a highly optimized feature descriptor for encoding edge and gradient properties of images, and human sketches' main property is just the edge and gradient.

Structured Feature Representation The concept of spatially structured feature representations is not new in the computer vision community. Many applications such as category recognition [13] and landmark images retrieval [14] have already proposed the general concept of a structured feature representation. Nevertheless, most of these structure encoding methods are quite specific to the problem domain they were designed to address, so are not directly applicable to the sketch representing problem. Many of them are designed for the image domain and work with BoF representations. For example, the spatial pyramid matching method [13] employs a series of grids over the image with increasingly coarse level. Then the representations at each grid level are summed up with an attached weight to form the final representation. It is designed for scene categorization and optimized for capturing frequently emerging local patterns in each scene category. However, this scheme is not effective for sketches, due to the large deformations and variations in highly abstract sketches resulting in weak structure information being captured by fixed-position cells. Another spatial BoF method [14] projects the 2D features onto certain lines or circles which are 1D space and then group the features by sectors in the 1D space. This concept of 1D encoding of local 2D features works well for landmark images where a dominant direction(s) may be readily obtained, but this property cannot be found generally in sketches.

Only a few works have proposed structured representation of sketches, in which topological relationships between sketch parts were utilized for improving matching accuracy [6, 15]. However, these methods are strictly restricted to some simple CAD and clip-art drawings and are thus not directly applicable to human sketches.

Attribute Learning Going beyond traditional structured and unstructured low-level features, attribute learning [21] has recently gained prominence in image [21, 22] and video [27, 22] recognition. Attributes aim to provide a powerful representation by computing a high-level semantic description of images. For example, bears have fur and claws, while zebras have fur and stripes. Computing this representation involves a category-level annotation of attribute properties, and an additional step of supervised learning where classifiers are trained to predict each attribute, after which the vector classifier posteriors for each attribute becomes the new representation for an instance. This is effective because the resulting representation is low-dimensional and discriminative by design, as human designed attributes are exactly those which humans use to distinguish categories. In this paper we investigate for the first time the use of attributes for sketch understanding. Not only do attributes provide a novel representation with which sketch categories can be distinguished, but this representation is synergistic with low-level features [27]. Moreover the semantic nature of attribute representation will allow novel tasks that go beyond sketch recognition, such as attribute-query and ranking.

Multiple Kernel Learning Previous studies either focus on one feature, e.g. HOG [1], or selecting the best performing feature [17] for sketch recognition. However, this ignores the potentially complementary cues contained in other features. SVM multiple-kernel learning (MKL) [28, 29, 30, 31] ([31] offers a good review on MKL) provides a route to discriminative recognition that can exploit multiple complementary features. MKL achieves state-of-the-art performance in a variety of vision areas [32, 31], for example: winning the PASCAL VOC 2009 object detection challenge by balancing dense and sparse textures and self-similarity; or color, shape and texture in recognizing flowers [32]. This is due to discriminatively learning how to weight features according to their informativeness. Moreover, they can automatically fuse multiple similarity metrics, which has also been a subject of comparative evaluation for sketches [5]. Recent MKL optimizers have improved computational efficiency [32], making them applicable for the large scale dataset [1] addressed here. We therefore go beyond existing work [1] and use MKL to discover not just the best single feature, but how each cue and similarity metric can be combined for best overall recognition performance.

3. Methodology

This section introduces the features, representations and classification models utilized for sketch recognition.

3.1. Features

3.1.1. ***Histogram of Oriented Gradients (HOG)***

HOG was first proposed by [18] for pedestrian detection. The gradients in each cell on a dense uniform grid are quantized into orientation bins that are then formed into a histogram. This feature is commonly reported to have best performance with sketches [4, 1, 2, 12].

3.1.2. ***Self-Similarity (SSIM)***

SSIM was proposed by [19]. For a feature point p, it compares a patch centered at p to nearby patches within a local region also centered at p, thus extract the "local self-similarity" for p. Then, the local SSIM descriptors are formed into a star graph model, and ensemble matching is employed to match the star graph models. SSIM has already been used on some very simple colored sketches [19], thus it is worthwhile to evaluate it with human sketches.

3.1.3. ***Daisy***

Daisy is based on histograms of gradients, like SIFT and GLOH [33], but utilizes a Gaussian weighting and circularly symmetrical kernel. It is very fast and efficient to compute densely [20]. Recent work of sketch tokens [34] has shown its effectiveness with sketches.

3.1.4. ***Attributes***

Unlike the previous features, the high-level attribute representation is itself the output of a supervised learning procedure. Attribute ground truth is defined by a binary class-attribute association matrix $\mathbf{A}$ (Figure 6), where each column specifies the attributes for that class. Given this matrix, a bank of M binary SVM attribute classifiers are independently trained to predict the presence or absence of each attribute. That is, for each attribute m, sketches from all categories with $a_m = 1$ are positive and sketches from categories with $a_m = 0$ are negative. The posterior $p(a_m|\mathbf{x})$ then reports the probability of a given sketch $\mathbf{x}$ having attribute m. The attribute representation of a sketch is then the M dimensional vector stacking the posterior probabilities for the presence of each attribute $A(\mathbf{x}) = [p(a_1|\mathbf{x}), \ldots, p(a_M|\mathbf{x})]$. Rather than utilizing these posteriors directly to predict the category as in [21], we use $A(\mathbf{x})$ as a new representation to be combined with the previous features by MKL. Details of SVM and MKL are described in Section 3.3.

