Pattern Recognition Letters 130 (2020) 73–82

Contents lists available at ScienceDirect

Pattern Recognition Letters

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

# A Hybrid convolutional neural network for sketch recognition☆

Xingyuan Zhang[a], Yaping Huang[a,*], Qi Zou[a], Yanting Pei[a], Runsheng Zhang[a], Song Wang[b]

[a] Beijing Key Laboratory of Traffic Data Analysis and Mining, Beijing Jiaotong University, No. 3 Shangyuancun, Beijing, 100044, China
[b] Department of Computer Science and Engineering University of South Carolina, Columbia, SC 29208 USA

ARTICLE INFO

*Article history:*
Available online 9 January 2019

*Keywords:*
Sketch recognition
Convolutional neural network
Feature extraction
Sketch-based image retrieval

ABSTRACT

With the popularity of touch-screen devices, it is becoming increasingly important to understand users' free-hand sketches in computer vision and human-computer interaction. Most of existing sketch recognition methods employ the similar strategies used in image recognition, relying on appearance information represented by hand-crafted features or deep features from convolutional neural networks. We believe that sketch recognition can benefit from learning both appearance and shape representation. In this paper, we propose a novel architecture, named Hybrid CNN, which is composed of A-Net and S-Net. They describe appearance information and shape information, respectively. Hybrid CNN is then comprehensively evaluated in the sketch classification and retrieval tasks on different datasets, including TU-Berlin, Sketchy and Flickr15k. Experimental results demonstrate that the Hybrid CNN achieves competitive accuracy compared with the state-of-the-art methods.



## 1. Introduction

Sketching is widely used in daily life, and free-hand sketch is a simple yet powerful tool for communicating, recording and expressing with each other. It has attracted more and more attention to recognize sketches due to the widespread use of touch-screens on portable devices. However, it is much difficult to interpret free-hand sketches automatically. Some of the reasons are that: 1) natural images contain abundant details of color or texture, whereas sketches are highly abstract and only contain quite limited shape information; 2) people may present the same object using very different drawing styles. Thus, it is a great challenge for a computer to achieve robust representation for sketch recognition tasks.

Generally, the existing sketch recognition methods follow the similar strategy with image recognition. The earlier methods use hand-crafted features, such as GF-HOG [1], SIFT [2], Self Similarity (SSIM) [3], HOG [4], Structure Tensor [5] and Fisher Vector [6]. These descriptors are often combined with bag of visual words (BOW) [7,8] to yield the global features. However, they still have a gap in recognition accuracy compared with human performance.

Recently, benefiting from the deep convolutional neural networks (DCNNs) [9,10] and large-scale sketch datasets, such as TU-Berlin [11] and SketchX [12], DCNNs are effectively explored for recognizing sketch objects. DCNNs can learn more distinctive features, and thus leverage sketch classification and retrieval performance in comparison with hand-crafted visual features. The first attempt to utilize CNNs for free-hand sketch recognition is the use of two popular CNNs: AlexNet [9] and LeNet [10], and the results of classification on sketch datasets demonstrate a great improvement compared with hand-crafted features. Then, more powerful frameworks are introduced in [13–17]. The classification performance on TU-Berlin dataset has been improved to 75.42% [13], 77.95% [16] and 80.42% [17] respectively. However, it is still far behind the accuracy of natural RGB image recognition.

The key issue in sketch recognition is to learn distinctive and powerful features. So far, most models only consider the appearance information, e.g., color and texture, while few studies consider the shape information. We believe that sketch recognition can be further improved by considering the features of both appearance and shape. Based on this, we propose a novel convolutional neural network-based architecture, named Hybrid CNN, for sketch recognition in this paper. Hybrid CNN consists of two stream CNNs to extract sketch features. One stream reflects appearance structure and the other stream extracts shape information. The success of our idea depends on the capability of extracting discriminative shape features for each sketch category. For this purpose, we further develop a shape CNN (S-Net), one stream of Hybrid CNN, which transforms one sketch into point set data and performs convolutional operation on it to extract shape features.

☆ **Conflict of interest:** We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

* Corresponding author.
*E-mail address:* yphuang@bjtu.edu.cn (Y. Huang).

https://doi.org/10.1016/j.patrec.2019.01.006
0167-8655/

In summary, the main contributions of this paper are three folds:

- We propose a novel Hybrid CNN architecture to address the problem of sketch recognition. Traditional models only consider appearance information, whereas Hybrid CNN considers not only appearance features but also shape features.
- We develop a point set-based deep neural network, S-Net, to extract shape features of a sketch, which is invariant for sketch rotation and transformation.
- We conduct comprehensive experiments on two tasks: sketch classification and sketch-based image retrieval. Our proposed two-stream framework achieves superior performance compared with the state-of-the-art methods.

The remainder of this paper is organized as follows. Section 2 briefly reviews related literatures. The proposed hybrid CNN is described in Section 3. The experimental results and analysis are presented in Sections 4 and 5. Finally, we summarize our contributions and future works in Section 6.

## 2. Related work

**Sketch Classification** A large number of methods have been proposed for sketch classification in recent decades. These methods generally share the similar idea of image classification. The pipeline usually consists of two steps: feature extraction and classification. First, we generate feature descriptors of the sketch. Then, classifiers are used to predict the class labels. Basically, these methods can be divided into two categories: BOW-based models and deep learning-based models. Eitz et al. [11] firstly introduce BOW representation for free-hand sketches and learn multi-class SVM to recognize input samples. Based on previous works [11], Li et al. [14,15] consider the local features and use the holistic structure, called star-graph, to represent a sketch. Schneide et al. [6] adopt the Fisher Vector to represent the sketch and achieve significant improvement in the final classification results. Li et al. [18] propose a new criterion to maximize the weighted harmonic mean of trace ratios based on linear discriminative analysis, which can be applied on a variety of classification tasks. Chang et al. [19] propose a new semantic pooling approach and gains superior performance for challenging event analysis tasks, e.g., event detection, recognition, and recounting.

Recently, inspired by the success of DCNNs in computer vision, many researchers attempt to develop deep models for sketch recognition. Yu et al. [16] design a new deep neural network, named Sketch-a-Net, to extract deep features. Sketch-a-Net achieves 77.95% classification accuracy on TU-Berlin dataset [11]. Zhang et al. [17] propose a triplet deep network consisting of three identical sub-convolutional networks, which achieves the performance of 80.42% on TU-Berlin sketch benchmark. Another related work in sketch classification is proposed by Su et al. [20], which use 2D image rendering from 3D shape and collect a single view to test the model. The proposed model achieves 87.2% accuracy on the simplified SketchClean dataset [6].

**Sketch-Based Image Retrieval** SBIR is a typical cross-domain retrieval task, where the query is an abstract sketch and the dataset consists of natural images. Therefore, most researches focus on extracting representative features sharing between sketches and images.

Early methods extract shallow features for SBIR. Eitz et al. [21] and Hu et al. [22] extend the BOW method [7,8] to SBIR. Zhu et al. [23] are first to directly augment the semantics of discrete image hash codes by exploring auxiliary contextual modalities, which boosts the performance of contend-based image retrieval. Low-level features mainly focus on GF-HOG [1], SIFT [2], SSIM [3], HOG [4], Structure Tensor [5], histogram of edge local orientations (HELO) [24] and SHELO [25]. Proposed by Shechtman and Irani [3], self-similarity (SSIM) captures the salient sketch geometric structure computed on repeated patch patterns. Meanwhile, SSIM is closely related to ensemble matching with the potential for different visual domains, e.g., sketches, paintings and drawings. Moreover, many works also attempt to obtain middle-level features. Belongie et al. [26] propose a novel descriptor named shape context, which uses the point correspondence to estimate an alignment transform. Cao et al. [27] employ mid-level structure and firstly develop inverse index structure for scalable SBIR. Qi et al. [28] propose Perceptual Edge to describe sketches, which has achieved higher performance in cluttered scenes. Some other works extract features on the line segments and circular arcs. Xiao and Co-authors [29,30] extract shape words from sketches and perform Chamfer matching to find the query results.

