

ContextSeg: Sketch Semantic Segmentation by Querying the Context with Attention

Jiawei Wang
Shandong University

Changjian Li
The University of Edinburgh

<https://enigma-li.github.io/projects/contextSeg/contextSeg.html>

Abstract

Sketch semantic segmentation is a well-explored and pivotal problem in computer vision involving the assignment of pre-defined part labels to individual strokes. This paper presents ContextSeg—a simple yet highly effective approach to tackling this problem with two stages. In the first stage, to better encode the shape and positional information of strokes, we propose to predict an extra dense distance field in an autoencoder network to reinforce structural information learning. In the second stage, we treat an entire stroke as a single entity and label a group of strokes within the same semantic part using an auto-regressive Transformer with the default attention mechanism. By group-based labeling, our method can fully leverage the context information when making decisions for the remaining groups of strokes. Our method achieves the best segmentation accuracy compared with state-of-the-art approaches on two representative datasets and has been extensively evaluated demonstrating its superior performance. Additionally, we offer insights into solving part imbalance in training data and the preliminary experiment on cross-category training, which can inspire future research in this field.

1. Introduction

Sketches are widely used as a human-computer interaction tool. Many studies keep exploring its capabilities in modeling [13, 14], retrieval [2, 3] and generation [1, 5]. Among these tasks, sketch interpretation serves as the foundation. However, it still remains challenging for computers due to the inherent ambiguity and sparsity of user sketches [31]. In this paper, we focus on sketch semantic segmentation, an essential task in finer-level sketch interpretation.

Sketches are typically represented in three data formats, i.e., the raster image, graph and point sequence. Recently, many learning-based approaches have adopted the aforementioned sketch representations revealing various advantages and disadvantages (see the inset table). Specif-

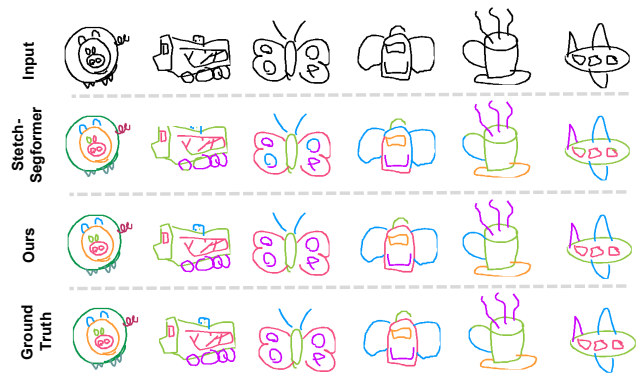


Figure 1. Given an input sketch, semantic segmentation is to assign labels to strokes based on their semantics so as to form semantic groups. Our method is robust to stroke variations achieving superior results (e.g., the correctly labeled airplane windows).

ically, image-based methods [16, 35, 36] take raster images as input, exploiting absolute coordinates to capture the proximity between strokes. However, they tend to perform poorly in learning

	Structure	Position	Proximity
Image	✗	✓	✓
Graph	✗	✓	✓
Sequence	✓	✗	✗
Ours	✓	✓	✓

the structural information about strokes and overlook the sequential relationships between them. Graph-based methods [29, 31, 34] take graphs as input, which consists of stroke points as nodes and point connections as edges. The graph representation also exploits absolute coordinates, which facilitate proximity learning between strokes but is challenging to capture the structural information of a stroke. Sequence-based methods [10, 15, 22, 30] take as input point sequences, which use relative coordinates to better encode structural information of individual strokes. However, they struggle to capture the proximity relationships and spatial information between strokes. Nevertheless, stroke structural and positional information and the proximity relationship between strokes are essential for sketch interpretation.

Motivated by the observations, we propose *ContextSeg*,

a two-stage approach marrying advantages of all three representations to capture better structural and positional information of each single stroke, and the proximity relationship between strokes. Specifically, at the stroke level, we treat an entire stroke instead of the stroke point as a single entity, represented as a raster image, where the positional information can be efficiently extracted by an autoencoder CNN network. Furthermore, due to the sparsity of stroke pixels, as well as the large geometric variation of a stroke depicting the drawing content, it is challenging to encode stroke structural information only by reconstruction. We thus propose a novel dense distance field regression task upon reconstruction, reinforcing structural information learning. At the sketch level, we treat all strokes within a sketch as a sequence of images. By augmenting the sequence with positional information, neural networks can well capture the proximity between strokes. As for the segmentation task, we exploit an auto-regressive Transformer, where instead of labeling one stroke at each time, we select a set of strokes belonging to the same semantic group at one time. Using the group-based prediction, context information, e.g., strokes possessing a part label and strokes remaining to be labeled, can be explicitly utilized by the Transformer decoder when making decisions for remaining strokes.

To demonstrate the effectiveness of our network, we have conducted experiments on two representative datasets, i.e., SPG [15] and CreativeSketch [9]. Experimental results including comparisons and ablation studies demonstrate our superior performance.

In summary, our main contributions are threefold:

- We propose a CNN-based network for stroke embedding learning, which features dense distance field prediction for capturing the structural information of a stroke.
- We exploit an auto-regressive Transformer network for segmentation, where we propose to label a group of strokes at one time fully leveraging context information leading to our SOTA performance.
- We propose a novel semantic-aware data augmentation mechanism attempting to address the data imbalance problem; and in benefiting from our network’s capability to extract and exploit contextual information, we achieved significant performance improvement in the challenging cross-category learning. Both the novel strategy and the preliminary experiment might inspire future research.

2. Related Work

Sketch Representation. In the field of computer vision, sketch representation learning has garnered widespread attention. It is a fundamental task in numerous downstream applications, i.e., sketch-based image retrieval (SBIR) [3, 24], sketch generation [23], and sketch classification [32]. Zhang et al. [32] introduced deep convolutional neural net-

works for learning sketch embedding. The network took triplets composed of sketches, positive real images, and negative real images as input to discover coherent visual structures between the sketch and its positive pairs. Sain et al. [24] employed a cross-modal VAE to disentangle sketches into shared semantic content and unique style contents, enhanced with meta-training for dynamic adaptation to unseen user styles, enabling style-agnostic SBIR. Furthermore, research has been conducted on the generalization properties of sketch embedding [1, 9, 17, 21].

