Deep Learning for Free-Hand Sketch: A Survey and A Toolbox

Peng Xu, Timothy M. Hospedales, *Member, IEEE,* Qiyue Yin, Yi-Zhe Song, *Senior Member, IEEE,* Tao Xiang, and Liang Wang, *Fellow, IEEE*

arXiv:2001.02600v2 [cs.CV] 26 Apr 2020

Abstract—Free-hand sketches are highly illustrative, and have been widely used by humans to depict objects or stories from ancient times to the present. The recent prevalence of touchscreen devices has made sketch creation a much easier task than ever and consequently made sketch-oriented applications increasingly popular. The progress of deep learning has immensely benefited free-hand sketch research and applications. This paper presents a comprehensive survey of the deep learning techniques oriented at free-hand sketch data, and the applications that they enable. The main contents of this survey include: (i) A discussion of the intrinsic traits and unique challenges of free-hand sketch, to highlight the essential differences between sketch data and other data modalities, *e.g.*, natural photos. (ii) A review of the developments of free-hand sketch research in the deep learning era, by surveying existing datasets, research topics, and the state-of-the-art methods through a detailed taxonomy and experimental evaluation. (iii) Promotion of future work via a discussion of bottlenecks, open problems, and potential research directions for the community. Finally, to support future sketch research and applications, we contribute `TorchSketch` – the first sketch-oriented open-source deep learning library, which is built on PyTorch and available at https://github.com/PengBoXiangShang/torchsketch/.

Index Terms—Free-Hand Sketch, Deep Learning, Survey, Introductory, Taxonomy, TorchSketch, Software Library.

1 Introduction

Free-hand sketch is a universal communication and art modality that transcends barriers to link human societies. It has been used from ancient times to today, comes naturally to children before writing, and transcends language barriers. Different from other related forms of expression such as professional sketch, forensic sketch, cartoons, technical drawing, and oil paintings, it requires no training and no special equipment. As such free-hand sketch is not bound by age, race, language, geography, or national boundaries. It can be regarded as an expression of the brain's internal representation of the world, whether perceived or imagined. Smiling faces for example, are always recognized by humans (Figure 1).

Sketches can convey many words, or even concepts that are hard to convey at all in words. Figure 1 shows several examples covering ancient and contemporary; literal and emotional; iconic and descriptive; abstract and concrete; and different media of drawing.

Free-hand sketch can be illustrative, despite its highly concise and abstract nature, making it useful in various scenarios such as communication and design. Therefore, free-hand sketch has been widely studied in computer vision and pattern recognition [1], [2], [3], [4], [5], computer graphics [6], [7], and human computer interaction [8], [9], [10], [11] communities. In particular, early research can be traced back to the 1960s and 1970s [12], [13].

However, free-hand sketch is fundamentally different to natural photos[1]. Sketch images provide a special data modality/domain that has both domain-unique challenges (*e.g.*, highly sparse, abstract, artist-dependent), as well as advantages (*e.g.*, lack of background, use of iconic representation). It is also unique in that, because its source is a dynamic 'pen' movement, free-hand sketch can be stored and processed in multiple representations. These include static pixel space (when rendered as an image), dynamic stroke coordinate space (when considered as a time series), and geometrical graph space (when considered topologically) – as discussed in Section 2. Thus from a pattern recognition or machine intelligence perspective, these unique traits of free-hand sketch often lead to sketch-specific model designs in order to exploit sketch-specific data properties and overcome sketch-specific challenges when analyzing sketches for recognition, generation, and so on. This survey will review these considerations and designs in detail.

Sketch research and applications in both industry and academia have boomed in recent years due to the prevalence of touchscreen devices (*e.g.*, smartphone, tablet) that make acquiring sketch data much easier than ever; as well as the rapid development of deep learning techniques that are achieving state-of-the-art performance in diverse artificial intelligence tasks. This boom has occurred on a several fronts: (i) Some classic research topics (*e.g.*, sketch recognition, sketch-based image retrieval, sketch-based 3D shape retrieval) have been re-studied in a deep learning context [2], [3], [4], [7], [14], [15] resulting in significant performance improvements. (ii) Some brand-new topics have been proposed based on deep learning, *e.g.*, deep learning based sketch generation/synthesis [5], sketch-based model generation [16], reinforcement learning based sketch abstraction [17], adversarial sketch based image editing [18], graph neural network based sketch recognition [19], graph convolution-based sketch semantic segmentation [20], and sketch based software prototyping [9]. (iii) Beyond global representation based tasks (*e.g.*, sketch recognition), more

1. Images can include both free-hand sketch and natural photos, *etc*. In this survey, "photo" denotes natural photo images obtained by a camera such as in ImageNet unless otherwise specified.

Fig. 1. Diverse free-hand sketches in human daily life.

fine-grained tasks have been further studied or proposed, *e.g.*, instance-level sketch-based image retrieval [4], and deep stroke-level sketch segmentation [21]. (iv) Compared with the conventional approach of representing sketches as static images [1], the trends of touchscreen acquisition and deep learning have underpinned progress on designing deep network architectures to exploit richer representations of sketch. Thanks to works such as SketchRNN [5], the sequential nature of free-hand sketches is now widely modeled by recurrent neural network (RNN). (v) More sketch-based applications have appeared, *i.e.*, the online sketch game QuickDraw[2] [5], sketch-based commodity search engine[3] [4], [22]. (vi) Some large-scale sketch datasets have been collected, *e.g.*, Sketchy [7] and Google QuickDraw[4] [5] – a million-scale sketch dataset (50M+).

1.1 Overview

This survey aims to review the state of the free-hand sketch community in deep learning era, hoping to bring insights to researchers and assist practitioners aiming to build sketch-based applications.

Previous Surveys and Scope This survey focuses on free-hand sketches, not including professional (forensic) facial sketch [23], [24], [25], [26], [27], professional pencil sketch [28], [29], professional landscape sketch [30], photo-like edge-maps (artificially rendered 'sketch') [31], [32], [33], cartoon/manga [34], [35], [36], well-drawn 3D sketch [37], or clip arts [35]. In this survey, "sketch" refers to "free-hand sketch", unless otherwise specified.

To our knowledge, only a few survey papers [38], [39], [40], [41], [42] were published in the free-hand sketch community in the past decade. However, these survey papers [38], [39], [40], [41], [42]: (i) only focus on two research topics, *i.e.*, free-hand sketch based recognition and image/3D retrieval. (ii) mainly review classic non-deep techniques. In contrast, the current boom in advanced deep methodologies, techniques (hashing), representations (sequential, topological), and novel applications (generation, segmentation) makes it timely to provide an up-to-date survey of the big picture of research on free-hand sketch.

Contributions We provide a comprehensive survey reviewing the state of the field with regards to deep learning techniques, as well as applications of free-hand sketch. In particular: (a) We discuss the intrinsic traits, and unique challenges and opportunities posed when working with free-hand sketch data. (b) We provide a detailed taxonomy of both datasets, and applications covering both uni-modal (sketch alone) and multi-modal (relating sketches to photos, text, *etc*) cases. For each specific task, the contemporary landscape of deep learning solutions are summarized, and milestone works are described in detail. (c) We discuss current bottlenecks, open problems, and potential research directions for free-hand sketch.

Furthermore, we contribute the first fully-featured sketch-oriented open source library, `TorchSketch`. A thriving open-source community and codebase is key to the development of any research field. However in free-hand sketch, much research is hard to reproduce; further, existing open source codes are scarce, and programmed on different deep learning frameworks. We hope that our contribution of `TorchSketch` can facilitate the sketch-oriented deep learning research and applications. Our PyTorch-based `TorchSketch`, provides comprehensive and user-friendly APIs to state-of-the-art sketch-specific deep learning techniques, including four kinds of network architectures (*i.e.*, CNN, GNN, RNN, TCN), processing procedures, data analysis/visualization/augmentation scripts, utility functions, *etc*. Furthermore, `TorchSketch` supports GPU based and Python built-in multi-processing acceleration. To demonstrate `TorchSketch` and facilitate future research, we provide an architecture-level quantitative comparison over the aforementioned four kinds of network architectures that have been applied to free-hand sketch to date. We hope that this will provide a common starting point for future

2. https://quickdraw.withgoogle.com
3. http://sketchx.eecs.qmul.ac.uk/demos/
4. https://github.com/googlecreativelab/quickdraw-dataset

TABLE 1
Notation definitions and abbreviated terms in this survey.

Notations	Descriptions
$\mathbf{M}$, $\mathbf{M}^T$	matrix $\mathbf{M}$ and its transpose
$\mathcal{X} = \{\mathbf{X}_n\}_{n=1}^N$	sketch sample set
$\mathbf{X}_m$, $\mathbf{X}_n$	m-th and n-th sketch samples in the sketch sample set $\mathcal{X}$
$\mathcal{Y} = \{y_n\}_{n=1}^N$	associated label set of $\mathcal{X}$
y_n	label of $\mathbf{X}_n$
$\mathcal{L}$	loss function
$\mathbf{\Theta}$	learnable parameters of neural network
$\mathcal{F}(\cdot)$	function mapping or feature extraction
$\mathcal{F}_\Theta(\cdot)$	neural network feature extraction, parameterized by Θ
$\mathcal{D}(\cdot,\cdot)$	distance metric, *e.g.*, ℓ_2 distance
λ	weighting factor
$\sum$	summation
α, β, γ	hyper parameters by manually setting
Abbreviated Terms	**Descriptions**
CNN	convolutional neural network
GNN	graph neural network
RNN	recurrent neural network
TCN	temporal convolutional neural network

research, improve reproducibility and comparability, and accelerate progress in research and applications.

Organization of This Survey The rest of this survey is organized as follows. Section 2 provides background knowledge of free-hand sketch, including intrinsic traits, domain-unique challenges, milestone techniques of the existing sketch-oriented deep learning works, *etc*. Section 3 summarizes the representative free-hand sketch datasets. In Section 4, we provide a comprehensive taxonomy for various sketch-based tasks, and describe the representative deep learning techniques in detail. Section 5 presents an overview and the main features of `TorchSketch` library, as well as some methodological comparisons based on `TorchSketch` implementation. Section 6 discusses open problems, bottlenecks, and potential research directions before the survey concludes in Section 7.

Throughout this survey, bold uppercase and bold lowercase characters denote matrices and vectors, respectively. Unless specified otherwise, mathematical symbols and abbreviated terms follow the conventions in Table 1.

2 Background

This section presents background knowledge, including: the intrinsic traits, and domain-unique challenges and opportunities of free-hand sketch – in particular the essential differences to natural photos; and a brief development history of deep learning for free-hand sketch, summarizing the milestone techniques.

2.1 Intrinsic Traits and Domain-Unique Challenges

Representation Free-hand sketch is a special kind of visual data, intrinsically different to natural photos that are the pixel-perfect copies of the real world. For efficient storage and fast calculation, free-hand sketch can be saved as a sparse matrix, or as a black and white image that ignores its sparsity Figure 2 (left images). Since sketch generation is a dynamic process, suitably captured sketches can also be represented as a sequence of strokes or pen coordinates (Figure 2, right images), which are represented in Euclidean space. In this regard, sketches share similarities with handwritten characters, yet are fundamentally different for their highly abstract and free-style nature (hand-writings are subject to to specific rules and a learning process). From another perspective, free-hand sketches can also be modeled as a sparsely connected graph where lines are edges in the graph. Compared with a sequence of euclidean coordinates, topological representation as a graph can provide a more flexible and abstract representation. As a result of this diversity of possible representations, various deep learning paradigms can be used to process sketches including CNNs, RNNs, GCNs, and TCNs.

Unique Challenges and Opportunities The unique challenges of free-hand sketch can be summarized as follows: (i) Highly Abstract: Humans use sketch to depict an object or event in very few strokes, reflecting the high-level semantics of a mental image. As shown in Figure 3, a pyramid can be depicted as a triangle in sketch, and several strokes depict a fancy handbag. (ii) Highly Diverse: Different persons have different drawing styles. For example, a near 'realistic' (close to photo edge-map) sketch image could be portrayed in different ways as exaggerated (c.f. caricatures), iconic (where details are omitted and the sketch is near symbolic), or artistic. Depending on subjective opinion about salience, different parts may also be included or omitted in a sketch. For instance, given a "cat", people differ on choice of drawing with/without body (see Figure 3). Finally, there is the mental viewpoint of different users, *e.g.*, whether they imagine an orthographic or perspective projection image (Figure 4). In Figure 3, we can see that different persons draw differing perspective views of an identical slipper. (iii) Highly Sparse: No matter the representation, free-hand sketch is a highly sparse signal compared to photographs. (iv) Finally, there are some unique challenges when collecting sketch, which will be discussed in detail in following (see Section 3.2).

Sketch also provides some unique opportunities compared to photos: (i) As a counterpoint to the sparsity challenge, sketch often lacks distracting background clutter compared to photos, which can benefit automated analysis [3]. (ii) If captured appropriately, the sequential nature of sketch generation can further be exploited to benefit analysis compared to static images [44]. (iii) The sparse and sequential nature of sketch also provides opportunities for high quality sketch generation, where image generation is hampered by the need to fill in pixel detail [5], [45]. (iv) Sketches can serve as a computer-interaction modality in a way that photos cannot [4], [44], due to the intuitive way humans can generate them without training.

Given these unique challenges and opportunities, it is often beneficial to design sketch-specific models to obtain best performance in various sketch-related applications.

2.2 A Brief History of Sketch in the Deep Learning Era

In the past five years, the free-hand sketch community has developed rapidly as summarized by Figure 5 from the perspectives of: tasks, datasets, representations and supervision. (i) In 2015, Sketch-a-Net [14] was proposed as the first CNN engineered specifically for free-hand sketch. It gained note as the first to achieve a recognition rate surpassing humans and helped to popularize deep learning for sketch analysis. (ii) In 2016, three fine-grained sketch-based image retrieval (FG-SBIR) datasets were released, *i.e.*,

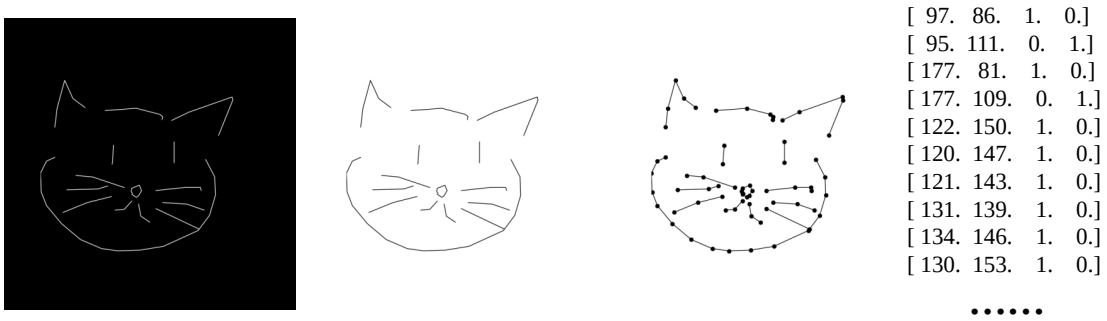


Fig. 2. Sketch-specific representations. Representations from left to right: sparse matrix (black background with white lines), dense picture (white background with black lines), graph, stroke sequence. Both graph and stroke sequence representations are based on the key stroke points. In stroke sequence, each key point is denoted as a four-tuple, where the first two entries and the last two entries represent the coordinates and pen state, respectively. See details in text.

Fig. 3. Illustrations of the major domain-unique challenges of free-hand sketch. Each column is a photo-sketch pair. Sketch is highly abstract. A pyramid can be depicted as a triangle in sketch, and a few strokes depict a fancy handbag. Sketch is highly diverse. Different persons draw distinctive sketches when given the identical reference, due to subjective salience (head vs. body), and drawing style.

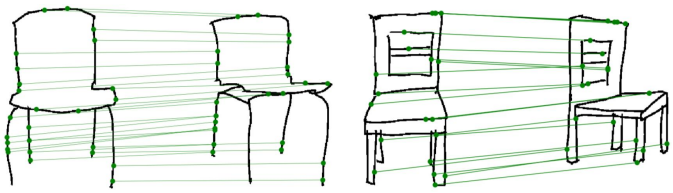


Fig. 4. Multi-view sketch pairs selected from [43]. Different users can imagine and draw the same object from different perspectives.

QMUL Shoe [4] and Chair [4], and Sketchy [7]. Combined with deep triplet ranking [46], these fine-grained cross-modal datasets motivated a wave of followup FG-SBIR and other fine-grained tasks. (iii) In 2017, Google released a million-scale sketch dataset, *i.e.*, Google QuickDraw, via the online game "QuickDraw". QuickDraw contains over 50M sketches collected from players around the global world, making it a rich and diverse dataset. Furthermore, based on the QuickDraw dataset, Ha *et al*. proposed "SketchRNN", a RNN-based deep Variational AutoEncoder (VAE) that can generate diverse sketches. This work motivated the community to go beyond considering sketches as static pictures to be processed by CNN; and inspired subsequent work to use stroke sequences as input and study temporal processing of sketches. In 2017, some sketch-based deep generative image models [47] began to appear in the top conferences in computer vision. (iv) From 2018 to date, based on deep learning techniques, various novel methodologies – *e.g.*, sketch hashing [15], sketch transformers [19]; and applications – *e.g.*, sketch abstraction [17], sketch-based photo classifier generation [16], sketch perceptual grouping [48] have been proposed. See Figure 5 for a chronological summary.