Due to the success of DCNNs, researchers have also developed powerful deep learning-based approaches which are capable of modeling high-level features for sketches. Qi et al. [31] train a Siamese CNN, which calculates similarity between free-hand sketches and edge-maps generated from corresponding images. Tu et al. [32] employ a triplet CNN: an anchor branch, which is used as the reference object; one branch represents positive examples, which should be similar to the anchor; the other branch represents negative examples, which should be as different as possible to the anchor. Experimental results on real datasets indicate that deep features show superior performance on SBIR compared with hand-crafted descriptors.

**Shape Feature Descriptor** Shape feature is widely used in many vision tasks, such as shape matching and classification. We can use shape feature to recognize and understand physical structures of objects. A large number of methods for shape similarity have been proposed. Generally, shape descriptors can be divided into two main categories: (1) image-based descriptors; (2) silhouette-based descriptors.

Image-based methods mainly include contour-based descriptor and region-based descriptor. Contour-based descriptors usually rely on boundary information of a shape. The most popular contour-based descriptors are curvature scale-space based descriptor [33,34], visual parts based correspondence [35,36], and wavelet shape descriptor [37]. In contrast, region-based methods consider the whole area of the object, and commonly use moment description to represent a shape, such as Fourier-based descriptors [38–40] and Zernike Moments [41].

Some methods focus on silhouette-based descriptor [26,42]. The idea is using point-set to match two different shapes. Belongie et al. [26] present a novel approach named shape context, and corresponding points on two similar shapes have similar shape context. The shape context describes the relative spatial distribution (distance and orientation) of landmark points around feature points, thus offering a globally discriminative representation. Following the above methods, Ling and Jacobs [42] use the inner-distance to extend the shape context for shape matching. It uses inner-distance to measure two shapes. The experiments demonstrate that the shape structure captured by inner-distance is better than the Euclidean distance.

Although deep learning-based models have been successful in image recognition, a few techniques based on deep models have been developed for learning shape features. We divide them into two categories: generative model and discriminative model. The former mainly includes deep belief networks [43] and the autoencoder [44,45]. The latter adopts deterministic layers, i.e., feedforward neural networks [46]. These DCNNs perform shape feature extraction on the regular grid data. However, it is not suitable to represent a shape in a regular grid format, because the regular gird data can not represent the relations between different segments of the shape. Therefore, in order to get powerful shape features,

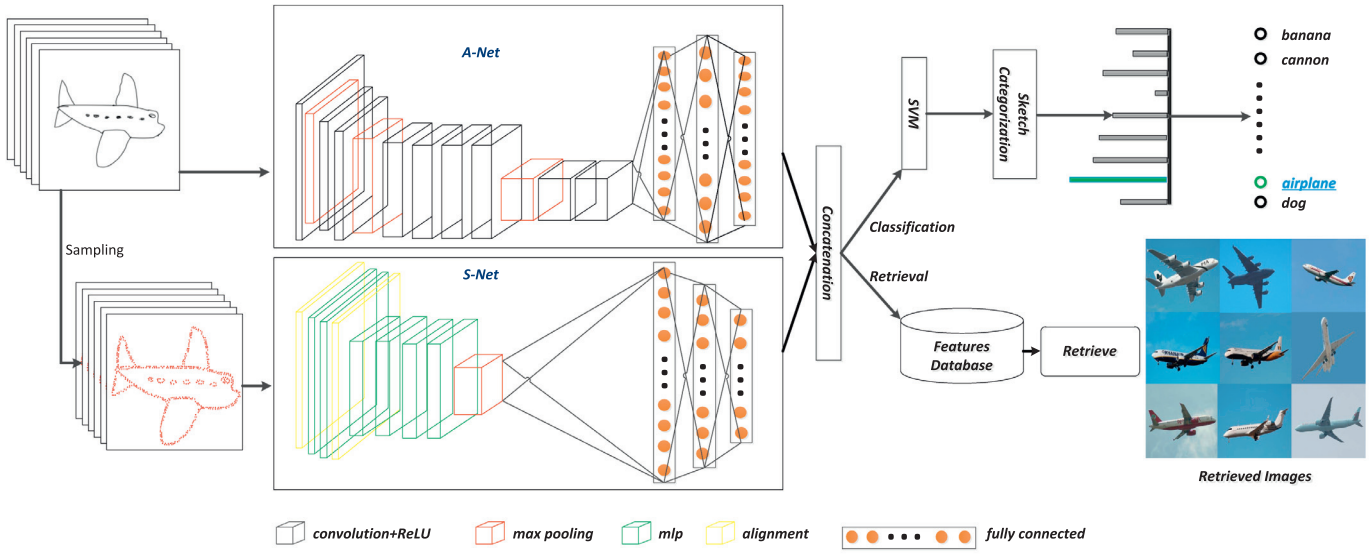



**Fig. 1.** The framework of Hybrid CNN with convolution neural network and the visualization of two-branch architecture. The first branch extracts appearance feature and the second branch extracts shape feature. A classifier, such as SVM, is applied to the concatenated feature obtained from the two branches for sketch classification. Answer of the proposed system to the input sketch is shown on the SBIR task. Note that the result set contains a very high percentage of airplanes as probably desired.

we first transform a sketch into a set of points, and then describe shape information by using point-set based convolutional neural networks. Finally, we combine the appearance and shape information to obtain the distinctive features by two stream CNNs architecture.

## 3. The proposed method

In this section, we illustrate the Hybrid CNN architecture consisting of two branches, and then we give the details of each part.

### 3.1. Pipeline of hybrid CNN

Sketch can naturally be described by both appearance and shape information. Therefore, we propose a two-branch CNN (Hybrid CNN) which combines appearance and shape information. The pipeline of Hybrid CNN is given in Fig. 1. The first branch named appearance CNN (A-Net) is to extract the appearance features, and the second branch (S-Net) is responsible for extracting shape features.

It is noted that two neural networks have different input types. The original sketch is directly used as the input of A-Net, and then we extract convolutional features and feed these feature vectors to fully connected layers. The output of A-Net is probabilities of each object class. On the other hand, in order to obtain distinctive shape features, we first sample a number of points from a sketch. Then we use stacked layers, including alignment, multi-layer perceptron (MLP) and pooling, to extract feature vectors as shape representation. The details of S-Net are described in Section 3.3.

Once the A-Net and S-Net have been trained separately, we can obtain both appearance features and shape features. Then, we normalize two feature vectors independently and concatenate them to obtain hybrid feature representation. Finally, the concatenated hybrid features are used for sketch recognition tasks, including sketch classification and SBIR. For sketch classification, the hybrid features are fed into a SVM classifier. For SBIR, we extract the feature vector of a query sketch and edge-maps generated from image datasets, and then we compare the distance to get the ranking results. As validated by our experiments on sketch classification and retrieval tasks, the representation of hybrid features is very powerful due to the consideration of both appearance and shape information.

**Table 1**
The parameter setting of A-Net.

| Type | Filter size | Filter num | Stride | Output size |
|---|---|---|---|---|
| Input | - | - | - | $225 \times 225$ |
| Conv | $15 \times 15$ | 64 | 3 | $71 \times 71$ |
| Max-Pooling | $3 \times 3$ | - | 2 | $35 \times 35$ |
| Conv | $5 \times 5$ | 128 | 1 | $31 \times 31$ |
| Conv | $3 \times 3$ | 128 | 1 | $31 \times 31$ |
| Max-Pooling | $3 \times 3$ | 2 | - | $15 \times 15$ |
| Conv | $3 \times 3$ | 256 | 1 | $15 \times 15$ |
| Conv | $3 \times 3$ | 256 | 1 | $15 \times 15$ |
| Conv | $3 \times 3$ | 256 | 1 | $15 \times 15$ |
| Conv | $3 \times 3$ | 256 | 1 | $15 \times 15$ |
| Max-Pooling | $3 \times 3$ | 2 | - | $7 \times 7$ |
| Conv | $7 \times 7$ | 512 | 1 | $1 \times 1$ |
| FC1 | $1 \times 1$ | 512 | 1 | $1 \times 1$ |
| FC2 | $1 \times 1$ | 4096 | 1 | $1 \times 1$ |
| FC3 | $1 \times 1$ | 250 | 1 | $1 \times 1$ |

### 3.2. First branch of hybrid CNN

We design A-Net to describe the appearance information. Since there have been a number of efforts in the development of deep neural networks to extract appearance features, we follow AlexNet [9] to build our A-Net. The detailed parameter setting is shown in Table 1.