Semantic Sketch Segmentation. Compared to previous works that relied on handcrafted features and complex models [6, 25, 26], many deep learning-based models have achieved outstanding performance in sketch semantic segmentation tasks [30, 31, 34]. Based on the data formats they use, existing methods can be broadly grouped into three classes: image-based [16, 35, 36], sequence-based [15, 22, 30], and graph-based [29, 31, 34] methods. Zhu et al. [35] proposed a dual-CNN approach for sketch segmentation and labeling, employing two networks with distinct kernel sizes to handle different sketch lengths, while enhancing performance through the integration of position and orientation as a triple-channel input using fused masks. Li et al. [15] proposed a sequence-based encoder-decoder architecture for sketch semantic segmentation, applying additional constraints in the loss function, specifically targeting reconstruction and global grouping consistency. Yang et al. [31] proposed a graph-based network and utilized two graph convolutional branches to extract the inter-stroke features and the intra-stroke features.

Vision Transformer. Other than natural language processing, Transformer [27] has been widely adopted in all kinds of computer vision tasks, e.g., classification [7, 8], detection [4], segmentation [19]. Unlike the auto-regressive decoding strategy, Carion et al. [4] proposed parallel decoding, given the absence of inherent order or sequence information between different bounding boxes in object detection tasks. Transformer was also adopted in the sketch domain, where Ribeiro et al. [23] encoded free-hand sketches in a vector format and effectively improved the performance of sketch-based image retrieval while Li et al. [14] treated sketch-based CAD modeling as a serializable translation problem and employed a Transformer-based network for stroke grouping.

3. Method

Given an input sketch \mathbf{S} represented by a sequence of strokes $\{s_i\}$, our goal is to segment them into groups based on their underlying semantics and assign the corresponding semantic label to the group of strokes. Figure 2 displays an overview of our method, which consists of two key modules, i.e., *stroke embedding* and *segmentation Transformer*.

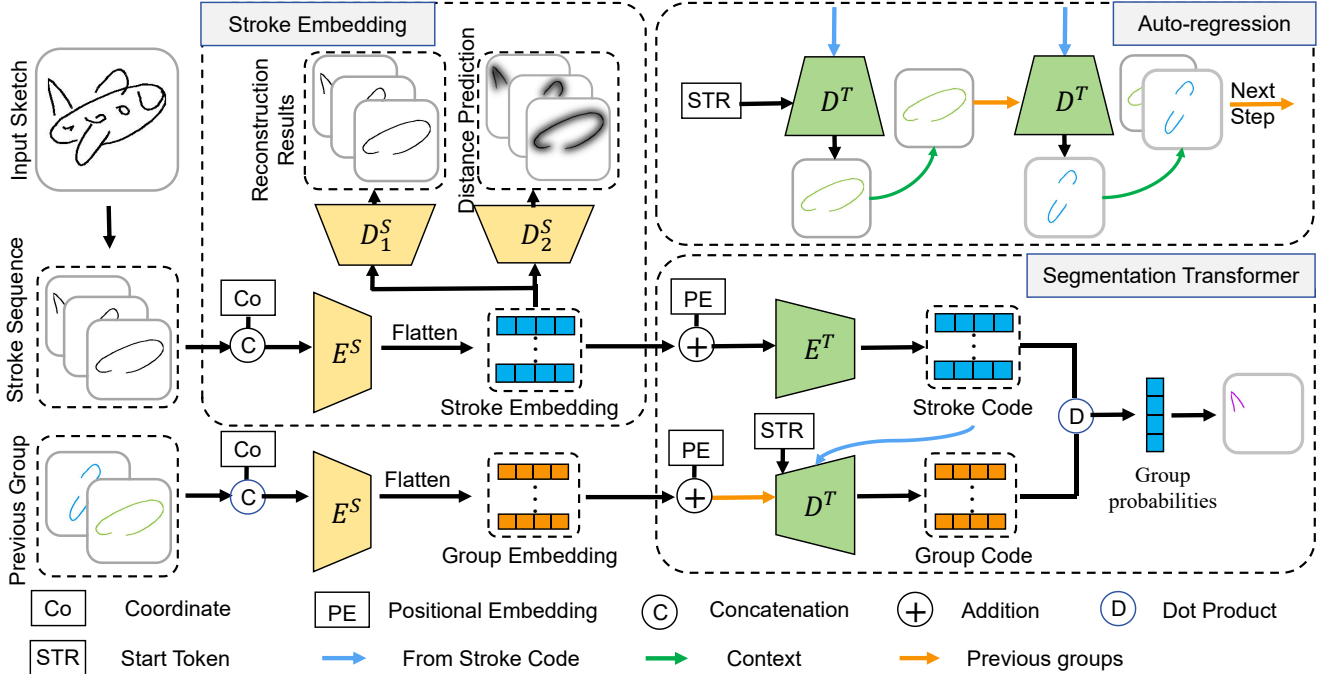


Figure 2. Overview of *ContextSeg*. Given an input sketch, it is first divided into a sequence of strokes, which are used to train our stroke embedding network – an autoencoder with an extra distance field output (Sec. 3.1). Then, the learned embeddings are sent to the segmentation Transformer operating in an auto-regressive manner (Sec. 3.2). The Transformer leverages contextual information, encompassing previously labeled strokes and remaining strokes, as input for the current step’s stroke labeling.

In what follows, we elaborate on the details.

3.1. Stroke Embedding

We employ a Transformer as our segmentation model, which accepts embedding vectors. To this end, given a stroke s_i , we first design our embedding learning network to obtain the corresponding stroke embedding s_i^e . Specifically, we represent each stroke s_i as a binary image $I(s_i) \in \{0, 1\}^{256 \times 256}$. Since we want the learned embedding to encode stroke positional information, we thus augment the input image channel with two additional coordinate channels following the practice in CoordConv [18]. Given the coordinate augmented stroke image, we leverage a 2DCNN-based autoencoder network as our backbone to extract stroke embedding. Both the encoder E^S and decoder D_1^S consist of a few 2DCNN layers with varying convolution kernels and feature dimensions. After the last layer of the encoder, we flatten the feature to obtain the embedding s_i^e . Note that, skip-connections are omitted for embedding learning. Ideally, through minimizing the disparity between the input $I(s_i)$ and the reconstructed image $D_1^S(E^S(I(s_i)))$, the trained network excels in reconstructing the input accurately and encoding the stroke effectively with high fidelity.