3.2. Representations

3.2.1. ***Bag-of-features Representation***

Bag-of-features (BoF) [35] is used as the baseline for sketch recognition and multiple features are employed to evaluate its performance, including HOG, SSIM and Daisy. We apply normalization to all the sketches by scaling them into a fixed size. The features of a sketch are extracted on local patches. And the patches are centered in the intersections of a regular grid on top of the sketch. We use relatively large-sized patches, due to the limited information contained by the sketch, thus those patches have overlapping areas. To construct the BoF, we first collect a large set of n features by random sampling. Those n features are clustered into V clusters via k-means. The mean values of the clusters are used to form a visual codebook : $\mathcal{U} = \{\boldsymbol{u}_i\}_{i=1}^{V}$. After the codebook is obtained, a feature f is then represented by a vector of probabilities of f belonging to each word $\boldsymbol{u}_i$. The probability is calculated with a Gaussian kernel :

$$p(f, \boldsymbol{u}_i) = \exp(-\|f - \boldsymbol{u}_i\|^2/2\sigma^2) \tag{1}$$

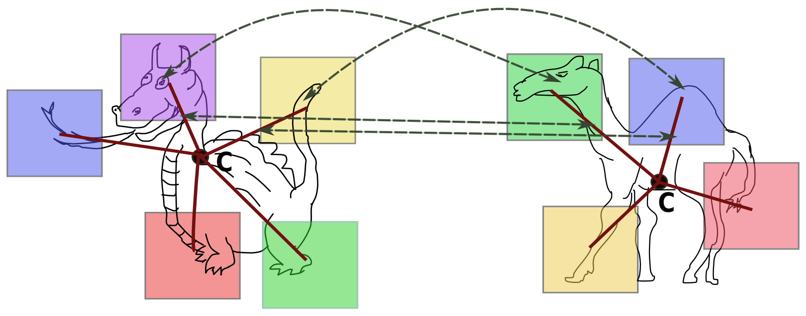


Figure 2: An illustration of star graphs for sketches and their ensemble matching. Patches are extracted from the sketch to construct the star graph, and each patch is connected to the graph center. The connections represent the patches' relative locations to the center. Different sketches can have different patch numbers. For two star graphs, both the patches and their corresponding relative locations are matched. Note that for illustrative clarity, only a few patches and matchings are shown for demonstration.

3.2.2. ***Star Graph and Ensemble Matching***

For the star graph representation, we apply the same normalization and grid as for BoF. The nodes of the graph are grid intersections close to the sketch strokes, so they can depict the structure of the sketch and different sketches have different numbers of nodes. In practice, we choose the grid intersections that have valid SSIM features, as they are just the intersections close to the strokes. Those grid intersections are applied to other features afterwards. The center of the star graph is the center of mass of those nodes.

We denote star graph as $G = (\mathcal{V}, \mathcal{E}, \mathcal{A})$, where $\mathcal{V}, \mathcal{E}, \mathcal{A}$ represent the graph nodes, edges and properties respectively. More specifically, $\mathcal{V} = \{v_i\}_{i=1}^{N_s} \cup c$ denotes all N_s sample points $\{v_i\}_{i=1}^{N_s}$ and the graph center c, while $e_i \in \mathcal{E}$ is the edge between v_i and c. Besides, $\boldsymbol{a}_{ic} \in \mathcal{A}$ is a relative location vector describing e_i, and $\boldsymbol{a}_i \in A$ denotes the feature descriptor corresponding to node v_i.

Ensemble Matching is the similarity metric employed here to compare star graphs. We formulate the similarity between two star graphs q (query) and t (target) as below:

$$P(G^q, G^t) = \prod_i P(\boldsymbol{a}_i^t | \boldsymbol{a}_i^q) P(\boldsymbol{a}_{ic}^t | \boldsymbol{a}_{ic}^q) \tag{2}$$

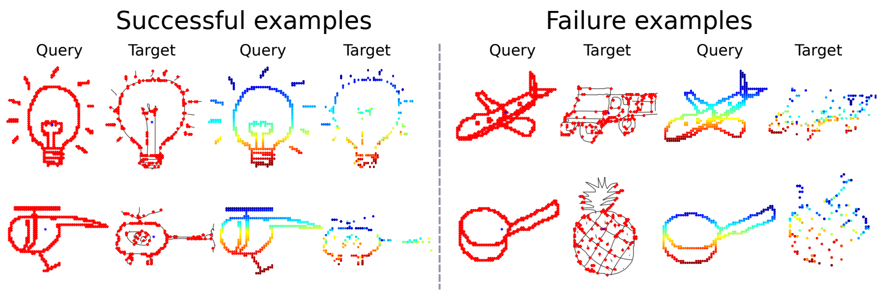


Figure 3: A visualization of ensemble matching. The left column includes successful examples and the right column includes failure cases. The image pair with red points indicates matched points. Multi-colored pairs indicate the detailed matching correspondence, where points with the same color are matched.

where $G^q = (\mathcal{V}^q, \mathcal{E}^q, \mathcal{A}^q)$ and $G^t = (\mathcal{V}^t, \mathcal{E}^t, \mathcal{A}^t)$ are the corresponding star graphs q and t. $P(\cdot,\cdot)$ represents the similarity. $P(\boldsymbol{a}_i^t|\boldsymbol{a}_i^q)$ calculates the similarity between feature descriptors $\boldsymbol{a}_i^t$ and $\boldsymbol{a}_i^q$ using a sigmoid function [19]:

$$P(\boldsymbol{a}_i^t|\boldsymbol{a}_i^q) = \frac{1}{1 + \|\boldsymbol{a}_i^t - \boldsymbol{a}_i^q\|_1}. \tag{3}$$

$P(\boldsymbol{a}_{ic}^t|\boldsymbol{a}_{ic}^q)$ computes the similarity of relative location vectors $\boldsymbol{a}_{ic}^t$ and $\boldsymbol{a}_{ic}^q$ using a Gaussian function [36] :

$$P(\boldsymbol{a}_{ic}^t|\boldsymbol{a}_{ic}^q) = \exp(-(\boldsymbol{a}_{ic}^t - \boldsymbol{a}_{ic}^q)^T S_L^{-1} (\boldsymbol{a}_{ic}^t - \boldsymbol{a}_{ic}^q)) \tag{4}$$

where S_L is a covariance matrix to allow for some deviations in the node locations. Figure 2 illustrates the nodes and edges of the star graph and the ensemble matching concept.