### 3.3. Second branch of hybrid CNN

The architecture of S-Net is inspired by shape matching in 2D shape [20,26,47] and it preserves local neighborhood for points of shape boundaries. Thus, we first transform a sketch into a set of points and then extract the point-set based features [48–50] to represent the shape information. Whereas the shape feature represented by point-set is different from other descriptors, and it needs to overcome the following difficulties:

- Invariance under sketch deformation. It is very common that different people draw the same image but the result has a variety of styles. Thus, the learned representation from sampling points should be invariant for shape rotation and translation from different styles.

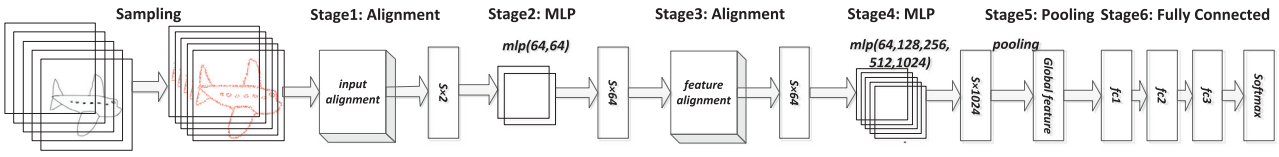



**Fig. 2.** Visualization of the S-Net network architecture. For the input sketch, the shape is depicted by red points. In its outputs, the S-Net network yields probabilities for each of the object categories.

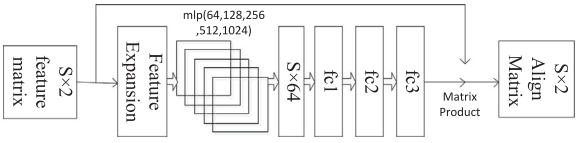



**Fig. 3.** The process of input alignment is calculated by $2 \times 2$ affine transformation matrix. First, the input of sketch is $S \times 2$, e.g., if we sample 512 points on the sketch, the input is $512 \times 2$ matrix. Second, we use $2 \times 2$ transform matrix to multiply the input, the return value will be $512 \times 2$.

- Invariance for sampling order. Compared with traditional CNNs dealing with regular grid data, we extract a large number of points to describe a sketch. However, the points are often sampled from different starting points, which results in different point perturbations for the same sketch.

To overcome the problems, we propose a unified architecture named S-Net, which is visualized in Fig. 2. The model consists of six processing stages to generate a global shape descriptor. S-Net takes point-set as input and outputs the corresponding class labels. First, because the input of S-Net is point-set, we design a preprocessing stage to generate point-set for each sketch. Second, we align the input point-set to a canonical space before feature extraction. Third, we extract features by a MLP network. Fourth, we predict another affine transformation matrix and align the point-set in feature space. Fifth, we use a MLP, a pooling and fully connected layers to extract high-level shape features for sketches. We will describe each stage sequentially.

### 3.3.1. Pre-processing stage

For the original sketches, we resize them to the size of $225 \times 225$ (stroke is represented by 0 and background is represented by 1). We then need to select a subset of points for the sketches $X = \{x_1, x_2, \ldots, x_n\}$. The point-sets can be selected randomly, but we find that it is advantageous to make sure the sample points have a certain minimum distance between different points. Thus, we choose a subset of points by iterative farthest point sampling (FPS). Compared with random sampling for all the points, this strategy has a better coverage for a sketch and guarantees the sampling as uniformly as possible. In our architecture, we sample 512 points on each sketch and each point is represented by two coordinates $(x, y)$.

### 3.3.2. Stage 1 and Stage 3—Alignment

The semantic information of points sampled from sketches should be invariant to certain geometric transformations, e.g., rotation and transformation. The question is how to solve the problems to align the different point-set to a canonical space before we extract sketch feature.

We design an alignment network to predict a $2 \times 2$ affine transformation matrix, and then we can multiply the input point-set by the estimated transformation matrix directly to align the input space. As shown in Fig. 3, we realize the alignment network by using a MLP network. The output size of each layer is 64, 128, 256, 512 and 1024, respectively. Then, a max pooling and three fully connected layers with output size 1024, 512 and 256 are attached.

In addition, the alignment stage can be further extended to the feature space. On stage 3, we estimate the transformation and align the feature vector by multiplying the transformation matrix. The architecture of alignment on feature space resembles the alignment on input space, but the dimension of transformation matrix is $64 \times 64$. It is also composed of basic modules such as MLP, max pooling and fully connected layers.

Considering the difficulty of optimization when the transformation matrix of feature space has much higher dimension than the original input space, we follow Charles et al. [49] to add a regularization term to cross-entropy as the training loss function.

$$\ell = \mathcal{J}(\theta) + \Im * \mathcal{L} \tag{1}$$

$$\mathcal{J}(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k} 1(y^{(i)} = j) \log \frac{\exp(a_k)}{\sum_{k^*=1}^{c} \exp(a_{k^*})}\right] \tag{2}$$

$$\mathcal{L} = ||\mathcal{I} - \mathcal{M}\mathcal{M}^T||_2^2 \tag{3}$$

where $\Im$ is the regularization weight of $\mathcal{L}$. $a_k = \sum_j w_{kj} z_j$ and $z_j$ is the output of hidden unit j. $\mathcal{I}$ is the input and $\mathcal{M}$ represents the feature alignment matrix predicted by aliment network. The orthogonal transformation remains the original information. Thus, the optimization becomes stable and the model will achieve better performance by adding regularization in the loss function as demonstrated in the experiment.

### 3.3.3. Stage 2 and Stage 4—MLP

Each MLP consists of no less than three layers, including one input layer, more than one hidden layer and one output layer. In our MLP, we set two hidden layers with neuron sizes both 64 and 64 on stage 2. Meanwhile we set five hidden layers on stage 4 with neuron sizes 64, 128, 256, 512 and 1024, respectively.

### 3.3.4. Stage 5—Pooling

As described in [51], the orderless of points is crucial for shape descriptor. In order to achieve the invariant of sampled point-set, we use max pooling to aggregate the information from all points. Here, max pooling takes $n$ vectors as input, which is calculated by the former MLP module. Thus, we get a new vector, which is further fed into the fully connected layers.

### 3.3.5. Stage 6—fully connected layers

The fully connected layers have full connections to the output of the previous layer. It simply multiplies the input by a weight matrix and adds a bias offset. There are three fully connected layers in our S-Net. The output of the third layer is the number of sketch categorization.

## 3.4. Hybrid CNN for sketch classification

After training A-Net and S-Net separately, a fusion process is used to integrate the appearance features and shape features, i.e. concatenation of $\mathcal{F}_{\text{fc}_2}^{app}$ and $\mathcal{F}_{\text{fc}_2}^{shape}$ with $\ell_1$ normalization, denoted by "$\mathcal{F}^+$".

$$\mathcal{F}^+ \leftarrow \left[\mathcal{F}_{fc_2}^{app}, \mathcal{F}_{fc_2}^{shape}\right] \tag{4}$$

We apply the hybrid feature representation to the sketch classification task. We train a binary SVM for each category independently. At test time, we directly use fusion features of each sketch and get prediction scores for each category. The details of sketch classification will be presented in Section 4.

### 3.5. Hybrid CNN for sketch retrieval

We further evaluate hybrid features on SBIR task. First, we use gPb [52] to extract edge-maps from natural image datasets. Then, we use Hybrid CNN to extract features independently on the generated edge-maps and the querying sketch. Finally, we use KNN classifier with Euclidean distance to retrieve the top $k$ candidate images for the sketch query. Retrieval performance of Hybrid CNN will be discussed in Section 5.

## 4. Sketch classification experiment

In this section, we evaluate our proposed Hybrid CNN on sketch classification task. We first give a description of datasets that are used to verify our method. Then we report the performance and discuss the results in details.

### 4.1. Datasets

We train and evaluate Hybrid CNN on two sketch datasets: TU-Berlin dataset [11] and Sketchy dataset [12].

- **TU-Berlin for classification**. TU-Berlin benchmark is first proposed for the aim of sketch recognition and classification. The dataset is composed of 20,000 hand-drawn sketches drawn by non-experts and contains 250 object categories, which covers most of the daily objects, e.g. airplane, turtle, bathtub and bicycle. The size of each sketch is $1111 \times 1111$. For each category, the author collects 80 sketch images. The human performance on this dataset is 73%.
- **Sketchy for classification.** Sketchy is a recent fine-grained dataset, which spans 125 categories and consists of 12,500 unique photographs of objects. The dataset also consists of 75,471 human sketches of objects inspired by those photographs. We use 50 sketches from each category for testing and the remaining sketches are used for training.

