Received June 21, 2019, accepted July 16, 2019, date of publication July 18, 2019, date of current version August 12, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2929804

SketchSegNet+: An End-to-End Learning of RNN for Multi-Class Sketch Semantic Segmentation

YONGGANG QI[1], (Member, IEEE), AND ZHENG-HUA TAN[2], (Senior Member, IEEE)
[1]School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China
[2]Department of Electronic Systems, Aalborg University, 9220 Aalborg, Denmark

Corresponding author: Yonggang Qi (qiyg@bupt.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61601042, in part by the 111 Project of China under Grant B08004 and Grant B17007, and in part by the Center for Data Science of Beijing University of Posts and Telecommunications.

ABSTRACT We investigate the problem of stroke-level sketch segmentation, which is to automatically assign strokes of a given sketch with semantic labels. Solving the problem of sketch segmentation opens the door for fine-grained sketch interpretation, which can benefit many novel sketch-based applications, including sketch recognition and sketch-based image retrieval. In this paper, we propose an approach for multi-class sketch semantic segmentation by considering it as a sequence-to-sequence generation problem. Specifically, an end-to-end learned network *SketchSegNet+*, built on recurrent neural networks (RNN), is presented to translate a sequence of strokes into a sequence of semantic labels. In addition, a large-scale stroke-level sketch segmentation dataset is constructed for the first time, which is composed of 150K annotated free-hand human sketch selected from *QuickDraw*. The dataset will be released publicly. The experimental results of stroke-level sketch semantic segmentation on this novel dataset and the SPG dataset demonstrate the effectiveness of our approach.

INDEX TERMS Stroke-level sketch segmentation, recurrent neural network.

I. INTRODUCTION

Free-hand human sketching is commonly used for recording and communication since ancient times. Closely with the ubiquitous touch-screen devices such as tablets and smart-phones recently, drawing sketches has become one of the most convenient ways for human-computer interaction. The booming of sketching in fact underlines the uniqueness of drawing sketches as it is more concise, informative and convenient comparing against typing words in many cases [1].

Study of sketch has lately attracted increased interest concerning both traditional tasks like sketch recognition [2], sketch-based image retrieval (SBIR) [2], as well as many new higher-level applications, such as sketch synthesis [3] and scene sketch understanding [4]. The fundamental issue in all these tasks is sketch interpretation or sketch understanding [5].

Interpreting a sketch is quite challenging since the ambiguity inherently exist in sketches [6]. First, sketches are abstract depictions, which are only composed of sparse black lines on a white background. Conventional pipelines for natural images are invalid for sketches since sketches lack dense visual cues such as color and texture, which the study of photograph heavily relies on. Secondly, there are large variations commonly existed in drawing appearance, which are caused by different drawing skills and style preferences of amateur painters.

Most previous works for sketch understanding are treated as the problem of object-level recognition, which lacks of finer-level interpretation. In this work, we aim to take one step further to investigate stroke-level sketch segmentation, which is to train machines to label sketch strokes with semantic annotations referring to part concepts. The problem of sketch segmentation [7] has begun to attract more attentions recently, since it is very fundamental and is a potential driving force for higher-level applications, such as sketch captioning [8], [9] and sketch generation [3], [10].

Despite some progress [5], [7], [9], [11] has been made toward the goal of sketch semantic segmentation, the drawing orders of strokes as well as the context amongst them have rarely been used, and we believe they are useful for this task. There are two main reasons behind the belief. First, the work in [12] has demonstrated that most people follow the same drawing pattern of orders for a specific sketch object. To draw

The associate editor coordinating the review of this manuscript and approving it for publication was Jianqing Zhu.

a face, for example, people tend to finish the outline of head, then the left eye followed by the right eye and the mouth at the last. Secondly, to determine the semantic meaning of strokes, contextual evidence should be meaningful. For example, one can easily interpret the meaning of the circles on a face sketch as eyes.

To model the drawing patterns and make better use of contextual cues among them, we treat stroke-level sketch segmentation as a sequence prediction problem, and a sequence-to-sequence model based on Variational Autoencoder (VAE) [13] is developed, which is similar to the proposed architecture in [14]. Specifically, similar to [14], a bidirectional RNN (BiRNN) [15] serves as an encoder, which takes a sketch as an input in a format of points list (a set of coordinate and pen actions), and outputs a latent vector. Afterwards, the latent vector is used to construct the initial hidden state of the decoder, together with each point in the drawing order, and the autoregressive RNN-based [16] decoder is used to predict the most possible label. The training loss in our model is mean squared error (MSE), which is quite effective for training an end-to-end RNN-based sketch segmentator.

To facilitate the learning of our deep network, we construct a large scale dataset of over 150K sketches with dense semantic labeling strokes based on QuickDraw [14]. Paper [17] presents a similar dataset which is based on TU-Berlin sketch dataset [12], but there are only 120 labeled sketches over just six classes. Lately, paper [18] contributes a much larger one (the *SPG* dataset), which is also built on QuickDraw [14], and contains 20K sketches across 25 categories. However, our dataset is different from Li's dataset in that ours has significantly larger variety in each class, and some complimentary object classes are provided as well.

Our primitive work was published in [19] which contributes (i) an initial version of the recurrent neural network for stroke-level sketch segmentation, i.e. *SketchSegNet*, and (ii) a sketch segmentation dataset over seven categories with 57K samples. In the present work, there are three major extensions regarding to algorithm, dataset and experiments. The contributions of this work are: Firstly, to the best of our knowledge, an end-to-end recurrent neural network is trained to solve the problem of stroke-level sketch segmentation across multiple categories. In particular, it is the first time that once the model is learned, it enables segmentation on various sketch objects simultaneously. On contrary, the previous work [19] is only capable of segmenting strokes within a specific class, that requires to learn one dedicated model for each target sketch object category. This is why we name the network as *SketchSegNet+*. Secondly, there is a large extension on the proposed sketch segmentation dataset, which is extended from seven categories to twenty nearly three-fold as big as the previous one. In total, there are about 150K sketches with every stroke annotated, named as *SketchSeg-150K*. In the last, extensive experiments on both our proposed *SketchSeg-150K* dataset and the *SPG* dataset [18] to evaluate our approach for stroke-level sketch segmentation.

II. RELATED WORK

A. SKETCH GROUPING

Sketch understanding is usually defined as a problem of sketch grouping. Given a sketch, the goal is to divide its strokes into clusters, each of which hopefully can correspond to a meaningful object part, e.g., in [11] and [18]. Paper [11] formulates the problem as a graph partition problem, and presents a ranking strategy with multiple Gestalt cues under a global optimization framework to group strokes. Paper [18] develops a sequence-to-sequence VAE model to output a stroke affinity matrix, followed by a post-processing of clustering to obtain the final grouping result.