We modify traditional ensemble matching in several ways to accelerate the matching process and improve the matching performance. First, a two step algorithm is employed to find the best match in the target for each node in the query. Multiple nodes in the query are allowed to match to the same node in the target, which we found to be better than a one-to-one matching policy. For a query node, the algorithm first finds the most similar D target features $\{\boldsymbol{a}_j^t\}_{j=1}^D$ in terms of feature descriptors (D is much smaller than the total feature amount in the target, and is set as 20 in our experiments) among all the nodes by exhaustive searching. Then, it calculates the location correlations only for these D features and it selects the node having the maximum overall similarity score. Second, to penalize the points not matched in the target, a penalty factor w is added which is defined as the proportion of the matched points in the target. Third, to generate the overall matching score above each node's matching score, the product rule employed in [36] is replaced with the sum rule which is proved to be the most resilient to estimation errors [37]. The sum is normalized by the number of nodes N_s in the query star graph. The new function for ensemble matching is then:

$$P(G^q, G^t) = w * \frac{\sum_{i \in N_s} \max_j P(\boldsymbol{a}_j^t | \boldsymbol{a}_i^q) P(\boldsymbol{a}_{jc}^t | \boldsymbol{a}_{ic}^q)}{N_s}. \tag{5}$$

Using Equation 5, the matching scores from G^q to G^t and from G^t to G^q are often different. To improve the stability of the final score, we average the scores for both directions:

$$P^f(G^q, G^t) = (P(G^q, G^t) + P(G^t, G^q))/2 \tag{6}$$

It is worth noting that if considered as a kernel function, Equation 6 is symmetric, and empirically we found it was always positive semi-definite in each cross-validation fold of our experiments. Several detailed examples of ensemble matching are shown in Figure 3. Only those points having both similar features and similar locations will be matched. It can be seen that ensemble matching addresses the holistic structure similarity well in the successful cases, and finds similar object parts in terms of structure in the failure cases.

3.3. Classification methods

3.3.1. ***Support Vector Machines***

The SVM classifiers are trained for each category to classify sketches. For category l with $\boldsymbol{x}$ being the sketch representation, the score function used to decide the class of a query $\boldsymbol{x}$ is:

$$c^l(\boldsymbol{x}) = \sum_{s=1}^{S} \alpha_s^l K(\boldsymbol{x}_s^l, \boldsymbol{x}) + b^l \tag{7}$$

where α_s are the coefficients, b is the bias, K is a kernel function and s indexes support vectors $\boldsymbol{x}_s$. The response $c^l(\boldsymbol{x})$ measures how likely the query belongs to the lth category.

RBF kernel is used for K in the case of BoF representation:

$$K(\boldsymbol{x}_s^l, \boldsymbol{x}) = \exp(-\gamma \|\boldsymbol{x}_s^l - \boldsymbol{x}\|_2^2). \tag{8}$$

In the star graph case, the RBF kernel is replaced with Equation 6:

$$K(\boldsymbol{x}_s^l, \boldsymbol{x}) = P^f(\boldsymbol{x}_s^l, \boldsymbol{x}). \tag{9}$$

And one-vs-all approach is employed for the multiclass classification task.

3.3.2. ***Multiple Kernel Learning***

Different features and representations have varying values for each category. In conventional SVMs, the kernel is defined on one feature type. Some features are more informative, but each feature may provide some complementary information. A weighted sum of kernels is therefore desirable to best utilize the discriminative power of each feature and representation. If we have a few kernels : $K_1, K_2, ...K_M$, using the same notation as Section 3.3.1, their weighted linear combination is then formulated as:

$$K(\boldsymbol{x}_s^l, \boldsymbol{x}) = \sum_{m=1}^{M} \beta_m K_m(\boldsymbol{x}_s^l, \boldsymbol{x}) \tag{10}$$

where $\{\beta_m\}_{m=1}^M$ are weights reflecting the contribution of each kernel. The classifier score function is then:

$$f^l(\boldsymbol{x}) = \sum_{m=1}^{M} \beta_m c_m^l(\boldsymbol{x}). \tag{11}$$

$c_m(\boldsymbol{x})$ are the score functions for m different features and defined in Equation 7. MKL is used to select the inter-kernel weights $\{\beta_m\}_{m=1}^M$ and the coefficients $\{\alpha_s\}$ for each feature kernel which maximize the accuracy using Equation (11).

We use four kinds of kernels as described below. Besides ensemble matching, each kernel is applied to all BoF features. And the dimension of $\boldsymbol{x}$ is t.

1. Linear kernel

$$K(\boldsymbol{x}_s^l, \boldsymbol{x}) = \langle \boldsymbol{x}_s^l, \boldsymbol{x} \rangle \tag{12}$$

2. RBF kernel, described in Equation 8.

3. Chi square kernel

$$K(\boldsymbol{x}_s^l, \boldsymbol{x}) = \sum_t \frac{2x_{st}^l x_t}{x_{st}^l + x_t} \tag{13}$$

4. Histogram intersection kernel

$$K(\boldsymbol{x}_s^l, \boldsymbol{x}) = \sum_t \min(x_{st}^l, x_t) \tag{14}$$

5. Ensemble matching kernel, described in Equation 9 (for star graph only).

4. Experiments

This sections first evaluates SVM's performance with different features and representations. Then a thorough evaluation of fusing the features and representations using a MKL model follows, which delivers the state-of-the-art performance. Finally, the attribute experiments and applications are demonstrated.

4.1. Dataset and general settings

Dataset We evaluate our methods on the sketch dataset with the most categories to date proposed by Eitz et al. [1], which has 250 categories and 20,000 sketches (80 sketches per category). The dataset was collected on Amazon Mechanical Turk from 1,350 non-expert subjects, thus the drawing style and sophistication level are diverse. Following [1], the sketches are normalized to 256×256 pixels.

Super-categories To define super-categories, we refer to WordNet [24]. The original category names is used to search their inherited hypernyms in WordNet. Then a few hypernyms that are representative and intuitive are chosen to be the super-categories' names, and each of them is used to group several original categories. Finally, 14 super-categories are defined to analyze different representations' performances. The specific super-categories' names and the number of categories in each are shown in Table 3.

Features Three basic types of features are evaluated in the proposed methods, including HOG, SSIM and Daisy. SSIM is computed with VGG's implementation [38]. The 'var_noise' parameter is set to 50,000, and the radical bins and angular bins are set to 5 and 12 respectively. On top of the 256×256 pixels sketch, a 51×51 grid is used to extract the sample points, and the local patch size is 90×90. VGG's saliency checking, homogeneity checking and second-nearest neighbor checking are all disabled, as they are not suitable for sketches. A customized homogeneity checking is utilized to keep all the sample points along the sketch contours and these sample points are also used for other features. HOG is computed using the VLFeat [39] implementation with each patch divided into 4×4 cells and the orientation is set to 4. Daisy is computed with CVLAB's implementation [20] with all the default settings as well.

Bag-of-features representation For all the mentioned features, a codebook of 1,000 visual words are used to obtain the BoF representation. 1,000,000 features are randomly sampled to generate the codebook via k-means clustering. The σ parameter for the Gaussian kernel is searched between [0.001,1].