3 Free-Hand Sketch Datasets

In the last decade numerous new free-hand sketch datasets have been collected, to satisfy the need for large-scale deep network training, and the growing diversity of sketch-related tasks considered by the community. This section will summarize these datasets, and further discuss some of the unique challenges in sketch-related data collection.

3.1 Sketch Datasets

Free-hand sketch datasets can be grouped in terms of: (i) single vs. multi-modal, and (ii) coarse vs. fine-grained. Single-modal datasets consist only of sketches and are typically used for recognition, sketch-sketch retrieval, grouping, segmentation and generation. Multi-modal datasets support cross-modal tasks by providing sketches paired with samples of from other modalities such as natural photo, 3D shape, text, or video. These are mainly used for cross-modal retrieval/matching, or cross-modal generation/synthesis. Coarse grained datasets (*e.g.*, TU-Berlin [6], QuickDraw [5]) are usually used for sketch recognition, sketch retrieval; while fine-grained datasets (*e.g.*, QMUL Shoe [4]) provide fine-grained visual details and manual annotations.

More specifically, coarse-grained single-modal datasets [5], [58] support sketch recognition and retrieval; while

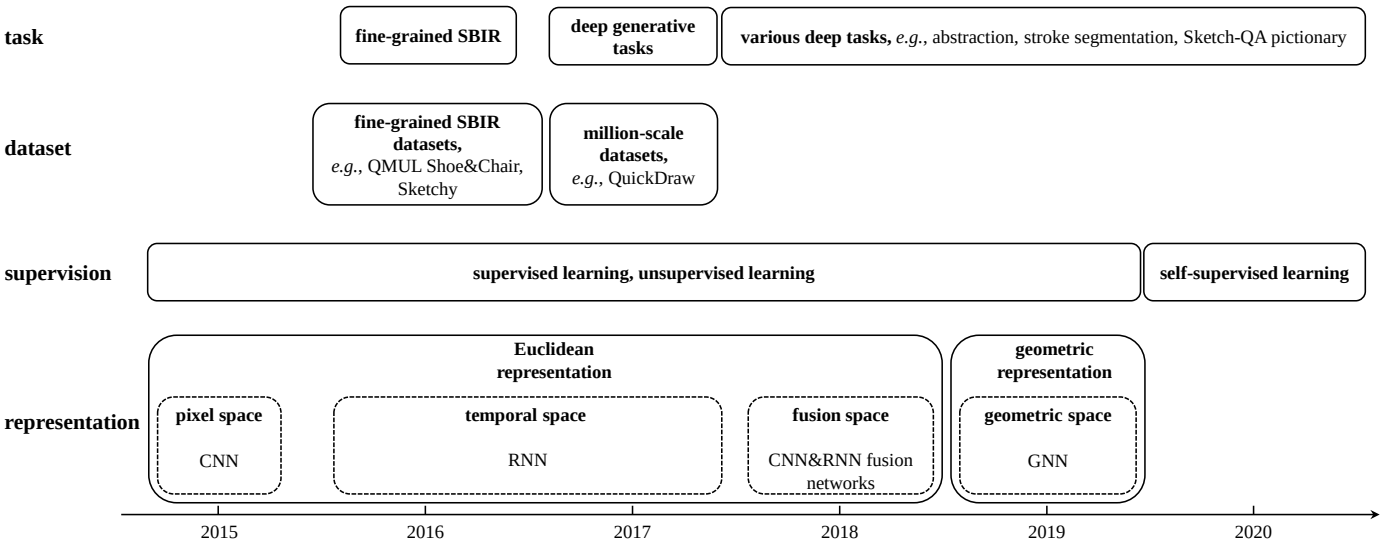


Fig. 5. Milestones of deep learning based free-hand sketch research, from the perspectives of task, dataset, supervision, and representation.

TABLE 2
Summary of the representative sketch datasets. "✓" denotes "yes/available/provided". Both 'grouping" and "segmentation" annotations refer to stroke-level. "K" and "M" mean "thousand" and "million", respectively. "Cat." means "category". Stroke "✓" denotes sketches provided as SVG files or coordinate arrays.

Single-Modal Datasets	Fine-Grained	Public	Modalities & Sample Amount	Cat.	Stroke	Object/ Scene	Instance Pairing	Annotations	Remarks
TU-Berlin [6]		✓	20K sketches	250	✓	o	-	class	
QuickDraw [5]		✓	50M+ sketches	345	✓	o	-	class	
QuickDraw-5-step [49]			38M+ sketches	345		o	-	class	
SPG [48]	✓	✓	20K sketches	25	✓	o	-	class, grouping	
SketchSeg-150K [21]	✓		150K sketches	20	✓	o	-	class, segmentation	57 semantic labels
SketchSeg-10K [50]	✓	✓	10K sketches	10		o	-	class, segmentation	24 semantic labels
SketchFix-160 [51]		✓	3904 sketches	160	✓	o	-	class, eye fixation	
Multi-Modal Datasets	**Fine-Grained**	**Public**	**Modalities & Sample Amount**	**Cat.**	**Stroke**	**Object/ Scene**	**Instance Pairing**	**Annotations**	**Remarks**
QMUL Shoe [4]	✓	✓	419 sketches, 419 photos	1		o	✓	pairing,triplet,attribute	21 binary attributes
QMUL Chair [4]	✓	✓	297 sketches, 297 photos	1		o	✓	pairing,triplet,attribute	15 binary attributes
QMUL Handbag [22]	✓	✓	568 sketches, 568 photos	1		o	✓	pairing	
Sketchy [7]	✓	✓	75K sketches, 12K photos	125	✓	o		class	12K objects
Sketch&UI [8]	✓		1998 sketches, 1998 photos	23		o	✓	class, pairing	UI
QuickDrawExtended [52]		✓	330K sketches, 204K photos	110		o		class	
SketchTransfer [53]			112.5K sketches, 90K CIFAR-10 photos	9		o		class	resolution of 32x32
TU-Berlin Extended [54]			20K sketches, 191K photos	250		o		class	
Sketch Flickr15K [1]		✓	330 sketches, 15K photos	33		o		class	
Aerial-SI [55], [56]			400 sketches, 3.3K photos	10		o, s		class	aerial scene
HUST-SI [57]		✓	20K sketches, 31K photos	250	✓	o		class	
SBSR [58]		✓	1814 sketches, 1814 3D models	161		o		class	
SHREC'13 [38]	✓	✓	7200 sketches, 1258 3D models	90		o		class	
SHREC'14 [59]	✓	✓	12680 sketches, 8987 3D models	171		o		class	
PACS DG [60]		✓	9991 (sketches, photos, cartoons, paintings)	7		o		class	domain generalization
Flickr1M [61]			500 sketches, 1.3M photos	100		o		class	
Cross-Modal Places [62]		✓	16K sketches, 11K descriptions, 458K spatial texts, 12K clip arts, 1.5M photos	205		s		class	
DomainNet [63]		✓	0.6M (cliparts, infographs, paintings, QuickDraw sketches, real photos, professional pencil sketches)	345	✓	o		class	

coarse-grained multi-modal datasets (*e.g.*, QuickDraw-Extended [52]) support category-level sketch-based image retrieval. Fine-grained single-modal datasets [21], [48] support perceptual grouping, segmentation, and parsing. Fined-grained multi-modal datasets (*e.g.*, QMUL Shoe [4]) provide the instance-level pairing information to support retrieval.

Table 2 summarizes representative sketch datasets of each type in terms of: modalities, size, number of categories, stroke information, annotation, *etc*. Note that SVG files are able to generate picture files such as JPEG and PNG, but not vice versa. We exclude some well-known but overly-small datasets such as [64].

3.2 Unique Challenges of Sketch Collection

Sketch Collection Strategies Existing collection approaches mainly include: (i) bespoke creation by researchers [21], [48], [50], (ii) crowd-sourcing selecting and matching on existing datasets, *e.g.*, Doodle2Sketch QuickDraw-Extended [52], (iii) crowd-sourcing drawing from scratch, *e.g.*, QMUL Shoe [4], Sketchy [7]. (iv) collecting via online drawing games, *e.g.*, Google QuickDraw [5]. (v) Web crawling of existing sketches [60], [63]. In particular, for fine-grained multi-modal sketch datasets, crowd-sourcing is widespread, since fine-grained drawing, selection, and matching are time-consuming.

Sketch Collection Challenges Free-hand sketch poses some some unique data collection challenges compared to other image types: (i) **Time-Sequence Nature** Sketching is a dynamic and temporally extended process. Thus collecting sketches as static raster images (*e.g.*, JPEG, PNG) is very limiting, and recording vector representation (*e.g.*, SVG[5]) together with stroke position and timing is preferred to support research. As a consequence, this means that collecting by web-crawling (which typically retrieves raster images), is less useful. (ii) **Cross-Modal Pairing** Collecting cross-modal datasets provides the additional challenge of pairing sketches and associated data in other modalities. One can start with existing images and sketches and pair them [65], or draw sketches specifically corresponding to given examples in other modalities [4], [7]. But both options are time-consuming. (iii) **Demographic Information** Since sketches are human-created, rather than pixel-perfect images captured by camera, it is important to ensure that sketch datasets are created by diverse demographics to ensure they are representative (an even stronger version of the challenge [66] posed by photo images). Furthermore, meta-data about the artists (*e.g.*, gender, nationality, skill level) may be important to store, both in order to study how different humans sketch, and also to ensure that any sketch-based applications are fair and unbiased [67]. However to our knowledge, this kind of meta-data has not been recorded in existing datasets.

Discussion These challenges all mean that the standard computer vision approach of web-crawling is poorly suited to collection of sketch data: It does not typically retrieve vector-graphic and time-stamped sketch generation, does not make it easy to obtain matched cross-modal pairs, and does not come with any demographic information. For these reasons, bespoke creation, crowd-sourcing, and generation as a byproduct through gamification are the recommended methods of collection.

4 Tasks and Methodology Taxonomy

According to the data modalities involved, free-hand sketch related tasks can be divided into single- and multi-modal tasks, where single-modality sketch analysis techniques are often used as building blocks for multi-modal methods. This section will define the popular sketch analysis tasks and introduce the corresponding deep learning methods, providing a detailed taxonomy. Figure 6 provides a tree diagram of the existing free-hand sketch tasks.

5. https://en.wikipedia.org/wiki/Scalable_Vector_Graphics

4.1 Single-Modal

These tasks study sketches in isolation without other data modalities. Key deep learning-based applications in this area include recognition, retrieval/hashing, generation, grouping, segmentation, and abstraction.

4.1.1 Recognition

Sketch recognition [6] aims to predict the class label of a given sketch, which is one of the most fundamental tasks in computer vision. It has a variety of practical applications including: interactive drawing systems that provide feedback to users [68], sketch-based science education [69], and games [5], [44]. Both object [5], [6] and scene [70], [71] categories have been studied from a recognition perspective. Notably sketch recognition techniques underpin the popular web game QuickDraw and WeChat mini-app Caihua Xiaoge, both released by Google.

Sketch recognition can be categorized into (i) offline and (ii) online recognition settings. Offline recognition systems take the whole sketch as input and predict a class label based on the complete sketch. Online recognition systems take the accumulated sketch strokes and continuously predict the class label, during sketching. Offline recognition methods are more common, but online methods can be used in more interactive real-time applications such as real-time drawing guidance [49], tracing, and interactive sketch retrieval.

There are several current trends in sketch recognition, building on underpinning deep learning progress: (i) From raster image to sequence representation; (ii) From global representation to local analysis; (iii) From Euclidean (CNN, RNN based) to topological analysis (GNN based), (iv) From fully-supervised learning to self-supervised learning.

Numerous deep models have now been proposed for free-hand sketch recognition [54], [72], [73], [74], [75], [76], [77], [78], [79], [80]. In the following, we review these models from three perspectives, *i.e.*, networks, loss functions, data augmentations (will be discussed separately in Section 4.1.2).

Networks Figure 5 (below) summarizes the evolution of the deep learning-based sketch representations. Moreover, Table 3 lists various networks that are engineered for free-hand sketches and capable of sketch recognition. We next introduce some representative networks.

(i) Sketch-a-Net [3], [14] was the first deep CNN designed for free-hand sketch. Compared with classic photo-oriented CNN architecture [86], the sketch-specific aspects of its architecture mainly include: (a) Considered the sparse low-texture nature of sketch, larger size (15×15) first layer filters are used to capture more context. (b) Local response normalization (LRN) [86] layers are removed for faster learning without sacrificing performance, since LRN is for "pixel brightness normalization", but most sketches are binary images. (c) Two novel sketch-specific data augmentation strategies are proposed, leveraging stroke appearance and stroke sequence. (d) Finally, an ensemble of Sketch-A-Net were combined using joint Bayesian fusion [87].

(ii) Sarvadevabhatla *et al*. [81] proposed a sketch recognition network to leverage the sequential process of sketching, where each training sketch is plotted as a continuous

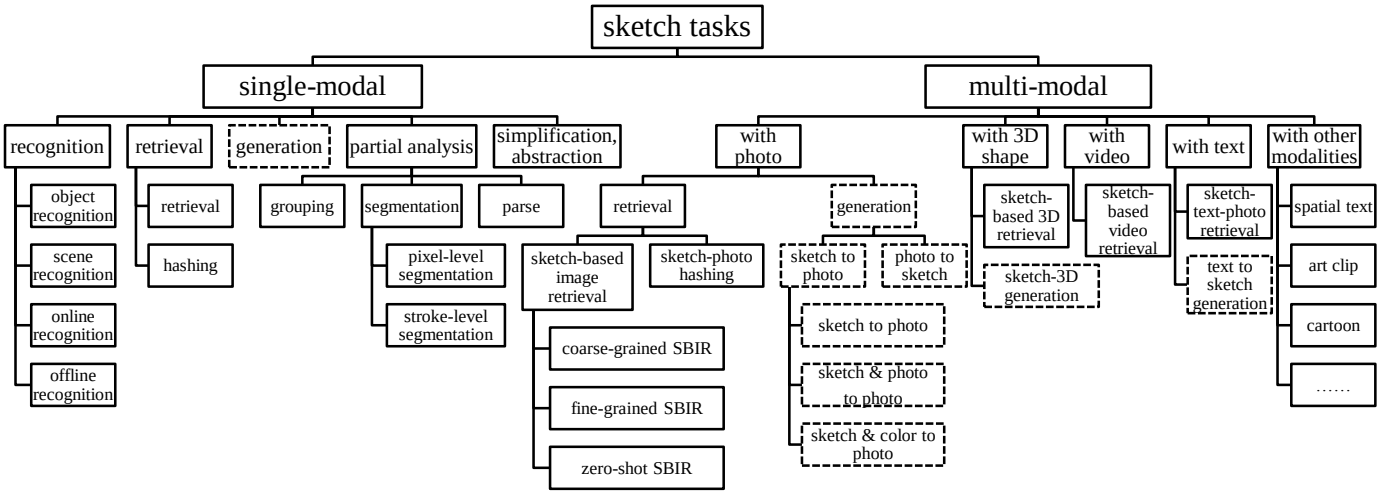


Fig. 6. A tree diagram of the sketch task taxonomy. Generative tasks are framed by dashed lines.

TABLE 3
Comparison of representative sketch recognition networks. "–" means that it is vague or not mentioned in the original paper. The reported performances are the recognition accuracy of "acc.@top1". Abbreviations in this table: "Ens.": ensemble; "stroke accu. pic.": stroke accumulated pictures; "R-FC": residual fully-connected layer; "pad.": padding; "tru.": truncation; "augm.": specific augmentations; "tran.": transformer.

Year	Model	Architecture	Layers	Params	Ens.	Pretrain	Input	Preprocess	Dataset	Accuracy
2015	Sketch-a-Net [14]	CNN	5 conv.	8.5M	✓		picture	augm. [14]	TU-Berlin [6] 250 cat.	0.7490
2016	AlexNet-FC-GRU [81]	CNN-to-RNN cascaded	–	–			stroke accu. pic.	–	TU-Berlin 160 cat.	0.8510
2018	SketchMate [15]	RNN	2 GRU	–			stroke vector	tru. & pad. [15]	QuickDraw 3.8M [15]	0.7788
2018	SketchMate [15]	CNN-RNN dual-branch	5 conv. & 2 GRU	–			picture & stroke vector	tru. & pad. [15]	QuickDraw 3.8M [15]	0.7949
2017	Jia *et al.* [82]	RNN-RNN dual branch	–	–	✓	CNN needs pretraining	CNN features of stroke accu. pic.	reflection, rotation, *etc.*	TU-Berlin	0.9220
2017	DVSF [83]	R-FC and RNN dual branch	–	–		CNN needs pretraining	CNN features of stroke accu. pic.	–	TU-Berlin	0.7960
2018	FBin DAB-Net [84]	binary CNN	–	–			picture	–	TU-Berlin	0.7370
2018	RNN→CNN [85]	RNN-to-CNN cascaded	2 LSTM & 5 conv.	–		CNN needs pretraining	stroke vector	augm. [3]	TU-Berlin	0.7849
2019	multi-graph tran. [19]	GNN	4 tran.	10M			stroke vector	–	QuickDraw subset [19]	0.7070

sequence of cumulative stroke pictures and the corresponding AlexNet [86] based deep features will be sent into a Gated Recurrent Unit (GRU) [88] network in sequence. This network is also able to work in online recognition mode, since it involves the intermediate status of the sketch.