However, the grouping results do not explicitly provide part labels, which prevents it from being interpreted thoroughly, hence prohibits from better implementing high-level applications. For instance, we believe that providing semantic meanings of strokes can facilitate fine-grained sketch-based image retrieval (FG-SBIR). Furthermore, sketch captioning is made possible in analogy with image captioning, if language-level interpretation is available. This can also facilitate sketch synthesis following a description-based approach.

B. SEMANTIC SKETCH SEGMENTATION

There is a growing body of literature regarding semantic sketch segmentation and labeling. Paper [7] proposed a data-driven approach for semantic segmentation and assigned labels on input sketches synchronously. But it requires a repository of 3D models, which come from the same object categories as the input sketch and already have been segmented and labeled, to pre-segment input sketch into parts by simply matching them. This is obviously sub-optimal since an extra construction of a 3D model repository is demanded. Later on, the work in [17] treats sketch segmentation as a graph partition problem, where each stroke is firstly scored the possibility of its belonging to a label, thus a Conditional Random Field (CRF) is applied to find the optimal configuration globally. In addition, instead of treating sketch segmentation as a stroke-level labeling problem, Sarvadevabhatla *et al.* [9] proposed the model of *SketchParse* to parse sketch regions into semantic parts, which is somehow treating sketch segmentation as the same as natural image segmentation.

However, all of them neglect to utilize the property of stroke orders. To make used of stroke drawing orders for stroke-level segmentation, we present a RNN-based model, inspired by [14] where a generative model Sketch-RNN is introduced for sketch generation. In the present work, instead of generating new sketch strokes, semantic part label of each sketch stroke is estimated by re-designing the decoder of Sketch-RNN. In particular, Gaussian Mixture Model (GMM) sampling in the model of Sketch-RNN is removed and replaced by a set of fully connected layers together with a softmax layer to output part labels. The early version is described in [19], and here we present the enhanced version named SketchSegNet+ because of its extension on having ability of segmentation over multiple classes, which can largely benefit its practical usage.

C. SEQUENTIAL REPRESENTATION OF STROKES FOR SKETCH ANALYSIS

It is an effective representation of drawings to express a sketch as a sequence of strokes in the form of a sequential vector, which is composed of a set of pen stroke actions involving the point coordinate on canvas and the pen state accordingly [14]. Upon this RNN-compatible data format, a number of works are introduced and achieve impressive results including sketch generation [14], sketch synthesis [3], sketch grouping [18] and sketch abstraction [20]. By representing a sketch as a list of points in a set of five elements vector, Ha and Eck [14] successfully proposed to learn a Sketch-RNN which is able to mimic the creative ability of human to generate sketches stroke by stroke. Song *et al.* [3] further showed how to abstract a photograph into sketch by using the sequential representation of strokes to learn a LSTM-driven synthesizer. Similarly, Muhammad *et al.* [20] present a method for sketch abstraction, which relies on the same stroke-sequence-based representation to work in a reinforcement learning framework as well.

Li *et al.* [18] proposed a sequence-to-sequence variational autoencoder (VAE) model for sketch grouping by modeling strokes sequentially, which is the most similar work to ours. However, unlike our network, their model is designed for stroke clustering that is unable to assign strokes with semantic part labels. Moreover, ours is trained in an end-to-end manner which might be more convenient for segmentation and grouping, or later usage in other applications. While the direct output of their model is an affinity matrix, which requires an additional post-processing of clustering for obtaining grouping results.

III. DATASET

This section presents our proposed stroke-level sketch segmentation (**SketchSeg-150K**) dataset. After manually labeling a limited number of sketches into semantic parts for each selected category, a strategy of learning-based sketch generation is used for largely enriching the dataset which can save huge and expensive efforts on annotation. This results in a large scale sketch semantic segmentation dataset which contains over 150K sketches over 20 classes is presented.

A. CATEGORY SELECTION

Our dataset is built upon *QuickDraw* [14], which contains 345 sketch object classes and is the largest doodling dataset up to date. 20 categories are selected according to the level of overall structural complexity of each class. In our case, more parts basically mean more complex structure of sketch drawing. According to this principle, 20 categories are selected, from simpler cases (like ''spoon'' and ''ice-cream'' with only two components) to some complicated ones (such as ''angel'' with more parts). Afterwards, we manually pick out a relatively small number of representative sketches for each of the selected 20 categories, and then manually assign each stroke with a semantic part label. These representative sketches are essentially the cluster centers in the feature space of a sketch object class. Some examples of labeled sketches can be found in Fig.1.

B. DATA AUGMENTATION

To further enrich the part labeled sketch drawings, a generative model is trained for data augmentation so that an exponentially increased number of annotated sketches with larger variety of appearances can be obtained with only very little manual efforts.

In particular, a Sketch-RNN [14] model is firstly trained for a specific category. Next, labeled sketch is used as input to the Sketch-RNN to generate several new sketches, which have different drawing appearances with the same label as shown in Fig.2. This is because the generated strokes are precisely in the same order as the sequence of input strokes. More specifically, given the learned Sketch-RNN model, the parameter *temperature*[1] is used to control the variety of generated sketches, thus more sketches with different level of complexity can be obtained. In our case, we set temperature with a range from 0.10 to 0.48 with a step of 0.02 and at each temperature, we generate 50 new sketches from each of the manually labeled sketch. The quality of the generated sketch is quite satisfied and we importantly verify the correctness of annotations manually by visualizing them just as shown in Fig.2. Finally, we obtain the stroke-level sketch segmentation dataset containing about 150K sketches with semantic part labels in total.

C. DATA FORMAT

We now describe the data format of the sketch collections and their labels in our dataset.

1) SKETCH COLLECTIONS

Similar to [14], we represent each of the sketch S in the dataset as a set of pen stroke status. More specifically, a sketch contains a sequence of points, and each point S_i is a 5-dimensional vector $[\Delta x_i, \Delta y_i, p_1, p_2, p_3]$. $d_i = [\Delta x_i, \Delta y_i] \in \mathbb{R}^2$ denotes the offset distance of the pen from the previous point in axles x and y. $p_i = [p_1, p_2, p_3] \in \mathbb{R}^3$ is a binary one-hot vector that denotes three states of the drawing pen: $p_1 = 1$ indicates the pen is touching the paper, otherwise $p_1 = 0$. Similarly, $p_2 = 1$ denotes the pen to be lifted from the paper at the next point, and $p_3 = 1$ indicates the end of a drawing.

2) LABELS

Given the predefined sketch object parts, each of the point of a sketch drawing is assigned with a semantic part label. In our case, the label of any point is denoted as a one-hot vector to be used during the training and testing stage. More specifically, in our case there are 57 part labels defined over the collected

[1]We refer reader to [14] for more details.