Star graph representation As described in Section 3.2.2, SSIM is used to decide which grid intersections are used to construct the star graph, and other features will adopt these intersections. The center of the star graph is the center of mass of these intersections.

Parameter searching and training data size We employ 4-fold cross validation scheme to search for parameters. For both SVM and MKL, the γ and C parameters are searched between $[2^{-2}, 2^{8}]$. A coarse grid search is performed with an interval of 2^2 to find a best value K, followed by a fine grid search with an interval of $2^{0.25}$ among $[2^{-1}K, 2K]$. All the 80 sketches in each category are used for cross validation, and we report the averaged accuracy on all 4 folds, following [1]'s practice. In Section 4.2, to evaluate the impact of training dataset size, the dataset is also separated into growing subsets (i.e., 20,40,60,80 sketches per category), and on each of the subset, the averaged 4-fold cross validation accuracy is reported. The attribute experiment employs a slightly different training/testing setting and is explained in more details in Section 4.4.

Support Vector Machines and Multiple Kernel Learning For single-kernel experiments we use the libsvm optimiser [40], and for multi-kernel experiments we use the UFOMKL optimiser [32].

4.2. Comparing different features' performance on SVM

We compare the BoF representation of HOG, SSIM and Daisy, and the star graph representation on HOG (due to the computational cost of ensemble matching, we just select HOG to work with star graph as it is the reported best performing feature), with SVM classifiers. Figure 4 shows their performance with incrementally increased training set size. Table 1 shows the recognition accuracies of each feature when using the full training set. It can be seen that star graph representation performs better than BoF representation, and HOG is still the best performing feature.

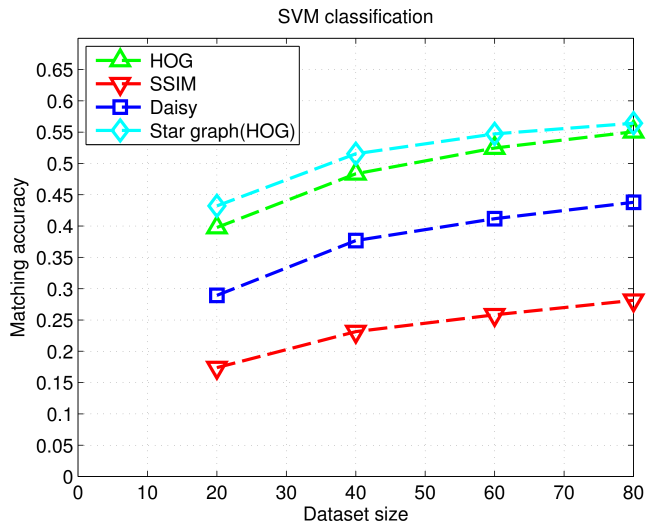


Figure 4: Performance comparison for varying data volume.

Table 1: Classification performance of different features with SVM using the full training set.

	HOG	SSIM	Daisy	Star graph(HOG)
Acc.	55.12%	27.99%	43.80%	**56.42%**

4.3. Fusing Features and Similarity Metrics using Multiple Kernel Learning

Given the varying informativeness of each feature on BoF, and the super-category analysis showing the variable effect of HOG and star graph, we next investigate whether MKL can fuse these features in a complementary way. We train an RBF kernel MKL classifier with three features including BoF representation of HOG, SSIM and Daisy. The star graph kernel is also included and computed with Equ. 9 . With the complementary cues of multiple representations on multiple features, recognition performance reaches 62.61% (RBF in Table 2(c)). Additionally, to show that each feature has contributed to the overall result, we computed the MKL results without one feature at a time. The results are shown in Table 2(a). We also show the weight β_m from Equ. 11 in Table 2(b) to help illustrating each feature's contribution, and the weights are generally consistent with the accuracy in Table 2(a), highlighting the contribution of star graph and HOG.

Beyond feature type, a pervasive design question in conventional sketch recognition is what is the right similarity metric to use for comparing images. Within the MKL framework, this question can be sidestepped as all similarity metrics can be used together synergistically. To demonstrate this, we further evaluated 3 additional kernel functions beyond RBF used thus far: linear, chi square (Chi2) and histogram intersection (HI) on all the features (star graph kernel is always included when using each kernel function, computed with Equ. 9). The performance of all kernel functions is shown in Table 2(c) with HI kernel yielding the highest accuracy of 65.45%. Then, we compute all the kernels for each feature, and use them all in MKL, which yields an even better result of 65.81% (also shown in Table 2(c)). This performance significantly exceeds the state-of-the-art performances in [17] and in [1], which are compared in Table 2(d). Importantly, these experiments show that not only does using all kernels and all features yield the best performance, but that tuning the choice of features and kernels [4, 12] is not necessary – the simple strategy of using them all together is best.

To provide a complete analysis on feature fusion, we also included results of

Table 2: (a) Recognition accuracy of MKL using all the features but excluding one feature each time to see the contribution of each feature (on RBF kernel). (b) The weight β_m (c.f. Equ. 11) is also shown for each feature to illustrate its relative importance in MKL (on RBF kernel). (c) The accuracies of low-, mid-, high-level fusions of the features with different kernel functions. The mid-level fusion (MKL) with all the features and all the kernel functions yields the best performance. (d) The performance comparison with previous works. The standard errors are also provided for all the accuracies when available.

(a)

Excluded	*HOG*	*SSIM*	*Daisy*	*Star graph(HOG)*
Acc.	$58.85 \pm 0.11\%$	$62.01 \pm 0.28\%$	$60.86 \pm 0.29\%$	$60.46 \pm 0.28\%$

(b)

Feature	*HOG*	*SSIM*	*Daisy*	*Star graph(HOG)*
Weight	0.0054	0.0047	0.0043	0.0098

(c)

Kernel	RBF	Chi2	HI	Linear	All
Mid-level Acc.	$62.61 \pm 0.34\%$	$63.78 \pm 0.48\%$	$65.45 \pm 0.61\%$	$55.09 \pm 0.45\%$	$65.81 \pm 0.58\%$
Low-level Acc.	$61.20 \pm 0.44\%$	$63.45 \pm 0.45\%$	$64.82 \pm 0.59\%$	$57.41 \pm 0.42\%$	$64.38 \pm 0.48\%$
High-level Acc.	$61.48 \pm 0.31\%$	$56.75 \pm 0.31\%$	$63.74 \pm 0.49\%$	$60.90 \pm 0.16\%$	$56.14 \pm 0.25\%$

