SketchMate: Deep Hashing for Million-Scale Human Sketch Retrieval

Anonymous CVPR submission

Paper ID 2763

Abstract

We propose a deep hashing framework for sketch retrieval that, for the first time, works on a multi-million scale human sketch dataset. Leveraging on this large dataset, we explore a few sketch-specific traits that were otherwise under-studied in prior literature. Instead of following the conventional sketch recognition task, we introduce the novel problem of sketch hashing retrieval which is not only more challenging, but also offers a better testbed for large-scale sketch analysis, since: (i) more fine-grained sketch feature learning is required to accommodate the large variations in style and abstraction, and (ii) a compact binary code needs to be learned at the same time to enable efficient retrieval. Key to our network design is the embedding of unique characteristics of human sketch, where (i) a two-branch CNN-RNN architecture is adapted to explore the temporal ordering of strokes, and (ii) a novel hashing loss is specifically designed to accommodate both the temporal and abstract traits of sketches. By working with a 3.8M sketch dataset, we show that state-of-the-art hashing models specifically engineered for static images fail to perform well on temporal sketch data. Our network on the other hand not only offers the best retrieval performance on various code sizes, but also yields the best sketch recognition performance when re-purposed for classification. Such superior retrieval and classification performances effectively demonstrate the benefit of our sketch-specific design.

1. Introduction

Sketches are different to photos. They exhibit a high-level of abstraction yet are surprisingly illustrative. With just a few strokes, they are able to encode an appropriate level of semanticness that depicts objects and communicate stories (*e.g*. ancient cave drawings). Such unique characteristics of sketches, together with the prevalence of touch-screen devices, to a large extent drove the recent surge of sketch research. Problems studied so far range from sketch recognition [3, 11, 30], sketch-based image retrieval (SBIR) [29, 20], to sketch synthesis [12].

Despite great strides made, a major obstacle facing all sketch research is the lack of freely available sketch data. Compared with photos where million-scale datasets had been readily accessible for almost a decade (*e.g*. ImageNet [2]), all aforementioned research worked with sub-million level crowd-sourced sketch datasets (20k for TU-Berlin [3] and 75k for Sketchy [20]). These datasets served as key enablers for the community, though have very recently started to bottleneck the progress of sketch research – sketch recognition performance had already gone far beyond human-level [30] on TU-Berlin [3], and steadily approaching human performance [18] for the problem of SBIR on Sketchy [20].

In particular, two unique traits of human sketches had been mostly overlooked: (i) sketches are highly abstract and iconic, whereas photos are pixel perfect depictions, (ii) sketching is a dynamic process other than a mere collection of static pixels. Such oversights can be partially attributed to the lack of a large and diverse dataset of stroke-level human sketches, since more data samples are required to broadly capture (i) the substantial variances on visual abstraction, and (ii) the highly complex temporal stroke configurations – an apple might look like an apple once drawn (though more abstract than photos), there is more than one way of drawing it. The seminal work of [30] on sketch recognition tackled these problems to some extent yet were limited in that (i) sketches are treated as static pixelmaps, where deep architecture for feature learning is limited to variants of photo CNNs, and (ii) temporal ordering information are modeled coarsely by temporally segmenting one sketch into three separate pixelmaps, which are then encoded using a multi-branch CNN. The very recent work of [5] was the first to fully acknowledge the temporal nature of sketches, and proposed a RNN-based generative model to synthesize novel sketches from scratch. In this paper, we combine RNN stroke modeling with conventional CNN under a dual-branch setting to learn better sketch feature representations. However, the problem of visual abstraction, especially how it can be accommodated under a large-scale retrieval setting remains unsolved.

In this paper, for the first time, we leverage on a newly

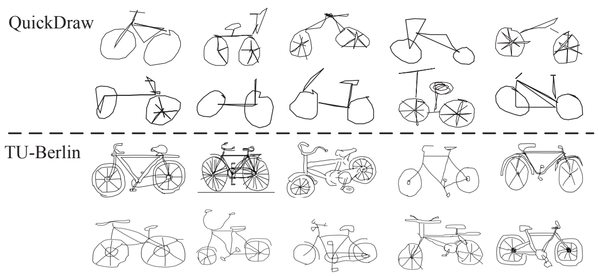


Figure 1. Sample sketches from QuickDraw-3.8M and TU-Berlin.

released multi-million human sketch dataset [5], and introduce the novel problem of sketch hashing retrieval (SHR). Different to the conventional task of sketch recognition where classification is usually performed by computing feature distances in Euclidean space [30], given a query sketch, SHR aims to compute an exhaustive ranking of all sketches in a very large test gallery. It is thus a more difficult problem than sketch recognition, since (i) more discriminative feature representations are needed to accommodate the much larger variations on style and abstraction , and meanwhile (ii) a compact binary code needs to be learned to facilitate efficient large-scale retrieval. Importantly, the availability of such a large dataset enabled us to better explore the aforementioned sketch-specific traits of being highly abstract and sequential in nature. In particular, we fully examine the temporal ordering of strokes through a two-branch CNN-RNN network, and address the abstraction problem by proposing a novel hashing loss that enforces more compact feature clusters for each sketch category in Hamming space.