However, unlike photorealistic images, a sketch is usually harder to encode due to its sparsity and large variation of the compositional strokes. For example, some strokes might be cluttered in a small region due to over-sketching

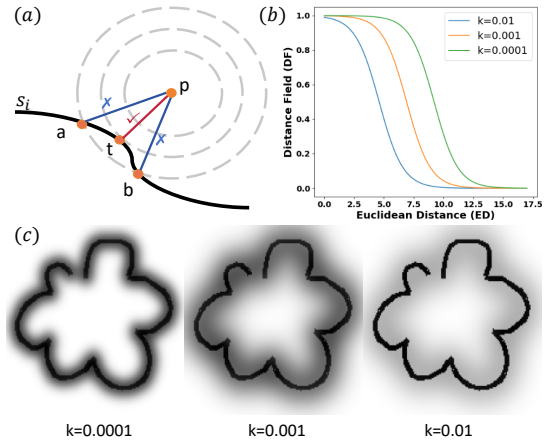


Figure 3. Stroke distance field. (a) Given an arbitrary point p in the image, we calculate the shortest Euclidean distance from p to the point t on the stroke. (b) The distance curves with three different k values. (c) Distance field maps of three typical k values.

(see the body of an ant in Fig. 4) and an individual stroke might exhibit large geometric variations based on the drawn content (see the wing of the angel in Fig. 5). These challenges fail the aforementioned baseline embedding network (Sec. 4.2, ablation study). Consequently, to address these difficulties, we design a novel distance field regression in addition to the reconstruction in the autoencoder.

Stroke Distance Field. Since we treat each stroke as an image $I(s_i)$, a natural choice to build the distance field is the

unsigned distance from arbitrary points in the image to the stroke. However, with a large region expanding the whole image into consideration, the structural information of the stroke is overlooked. We thus propose a novel distance field (dubbed as DF^s) concentrating only a narrow band around the stroke, defined over all the points on the image:

$$DF^s(s_i) = \frac{1}{1 + k \times e^{dis_{s_i}^p}} \quad (1)$$

where $dis_{s_i}^p$ calculates the shortest Euclidean distance from a query point p on the image to the stroke s_i (Fig. 3(a)), while k controls both the width of the narrow band as well as the decay rate of distance values after leaving the stroke (Fig. 3(b)). As shown in Fig. 3(c), we experimentally set k at 0.001, producing a dense field within an appropriate band and the surrounding values depicting the stroke variation.

The advantages of using DF^s are two-fold. On the one hand, by only attending the narrow band around the stroke with a smooth value decay, the structural information of the stroke is forced to be captured. On the other hand, the distance function maps any Euclidean distance to $[0, 1]$, which is easy to be regressed by neural networks.

Network Configuration and Training. We thus augment the baseline autoencoder with a second decoding branch (i.e., D_2^S in Fig. 2) to output a dense distance field respecting the input stroke. To train the embedding network, we minimize the following loss function:

$$\mathcal{L}_{em} = \mathcal{L}_{recon} + \gamma \mathcal{L}_{dis}, \quad (2)$$

where γ is a balancing weight, is set at 0.5 empirically. \mathcal{L}_{recons} is the image reconstruction loss, defined as:

$$\mathcal{L}_{recons} = \|I(s_i) - D_1^S(E^S(I(s_i)))\|^2, \quad (3)$$

and \mathcal{L}_{dis} is defined as:

$$\mathcal{L}_{dis} = \|DF^s(s_i) - D_2^S(E^S(I(s_i)))\|^2. \quad (4)$$

For any stroke image $I(s_i)$, we first calculate the distance map $DF^s(s_i)$ as a pre-processing step (see three examples in Fig. 3 (c)). After training the network, we simply discard the two decoders and fix the encoder to obtain the stroke embedding $s_i^e = E^S(I(s_i))$.

Embedding of a Group of Strokes. Other than the single strokes, we also encode a set of strokes corresponding to a semantic group, which is used in our segmentation Transformer. Specifically, suppose g_i is a group of strokes, and $I(g_i)$ is the corresponding image. We use all these group images as extra training data for the aforementioned embedding network training and obtain the group embedding $g_i^e = E^s(I(g_i))$ as same as for a single stroke.

3.2. Segmentation Transformer

As shown in Fig. 2, our segmentation Transformer is built upon [27], running in an auto-regressive manner with self- and cross-attention mechanisms in the encoder E^T and D^T . Specifically, given the set of stroke embeddings $\{s_i^e\}$, we first augment them with the sinusoidal positional embedding to help distinguish one from the others. Then, the encoder E^T takes as input a sequence of augmented stroke embeddings and applies a few self-attention layers to produce the set of stroke codes $\{s_i\}$, which servers as one of the inputs to the decoder D^T .

As for the decoding, we use the auto-regressive scheme, i.e., when making the decision of the current iteration step, the context information from the previous predictions serves as input to the decoder. The auto-regression will stop until nothing remains to be decided for the current step. In our case, the context information is the labeled strokes in the previous steps and the strokes remain to be labeled. In each decoding iteration, a possible solution is to assign a label to the current stroke. However, even with context information, this stroke-based decoding ignores the group information. Instead, inspired by the Pointer Networks [28], our decoder outputs a group code \mathbf{g}_j each time by considering the labeled groups of strokes as context, and the group code is leveraged as a pointer to select strokes belonging to the current semantic group by dot product comparison leading to a group probability. The selection process is defined as:

$$p_i^j = \sigma(s_i \cdot \mathbf{g}_j), \quad (5)$$

where (\cdot) is the dot produce, and σ is the sigmoid activation. In practice, when the probability p_i^j is greater than 0.5, we consider the stroke s_i belonging to the j -th group.

Network Training. To train the network, we can simply minimize the cross-entropy loss [33] applied to a group of strokes at one time. However, we observe that the training data has a severe imbalance in terms of the stroke distribution over all semantic groups, e.g., all the ants have the body and leg groups, each with many strokes, while only a few of them have the eye group with one or two strokes. We thus employ the focal loss to train our segmentation Transformer, defined as:

$$\begin{aligned} \mathcal{L}_{seg} = & \sum_{g_j} \sum_{s_i} -\mathbb{1}_{s_i \in g_j} (1 - p_i^j)^\gamma \log(p_i^j) \\ & - (1 - \mathbb{1}_{s_i \in g_j}) (p_i^j)^\gamma \log(1 - p_i^j), \end{aligned} \quad (6)$$

where $\mathbb{1}$ is the indicator function, and γ is the focusing parameter (2.0 in our case) in the modulating factor.