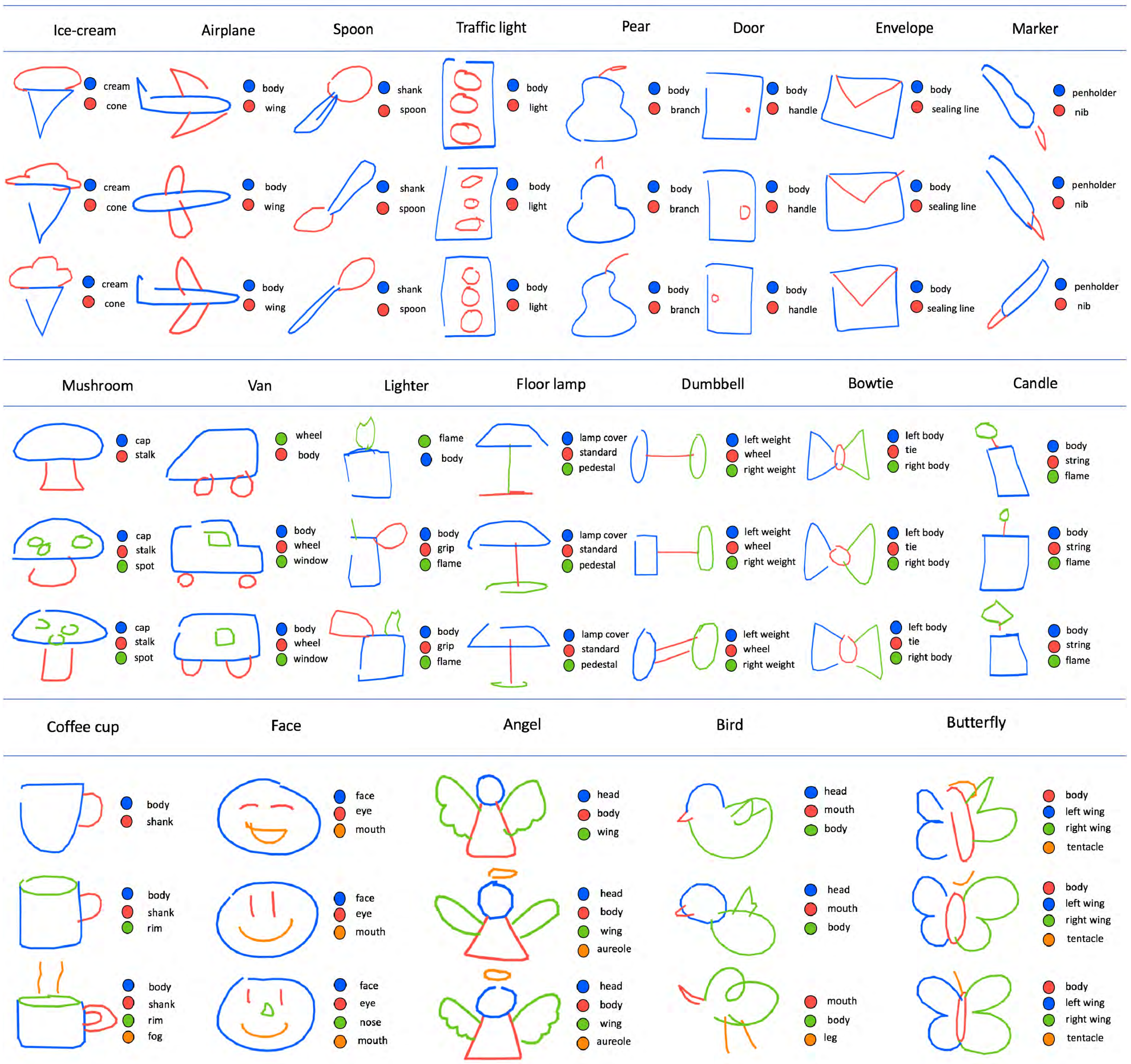

FIGURE 1. Examples of labeled sketches. We predefined the possible labels according to their parts for each of the sketch category, such as ice-cream can be constructed by cream and a cone, and an angel consists of four possible parts: Head, body, wings and aureole.

20 sketch classes, therefore a 57-dimensional one-hot vector is used to represent each label. For instance, assuming "head", "body", "wing" and "aureole" are components of "angel", where one of their corresponding indexes in the 57d one-hot vector will be set to 1 according to the annotation of the stroke.

IV. METHODOLOGY

The overview of our approach is illustrated in Fig. 3. As shown in the figure, the problem of multi-class sketch semantic segmentation is treated as a sequence-to-sequence generative learning problem. In particular, this is achieved by (i) training a bidirectional RNN to encode an input sketch S (a set of points) into a latent vector z, (ii) learning an autoregressive RNN to predict semantic label L_i for the next point by summarizing the information from previous observations in a sufficient statistic h_{i-1}. Our network is trained end-to-end in a supervised manner via back propagation. The network details are described in the following.

A. NETWORK ARCHITECTURE

As is shown in Fig. 3, the proposed network is a VAE-based sequence-to-sequence model, where a bidirectional LSTM module and an autoregressive LSTM module serve as the encoder and the decoder, respectively.

1) ENCODER

The encoder is a BLSTM [21] that takes a sketch sequence S and its reversely ordered sequence $S_{reverse}$ as input, and

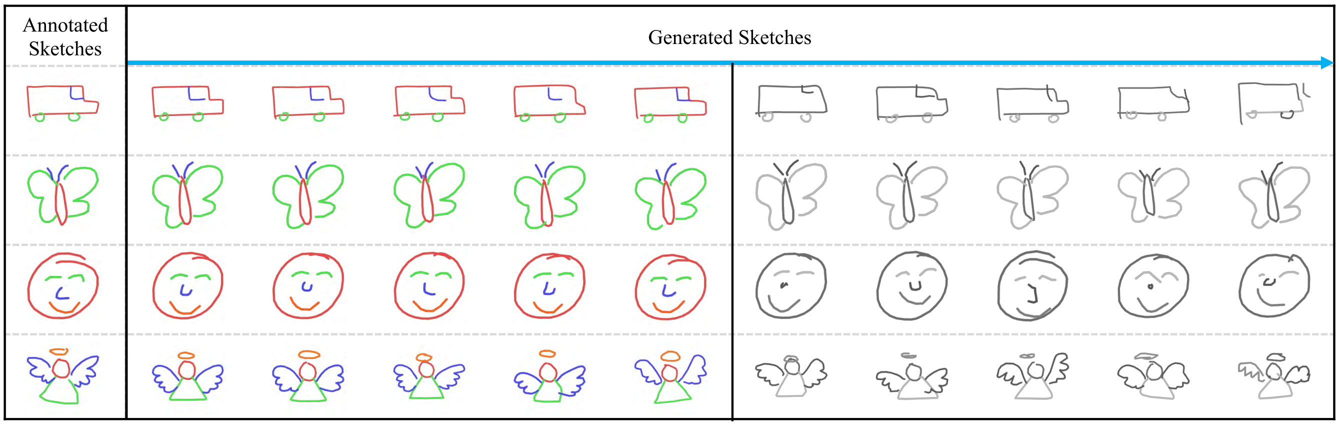

FIGURE 2. Illustration of data augmentation from manually labeled sketches. The same color indicates the same semantic concept. The blue arrow denotes an increase of the parameter "temperature" of Sketch-RNN. We can see that the augmented sketches (Colored) exhibit different varieties of strokes comparing with the input annotated sketches, and also preserve perfect labeling. We abandon the generated sketches (Gray) which are too different.

