Sketch Down the FLOPs: Towards Efficient Networks for Human Sketch

Aneeshan Sain[1] Subhajit Maity[2*] Pinaki Nath Chowdhury[1] Shubhadeep Koley[1]
Ayan Kumar Bhunia[1] Yi-Zhe Song[1]
[1] SketchX, CVSSP, University of Surrey, United Kingdom.
[2] Department of Computer Science, University of Central Florida, United States.
{a.sain, p.chowdhury, s.koley, a.bhunia, y.song}@surrey.ac.uk; Subhajit@ucf.edu

Abstract

As sketch research has collectively matured over time, its adaptation for at-mass commercialisation emerges on the immediate horizon. Despite an already mature research endeavour for photos, there is no research on the efficient inference specifically designed for sketch data. In this paper, we first demonstrate existing state-of-the-art efficient light-weight models designed for photos do not work on sketches. We then propose two sketch-specific components which work in a plug-n-play manner on any photo efficient network to adapt them to work on sketch data. We specifically chose fine-grained sketch-based image retrieval (FG-SBIR) as a demonstrator as the most recognised sketch problem with immediate commercial value. Technically speaking, we first propose a cross-modal knowledge distillation network to transfer existing photo efficient networks to be compatible with sketch, which brings down number of FLOPs and model parameters by 97.96% *percent and* 84.89% *respectively. We then exploit the abstract trait of sketch to introduce a RL-based canvas selector that dynamically adjusts to the abstraction level which further cuts down number of FLOPs by two thirds. The end result is an overall reduction of* 99.37% *of FLOPs (from* 40.18*G to* 0.254*G) when compared with a full network, while retaining the accuracy (*33.03% *vs* 32.77%*) – finally making an efficient network for the sparse sketch data that exhibit even fewer FLOPs than the best photo counterpart.*

1. Introduction

A significant movement in the late computer vision literature has been that of model efficiency. This effort has been largely driven by the on-the-edge deployments of vision models. With great strides made on networks specifically tackling photo data (MobileNet [54], EfficientNet [63], etc), there however has been surprisingly no work targeting human sketches [72]. This is despite the abundance of work specifically addressing sketch data [47] and its unique traits:

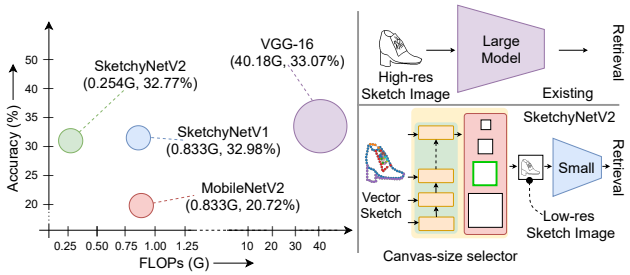

Figure 1. Our SketchyNetV1 compresses *existing* heavy FG-SBIR networks to deliver smaller models. Further enhanced via a canvas-selector module our SketchyNetV2 model minimises sketch-resolution dynamically to reduce FLOPs.

sequential [22], abstract [60], or stroke-wise [10], style-diversity [51], and data-scarcity [6, 20, 43]. This is particularly disappointing for the problem of fine-grained sketch-based image retrieval (FG-SBIR), which is perhaps the single-most studied sketch task [2, 7, 9, 13, 40, 43, 53, 75] that has already matured for at-mass commercial adoption, and is our problem of choice in this paper.

One would think that state-of-the-art models designed for photos would work as-is for sketches, treating sketch data as raster images (which sketch is not). As our first contribution, we preempt this thought via a pilot study to prove its false nature. In particular, we borrow MobileNetv2 [54] as a backbone network to construct a typical triplet loss based network for FG-SBIR. Results show a reduction of almost 37% (*relative*) in retrieval accuracy when compared with a typical VGG-16 backbone [57]. The focus of this paper is however not to design entirely new sketch-specific efficient networks from scratch – lots of good efficient photo networks already exist. Our goal instead is to adapt existing photo networks to work with sketch, by introducing generic sketch-specific designs that work in a plug-n-play manner.

We first tackle the problem highlighted in the said pilot study – to make existing photo efficient networks perform on the same level as large networks on FG-SBIR. We argue the reason behind why the likes of MobileNetV2 did not work off-the-shelf is that the limited set of parameters could not accommodate the fine-grained nature of sketches (i.e.,

*Work done as an intern at SketchX before joining UCF.

sketches are highly expressive in capturing fine-grained visual details [72]) for sketch and photo matching. For that, we introduce a *knowledge distillation* (KD) paradigm [48] to adapt this sketch trait onto efficient photo networks. The core idea is to induce strong semantic knowledge between a sketch and its photo, in the embedding space of a student, from a deep and high performing teacher. This particularly helps the efficient model to achieve on par performance with the deep variant without having a large parameter size. However, ours being a cross-modal retrieval, naively adapting standard KD paradigms involving logit distillation [3] would be sub-optimal. We therefore aim to preserve cross-modal pairwise distances among instances, and in turn the structural consistency of the teacher's discriminative embedding space in the student. The result is already an efficient FG-SBIR model (which we term SketchyNetV1) that reduces the number of FLOPs from 40.18G (VGG-16 based model[51]) to 0.833G (our MobileNetV2-based) – a roughly 48 times reduction (Fig. 1 left).