More specifically, we first construct a dataset of 3,829,500 human sketches, by randomly sampling from every category of the Google QuickDraw dataset [5], which we term as “QuickDraw-3.8M”. This dataset is highly noisy when compared with TU-Berlin, for that (i) users had only 20 seconds to draw, and (ii) no specific post-processing was performed. Figure 1 offers a visual comparison between the two datasets. We then analyze the intrinsic data traits of sketch and design a novel end-to-end deep hashing model to conduct fast retrieval. The main contributions of this paper can be summarized as:

- For the first time, we introduce the problem of sketch hashing retrieval on a multi-million scale human sketch dataset, and propose a deep hashing network that directly accommodates the key characteristics of human sketch. We show that our network is able to outperform state-of-the-art alternatives specifically designed for photo-photo and sketch-photo retrieval, highlighting the advantage of our sketch-specific design. Moreover, our network also achieves state-of-the-art performance when re-purposed for the task of sketch recognition.
- We propose a novel multi-branch CNN-RNN architecture that specifically encode the temporal ordering information of sketches to learn a more fine-grained feature representation. We find that stroke-level temporal information is indeed helpful in sketch feature learning in that it alone can outperform CNN features for the sketch recognition task, and offers the best performance when combined with CNN features.
- We design a novel hashing loss to accommodate the abstract nature of sketches, especially on such a large dataset where noise is also present. More specifically, we propose a sketch center loss to learn more compact feature clusters for each object category and in turn improve retrieval performance.

The rest of the paper is organized as follows: Section 2 briefly summarizes related work. Section 3 describes our proposed deep hashing model for large-scale sketch retrieval. Experimental results and discussion are presented in Section 4. Finally, we draw some conclusions in Section 5.

2. Related Work

Sketch Dataset Collecting sketches is hard, and even harder when the sketch is asked to draw based on the mental imaginary other than an abstract concept. This constitutes the main reason stalling the systematic and scalable research on sketches. Until the proliferation of touchscreen devices, few middle-scale sketch datasets [3, 29, 23, 20] have been collected. This is possible by resorting to crowdsourcing platform (*e.g.* Amazon Mechanical Turk) to ask the participant either draw by hand or slide with a mouse. However still, these dataset size normally ranges from hundreds to thousands, orders of magnitude smaller than other traditional meta vision datasets, *i.e.* ImageNet [2], thus infeasible for large-scale deep hashing exploration that are inherently data-hungry. Very recently, this problem has been greatly alleviated by Ha and Eck [5], which contributed a dataset containing 50 millions of sketches crossing 345 categories. However, these sketches collected by the worldwide participants without any manual supervision contain considerable amount of noisy samples, requiring special care to take with.

Sketch Recognition A few shallow hand-crafted feature learning methods [3, 11] have been proposed for sketch recognition task, where they used support vector machine (SVM) as the classifier and differed only in what hand-crafted features borrowed from photos are used as representation. In particular, Li *et al.* [11] demonstrated that histogram of oriented gradients (HOG) generally outperformed other local features, while by fusing them together

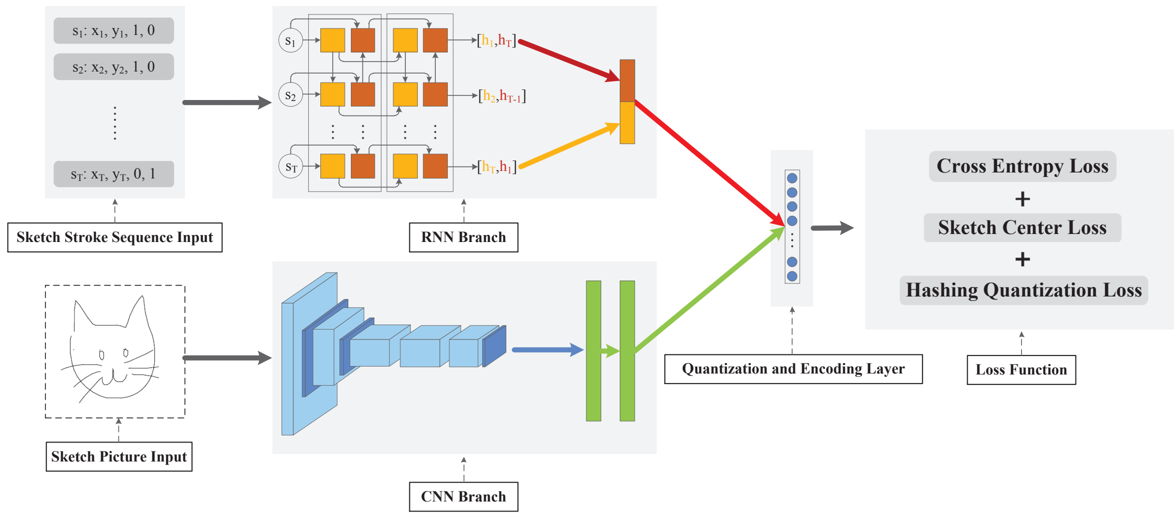


Figure 2. An illustration of our two-branch CNN-RNN deep sketch hashing retrieval network. Best viewed in color.

under multiple kernel learning further improved the performance. The ground-breaking work of Yu *et al.* [31], for the first time beat human performance on sketch recognition task by utilizing the discriminative power of a deep convolutional neural network, where subsequent work exploited stroke-level temporal information either from heuristic data augmentation perspective [30] or a learned model level [21]. In this paper, we advance the sketch recognition problem one step forward – given a sketch query, efficiently and accurately retrieve semantically-similar sketches from million-scale gallery within a compact deep hashing space, termed sketch hashing retrieval.