(d)

Methods	SVM [1]	2-step [17]	MKL(All)
Acc.	56%	61.5%	**65.81%**

two alternative strategies for fusion: a) "low-level" feature stacking, and b) the "high-level" classifier voting. Our MKL method is referred to as the mid-level fusion, because it learns weights for the similarity metrics. For low-level fusion, we concatenate the BoF of the 3 basic features and use the chosen kernel function to obtain the kernel matrix, which is then averaged with the star graph kernel (computed with Equ. 9)) without weighting. This averaged kernel matrix is then used in SVM for classification. For the high-level fusion, we make a kernel for each feature with the chosen kernel function (star graph kernel is computed with Equ. 9) and train an SVM classifier for each of them. The output of this bank of SVMs is combined with majority voting to obtain the final classification

Table 3: Comparison of SVM recognition performance grouped by super-category, using BoF and Star graph (Star) on HOG . The number of categories in each super-category is in parentheses. MKL results are also stated.

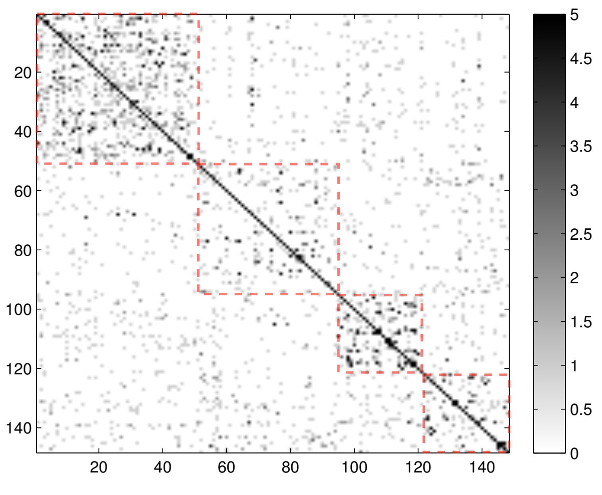
	Animal(48)	Plant(18)	Vehicle(27)	Electrics(27)	Clothing(8)
BoF	43.01%	61.62%	51.06%	55.42%	65.94%
Star	45.31%	63.97%	53.06%	58.38%	74.38%
MKL	53.47%	73.01%	58.94%	63.84%	75.78%
	Furniture(14)	Body part(20)	Building(10)	Sport(6)	Food(9)
BoF	47.32%	63.69%	53.63%	61.25%	59.86%
Star	50.63%	68.19%	57.75%	62.92%	56.67%
MKL	58.21%	73.25%	66.75%	71.46%	70.42%
	Instrument(7)	Commodity(45)	Weapon(6)	Nature(5)	
BoF	57.14%	59.97%	58.33%	78.75%	
Star	58.39%	56.42%	61.88%	67.25%	
MKL	68.04%	68.86%	70.00%	89.25%	

result. Those results are also shown in Table 2(c).

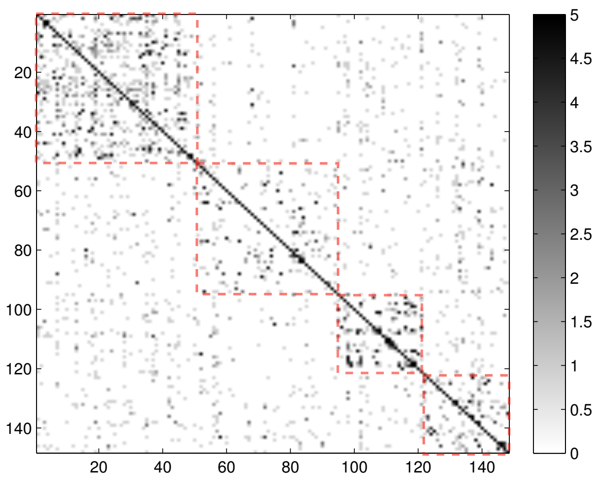
To offer insight into what types of sketches each representation is better at, the per super-category performance of SVM on star graph and BoF is provided in Table 3. Although for the overall result, star graph is only slightly better, star graph is evidently better at 11 super-categories, while BoF is better at only 3 super-categories. The per super-category performance of MKL is also shown in Table 3. After employing both representations, MKL achieves the best of both, with top results on every super-category. For the super-category analysis, we also show the confusion matrices of the 4 biggest super-categories (animal, commodity, vehicle and electrical_device) in Figure 5. It can be seen that the inside super-category confusions are much bigger than the between super-category confusions, especially for the animal and the vehicle super-categories, as the categories inside these two super-categories have more similar topology structures.

4.4. Attributes for Classification

To see how attributes can help improving the recognition inside a super-category, we pick the animal super-category to perform a preliminary experi-



(a)



(b)

Figure 5: The confusion matrix of BoF and Star graph on HOG for 4 major super-categories : animal, commodity, vehicle, electrical_device. The matrices are sorted by category. Red dotted rectangles highlight within-category versus across-category confusion. The matrices are exaggerated via mapping values from 1 to 5 to the whole color range so numbers above 5 are shown the same color as 5 and small values are more clearly observed. The confusions inside most super-categories are much higher than the confusions between super-categories.

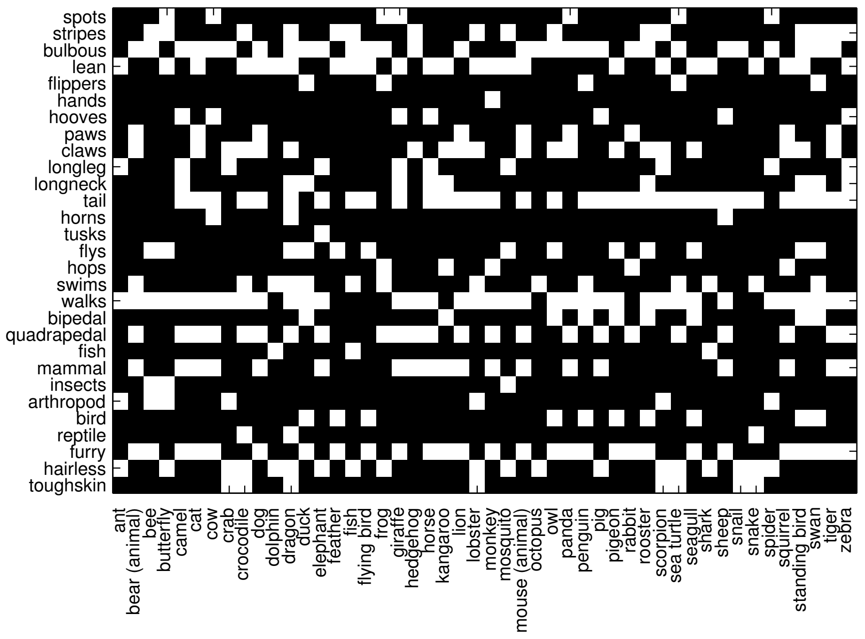


Figure 6: The ground-truth class-attribute matrix used in our experiments. X axis (horizontal) indicates categories and the Y axis (vertical) indicates attributes (The attributes are borrowed from [21]).