This abstract nature of human sketch [16] can also be exploited, to further reduce the FLOPs. The hypothesis is that since sketches are essentially abstract depictions of images, they should carry a similar level of semantic information even when rendered at much lower resolutions, and that more abstract a sketch is the more information compact it is (i.e., can sustain even lower resolutions). For that, we conducted another pilot study (Sec. 3), where we show unlike photos, sketches can sustain retrieval accuracy with much lower resolutions (4% vs 15% at 32×32), and 30% of those query sketches achieving perfect retrieval at full-resolution, when downsized to 32×32 can still retrieve accurately (Fig. 2 right). This motivated us to design a canvas selector that can choose the right resolution *on-the-fly* to render a sketch before feeding to the network. This is thanks to another unique trait of sketch data – they come in vector format and can be freely rendered to any resolution without extra overhead. Accomplishing this however is non-trivial since this selection process of canvas-size involves a discrete decision, thus invoking non-differentiability. We thus leverage reinforcement learning [9] to bypass these issues and train an abstraction-aware *canvas-size selector* as a policy network for our model, which estimates the optimal canvas-size required for the input sketch. The FG-SBIR model, acting as a critic, calculates reward for the canvas-selector, balancing between accuracy and FLOPs, that trains the policy network. With this further trait of sketches addressed, our final model (SketchyNetV2) was able to further reduce FLOPs by a factor of 3.28 (from 0.833G to 0.254G), while retaining a retrieval accuracy similar to the full model (33.03 vs 32.77), exactly as one would imagine sketch efficient networks to entail (sketches being sparse).

In summary, our contributions are: (a) the first investigation towards efficient sketch networks from existing photo ones (b) a knowledge distillation framework specifically designed to cater to the fine-grained nature of sketches, (c) a canvas-size selector that caters to the abstract nature of sketches, which further reduces FLOPs by two-thirds. Extensive experiments show our method to perform at par with existing state-of-the-arts, while being superiorly efficient.

2. Related Work

Fine-grained SBIR: Compared to category-level SBIR [17, 19, 36, 52], Fine-Grained SBIR (FG-SBIR) focuses on retrieving a single photo instance from a category-specific gallery based on a query sketch. Introduced as a deep triplet-ranking Siamese network [75], FG-SBIR has evolved with attention-based modules [59], textual tags [58], hybrid cross-domain generation [42], hierarchical co-attention [50], and various pre-training strategies like mixed modal jigsaw solving [44], reinforcement learning [6], and transformer-based knowledge distillation [53]. Addressing sketch-specific traits like style diversity [51], data scarcity [7], and sketch stroke redundancy [9], improvements have been made for enhanced retrieval [53]. Recently it was extended to scene-level (retrieve a scene-image, given a scene-sketch), employing cross-modal region associativity [13] and enhanced using text-queries [46]. However, sketch-specific computational efficiency has largely been ignored – which we address here in this paper.

Towards Computational Efficiency: Addressing large-scale deployment, research emerging in computational efficiency [1], include network pruning [37], quantisation [21], binarisation [18], knowledge distillation [25], input-size optimisation [62], or their combinations [1]. Network *pruning* [23, 68] involves discovering and dropping low impact weights or channels of a pre-trained large network, while retaining its original accuracy. Contrarily, *quantisation* directly reduces the bit-width of parameter-values [21] and gradients [77], by replacing floating-point computations with faster and cheaper low-precision fixed-point numbers [34]. Its extreme version, *binarisation* [18], binarises weights and activations. *Knowledge Distillation* aims at transferring knowledge of a large pre-trained *teacher* network to a smaller *student* for cost-effective deployment. Existing techniques leverage output logits [3], hidden layers [48], attention-maps [76], or neuron selectivity pattern [28] of pre-trained teachers for the same. Self-distillation [4] employs the same network for both student and teacher models, to further reduce compute. *Input-resolution optimisation* on the other hand involves replacing a high-resolution input image with its down-scaled low-resolution version [62], to reduce overall compute [73], while retaining accuracy. Relevant works explored dynamic usage of high-resolution data [64] or patch proposals for strategic image-cropping [69], where the learnable resizer [62] operates on the original image resolution. All such

Table 1. Performance & computational cost for various networks

Model	FLOPS	Parameters	Time (ms)	Acc@1
VGG-19 [57]	51.06G	20.02M	101	33.63
VGG-16 [57]	40.18G	14.71M	084	33.03
ResNet-101 [24]	20.50G	42.50M	155	21.77
DenseNet-201 [26]	11.40G	18.09M	335	23.62
ResNet-50 [24]	10.76G	23.51M	086	21.32
InceptionV3 [75]	07.72G	21.79M	134	24.27
ResNet-18 [24]	4.76G	11.18M	036	20.72
EfficientNet [63]	1.05G	04.01M	112	22.47
MobileNetV2 [54]	0.83G	02.22M	070	20.85

methods however cater to compressing image-specific models, which if used off-the-shelf for FG-SBIR would fail to address the sparse manner in which sketches are represented [8]. We thus propose a *sketch-specific* light-weight FG-SBIR model, tackling sparsity in sketches.

Reinforcement Learning in Vision (RL): Applying RL [29] to computer vision problems [33, 67] is commonplace of late. Whenever quantifying the *goodness* of the network's state is non-differentiable, like selecting which regions to crop in an image [31], RL comes in handy. In sketch community, RL has found use in retrieval [6, 7], modelling sketch abstraction [39, 40], designing competitive sketching agents [5] and as a stroke-selector [9]. Here, we leverage RL for optimising a canvas-size selector for a computationally efficient retrieval model.