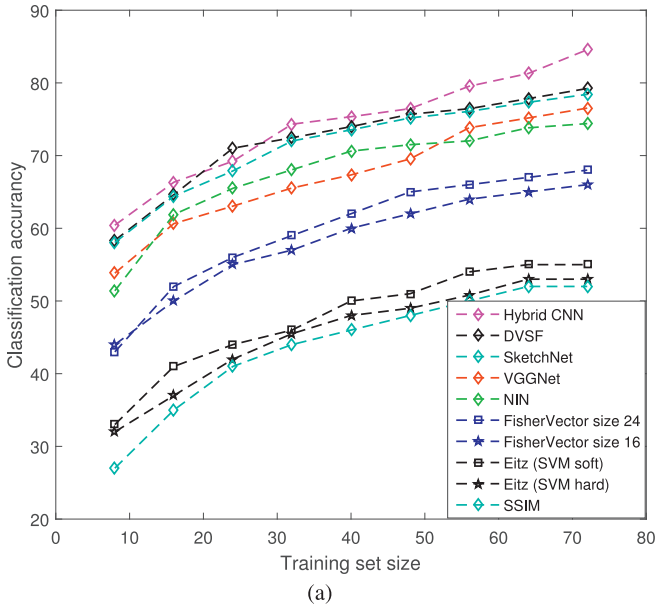
### 4.2. Implementation details

We train the model using SGD on TensorFlow. For A-Net, we use a learning rate of 0.01, momentum of 0.9, and a batch size of 64. We decrease $\alpha \leftarrow 0.3\alpha$ every epoch and terminate the optimization after 15 epochs. Bias terms are initialized of 0.1 and initial weights are distributed as a Gaussian with $\sigma = 0.01$ and $\mu = 0$.

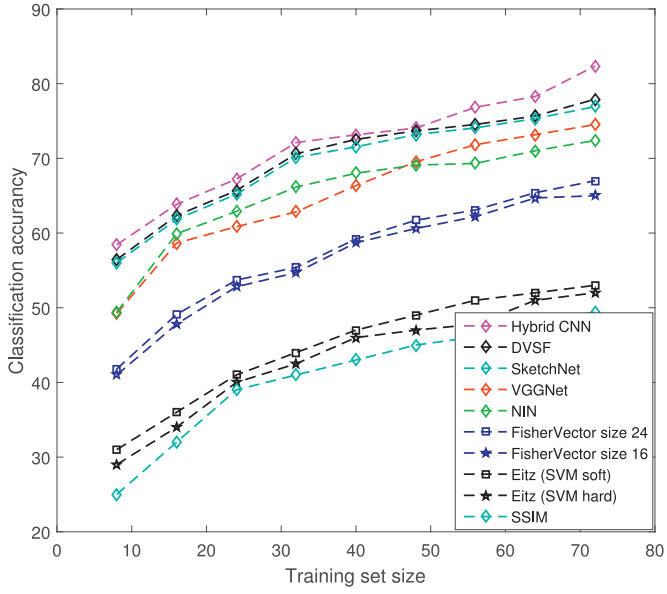
For S-Net, we use dropout with keeping ratio of 0.7 on the last three fully connected layers. The decay rate for batch normalization starts from 0.5 and is gradually increased to 0.99. We use adam optimizer with a learning rate of 0.001, momentum of 0.9, and batch size of 64. The learning rate is divided by 2 every 5 epochs. A complete model takes 2–4 hours to converge with Titan XP GPU.

### 4.3. Comparison with state-of-the-art methods

We follow the related work [11] and select different number of sketches of each category for training. Thus, we form 9 kinds of training splits, e.g., we randomly choose 8, 16, 24, 32, 40, 48, 56, 64 and 72 sketches as the training dataset of each class. Furthermore, we randomly select 10% from the remaining data as the validation dataset in order to estimate robust model parameters.




(a)




(b)

**Fig. 4.** Classification accuracies of different features using (a) TU-Berlin dataset (b) Sketchy dataset.

Hybrid CNN is compared with seven state-of-the-art baselines, which can be divided into two categories based on the type of features they use to represent sketches. One is hand-engineered features, and the other is deep learning-based features. For these hand-crafted features, we use Fisher Vector [6], SSIM [3] and Eitz (SVM soft and hard) [11] as the baselines. For deep features, we select Network in Network [53], VGGnet [54], SketchNet [17] and DVSF [55] to serve as the baselines.

### 4.4. Experiment analysis

According to Fig. 4, we can draw several conclusions from the results.

- Our proposed Hybrid CNN consistently outperforms the baselines and the performance reaches 84.42% and 82.74% on Tu-Berlin and Sketchy, respectively. This can demonstrate the im-

**Table 2**
Evaluation of different branches on TU-Berlin dataset.

| Model | Classification accuracy |
|---|---|
| A-Net | 80.02 |
| S-Net | 69.87 |
| Hybrid CNN | 84.42 |

portant role of shape features to predict the category of each sketch.

- We can find that the performance of deep features, e.g., NIN and VGGNet are not always consistently higher. For example, when the number of training sketches is less than 50, NIN is higher than VGGNet. However, when the number of training sketches is larger than 50, VGGNet is higher than NIN. The reason may be over-fitting due to the limited number of training sketches for each category. But Hybrid CNN is superior to the baselines on all training set size.
- We can observe that the overall performance of hand-crafted features on the TU-Berlin and Sketchy is worse than deep features. Moreover, the performance of deep features is much higher than human when trained with enough training sketch images. For example, when the number of training sketches for each category is 45, the classification accuracy is 73.54% using SketchNet, which achieves super-human recognition performance. Hybrid CNN achieves the comparable performance only using 35 training samples. The competitive accuracy of Hybrid CNN suggests that considering shape information is of great benefit to sketch classification.

### 4.5. Further analysis

In this section, we analyze the effect of several parameters of our method on two datasets: TU-Berlin and Sketchy datasets.

#### 4.5.1. Shape ablation studies

We investigate the classification performance with and without shape features on Sketchy. As shown in Fig. 5, when combining shape features, classification accuracy of more than 50% categories is higher than 70%, and the accuracy of 7 categories is below 45%. Furthermore, when adding shape features extracted by S-Net, sketch classification accuracy for most categories is significantly improved, and the overall accuracy is raised from 78.61% to 82.74%.

To further evaluate the impact of shape features, we give an ablation study as shown in Fig. 6. We use t-SNE [56] to embed the global 6144-D feature vector extracted from Hybrid CNN into a 2D space to visualize hybrid features.

Fig. 6a shows the 2D embedding feature space on the test dataset of TU-Berlin. Obviously, when we consider shape features, the similar sketches can be well separated according to the corresponding semantic categories. On the contrary, if we remove shape features, the feature distribution in Fig. 6b is overlapping in the center area, which brings out significant drop of classification performance compared with the consideration of shape feature.

#### 4.5.2. Evaluation of different branches

We use A-Net and S-Net to evaluate the classification accuracy respectively. Table 2 shows that the performance of S-Net is lower than that of A-Net. However, when combined with the shape information, our Hybrid CNN outperforms the A-Net by about 4.4%. These results demonstrate that extracting discriminative shape features could boost the performance of sketch classification.

#### 4.5.3. Using different networks

We perform the experiments based on several deep models, including VGGNet and GoogLeNet. Table 3 presents the results on TU-Berlin dataset. We observe that the models consistently achieve better performance when neural network goes deeper. The VGGNet+S-Net and GoogLeNet+S-Net improve the accuracy by 0.34% and 0.65% over the original Alexnet+S-Net respectively. This suggests that we can further leverage the performance of Hybrid CNN by simply employing more powerful deep models for A-Net branch.