ment. We borrowed some attributes from [21] and defined a few more attributes by searching the category names' inherited hypernyms in WordNet. Finally we selected 29 attributes for the animal set. The category/attribute table is shown in Figure 6.

To demonstrate the contribution of the attributes, we use the best MKL result with all the features as the baseline, and compare with the MKL result when adding the attribute feature. We also use SVM to test how attributes perform alone. A different evaluation scheme is adopted compared to the previous sections, as two loops of training are needed for a fair comparison: the attribute classifiers and the MKL/SVM classifiers. We divide the 80 sketches of each category into 2 subsets : s_1, s_2, with s_1 for training and testing attribute classifiers on HOG, s_2 for training and testing the MKL/SVM classifiers. On s_1, for each attribute classifier, we select its parameters γ and C by 4-fold cross validation

Table 4: (a) The classification accuracies of the attribute classifiers. (b) The comparison of recognition accuracies by using MKL on all the features, MKL on all the features with attributes and SVM on attributes.

(a)

Attribute	spots	stripes	bulbous	lean	flippers	hands	hooves	paws
Accuracy	86.30%	74.84%	71.93%	69.58%	91.04%	97.92%	91.67%	84.17%
Attribute	longneck	tail	horns	tusks	flys	hops	swims	walks
Accuracy	86.04%	79.48%	94.48%	98.59%	81.46%	90.78%	77.81%	85.94%
Attribute	fish	mammal	insects	arthropod	bird	reptile	furry	hairless
Accuracy	96.15%	82.40%	95.36%	88.91%	88.54%	95.47%	76.20%	76.20%
Attribute	claws	longleg	bipedal	quadrapedal	toughskin			
Accuracy	72.81%	84.64%	88.75%	80.00%	87.76%			

(b)

	MKL(All)&Attributes	MKL(All)	Attributes
Accuracy	**52.39%**	50.63%	36.77%

also among $[2^{-2}, 2^8]$. When the attribute classifiers are obtained, they are used to compute the attribute features for s_2. Then s_2 is used to train and test the MKL/SVM classifiers through the same type of 4-fold cross validation. s_1 and s_2 are both set 40 in our experiment.

The recognition rate of each attribute classifier is shown in Table 4(a) and demonstrates that, despite the sparsity of features available in sketches, attributes are quite reliably detected. Table 4(b) offers a comparison of the recognition accuracy of MKL without attributes, MKL with attributes, and SVM solely with attributes. Evidently attributes can further improve the recognition of sketches. This is because they provide a representation which is discriminative by design – highlighting individual semantic properties that are useful for distinguishing categories. It is thus reasonable to expect that attribute definitions for other super-categories besides animals should also provide solid improvements in results.

A prominent character of an attribute representation is that when training data is limited, it can obtain better performance compared to other features

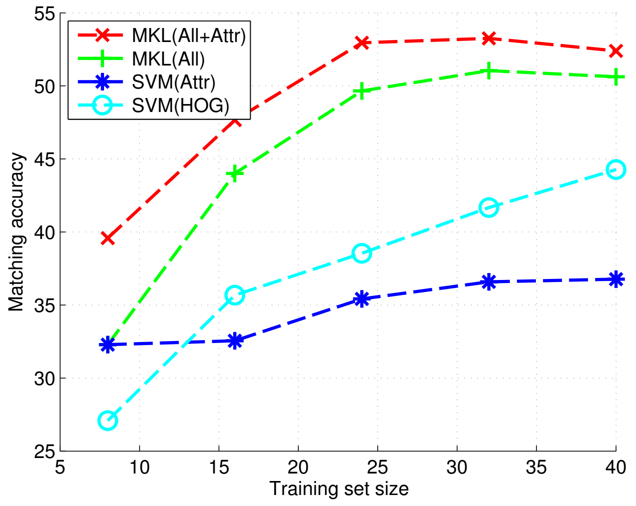


Figure 7: Matching accuracy versus training size for SVM on HOG (SVM(HOG)), SVM on attributes (SVM(Attr)), MKL on all the previous features (MKL(All)) and MKL(All) with attributes (MKL(All+Attr)).

due to their low-dimensional representation and sharing of statistical strength across attributes [27, 22]. To investigate this, we increase the training data size of s_2 from 8 to 40 with an interval of 8 to show how the training size affects the accuracy of attributes versus MKL trained with the other previous low-level features. We also offer HOG result alone here, as attributes are computed from HOG only and thus attributes versus HOG provides a direct comparison of the representation versus the feature. The result is shown in Figure 7. Clearly attributes noticeably outperform HOG in the very low-training data regime. However, with additional data HOG eventually outperforms attributes. This is due to enough data obtained to learn the higher dimensional HOG data; versus the eventual saturation of attribute performance due to imperfection in attribute detection. However interestingly, the attributes consistently provide a complementary cue to all the other low-level features as MKL(All+Attr) is consistently better than MKL(All), especially in the low-data regime.

4.5. Applications using Attributes

A possible interesting future direction for sketch attributes is to provide the opportunity for novel sketch-understanding based applications. A first application is to allow the user to retrieve sketches by attribute rather than by category, by sorting sketches via attribute classifier rather than category classifier (e.g., spotty or stripy). The first results in the sorted list possess the attribute with high probability and the last few results possess the attribute with low probability (or equivalently the inverse attribute with high probability, e.g., long legs versus short legs). Figure 8 illustrates this for three attributes in the top row of each section. A second application is to allow the user to retrieve sketches based on a combination of category and attribute. There are various potential ways to achieve this, but we illustrate the concept by querying the category first and then sorting by attributes. In the second row of each section in Figure 8 we show the results of sorting attributes within ground truth categories (thus separating categorization errors from attribute-sorting errors). In the third rows, we show the results for a fully automated query which retrieves the top 20 confident sketches for the specified category, and then sort these by the attribute scores. In each case, both the top 5 results and bottom 5 results for each category are shown to illustrate the contrast.