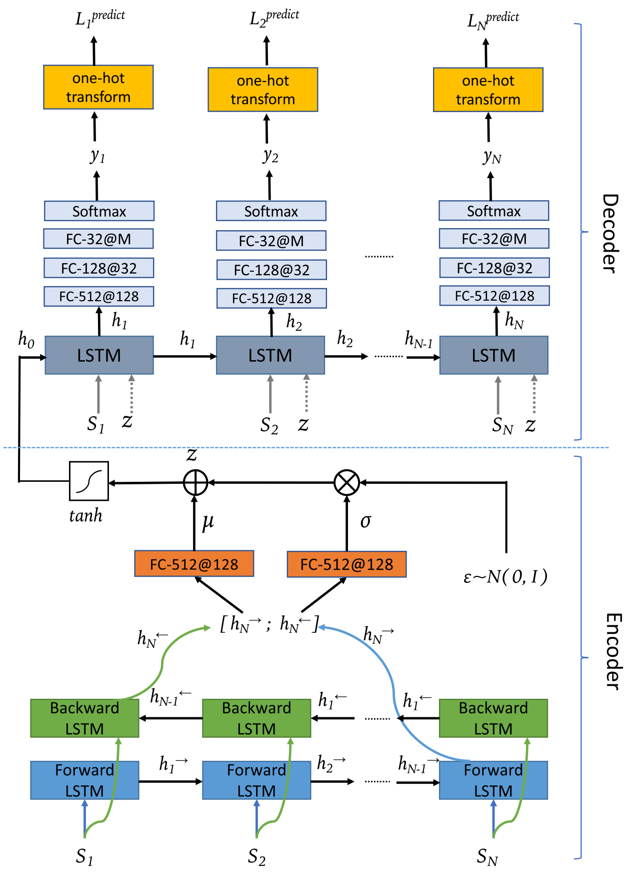

FIGURE 3. Network Architecture. The encoder is a BLSTM which takes a stroke sequence $S = [S_1, S_2 \ldots, S_N]$ as input, and outputs a latent vector z. The decoder is an unidirectional LSTM with three fully connected layers and a softmax layer to estimate label of each stroke.

outputs a latent vector z as a representation for the input sketch. The use of BLSTM enables the encoder to capture long-term dependencies of strokes.

In particular, for an input sketch in the format of a stroke sequence $S = [S_1, S_2 \ldots, S_N]$, a forward hidden state sequence is generated, during the forward propagation process of BLSTM, as follows:

$$h_{forward} = [\overrightarrow{h_1}, \overrightarrow{h_2}, \ldots, \overrightarrow{h}_N] \tag{1}$$

Simultaneously, the reverse sequence $S_{reverse} = [S_N, S_{N-1} \ldots, S_1]$ is also fed into the encoder, generating a backward hidden state sequence

$$h_{backward} = [\overleftarrow{h_1}, \overleftarrow{h_2}, \ldots, \overleftarrow{h}_N] \tag{2}$$

Afterwards, the two very last hidden states $\overrightarrow{h_N}$, $\overleftarrow{h_N}$ are concatenated together as a joint hidden state:

$$h = [\overrightarrow{h_N};\ \overleftarrow{h_N}] \tag{3}$$

Then the joint hidden state h is fed into two separate fully connected layers to form vectors μ and σ of the same size $N_z = 128$ in our case. Essentially, μ is the mean of the posterior distribution $p_z(z|X)$ learned by the encoder and σ is standard deviation.

To this end, the latent vector z can be formed by $z = \mu + \sigma \cdot \epsilon$ which is a vector of independent and identically distributed Gaussian variables of size N_z, where $\epsilon \sim \mathcal{N}(0, I)$ is a standard Gaussian noise.

2) DECODER

To model the distribution of the label of each stroke sequence, we adopt a unidirectional LSTM [22] to generate labels under the condition of latent variable z given by the encoder. The initial hidden state h_0 and cell state c_0 are computed through a hyperbolic tangent function:

$$h_0 = tanh(W_h z + b_h)$$

$$c_0 = tanh(W_c z + b_c) \tag{4}$$

Note that the first input of the decoder is the first point S_1 of the input sketch S, which is different from Sketch-RNN [14] which takes $S_0 = [0, 0, 1, 0, 0]$ as initial point. At each time-step t, the point vector S_i is concatenated with the latent vector z and serves as the input vector for LSTM:

$$x_t = [S_t; z] \tag{5}$$

Accordingly, the input gate i_t, forget gate f_t, output gate o_t, cell state c_t and hidden state h_t are defined as follows:

$$\begin{aligned} i_t &= sigmoid(W_i x_t + U_i h_{t-1} + b_i) \\ f_t &= sigmoid(W_f x_t + U_f h_{t-1} + b_f) \\ o_t &= sigmoid(W_o x_t + U_o h_{t-1} + b_o) \\ \tilde{c_t} &= tanh(W_c x_t + U_c h_{t-1} + b_c) \\ c_t &= i_t \odot \tilde{c_t} + f_t \odot c_{t-1} \\ h_t &= o_t \odot tanh(c_t) \end{aligned} \tag{6}$$

After all hidden states $h_d = [h_0, h_1, \ldots, h_N]$ are obtained by equations above, they are used for part label prediction for each input point. More specifically, h_d is fed into three fully connected layers followed by a softmax layer for classification. The output of softmax layer is denoted as $Y = [Y_1, Y_2 \ldots, Y_N]$, where $Y_i = [y_1, \ldots, y_n]$, $\sum_{j=1}^{n} y_j = 1$, $n = 57$ is the number of possible annotations for sketch parts in our case. Hence, y_j is the probability of the input point belonging to the corresponding $j - th$ entry in the label searching space. Afterwards, one-hot encoding is performed to encode Y_i into a one-hot vector L_i^{pre}, that all the elements are 0 but one and only one being 1 that indicates the label predicted.

$$L_i^{pre} = \delta(Y_i) \tag{7}$$

B. LOSS

A mean squared error (MSE) for VAE is applied for training our network, which directly optimizes for label prediction by minimizing the distance between the predicted labels and ground truth. It is formally defined as:

$$Loss_{MSE} = \frac{1}{n} \sum_{i=1}^{N} | L_i^{gt} - L_i^{pre} |^2 \tag{8}$$

where L_i^{pre} is the estimated semantic part label and L_i^{gt} is the ground truth. $L_i^{gt} = [l_1, l_2, \ldots, l_n] \in \mathbb{R}^n (n = 57)$ is a one-hot vector indicating which part the input data point belongs to. In this way, our proposed sequence-to-sequence model can be trained to generate a label for each input data point of sketch strokes at the test stage.