(iii) Similarly, He *et al*. [83] proposed the deep visual-sequential fusion (DVSF) net to capture spatial and temporal patterns of sketches simultaneously. For each training sketch, its three accumulation sub-pictures (with 60%, 80%, 100% of strokes) go through three-way CNNs (ResNet-18 [89]) to produce deep features, which are fed into both visual and sequential networks. In particular, the visual and sequential networks are implemented by residual fully-connected (R-FC) and Residual Long Short Term Memory (R-LSTM) [90] layers respectively. The visual and sequential paths are integrated by a fusion layer for final recognition.

(iv) In 2017, Ha and Eck proposed the groundbreaking SketchRNN [5], which performs representation learning through its Variational Inference (VI) [91] based sequential sketch generation model. Distinctively different to the prior stroke accumulated sub-picture representations, the key points of sketch strokes are directly fed into the RNN backbone of SketchRNN. In particular, as illustrated in Figure 2, each key point is denoted as a vector consisting of two coordinate bits (*i.e.*, horizontal and vertical coordinates) and the corresponding flag bits. The flag bits indicate the start/end of a stroke by pen state. The encoder backbone of SketchRNN also performs well for sketch recognition [6].

(v) Xu *et al*. proposed the sketch hashing network SketchMate [15], where the backbone is a CNN-RNN dual branch network, using CNN to extract abstract visual concepts and RNN to model human temporal stroke order. The CNN branch takes in the raster pixel sketch pictures; and the RNN branch process the vector sketch (*i.e.*, key point coordinates). The branches are combined by a late-fusion layer. This network demonstrates the complementarity of visual and temporal embedding spaces of sketch representation. This CNN-RNN dual-branch modeling idea has been widely applied to other sketch tasks, *e.g.*, SPFusionNet [50] for sketch semantic segmentation. Addition to the parallel frameworks of CNN and RNN, some cascaded frameworks (*e.g.*, RNN-to-CNN [85]) also have been studied.

6. https://github.com/payalbajaj/sketch_rnn_classification

(vi) A sketch can be represented as the sparsely connected graphs in topological space. Multi-Graph Transformer (MGT) [19] is a GNN model that learns both geometric structure and temporal information from sketch graphs. MGT injects domain knowledge into Graph Transformers through sketch-specific graphs. In particular, MGT represents each sketch as multiple intra-stroke and extra-stroke graphs, to model its local and global topological stroke structure, respectively.

(vii) While prior approaches to sketch recognition are based on supervised learning (*e.g.*, [3], [81], [83]), [92] provided the first investigation of self-supervised representation learning for sketch, proposing a new temporal convolutional neural network (TCN) designed to support this.

Loss Functions Most of the previous deep sketch recognition methods use cross-entropy softmax loss to train deep neural networks. An active research question is whether sketch-specific loss functions can help to further improve recognition performance. To that end, Xu *et al*. proposed the sketch-specific center loss [15] for million-scale sketches, based on a staged-training strategy. The basis is that the image entropy distribution of each sketch category is a truncated Gaussian distribution. Inspired by classical Bayesian decision theory [93], Mishra *et al*. proposed a novel metric loss to drive the pretained deep neural network to minimize the Bayesian risk of mis-classifying sketch pairs that were randomly selected within each mini-batch [94]. Based on this Bayesian risk loss, sketch recognition needs two-stage training. After obtained the features, a linear SVM [95] is trained as the classifier.

Summary and Discussion In this subsection, we reviewed sketch recognition related deep learning works, from the perspective of architecture and loss functions.

Promising areas of future work include: Online sketch recognition, motivated by practical human-computer interaction applications; going beyond the mainstream line of supervised sketch recognition to investigate semi-supervised, self-supervised and unsupervised learning for recognition; zero-shot [71] and few-shot sketch recognition; and multi-task learning [78], [96] to simultaneously solve sketch recognition with other tasks.

4.1.2 Data Augmentations

Note that all the sketch-specific data augmentation methods discussed in this subsection can be applied to both sketch recognition and all the other sketch involved tasks, *e.g.*, sketch-based image retrieval, sketch-related generation.

(i) When represented as a raster picture, most common data augmentations designed for natural photos [97] can be applied to sketch, *e.g.*, horizontal reflection/mirroring, rotation, horizontal shift, vertical shift, central zoom. These augmentations have already been evaluated by the early sketch-oriented deep learning works [14], [98]. However, random cropping is likely unsuitable for sketch since partial sketches are often too sparse to recognize even for humans.

(ii) Stroke thickening/dilation can be used for free-hand sketch. As discussed in some previous works on spatially-sparse convolutional neural networks [99], the subtle details of sparse strokes of sketches can be lost after multiple layers of convolution. Thus this can be useful for deep neural networks that process sketches as image input.

(iii) Yu *et al*. [3] proposed to remove the strokes to obtain more diverse sketches. Based on the observation [6] that humans tend to draw outlines first before the detail, heuristics can be proposed to remove strokes with probability dependent on their sequential order.

(iv) Zheng *et al*. [100] proposed a Bezier pivot based deformation (BPD) strategy and a mean stroke reconstruction (MSR) approach. These do not need any temporal information in the sketch. The main idea of MSR is to generate novel sketches with smaller intra-class variance.

(v) Liu *et al*. [101] proposed two sketch-specific data augmentation strategies: (a) Manually extract some strokes from sketch SVG files to construct noise stroke masks. Then, randomly apply the noise stroke masks to the original sketches to synthesize augmented sketches. (b) Randomly extract a patch from one sketch, and attach it to a given sketch.

(vi) Muhammad *et al*. [17] applied reinforcement learning to learn a sketch abstraction model that preserves the semantics of the sketch. Once trained, this model can be applied to generate augmentations of an input sketch at different abstraction levels.

Compared with augmentations on full image (*e.g.*, rotation, shift), the augmentation strategies above make better use of stroke information both locally and globally. However, only [17] makes (limited) use of human sketch variability to perform augmentation. In future an interesting direction is to learn from the variation in sketch style between different humans, and treat sketch augmentation as a cross-human style transfer problem.

4.1.3 Retrieval and Hashing

Sketch retrieval [49], [102], [103] methods aim to use a query sketch to retrieve similar samples from a gallery or database of sketches. Sketch retrieval is a challenging task due to abstraction, intra-class variation, drawing style variation, and feature sparsity. These properties make it difficult to localize repeatable feature points across sketches (*e.g.*, the manner of SIFT [104]) in order to perform retrieval with classic interest-point based retrieval approaches [104]. In the deep learning era, end-to-end feature learning has outperformed shallow features on various retrieval tasks in computer vision, and CNNs have also been used for sketch retrieval.

Common practice in image retrieval is to use CNNs to learn a vector embedding, and then perform retrieval/matching as Nearest Neighbor search. Therefore, most existing deep sketch retrieval models work in a similar metric learning manner, with research focusing on CNN architectures and loss function designs for effective sketch matching. Wang *et al*. [102] proposed a representative sketch retrieval framework, which has two key components, *i.e.*, pure convolutional layer based Siamese CNN backbone, and ℓ_1 norm distance based pair-wise loss between query and gallery images. The idea is two-fold: (i) Use the convolutional feature map to preserve the spatial information for sketches without point correspondence. (ii) Compute distance in feature space, and optimize for similar pairs to be nearby while different pairs are far apart.

Given the growing number of images available, there is also an increasing concern about scalability of retrieval, leading to studies of hashing-based methods where all

sketches are encoded and searched as binary hash code vectors, rather than real-valued vectors. Xu *et al*. [15] proposed the first deep sketch-hashing model. Their deep sketch hashing used a dual-branch CNN-RNN network to exploit both global appearance and local sequential stroke information, as well as a new center loss variant to ensure the learned embedding is more semantically meaningful.

The sketch retrieval/hashing methods mentioned so far exploit supervised information. If class labels are unavailable, adversarial training can be used to learn a feature representation for sketches. Based the Generative Adversarial Network (GAN) [105], Creswell *et al*. [103] proposed Sketch-GAN for unsupervised sketch retrieval, where both the query and gallery sketches are represented by the output features of the discriminator network.

4.1.4 Generation

Sketch generation has grown rapidly in recent years as deep learning-based approaches easily outperform earlier classic sketch generators [106], [107]. Sketch generation has several practical applications, *e.g*., synthesizing novel pictures, assisting artist design, and finishing incomplete sketches. It can be addressed using various deep learning tools, *e.g*., recurrent [108], variational autoencoder (VAE) [5], [109], [110], Generative Adversarial Network (GAN) [111], VAE-GAN [111], and reinforcement learning (RL) [111], [112].

The seminal model SketchRNN [5] is a sequence-to-sequence VAE for conditional and unconditional generation of vector sketches. Its encoder and decoder are implemented by bidirectional RNN [113] and unidirectional RNN, respectively. As stated earlier, free-hand sketches can be represented as a sequence of keypoints defining strokes. The main idea of SketchRNN is to simulate human sketching by sequential generation of these key points in terms of location and pen up/down status.

As shown in Figure 7, the VAE encoder of SketchRNN takes vector sketches as input, and encodes it as a vector $\mathbf{h}$, which is the RNN's last hidden state. This vector will be further encoded as two parameters μ and σ to model a Gaussian distribution $N(\mu, \sigma)$, from which a latent vector $\mathbf{z}$ will be sampled. Then, the Long Short Term Memory (LSTM) based VAE decoder will generate the coordinates and pen states of the key stroke points, conditional on $\mathbf{z}$. In particular, the coordinate and state for each key point is sampled from a Gaussian mixture model (GMM), and also used as input for the next decoder step. To improve SketchRNN to deal with multi-class generation, Cao *et al*. [114] propose a generative model named as "AI-Sketcher", which is also a VAE based network.

Another line of work within sketch generation uses differentiable rendering [115] or reinforcement learning [111], [112], [116], [117] to train policies that draw sketches iteratively according to different criteria such as adversarial training against human sketches [116]. This line of work often considers factors not addressed by SketchRNN such as brush style and color. By considering an interpretable latent representation of sketches, such methods can also potentially be used to de-render sketches into programs or symbols [118], [119].

Recently, there are several trends in sketch generation, notably: (i) Fine-grained sketch generation [110]. (ii) A novel evaluation metric "Ske-score" [111], aims to provide a better metric to quantify the goodness of generated vector sketches. (iii) Transformer-based architectures [120], [121] are being applied to sketch generation.

4.1.5 Grouping, Segmentation, and Parsing

Compared with sketch recognition, retrieval, and generation, there are several more fine-grained single-modal sketch analysis tasks: perceptual grouping, segmentation, and parsing. These tasks need sketch analysis at the local (stroke) level. Besides their intrinsic interest, these local sketch understanding techniques can also benefit other global tasks such as sketch-based image retrieval, sketch-based video retrieval [122], and sketch generation/synthesis. We next review recent advances in these areas.

Sketch Perceptual Grouping (SPG) Humans have the ability to perceptually group visual cues into semantic object parts/components, which has been widely researched in Gestalt psychology area [123], [124]. As shown in Figure 8, humans are able to perceptually group sketch strokes into semantic parts, *e.g*., airplane grouped into fuselage and wings. Thus, sketch perceptual grouping (SPG) is to imitate the human ability to group strokes into semantic parts. SPG has been studied with pre-deep learning methods [125], [126], [127], however progress has advanced rapidly since then. One representative application of SPG is to simplify sketches [128]. Moreover, SPG can also be used for sketch recognition [129], sketch semantic segmentation, synthesis [107], retrieval, fine-grained sketch-based image retrieval (FG-SBIR), sketch-based video retrieval [122], *etc*.

Li *et al*. [48], [130] contributed the largest SPG dataset to date of $20,000$ manually-annotated sketches across 25 object categories, and propose a universal deep grouper that can be applied to sketches of any category. Specifically, this deep universal grouper is also a sequence-to-sequence VAE with both generative and discriminative objectives: (i) Its generative loss enables the grouper to have the ability to handle unseen object categories and datasets. (ii) Its discriminative loss consists of a local grouping loss and a novel global grouping loss, to guarantee both local and global consistency in the grouping outputs.

Non-deep group methods mainly relied on thresholding low-level geometric properties among the strokes, often resulting in strokes with similar geometry but different semantics being grouped. Contemporary SPG methods consider more high-level semantic and temporal information due to their deep and recurrent representations.

Sketch Semantic Segmentation (SSS) Sketch semantic segmentation has drawn attention [125], [131], [132] in the free-hand sketch community as a classic topic prior to deep learning. Essentially, SPG performs stroke-level clustering, while SSS provides stroke-level classification. *i.e*., SSS provides explicit part labels (category names) for each stroke, while grouping only provides aggregation relationships. SSS and SPG are analogous to the classic (supervised) semantic segmentation [133] and unsupervised segmentation [134] respectively [48]. So SSS needs stronger supervision during training, namely what stroke categories, in contrast to SPG's stroke grouping.

Sketch semantic segmentation can potentially be addressed by conventional photo segmentation CNNs. How-

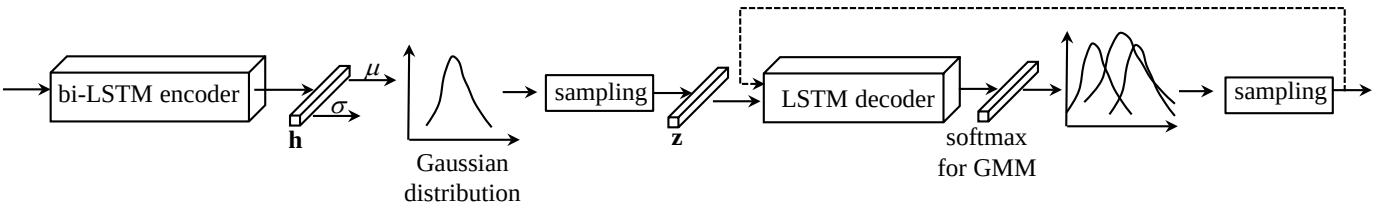


Fig. 7. The pipeline of SketchRNN [5]. The dotted arrow line denotes the recurrent processing of LSTM decoder. For simplicity, the recurrent processing of bi-LSTM encoder is not shown here.

ever, these do not exploit the vector representation of strokes or their temporal patterns, loading to the development of sketch-specific segmentation models.

Existing deep models for sketch semantic segmentation [20], [21], [50], [135], [136] can be grouped according to architectures: CNN, RNN, CNN-RNN dual-branch, GCN, VAE based models, *etc*.

Li *et al*. [137] trained a CNN-based network to transfer well-annotated segmentation and labels from a 3D dataset to sketch domain. They used annotated 3D data [131], [138] to produce edge-maps with partial annotations as the synthetic sketch data to train the segmentation network.

Qi *et al*. proposed SketchSegNet [136] and SketchSegNet+ [21]. SketchSegNet+ [21] considers the sketch stroke orderings and is able to work over multiple object categories. In particular, SketchSegNet and SketchSegNet+ work in RNN-based VAE framework, where the Gaussian mixture model (GMM) layers of SketchRNN are replaced with fully-connected softmax layers to predict the part labels. StrokeRNN [135] uses the same encoder as SketchRNN but extends the decoder to predict segmentation.

Besides the RNN backbone, other backbones have also been explored on sketch segmentation. SPFusionNet [50] uses late fusion of CNN-RNN branches to represent sketches for segmentation. SketchGCN [20] is a graph convolutional neural network for sketch semantic segmentation. It uses a mixed pooling block to fuse the intra-stroke and inter-stroke features from its two-branch architecture.

Sketch Parsing Recently, a new concept of "sketch parsing" [139], [140], [141], [142] has gained traction. As a kind of fine-grained semantic understanding of sketch, sketch parsing has already been applied to assist other sketch tasks [139], *e.g*., sketch-based image retrieval (SBIR). Sketch parse is related to sketch semantic segmentation. However, as shown in Figure 9, the goal is to perform pixel-wise segmentation of the semantic regions defined by the sketch, rather than the sketch strokes as in SSS. Existing models for sketch parsing thus far only use CNN-base networks to represent sketch the, *e.g*., SFSegNet [140] uses Deep Fully Convolutional Networks (FCN) [143].