4. Experiments

In what follows, we present visual and statistical evaluations and provide implementation details in the supplementary.

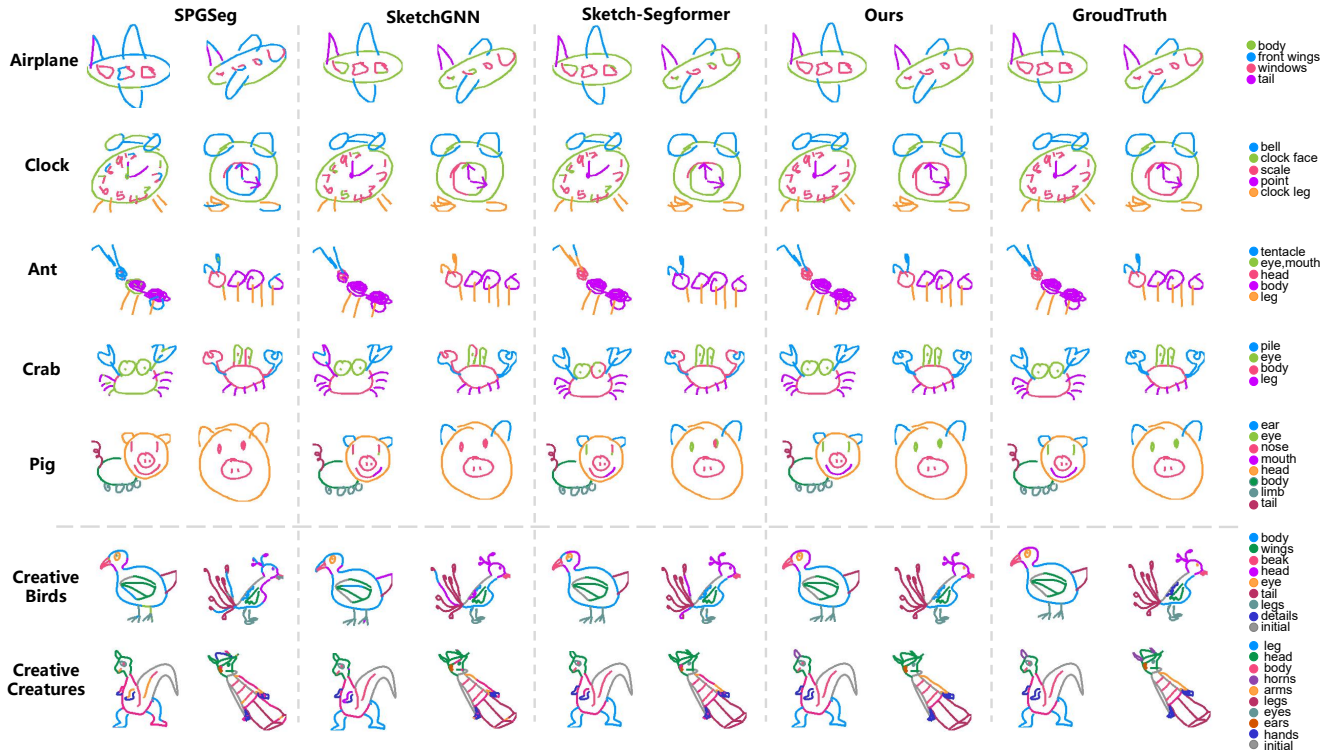


Figure 4. Visual comparison with three competitors on the SPG and the CreativeSketch datasets.

Datasets. Following existing work, we use both the SPG [15] and CreativeSketch [9] (dubbed as CS) datasets in this paper. Specifically, the SPG dataset contains sketches from 25 categories, each with 800 sketches. As in [15, 31], we use the same 20 selected categories and employ the same data splitting as described in [30], i.e., 700 sketches for training, 100 sketches for testing. We use all the training data from 20 categories for the embedding network training, while the segmentation Transformer is trained per category. As described in Sec. 3.1, we collected both single strokes and groups of strokes to train the embedding network. As for the CS dataset, it consists of 2 categories with much more complex sketches forming 8 and 17 semantic parts, respectively. We then randomly select 3000 sketches from each category with a 2500-500 training and testing split. Similarly, we use all 5000 sketches from both categories to train the embedding networks, while training the segmentation Transformer per category. All the sketches are scaled to fit into 256×256 images.

Evaluation Metrics. We have adopted three metrics [11, 14, 30] to evaluate performance, i.e., stroke accuracy (SAcc), grouping accuracy (GAcc) and component accuracy (CAcc), and their definitions can be found in the supplementary. Intuitively, stroke accuracy provides a fine-grained evaluation inspecting each stroke, while both the grouping accuracy and component accuracy quantify the semantic-based accuracy of all strokes. Compared with

GAcc, CAcc is more vulnerable to the long tail problem regarding the stroke distribution over semantic groups.

4.1. Comparison

We compare our approach with three SOTA methods, i.e., SPGSeg [15], SketchGNN [31] and Sketch-Segformer [34]. All three methods are trained from scratch using their default parameters on our dataset. In the following, we introduce the quantitative and qualitative results.

Quantitative comparisons are shown in Tab. 1, wherein we obtain the best average and per-category results (except for the face category) over all three metrics on both the SPG and CS datasets. Specifically, in terms of component accuracy, our approach achieved notable advancements of 1.4%, 1.7%, and 18.7% compared to Sketch-Segformer, SketchGNN, and SPGSeg, respectively, on the SPG dataset. Also, on the more challenging CS dataset, substantial improvements in component accuracy were observed, reaching 11.9%, 13.3%, and 22.9% in comparison to Sketch-Segformer, SketchGNN, and SPGSeg, respectively. Regarding the face category, our method demonstrates comparable accuracies, albeit slightly lower. This outcome is attributed to the presence of highly overlapped strokes, particularly representing hairs and eyes, which pose challenges to precise embedding learning and subsequently impact grouping predictions. In Fig. 4, qualitative results are showcased, illustrating our method’s capability to successfully identify strokes delineating intricate semantic details.