3. Pilot Study

Backbone Network for FG-SBIR: Amongst popular backbones used in recent FG-SBIR literature [6, 7, 50, 51], are VGG-16 [30] and Inception-V3 [75]. Others like ResNet18 [24] and EfficientNet [63], though cheaper and prevalent in vision [24] are rarely used in FG-SBIR. Exploring their potential in FG-SBIR, we train a *baseline* FG-SBIR model, plugging in popular ones as backbones for our study. Table 1 reports their parameter-count, FLOPs, time for feature-extraction per sample and Top-1 accuracy (Sec. 5) on standard QMUL-ShoeV2 dataset [75]. Despite scoring more than VGG-16 on image-classification [65], ResNet50 [24] performs poorly – indicating that modelling fine-grained sketch-photo association needs networks with higher FLOPs. Despite delivering better accuracies, heavier networks cost far more resources. We thus aim for a *lighter* model that achieves *accuracy like a larger* one.

Dynamic Sketch-canvas-size: Unlike photos that hold pixel-dense information, sketches are sparse black and white lines. This begs the question, if a sketch rendered at a higher resolution with added computational burden would convey any extra semantic information than at a lower one. To answer this, we focus on vector-format of sketches [7], where every sketch vector s_v can be characterised as a sequence of points (v_1,v_2, $\cdots$,v_T), and rendered to any canvas-size (c) as a raster-image (s_r) via *rasterisation* $\mathcal{R}^c(\cdot) : s_v \rightarrow s_r^c$ where $s_r^c \in \mathbb{R}^{c\times c\times 3}$. A *single baseline* FG-SBIR model (VGG-16) is trained and tested on full-resolution images, and sketches rendered at varying

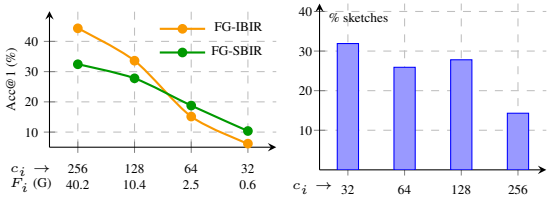

Figure 2. Pilot study for varying canvas-size (see text above).

canvas-sizes from C=$\{32 \times 32, \cdots, 256 \times 256\}$. Comparing sparse sketches *versus* photos, we train an equivalent fine-grained image-based image retrieval (FG-IBIR) where only the query against a positive photo is constructed using random augmentation [11] and resized to corresponding canvas-size c_i. Gallery photo-features for both are kept pre-computed. While *FG-IBIR* accuracy falls rapidly (Fig. 2 left), FG-SBIR stays relatively stable against decreasing canvas-sizes, as photos (unlike sketches) containing pixel-dense perfect information, lose a lot of it while down-scaling. Furthermore, *positive* accuracy of FG-SBIR at 32×32, shows some sketches to hold sufficient semantic information for retrieval even at minimal canvas-size. Fig. 2 (right) shows overall QMUL-ShoeV2[75] statistics where each bar represents the *smallest* canvas-size, at which *bar-height* percentage of sketches achieve perfect retrieval (taking total sketches achieving perfect retrieval at 256×256 as 100%). This infers that every sketch has its own optimal canvas, and fixing it (Fig. 2 left), for the entire dataset would always be sub-optimal. Moreover, the optimal canvas-size for a sketch, depends on its extent of sparsity, for which neither hard-ground-truth exists, nor is it knowable from the user's end. Finally, brute force discovery at every size being infeasible during inference, motivates us to design a *learnable* adaptive canvas-size selector for sketches.

4. Methodology

Overview: Our pilot study motivates us to extend a standard FG-SBIR framework towards computational efficiency by first introducing *SketchyNetV1* via knowledge distillation, and then taking it further to *SketchyNetV2* via dynamic sketch canvas-size selection. Importantly, these are done without hurting the original performance of large-scale slower baseline model. In particular, there are two phases of improvement for computational efficiency. Firstly, we train a **t**eacher model F_T as a standard baseline FG-SBIR model (Sec. 3). Once trained, its rich semantic knowledge is transferred to a smaller **st**udent network F_{st} via a knowledge distillation (KD) paradigm, which we name as *SketchyNetV1*. Next, we train a canvas-size selector that adaptively decides optimal canvas-size or resolution per sketch for *SketchyNetV2* that would be sufficient to preserve the retrieval performance while reducing the number of FLOP operations in feature extraction significantly.

***Baseline* FG-SBIR model:** Keeping aside complex pre-training [44] or joint-training [7], we focus on a simple

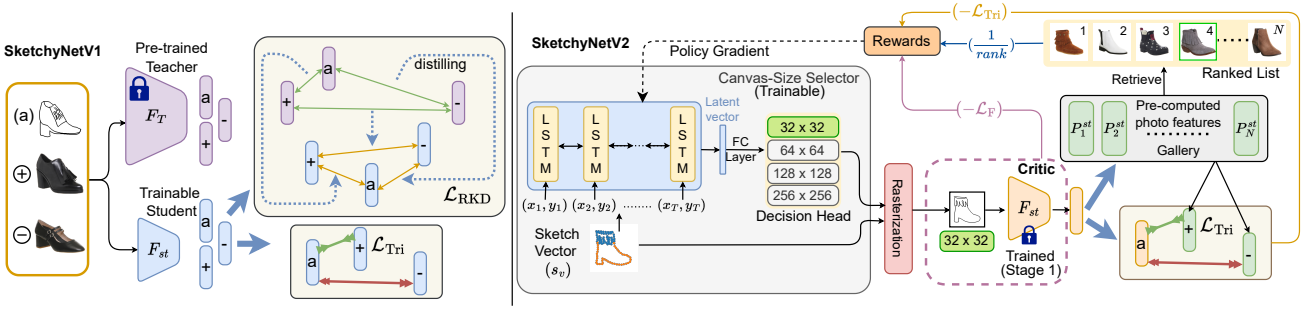

Figure 3. Two stage framework. (Left) SketchyNetV1: A smaller **st**udent network (F_{st}) is trained from a larger pre-trained **t**eacher (F_T) via KD (§4.1). (Right) We progress to SketchyNetV2 by training a canvas-size selector directed by objectives of increasing performance and reducing compute. It takes sketch as a vector (s_v) and aims to predict an optimal canvas-size at which the sketch is rasterised (s_r). s_r when processed by F_{st} minimises overall FLOPS, while retaining accuracy of corresponding full-resolution sketch-image.