Deep Hashing Learning Hashing learning is an important research topic for fast image retrieval, where conventional hashing methods (Locality-Sensitive Hashing (LSH) [1], Spectral Hashing (SH) [26], Iterative Quantization (ITQ) [4]) mainly involve learning projections and quantization strategies, which usually take hand-crafted features as image representation. With deep learning showing remarkable effects [10, 24, 6, 7] and sweeping across computer vision tasks, deep hashing learning also found its way to this garden of bliss [28, 14, 22], showing additional superiority of better preserving the semantic information compared with shallow methods. In the initial setting, feature representation and hashing coding were learned in separate stages [28], where subsequent work [14, 32, 15] found more elegant performance through joint end-to-end training.

To our best knowledge, only one previous work [16] has specifically designed deep hashing framework targeting on sketch domain, where they introduced a semi-heterogeneous deep architecture by incorporating the cross-view similarity and cross-category semantic loss and presented impressive results over several baselines. However, their limitations mainly lay in that (i) the temporal information coming inherently from sketch drawings is neglected and (ii) the dataset [20] they are evaluating on is small, leaving whether there are additional challenges on million-scale [5] dataset unknown, where in this paper we actively address the two issues.

3. Methodology

3.1. Problem Formulation

Let $\mathcal{K} = \{K_n = (\mathbf{P}_n, \mathbf{S}_n)\}_{n=1}^N$ be N sketch sample pairs cross L possible categories and $\mathcal{Y} = \{y_n\}_{n=1}^N$ be their respective category labels. Each sketch sample K_n consists of a sketch $\mathbf{P}_n$ in raster pixel space and a corresponding sketch stroke sequence $\mathbf{S}_n$. We aim to learn a mapping $\mathcal{M} : \mathcal{K} \rightarrow \{0,1\}^{H \times N}$, which maps sketches into a low dimensional (H dimension) space $\mathbf{f}_n$ that further translated into H-bit binary codes $\mathbf{B} = \{\mathbf{b}_n\}_{n=1}^N \in \{0,1\}^{H \times N}$, while maintaining relevancy in accordance with the semantic and visual similarity amongst sketch data.

3.2. Two-branch CNN-RNN Network

Overview As previously stated, learning discriminative sketch features is a very challenging task due to the high degree of variations in style and abstraction. This problem is made worse under a large-scale retrieval setting since better feature representations are needed for more fine-grained feature comparison. Despite shown to be successful on a much smaller sketch dataset [30], CNN-based network completely abandons the natural stroke-level temporal information of human sketches, which can now be modeled by a RNN network, thanks to the ground-breaking work by

[5]. In this work, we, for the first time, propose to combine the best from the both world for human sketches – utilizing CNN to extract abstract high-level concepts and RNN to model human sketching temporal orders. With additional discriminative power (temporal cue) injected in, we hope this can lead to better feature learning.

Two-branch Late-fusion As illustrated in Figure 2, our two-branch encoder consists of three sub-modules: (1) a CNN encoder takes in a raster pixel sketch and translates into a high-dimensional space; (2) a RNN encoder takes in a vector sketch and outputs its final time-step state; (3) branch interaction via a late-fusion layer by concatenation. This enables our learned feature to benefit from both vector and raster sketch.

Quantization Encoding layer After the final fusion layer, we have to encode that deep feature into the low-dimensional real-valued hashing feature $\mathbf{f}_n$ (one fully connected layer with sigmoid activation), which will be further transformed to the hashing code, $\mathbf{b}_n$. The transformation function goes as follows:

$$\mathbf{b}_n = sgn(\mathbf{f}_n - \mathbf{0.5}), \;\; n \in (1, N). \tag{1}$$

Learning Objective To obtain the hashing feature $\mathbf{f}_n$ and hashing code $\mathbf{b}_n$, we could train the network end-to-end using two common losses similar to those found in image hashing networks [14]. The first comes with the cross entropy loss (CEL) for K_n calculated on L-way softmax:

$$\mathcal{L}_{cel} = \frac{1}{N}\sum_{n=1}^{N} -\log \frac{\mathrm{e}^{\mathbf{W}_{y_n}^T \mathbf{f}_n + \mathbf{b}_{y_n}}}{\sum_{j=1}^{L} \mathrm{e}^{\mathbf{W}_j^T \mathbf{f}_n + \mathbf{b}_j}}, \tag{2}$$

where $\mathbf{W}_j \in \mathbb{R}^H$ is the jth column of the weights $\mathbf{W} \in \mathbb{R}^{H \times L}$ between the quantization-encoding layer and L-way softmax outputs. The second loss is the quantization loss (QL) that is used to reduce the error caused by quantization-encoding:

$$\mathcal{L}_{ql} = \|\mathbf{b}_n - \mathbf{f}_n\|_2^2, \; s.t. \; \mathbf{b}_n \in \{0, 1\}^H, \tag{3}$$

3.3. Sketch Center Loss

In theory, these two losses should perform reasonably well on discriminating category-level semantics, however, our large-scale sketch dataset presents an unique challenges – sketch are highly abstract, often making semantically different categories to exhibit similar appearance (see Figure 3(a) for an example of 'dog' vs. 'pig'). We need to make sure such abstract nature of sketches do not hinder overall retrieval performance.

The common center loss (CL) was proposed in [27] to tackle such a problem by introducing the concept of class centers, $\mathbf{c}_{y_n}$, to characterize the intra-class variations. Class centers should be updated as deep features change, in other words, the entire training set should be taken into account and features of every class should be averaged in each iteration. This is clearly unrealistic and normally compromised by updating only within each mini-batch. This problem is even more salient under our sketch hashing retrieval setting – (1) for million-scale hashing, updating common center within each mini-batch can be highly inaccurate and even misleading (as shown in later experiments), and this problem is worsened by the abstract nature of sketches in that only seeing sketches within one training batch doesn't necessarily provide useful and representative gradients for class centers; (2) despite of more compact internal category structures (Figure 3(b)) with common center loss, there is no explicit constraint to set apart between each, as a direct comparison with Figure 3(c).