Table 1. Statistical comparison with SOTA methods on the SPG and CreativeSketch datasets, all three metrics are reported.

	Category	SPGSeg[15]			SketchGNN[31]			Sketch-Segformer[34]			Ours		
		SAcc	GAcc	CAcc	SAcc	GAcc	CAcc	SAcc	GAcc	CAcc	SAcc	GAcc	CAcc
SPG	Airplane	72.1	85.9	70.1	91.1	94.3	86.2	92.4	94.4	87.4	93.2	94.9	89.5
	Alarm clock	82.4	89.5	73.5	95.9	97.1	91.0	95.7	96.4	90.6	96.3	97.6	93.3
	Ambulance	69.3	84.6	61.6	90.6	94.8	81.9	91.4	95.1	82.3	92.2	95.6	86.0
	Ant	58.9	85.4	51.5	91.4	95.4	82.1	92.7	95.9	82.4	94.8	96.2	86.1
	Apple	75.9	83.9	69.5	91.1	94.1	89.5	92.5	94.6	90.1	94.2	94.9	92.7
	Backpack	64.8	84.6	59.5	84.1	93.5	79.3	85.3	93.7	80.4	87.8	94.5	83.9
	Basket	83.1	86.5	73.9	95.9	97.4	93.9	96.7	96.8	93.5	97.8	98.0	94.2
	Butterfly	82.4	87.5	79.5	97.9	98.1	96.1	97.7	97.9	95.9	98.8	98.6	96.5
	Cactus	76.7	89.9	75.4	95.2	97.3	93.3	96.1	97.4	93.4	96.9	97.7	93.8
	Calculator	88.7	90.5	77.4	98.4	98.3	97.2	98.5	98.3	97.1	99.2	98.7	97.5
	Campfire	90.2	93.2	79.5	95.3	96.6	92.5	96.7	96.8	92.7	97.1	97.1	92.8
	Candle	74.9	87.6	65.9	97.6	98.1	95.8	98.1	98.3	95.9	98.7	98.5	96.0
	Coffee cup	86.6	88.9	83.6	97.8	98.4	94.5	98.3	98.6	94.7	98.9	98.8	94.7
	Crab	71.3	84.9	69.3	94.0	95.8	87.9	93.9	96.4	89.9	95.6	96.4	91.4
	Duck	74.2	83.5	68.5	96.3	97.3	90.6	96.1	96.1	90.2	97.3	97.8	93.0
	Face	81.9	87.5	83.6	96.2	98.2	94.3	97.2	98.6	94.9	96.4	98.4	94.8
	Ice cream	80.5	86.9	88.4	94.5	95.6	91.0	94.4	95.6	90.9	95.9	96.5	92.4
	Pig	77.9	83.7	74.9	96.3	98.1	94.9	96.1	98.2	94.8	97.8	98.8	95.6
Pineapple	91.5	92.7	85.6	95.2	96.2	91.0	96.3	96.7	91.6	97.3	97.0	92.5	
Suitcase	91.9	93.1	87.6	96.4	97.8	95.1	97.3	98.3	95.6	97.6	98.6	95.9	
Average		78.8	87.5	73.9	94.6	96.6	90.9	95.2	96.7	91.2	96.2	97.2	92.6
CS	Birds	56.4	81.5	37.2	67.4	85.3	43.4	68.7	86.5	45.0	70.2	89.3	65.8
	Creatures	37.8	76.5	28.5	46.6	80.4	41.6	47.2	81.3	42.7	50.1	85.6	45.8
	Average	47.1	79.0	32.9	57.0	82.9	42.5	58.0	83.9	43.9	60.2	87.5	55.8

Table 2. Quantitative results of our ablation study on the SPG dataset.

Method	Airplane			Calculator			Face			Ice cream			Average		
	SAcc	GAcc	CAcc	SAcc	GAcc	CAcc	SAcc	GAcc	CAcc	SAcc	GAcc	CAcc	SAcc	GAcc	CAcc
Sketchformer-Seg	76.4	84.6	64.7	82.5	87.5	78.4	75.4	84.2	69.1	77.3	85.4	62.1	77.9	85.4	68.6
Ours w/o CC	85.6	87.5	82.4	91.1	92.1	89.2	91.7	90.3	87.3	87.9	90.1	86.4	89.1	90.0	86.3
Ours w/o DF	90.9	92.4	86.5	96.7	97.1	94.5	94.6	95.1	92.3	93.4	94.5	90.4	93.9	94.8	90.9
Ours-PD	91.1	92.5	86.8	97.2	97.8	95.1	95.3	96.2	93.1	94.1	95.3	91.2	94.4	95.5	91.6
Ours	93.2	94.9	89.5	99.2	98.7	97.5	96.4	98.4	94.8	95.9	96.5	92.4	96.2	97.1	93.6

Notably, it accurately captures elements such as clock numbers, as well as the wings and tail of a bird. More analysis and failure cases can be found in the supplementary.

4.2. Ablation Study

Stroke Embedding. We modify the stroke embedding module, leading to three alternative embedding networks:

- Sketchformer [23] - replace our whole embedding network with Sketchformer.
- w/o CC - remove the coordinate channels of our embedding network.
- w/o DF - remove the distance field regression decoder.

We train all three networks using our datasets from scratch until convergence. To understand the performance, we first inspect the embedding ability by comparing the reconstructed sketch. As shown in Fig. 5, given the input sketch, Sketchformer can reconstruct a clean but simplified sketch, i.e., the reconstruction either misses a few strokes (e.g., the fingers) or has strokes with inadequate variation

(e.g., the wing of the angel). Without the coordinate channels (w/o CC) or the distance regression branch (w/o DF), the reconstructed sketch exhibits a significant blur, hard to recognize. Instead, our reconstruction almost recovers the input with only a few artifacts when the strokes are cluttered (e.g., the texture of the pineapple or bag), which indicates the higher fidelity of our learned embedding.

To further examine the segmentation performance with specific embeddings, we link the three embedding variants with our segmentation Transformer (denoted as Sketchformer-Seg, Ours w/o CC, and Ours w/o DF, respectively), and train them on four typical categories (i.e., Airplane, Calculator, Face, and Ice cream) from the SPG dataset. Statistical results are shown in Tab. 2. Sketchformer-Seg obtained the worst segmentation results due to the under-representation of its embedding. Compared with Ours, the learned embeddings from w/o CC and w/o DF significantly suppress the segmentation ability.