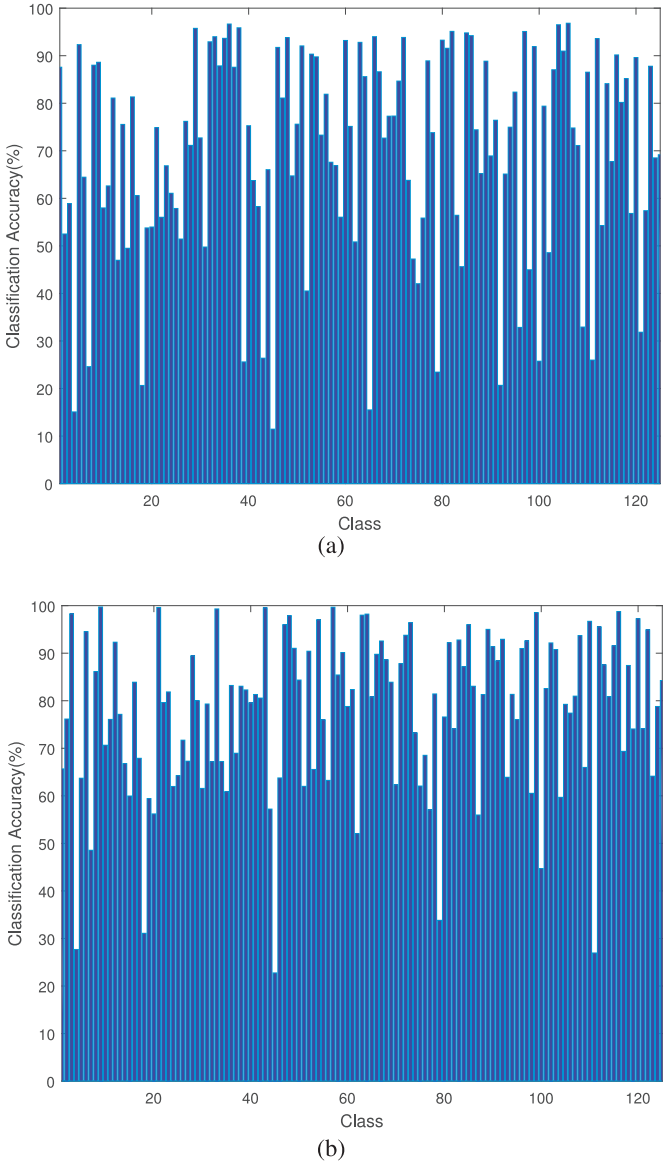



**Fig. 5.** Sketch classification performance by Hybrid CNN on Sketchy (a) without shape feature (b) with shape feature.

**Table 3**
Training different networks and evaluating the effects of feature fusion. The result is classification accuracy on TU-Berlin dataset.

| Model | Classification accuracy |
|---|---|
| A-Net | 80.02 |
| VGGNet | 80.81 |
| GoogLeNet | 81.12 |
| Hybrid CNN | 84.42 |
| Hybrid CNN (VGGNet+S-Net) | 84.76 |
| Hybrid CNN (GoogLeNet+S-Net) | 85.07 |

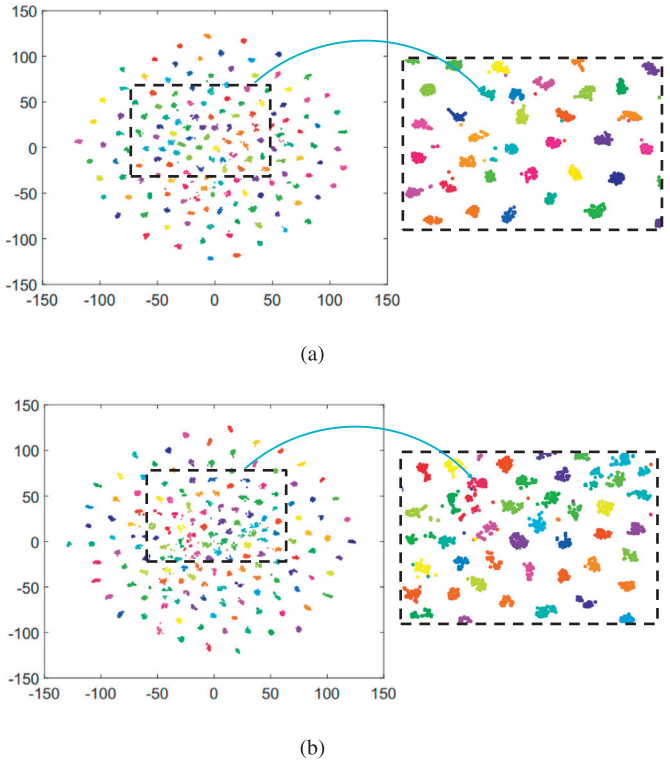



**Fig. 6.** We use t-SNE technique to visualize the features on TU-Berlin test dataset. Different colors represent different categories, and the dash box amplifies the distribution of some sketches. (a) 2D embedding visualization of sketch features combined with shape feature. (b) 2D embedding visualization of sketch features without shape feature.

**Table 4**
Effect of feature alignment. The result is classification accuracy on TU-Berlin dataset.

| Alignment | Classification accuracy |
|---|---|
| none | 82.67 |
| +input($2 \times 2$) | 83.15 |
| +feature+reg($64 \times 64$) | 83.06 |
| +all | 84.42 |

*4.5.4. Impact of parameters*

**Feature Alignment.** An important factor of our model is feature alignment used in the S-Net. We demonstrate the effects of input alignment and feature alignment for the classification accuracy on TU-Berlin in Table 4. We can see that the architecture without alignment also achieves a comparable performance. On the one hand, if we add input alignment, the performance boosts about 0.78%. On the other hand, if we further add the feature alignment and regulation in the loss function, the performance is about 83.06% compared with original 82.37%. Finally, if we add two alignments, the final accuracy is 84.42% and has a significant improvement about 2.05%.

**Regularization Parameter.** The regularization parameter could also influence the classification performance. Therefore we perform an experiment to investigate the impact of regularization parameter $\Im$. As shown in Table 5, we vary $\Im$ from 0.005 to 0.06, finding that $\Im = 0.01$ achieves the best accuracy on TU-Berlin dataset.

*4.5.5. Robustness test*

**Number of Input Points.** We investigate the effect of the number of input points on sketch classification. Fig. 7 shows some examples of different number of input points. We evaluate the classification performance on TU-Berlin and Sketchy, and we summarize the results in Fig. 8. Note that the accuracy of Hybrid CNN at 512 input points is on par with 1024 input points, and the performance saturates when the number of points is around 1000 points. Thus, to reduce the time complexity of calculation, we set 512 input points in all the experiments. Moreover, as shown in Fig. 8, when the number of input points is further reduced to 256, 128, 64 and 32, the accuracy drops drastically.

**Table 5**
Effects of regularization weight $\Im$. The result is classification accuracy on TU-Berlin dataset.

| Regularization weight $\Im$ | Classification accuracy |
|---|---|
| 0.005 | 83.71 |
| 0.01 | 84.42 |
| 0.03 | 83.86 |
| 0.06 | 82.67 |

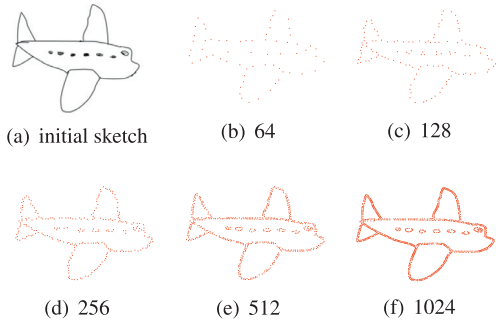



**Fig. 7.** Visualization with the variation of points.

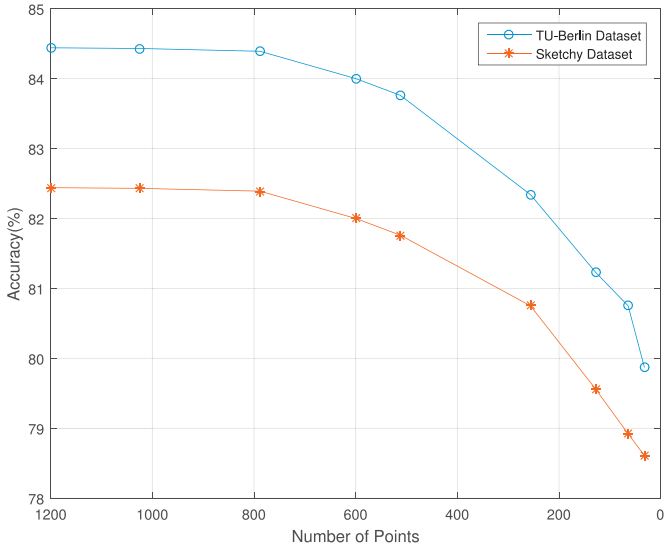



**Fig. 8.** Classification accuracy performance with respect to the variation of point number on TU-Berlin and Sketchy.

**Negative Data Points.** In order to investigate the influence of the negative data points, we add Gaussian noise to each point independently with different perturbation noise standard for training on TU-Berlin and Sketchy datasets. Fig. 9 illustrates several noisy data point samples, and Fig. 10 shows the classification results. As shown in Fig. 10, our proposed Hybrid CNN still has higher than 80% and 83.5% accuracy on Sketchy and TU-Berlin datasets respectively, even when the noise standard reaches 0.05. The results suggest that our method is robust to the Gaussian noise. Obviously, how to improve the robustness for negative samples is an important issue, thus many efforts try to alleviate the effects of negative samples. For example, in [57], the authors assign the negative