triple-branch state-of-the-art Siamese network [75] which remains a strong baseline so far [7, 50, 51]. A VGG-16 [30] pre-trained on ImageNet [49] acts as the shared backbone for each branch. Given an input image I, a d dimensional feature embedding is extracted from the backbone network $\mathcal{F}$ using global average pooling followed by l_2 normalisation as $f_I \in \mathbb{R}^d$. The model is trained on triplets comprising features of an anchor sketch (f_s), its matching positive (f_p) photo and a random non-matching negative photo (f_n), via a triplet loss [70]. Triplet loss ($\mathcal{L}_{\text{Tri}}$) aims to minimise the distance between sketch and positive photo, while maximising the same from its negative (n). With $\delta(a,b) = ||a-b||^2$, as a distance function and a margin $m > 0$, we have:

$$\mathcal{L}_{\text{Tri}} = \max\{0, m + \delta(f_s, f_p) - \delta(f_s, f_n)\} \tag{1}$$

4.1. SketchyNetV1: Smaller Model via KD

Based on our pilot study (Sec. 3), if we naively train from an ImageNet pre-trained standard smaller network (e.g. MobileNetV2 [54] – usually popular for edge devices), with simple supervised triplet loss (Eq.1) it can hardly reach the accuracy (Table 1) obtained by a large network like VGG-16. Therefore apart from training via a supervised loss on the sketch-positive-negative triplet ($\mathcal{L}^{st}_{\text{Tri}}$ as in Eq.1), we aim to use the pre-trained larger teacher (e.g. VGG-16) for additional supervision. Accordingly, we adhere to Knowledge Distillation (KD) paradigm [48], a conventional strategy for model compression, to deliver a light-weight student model from the larger teacher network [48]. Applying KD to our FG-SBIR paradigm is however non-trivial. Unlike traditional KD methods, which often involve logit distillation for classification tasks, our scenario involves cross-modal retrieval. Here, the output is a continuous d-dimensional feature within a joint-embedding space. Furthermore, straightforward regression between teacher and student features for sketch and photo branches may encounter compatibility issues, given the disparity in the embedding spaces. Addressing this, if the dimensions of teacher and student embeddings differ, an additional feature transformation layer is necessary for alignment [48]. Overcoming these issues, we aim to distil the inter-feature distances from respective teacher networks to the student thus preserving structural orientation of features in the teacher's embedding space to that in the student, during KD (Fig. 3 (left)).

Given features of sketch-positive-negative triplet of teacher (f^T_s, f^T_p, f^T_n), we calculate the inter-feature Euclidean distances in the output representation space of teacher (F_T) using $\delta(\cdot,\cdot)$ (Eq.1), as $d^T_{sp} = \delta(f^T_s, f^T_p)$; $d^T_{sn} = \delta(f^T_s, f_n)$ and $d^T_{pn} = \delta(f^T_p, f^T_n)$. Passing the same samples through the student, we similarly obtain d^{st}_{sp}, d^{st}_{sn} and d^{st}_{pn} using ($f^{st}_s, f^{st}_p, f^{st}_n$). A smooth l_1 loss distils the l_2 distances from the teacher to the student. With $\mathcal{L}_\delta$ as a Huber-loss [27], relational distillation loss for sketch-photo pair between teacher and student is given as,

$$\mathcal{L}^{sp}_{\text{RKD}} = \mathcal{L}_\delta(d^T_{sp}, d^{st}_{sp}) \quad \text{where,}$$
$$\mathcal{L}_\delta(a,b) = \begin{cases} \frac{1}{2}(a-b)^2 & \text{if } |a-b| < \beta \\ \beta(|a-b| - \frac{1}{2}\beta) & \text{otherwise} \end{cases} \tag{2}$$

Similarly, we obtain $\mathcal{L}^{sn}_{\text{RKD}}$ and $\mathcal{L}^{pn}_{\text{RKD}}$, to compute distillation loss, $\mathcal{L}_{\text{RKD}} = \mathcal{L}^{sp}_{\text{RKD}} + \mathcal{L}^{sn}_{\text{RKD}} + \mathcal{L}^{pn}_{\text{RKD}}$. With hyperparameter λ, overall student-training objective becomes:

$$\mathcal{L}^{st}_{\text{trn}} = \lambda\mathcal{L}_{\text{Tri}} + (1-\lambda)\mathcal{L}_{\text{RKD}} \tag{3}$$

The student trains on varying sketch-canvas-sizes (C), using their respective full resolution for teacher, to ensure scale-invariance as $\mathcal{L}^{st*}_{\text{trn}} = \frac{1}{4}\sum_{i=1}^{4}\mathcal{L}^{st(c_i)}_{\text{trn}}$. However, as optimal canvas-size varies across sketches (Fig. 2), we need a learnable canvas-size selector, dynamically predicting the optimal size for each sketch.