4.1.6 Simplification and Abstraction

Sketch simplification has been widely studied [128], [144], [145] by the computer graphic community to merge strokes within a given sketch [145]. A typical pipeline is two-stage: sketch simplification [146], which geometrically clusters strokes into groups, *e.g*., by Gestalt principles; and generating a new line for each group. These graphic-community

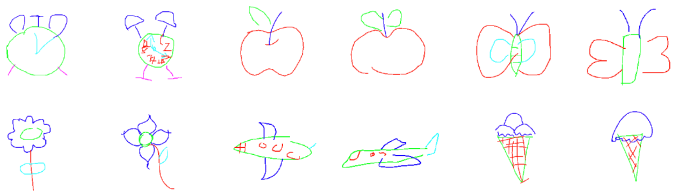


Fig. 8. Sketch samples of SPG dataset [130] (alarm clock , apple, butterfly, flower, airplane, ice cream). Semantically meaningful stroke groups are annotated by colors. Best viewed in color.

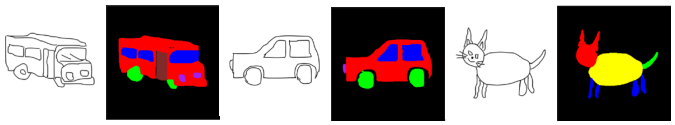


Fig. 9. Sketches (bus, car, cat) and ground truth annotations selected from sketch parsing paper [142]. The semantic parts and background are annotated by colors. Best viewed in color.

solutions were typically geometry-based and not posed as machine learning problems.

A different take on simplification focuses on the epitome of a sketch [147]. This was recently studied under the guise of "stroke-level sketch abstraction" in the free-hand sketch community. Stroke-level abstraction [17], [148] aims to abstract sketches by removing strokes that do not affect the recognizability of the sketch. Solving this problem provides several benefits: (1) It learns stroke saliency as a byproduct [17] – strokes that contribute the most to recognizability are the most salient. (2) It can be used to synthesize sketches of variable abstraction for generation, or data augmentation of discriminative sketch models [17] (3) It can be used for summarization and compression more broadly [148].

The stroke-level abstraction task can be seen as a discrete combinatorial optimization problem, and thus is intractable to solve with traditional methods. This was tackled in [17] by training a reinforcement learning (RL) policy to include/exclude each stroke in the sequence, while trading off between number of included strokes and recognizability. The RL-based abstraction idea was extended by [148] to reorder input strokes, rather than being constrained to the original input sequence; and further to enable customizing of the abstraction goal to preserve different aspects of 'recognizability' such as category vs. attributes.

Discussion Despite this good progress, simplification through *merging* multiple strokes into a coarser replacement,

rather than simply filtering them, remains an open question for deep learning-based sketch analysis.

4.2 Multi-Modal

Free-hand sketch has several cross-modal applications when paired with other data modalities. In this section we review sketch-related cross-modal topics including visual (*e.g.*, natural photo, 3D shape, video) and textual domains.

Nowadays, most visual retrieval approaches work under the "query-by-example" (QBE) [149] setting where users provide examples of the content that they seek. Compared with other query modalities (*e.g.*, photo, video, text), sketch has several unique advantages: In some scenarios users do not know the name of the object that they seek, or find it hard to describe (such as fine-grained details of a fashion item) in order to query-by-text. Meanwhile, it may be difficult or impractical to provide photos or video examples of the object that they seek. Sketch-based image retrieval provides a query modality where users express their target object by rendering their mental image in sketch. It is particularly useful when searching at the fine-grained instance-level. Thus sketch can be used as a modality to retrieve natural photo, manga [150], 3D shape, video, *etc*.

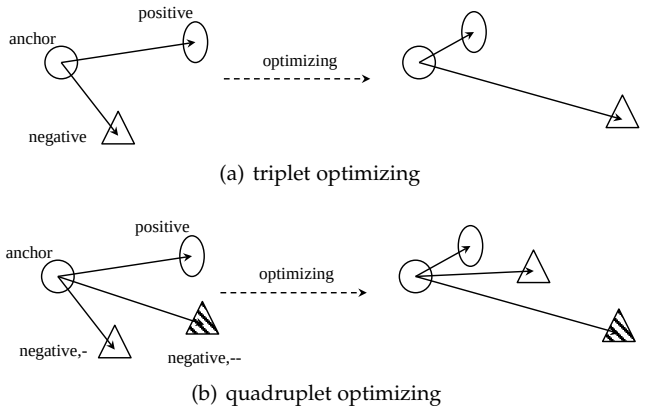
4.2.1 Sketch-Photo Retrieval

Sketch-photo retrieval is also known as sketch-based image retrieval (SBIR) [1], [4], [151], [152][7]. SBIR is challenging for all the reasons that sketch-analysis in general is challenging (sparse and abstract input). It is particularly challenging because of the difficulty of comparing sparse line drawings with dense pixel representations, especially when the input could be a very abstract, or iconic (symbolic) representation that is hard to compare directly to accurate perspective projection photos.

Figure 6 includes a taxonomy for SBIR. From the perspective of evaluation criterion, SBIR can be divided into conventional/coarse-grained SBIR (*i.e.*, category-level SBIR), mid-grained [153], and fine-grained SBIR (*i.e.*, instance-level SBIR). FG-SBIR is essentially a kind of instance-level retrieval [154]. From the perspective of retrieval embedding space, SBIR can be divided into common nearest-neighbor and fast hashing-based retrieval. From the perspective of supervision involved in training, SBIR can be divided into fully-supervised and zero-shot retrieval.

Category and Instance Level SBIR In coarse-grained SBIR, given a target sketch as query, a ranking list is returned based on the similarity (*e.g.*, Euclidean or Hamming distance). The retrieval is judged as correct, if the photo ranked at the top has the identical class label as the query. However, in fine-grained SBIR, the retrieval is judged as correct only when the returned photo is from the same *instance* pair as the query sketch. Based on SBIR ideas, several sketch-based commodity search engines have been implemented, *e.g.*, sketch-based skirt image retrieval [155], fine-grained sketch-based shoe [4], [22], chair [4], [22], and handbag [22] retrieval systems.

7. Note that the common setting for SBIR is sketch as a query modality for images, but most methods enable either modality to be used as a query if desired.



(a) triplet optimizing

(b) quadruplet optimizing

Fig. 10. Illustration of triplet and quadruplet ranking based optimizing objectives. The lengths of solid arrows denote the distances in embedding spaces. In quadruplet illustration, the hollow triangle denotes the negative sample from the category of anchor, while the shaded one denotes the negative sample from the remaining categories.

Some previous SBIR works [156], [157], [158] have used edge-maps (image contours) of photos as an approximation to corresponding sketch images in order to perform matching. Canny edge detector [159], Edge Boxes toolbox [160], and holistically-nested edge detection (HED) [161] were usually used to extract the edges from natural photos. However, this kind of hand-designed process is now commonly replaced by end-to-end deep feature learning.

Deep Sketch-based image retrieval (SBIR) has been widely studied [4], [7], [162], [163], [164], [165], [166], [167], [168], [169], [170], [171], [172], [173] in recent years. Existing SBIR solutions generally aim to train a joint embedding space where sketch and photo can be compared using nearest neighbor techniques. Common embedding learning approaches include: (a) contrastive comparison based methods (implemented by pair-wise loss [102]), (b) ranking based methods [4], [7], (c) reinforcement learning based methods [174], (d) deep canonical correlation analysis (DCCA) [175] based methods [176], (e) cross-domain dictionary learning [177], *etc*. The most widely-studied methods are ranking-based, including triplet ranking [166], [178], [179] and quadruplet ranking [180].

Ranking-Based SBIR We next introduce the popular triplet- and quadruplet-ranking SBIR methods in detail. As shown in Figure 10, given a sketch anchor $\mathbf{X}_n$ and its positive and negative photo retrieval candidates ($\mathbf{X}_{n,+}$, $\mathbf{X}_{n,-}$), the goal of triplet ranking is

$$\mathcal{D}(\mathcal{F}(\mathbf{X}_n), \mathcal{F}(\mathbf{X}_{n,+})) < \mathcal{D}(\mathcal{F}(\mathbf{X}_n), \mathcal{F}(\mathbf{X}_{n,-})). \quad (1)$$

where $\mathcal{D}(\cdot,\cdot)$ is a distance metric (*e.g.*, ℓ_2 distance). In common practice [4], [166], the negative sample is usually selected from the same class as the anchor. Specifically, the loss function of triplet ranking is typically

$$\mathcal{L}_{triplet} = \sum_{n=1}^{N} \max(0, \Delta + \|\mathcal{F}_\Theta(\mathbf{X}_n) - \mathcal{F}_\Theta(\mathbf{X}_{n,+})\|_2^2 - \|\mathcal{F}_\Theta(\mathbf{X}_n) - \mathcal{F}_\Theta(\mathbf{X}_{n,-})\|_2^2), \quad (2)$$

where Δ is the margin to guarantee the minimum distance between the embedding pairs of $\{\mathcal{F}(\mathbf{X}_n), \mathcal{F}(\mathbf{X}_{n,+})\}$ and $\{\mathcal{F}(\mathbf{X}_n), \mathcal{F}(\mathbf{X}_{n,-})\}$.

For quadruplet ranking [180], the input atom is a quadruplet of anchor $\mathbf{X}_n$, positive candidate $\mathbf{X}_{n,+}$, negative candidate $\mathbf{X}_{n,-}$ from the class of anchor, negative candidate $\mathbf{X}_{n,--}$ from a different class to anchor. As illustrated in Figure 10, the goal of quadruplet ranking is to ensure

$$\begin{aligned}\mathcal{D}(\mathcal{F}(\mathbf{X}_n), \mathcal{F}(\mathbf{X}_{n,+})) &< \mathcal{D}(\mathcal{F}(\mathbf{X}_n), \mathcal{F}(\mathbf{X}_{n,-})) \\ &< \mathcal{D}(\mathcal{F}(\mathbf{X}_n), \mathcal{F}(\mathbf{X}_{n,--})).\end{aligned} \tag{3}$$

Based on this, quadruplet ranking is essentially multi-task or multiple triplet ranking by constructing two extra triplet relationships, in order to encode more semantic information into the embedding space. In particular, Seddati *et al*. [180] constructed three triplets from each quadruplet, including $triplet_a = \{\mathbf{X}_n, \mathbf{X}_{n,+}, \mathbf{X}_{n,-}\}$, $triplet_b = \{\mathbf{X}_n, \mathbf{X}_{n,+}, \mathbf{X}_{n,--}\}$, and $triplet_c = \{\mathbf{X}_n, \mathbf{X}_{n,-}, \mathbf{X}_{n,--}\}$. Therefore, the quadruplet ranking loss is defined as

$$\mathcal{L}_{quadruplet} = \mathcal{L}_{triplet_a} + \lambda_b \mathcal{L}_{triplet_b} + \lambda_c \mathcal{L}_{triplet_c}, \tag{4}$$

where λ_b, λ_c are the weights.

In ranking-based SBIR, the anchor is usually from the sketch domain, and other samples are photos. Both triplet- and quadruplet-ranking can be used for either category-level or instance-level SBIR tasks.

The essential principle of triplet loss is using local partial orderings to establish a global ordered relationship in the embedding space, so that triplet ranking [46] can be understood as Topological Sorting[8]. The triplet annotations work as a partially ordered set. Compared with other loss functions, the main advantages of triplet loss are: (i) It helps to involve more local partial orderings and annotations to learn more fine-grained embedding space. (ii) Given a limited number of N training samples, their triplet orderings have C_N^3 combinations, producing significant annotation augmentation. This is beneficial for training deeper networks on smaller sketch datasets. It should be noted that the performance of triplet loss is heavily dependent on (a) the choice of margin parameter and (b) the triplet construction strategy.

Note also that ranking-based SBIR models can be improved by multi-task training along with classification [7], [181], [182]. Furthermore, rather than purely discriminative training, SBIR can also be tackled by generating one modality from the other [183], *e.g.*, using conditional GAN [184]; or using generative losses to regularize discriminative training [185]. SBIR training can also be combined with post-processing re-ranking [156], [158], [186] to refine the initial learned embedding spaces.

Network Architectures in SBIR SBIR models generally need two or more branches to process sketches and photos for comparison using the metrics introduced above. As shown in Figure 11, both triplet and quadruplet ranking models can use backbone networks that are: (i) Siamese, (ii) semi-heterogeneous, and (iii) heterogeneous: (i) Siamese networks [4] use full weight-parameter sharing across branches. (ii) Semi-heterogeneous network [187], [188] use partial weight sharing across the branches. Typically early layers are modality specific, and weight-tied layers are at deeper layers. (iii) In heterogeneous networks [189], the sketch branch uses independent parameters to the photo branches. The trade-offs underlying these architectures are that sharing weights enables more data (both sketch and photo) to be used to estimate parameters, reducing over-fitting. But separating weights enables the sketch/photo branches to adapt more specifically to their respective domains. Weight sharing considerations are discussed in more detail in [181].

8. https://en.wikipedia.org/wiki/Topological_sorting

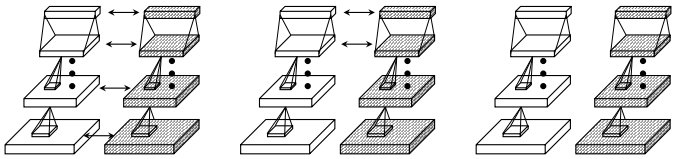


Fig. 11. Different weight sharing manners (left: Siamese, middle: semi-heterogeneous, right: heterogeneous) for CNN-based cross-modal networks. The hollow and shaded networks denote the branches for sketches and photos, respectively. The double sided arrows indicate sharing of weights.

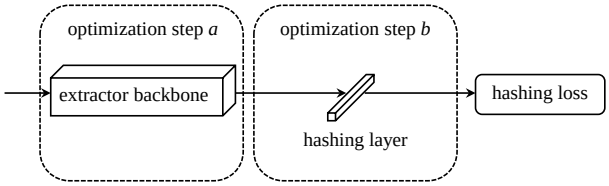


Fig. 12. Illustration of a classical deep hashing pipeline, where the extractor backbone and hashing layer are alternatively optimized in two separate steps.

Hashing-Based SBIR In order to achieve faster sketch-based image retrieval, recent research has studied optimizing the feature coding (*e.g.*, sketch-image hashing [190], [191]), and the feature map (*e.g.*, Asymmetric Feature Maps [192]).

In particular, sketch-image hashing (or hashing SBIR) has gained attention. Liu *et al*. [190], propose the first deep hashing model for SBIR, which is a classic deep hashing pipeline including: (i) feature extractor network, (ii) hashing layer with binary constraints, and (iii) hashing loss (see Figure 12). This classic pipeline in Figure 12 has been widely studied in photo oriented deep hashing [193], [194], where the hashing layer is typically fully-connected with sigmoid or tanh activation, and a discrete binary constraint. The loss functions of deep hashing models are often non-differentiable, due to the discrete binary constraints. Thus common practice is that the feature extractor backbone and hashing layer are alternatively optimized in two separate steps by fixing one and optimizing the other.

Existing SBIR hashing models work on the SBIR benchmarks, *i.e.*, Sketchy [7] (75K sketches) and TU-Berlin Extended [54] (20K). The scale of these benchmarks is not yet large enough to thoroughly test hashing SBIR methods.

Discussion Current issues in SBIR include: Self-supervised pre-training for SBIR [195], optimizing SBIR for early retrieval using partially drawn sketches, for example using reinforcement learning [174]; investigating whether costly sketch-photo annotation pairs can be replaced with edge-maps [196] and cross-category generalization of SBIR which is discussed next.

4.2.2 Zero-Shot SBIR

Many existing SBIR works assume that categories to be queried are included in the training set. In recent years, motivated by the zero-shot validation criterion for supervised photo retrieval [197], zero-shot sketch-based image retrieval (ZS-SBIR) has also been studied [52], [173], [198], [199], [200], [201], [202], [203], [204], [205], [206], [207], [208], [209]. Similar to natural photo zero-shot learning/recognition [210], [211], [212], ZS-SBIR systems aim to enable query and retrieval of categories that are from *unseen* categories. *i.e.*, categories that have not been involved in training stage. This is important in practice, *e.g.*, for an e-commerce application of SBIR, where new products should ideally be enrolled in the search engine without requiring re-training.

ZS-SBIR systems can follow conventional zero-shot learning methods [210], [211], [212] in exploiting auxiliary knowledge such as word vectors [213], attributes [214], or class hierarchy to define the model for the unseen class. However, directly synthesizing a retrieval model for novel-classes with auxiliary knowledge leads to the same challenges of ZSL (cross-category domain-shift [210]; inconvenient need to specify nameable categories at testing-time [210]). Meanwhile it would entail new challenges specific to SBIR: (i) Knowledge transfer needs to occur across both sketch and photo views. (ii) Some kinds of auxiliary knowledge may not make sense for sketch (*e.g.*, *banana-is-yellow* may be visible in photo but not sketch). Meanwhile, auxiliary knowledge transfer is not strictly necessary for retrieval in the way that it is for category recognition. Therefore many ZS-SBIR methods tackle the problem in a *domain generalization* [60] manner. That is, training a robust matching network on the training categories and then applying it directly to unseen testing categories.