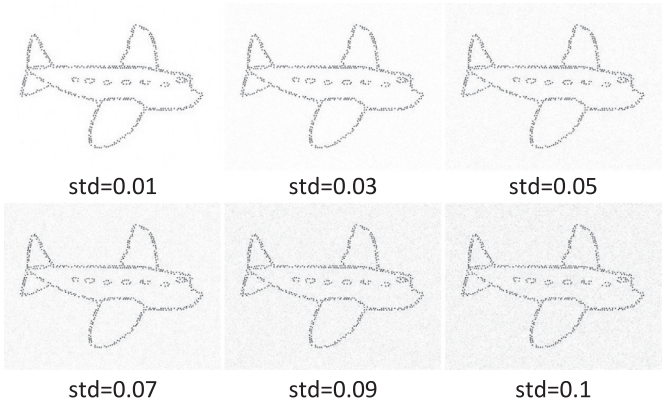



**Fig. 9.** Airplane sketches with different-level Gaussian noise.

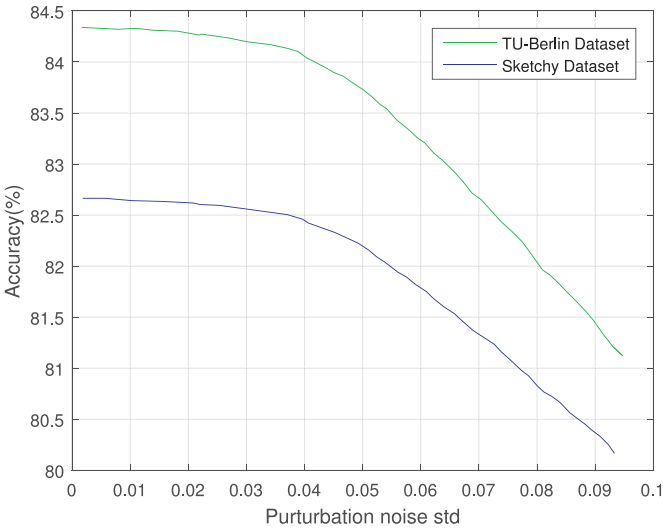



**Fig. 10.** Classification accuracy with respect to the effects of negative data points on TU-Berlin and Sketchy.

videos fine-grained labels to capture more informative cues, thus bringing a new insight for dealing with negative data samples.

## 5. Sketch-based image retrieval experiment

In this section, we show the application of Hybrid CNN on SBIR task.

### 5.1. Datasets

We train and validate the model on two sketch datasets: Sketchy dataset [12] and Flickr15k dataset [58].

- **Sketchy for SBIR.** As introduced above, we use all the images and corresponding sketches to perform the SBIR task.
- **Flickr15k for SBIR.** Flickr15k consists of 15,024 images sampled from Flickr. They are labeled into 33 categories based on different kinds of shapes. Thus, they get 330 free-hand drawn sketches drawn by 10 non-expert sketchers. They set all the original real images as retrieval candidates and 330 sketches as queries.
- **Flickr15k-Large for SBIR.** Flickr15k-Large is a new dataset collected by us following the Flickr15k. Since Flickr15k benchmark only contains 10 sketches for each category, we extend the dataset by performing the following data augmentation: 1) we flip the sketches horizontally and vertically; 2) we rotate the results by $\pm 10$ and $\pm 15$ degrees; 3) we central zoom $\pm 1.2$ and $\pm 1.5$ times of the sketch height; 4) we consider planar motions of sketches, i.e., the horizontal and vertical translations by $\pm 10$ and $\pm 15$ pixels. In our experiment, we divide the dataset into two parts: two-thirds are used for training and the remaining are used for testing.

**Table 6**
SBIR comparison results (MAP) on the Sketchy benchmark.

| Methods | MAP |
|---|---|
| HOG [4] | 0.115 |
| GF-HoG [1] | 0.157 |
| SHELO [25] | 0.161 |
| Sketch-a-Net [59] | 0.208 |
| Siamese CNN [31] | 0.481 |
| GN Triplet [12] | 0.529 |
| **Hybrid CNN(without shape feature)** | **0.553** |
| **Hybrid CNN(with shape feature)** | **0.574** |

**Table 7**
SBIR comparison results (MAP) on the Flickr15K-Large benchmark.

| Methods | Vocabulary size | MAP |
|---|---|---|
| StructureTensor [5] | non-BoW | 0.0835 |
| StructureTensor [5] | 500 | 0.0898 |
| ShapeContext [26] | 3500 | 0.0914 |
| SIFT [2] | 1000 | 0.1011 |
| HOG [4] | 3000 | 0.1193 |
| GF-HOG [1] | 3500 | 0.1322 |
| 3Dshape [62] | - | 0.1891 |
| Siamese CNN [31] | - | 0.1954 |
| Triplet CNN [32] | - | 0.2445 |
| Half-sharing Triplet CNN [60] | - | 0.2585 |
| $\text{CPRL}_{\text{CNN}+LKS}$ [61] | - | 0.2634 |
| **Hybrid CNN(without shape feature)** | - | **0.2656** |
| **Hybrid CNN(with shape feature)** | - | **0.2874** |

### 5.2. Comparison with state-of-the-art methods

In this section, we evaluate the model on the Sketchy and Flickr15k-Large by qualitative and quantitative comparisons. The compared methods include hand-crafted features and deep learning-based features.

For the Sketchy, hand-crafted features mainly focus on GF-HOG [1], HOG [4] and SEHLO [25]. Deep features mainly focus on GN Triplet [12], Siamese CNN [31] and Sketch-a-Net [59]. Specially, for the latter methods, we fine-tune the model of [59] on TU-Berlin. Meanwhile, we use the public models of GN Triplet directly. In addition, because there is a lack of stroke order information for the Sketchy, we only make use of a single channel Sketch-a-Net proposed in [59]. The comparison of MAP over different SBIR methods on the Sketchy is summarized in Table 6. Moreover, we show some query examples with top-15 SBIR retrieval results in Fig. 11.

To further validate Hybrid CNN framework, we use Flickr15k-Large as the benchmark for SBIR experiment. For the hand-crafted features, we further divide them into non-BoW methods and BoW-based methods. For the non-BOW methods, StructureTensor [5] is served as a baseline. For the BOW methods, we use Gradient Field HOG (GF-HOG) [1], SIFT [2], StructureTensor [5] and Shape Context [26] as baselines. For the deep features, we employ Siamese CNN [31], Triplet CNN [32], Half-Sharing Triplet CNN [60], $\text{CPRL}_{\text{CNN}+LKS}$ [61] and 3Dshape [62] as baselines.

Quantitative and qualitative results are shown in Table 7 and Fig. 12. Table 7 reports MAP, produced by averaging AP over all the sketch queries on the testing Flickr15k-Large.