4.2. SketchyNetV2: Adaptive canvas-size Selector

Overview: Understanding that an input at a lower resolution, would incur lesser FLOPs on evaluation than a higher one, we upgrade *SketchyNetV1* (F_{st}) to a more computationally efficient *SketchyNetV2* by introducing an adaptive canvas-size selector. It is designed in the vector modality as it is much cheaper [71] than its raster-image counterpart, and can dynamically encode the varying abstraction level of sketches. Furthermore, predicting the optimal canvas-size from vector modality, would make the corresponding rasterisation operation $\mathcal{R}^c(\cdot)$ at lower sizes cheaper, besides reducing FLOPs in feature extraction. Taking sketch vector s_v as input, ψ_C predicts the probability $p(c|s_v)$ of choosing one of K

discreet canvas-sizes from a set of $C = \{32 \times 32, \cdots, 256 \times 256\}$. s_v is then rasterised to $s_r^c \in \mathbb{R}^{c \times c \times 3}$ with the optimal canvas-size c, as $\mathcal{R}^c(\cdot) : s_v \rightarrow s_r^c$, and fed to F_{st} for retrieval.

Selecting an optimal canvas-size however is an ill-posed problem. Firstly, there are no explicit labels representing the optimal canvas-size for a sketch. Secondly, annotating the optimal canvas-size for the whole dataset via brute-force iteration is computationally impractical. We therefore use the pre-trained F_{st} as a critic to guide the learning of sketch canvas-size selector via Reinforcement Learning [29] as rasterisation is a non-differentiable operation. There are two major objectives here: a) retain the original resolution accuracy and b) encourage the use of the lower resolution to improve the computational efficiency.

Model: Any sequential network from RNN [17] or Transformer [35] families can be used to encode the vector-sketch s_v. To recap, sketch-vector (s_v) represents a sequence of points [v_1,v_2,...,v_T], where $v_t = (x_t, y_t, q_t^1, q_t^2, q_t^3) \in \mathbb{R}^{T \times 5}$; T is the sequence length, (x_t, y_t) denotes the absolute coordinates in a normalised $c \times c$ canvas, while the last three represent pen-states [22]. Furthermore, the complexity of canvas-size selector being dependent on sequence-length (T) varying across sketches, we use Douglas Peucker Algorithm [66] to limit sequence-length across all sketch-samples at T_{max} without losing visual representation. We feed v_t at every time step to a simple 1-layer GRU network with $\mathbb{R}^{d_v}$ hidden states and take the final hidden state as the encoded latent representation ($f_{s^\text{T}} \in \mathbb{R}^{d_v}$). Next, we apply a linear layer (γ) to get $p(c|s_v) = \texttt{softmax}(W_\gamma f_{s^T} + b_\gamma)$, where $W_\gamma \in \mathbb{R}^{d_v \times K}$ and $b \in \mathbb{R}^K$, as shown in Fig. 3 (right). Given $p(c|s_v) \in \mathbb{R}^K$ over K different canvas-sizes for every s_v, canvas-size c can be sampled from categorical distribution as $c_{pred} \sim \texttt{categorical}([p(c_1|s_v), \cdots, p(c_K|s_v)])$. Accordingly, s_v is rasterised to s_r at canvas-size c_{pred}, which is then fed to F_{st}. In particular, given a sketch s_v, canvas-size selector ψ_C acting as a policy network, takes *action* of selecting the canvas-size c_{pred} from action space C, and ψ_C is optimised over a *reward* calculated by the SketchyNetV1 model (F_{st}) acting as the critic.

Reward Design: The objectives of our canvas-selector are to choose an optimal canvas-size such that: (i) accuracy is retained (ii) overall FLOPs is minimised. We therefore design our reward from two perspectives. Now as F_{st} is fixed, we pre-compute the features of all gallery photos. During training, we only extract the feature of *rasterised* sketch and calculate its corresponding rank using the pre-computed photo-features. Furthermore, taking one sketch we pre-compute FLOPs for every canvas-size in C. From accuracy perspective, the aim is to select the optimal canvas-size c_opt from C for each s_v such that the rendered s_r can retrieve the matching photo p at numerically lowest rank r (best r=1). Following conventional norm of reward maximisation, we define accuracy reward (R_acc) as weighted (by hyper-parameters λ_r, λ_Tri) summation of inverse of the rank (r) and negative triplet loss (following Eq.1) as:

$$R_\text{acc} = \lambda_r(1/r) + \lambda_\text{Tri}(-\mathcal{L}_\text{Tri}) \tag{4}$$

From the compute perspective, the selection criterion should reward choosing a lower canvas-size. Moreover, higher performance being naturally inclined towards a higher-canvas-size thus increasing overall compute, we need an objective dedicated towards reducing computational cost. We thus propose a FLOPs constraint regularisation to guide the learning of canvas-selector as:

$$\mathcal{L}_\text{F} = \frac{\sum_{j=1}^{K}(q_j \cdot \eta_j)}{q_{max} - q_{min}} \tag{5}$$

where, $\eta_i = p(c_i|s_v)$ and q_j is the pre-computed FLOPs value for the j-th canvas-size with maximum at q_{max} and minimum at q_{min}. Treating (-$\mathcal{L}_\text{F}$) as a reward from the compute perspective, we define our compute reward as $R_\text{comp} = (-\mathcal{L}_\text{F})$. Taking λ_F as a balancing hyper-parameter, we combine both perspectives as our total reward:

$$R_\text{Tot} = \lambda_\text{F} R_\text{comp} + (1 - \lambda_\text{F}) R_\text{acc} \tag{6}$$

Optimisation Objective: Finally, we train the canvas-size selector using popular Policy Gradient [61] method which is both simple to implement and requires less hyper-parameter tuning than other state-of-the-art alternatives [56]. If $p(c|s_v^i)$ be the probability of selecting a specific canvas-size for i^{th} sketch sample s_v^i and the corresponding reward be R_Tot^i, the objective function for policy network (canvas-size selector) over a batch of size B becomes:

$$\mathcal{L}_{PG}(\theta) = -\frac{1}{B}\sum_{i=1}^{B} \log p(c|s_v^i) \cdot R_\text{Tot}^i \tag{7}$$

5. Experiments

Datasets: We evaluated on the following publicly available datasets having fine-grained sketch-photo association. **QMUL-ShoeV2** [75] and **QMUL-ChairV2** [75] contains 6730/2000 and 1800/400 sketches/photos respectively. Keeping 679/200 (525/100) sketches/photos for evaluation from ShoeV2 (ChairV2) respectively we use the rest for training. **Sketchy** [55] contains 125 categories with 100 photos each, having around 5 sketches per photo. While, training uses a standard train-test split [55] of 9:1, during inference we construct a challenging gallery using photos across one category for retrieval. For scene-level FG-SBIR evaluation we use **FS-COCO** [15] which includes 10,000 unique sketch-photo pairs with a 7:3 train/test split.