Thus common approaches are to train ranking [52], [207] or generative [201], [202] models for retrieval, which are enhanced and mode robust by constraints such as domain-alignment losses [52], [201], [207] and auxiliary semantic knowledge reconstruction [52], [201]. In these cases the auxiliary semantic knowledge is only used to constrain representation learning at test time and is not required during training time as for conventional ZSL – thus maintaining the vision that SBIR should only depend on ability to depict and not to verbally describe.

Current directions include extending SBIR to the generalized zero-shot setting, where testing categories are a mix of training and unseen categories [201], [207]; extending sketch-photo hashing to the zero-shot setting [215]; and training SBIR without paired samples [200].

4.2.3 Sketch-Photo Generation

Sketch and photo based mutual generation (translation/synthesis) is a classic cross-modal topic of sketch research covering both: (i) sketch-to-photo generation [216], (ii) photo-to-sketch generation [45], [217], [218], [219]. In particular, sketch-to-photo generation methods have addressed: (a) sketch to photo, (b) sketch & photo to photo [18], [220], (c) sketch/edge & color to photo [47]. Sketch and photo based generation can be used to help users to create or design novel images in various practical applications: sketch-based photo editing [18], [220], [221], sketch to painting generation [222], cloth design [223], [224], sketch to natural photo generation [225], *etc*. In some cases, sketch-photo generations also involve style transfer [226], [227].

Note that: (i) Sketch-to-photo generation aims to solve cross-modal translation from abstract and sparse line drawings to pixel space, different to well-drawn sketch colorization [228], [229], [230]. (ii) Photo-to-sketch generation does not refer to extracting edge-map from natural photos [160], [231] (edge-maps of literal perspective projections), but instead needs models that learn to mimic human sketching and abstract drawing style.

Sketch and photo generation methods have been widely studied [101], [232], [233], [234], [235] based on various Generative Adversarial Net (GAN) [105] variants including conditional GAN [236], cycle GAN [232], and texture GAN [237].

Sketch-photo cross-modal generation is distinctively different to conventional photo-to-photo translation [233]. Existing photo-to-photo models can assume pixel-wise alignment between inputs and outputs. However, this requirement is strongly violated in the case of sketches and photos. Indeed no simple warping can provide pixel-wise alignment between sketch and photo given the potentially abstract or iconic nature of sketches. Thus existing sketch-photo synthesis work has made efforts to work around this issue, *e.g.*, using contextual GAN [238].

Generating sketch images as pixels suffers from the problem that blurriness in an image that should be made of sharp edges is very visible. However, an important difference between sketch-photo translation and photo-photo translation is that the raw format of sketch data is often a time-series of way-points. If such raw sketch representation is to be used, the encoder or decoder should be an RNN rather than CNN. The first example of such sequential vectorized photo-sketch translation is in [45], where sharp sketches are sequentially produced using a recurrent decoder.

Other current considerations are similar to that in image-image translation including building sketch-photo translation models that work based on unpaired samples.

4.2.4 Sketch-3D Retrieval

Sketch-3D retrieval refers to using sketch as query to retrieve 3D models [239]. Compared to SBIR, 3D retrieval is more challenging due to the larger domain gap between 2D sketch and 3D model.

Sketch-3D retrieval was studied well before the deep learning era [38]. Classic approaches often proceeded in a two-stage manner [240]: (i) View selection: Use an automatic procedure to select representative viewpoints of a given 3D model, hoping that one of the selected viewpoints is similar to that of the query sketches; and (ii) Projection and Matching: Project each selected view of the 3D model into 2D space by line rendering algorithm [241]. Then match the sketch against the 2D projections of the model based on predefined features such as SIFT [104]. See Figure 13 for an illustration. However, as argued in some previous work [2], view selection is a bottleneck of the two-stage approach, the "best" views is subjective and ambiguous. Moreover, matching based on hand-crafted features is inaccurate.

Gradually, sketch based 3D model retrieval has been studied within the end-to-end deep learning paradigm [2],

Fig. 13. Illustration of view matching across sketch and 3D shape. Images (from left to right: sketch, 3D shape, three random views of the 3D shape) are selected from [242].

[242], [243], [244], [245], [246]. As a representative method, Wang *et al*. [2] proposed to use two Siamese networks to learn the sketch and projected views directly in the end-to-end manner, which takes a quadruplet of sketches and projected viewpoints as input, and uses multiple pair-wise losses. In each quadruplet, two sketches ($\mathbf{X}_1$, $\mathbf{X}_2$) and two viewpoints ($\mathbf{V}_1$, $\mathbf{V}_2$) are randomly selected from sketch and 3D domains, respectively. For simplicity they assume that $\mathbf{X}_1$ and $\mathbf{X}_2$ are from the same category sharing a Siamese network, while $\mathbf{V}_1$ and $\mathbf{V}_2$ are also from the same category sharing another Siamese network. The quadruplet loss is

$$\begin{aligned}\mathcal{L}(\mathbf{X}_1,\mathbf{X}_2,\mathbf{V}_1,\mathbf{V}_2) = \mathcal{L}_{pair}(\mathbf{X}_1,\mathbf{X}_2) \\ + \mathcal{L}_{pair}(\mathbf{V}_1,\mathbf{V}_2) + \mathcal{L}_{pair}(\mathbf{X}_1,\mathbf{V}_1),\end{aligned} \tag{5}$$

The loss function terms $\mathcal{L}_{pair}(\mathbf{X}_1,\mathbf{X}_2)$ and $\mathcal{L}_{pair}(\mathbf{V}_1,\mathbf{V}_2)$ enable the network to learn category-level similarity within each domain; while the $\mathcal{L}_{pair}(\mathbf{X}_1,\mathbf{V}_1)$ term forces the network to learn cross-modal similarity. Given input samples a and b, the pair-wise loss function is defined as:

$$\mathcal{L}_{pair}(a,b) = \begin{cases} \alpha\mathcal{D}(\mathcal{F}_a(a),\mathcal{F}_b(b)), & \text{if } y_a \neq y_b, \\ \beta e^{\gamma\mathcal{D}(\mathcal{F}_a(a),\mathcal{F}_b(b))}, & \text{otherwise}, \end{cases} \tag{6}$$

where y_a and y_b are the corresponding class labels, and $\mathcal{F}_a(\cdot)$ and $\mathcal{F}_b(\cdot)$ denote the feature extractions that have been applied to a and b, respectively.

Besides this pair-wise deep metric learning, other deep metric learning methods also can be applied to Sketch-3D matching, *e.g.*, triplet ranking [247], [248] and deep correlation metric learning [249], [250].

Moreover, some previous works also studied how to represent 3D models more comprehensively in sketch based 3D retrieval tasks. For instance, Xie *et al*. [251] proposed to represent 3D models by computing the Wasserstein distance [252] based barycenters of multiple projections of 3D models.

4.2.5 Sketch-3D Generation

Sketch to 3D model generation is also an interesting cross-modal research topic analogous to the sketch-to-image generation discussed in Section 4.2.3. Using sketch to generate 3D models/shapes [253] is extremely challenging but has important applications such as sketch-based product design [254]. Compared to the other tasks discussed, this is relatively under-studied thus far. The existing deep learning based sketch-to-3D generation models are engineered for highly well-drawn or professional pencil sketches [255], [256]. Recently, 3D-to-sketch [257] generation has also been explored in a deep learning manner.

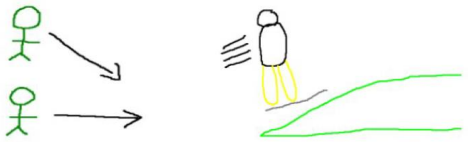


Fig. 14. Sketch samples randomly selected from sketch based video retrieval dataset TSF [258]. The left sketch depicts that two persons are approaching each other. The right one depicts that a person is gliding up the hillside.

4.2.6 Sketch-Video Retrieval

Sketch based video retrieval (SBVR) has been studied [258] prior to contemporary deep learning. SBVR is also extremely challenging due to the huge domain gap between free-hand sketch and video. In SBVR applications using sketch as query has the advantage that humans can use lines or arrow vectors to describe moving or other dynamic scenes, as illustrated in Figure 14. Thus sketch can be used to not only depict static objects and scenes, but also motion information.

Performing SBVR using sketches conveying both appearance and motion is challenging due to the need to segment the motion and appearance information from the sketch and use it to address the corresponding channels in the video gallery for retrieval. To address this challenge, a multi-stream multi-modality deep network was proposed in [259]. This study further extended SBVR to fine-grained SBVR to perform instance-level retrieval of videos given sketch queries.

Finally, we note that, motion sketch based crowd video retrieval [260], [261] has been studied recently, which is useful for video surveillance analysis.

4.2.7 Other Sketch-Related Multi-Modal Tasks

Recently some other interesting sketch based multi-modal tasks have emerged, *e.g.*, text-to-sketch generation [262] (*e.g.*, text instruction based conversational authoring of sketches [263]), sketch-based photo classifier generation [16], sketch-based pictionary games [44], [264], and sketch to photo contour transfer [265].

5 TorchSketch: The First Library for Free-Hand Sketch

In the free-hand sketch field, progress thus far has been hampered by lack of high-quality open-source software. To address this issue and support sketch-based research going forward, we contribute the first open-source sketch-focused deep learning library. Our library, termed `TorchSketch`, is built on top of PyTorch. In this section we briefly overview the major modules, functions, and features of this library. Further documentation can be found at https://github.com/PengBoXiangShang/torchsketch/. In this section, we also use `TorchSketch` to report the first controlled experimental comparison of sketch-related deep neural network architectures, including CNN, RNN, GNN, and TCN.

5.1 Modules and Functions

As shown in Figure 15, `TorchSketch` has three main modules: (i) `networks`, (ii) `data`, and (iii) `utils`.

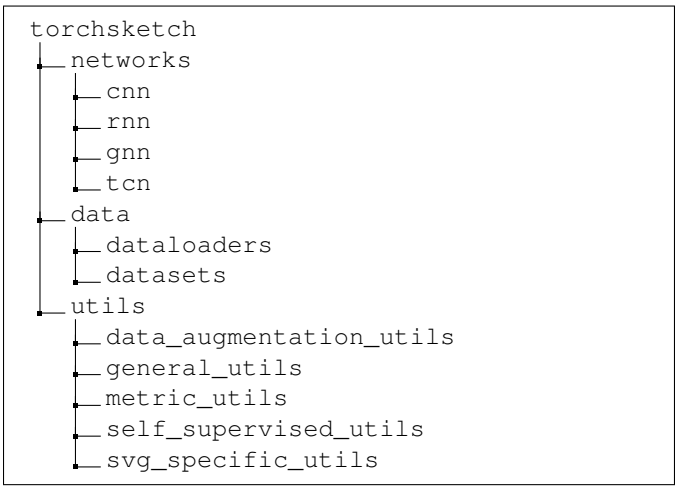


Fig. 15. Major module structure of `TorchSketch` library. Detailed APIs are not shown here due to the limited page space. See details in text and our GitHub page.

`networks`: has four sub-modules (*i.e.*, `cnn` , `rnn`, `gnn`, `tcn`), corresponding to four kinds of network architectures that can be applied to sketches. Each network is implemented as an independent class file, and encapsulated in the corresponding sub-module. The `gnn` sub-module provides implementations of graph convolutional network (GCN), graph attention network (GAT), graph transformer (GT), and multi-graph transformer (MGT).
`data`: has two sub-modules, `dataloaders` and `datasets`. `dataloaders` provides dataloaders for frequently-used sketch datasets, *e.g.*, TU-Berlin, Sketchy, QuickDraw. `datasets` provides the specific API for each dataset, integrating a series of functions including downloading, extraction, cleaning, MD5 checksum, and other preprocessing.
`utils`: provides diverse utility functions relevant to both free-hand sketch and more general deep learning. It includes five main sub-modules, as shown in Figure 15. Notably it includes a `self_supervised_utils` sub-module, providing algorithms for self-supervised learning in both free-hand sketch and natural photos.

The documents and sample code demonstrating how to use `TorchSketch` are also provided and will be continuously updated on GitHub [9] and PyPI [10] pages.

5.2 Major Features

The major features of `TorchSketch` are as follows.

Basic Features (i) `TorchSketch` supports both GPU-based and Python built-in multi-processing acceleration. Its efficient tensor computation and parallel computation only rely on PyTorch and Python, without involving other complex dependencies. (ii) It makes full use of the powerful functional programming features of Python. Numerous APIs of its `utils` module can be regarded as input parameters and passed into a unified multi-processing acceleration higher-order function. This makes parallel processing acceleration more user-friendly, and easier to use. (iii) It adopts as many pure functions as possible for stability and to avoid side effects. Overall `TorchSketch` is modular, flexible, and extensible, without overly complex design patterns and excessive encapsulation.

For Sketch Research (iv) `TorchSketch` integrates state-of-the-art sketch-oriented deep learning techniques such as CNN, RNN, GCN, and TCN – including the latest methods such as multi-graph transformer. (v) This library is supports numerous sketch datasets and formats including SVG, NumPy, PNG and JPEG – by providing numerous format-specific and format-conversion APIs. For instance, the sketch-specific data augmentation APIs are compatible with these formats. Furthermore, there is a unified format-convert API with parallel-processing acceleration, and 10+ SVG-specific APIs within the `utils` module. (vi) The library also provides several methods to support self-supervised learning for sketch.

Beyond Sketch Research (vii) Finally, there are some universal components that are applicable to deep learning beyond sketch. For instance, `metric_utils` provides APIs that compute various distances via torch tensor based matrix manipulation, which can be accelerated by GPUs and used for other computer vision tasks, *e.g.*, image retrieval, instance-level image retrieval, person re-identification (Re-ID), and hashing. All the components in `general_utils` are applicable to deep learning more broadly.

5.3 Experimental Comparison

As discussed in Section 2.1, sketch can be represented in several diverse formats such as raster image vs. waypoint sequence. These representations lend themselves to different neural network architectures. We therefore take the opportunity to use `TorchSketch` to perform the first thorough comparison of neural network architectures for sketch-recognition.

To this end we follow [19] in using 414K sketches drawn from QuickDraw. These are organized into training, validation, and test splits composed of 1000, 100, and 100 sketches respectively from each of the 345 QuickDraw categories. Following [15], we truncate or pad sketch samples to a uniform length of 100 key points/steps to facilitate efficient training of RNN, GNN, and TCN -based models, where each time-step is a $4D$ input (*i.e.*, two coordinates and two pen state bits). Sketch recognition is a fundamental topic within the field, and the QuickDraw sketches are subject to realistic abstraction, noise and drawing diversity. We therefore hope that this benchmark can help practitioners while supporting future research in the field.

The recognition accuracy across these architectures, as implemented by `TorchSketch`, are reported in Table 4. We can analyze these results with comparisons within and across architecture categories. Within each architecture, we can observe from Table 4: (i) For CNNs, the deeper networks (*e.g.*, DenseNet-161) have no obvious advantage compared to the shallower networks (*e.g.*, AlexNet, VGG). This is likely due to the sparsity of sketch where redundant convolution and pooling operations loose information about the sparse pixels. (ii) For RNNs, bidirectional networks outperform the unidirectional networks by a clear margin (0.66+ vs. 0.60+). This makes sense as human sketch ordering is

9. https://github.com/PengBoXiangShang/torchsketch/
10. https://pypi.org/project/torchsketch

TABLE 4
Comparison of recognition accuracy for different network architectures on a subset [19] of QuickDraw [5].

Architecture & Network		Input	Recognition Accuracy			Parameter Amount
			acc.@1	acc.@5	acc.@10	
Convolutional Neural Networks (CNNs)	AlexNet [86]	picture	0.6808	0.8847	0.9203	58,417,305
	VGG-11 [266]		0.6743	0.8814	0.9191	130,179,801
	VGG-13 [266]		0.6808	0.8881	0.9232	130,364,313
	VGG-16 [266]		0.6837	0.8889	0.9253	135,674,009
	VGG-19 [266]		0.6908	0.8839	0.9208	140,983,705
	Inception V3 [267]		0.7422	0.9189	0.9437	25,315,474
	ResNet-18 [89]		0.7164	0.9072	0.9381	11,353,497
	ResNet-34 [89]		0.7154	0.9083	0.9375	21,461,657
	ResNet-50 [89]		0.7043	0.8987	0.9303	24,214,937
	ResNet-101 [89]		0.7071	0.8992	0.9317	43,207,065
	ResNet-152 [89]		0.6924	0.8973	0.9312	58,850,713
	DenseNet-161 [268]		0.7008	0.8971	0.9302	27,234,105
	DenseNet-169 [268]		0.7173	0.9050	0.9358	13,058,905
	DenseNet-201 [268]		0.7050	0.9013	0.9331	18,755,673
	MobileNet V2 [269]		0.7310	0.9161	0.9429	2,665,817
Recurrent Neural Networks (RNNs)	LSTM	stroke vector	0.6068	0.8416	0.8931	2,593,881
	Bi-directional LSTM		0.6665	0.8820	0.9189	5,553,241
	GRU		0.6224	0.8574	0.9055	2,000,473
	Bi-directional GRU		0.6768	0.8854	0.9234	5,419,097
Graph Neural Networks (GNNs)	Graph Convolutional Network (GCN) [270]	stroke vector	0.6800	0.8869	0.9224	6,948,441
	Graph Attention Network (GAT) [271]		0.6977	0.8952	0.9298	11,660,889
	Vanilla Transformer [272]		0.5249	0.7802	0.8486	14,029,401
	Multi-Graph Transformer (Base) [19]		0.7070	0.9030	0.9351	10,096,601
	Multi-Graph Transformer (Large) [19]		0.7280	0.9106	0.9387	39,984,729
Textual Convolutional Network (TCN)	TCN [92]	stroke vector	0.5511	0.8020	0.8646	2,750,873

only loosely consistent [107]. (iii) For GNNs, multi-graph transformer (MGT) outperforms graph convolutional network (GCN) and graph attention network (GAT).