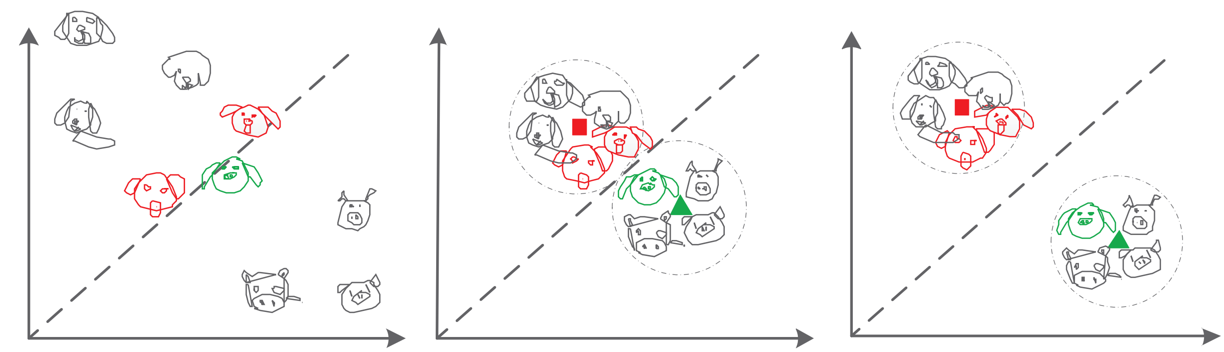
These issues call for a sketch-specific center loss that is able to deal with million-scale hashing retrieval. For sketch hashing, we need compact and discriminative features to aggregate samples belonging to the same category and segregate the visually confusing categories. Thus, an natural intuition would be: is it possible if we can find a *fixed* but representative center feature for each class, so to avoid the computational complexity during training, and meanwhile enforcing semantics between sketch categories.

We propose *sketch center loss* that is specifically designed for million-scale sketch hashing retrieval. This is done by (i) first pretraining CNN-RNN separately for sketch recognition task and then fine-tuning with our full model, both with softmax cross entropy loss only; (ii) obtain class feature center $\mathbf{c}_{y_n}$ by calculating the mean of the hashing feature $\mathbf{f}_n$ for the *noise-removal* sketches (detailed later) of that class based on the pretrained model. By doing so, in the final fine-tuning stage, we train end-to-end with a fixed center for each class, thus providing meaningful gradients during each training iteration, and we empirically find a significant performance boost under this sketch-specific center loss. We hence define our sketch center loss as:

$$\mathcal{L}_{scl} = \frac{1}{N}\sum_{n=1}^{N} \|\mathbf{f}_n - \mathbf{c}_{y_n}\|_2^2 \;, \tag{4}$$

Noise Removal with Image Entropy Key ingredient to a successful sketch center loss is the guarantee of non-noisy data (outliers), as it will significantly affect the class feature centers. However, datasets collected with crowdsourcing without manual supervision are inevitable to noise. Here we propose a noisy data removal technique to greatly alleviate such issues by resorting to image entropy. We define image entropy for sketch data as:

$$\mathcal{H} = \sum_{i=0,255} -P_i \log P_i \;, \tag{5}$$



(a) cross entropy loss (b) cross entropy loss + common center loss (c) cross entropy loss + sketch center loss

Figure 3. Geometric interpretation of sketch feature layout obtained by different loss function. The dashed line denotes the softmax decision boundary. The dot dashed circles denote the respective clustering of two confusing classes (dog and pig). The red and green patterns denote the outliers of dog and pig, respectively. The red solid square and the green solid triangle denote the feature centers. Best viewed in color.

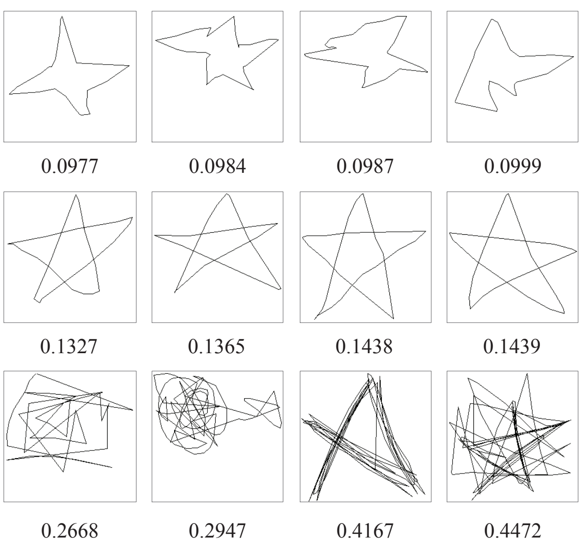


Figure 4. Some "star" samples randomly selected from our training set and their corresponding entropy values. The stars in first and third lines are outliers or noise points ($entropy \in (0, 0.1051) \cup (0.1721, 1)$). The stars in the second line are normal values ($entropy \in (0.1051, 0.1721)$).

where P_i is the proportion of the gray pixel values i in each sketch.

Now given a category of sketch, we can get entropy for each sketch and use Kernel Smoothing Density Estimation (KSDE) to calculate its probability density function (PDF). We empirically find keeping the middle 90% of the PDF gives us best results. In Figure 4, we visualize the randomly sampled star samples and its entropy values. As can be seen that low entropy sketches tend to be more abstract, yet high entropy ones tend to be more messy, while the middle range ones depict good looking stars.