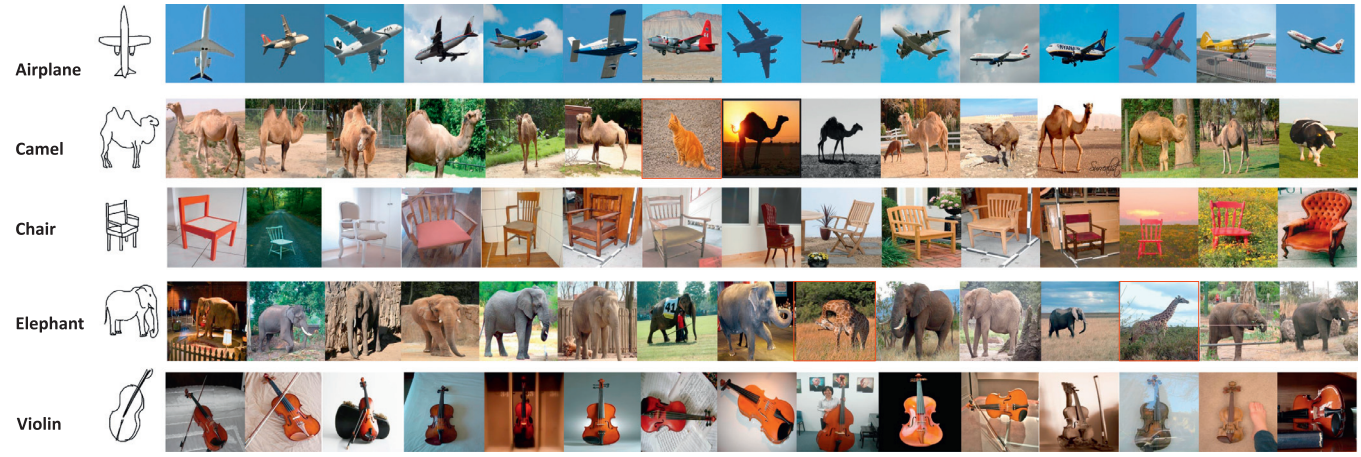



**Fig. 11.** The visualization of our SBIR: five example query sketches with their top-15 retrieval results on Sketchy dataset by Hybrid CNN model. Red boxes indicates false positives.

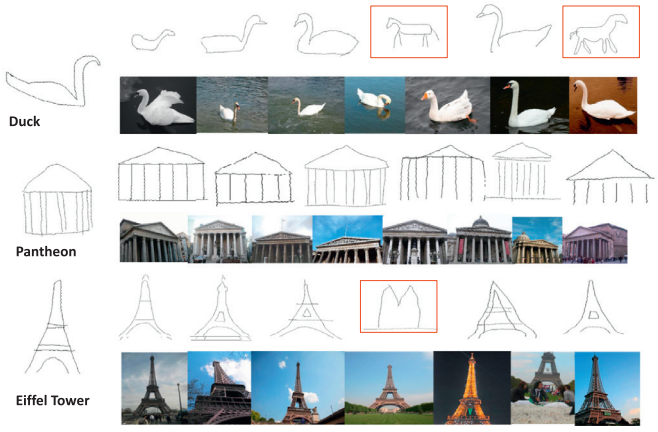



**Fig. 12.** Representative SBIR results on Flickr15K-Large using a sketch as query and sketches or images as the retrieval results. For each query, we get two results: one for cross domain search and the other for intra-domain. Non-relevant retrieval results are outlined in red box. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

### 5.3. Experiment analysis

As shown in Table 6, we can draw the following conclusions:

- Hybrid CNN outperforms all the baselines and obtains a better performance on Sketchy with a MAP of 0.574. This indicates that shape feature is beneficial for capturing structure information of sketches.
- It is notable that all the deep learning-based methods outperform the traditional methods that use hand-crafted features by a large gap.
- For the deep feature of Sketch-a-Net in [59], the retrieval images all contain well-aligned objects, and the background is removed. The process makes the edge-maps be well represented, and the result has almost identical patterns with free-hand sketches. However, all images in our task are realistic with complex backgrounds, which leads to a big gap between sketches and edge-maps. Similar problems also exist for HOG, GF-HOG and SHELO.

We report the comparison in Table 7. Hybrid CNN outperforms all other methods by a significant margin. Specially, our proposed approach achieves a MAP of 0.2874 and improves the accuracy by 0.02 compared with $\text{CPRL}_{\text{CNN}+LKS}$. It proves that Hybrid CNN is beneficial for capturing shape feature on sketches, which could help improve the retrieval performance. But the lack of sketches leads to overall performance on Flickr15k-Large fairly low. In addition, Fig. 12 presents several intra-domain and cross-domain retrieval results where false positives are outlined in red box. We can see that due to lacking enough sketch training data, false-positive results mainly focus on sketches.

## 6. Conclusion

In this paper, we propose a deep-learning based framework for sketch recognition named of Hybrid CNN. Hybrid CNN obtains efficient and comprehensive representation of sketches, and the shape features leverage accuracy of sketch recognition by 2%–5% over the existing state-of-the-art. Based on the proposed method, we demonstrate state-of-the-art performance on sketch classification and SBIR tasks by TU-Berlin, Sketchy and Flickr15K-Large datasets.

In the future, although deep learning-based features are much better than hand-crafted features, the supervised learning methods require expensive labeled data. Thus, semi-supervised deep neural networks for sketch recognition will be investigated.

## Acknowledgements

This work is supported by the National Natural Science Foundation of China (61273364, 61473031, and 61472029), the Fundamental Research Funds for the Central Universities (2016YJS041, 2018YJS035).

## References

[1] R. Hu, S. James, T. Wang, J. Collomosse, Markov random fields for sketch based video retrieval, in: ACM Conference on International Conference on Multimedia Retrieval, 2013, pp. 279–286.
[2] D.G. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vis. 60 (2) (2004) 91–110.
[3] E. Shechtman, M. Irani, Matching local self-similarities across images and videos, in: Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on, 2007, pp. 1–8.
[4] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: IEEE Computer Society Conference on Computer Vision Pattern Recognition, 2005, pp. 886–893.
[5] M. Eitz, K. Hildebrand, T. Boubekeur, M. Alexa, A descriptor for large scale image retrieval based on sketched feature lines, in: Eurographics Symposium on Sketch-Based Interfaces and Modeling, 2009, pp. 29–36.
[6] T. Tuytelaars, Sketch classification and classification-driven analysis using fisher vectors, ACM, 2014.
[7] T. Joachims, Text categorization with support vector machines: learning with many relevant features, in: European Conference on Machine Learning, Springer, 1998, pp. 137–142.