Across the architectures, we can observe: (i) Thus far the best CNN networks (InceptionV3) outperform the best sequential networks. This may be because the sequential networks (RNN, GNN, and TCN) truncate the input coordinate sequences. (ii) Among GNNs, the multi-graph transformer [19] comes closest to matching peak CNN performance. (iii) Compared with CNNs and GNNs, RNNs and TCN have significantly less parameters. However (iv) TCN, performs unsatisfactorily in this fully-supervised setting.

6 Discussion

6.1 Open Problems

6.1.1 Data and Annotations

Annotation Single-modal sketch datasets have begun to provide some annotation such as grouping [130] and segmentation [21]. However more fine-grained annotation of this type is necessary, particularly sketch attributes are lacking. As discussed in Section 4.2.3, existing multi-modal sketch benchmark annotation is primarily in terms of *pairings* (*e.g.*, sketch-photo, sketch-3D). However, fine-grained/local annotations (such as stroke-contour, parts, and attributes) would enable richer cross-modal alignment models to be learned.

Meta-Data and Fairness Unlike photos, sketches are uniquely influenced by the demographics, perception, memory, and drawing style of the artist; as well as the conditions under which they are drawn (*i.e.*, time-limited or not). Most existing datasets do not take care to acquire balanced samples of users across background, age, gender, *etc*; or record such meta-data about their participants. However, such sampling and meta-data are necessary for studying changes in drawing style with these covariates, as well as for ensuring that sketch-based applications work well for users of different types. Furthermore, existing sketch datasets are mainly created as bespoke efforts by researchers or casual participants in online games with time-pressure. These may lead to sketches that are either excessively well dawn, or too poorly drawn. Data collected under a variety of drawing conditions, and meta-data about those conditions, would help sketch research in future.

6.1.2 Architectures and Sketch-Specific Design

Architectures As discussed in Section 5.3 and Table 4, there are a variety of network architectures that can be applied to sketch, and these lead to a range of performances. These results suggest that performance will continue to advance as better architectures within each family are developed. Thus the best architecture for sketch perception is still an open question.

Sketch-Specific Design Another open question is to what extent sketch-specific designs are important vs. generic computer vision architectures and learning algorithms. Clearly sketch has unique challenges (sequential time-series nature, sparsity, abstraction, artist style, *etc*) that can be better taken into account with sketch-specific designs. However the broader vision and learning community can bring greater effort to bear on developing more advanced general purpose models. Therefore it remains to be seen in which sketch applications sketch-specific designs can take a decisive lead over generic architectures and algorithms. For example, sketch-specific designs may be more important in fine-grained tasks such as segmentation, grouping, and FG-SBIR compared to the simplest coarse-grained object categorization task.

6.2 Potential Research Directions

In this section we identify some potential research directions that we believe are promising in future:

Diverse Sketch Subjects Existing sketch datasets and applications focus on sketches depicting objects and scenes. However, in practical applications users may be interested in machines understanding more diverse sketched concepts, *e.g.*, sheets, curves, histograms [273], maps [274], and user interface (UI) prototype drawings [275]. Sketch also can be studied with hand-written character together.

Sketch Pressure and Color Existing free-hand sketches are collected by common touch-screen devices, *e.g.*, mobile phone, tablet, *etc*. Here the position of strokes is the main feature. However recent devices can increasingly sense pressure along strokes, providing a new sketch feature to analyze. Similarly, existing sketch analysis has mostly focused on black or grayscale sketches. Many drawing applications allow lines to be colored, and existing sketch analysis methods can be upgraded to analyze this cue.

Diversity, Style, and Robustness Sketch is particularly subjective in terms of influence by the user's drawing style. Robustness to style is an under-studied area of sketch research. Similarly, taking the native format of sketch, existing adversarial attack studies could be extended to sketch domain by studying adversarial *strokes* or waypoints rather than pixel perturbations.

Sketch as a Robustness Test Sketch images differ dramatically from photos, yet are easily recognized by humans. Sketch images such as ImageNet-Sketch [32], SketchTransfer [53], and PACS [60] can thus be used to benchmark the robustness and generalization ability of more general image-oriented deep learning models. Going beyond recognition, similar robustness benchmarks could be established on other instance-level retrieval problems such as person Re-ID.

Robot Sketching Finally, extending generative sketch models to embodied robot drawing [276] is an interesting direction.

7 Conclusion

This survey reviewed the landscape of contemporary deep-learning based sketch research. We introduced the unique aspects of sketch in terms of sketch-specific challenges and diverse potential representations, and analyzed both existing datasets and existing methods in terms of a rich ecosystem of uni-modal and multi-modal sketch analysis tasks. We discussed open problems and under-studied research directions throughout. Finally, to support future sketch-based research we contribute `TorchSketch` as an efficient and full featured open-source implementation of several state-of-the-art sketch-oriented deep learning techniques. We hope this survey will help new researchers and practitioners get up to speed, provide a convenient reference for sketch experts, and encourage future progress in this exciting field.

References

[1] R. Hu and J. Collomosse, "A performance evaluation of gradient field hog descriptor for sketch based image retrieval," *CVIU*, 2013.

[2] F. Wang, L. Kang, and Y. Li, "Sketch-based 3d shape retrieval using convolutional neural networks," in *CVPR*, 2015.

[3] Q. Yu, Y. Yang, F. Liu, Y.-Z. Song, T. Xiang, and T. M. Hospedales, "Sketch-a-net: A deep neural network that beats humans," *IJCV*, 2017.

[4] Q. Yu, F. Liu, Y.-Z. Song, T. Xiang, T. M. Hospedales, and C.-C. Loy, "Sketch me that shoe," in *CVPR*, 2016.

[5] D. Ha and D. Eck, "A neural representation of sketch drawings," in *ICLR*, 2018.

[6] M. Eitz, J. Hays, and M. Alexa, "How do humans sketch objects?" *TOG*, 2012.

[7] P. Sangkloy, N. Burnell, C. Ham, and J. Hays, "The sketchy database: learning to retrieve badly drawn bunnies," *TOG*, 2016.

[8] F. Huang, J. F. Canny, and J. Nichols, "Swire: Sketch-based user interface retrieval," in *CHI*, 2019.

[9] S. Suleri, V. P. Sermuga Pandian, S. Shishkovets, and M. Jarke, "Eve: A sketch-based software prototyping workbench," in *CHI*, 2019.

[10] K. C. Kwan and H. Fu, "Mobi3dsketch: 3d sketching in mobile ar," in *CHI*, 2019.

[11] D. Gasques, J. G. Johnson, T. Sharkey, and N. Weibel, "What you sketch is what you get: Quick and easy augmented reality prototyping with pintar," in *CHI*, 2019.

[12] I. E. Sutherland, "Sketchpad a man-machine graphical communication system," *Simulation*, 1964.

[13] C. F. Herot, "Graphical input through machine recognition of sketches," *TOG*, 1976.

[14] Q. Yu, Y. Yang, Y.-Z. Song, T. Xiang, and T. M. Hospedales, "Sketch-a-net that beats humans," in *BMVC*, 2015.

[15] P. Xu, Y. Huang, T. Yuan, K. Pang, Y.-Z. Song, T. Xiang, T. M. Hospedales, Z. Ma, and J. Guo, "Sketchmate: Deep hashing for million-scale human sketch retrieval," in *CVPR*, 2018.

[16] C. Hu, D. Li, Y.-Z. Song, T. Xiang, and T. M. Hospedales, "Sketch-a-classifier: Sketch-based photo classifier generation," in *CVPR*, 2018.

[17] U. Riaz Muhammad, Y. Yang, Y.-Z. Song, T. Xiang, and T. M. Hospedales, "Learning deep sketch abstraction," in *CVPR*, 2018.

[18] T. Portenier, Q. Hu, A. Szabo, S. A. Bigdeli, P. Favaro, and M. Zwicker, "Faceshop: Deep sketch-based face image editing," *TOG*, 2018.

[19] P. Xu, C. K. Joshi, and X. Bresson, "Multi-graph transformer for free-hand sketch recognition," *arXiv preprint arXiv:1912.11258*, 2019.

[20] L. Yang, J. Zhuang, H. Fu, K. Zhou, and Y. Zheng, "Sketchgcn: Semantic sketch segmentation with graph convolutional networks," *arXiv preprint arXiv:2003.00678*, 2020.

[21] Y. Qi and Z.-H. Tan, "Sketchsegnet+: An end-to-end learning of rnn for multi-class sketch semantic segmentation," *Access*, 2019.

[22] J. Song, Q. Yu, Y.-Z. Song, T. Xiang, and T. M. Hospedales, "Deep spatial-semantic attention for fine-grained sketch-based image retrieval," in *ICCV*, 2017.

[23] S. Ouyang, T. M. Hospedales, Y.-Z. Song, and X. Li, "Forgetmenot: Memory-aware forensic facial sketch matching," in *CVPR*, 2016.

[24] C. Hu, D. Li, Y.-Z. Song, and T. M. Hospedales, "Now you see me: Deep face hallucination for unviewed sketches." in *BMVC*, 2017.

[25] S. Nagpal, M. Singh, R. Singh, M. Vatsa, A. Noore, and A. Majumdar, "Face sketch matching via coupled deep transform learning," in *ICCV*, 2017.

[26] D.-P. Fan, S. Zhang, Y.-H. Wu, Y. Liu, M.-M. Cheng, B. Ren, P. L. Rosin, and R. Ji, "Scoot: A perceptual metric for facial sketches," in *ICCV*, 2019.

[27] L. Pang, Y. Wang, Y.-Z. Song, T. Huang, and Y. Tian, "Cross-domain adversarial feature learning for sketch re-identification," in *MM*, 2018.

[28] M. Huang, J. Lin, N. Chen, W. An, and W. Zhu, "Reversed sketch: A scalable and comparable shape representation," *PR*, 2018.

[29] Q. Zheng, Z. Li, and A. Bargteil, "Learning to shade hand-drawn sketches," in *CVPR*, 2020.

[30] S.-B. Chen, P.-C. Wang, B. Luo, C. H. Ding, and J. Zhang, "Sragan: Generating colour landscape photograph from sketch," in *IJCNN*, 2019.

[31] M. R. Amer, S. Yousefi, R. Raich, and S. Todorovic, "Monocular extraction of 2.1 d sketch using constrained convex optimization," *IJCV*, 2015.

[32] H. Wang, S. Ge, Z. Lipton, and E. P. Xing, "Learning robust global representations by penalizing local predictive power," in *NeurIPS*, 2019.

[33] K. Chen, I. Rabkina, M. D. McLure, and K. D. Forbus, "Human-like sketch object recognition via analogical learning," in *AAAI*, 2019.

[34] X. Han, K. Hou, D. Du, Y. Qiu, Y. Yu, K. Zhou, and S. Cui, "Caricatureshop: Personalized and photorealistic caricature sketching," *arXiv preprint arXiv:1807.09064*, 2018.

[35] 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 *ECCV*, 2018.

[36] L. Zhao, F. Han, X. Peng, X. Zhang, M. Kapadia, V. Pavlovic, and D. N. Metaxas, "Cartoonish sketch-based face editing in videos using identity deformation transfer," *Computers Graphics*, 2019.

[37] J. Delanoy, M. Aubry, P. Isola, A. A. Efros, and A. Bousseau, "3d sketching using multi-view deep volumetric prediction," *CGIT*, 2018.

[38] B. Li, Y. Lu, A. Godil, T. Schreck, B. Bustos, A. Ferreira, T. Furuya, M. J. Fonseca, H. Johan, T. Matsuda *et al.*, "A comparison of methods for sketch-based 3d shape retrieval," *CVIU*, 2014.

[39] N. Prajapati and G. Prajapti, "Sketch based image retrieval system for the web-a survey," *IJCSIT*, 2015.

[40] M. Indu and K. Kavitha, "Survey on sketch based image retrieval methods," in *ICCPCT*, 2016.

[41] Y. Li and W. Li, "A survey of sketch-based image retrieval," *Machine Vision and Applications*, 2018.

[42] X. Zhang, X. Li, Y. Liu, and F. Feng, "A survey on freehand sketch recognition and retrieval," *Image and Vision Computing*, 2019.

[43] D. Yu, L. Li, Y. Zheng, M. Lau, Y.-Z. Song, C.-L. Tai, and H. Fu, "Sketchdesc: Learning local sketch descriptors for multi-view correspondence," *arXiv preprint arXiv:2001.05744*, 2020.

[44] R. K. Sarvadevabhatla, S. Surya, T. Mittal, and V. B. Radhakrishnan, "Pictionary-style word-guessing on hand-drawn object sketches: dataset, analysis and deep network models," *TPAMI*, 2020.

[45] J. Song, K. Pang, Y.-Z. Song, T. Xiang, and T. M. Hospedales, "Learning to sketch with shortcut cycle consistency," in *CVPR*, 2018.

[46] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *CVPR*, 2015.

[47] P. Sangkloy, J. Lu, C. Fang, F. Yu, and J. Hays, "Scribbler: Controlling deep image synthesis with sketch and color," in *CVPR*, 2017.

[48] K. Li, K. Pang, J. Song, Y.-Z. Song, T. Xiang, T. M. Hospedales, and H. Zhang, "Universal sketch perceptual grouping," in *ECCV*, 2018.

[49] J. Choi, H. Cho, J. Song, and S. M. Yoon, "Sketchhelper: Real-time stroke guidance for freehand sketch retrieval," *TMM*, 2019.

[50] F. Wang, S. Lin, H. Wu, H. Li, R. Wang, X. Luo, and X. He, "Sp-fusionnet: Sketch segmentation using multi-modal data fusion," in *ICME*, 2019.

[51] R. K. Sarvadevabhatla, S. Suresh, and R. V. Babu, "Object category understanding via eye fixations on freehand sketches," *TIP*, 2017.

[52] S. Dey, P. Riba, A. Dutta, J. Llados, and Y.-Z. Song, "Doodle to search: Practical zero-shot sketch-based image retrieval," in *CVPR*, 2019.

[53] A. Lamb, S. Ozair, V. Verma, and D. Ha, "Sketchtransfer: A new dataset for exploring detail-invariance and the abstractions learned by deep networks," in *WACV*, 2020.

[54] H. Zhang, S. Liu, C. Zhang, W. Ren, R. Wang, and X. Cao, "Sketchnet: Sketch classification with web images," in *CVPR*, 2016.

[55] T. Jiang, G.-S. Xia, and Q. Lu, "Sketch-based aerial image retrieval," in *ICIP*, 2017.

[56] T.-B. Jiang, G.-S. Xia, Q.-K. Lu, and W.-M. Shen, "Retrieving aerial scene images with learned deep image-sketch features," *JCST*, 2017.

[57] X. Wang, X. Duan, and X. Bai, "Deep sketch feature for cross-domain image retrieval," *Neurocomputing*, 2016.

[58] M. Eitz, R. Richter, T. Boubekeur, K. Hildebrand, and M. Alexa, "Sketch-based shape retrieval," *TOG*, 2012.

[59] B. Li, Y. Lu, C. Li, A. Godil, T. Schreck, M. Aono, M. Burtscher, H. Fu, T. Furuya, H. Johan *et al.*, "Shrec14 track: Extended large scale sketch-based 3d shape retrieval," in *Eurographics workshop on 3D object retrieval*, 2014.

[60] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales, "Deeper, broader and artier domain generalization," in *ICCV*, 2017.

[61] C. Xiao, C. Wang, L. Zhang, and L. Zhang, "Sketch-based image retrieval via shape words," in *ICMR*, 2015.

[62] L. Castrejon, Y. Aytar, C. Vondrick, H. Pirsiavash, and A. Torralba, "Learning aligned cross-modal representations from weakly aligned data," in *CVPR*, 2016.

[63] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation," in *ICCV*, 2019.

[64] T. Kato, T. Kurita, N. Otsu, and K. Hirata, "A sketch retrieval method for full color image database-query by visual example," in *ICPR*, 1992.