Implementation Details: Considering a standard large FG-SBIR network as a teacher, our proposed student uses a cheaper backbone feature extractor MobileNetV2 [54], a standard for mobile end-devices. Margin is set to $m = 0.2$ for triplet loss (Secs. 3 and 4.1). We use an Adam optimiser at a learning rate of 0.0001 and a batchsize of 16 for 200 epochs. For practicality, we chose discrete canvas-sizes (C) instead of a continuous action space, where $C = \{32 \times 32, 64 \times 64, 128 \times 128, 256 \times 256\}$. Canvas-size selector ($\psi_C$) is modelled using a simple GRU network, with embedding size 128, and trained via reinforcement learning with a learning rate of 0.0001 and a batchsize of 32 for 500 epochs, keeping T_{max}=100 for vector-sketches. For the action space, we take varying canvas-sizes from C and set $\beta = 1$ in Eq.2. λ, λ_r, λ_Tri and λ_F are empirically set to 0.5, 0.4, 0.48 and 0.35 (Fig.5) respectively. While training ψ_C, sketches are rendered at different completion (30%, 35%, ... 100%) levels to simulate varying levels of *abstraction*, ensuring a *sketch-abstraction-aware* canvas-selector (*e.g.*, a *smaller* canvas for *less detailed* sketch).

Evaluation Metric: *Accuracy* for FG-SBIR is the percentage of sketches where true-match photos are ranked in the Top-q (q = 1,10) lists. *Computational efficiency* is measured via parameter-count and average FLOPs – lower is better. For average FLOPs in Table 2, we take the combined sum of FLOPs per sample [12] from (i) canvas-size selector and (ii) retrieval model, using the predicted resolution, and average across the validation set.

Table 2. Quantitative Analysis on FG-SBIR. Best viewed when *zoomed*.