Parallel Decoding. Other than the used auto-regressive de-

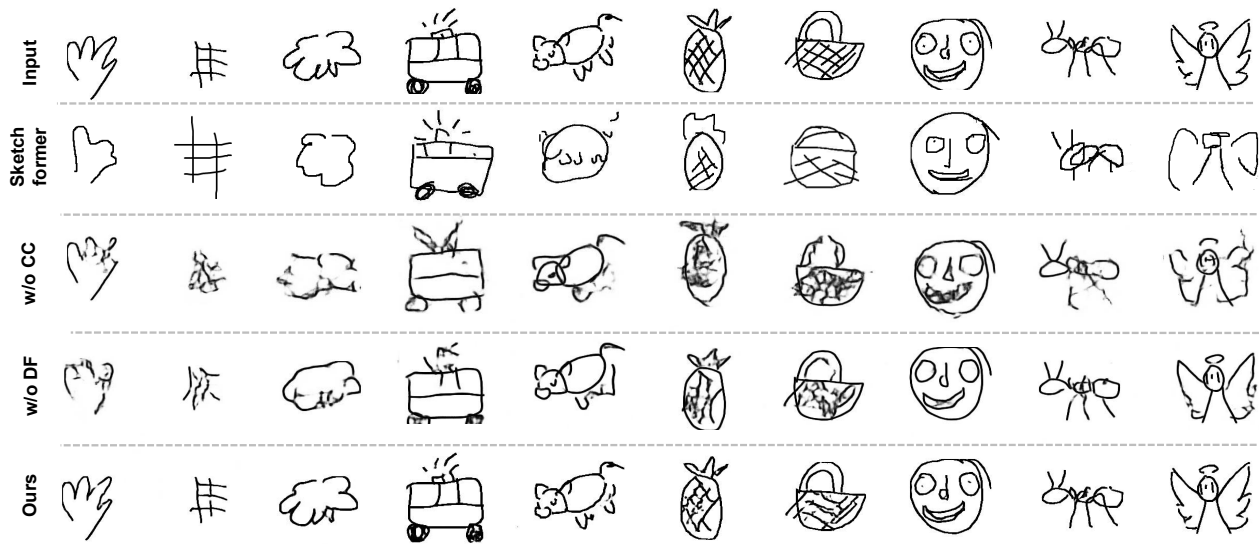


Figure 5. Sketch reconstruction results of our ablation study on different stroke embedding networks.

coding, the parallel decoding scheme is frequently used in Transformer training and testing. Following [4], we thus change our auto-regressive decoding to the parallel decoding scheme (denoted as Ours-PD) and report the evaluation metrics in Tab. 2. The learned decoder input embeddings cannot explicitly access the context information, thus leading to notable performance degradation (e.g., 93.6% vs. 91.6% of the average CAcc).

Additional Ablation Study. We have conducted more experiments on a) positional encoding of strokes and b) distance field-only decoding in the embedding learning, c) stroke-based auto-regressive decoding, and d) the group order in auto-regressive decoding, more results statistical and visual results can be found from the supplementary.

5. Discussion

5.1. Cross-Category Semantic Segmentation

For sketch semantic segmentation, cross-category training is challenging due to the diversity of training data categories. However, we hypothesize that if different categories have semantically, geometrically, and positionally similar parts, they should be mutually helpful to each other, and our segmentation Transformer excels at learning such contextual information thus benefiting from cross-category training. To verify this point, we conducted a preliminary experiment by manually selecting five categories - crab, ant, butterfly, airplane, and pig as the combined training data. After collecting all the parts, we merge the eye, mouth, nose, and ear into a single part - "face" because these parts all semantically and positionally compose the face part, and their stroke frequencies are low. Thus, we use nine parts, i.e., pliers, body, legs, head, tentacle, wings, tail, window, and face in the combined dataset.

After training the network, statistical evaluation is reported in Tab. 3, where clear improvements (green color) can be seen from all five categories, especially the ant face category with a remarkable 13.1% improvement, validating the superiority of our method in detecting and exploiting contextual information and the feasibility of cross-category training. Performance degradation is observed from the face and pliers parts of the crab, and the tail wing part of the airplane. Because these parts are only semantically but not positionally or geometrically similar to others, for example, the eyes of some crabs are out of the head, and the tail of an airplane has more complex patterns (e.g., triangles) than a simple curved tail. Qualitative results are presented in Fig. 6 with clear visual improvements consistent with statistics.

5.2. Sementic-aware Data Augmentation

In the SGP dataset, there is a significant imbalance in terms of the stroke distribution over different parts within certain categories. For example, in the "ant" category, the strokes for the "eye" part account for only 3.2% of all strokes, and the stroke accuracy for this part is also notably low at 69.3% even with the focal loss. We thus propose a novel semantic-aware data augmentation by copying and pasting. Specifically, as shown in Fig. 7, we select the "ant" and "ambulance" categories as the playground since they exhibit the aforementioned imbalance (see Tab. 4, the before column). We first divide the training examples into two sets with or without the "eye,mouth" part, denoted as A and B. We then randomly select a template example from A and copy the interested part to B based on the semantic constraints, e.g., the "eye,mouth" part must appear inside the head. Scaling is necessary to fit the added part to target examples in set B. Additionally, we introduce part-level rotation and offset perturbations to further improve the diversity. By copy-

Table 3. The statistical comparison before and after the cross-category training on **SAcc**.