V. EXPERIMENTS

In this section, we first compare our method against with two state-of-the-art sketch grouping algorithms for sketch grouping (a related task), universal perceptual grouping [18] and perceptual grouping [11] are served as alternative competitors on the sketch grouping task. Then we report the results of sketch semantic segmentation obtained by our approach.

A. DATASET

Two datasets are used for evaluation on both sketch grouping and sketch semantic segmentation, including our *SketchSeg-150K* dataset and the *SPG* dataset proposed in [18]. There are 20 categories in the *SketchSeg-150K* dataset, and data for each category is split into two equal sets for training and testing. As for the *SPG* dataset, we follow the same setting of data split, namely 700 sketches for training and the rest 100 for testing for each category.

B. IMPLEMENTATION DETAILS

In our case, an input sketch is converted by the encoder into a 128-dimensional latent vector z, hence can be further used to predict the stroke labels by the decoder. The dropout probability for both the encoder and decoder are set to 0.9. We set the learning rate to 10^{-3} with a batch size of 30 to train the model and Adam optimizer [23] is applied for optimization. The learning rate decay is set to 0.9999 and the minimum learning rate is 10^{-5}. The implementation of our model is based on Pytorch and we train the model on a single GeForce GTX 1080Ti. The average time for training a model is about 2 hours.

C. EVALUATION METRIC

Two strategies are used for evaluation, one is *sketch grouping*, which measures if two data points are correctly grouped together. Note that it does not necessarily require to assign semantic part label in this case. The other is *sketch semantic segmentation* that evaluates if the part label of each data point is correctly predicted.

1) SKETCH GROUPING

Following [18], we use three metrics for measuring the performance of sketch grouping, including variation of information (**VOI**), probabilistic rand index (**PRI**), and segmentation covering (**SC**) as original defined in [24]. They are used in the context of sketch grouping, and the detailed definitions are: **VOI** measures the distance of average conditional entropy calculated between two groups, i.e., the predicted and ground truth groups. A smaller number indicates a better grouping result. **PRI** is the compatibility of assignments between pairs of stroke segments in each group. **SC** examine the overlapping between the predicted result of grouping and the ground truth.

2) SKETCH SEMANTIC SEGMENTATION

Two metrics proposed by Huang *et al.* [7] are used for evaluation, which can complement each other to measure the accuracy of sketch segmentation. The first is *P-metric*, i.e., stroke-based Accuracy (**SA**), which is a pixel-level measurement that measures the percentage of the offsets

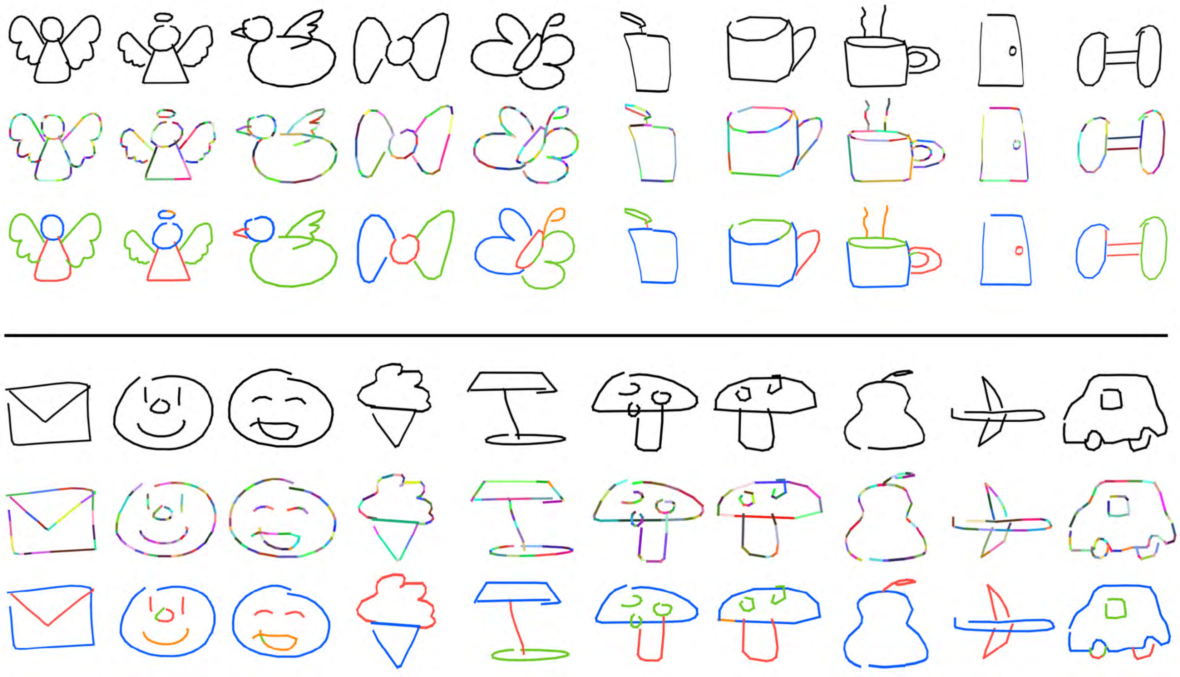

FIGURE 4. Example visualization of before and after grouping by our model. Top: Input sketch with black lines. Middle: Color coded strokes before grouping. Bottom: Strokes after grouping. (Best viewed in color, as strokes and grouping results are color coded.)

or pixels that are assigned with the correct labels for a sketch. A threshold of 0.8 is set, which means a stroke is corrected labeled only if over 80% of it's pixels are correctly labeled. The second is *C-metric*, i.e., component-based Accuracy (**CA**), which evaluates how well the sketch components or parts can be found. Specifically, one part is correctly discovered if 80% of its strokes are marked with the correct part label in our case.

D. COMPETITORS

As there are no existing multi-class sketch object segmentation approaches, two state-of-the-art models on sketch grouping are used as competitors, including perceptual edge grouping (**Edge-PG**) [11] and universal sketch perceptual grouping (**USPG**) [18]. **Edge-PG** is a non-deep learning method, which formulates the problem of grouping as a graph-cut optimization problem. Specifically, two gestalt principles, namely proximity and continuity, are combined by a learning-to-rank algorithm (RankSVM) to form the affinity matrix. Then the optimal grouping can be obtained by optimizing a graph partition problem given by the affinity matrix. **USPG** is the first deep neural network based approach for sketch grouping. They proposed a variant of the sequence-to-sequence variational autoencoder (VAE), which is able to obtain a representation of each sketch stroke effective for grouping. The representation thus can be used to compute a stroke affinity matrix suggesting the possibilities of grouping every pair of strokes. Different from our end-to-end approach, **USPG** requires an additional clustering step applied to the obtained stroke affinity matrix to output the final grouping result.