Full Learning Objective By combining the above, our full objective becomes:

$$\mathcal{L}_{full} = \mathcal{L}_{cel} + \lambda_{ql}\mathcal{L}_{ql} + \lambda_{scl}\mathcal{L}_{scl}, \tag{6}$$

where $\lambda_{ql}, \lambda_{scl}$ control the relative importance of each loss.

4. Experiments

4.1. Datasets and Settings

Dataset Splits and Preprocessing Google QuickDraw dataset [5] contains 345 object categories with more than 100,000 free-hand sketches for each category. Despite the large-scale sketches publicly available, we empirically find out that a number of around 10,000 sketches suffices for a sufficient representation of each category and thus randomly choose 9000, 1000 from which for training and validation respectively. For evaluation, we form our query and retrieval gallery set by randomly choosing 100 and 1000 sketches from each category. A detailed illustration of the dataset split can be found at Table 1. Overall, this constitutes an experimental dataset of 3,829,500 sketches, standing itself on a million-scale analysis of sketch specific hashing problem, an order of magnitude larger than previous state-of-the-art research [16], which we term as "QuickDraw-3.8M". We scale the raster pixel sketch to $224 \times 224 \times 3$, with each brightness channel tiled equally, while processing the vector sketch same as with [5], with one critical exception – rather than treating pen state as a sequence of three binary switches, *i.e.* continue ongoing stroke, start a new stroke and stop sketching, we reduce to two states by eliminating the sketch termination signal for faster training, leading each time-step input to the RNN module a four-dimensional input (point coordinates + pen state).

Staged Pretraining We implement our model as staged training due to the inherently different internal structures and learning schemes between CNN-based and RNN-based

Splits	Number per cate	Amount
Training	9000	$9000 \times 345 = 3105000$
Validation	1000	$1000 \times 345 = 345000$
Retrieval	1000	$1000 \times 345 = 345000$
Query	100	$100 \times 345 = 34500$

Table 1. Dataset Splits on QuickDraw [5] for our experiments.

models and the necessary request for representative and robust class centers: (1) Separately pretrain the CNN, RNN branch on the QuickDraw-3.8M dataset with softmax cross entropy loss; (2) Fine-tune our full model end-to-end with softmax cross entropy loss; (3) Fine-tune our full model with softmax entropy loss, sketch center loss and quantization loss.

Implementation Details Our RNN-based encoder uses bidirectional Gated Recurrent Units with two layers, with a hidden size of 512 for each layer, and the CNN-based encoder follows the AlextNet [10] architecture with major difference at removing the local response normalization for faster training. We implement our model on one single Pascal TitanX GPU card, where for each pretraining stage, we train for $20, 5, 5$ epochs, taking about $20, 5, 10$ hours respectively. We set the importance weights $\lambda_{scl} = 0.01$ and $\lambda_{ql} = 0.0001$ during training and find this simple strategy works well. The model is trained end to end using the Adam optimizer [9]. The learning rate starts at 0.01 and decays exponentially every 10 epochs by one order of magnitude. We report the mean average precision (MAP) and precision at top-rank 200 (precision@200), same with previous deep hashing methods[14, 32, 15, 16] for a fair comparison.

4.2. Competitors

We compare our sketch hashing retrieval model with several state-of-the-art deep hashing approaches and for a fair comparison, we evaluate all competitors under same criteria.

DLBHC [14]: We compare with replacing our two-branch CNN-RNN module with a single-branch CNN module, where softmax cross entropy loss is used for joint feature and hashing code learning.

DSH-Supervised[15]: This also corresponds to a single-branch CNN model, but with noticeable difference in how to model the category-level discrimination, where pairwise contrastive loss is used based on the semantic pairing labels. We generate online image pairs within each training batch.

DSH-Sketch[16]: This is proposed to specifically target on modeling the sketch-photo cross-domain relations with a semi-heterogeneous network. To fit in our setting, we adopt the single-branch paradigm and their semantic factorization loss only, where word vector is assumed to represent the visual category. We keep other settings the same.

Moreover, we compare six unsupervised (Principal Component Analysis Iterative Quantization (**PCA-ITQ**) [4], Locality-Sensitive Hashing (**LSH**) [1], Spectral Hashing (**SH**) [26], Locality-Sensitive Hashing from Shift-Invariant Kernels (**SKLSH**) [19], Density Sensitive Hashing (**DSH**) [8], Principal Component Analysis Hashing (**PCAH**) [25]) and two supervised (Supervised Discrete Hashing (**SDH**) [22], Canonical Correlation Analysis Iterative Quantization (**CCA-ITQ**) [4]) shallow hashing methods, where deep features are fed into directly for learning. It's noteworthy that running each of the above eight tasks needs about $100 - 200$ GB memory. Limited by this, we train a smaller model and use $256d$ deep feature (extracted from the fusion layer) as inputs.