	Methods	Canvas-size	Params	ShoeV2 [75]			ChairV2 [75]			Sketchy [55]			FSCOCO [15]		
		c×c	(mil.)	Top1 (%)	Top10 (%)	FLOPS (G)	Top1 (%)	Top10 (%)	FLOPS (G)	Top1 (%)	Top10 (%)	FLOPs (G)	Top1 (%)	Top10 (%)	FLOPS (G)
State-of-the-Arts	Triplet-SN [75]	32x32	8.75	09.77 (↓18.94)	24.82 (↓46.74)	0.083	15.61 (↓32.04)	29.86 (↓54.38)	0.083	4.23 (↓11.12)	12.58 (↓23.13)	0.083	0.76 (↓3.94)	6.22 (↓14.8)	0.083
		64x64	8.75	17.63 (↓11.08)	44.96 (↓26.60)	0.338	30.25 (↓17.4)	54.21 (↓30.03)	0.338	9.68 (↓5.67)	22.45 (↓13.26)	0.338	1.85 (↓2.85)	7.33 (↓13.69)	0.338
		128x128	8.75	25.01 (↓3.70)	62.35 (↓9.21)	1.397	40.61 (↓7.04)	72.68 (↓11.56)	1.397	12.98 (↓2.37)	29.63 (↓6.08)	1.397	3.28 (↓1.42)	15.92 (↓5.1)	1.397
		256x256	8.75	28.71	71.56	5.280	47.65	84.24	5.280	15.35	35.71	5.280	4.7	21.02	5.280
		SketchyNetV1	2.22	28.46 (↓0.25)	69.01 (↓2.55)	0.833	46.28 (↓0.37)	82.33 (↓1.91)	0.833	14.85 (↓0.5)	34.68 (↓1.03)	0.833	4.22 (↓0.48)	18.33 (↓2.69)	0.833
		SketchyNetV2	2.27	27.89 (↓0.82)	68.76 (↓2.80)	0.264	45.98 (↓1.67)	80.14 (↓4.10)	0.294	14.21 (↓1.14)	34.16 (↓1.55)	0.321	3.98 (↓0.72)	17.64 (↓3.38)	0.423
	HOLEF-SN [59]	32x32	9.31	10.84 (↓20.9)	27.28 (↓48.5)	0.096	18.72 (↓34.69)	31.68 (↓55.88)	0.096	6.01 (↓10.69)	14.12 (↓24.78)	0.096	0.86 (↓4.04)	6.93 (↓14.78)	0.096
		64x64	9.31	19.83 (↓11.91)	51.72 (↓24.06)	0.387	34.32 (↓19.09)	56.46 (↓31.1)	0.387	9.92 (↓6.78)	24.38 (↓14.52)	0.387	2.11 (↓2.79)	7.86 (↓13.85)	0.387
		128x128	9.31	26.78 (↓4.96)	66.28 (↓9.5)	1.545	45.74 (↓7.67)	75.59 (↓11.97)	1.545	13.81 (↓2.89)	33.68 (↓5.22)	1.545	3.65 (↓1.25)	16.32 (↓5.39)	1.545
		256x256	9.31	31.74	75.78	5.758	53.41	87.56	5.758	16.70	38.90	5.758	4.9	21.71	5.758
		SketchyNetV1	2.22	31.59 (↓0.15)	73.96 (↓1.82)	0.833	53.27 (↓0.14)	86.93 (↓0.63)	0.833	16.25 (↓0.45)	38.21 (↓0.69)	0.833	4.56 (↓0.34)	19.92 (↓1.79)	0.833
		SketchyNetV2	2.27	30.86 (↓0.88)	72.56 (↓3.22)	0.259	52.74 (↓0.67)	84.02 (↓3.54)	0.289	15.91 (↓0.79)	37.88 (↓1.02)	0.315	4.04 (↓0.86)	18.63 (↓3.08)	0.417
	Triplet-RL [6]	32x32	22.1	11.98 (↓22.12)	28.11 (↓50.71)	0.142	20.53 (↓36.01)	32.59 (↓57.02)	0.142	1.76 (↓2.94)	3.88 (↓6.47)	0.142	–	–	–
		64x64	22.1	21.28 (↓12.82)	54.71 (↓24.11)	0.577	36.09 (↓20.45)	57.33 (↓32.28)	0.577	3.04 (↓1.66)	6.79 (↓3.56)	0.577	–	–	–
		128x128	22.1	28.83 (↓5.27)	67.84 (↓10.98)	2.299	48.38 (↓8.16)	77.25 (↓12.36)	2.299	4.12 (↓0.58)	9.05 (↓1.30)	2.299	–	–	–
		256x256	22.1	34.10	78.82	6.041	56.54	89.61	6.041	4.70	10.35	6.041	–	–	–
		SketchyNetV1	2.22	33.88 (↓0.22)	77.15 (↓1.67)	0.833	55.92 (↓0.62)	88.20 (↓1.41)	0.833	4.59 (↓0.11)	10.21 (↓0.14)	0.833	–	–	–
		SketchyNetV2	2.27	33.26 (↓0.84)	76.84 (↓1.98)	0.255	55.14 (↓1.40)	87.31 (↓2.30)	0.285	4.51 (↓0.19)	9.89 (↓0.46)	0.308	–	–	–
	StyleVAE [51]	32x32	25.37	12.68 (↓23.79)	28.69 (↓53.14)	0.125	22.48 (↓40.38)	32.10 (↓59.04)	0.125	06.92 (↓12.7)	16.25 (↓29.53)	0.125	–	–	–
		64x64	25.37	22.93 (↓13.54)	57.71 (↓24.12)	0.508	40.26 (↓22.6)	58.23 (↓32.91)	0.508	12.46 (↓7.16)	29.62 (↓16.16)	0.508	–	–	–
		128x128	25.37	30.91 (↓5.56)	70.06 (↓11.77)	2.024	53.88 (↓8.98)	78.64 (↓12.5)	2.024	17.15 (↓2.47)	38.84 (↓6.94)	2.024	–	–	–
		256x256	25.37	36.47	81.83	5.642	62.86	91.14	5.642	19.62	45.78	5.642	–	–	–
		SketchyNetV1	2.22	36.11 (↓0.36)	79.63 (↓2.20)	0.833	62.23 (↓0.63)	88.95 (↓2.19)	0.833	19.48 (↓0.12)	44.02 (↓1.76)	0.833	–	–	–
		SketchyNetV2	2.27	35.88 (↓0.59)	78.46 (↓3.37)	0.251	61.95 (↓0.91)	87.68 (↓3.46)	0.281	19.10 (↓0.50)	43.57 (↓2.21)	0.304	–	–	–
	Partial-OT [14]	32x32	22.1	–	–	–	–	–	–	–	–	–	6.11 (↓18.05)	24.18 (↓29.74)	0.196
		64x64	22.1	–	–	–	–	–	–	–	–	–	12.32 (↓11.84)	37.65 (↓16.27)	0.687
		128x128	22.1	–	–	–	–	–	–	–	–	–	19.61 (↓4.55)	45.12 (↓8.8)	2.876
		256x256	22.1	–	–	–	–	–	–	–	–	–	24.16	53.92	6.512
		SketchyNetV1	2.22	–	–	–	–	–	–	–	–	–	23.89 (↓0.27)	51.29 (↓2.63)	0.833
		SketchyNetV2	2.27	–	–	–	–	–	–	–	–	–	23.77 (↓0.39)	50.68 (↓3.24)	0.325
Baselines	B-BFR	224x224	4.61	24.32	61.23	2.602	43.39	76.33	2.602	10.47	24.34	2.602	2.15	11.61	2.602
	B-DRS	Dynamic	9.36	25.12	68.84	5.493	44.78	78.14	6.187	11.87	26.58	6.245	3.86	17.63	6.291
	B-Crop	Dynamic	4.92	20.32	52.12	6.562	35.07	57.68	6.597	08.27	20.19	6.629	1.98	09.67	6.688
	B-Regress [48]	Dynamic	2.27	22.82	61.33	0.261	38.42	64.65	0.291	08.69	21.98	0.323	2.07	10.15	0.375
	B-AKD [76]	Dynamic	2.27	23.67	64.26	0.268	38.61	71.88	0.295	12.67	27.71	0.331	2.91	14.01	0.392
	B-PKT [45]	Dynamic	2.27	24.31	65.91	0.259	42.68	73.66	0.288	12.27	28.31	0.317	3.45	16.22	0.372
	B-VGG16-SN	256x256	14.71	33.03	78.51	40.18	52.16	84.08	40.18	18.61	41.25	40.18	5.16	23.62	40.18
	B-ThiNet [37]	256x256	8.32	7.51 (↓25.52)	22.34 (↓56.17)	9.342	12.31 (↓39.85)	26.63 (↓57.45)	9.342	3.16 (↓15.45)	12.73 (↓28.52)	9.342	0.65 (↓4.51)	5.35 (↓18.27)	9.342
	B-Prune [38]	256x256	6.17	3.92 (↓29.11)	17.32 (↓61.19)	8.138	7.98 (↓44.18)	19.82 (↓64.26)	8.138	1.45 (↓17.16)	8.22 (↓33.03)	8.138	0.53 (↓4.63)	4.97 (↓18.65)	8.138
	B-VGG16-SN	SketchyNetV2	2.27	32.77 (↓0.26)	78.21 (↓0.3)	0.254	50.75 (↓1.41)	82.21 (↓1.87)	0.283	16.55 (↓2.06)	38.66 (↓2.59)	0.332	4.69 (↓0.47)	21.76 (↓1.86)	0.407