[65] Y. Li, T. M. Hospedales, Y.-Z. Song, and S. Gong, "Fine-grained sketch-based image retrieval by matching deformable part models," in *BMVC*, 2014.

[66] T. de Vries, I. Misra, C. Wang, and L. van der Maaten, "Does object recognition work for everyone?" in *CVPR Workshops*, 2019.

[67] S. Barocas, M. Hardt, and A. Narayanan, *Fairness and Machine Learning*. fairmlbook.org, 2019, http://www.fairmlbook.org.

[68] Y. Matsui, T. Shiratori, and K. Aizawa, "Drawfromdrawings: 2d drawing assistance via stroke interpolation with a sketch database," *TVCG*, 2016.

[69] K. D. Forbus, B. Garnier, B. Tikoff, W. Marko, M. Usher, and M. McLure, "Sketch worksheets in stem classrooms: Two deployments," in *AAAI*, 2018.

[70] Y. Ye, Y. Lu, and H. Jiang, "Human's scene sketch understanding," in *ICMR*, 2016.

[71] Y. Xie, P. Xu, and Z. Ma, "Deep zero-shot learning for scene sketch," *arXiv preprint arXiv:1905.04510*, 2019.

[72] O. Seddati, S. Dupont, and S. Mahmoudi, "Deepsketch: deep convolutional neural networks for sketch recognition and similarity search," in *CBMI*, 2015.

[73] Y. Zhang, Y. Zhang, and X. Qian, "Deep neural networks for free-hand sketch recognition," in *Pacific Rim Conference on Multimedia*, 2016.

[74] J. Guo, C. Wang, E. Roman-Rangel, H. Chao, and Y. Rui, "Building hierarchical representations for oracle character and sketch recognition," *TIP*, 2016.

[75] P. Ballester and R. M. Araujo, "On the performance of googlenet and alexnet applied to sketches," in *AAAI*, 2016.

[76] O. Seddati, S. Dupont, and S. Mahmoudi, "Deepsketch 2: Deep convolutional neural networks for partial sketch recognition," in *CBMI*, 2016.

[77] H. Zhang, P. She, Y. Liu, J. Gan, X. Cao, and H. Foroosh, "Learning structural representations via dynamic object landmarks discovery for sketch recognition and retrieval," *TIP*, 2019.

[78] O. Seddati, S. Dupont, and S. Mahmoudi, "Deepsketch2image: deep convolutional neural networks for partial sketch recognition and image retrieval," in *MM*, 2016.

[79] K. Zhang, W. Luo, L. Ma, and H. Li, "Cousin network guided sketch recognition via latent attribute warehouse," in *AAAI*, 2019.

[80] X. Zhang, Y. Huang, Q. Zou, Y. Pei, R. Zhang, and S. Wang, "A hybrid convolutional neural network for sketch recognition," *PRL*, 2020.

[81] R. K. Sarvadevabhatla, J. Kundu *et al.*, "Enabling my robot to play pictionary: Recurrent neural networks for sketch recognition," in *MM*, 2016.

[82] Q. Jia, M. Yu, X. Fan, and H. Li, "Sequential dual deep learning with shape and texture features for sketch recognition," *arXiv preprint arXiv:1708.02716*, 2017.

[83] J.-Y. He, X. Wu, Y.-G. Jiang, B. Zhao, and Q. Peng, "Sketch recognition with deep visual-sequential fusion model," in *MM*, 2017.

[84] A. Prabhu, V. Batchu, S. A. Munagala, R. Gajawada, and A. Namboodiri, "Distribution-aware binarization of neural networks for sketch recognition," in *WACV*, 2018.

[85] L. Li, C. Zou, Y. Zheng, Q. Su, H. Fu, and C.-L. Tai, "Sketch-r2cnn: An attentive network for vector sketch recognition," *arXiv preprint arXiv:1811.08170*, 2018.

[86] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NeurIPS*, 2012.

[87] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun, "Bayesian face revisited: A joint formulation," in *ECCV*, 2012.

[88] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[89] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[90] A. v. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," *arXiv preprint arXiv:1601.06759*, 2016.

[91] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[92] P. Xu, Z. Song, Q. Yin, Y.-Z. Song, and L. Wang, "Deep self-supervised representation learning for free-hand sketch," *arXiv preprint arXiv:2002.00867*, 2020.

[93] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification*. John Wiley & Sons, 2012.

[94] A. Mishra and A. K. Singh, "Deep embedding using bayesian risk minimization with application to sketch recognition," in *ACCV*, 2018.

[95] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, 1995.

[96] F. Liu, X. Deng, Y.-K. Lai, Y.-J. Liu, C. Ma, and H. Wang, "Sketchgan: Joint sketch completion and recognition with generative adversarial network," in *CVPR*, 2019.

[97] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *IJCV*, 2015.

[98] R. K. Sarvadevabhatla and R. V. Babu, "Freehand sketch recognition using deep features," *arXiv preprint arXiv:1502.00254*, 2015.

[99] B. Graham, "Spatially-sparse convolutional neural networks," *arXiv preprint arXiv:1409.6070*, 2014.

[100] Y. Zheng, H. Yao, X. Sun, S. Zhang, S. Zhao, and F. Porikli, "Sketch-specific data augmentation for freehand sketch recognition," *arXiv preprint arXiv:1910.06038*, 2019.

[101] R. Liu, Q. Yu, and S. Yu, "An unpaired sketch-to-photo translation model," *arXiv preprint arXiv:1909.08313*, 2019.

[102] F. Wang and Y. Li, "Spatial matching of sketches without point correspondence," in *ICIP*, 2015.

[103] A. Creswell and A. A. Bharath, "Adversarial training for sketch retrieval," in *ECCV*, 2016.

[104] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, 2004.

[105] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *NeurIPS*, 2014.

[106] G. Hinton and V. Nair, "Inferring motor programs from images of handwritten digits," in *NeurIPS*, 2005.

[107] Y. Li, Y.-Z. Song, T. M. Hospedales, and S. Gong, "Free-hand sketch synthesis with deformable stroke models," *IJCV*, 2017.

[108] K. Sasaki and T. Ogata, "Adaptive drawing behavior by visuomotor learning using recurrent neural networks," *IEEE Transactions on Cognitive and Developmental Systems*, 2019.

[109] N. Jaques, J. McCleary, J. Engel, D. Ha, F. Bertsch, R. Picard, and D. Eck, "Learning via social awareness: Improving a deep generative sketching model with facial feedback," *arXiv preprint arXiv:1802.04877*, 2018.

[110] A. Jenal, N. Savinov, T. Sattler, and G. Chaurasia, "Rnn-based generative model for fine-grained sketching," *arXiv preprint arXiv:1901.03991*, 2019.

[111] S. Balasubramanian, V. N. Balasubramanian *et al.*, "Teaching gans to sketch in vector format," *arXiv preprint arXiv:1904.03620*, 2019.

[112] T. Zhou, C. Fang, Z. Wang, J. Yang, B. Kim, Z. Chen, J. Brandt, and D. Terzopoulos, "Learning to doodle with stroke demonstrations and deep q-networks," in *BMVC*, 2018.

[113] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *TSP*, 1997.

[114] N. Cao, X. Yan, Y. Shi, and C. Chen, "Ai-sketcher: A deep generative model for producing high-quality sketches," in *AAAI*, 2019.

[115] N. Zheng, Y. Jiang, and D. Huang, "Strokenet: A neural painting environment," in *ICLR*, 2019.

[116] Y. Ganin, T. Kulkarni, I. Babuschkin, S. M. A. Eslami, and O. Vinyals, "Synthesizing programs for images using reinforced adversarial learning," in *ICML*, 2018.

[117] J. F. Mellor, E. Park, Y. Ganin, I. Babuschkin, T. Kulkarni, D. Rosenbaum, A. Ballard, T. Weber, O. Vinyals, and S. Eslami, "Unsupervised doodling and painting with improved spiral," *arXiv preprint arXiv:1910.01007*, 2019.

[118] K. Ellis, D. Ritchie, A. Solar-Lezama, and J. B. Tenenbaum, "Learning to infer graphics programs from hand-drawn images," in *NeurIPS*, 2018.

[119] V. Egiazarian, O. Voynov, A. Artemov, D. Volkhonskiy, A. Safin, M. Taktasheva, D. Zorin, and E. Burnaev, "Deep vectorization of technical drawings," *arXiv preprint arXiv:2003.05471*, 2020.

[120] L. S. F. Ribeiro, T. Bui, J. Collomosse, and M. Ponti, "Sketchformer: Transformer-based representation for sketched structure," in *CVPR*, 2020.

[121] S. Wieluch and F. Schwenker, "Strokecoder: Path-based image generation from single examples using transformers," *arXiv preprint arXiv:2003.11958*, 2020.

[122] J. P. Collomosse, G. McNeill, and L. Watts, "Free-hand sketch grouping for video retrieval," in *ICPR*, 2008.

[123] J. Wagemans, J. H. Elder, M. Kubovy, S. E. Palmer, M. A. Peterson, M. Singh, and R. von der Heydt, "A century of gestalt psychology in visual perception: I. perceptual grouping and figure–ground organization." *Psychological bulletin*, 2012.

[124] K. Koffka, *Principles of Gestalt psychology*. Routledge, 2013.

[125] Z. Sun, C. Wang, L. Zhang, and L. Zhang, "Free hand-drawn sketch segmentation," in *ECCV*, 2012.

[126] Y. Qi, J. Guo, Y.-Z. Song, T. Xiang, H. Zhang, and Z.-H. Tan, "Im2sketch: Sketch generation by unconflicted perceptual grouping," *Neurocomputing*, 2015.

[127] 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 *CVPR*, 2015.

[128] X. Liu, T.-T. Wong, and P.-A. Heng, "Closure-aware sketch simplification," *TOG*, 2015.

[129] X. Wang, X. Chen, and Z. Zha, "Sketchpointnet: A compact network for robust sketch recognition," in *ICIP*, 2018.

[130] K. Li, K. Pang, Y.-Z. Song, T. Xiang, T. M. Hospedales, and H. Zhang, "Toward deep universal sketch perceptual grouper," *TIP*, 2019.

[131] Z. Huang, H. Fu, and R. W. Lau, "Data-driven segmentation and labeling of freehand sketches," *TOG*, 2014.

[132] R. G. Schneider and T. Tuytelaars, "Example-based sketch segmentation and labeling using crfs," *TOG*, 2016.

[133] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *TPAMI*, 2017.

[134] C. Wang, B. Yang, and Y. Liao, "Unsupervised image segmentation using convolutional autoencoder with total variation regularization as preprocessing," in *ICASSP*, 2017.

[135] K. Kaiyrbekov and M. Sezgin, "Stroke-based sketched symbol reconstruction and segmentation," *arXiv preprint arXiv:1901.03427*, 2019.

[136] X. Wu, Y. Qi, J. Liu, and J. Yang, "Sketchsegnet: A rnn model for labeling sketch strokes," in *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2018.

[137] L. Li, H. Fu, and C.-L. Tai, "Fast sketch segmentation and labeling with deep learning," *IEEE computer graphics and applications*, 2018.

[138] L. Yi, V. G. Kim, D. Ceylan, I. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, L. Guibas *et al.*, "A scalable active framework for region annotation in 3d shape collections," *TOG*, 2016.

[139] R. K. Sarvadevabhatla, I. Dwivedi, A. Biswas, S. Manocha *et al.*, "Sketchparse: Towards rich descriptions for poorly drawn sketches using multi-task hierarchical deep networks," in *MM*, 2017.

[140] J. Jiang, R. Wang, S. Lin, and F. Wang, "Sfsegnet: Parse freehand sketches using deep fully convolutional networks," in *IJCNN*, 2019.

[141] K. Mukherjee, R. X. Hawkins, and J. E. Fan, "Communicating semantic part information in drawings," in *Annual Conference of the Cognitive Science Society*, 2019.

[142] Y. Zheng, H. Yao, and X. Sun, "Deep semantic parsing of freehand sketches with homogeneous transformation, soft-weighted loss, and staged learning," *arXiv preprint arXiv:1910.06023*, 2019.

[143] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, 2015.

[144] Y. Chien, W.-C. Lin, T.-S. Huang, and J.-H. Chuang, "Line drawing simplification by stroke translation and combination," in *ICGIP*, 2014.

[145] T. Ogawa, Y. Matsui, T. Yamasaki, and K. Aizawa, "Sketch simplification by classifying strokes," in *ICPR*, 2016.

[146] P. Barla, J. Thollot, and F. X. Sillion, "Geometric clustering for line drawing simplification," in *TOG*, 2005.

[147] R. K. Sarvadevabhatla *et al.*, "Eye of the dragon: Exploring discriminatively minimalist sketch-based abstractions for object categories," in *MM*, 2015.

[148] U. R. Muhammad, Y. Yang, T. M. Hospedales, T. Xiang, and Y.-Z. Song, "Goal-driven sequential data abstraction," in *ICCV*, 2019.

[149] Y.-P. Tan, S. R. Kulkarni, and P. J. Ramadge, "A framework for measuring video similarity and its application to video query by example," in *ICIP*, 1999.

[150] Y. Matsui, "Challenge for manga processing: Sketch-based manga retrieval," in *MM*, 2015.

[151] P. Xu, K. Li, Z. Ma, Y.-Z. Song, L. Wang, and J. Guo, "Cross-modal subspace learning for sketch-based image retrieval: A comparative study," in *IC-NIDC*, 2016.

[152] P. Xu, Q. Yin, Y. Huang, Y.-Z. Song, Z. Ma, L. Wang, T. Xiang, W. B. Kleijn, and J. Guo, "Cross-modal subspace learning for fine-grained sketch-based image retrieval," *Neurocomputing*, 2018.

[153] T. Bui, L. Ribeiro, M. Ponti, and J. Collomosse, "Deep manifold alignment for mid-grain sketch based image retrieval," in *ACCV*, 2018.

[154] L. Zheng, Y. Yang, and Q. Tian, "Sift meets cnn: A decade survey of instance retrieval," *TPAMI*, 2017.

[155] S.-i. Kondo, M. Toyoura, and X. Mao, "Sketch based skirt image retrieval," in *Proceedings of the 4th Joint Symposium on Computational Aesthetics, Non-Photorealistic Animation and Rendering, and Sketch-Based Interfaces and Modeling*, 2014.

[156] S. D. Bhattacharjee, J. Yuan, W. Hong, and X. Ruan, "Query adaptive instance search using object sketches," in *MM*, 2016.

[157] J. Lei, K. Zheng, H. Zhang, X. Cao, N. Ling, and Y. Hou, "Sketch based image retrieval via image-aided cross domain learning," in *ICIP*, 2017.

[158] S. D. Bhattacharjee, J. Yuan, Y. Huang, J. Meng, and L. Duan, "Query adaptive multiview object instance search and localization using sketches," *TMM*, 2018.

[159] J. Canny, "A computational approach to edge detection," *TPAMI*, 1986.

[160] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *ECCV*, 2014.

[161] S. Xie and Z. Tu, "Holistically-nested edge detection," in *ICCV*, 2015.

[162] S. Chopra, R. Hadsell, Y. LeCun *et al.*, "Learning a similarity metric discriminatively, with application to face verification," in *CVPR*, 2005.

[163] P. Xu, Q. Yin, Y. Qi, Y.-Z. Song, Z. Ma, L. Wang, and J. Guo, "Instance-level coupled subspace learning for fine-grained sketch-based image retrieval," in *ECCV Workshops*, 2016.

[164] Y. Qi, Y.-Z. Song, H. Zhang, and J. Liu, "Sketch-based image retrieval via siamese convolutional neural network," in *ICIP*, 2016.

[165] J. Song, Y.-Z. Song, T. Xiang, T. M. Hospedales, and X. Ruan, "Deep multi-task attribute-driven ranking for fine-grained sketch-based image retrieval." in *BMVC*, 2016.

[166] Q. Yu, X. Chang, Y.-Z. Song, T. Xiang, and T. M. Hospedales, "The devil is in the middle: Exploiting mid-level representations for cross-domain instance matching," *arXiv preprint arXiv:1711.08106*, 2017.

[167] Y. Yan, X. Wang, X. Yang, X. Bai, and W. Liu, "Joint classification loss and histogram loss for sketch-based image retrieval," in *ICIG*, 2017.

[168] J. Collomosse, T. Bui, M. J. Wilber, C. Fang, and H. Jin, "Sketching with style: Visual search with sketches and aesthetic context," in *ICCV*, 2017.

[169] J. Song, Y.-Z. Song, T. Xiang, and T. M. Hospedales, "Fine-grained image retrieval: the text/sketch input dilemma," in *BMVC*, 2017.

[170] F. Huang, Y. Cheng, C. Jin, Y. Zhang, and T. Zhang, "Deep multimodal embedding model for fine-grained sketch-based image retrieval," in *SIGIR*, 2017.

[171] S. Dey, A. Dutta, S. K. Ghosh, E. Valveny, J. Lladós, and U. Pal, "Learning cross-modal deep embeddings for multi-object image retrieval using text and sketch," in *ICPR*, 2018.