4.3. Results and Discussions

Comparison against Deep Hashing Competitors: We compare our full model and the three state-of-the-art deep hashing methods. Table 2 shows the results for sketch hashing retrieval under both metrics. We make the following observations: (i) Our model consistently outperforms previous state-of-the-art deep hashing methods by a significant margin, with 6.11/8.36 and 5.50/4.79 percent improvements (MAP/Precision@200) over the best performing competitor at 16-bit and 64-bit respectively. (ii) The gap between our model and DLBHC suggests the benefits of combining segment-level temporal information exhibited in a vector sketch with static pixel visual cues, the basis forming our CNN-RNN two-branch network, which may credit to (1) despite human tends to draw abstractly, they do share certain category-level coherent drawing styles, *i.e.* starting with a circle when sketching a sun, such that introducing additional discriminative power; (2) CNN suffers from sparse pixel image input [31] but prevails at building conceptual hierarchy [17], where RNN-based vector input brings the complements. (iii) DSH-Supervised is unsuitable for retrieval across a large number of categories due to the incident imbalanced input of positive and negative pairs [13]. This shows the importance of metric selection under universal (hundreds of categories) million-scale sketch hashing retrieval, where softmax cross entropy loss generally works better, while pairwise contrastive loss hardly constrain the feature representation space and word vector can be misleading, *i.e.* basketball and apple are similar in terms of shape abstraction, but pushing further away under semantic distance.

Comparison against Shallow Hashing Competitors: In Table 3, we report the performance on several shallow hashing competitors, as a direct comparison with the deep hashing methods in Table 2, where we can observe that (i) shallow hashing learning generally fails to compete with joint end-to-end deep learning, where supervised shallow methods outperform unsupervised competitors; (ii) Under the shallow hashing learning context, deep features outperform shallow hand crafted features by one order of magnitude.

No.	Model	Mean Average Precision				Precision @200			
		16 bits	24 bits	32 bits	64 bits	16 bits	24 bits	32 bits	64 bits
1	DLBHC [14]	0.5453	0.5910	0.6109	0.6241	0.5142	0.5917	0.6169	0.6403
2	DSH-Supervised [15]	0.0512	0.0498	0.0501	0.0531	0.0510	0.0512	0.0501	0.0454
3	DSH-Sketch [16]	0.3855	0.4459	0.4935	0.6065	0.3486	0.4329	0.4823	0.6040
4	Our+CEL	0.5969	0.6196	0.6412	0.6525	0.5817	0.6292	0.6524	0.6730
5	Our+CEL+CL	0.5567	0.5856	0.5911	0.6136	0.5578	0.6038	0.6140	0.6412
6	Our+CEL+SCL	0.6016	0.6371	0.6473	0.6767	0.5928	0.6298	0.6543	0.6875
7	Our+CEL+SCL+QL (Full)	**0.6064**	**0.6388**	**0.6521**	**0.6791**	**0.5978**	**0.6324**	**0.6603**	**0.6882**

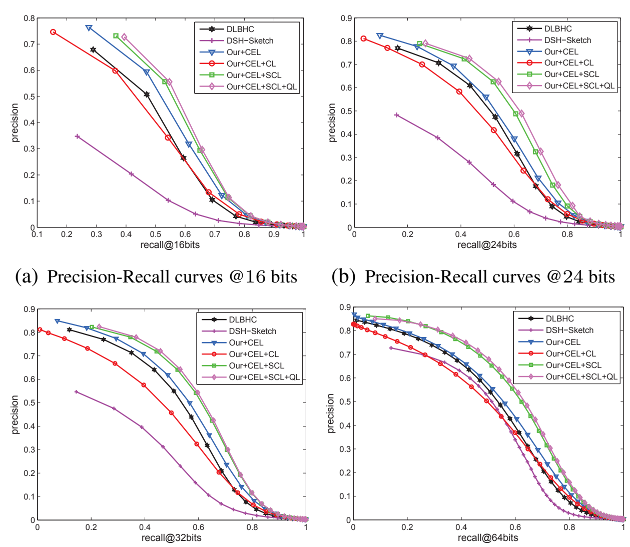
Table 2. Comparison with state-of-the-art deep hashing methods and our model variants on on QuickDraw-3.8M retrieval gallery.

		Unsupervised						Supervised	
		PCA-ITQ [4]	LSH [1]	SH [26]	SKLSH [19]	DSH [8]	PCAH [25])	SDH [22]	CCA-ITQ [4]
HOG	16 bits	0.0222	0.0110	0.0166	0.0096	0.0186	0.0166	0.0160	0.0185
	24 bits	0.0237	0.0121	0.0161	0.0105	0.0183	0.0161	0.0186	0.0195
	32 bits	0.0254	0.0128	0.0156	0.0108	0.0224	0.0155	0.0219	0.0208
	64 bits	0.0266	0.0167	0.0157	0.0127	0.0243	0.0146	0.0282	0.0239
deep feature	16 bits	0.4414	0.3327	0.4177	0.0148	0.3451	0.4375	0.5781	0.3638
	24 bits	0.5301	0.4472	0.5102	0.0287	0.4359	0.5224	0.6045	0.4623
	32 bits	0.5655	0.5001	0.5501	0.0351	0.4906	0.5576	0.6133	0.5168
	64 bits	**0.6148**	**0.5801**	**0.5956**	**0.0605**	**0.5718**	**0.6056**	**0.6273**	**0.5954**

Table 3. Comparison with shallow hashing competitors on QuickDraw-3.8M retrieval gallery.

model	Sketch-a-Net [30]	ResNet 50 [6]	our RNN branch	our CNN branch	our RNN&CNN + CEL	our RNN&CNN + CEL + SCL
accuracy	0.6871	0.7864	0.7788	0.7376	0.7949	**0.8051**

Table 4. Comparison with state-of-the-art recognition models and our model variants on sketch recognition task on QuickDraw-3.8M retrieval gallery.