5. Conclusions

The high internal structure complexity and lack of visual cues, are the two major challenges for sketch recognition. In this work, we propose a star graph representation that captures both the holistic structure and local features to address the internal structural complexity problem. To further account for the lack of visual cue problem, we employ a MKL framework that fuses several popular features known to work with sketches. Extensive experiments on the most diverse free-hand human sketches to date show significant improvement over the state-of-the-art, from 61.5% to 65.81% (human accuracy being 73.1%). Very recently, [41] demonstrated that Fisher Vectors, an advanced feature representa-

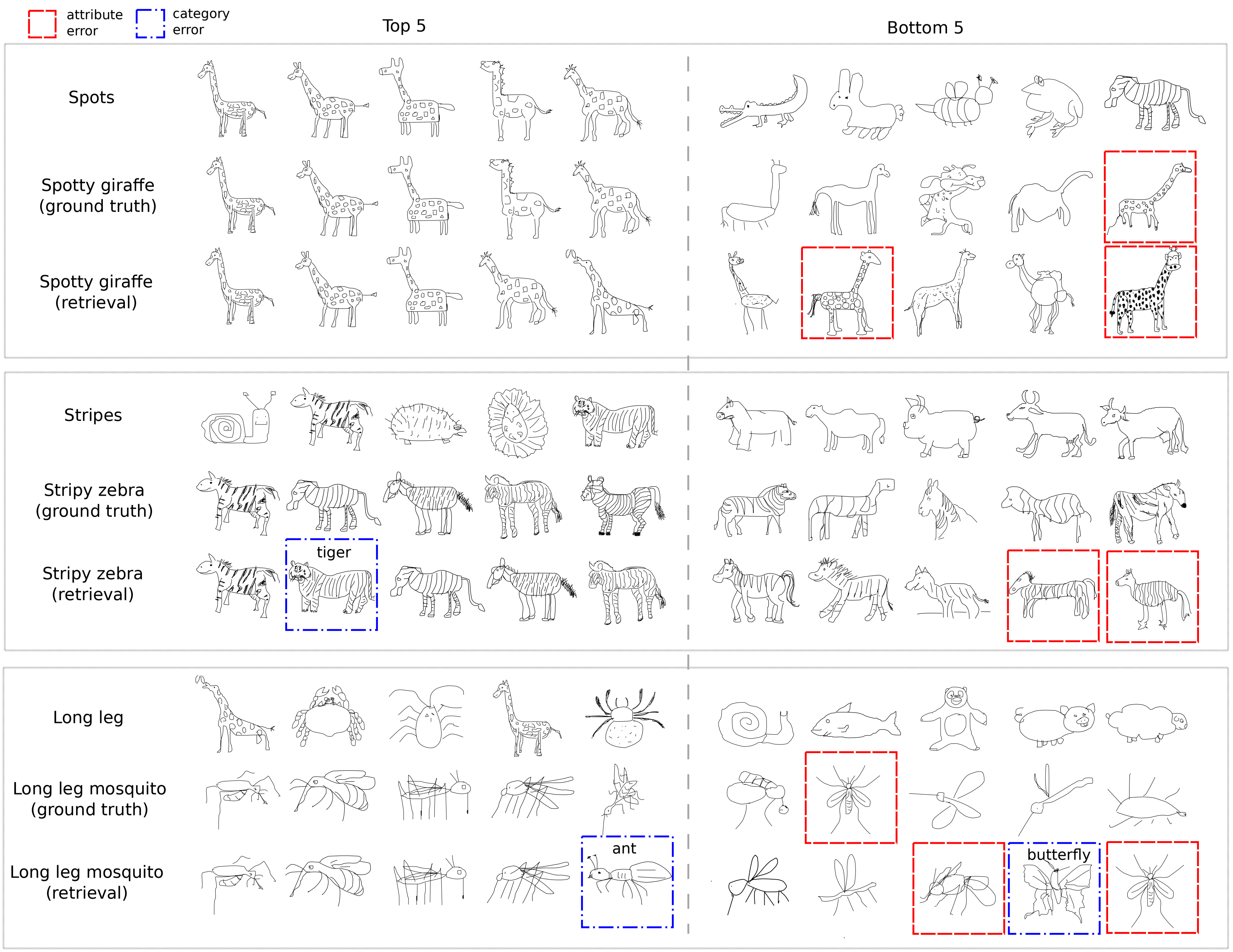

attribute error
category error
Top 5
Bottom 5
Spots
Spotty giraffe (ground truth)
Spotty giraffe (retrieval)
Stripes
Stripy zebra (ground truth)
Stripy zebra (retrieval)
tiger
Long leg
Long leg mosquito (ground truth)
Long leg mosquito (retrieval)
ant
butterfly

Figure 8: The results of unconditional attribute query (top rows), class-attribute query with ground truth class (middle rows) and automatically recognised class (bottom rows). Both of the top 5 and last 5 results are selected here to demonstrate the contrast. Rectangles refer to mistakes.

tion scheme successfully applied to image recognition, can be adapted to sketch recognition and achieve near-human accuracy (68.9%). In comparison to our method, Fisher Vector has the disadvantage of much higher memory footprint due to its high dimensionality. However, it is worth noting that the proposed MKL framework can also embed Fisher Vector in place of BoF. Over and above that, we for the first time study attributes for sketches, and demonstrate their effectiveness in reducing confusion inside one super-category. Moreover, we show how the high-level semantic nature of the attribute feature allows novel applications such as query by attribute or class-attribute description. In the future, increasing matching efficiency of the proposed structured representation while keeping the recognition performance by adjusting the structure scheme, and applying uniform attributes to all the super-categories are promising directions to proceed on sketches.

References

[1] E. Mathias, H. James, A. Marc, How do humans sketch objects?, ACM TOG (Proceedings SIGGRAPH) 31 (4) (2012) 44:1–44:10.