Ant				Crab				Pig				Butterfly				Airplane			
Part	before	after	change	Part	before	after	change	Part	before	after	change	Part	before	after	change	Part	before	after	change
Face	69.3	82.4	+13.1	Face	88.4	88.1	-0.3	Face	96.0	96.4	+0.4	Body	98.4	98.6	+0.2	Body	95.2	95.9	+0.7
Head	91.7	92.8	+1.1	Body	95.1	96.2	+1.1	Head	97.6	98.3	+0.7	Tentacle	98.6	99.1	+0.5	Front wings	96.3	96.7	+0.4
Tentacle	88.7	91.8	+3.1	Legs	96.9	97.3	+0.4	Body	93.9	95.0	+1.1	Wings	99.3	99.4	+0.1	Windows	85.0	86.4	+1.4
Body	95.9	96.4	+0.5	Pliers	96.4	96.1	-0.3	Legs	98.5	98.7	+0.2					Tail wings	87.1	84.8	-2.3
Legs	96.2	96.5	+0.3					Tails	96.7	98.9	+2.2								
Average	88.4	92.0	+3.6	Average	94.2	94.4	+0.2	Average	96.5	97.5	+1.0	Average	98.8	99.0	+0.2	Average	90.9	91.0	+0.1

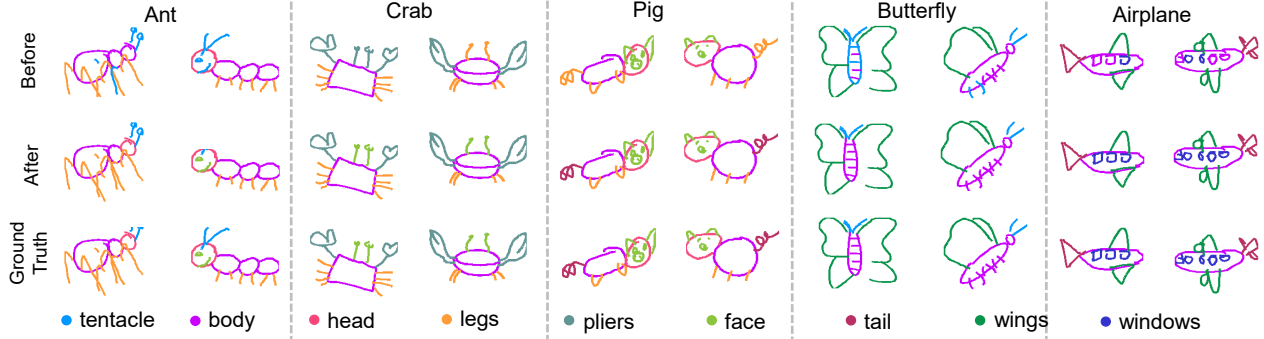


Figure 6. Visual segmentation results before and after cross-category training.

Table 4. The statistical evaluation of each part before and after using the semantic-aware data augmentation on **SAcc**.

Ant				Ambulance			
Part	before	after	change	Part	before	after	change
Eye,mouth	69.3	89.1	+19.8	Bell	97.5	97.4	-0.1
Head	91.7	91.9	+0.2	Body	97.5	97.9	+0.4
Tentacle	88.7	88.8	+0.1	Windows	58.6	76.2	+17.6
Body	95.9	95.8	-0.1	Cross	97.1	96.8	-0.3
Legs	96.2	96.9	+0.7	Wheels	94.8	96.1	+1.3
Average	88.4	92.5	+4.1	Average	89.1	92.9	+3.8

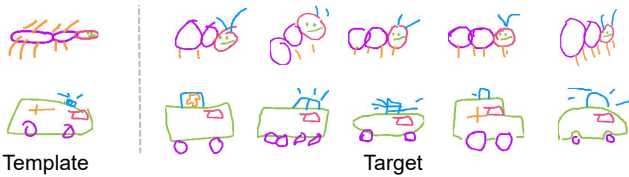


Figure 7. Given the eye of a template ant or a window of a template ambulance, we copy that part and paste it into the target sketches based on semantic constraints.

ing and pasting the rare part, we improve its occurrence to around 50% in the whole category. For the ambulance, a similar process is applied to the “window” part.

We use the augmented data to train our segmentation network. Statistical results are presented in Tab. 4, where remarkable improvements of the “eye,mouth” and “window” parts are observed, indicating the efficiency of the semantic-aware data augmentation. Visual results shown in Fig. 8 support the statistical observation.

5.3. Invariance Test

Following [31, 34], we also conducted invariance tests in terms of anti-rotation and anti-offset. As shown in Tab. 6

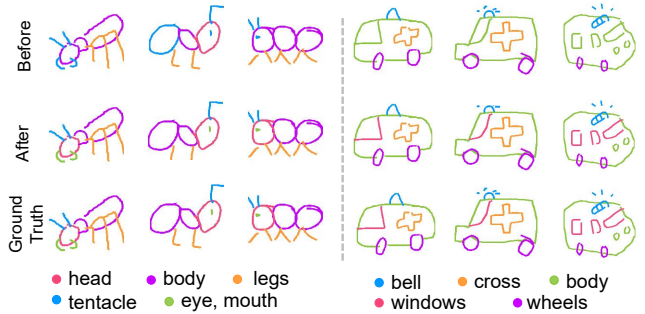


Figure 8. Visual results of the semantic segmentation with the novel data augmentation.

in the supplementary, our performance drops when increasing the rotation angle and the offset distance of the testing examples, but we still obtain superior accuracy than SketchGNN [31] and Sketch-Segformer [34], which indicates the excellent robustness of our approach. See the supplementary for detailed configuration and results.

6. Conclusion

In this paper, we proposed *ContextSeg*, for sketch semantic segmentation, achieving the best segmentation accuracy over SOTA methods. Comprehensive evaluations validate our superior performance, and preliminary experiments on cross-category training and semantic-aware data augmentation suggest inspiring research directions. However, our embedding network is not perfect and cannot encode strokes with rapid and large variations, while our segmentation Transformer fails to label strokes that are highly cluttered, e.g., the hairs and eyelashes.