(a) Precision-Recall curves @16 bits (b) Precision-Recall curves @24 bits

(c) Precision-Recall curves @32 bits (d) Precision-Recall curves @64 bits

Figure 5. Precision recall curves on QuickDraw-3.8M retrieval gallery. Best viewed in color.

len \ model / dist		Our+CEL	Our+CEL+CL	Our+CEL+SCL	Our+CEL+SCL+QL (Full)
16 bits	d_1	0.7501	0.5297	0.5078	0.5800
	d_2	4.9764	3.2841	4.2581	4.8537
	d_1/d_2	0.1665	0.1721	**0.1257**	0.1290
	MAP	0.5969	0.5567	0.6016	**0.6064**
24 bits	d_1	1.2360	0.8285	0.6801	0.8568
	d_2	6.1266	4.0388	5.0221	6.2243
	d_1/d_2	0.2017	0.2051	**0.1354**	0.1377
	MAP	0.6196	0.5856	0.6374	**0.6388**
32 bits	d_1	2.0066	1.5124	1.0792	1.2468
	d_2	8.9190	7.3120	7.5340	8.6675
	d_1/d_2	0.2250	0.2068	**0.1432**	0.1439
	MAP	0.6412	0.5911	0.6473	**0.6521**
64 bits	d_1	4.7040	3.5828	1.6109	2.5231
	d_2	15.4719	14.1112	11.6815	17.6179
	d_1/d_2	0.3040	0.2539	**0.1379**	0.1432
	MAP	0.6525	0.6136	0.6767	**0.6791**

Table 5. Statistic analysis for distances in the feature space of QuickDraw-3.8M under our model variants. d_1 and d_2 denote intra-class distance and inter-class distance, respectively.

Component Analysis: We have evaluated the effectiveness of different components of our model in Table 2. Specifically, we construct our model training with different loss combinations, including softmax cross entropy loss (Our+CEL), softmax cross entropy plus common center loss (Our+CEL+CL), softmax cross entropy plus sketch center loss (Our+CEL+SCL), softmax cross entropy plus sketch center loss plus quantization loss (Our+CEL+SCL+QL), which arrives our full model. We find that with cross entropy loss alone under our two-branch CNN-RNN model suffices to outperform best competitor, where by adding sketch center loss and quantization loss further boost the performance. It's noteworthy that adding common center loss harms the performance quite signifi-

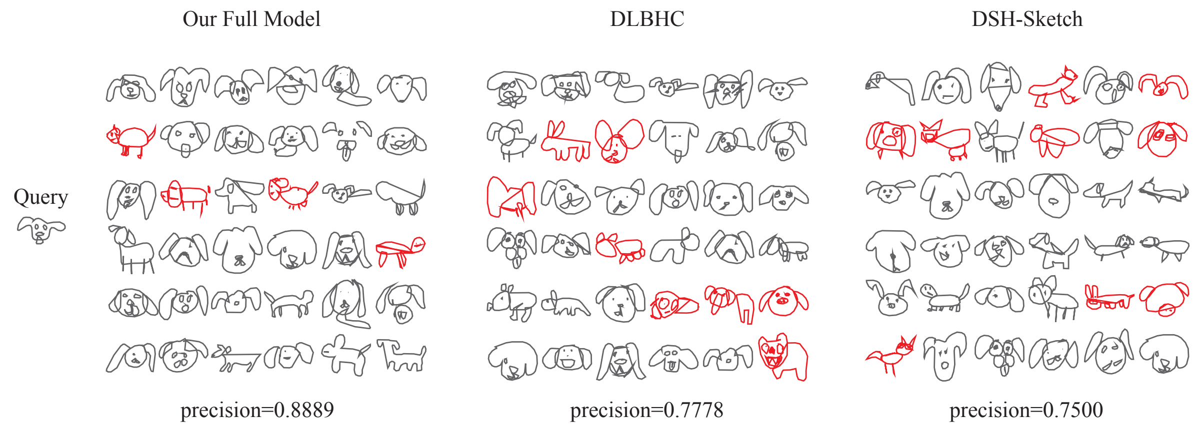


Figure 6. Qualitative comparison of top 36 retrieval results of our model and state-of-the-art deep hashing methods for query (dog) at 64 bits on QuickDraw-3.8M retrieval gallery. Red sketches indicates false positive sketch. The retrieval precision is obtained by computing the proportion of true positive sketch. Best viewed in color.

	16 bit	24 bit	32 bit	64 bit
Retrieval time per query (s)	0.089	0.126	0.157	0.286
Memory load (MB) 345,000 gallery sketches)	612	667	732	937

Table 6. Retrieval time (s) per query and memory load (MB) on QuickDraw-3.8M retrieval gallery.

cantly, validating our sketch-specific center loss design. In Figure 5, we plot the precision-recall curves for all above-mentioned methods under 16, 24, 32 and 64 bit hashing codes respectively, which further matched our hypothesis.

Further Analysis on Sketch Center Loss: To statistically illustrate the effectiveness of our sketch center loss, we calculate the average ratio of the intra-class distance $d1$ and inter-class distance $d2$, termed as $d1/d2$, among our 345 training categories. A lower value of such score indicates a better feature space learning, since the objects within the same category tend to cluster tighter and push further away with those of different semantic labels, as forming a more discriminative feature space. In Table 5, we witness significant improvement on the category structures brought by the sketch center loss across all hashing length setting (Our+CEL vs. Our+CEL+SCL), where on contrary, common center even undermines the performance (Our+CEL vs. Our+CEL+CL), which in accordance with what we've observed in Table 2.