[8] A. McCallum, K. Nigam, et al., A comparison of event models for naive Bayes text classification, in: AAAI-98 Workshop on Learning for Text Categorization, 752, Citeseer, 1998, pp. 41–48.
[9] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: International Conference on Neural Information Processing Systems, 2012, pp. 1097–1105.
[10] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
[11] M. Eitz, J. Hays, M. Alexa, How do humans sketch objects? ACM Trans. Graph. 31 (4) (2012) 1–10.
[12] P. Sangkloy, N. Burnell, C. Ham, J. Hays, The sketchy database: learning to retrieve badly drawn bunnies, ACM Trans. Graph. 35 (4) (2016) 119.
[13] O. Seddati, S. Dupont, S. Mahmoudi, Deepsketch: deep convolutional neural networks for sketch recognition and similarity search, in: International Workshop on Content-Based Multimedia Indexing, 2015, pp. 1–6.
[14] Y. Li, Y.Z. Song, S. Gong, Sketch recognition by ensemble matching of structured features, in: British Machine Vision Conference, 2013, pp. 35.1–35.11.
[15] Y. Li, T.M. Hospedales, Y.Z. Song, S. Gong, Free-hand sketch recognition by multi-kernel feature learning, Comput. Vis. Image Understanding 137 (2015) 1–11.
[16] Q. Yu, Y. Yang, Y.Z. Song, T. Xiang, T. Hospedales, Sketch-a-net that beats humans (2015).
[17] H. Zhang, S. Liu, C. Zhang, W. Ren, R. Wang, X. Cao, Sketchnet: sketch classification with web images, in: Computer Vision and Pattern Recognition, 2016, pp. 1105–1113.
[18] Z. Li, F. Nie, X. Chang, Y. Yang, Beyond trace ratio: weighted harmonic mean of trace ratios for multiclass discriminant analysis, IEEE Trans. Knowl. Data Eng. 29 (10) (2017) 2100–2110.
[19] X. Chang, Y.-L. Yu, Y. Yang, E.P. Xing, Semantic pooling for complex event analysis in untrimmed videos, IEEE Trans. Pattern Anal. Mach. Intell. 39 (8) (2017) 1617–1632.
[20] H. Su, S. Maji, E. Kalogerakis, E. Learnedmiller, Multi-view convolutional neural networks for 3d shape recognition (2015) 945–953.
[21] M. Eitz, K. Hildebrand, T. Boubekeur, M. Alexa, Sketch-based image retrieval: benchmark and bag-of-features descriptors, IEEE Trans. Vis. Comput. Graph. 17 (11) (2011) 1624–1636.
[22] R. Hu, M. Barnard, J. Collomosse, Gradient field descriptor for sketch based retrieval and localization, in: IEEE International Conference on Image Processing, 2010, pp. 1025–1028.
[23] L. Zhu, Z. Huang, Z. Li, L. Xie, H.T. Shen, Exploring auxiliary context: discrete semantic transfer hashing for scalable image retrieval, IEEE Trans. Neural Netw. Learn. Syst. (99) (2018) 1–13.
[24] J.M. Saavedra, B. Bustos, An Improved Histogram of Edge Local Orientations for Sketch-Based Image Retrieval, Springer Berlin Heidelberg, 2010.
[25] J.M. Saavedra, Sketch based image retrieval using a soft computation of the histogram of edge local orientations (s-helo), in: IEEE International Conference on Image Processing, 2015, pp. 2998–3002.
[26] S. Belongie, J. Malik, J. Puzicha, Shape matching and object recognition using shape contexts, in: IEEE International Conference on Computer Science and Information Technology, 2010, pp. 483–507.
[27] Y. Cao, C. Wang, L. Zhang, L. Zhang, Edgel index for large-scale sketch-based image search, in: Computer Vision and Pattern Recognition, 2011, pp. 761–768.
[28] Y. Qi, Y.Z. Song, T. Xiang, H. Zhang, T. Hospedales, Y. Li, J. Guo, Making better use of edges via perceptual grouping, in: Computer Vision and Pattern Recognition, 2015, pp. 1856–1865.
[29] C. Xiao, C. Wang, L. Zhang, L. Zhang, Sketch-based image retrieval via shape words, in: ACM on International Conference on Multimedia Retrieval, 2015, pp. 571–574.
[30] Y. Zheng, H. Yao, S. Zhao, Y. Wang, Discovering discriminative patches for free–hand sketch analysis, Multimedia Syst. 23 (6) (2017) 691–701.
[31] Y. Qi, Y.Z. Song, H. Zhang, J. Liu, Sketch-based image retrieval via siamese convolutional neural network, in: IEEE International Conference on Image Processing, 2016, pp. 2460–2464.
[32] B. Tu, L. Ribeiro, M. Ponti, J. Collomosse, Generalisation and sharing in triplet convnets for sketch based visual search (2016).
[33] S. Abbasi, Efficient and Robust Retrieval by Shape Content through Curvature Scale Space, 1996.
[34] F. Mokhtarian, A.K. Mackworth, A theory of multiscale, curvature-based shape representation for planar curves, IEEE Trans. Pattern Anal. Mach. Intell. 14 (8) (1992) 789–805.
[35] L.J. Latecki, R. Lakmper, Contour-based shape similarity, Proc. SPIE 3454 (1999) 617–624.
[36] L.J. Latecki, R. Lakmper, Shape similarity measure based on correspondence of visual parts, IEEE Trans. Pattern Anal. Mach. Intell. 22 (10) (2000) 1185–1190.
[37] G.H. Chuang, C.J. Kuo, Wavelet descriptor of planar curves: theory and applications., IEEE Trans. Image Process. 5 (1) (1996) 56–70.
[38] Y. Gdalyahu, D. Weinshall, Flexible syntactic matching of curves and its application to automatic hierarchical classification of silhouettes, IEEE Trans. Pattern Anal. Mach. Intell. 21 (12) (1999) 1312–1328.
[39] E. Persoon, K.S. Fu, Shape discrimination using fourier descriptors., IEEE Trans. Syst. Man Cybern. 7 (3) (1986) 388–397.
[40] C.T. Zahn, R.Z. Roskies, Fourier Descriptors for Plane Closed Curves, IEEE Computer Society, 1972.
[41] Z. Ma, B. Kang, K. Lv, M. Zhao, Nonlinear radon transform using zernike moment for shape analysis., Comput. Math. Methods Med. 2013 (3) (2013) 208402.
[42] H. Ling, D.W. Jacobs, Shape classification using the inner-distance., IEEE Trans. Pattern Anal. Mach. Intell. 29 (2) (2007) 286.
[43] S. Bu, Z. Liu, J. Han, J. Wu, R. Ji, Learning high-level feature by deep belief networks for 3-d model retrieval and recognition, IEEE Trans. Multimedia 16 (8) (2014) 2154–2167.
[44] J. Xie, Y. Fang, F. Zhu, E. Wong, Deepshape: deep learned shape descriptor for 3d shape matching and retrieval, IEEE Trans. Pattern Anal. Mach. Intell. 39 (7) (2015) 1335–1345.
[45] Z. Zhu, X. Wang, S. Bai, C. Yao, Deep learning representation using autoencoder for 3d shape retrieval, in: International Conference on Security, Pattern Analysis, and Cybernetics, 2016, pp. 279–284.
[46] Y. Fang, J. Xie, G. Dai, M. Wang, F. Zhu, T. Xu, E. Wong, 3d deep shape descriptor, in: Computer Vision and Pattern Recognition, 2015, pp. 2319–2328.
[47] C.R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, L.J. Guibas, Volumetric and multi-view CNNS for object classification on 3d data (2016) 5648–5656.
[48] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. Salakhutdinov, A. Smola, Deep sets (2017).
[49] R.Q. Charles, H. Su, K. Mo, L.J. Guibas, Pointnet: deep learning on point sets for 3d classification and segmentation (2016).
[50] M. Jaderberg, K. Simonyan, A. Zisserman, K. Kavukcuoglu, Spat. Transformer Netw. (2015) 2017–2025.
[51] O. Vinyals, S. Bengio, M. Kudlur, Order matters: sequence to sequence for sets, Comput. Sci. (2015).
[52] P. Arbelçez, M. Maire, C. Fowlkes, J. Malik, Contour detection and hierarchical image segmentation., IEEE Trans. Pattern Anal. Mach. Intell. 33 (5) (2011) 898–916.
[53] M. Lin, Q. Chen, S. Yan, Network in network, Comput. Sci. (2013).
[54] K. Chatfield, K. Simonyan, A. Vedaldi, A. Zisserman, Return of the devil in the details: delving deep into convolutional nets, Comput. Sci. (2014).
[55] J.Y. He, X. Wu, Y.G. Jiang, B. Zhao, Q. Peng, Sketch recognition with deep visual-sequential fusion model, in: ACM on Multimedia Conference, 2017, pp. 448–456.
[56] A. Prezgonzçlez, M. Vergara, J.L. Sanchobru, L.J.P. van der Maaten, G.E. Hinton, D. Shanmugapriya, G. Padmavathi, J. Kubo, P.E.E. Gantz, I. Science, Visualizing data using t-sne, J. Mach. Learn. Res. 9 (2605) (2008) 2579–2605.
[57] Z. Ma, X. Chang, Y. Yang, N. Sebe, A.G. Hauptmann, The many shades of negativity, IEEE Trans. Multimedia 19 (7) (2017) 1558–1568.
[58] R. Hu, J. Collomosse, A performance evaluation of gradient field hog descriptor for sketch based image retrieval, Comput. Vis. Image Understanding Cviu 117 (7) (2013) 790–806.
[59] Q. Yu, F. Liu, Y.Z. Song, T. Xiang, T.M. Hospedales, C.L. Chen, Sketch me that shoe, in: Computer Vision and Pattern Recognition, 2016, pp. 799–807.
[60] T. Bui, L. Ribeiro, M. Ponti, J. Collomosse, Compact descriptors for sketch-based image retrieval using a triplet loss convolutional neural network, Comput. Vis. Image Understanding (2017).
[61] D. Xu, X. Alamedapineda, J. Song, E. Ricci, N. Sebe, Cross-paced representation learning with partial curricula for sketch-based image retrieval., IEEE Trans. Image Process. PP (99) (2018).
[62] F. Wang, L. Kang, Y. Li, Sketch-based 3d shape retrieval using convolutional neural networks, Comput. Sci. (2015) 1875–1883.