[172] Y. Wang, F. Huang, Y. Zhang, R. Feng, T. Zhang, and W. Fan, "Deep cascaded cross-modal correlation learning for fine-grained sketch-based image retrieval," *PR*, 2019.

[173] K. Pang, K. Li, Y. Yang, H. Zhang, T. M. Hospedales, T. Xiang, and Y.-Z. Song, "Generalising fine-grained sketch-based image retrieval," in *CVPR*, 2019.

[174] A. K. Bhunia, Y. Yang, T. M. Hospedales, T. Xiang, and Y.-Z. Song, "Sketch less for more: On-the-fly fine-grained sketch based image retrieval," in *CVPR*, 2020.

[175] G. Andrew, R. Arora, J. Bilmes, and K. Livescu, "Deep canonical correlation analysis," in *ICML*, 2013.

[176] F. Huang, C. Jin, Y. Zhang, and T. Zhang, "Towards sketch-based image retrieval with deep cross-modal correlation learning," in *ICME*, 2017.

[177] D. Xu, X. Alameda-Pineda, J. Song, E. Ricci, and N. Sebe, "Cross-paced representation learning with partial curricula for sketch-based image retrieval," *TIP*, 2018.

[178] C. Li, Y. Zhou, and J. Yang, "Sketch-based image retrieval via a semi-heterogeneous cross-domain network," in *ICME Workshops*, 2019.

[179] J. Collomosse, T. Bui, and H. Jin, "Livesketch: Query perturbations for guided sketch-based visual search," in *CVPR*, 2019.

[180] O. Seddati, S. Dupont, and S. Mahmoudi, "Quadruplet networks for sketch-based image retrieval," in *ICMR*, 2017.

[181] T. Bui, L. Ribeiro, M. Ponti, and J. Collomosse, "Generalisation and sharing in triplet convnets for sketch based visual search," *arXiv preprint arXiv:1611.05301*, 2016.

[182] H. Lin, Y. Fu, P. Lu, S. Gong, X. Xue, and Y.-G. Jiang, "Tc-net for isbir: Triplet classification network for instance-level sketch based image retrieval," in *MM*, 2019.

[183] L. Guo, J. Liu, Y. Wang, Z. Luo, W. Wen, and H. Lu, "Sketch-based image retrieval using generative adversarial networks," in *MM*, 2017.

[184] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[185] K. Pang, Y.-Z. Song, T. Xiang, and T. M. Hospedales, "Cross-domain generative learning for fine-grained sketch-based image retrieval." in *BMVC*, 2017.

[186] F. Huang, C. Jin, Y. Zhang, K. Weng, T. Zhang, and W. Fan, "Sketch-based image retrieval with deep visual semantic descriptor," *PR*, 2018.

[187] T. Bui, L. Ribeiro, M. Ponti, and J. Collomosse, "Compact descriptors for sketch-based image retrieval using a triplet loss convolutional neural network," *CVIU*, 2017.

[188] J. Lei, Y. Song, B. Peng, Z. Ma, L. Shao, and Y.-Z. Song, "Semi-heterogeneous three-way joint embedding network for sketch-based image retrieval," *TCSVT*, 2019.

[189] H. Zhang, C. Zhang, and M. Wu, "Sketch-based cross-domain image retrieval via heterogeneous network," in *VCIP*, 2017.

[190] L. Liu, F. Shen, Y. Shen, X. Liu, and L. Shao, "Deep sketch hashing: Fast free-hand sketch-based image retrieval," in *CVPR*, 2017.

[191] J. Zhang, F. Shen, L. Liu, F. Zhu, M. Yu, L. Shao, H. Tao Shen, and L. Van Gool, "Generative domain-migration hashing for sketch-to-image retrieval," in *ECCV*, 2018.

[192] G. Tolias and O. Chum, "Asymmetric feature maps with application to sketch based retrieval," in *CVPR*, 2017.

[193] K. Lin, H.-F. Yang, J.-H. Hsiao, and C.-S. Chen, "Deep learning of binary hash codes for fast image retrieval," in *CVPR Workshops*, 2015.

[194] J. Wang, T. Zhang, N. Sebe, H. T. Shen *et al.*, "A survey on learning to hash," *TPAMI*, 2017.

[195] K. Pang, Y. Yang, T. Hospedales, T. Xiang, and Y.-Z. Song, "Solving mixed-modal jigsaw puzzle for fine-grained sketch-based image retrieval," in *CVPR*, 2020.

[196] F. Radenovic, G. Tolias, and O. Chum, "Deep shape matching," in *ECCV*, 2018.

[197] A. Sablayrolles, M. Douze, N. Usunier, and H. Jégou, "How should we evaluate supervised hashing?" in *ICASSP*, 2017.

[198] P. Lu, G. Huang, Y. Fu, G. Guo, and H. Lin, "Learning large euclidean margin for sketch-based image retrieval," *arXiv preprint arXiv:1812.04275*, 2018.

[199] W. Thong, P. Mettes, and C. G. Snoek, "Open cross-domain visual search," *arXiv preprint arXiv:1911.08621*, 2019.

[200] T. Dutta and S. Biswas, "Style-guided zero-shot sketch-based image retrieval," in *BMVC*, 2019.

[201] A. Dutta and Z. Akata, "Semantically tied paired cycle consistency for zero-shot sketch-based image retrieval," in *CVPR*, 2019.

[202] S. Kiran Yelamarthi, S. Krishna Reddy, A. Mishra, and A. Mittal, "A zero-shot framework for sketch based image retrieval," in *ECCV*, 2018.

[203] J. Li, Z. Ling, L. Niu, and L. Zhang, "Bi-directional domain translation for zero-shot sketch-based image retrieval," *arXiv preprint arXiv:1911.13251*, 2019.

[204] A. Pandey, A. Mishra, V. Kumar Verma, and A. Mittal, "Adversarial joint-distribution learning for novel class sketch-based image retrieval," in *ICCV Workshops*, 2019.

[205] V. Kumar Verma, A. Mishra, A. Mishra, and P. Rai, "Generative model for zero-shot sketch-based image retrieval," in *CVPR Workshops*, 2019.

[206] Q. Liu, L. Xie, H. Wang, and A. L. Yuille, "Semantic-aware knowledge preservation for zero-shot sketch-based image retrieval," in *ICCV*, 2019.

[207] A. Pandey, A. Mishra, V. K. Verma, A. Mittal, and H. Murthy, "Stacked adversarial network for zero-shot sketch based image retrieval," in *WACV*, 2020.

[208] X. Xu, C. Deng, M. Yang, and H. Wang, "Progressive domain-independent feature decomposition network for zero-shot sketch-based image retrieval," *arXiv preprint arXiv:2003.09869*, 2020.

[209] Z. Zhang, Y. Zhang, R. Feng, T. Zhang, and W. Fan, "Zero-shot sketch-based image retrieval via graph convolution network," in *AAAI*, 2020.

[210] Y. Fu, T. M. Hospedales, T. Xiang, and S. Gong, "Transductive multi-view zero-shot learning," *TPAMI*, 2015.

[211] Y. Fu, T. Xiang, Y.-G. Jiang, X. Xue, L. Sigal, and S. Gong, "Recent advances in zero-shot recognition: Toward data-efficient understanding of visual content," *IEEE Signal Processing Magazine*, 2018.

[212] W. Wang, V. W. Zheng, H. Yu, and C. Miao, "A survey of zero-shot learning: Settings, methods, and applications," *TIST*, 2019.

[213] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NeurIPS*, 2013.

[214] C. H. Lampert, H. Nickisch, and S. Harmeling, "Attribute-based classification for zero-shot visual object categorization," *TPAMI*, 2013.

[215] Y. Shen, L. Liu, F. Shen, and L. Shao, "Zero-shot sketch-image hashing," in *CVPR*, 2018.

[216] C. Gao, Q. Liu, Q. Xu, J. Liu, L. Wang, and C. Zou, "Image generation from freehand scene sketches," *arXiv preprint arXiv:2003.02683*, 2020.

[217] M. Li, Z. Lin, R. Mech, E. Yumer, and D. Ramanan, "Photo-sketching: Inferring contour drawings from images," in *WACV*, 2019.

[218] Y. Zhang, G. Su, Y. Qi, and J. Yang, "Unpaired image-to-sketch translation network for sketch synthesis," in *VCIP*, 2019.

[219] M. Kampelmuhler and A. Pinz, "Synthesizing human-like sketches from natural images using a conditional convolutional decoder," in *WACV*, 2020.

[220] Y. Jo and J. Park, "Sc-fegan: Face editing generative adversarial network with user's sketch and color," in *ICCV*, 2019.

[221] S. Yang, Z. Wang, J. Liu, and Z. Guo, "Deep plastic surgery: Robust and controllable image editing with human-drawn sketches," *arXiv preprint arXiv:2001.02890*, 2020.

[222] J. Li, S. Liu, and M. Cao, "Line artist: A multiple style sketch to painting synthesis scheme," *arXiv preprint arXiv:1803.06647*, 2018.

[223] M. Li, A. Sheffer, E. Grinspun, and N. Vining, "Foldsketch: Enriching garments with physically reproducible folds," *TOG*, 2018.

[224] T. Y. Wang, D. Ceylan, J. Popovic, and N. J. Mitra, "Learning a shared shape space for multimodal garment design," *arXiv preprint arXiv:1806.11335*, 2018.

[225] A. Ghosh, R. Zhang, P. K. Dokania, O. Wang, A. A. Efros, P. H. S. Torr, and E. Shechtman, "Interactive sketch & fill: Multiclass sketch-to-image translation," in *ICCV*, 2019.

[226] J. Collomosse, T. Bui, M. J. Wilber, C. Fang, and H. Jin, "Sketching with style: Visual search with sketches and aesthetic context," in *ICCV*, 2017.

[227] B. Liu, K. Song, and A. Elgammal, "Sketch-to-art: Synthesizing stylized art images from sketches," *arXiv preprint arXiv:2002.12888*, 2020.

[228] C. Zou, H. Mo, R. Du, X. Wu, C. Gao, and H. Fu, "Lucss: Language-based user-customized colourization of scene sketches," *arXiv preprint arXiv:1808.10544*, 2018.

[229] L. Zhang, C. Li, T.-T. Wong, Y. Ji, and C. Liu, "Two-stage sketch colorization," *TOG*, 2018.

[230] C. Zou, H. Mo, C. Gao, R. Du, and H. Fu, "Language-based colorization of scene sketches," *TOG*, 2019.

[231] X. Soria, E. Riba, and A. D. Sappa, "Dense extreme inception network: Towards a robust cnn model for edge detection," *arXiv preprint arXiv:1909.01955*, 2019.

[232] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *ICCV*, 2017.

[233] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *CVPR*, 2017.

[234] W. Chen and J. Hays, "Sketchygan: Towards diverse and realistic sketch to image synthesis," in *CVPR*, 2018.

[235] W. Xia, Y. Yang, and J.-H. Xue, "Cali-sketch: Stroke calibration and completion for high-quality face image generation from poorly-drawn sketches," *arXiv preprint arXiv:1911.00426*, 2019.

[236] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[237] W. Xian, P. Sangkloy, V. Agrawal, A. Raj, J. Lu, C. Fang, F. Yu, and J. Hays, "Texturegan: Controlling deep image synthesis with texture patches," in *CVPR*, 2018.

[238] Y. Lu, S. Wu, Y.-W. Tai, and C.-K. Tang, "Image generation from sketch constraint using contextual gan," in *ECCV*, 2018.

[239] T. Shao, W. Xu, K. Yin, J. Wang, K. Zhou, and B. Guo, "Discriminative sketch-based 3d model retrieval via robust shape matching," in *Computer Graphics Forum*, 2011.

[240] B. Li, Y. Lu, and J. Shen, "A semantic tree-based approach for sketch-based 3d model retrieval," in *ICPR*, 2016.

[241] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, and A. Santella, "Suggestive contours for conveying shape," *TOG*, 2003.

[242] H. Li, H. Wu, X. He, S. Lin, R. Wang, and X. Luo, "Multi-view pairwise relationship learning for sketch based 3d shape retrieval," in *ICME*, 2017.

[243] F. Zhu, J. Xie, and Y. Fang, "Learning cross-domain neural networks for sketch-based 3d shape retrieval," in *AAAI*, 2016.

[244] Y. Ye, B. Li, and Y. Lu, "3d sketch-based 3d model retrieval with convolutional neural network," in *ICPR*, 2016.

[245] J. Chen and Y. Fang, "Deep cross-modality adaptation via semantics preserving adversarial learning for sketch-based 3d shape retrieval," in *ECCV*, 2018.

[246] J. Chen, J. Qin, L. Liu, F. Zhu, F. Shen, J. Xie, and L. Shao, "Deep sketch-shape hashing with segmented 3d stochastic viewing," in *CVPR*, 2019.

[247] A. Qi, Y.-Z. Song, and T. Xiang, "Semantic embedding for sketch-based 3d shape retrieval." in *BMVC*, 2018.

[248] S. Kuwabara, R. Ohbuchi, and T. Furuya, "Query by partially-drawn sketches for 3d shape retrieval," in *2019 International Conference on Cyberworlds*, 2019.

[249] G. Dai, J. Xie, F. Zhu, and Y. Fang, "Deep correlated metric learning for sketch-based 3d shape retrieval," in *AAAI*, 2017.

[250] G. Dai, J. Xie, and Y. Fang, "Deep correlated holistic metric learning for sketch-based 3d shape retrieval," *TIP*, 2018.

[251] J. Xie, G. Dai, F. Zhu, and Y. Fang, "Learning barycentric representations of 3d shapes for sketch-based 3d shape retrieval," in *CVPR*, 2017.

[252] V. I. Bogachev and A. V. Kolesnikov, "The monge-kantorovich problem: achievements, connections, and perspectives," *Russian Mathematical Surveys*, 2012.

[253] L. Wang, C. Qian, J. Wang, and Y. Fang, "Unsupervised learning of 3d model reconstruction from hand-drawn sketches," in *MM*, 2018.

[254] Y. Shen, C. Zhang, H. Fu, K. Zhou, and Y. Zheng, "Deepsketchhair: Deep sketch-based 3d hair modeling," *arXiv preprint arXiv:1908.07198*, 2019.

[255] H. Huang, E. Kalogerakis, E. Yumer, and R. Mech, "Shape synthesis from sketches via procedural models and convolutional networks," *TVCG*, 2016.

[256] X. Han, C. Gao, and Y. Yu, "Deepsketch2face: a deep learning based sketching system for 3d face and caricature modeling," *TOG*, 2017.

[257] M. Ye, S. Zhou, and H. Fu, "Deepshapesketch: Generating hand drawing sketches from 3d objects," in *IJCNN*, 2019.

[258] J. P. Collomosse, G. McNeill, and Y. Qian, "Storyboard sketches for content based video retrieval," in *ICCV*, 2009.
[259] P. Xu, K. Liu, T. Xiang, T. M. Hospedales, Z. Ma, J. Guo, and Y.-Z. Song, "Fine-grained instance-level sketch-based video retrieval," *arXiv preprint arXiv:2002.09461*, 2020.
[260] S. Wu, H. Su, S. Zheng, H. Yang, and Q. Zhou, "Motion sketch based crowd video retrieval via motion structure coding," in *ICIP*, 2016.
[261] S. Wu, H. Yang, S. Zheng, H. Su, Q. Zhou, and X. Lu, "Motion sketch based crowd video retrieval," *Multimedia Tools and Applications*, 2017.
[262] F. Huang and J. F. Canny, "Sketchforme: Composing sketched scenes from text descriptions for interactive applications," *arXiv preprint arXiv:1904.04399*, 2019.
[263] F. Huang, E. Schoop, D. Ha, and J. Canny, "Scones: towards conversational authoring of sketches," in *International Conference on Intelligent User Interfaces*, 2020.
[264] R. K. Sarvadevabhatla, S. Surya, T. Mittal, and R. V. Babu, "Game of sketches: Deep recurrent models of pictionary-style word guessing," in *AAAI*, 2018.
[265] K. Pang, D. Li, J. Song, Y.-Z. Song, T. Xiang, and T. M. Hospedales, "Deep factorised inverse-sketching," in *ECCV*, 2018.
[266] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[267] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *CVPR*, 2016.
[268] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *CVPR*, 2017.
[269] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *CVPR*, 2018.
[270] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR*, 2017.
[271] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," in *ICLR*, 2018.
[272] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017.
[273] J. C. Roberts, C. Headleand, and P. D. Ritsos, "Sketching designs using the five design-sheet methodology," *TVCG*, 2015.
[274] F. Boniardi, A. Valada, W. Burgard, and G. D. Tipaldi, "Autonomous indoor robot navigation using a sketch interface for drawing maps and routes," in *ICRA*, 2016.
[275] V. Jain, P. Agrawal, S. Banga, R. Kapoor, and S. Gulyani, "Sketch2code: Transformation of sketches to ui in real-time using deep neural network," *arXiv preprint arXiv:1910.08930*, 2019.
[276] A. Kotani and S. Tellex, "Teaching robots to draw," in *ICRA*, 2019.