Qualitative Evaluation: In Figure 6, we qualitatively compare our full model with DLBHC [14] and DSH-Sketch [16] on the dog category. It's interesting to observe (i) how our model makes less semantic mistakes; (ii) how our mistake is more reasonably understandable, *i.e.* sketch is confusing in itself in most of our falsely-retrieved sketches, while in other methods some clear semantic errors take place (e.g, pigs and rabbits).

Running Cost: We report the running cost as retrieval time (s) per query and memory load (MB) on QuickDraw-3.8M retrieval gallery (345,000 sketches) in Table 6, which even on million-scale can still achieve real-time retrieval performance.

4.4. Generalization to Sketch Recognition

To validate the generality of our sketch-specific design, we apply our two-branch CNN-RNN network to sketch recognition task, by directly adding a $2048d$ fully connected layer after joint fusion layer and before the 345-way classification layer. We compare with two state-of-the-art classification networks – Sketch-a-net [30] and ResNet-50 [6], where all above experiments are evaluated on the QuickDraw-3.8M retrieval gallery set. We demonstrate the results in Table 4, where following conclusion can drawn: (i) Exploiting the sketching temporal orders is important, and by combining the traditional static pixel representation, more discriminative power is obtained ($79.49\% vs. 68.71\%$). (ii) Sketch center loss generalizes to sketch recognition task, bringing additional benefits.

5. Conclusion

In this paper, we set out to study the novel problem of sketch hashing retrieval. By leveraging on a large-scale dataset of 3.8M human sketches, we explore the unique traits of sketches that were otherwise understudied in prior art. In particular, we show the benefit of stroke ordering information by encoding it in a CNN-RNN architecture, and we introduce a novel hashing loss that accommodates the abstract nature of sketches. Our hashing model outperforms all shallow and deep alternatives, and yields state-of-the-art performance when re-purposed for sketch recognition.

References

[1] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *IEEE Symposium on Foundations of Computer Science*, 2006. 3, 6, 7

[2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 1, 2

[3] M. Eitz, J. Hays, and M. Alexa. How do humans sketch objects? *ACM Transactions on Graphics (TOG)*, 2012. 1, 2

[4] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *TPAMI*, 2013. 3, 6, 7

[5] D. Ha and D. Eck. A neural representation of sketch drawings. *arXiv preprint arXiv:1704.03477*, 2017. 1, 2, 3, 4, 5, 6

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3, 7, 8

[7] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten. Densely connected convolutional networks. In *CVPR*, 2017. 3

[8] Z. Jin, C. Li, Y. Lin, and D. Cai. Density sensitive hashing. *IEEE Transactions on Cybernetics*, 2014. 6, 7

[9] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 3, 6

[11] Y. Li, T. M. Hospedales, Y.-Z. Song, and S. Gong. Free-hand sketch recognition by multi-kernel feature learning. *CVIU*, 2015. 1, 2

[12] Y. Li, Y.-Z. Song, T. M. Hospedales, and S. Gong. Free-hand sketch synthesis with deformable stroke models. *IJCV*, 2017. 1

[13] J. Lin, Z. Li, and J. Tang. Discriminative deep hashing for scalable face image retrieval. In *IJCAI*, 2017. 6

[14] 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. 3, 4, 6, 7, 8

[15] H. Liu, R. Wang, S. Shan, and X. Chen. Deep supervised hashing for fast image retrieval. In *CVPR*, 2016. 3, 6, 7

[16] L. Liu, F. Shen, Y. Shen, X. Liu, and L. Shao. Deep sketch hashing: Fast free-hand sketch-based image retrieval. In *CVPR*, 2017. 3, 5, 6, 7, 8

[17] A. Mahendran and A. Vedaldi. Visualizing deep convolutional neural networks using natural pre-images. *IJCV*, 2016. 6

[18] 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. 1

[19] M. Raginsky and S. Lazebnik. Locality-sensitive binary codes from shift-invariant kernels. In *NIPS*, 2009. 6, 7

[20] P. Sangkloy, N. Burnell, C. Ham, and J. Hays. The sketchy database: learning to retrieve badly drawn bunnies. *ACM Transactions on Graphics (TOG)*, 2016. 1, 2, 3

[21] R. K. Sarvadevabhatla, J. Kundu, and V. B. R. Enabling my robot to play pictionary: Recurrent neural networks for sketch recognition. In *ACM Multimedia*, 2016. 3

[22] F. Shen, C. Shen, W. Liu, and H. Tao Shen. Supervised discrete hashing. In *CVPR*, 2015. 3, 6, 7

[23] 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. 2

[24] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 3

[25] X.-J. Wang, L. Zhang, F. Jing, and W.-Y. Ma. Annosearch: Image auto-annotation by search. In *CVPR*, 2006. 6, 7

[26] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *NIPS*, 2009. 3, 6, 7

[27] Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, 2016. 4

[28] R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan. Supervised hashing for image retrieval via image representation learning. In *AAAI*, 2014. 3

[29] Q. Yu, F. Liu, Y.-Z. Song, T. Xiang, T. M. Hospedales, and C.-C. Loy. Sketch me that shoe. In *CVPR*, 2016. 1, 2

[30] 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. 1, 2, 3, 7, 8

[31] Q. Yu, Y. Yang, Y.-Z. Song, T. Xiang, and T. M. Hospedales. Sketch-a-net that beats humans. In *BMVC*, 2015. 3, 6

[32] F. Zhao, Y. Huang, L. Wang, and T. Tan. Deep semantic ranking based hashing for multi-label image retrieval. In *CVPR*, 2015. 3, 6