E. RESULTS AND ANALYSIS

1) QUALITATIVE RESULTS

Fig. 4 illustrates some examples of grouping results by our model. We can observe that the overall results of segmentation are quite close to the ground truth, despite some inaccurate labeled strokes exist. We further demonstrate some example grouping results of competitors for comparison, which are shown in Fig. 5 and Fig. 6. It is clearly observed that ours outperforms others with finer grouping strokes. **USPG** and **Edge-PG** might break an object part into segments, such as "Angel", "Dumbbell" and "Ice cream" in the dataset *SketchSeg-150K* or unify strokes together which belong to different groups, such as "Airplane" in the dataset *SPG*. In addition, ours can perfectly group strokes belong to the same group but geometrical far apart as humans do, such as eyes of "Pig" and bottons of "Caculator" as shown in Fig. 6.

2) QUANTITATIVE RESULTS

Quantitative results are shown in Table 1 and Table 2. We can observe from Table 1 that, the performance of our model clearly outperforms the other two competitors over all the 20 sketch categories of the proposed *SketchSeg-150K* dataset regards to the sketch grouping metric. In addition, we can achieve a 89% accuracy of pixel-level grouping (**SA**) and a 87% accuracy of part-level grouping (**CA**) evaluating on the metric of semantic sketch segmentation. Furthermore, we can obtain a similar results on the SPG dataset [18]. Specifically,

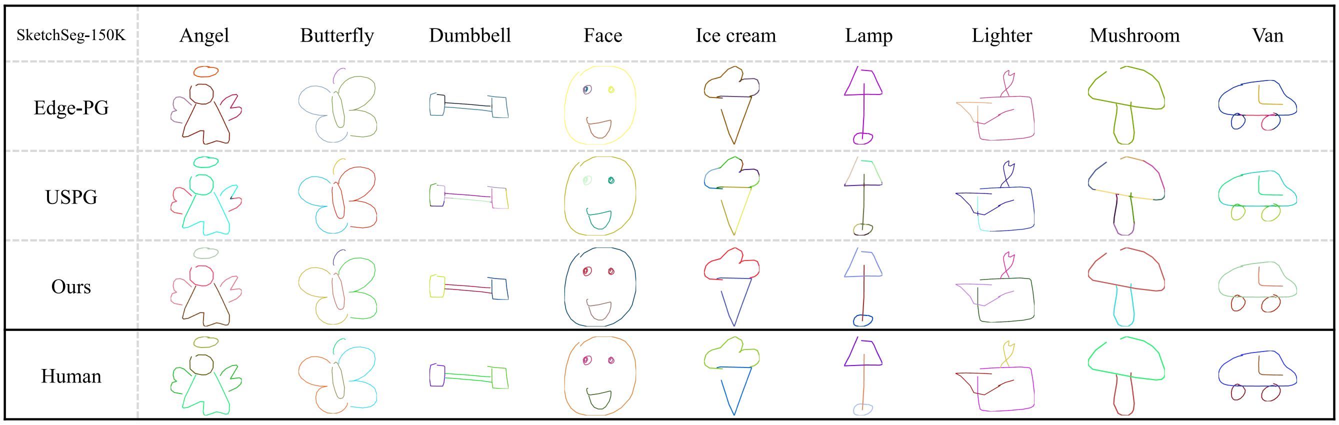

FIGURE 5. Comparison of example grouping results on SketchSeg-150K dataset. Strokes are color coded, where the same color denotes the same group.

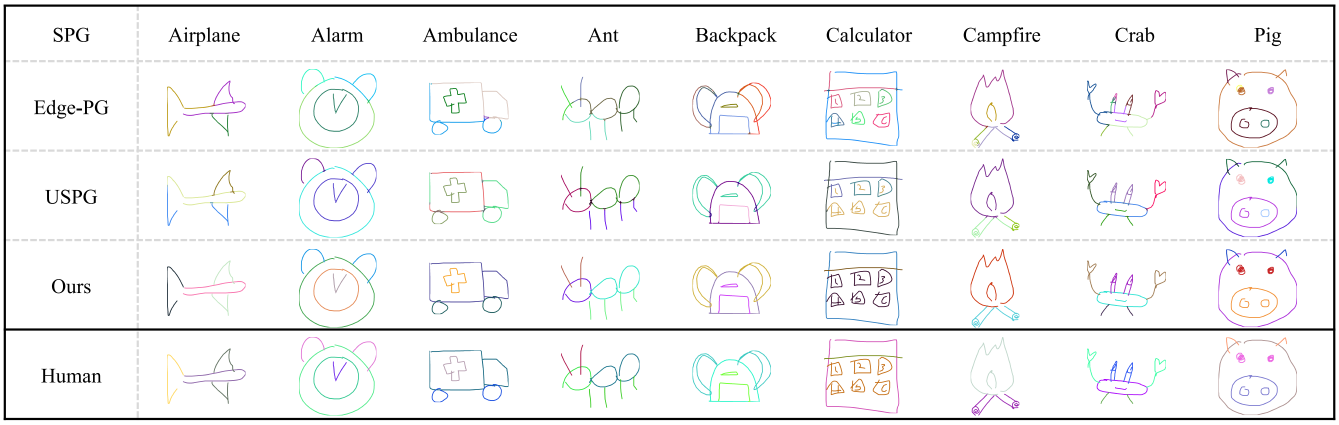

FIGURE 6. Comparison of example grouping results on SPG dataset.

TABLE 1. Comparative results on dataset sketchseg-150K. '-': Not Available. '↑': Larger is Better. '↓': Lower is Better.