References

- [1] Ankan Kumar Bhunia, Salman Khan, Hisham Cholakkal, Rao Muhammad Anwer, Fahad Shahbaz Khan, Jorma Laaksonen, and Michael Felsberg. Doodleformer: Creative sketch drawing with transformers. In *European Conference on Computer Vision*, pages 338–355. Springer, 2022. 1, 2
- [2] Tu Bui, Leo Ribeiro, Moacir Ponti, and John Collomosse. Compact descriptors for sketch-based image retrieval using a triplet loss convolutional neural network. *Computer Vision and Image Understanding*, 164:27–37, 2017. 1
- [3] Tu Bui, Leonardo Ribeiro, Moacir Ponti, and John Collomosse. Sketching out the details: Sketch-based image retrieval using convolutional neural networks with multi-stage regression. *Computers & Graphics*, 71:77–87, 2018. 1, 2
- [4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 2, 7
- [5] Shin-I Cheng, Yu-Jie Chen, Wei-Chen Chiu, Hung-Yu Tseng, and Hsin-Ying Lee. Adaptively-realistic image generation from stroke and sketch with diffusion model. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 4054–4062, 2023. 1
- [6] Adrien Delaye and Kibok Lee. A flexible framework for online document segmentation by pairwise stroke distance learning. *Pattern Recognition*, 48(4):1197–1210, 2015. 2
- [7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2
- [8] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6824–6835, 2021. 2
- [9] Songwei Ge, Vedanuj Goswami, C Lawrence Zitnick, and Devi Parikh. Creative sketch generation. *arXiv preprint arXiv:2011.10039*, 2020. 2, 5
- [10] David Ha and Douglas Eck. A neural representation of sketch drawings. *arXiv preprint arXiv:1704.03477*, 2017. 1
- [11] Zhe Huang, Hongbo Fu, and Rynson WH Lau. Data-driven segmentation and labeling of freehand sketches. *ACM Transactions on Graphics (TOG)*, 33(6):1–10, 2014. 5
- [12] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3
- [13] Changjian Li, Hao Pan, Yang Liu, Xin Tong, Alla Sheffer, and Wenping Wang. Robust flow-guided neural prediction for sketch-based freeform surface modeling. *ACM Transactions on Graphics (TOG)*, 37(6):1–12, 2018. 1
- [14] Changjian Li, Hao Pan, Adrien Bousseau, and Niloy J Mitra. Free2cad: Parsing freehand drawings into cad commands. *ACM Transactions on Graphics (TOG)*, 41(4):1–16, 2022. 1, 2, 5
- [15] Ke Li, Kaiyue Pang, Yi-Zhe Song, Tao Xiang, Timothy M Hospedales, and Honggang Zhang. Toward deep universal sketch perceptual grouper. *IEEE Transactions on Image Processing*, 28(7):3219–3231, 2019. 1, 2, 5, 6
- [16] Lei Li, Hongbo Fu, and Chiew-Lan Tai. Fast sketch segmentation and labeling with deep learning. *IEEE computer graphics and applications*, 39(2):38–51, 2018. 1, 2
- [17] Fang Liu, Xiaoming Deng, Yu-Kun Lai, Yong-Jin Liu, Cuixia Ma, and Hongan Wang. Sketchgan: Joint sketch completion and recognition with generative adversarial network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5830–5839, 2019. 2
- [18] Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. *Advances in neural information processing systems*, 31, 2018. 3
- [19] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 2
- [20] Tsvetomila Mihaylova and André FT Martins. Scheduled sampling for transformers. *arXiv preprint arXiv:1906.07651*, 2019. 3
- [21] Kaiyue Pang, Ke Li, Yongxin Yang, Honggang Zhang, Timothy M Hospedales, Tao Xiang, and Yi-Zhe Song. Generalising fine-grained sketch-based image retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 677–686, 2019. 2
- [22] Yonggang Qi and Zheng-Hua Tan. Sketchsegnet+: An end-to-end learning of rnn for multi-class sketch semantic segmentation. *Ieee Access*, 7:102717–102726, 2019. 1, 2
- [23] Leo Sampaio Ferraz Ribeiro, Tu Bui, John Collomosse, and Moacir Ponti. Sketchformer: Transformer-based representation for sketched structure. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14153–14162, 2020. 2, 6
- [24] Aneeshan Sain, Ayan Kumar Bhunia, Yongxin Yang, Tao Xiang, and Yi-Zhe Song. Stylemeup: Towards style-agnostic sketch-based image retrieval. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8504–8513, 2021. 2
- [25] Rosália G Schneider and Tinne Tuytelaars. Example-based sketch segmentation and labeling using crfs. *ACM Transactions on Graphics (TOG)*, 35(5):1–9, 2016. 2
- [26] Zhenbang Sun, Changhu Wang, Liqing Zhang, and Lei Zhang. Free hand-drawn sketch segmentation. In *Computer Vision—ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7–13, 2012, Proceedings, Part I 12*, pages 626–639. Springer, 2012. 2
- [27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2, 4

- [28] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. *Advances in neural information processing systems*, 28, 2015. [4](#)
- [29] Fei Wang, Shujin Lin, Hanhui Li, Hefeng Wu, Tie Cai, Xiaonan Luo, and Ruomei Wang. Multi-column point-cnn for sketch segmentation. *Neurocomputing*, 392:50–59, 2020. [1](#), [2](#)
- [30] Xingyuan Wu, Yonggang Qi, Jun Liu, and Jie Yang. Sketch-segnet: A rnn model for labeling sketch strokes. In *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2018. [1](#), [2](#), [5](#)
- [31] Lumin Yang, Jiajie Zhuang, Hongbo Fu, Xiangzhi Wei, Kun Zhou, and Youyi Zheng. Sketchgnn: Semantic sketch segmentation with graph neural networks. *ACM Transactions on Graphics (TOG)*, 40(3):1–13, 2021. [1](#), [2](#), [5](#), [6](#), [8](#), [3](#), [4](#)
- [32] Hua Zhang, Si Liu, Changqing Zhang, Wenqi Ren, Rui Wang, and Xiaochun Cao. Sketchnet: Sketch classification with web images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1105–1113, 2016. [2](#)
- [33] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018. [4](#)
- [34] Yixiao Zheng, Jiyang Xie, Aneeshan Sain, Yi-Zhe Song, and Zhanyu Ma. Sketch-segformer: Transformer-based segmentation for figurative and creative sketches. *IEEE Transactions on Image Processing*, 2023. [1](#), [2](#), [5](#), [6](#), [8](#), [3](#), [4](#)
- [35] Xianyi Zhu, Yi Xiao, and Yan Zheng. Part-level sketch segmentation and labeling using dual-cnn. In *Neural Information Processing: 25th International Conference, ICONIP 2018, Siem Reap, Cambodia, December 13-16, 2018, Proceedings, Part I 25*, pages 374–384. Springer, 2018. [1](#), [2](#)
- [36] Xianyi Zhu, Yi Xiao, and Yan Zheng. 2d freehand sketch labeling using cnn and crf. *Multimedia Tools and Applications*, 79(1-2):1585–1602, 2020. [1](#), [2](#)