[2] R. Hu, M. Barnard, J. Collomosse, Gradient field descriptor for sketch based retrieval and localization, in: ICIP, 2010, pp. 1025–1028.

[3] Y. Cao, H. Wang, C. Wang, Z. Li, L. Zhang, L. Zhang, Mindfinder: interactive sketch-based image search on millions of images, in: International Conference on Multimedia, 2010, pp. 1605–1608.

[4] E. Mathias, H. Kristian, B. Tamy, A. Marc, Sketch-based image retrieval: Benchmark and bag-of-features descriptors, TVCG 17 (11) (2011) 1624–1636.

[5] R. Hu, J. Collomosse, A performance evaluation of gradient field hog descriptor for sketch based image retrieval, CVIU 117 (2013) 790–806.

[6] M. Fonseca, A. Ferreira, J. Jorge, Sketch-based retrieval of complex drawings using hierarchical topology and geometry, Computer-Aided Design 41 (12) (2009) 1067–1081.

[7] D. M. B. D. B. Walther B. Chai, E. Caddigan, L. Fei-Fei, Simple line drawings suffice for functional mri decoding of natural scene categories, in: Proc. Nat. Acad. of Sci (PNAS), 2011, pp. 9661–9666.

[8] T. Lu, C. Tai, F. Su, S. Cai, A new recognition model for electronic architectural drawings, Computer Aided Design 37 (10) (2005) 1053–1069.

[9] M. Jabal, M. Rahim, N. Othman, Z. Jupri, A comparative study on extraction and recognition method of CAD data from CAD drawings, in: Information Management and Engineering, International Conference on, 2009, pp. 709–713.

[10] Y. Cao, Z. Zhang, I. Czogiel, I. Dryden, S. Wang, 2D nonrigid partial shape matching using MCMC and contour subdivision, in: CVPR, 2011, pp. 2345–2352.

[11] S. Belongie, J. Malik, J. Puzicha, Shape matching and object recognition using shape contexts, PAMI 24 (4) (2002) 509–522.

[12] E. Mathias, H. Kristian, B. Tamy, A. Marc, An evaluation of descriptors for large-scale image retrieval from sketched feature lines, Computers & Graphics 34 (5) (2010) 482–498.

[13] S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, in: CVPR, 2006, pp. 2169–2178.

[14] Y. Cao, C. Wang, Z. Li, L. Zhang, L. Zhang, Spatial-bag-of-features, in: CVPR, 2010, pp. 3352–3359.

[15] P. Sousa, M. Fonseca, Sketch-based retrieval of drawings using spatial proximity, Journal of Visual Language and Computing 21 (2) (2010) 69–80.

[16] Z. Sun, C. Wang, L. Zhang, L. Zhang, Free hand-drawn sketch segmentation, in: ECCV, 2012, pp. 626–639.

[17] Y. Li, Y. Song, S. Gong, Sketch recognition by ensemble matching of structured features, in: British Machine Vision Conference (BMVC), 2013.

[18] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, CVPR, 2005, pp. 886–893.

[19] E. Shechtman, M. Irani, Matching local self-similarities across images and videos, in: CVPR, 2007, pp. 1–8.

[20] E. Tola, V. Lepetit, P. Fua, Daisy: An efficient dense descriptor applied to wide baseline stereo, PAMI 32 (5) (2010) 815–830.

[21] C. Lampert, H. Nickisch, S. Harmeling, Learning to detect unseen object classes by between-class attribute transfer, in: CVPR, 2009, pp. 951 – 958.

[22] Y. Fu, T. Hospedales, T. Xiang, S. Gong, Learning multi-modal latent attributes, PAMI.

[23] L. Li, H. Su, Y. Lim, L. Fei-Fei, Objects as attributes for scene classification, in: ECCV, 2010.

[24] G. Miller, Wordnet: A lexical database for english, COMMUNICATIONS OF THE ACM.

[25] C. Zitnick, D. Parikh, Bringing semantics into focus using visual abstraction, in: CVPR, 2013, pp. 3009–3016.

[26] P. Sousa, M. Fonseca, Geometric matching for clip-art drawing retrieval, Journal of Visual Communication and Image Representation 20 (2) (2009) 71–83.

[27] J. Liu, B. Kuipers, S. Savarese, Recognizing human actions by attributes, in: CVPR, 2011.

[28] F. R. Bach, G. R. G. Lanckreit, M. I. Jordan, Multiple kernel learning, conic duality, and the smo algorithm, 2004.

[29] M. Varma, D. Ray, Learning the discriminative power-invariance trade-off, 2007.

[30] A. Vedaldi, V. Gulshan, M. Varma, A. Zisserman, Multiple kernels for object detection, 2009.

[31] M. Gönen, E. Alpaydın, Multiple kernel learning algorithms, The Journal of Machine Learning Research 12 (2011) 2211–2268.

[32] F. Orabona, L. Jie, Ultra-fast optimization algorithm for sparse multi kernel learning, in: ICML, 2011, pp. 249–256.

[33] K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors, IEEE Transactions on Pattern Analysis & Machine Intelligence.

[34] J. Lim, C. Zitnick, P. Dollr, Sketch tokens: A learned mid-level representation for contour and object detection, in: CVPR, 2013, pp. 3158 – 3165.

[35] J. Sivic, A. Zisserman, Video Google: A text retrieval approach to object matching in videos, in: ICCV, Vol. 2, 2003, pp. 1470–1477.

[36] O. Boiman, M. Irani, Detecting irregularities in images and in video, IJVC 74 (1) (2007) 17–31.

[37] J. Kittler, M. Hatef, R. Duin, J. Matas, On combining classifiers, PAMI 20 (3) (1998) 226–239.

[38] K. Chatfield, J. Philbin, A. Zisserman, Efficient retrieval of deformable shape classes using local self-similarities, in: Workshop on Non-rigid Shape Analysis and Deformable Image Alignment, ICCV, 2009, pp. 264–271.

[39] A. Vedaldi, B. Fulkerson, VLFeat: An open and portable library of computer vision algorithms, `http://www.vlfeat.org/` (2008).

[40] C. Chang, C. Lin, LIBSVM: A library for support vector machines, ACM Transactions on Intelligent Systems and Technology 2 (2011) 27:1–27:27.

[41] R. G. Schneider, T. Tuytelaars, Sketch classification and classification-driven analysis using fisher vectors, ACM Trans. Graph.