Category \ Method	Edge-PG [13]					USPG [20]					Ours				
	VOI ↓	PRI ↑	SC ↑	SA ↑	PA ↑	VOI ↓	PRI ↑	SC ↑	SA ↑	PA ↑	VOI ↓	PRI ↑	SC ↑	SA ↑	PA ↑
Plane	1.06	0.77	0.64	-	-	0.97	0.80	0.65	-	-	**0.24**	**0.95**	**0.88**	**0.86**	**0.85**
Marker	0.84	0.75	0.73	-	-	1.81	0.55	0.45	-	-	**0.48**	**0.86**	**0.77**	**0.61**	**0.55**
Ice cream	1.02	0.65	0.63	-	-	1.51	0.70	0.52	-	-	**0.40**	**0.91**	**0.82**	**0.72**	**0.69**
Spoon	0.56	0.82	0.81	-	-	1.81	0.55	0.45	-	-	**0.06**	**0.98**	**0.97**	**0.85**	**0.81**
Traffic light	0.39	0.91	0.85	-	-	1.82	0.74	0.56	-	-	**0.04**	**0.99**	**0.98**	**0.96**	**0.96**
Pear	0.37	0.87	0.87	-	-	2.00	0.41	0.40	-	-	**0.07**	**0.98**	**0.97**	**0.99**	**0.98**
Door	0.08	0.97	0.97	-	-	1.82	0.42	0.43	-	-	**0.01**	**0.99**	**0.99**	**0.99**	**0.99**
Envelope	1.04	0.64	0.62	-	-	1.77	0.60	0.45	-	-	**0**	**1**	**1**	**1.00**	**0.99**
Mushroom	0.89	0.75	0.68	-	-	1.44	0.69	0.53	-	-	**0.32**	**0.92**	**0.85**	**0.70**	**0.66**
Van	1.41	0.68	0.59	-	-	0.83	0.81	0.71	-	-	**0.24**	**0.94**	**0.90**	**0.87**	**0.84**
Lighter	1.12	0.71	0.64	-	-	1.36	0.72	0.55	-	-	**0**	**1**	**1**	**0.99**	**0.98**
Lamp	1.38	0.63	0.48	-	-	1.38	0.77	0.55	-	-	**0.05**	**0.99**	**0.95**	**0.95**	**0.94**
Dumbbell	1.24	0.62	0.53	-	-	1.21	0.79	0.58	-	-	**0.01**	**0.99**	**0.99**	**0.99**	**0.99**
Bowtie	1.05	0.71	0.62	-	-	1.44	0.75	0.52	-	-	**0**	**1**	**1**	**0.99**	**1.00**
Candle	0.66	0.81	0.76	-	-	1.40	0.66	0.54	-	-	**0.24**	**0.94**	**0.90**	**0.83**	**0.69**
Angel	1.34	0.74	0.55	-	-	1.30	0.74	0.50	-	-	**0.14**	**0.99**	**0.93**	**0.89**	**0.86**
Bird	1.61	0.71	0.54	-	-	0.99	0.78	0.63	-	-	**0.01**	**0.99**	**0.99**	**0.98**	**0.97**
Butterfly	1.06	0.78	0.65	-	-	0.83	0.82	0.67	-	-	**0.01**	**1**	**1**	**0.95**	**0.95**
Coffee cup	1.20	0.70	0.59	-	-	1.29	0.74	0.55	-	-	**0.28**	**0.95**	**0.87**	**0.77**	**0.74**
Face	0.07	0.98	0.97	-	-	0.88	0.79	0.69	-	-	**0.04**	**0.99**	**0.98**	**0.94**	**0.91**
Average	0.94	0.75	0.68	-	-	1.36	0.69	0.55	-	-	**0.13**	**0.97**	**0.94**	**0.89**	**0.87**

our method achieve the best over all classes except for "Alarm clock", "Apple", "Backpack", "Campfire" and "Coffee cup", but ours are quite close to the best score on these categories, e.g., 0.46 vs 0.57 (VOI), 0.93 vs 0.91 (PRI) and 0.83 vs 0.81 (SC) for "Alarm clock", or a better score is obtained when it comes to a single metric, e.g., PRI score

TABLE 2. Comparative results on dataset SPG. '–': Not Available. '↑': Larger is Better. '↓': Lower is Better.

Method / Category	Edge-PG [13]					USPG [20]					Ours				
	VOI $\downarrow$	PRI $\uparrow$	SC $\uparrow$	SA $\uparrow$	PA $\uparrow$	VOI $\downarrow$	PRI $\uparrow$	SC $\uparrow$	SA $\uparrow$	PA $\uparrow$	VOI $\downarrow$	PRI $\uparrow$	SC $\uparrow$	SA $\uparrow$	PA $\uparrow$
Airplane	0.72	0.80	0.71	-	-	0.58	0.88	0.78	-	-	**0.42**	**0.94**	**0.85**	**0.75**	**0.68**
Alarm clock	0.59	0.84	0.73	-	-	**0.46**	**0.93**	**0.83**	-	-	0.57	0.91	0.81	**0.66**	**0.64**
Ambulance	1.35	0.67	0.60	-	-	0.67	0.86	0.77	-	-	**0.43**	**0.95**	**0.84**	**0.71**	**0.69**
Ant	1.32	0.68	0.62	-	-	0.86	0.83	0.69	-	-	**0.32**	**0.97**	**0.87**	**0.72**	**0.70**
Apple	0.54	0.88	0.79	-	-	**0.25**	0.92	**0.91**	-	-	0.31	**0.94**	0.90	**0.88**	**0.82**
Backpack	1.29	0.70	0.61	-	-	**0.57**	0.88	**0.79**	-	-	0.65	**0.92**	0.78	**0.62**	**0.60**
Basket	1.27	0.71	0.59	-	-	0.76	0.84	0.74	-	-	**0.38**	**0.96**	**0.86**	**0.73**	**0.75**
Butterfly	1.30	0.69	0.58	-	-	0.83	0.76	0.65	-	-	**0.24**	**0.97**	**0.91**	**0.86**	**0.78**
Cactus	0.86	0.82	0.71	-	-	0.51	0.90	0.83	-	-	**0.29**	**0.96**	**0.90**	**0.83**	**0.80**
Calculator	0.98	0.77	0.68	-	-	0.50	0.86	0.83	-	-	**0.17**	**0.97**	**0.94**	**0.91**	**0.92**
Campfire	1.05	0.71	0.65	-	-	**0.28**	0.95	**0.91**	-	-	0.33	**0.96**	0.89	**0.83**	**0.78**
Candle	1.47	0.65	0.57	-	-	0.89	0.78	0.69	-	-	**0.37**	**0.93**	**0.87**	**0.81**	**0.74**
Coffee cup	0.85	0.83	0.68	-	-	**0.38**	0.91	**0.86**	-	-	0.57	**0.91**	0.81	**0.72**	**0.72**
Crab	1.29	0.69	0.56	-	-	0.69	0.81	0.74	-	-	**0.39**	**0.96**	**0.86**	**0.77**	**0.70**
Duck	0.95	0.74	0.68	-	-	0.86	0.83	0.69	-	-	**0.41**	**0.96**	**0.87**	**0.74**	**0.69**
Face	1.24	0.69	0.61	-	-	0.81	0.84	0.74	-	-	**0.41**	**0.93**	**0.86**	**0.75**	**0.66**
Ice cream	0.79	0.82	0.71	-	-	0.41	0.94	0.85	-	-	**0.34**	**0.95**	**0.89**	**0.80**	**0.75**
Pig	1.55	0.63	0.50	-	-	0.63	0.84	0.78	-	-	**0.41**	**0.95**	**0.85**	**0.70**	**0.69**
Pineapple	0.63	0.83	0.72	-	-	0.50	0.93	0.82	-	-	**0.48**	**0.94**	**0.84**	**0.77**	**0.75**
Suitcase	0.58	0.82	0.75	-	-	0.54	0.89	0.83	-	-	**0.44**	**0.92**	**0.84**	**0.75**	**0.69**
Average	1.03	0.75	0.65	-	-	0.59	0.87	0.79	-	-	**0.39**	**0.94**	**0.86**	**0.77**	**0.73**

on "Apple". For the sketch semantic segmentation results, lower scores are obtained on the *SPG* dataset compared with the results on the *SketchSeg-150K* dataset. This is because the complexity of sketch objects between these two datasets is different.

VI. CONCLUSION

In this paper, we presented a novel sequence-to-sequence model of a encoder-decoder structure based on recurrent neural networks for sketch segmentation and labeling. Essentially, it learns from the human habit of stroke drawing orders as well as the contextual information among strokes in the encoder, and thus the decoder is able to determine the semantic labels of input strokes in sequence. Furthermore, our proposed method is an end-to-end framework for sketch semantic segmentation, and our method is able to work across multiple categories, which is of great importance for practical usage. In addition, a novel sketch segmentation dataset contains over 150K sketches was proposed based on QuickDraw for the first time. Experimental results validated the effectiveness of our proposed method.

ACKNOWLEDGMENT

The authors gratefully acknowledge the support of NVIDIA Corporation for the donation of the GPUs used for this research.

REFERENCES

[1] Y. Li, T. M. Hospedales, Y.-Z. Song, and S. Gong, "Intra-category sketch-based image retrieval by matching deformable part models," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, Nottingham, U.K., Sep. 2014.

[2] X. Wang and X. Tang, "Face photo-sketch synthesis and recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 11, pp. 1955–1967, Nov. 2009.

[3] J. Song, K. Pang, Y.-Z. Song, T. Xiang, and T. M. Hospedales, "Learning to sketch with shortcut cycle consistency," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2018, pp. 801–810.

[4] C. Zou, Q. Yu, R. Du, H. Mo, Y.-Z. Song, T. Xiang, C. Gao, B. Chen, and H. Zhang, "Sketchyscene: Richly-annotated scene sketches," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2018, pp. 438–454.

[5] L. Li, H. Fu, and C.-L. Tai, "Fast sketch segmentation and labeling with deep learning," *IEEE Comput. Graph. Appl.*, vol. 39, no. 2, pp. 38–51, Apr. 2018.

[6] Y. Qi, Y. Z. Song, H. Zhang, and J. Liu, "Sketch-based image retrieval via Siamese convolutional neural network," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 2460–2464.

[7] Z. Huang, H. Fu, and R. W. H. Lau, "Data-driven segmentation and labeling of freehand sketches," *ACM Trans. Graph.*, vol. 33, no. 6, Nov. 2014, Art. no. 175.

[8] Y. Choi, "Sketch-to-text generation: Toward contextual, creative, and coherent composition," in *Proc. 9th Int. Natural Lang. Gener. Conf.*, 2016, p. 40.

[9] R. K. Sarvadevabhatla, I. Dwivedi, A. Biswas, S. Manocha, and B. R. Venkatesh, "SketchParse: Towards rich descriptions for poorly drawn sketches using multi-task hierarchical deep networks," in *Proc. 25th ACM Int. Conf. Multimedia*, Oct. 2017, pp. 10–18.

[10] Y. Qi, J. Guo, Y.-Z. Song, T. Xiang, H. Zhang, and Z.-H. Tan, "Im2sketch: Sketch generation by unconflicted perceptual grouping," *Neurocomputing*, vol. 165, pp. 338–349, Oct. 2015.

[11] Y. Qi, Y. Z. Song, T. Xiang, H. Zhang, T. Hospedales, Y. Li, and J. Guo, "Making better use of edges via perceptual grouping," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1856–1865.

[12] M. A. M. Eitz and J. Hays, "How do humans sketch objects?" *ACM Trans. Graph.*, vol. 31, no. 4, pp. 44–51, Jul. 2012.

[13] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *Proc. 2nd Int. Conf. Learn. Represent. (ICLR)*, Banff, AB, Canada, Apr. 2014.

[14] D. Ha and D. Eck, "A neural representation of sketch drawings," 2017, *arXiv:1704.03477*. [Online]. Available: https://arxiv.org/abs/1704.03477

[15] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.

[16] A. Inselberg and B. Dimsdale, "Parallel coordinates for visualizing multi-dimensional geometry," *Comput. Graphics.*, vol. 1, pp. 25–44, 1987.

[17] R. G. Schneider and T. Tuytelaars, "Example-based sketch segmentation and labeling using CRFs," *ACM Trans. Graph.*, vol. 35, no. 5, Sep. 2016, Art. no. 151.

[18] K. Li, K. Pang, J. Song, Y.-Z. Song, T. Xiang, T. M. Hospedales, and H. Zhang, "Universal sketch perceptual grouping," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 582–597.

[19] X. Wu, Y. Qi, J. Liu, and J. Yang, "Sketchsegnet: A Rnn model for labeling sketch strokes," in *Proc. IEEE 28th Int. Workshop Mach. Learn. Signal Process. (MLSP)*, Sep. 2018, pp. 1–6.

[20] U. R. Muhammad, Y. Yang, Y.-Z. Song, T. Xiang, and T. M. Hospedales, "Learning deep sketch abstraction," 2018, *arXiv:1804.04804*. [Online]. Available: https://arxiv.org/abs/1804.04804

[21] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Netw.*, vol. 18, nos. 5–6, pp. 602–610, 2005.

[22] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," *ACM Trans. Graph.*, vol. 43, p. 1–20, Jul. 1999.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980

[24] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, May 2011.

YONGGANG QI received the Ph.D. degree in signal processing from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2015, where he is currently an Assistant Professor with the School of Information and Communication Engineering (SICE). He has published over ten papers on sketch understanding and its applications. His research interests include perceptual grouping, sketch-based tasks evolving SBIR, sketch recognition, sketch generation, and language-based sketch understanding.

ZHENG-HUA TAN (M'00–SM'06) received the B.Sc. and M.Sc. degrees in electrical engineering from Hunan University, Changsha, China, in 1990 and 1996, respectively, and the Ph.D. degree in electronic engineering from Shanghai Jiao Tong University (SJTU), Shanghai, China, in 1999.

He is currently a Professor with the Department of Electronic Systems and the Co-Head of the Centre for Acoustic Signal Processing Research (CASPR), Aalborg University, Aalborg, Denmark. He was a Visiting Scientist with the Computer Science and Artificial Intelligence Laboratory (CSAIL), Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, an Associate Professor with the Department of Electronic Engineering, SJTU, and a Postdoctoral Fellow with the Department of Computer Science, Korea Advanced Institute of Science and Technology, Daejeon, South Korea. His research interests include machine learning, deep learning, pattern recognition, speech and speaker recognition, noise-robust speech processing, multimodal signal processing, and social robotics. He has authored/coauthored about 200 publications in refereed journals and conference proceedings. He is a member of the IEEE Signal Processing Society Machine Learning for Signal Processing Technical Committee (MLSP TC). He is the General Chair of the IEEE MLSP 2018 and was a Technical Program Co-Chair of the IEEE Workshop on Spoken Language Technology (SLT 2016). He has served as an Editorial Board Member/Associate Editor for *Computer Speech and Language*, *Digital Signal Processing*, and *Computers and Electrical Engineering*. He was a Lead Guest Editor of the IEEE Journal of Selected Topics in Signal Processing and a Guest Editor of several journals, including *Neurocomputing*.