5.1. Competitors

We evaluate against State-of-the-Arts (SoTAs) and a few relevant curated baselines (**B-**). **(i) SoTAs:** Unlike *Triplet-SN* [75], *HOLEF-SN* [59], *Triplet-RL* [6] (we report *full*-sketch evaluation) and *StyleVAE* [51]; *Partial-OT* [14] specifically caters to scene-level FG-SBIR. For fairness, we vary input-resolution of sketches during inference (fixed at their original 256 × 256 setup during training). We also adapt them as teachers, to judge our accuracy retainment after model-compression (*SketchyNetV1*) using full-res sketch and, further optimising canvas-size with canvas-selector (*SketchyNetV2*). **(ii) KD baselines:** Here we re-purpose existing works in a cross-modal setting, keeping our canvas-selector same, with *Triplet-SN* as the teacher. *B-Regress* – regresses features from teacher and student for both sketches and images, aligning their dimensions with an extra transformation layer [48], using a l_2 regression loss. Following [45] *B-PKT* calculates the conditional probability density for any pair of points in the teacher's embedding space [45], which models the probability of any two samples being close together. Sampling 'N' instances, it establishes a probability distribution for pairwise interactions within that space. Simultaneously, in the student's embedding space, a comparable distribution is derived, and the model minimises the divergence between them using a KL-divergence loss. *B-AKD* follows [76] in using spatial attention maps for knowledge transfer using similar dimensional matching like *B-Regress*. **(iii) Resizing strategies:** Keeping the same KD-paradigm with *Triplet-SN*-teacher, we vary the input-size resizing module following existing works. *B-BFR* follows [62] in designing a bilinear feature resizer amidst the intermediate layers of a CNN network, to output a down-scaled image. We choose its resized resolution of 224×224 (lowest) for subsequent evaluation. *B-DRS* follows [12] in training a convolutional module that takes an image in full-resolution, calculates a probability vector on available canvas-sizes, and transforms it into binary decisions, indicating the scale factor of selection. *B-Crop* follows [32] in meta-learning an image-cropping module that crops the input image to a target resolution for low compute evaluation. **(iv) Pruning alternatives:** Here we adapt two VGG16-backboned methods off-the-shelf from *network-pruning family* among other efficiency paradigms [18, 21, 62], in FG-SBIR setup – *B-ThiNet* [37] and *B-Prune* [38]. For comparative reference we report performance of a VGG16-backboned *Triplet-SN* network (*B-VGG16-SN*) and its *SketchyNetV2* variant.

5.2. Performance and Compute Analysis:

Analysis on Accuracy: Table 2 reports the quantitative evaluation of our model against others. *Triplet-SN* [75] and *HOLEF-SN* [59] perform lower owing to weak backbones of Sketch-A-Net [74]. Enhanced by its RL-optimised reward function *Triplet-RL*[6] scores higher (2.36%↑ Top1 on ShoeV2) but fails to surpass *StyleVAE* [51], due to its delicate meta-learned disentanglement module addressing style-diversity. Being trained on region-wise sketch-photo associativity, *Partial-OT* [14] outperforms all others significantly on scene-level sketches (FS-COCO). We also notice a

gradual drop in accuracy of SoTAs, when subjected to decreasing resolution of input-sketches suggesting that some methods are inherently accustomed to handling scale-shifts better than others, but are inferior to our dynamic approach (*SketchyNetV2*). Importantly, when most SoTAs are employed as teachers, their respective students (*SketchyNetV1* variants) are seen to retain their respective original accuracy with a marginal drop of $\approx 0.5 - 1\%$ overall, thanks to our efficient KD paradigm. Having different teacher-student embedding spaces both *B-Regress* and *B-AKD*'s paradigm of directly regressing on features proves incompatible to a cross-modal retrieval setting, dropping accuracy (compared to *T-Triplet-SN*), whereas lacking cross-modal discrimination *B-PKT* performs lower than ours, proving our superiority in retaining accuracy.

Figure 4. Exemplar sketches at their optimal canvas-size

Analysis on Compute: Although FLOPs [12] decrease (Table 2) with a reduction in input resolution across all SoTAs, it comes at a severe cost of accuracy (≈21% average drop from 256×256 to 32×32 on ShoeV2). This is mainly because *ideal* canvas-size for *optimal* retrieval varies across samples (Fig. 2 right) – thus subjecting all samples to a fixed canvas-size proves detrimental for accuracy. However, a positive Acc@1 score of ≈11% even at a low canvas-size of 32×32 for every SoTA shows that some sketches are indeed recognisable if the model is trained properly. This is verified further by *SketchyNetV2* variant of respective SoTAs, that secures the lowest FLOPs per SoTA, despite maintaining accuracy at par with its SoTA-counterpart – thanks to our RL-guided canvas-selector, which can allot an optimal canvas-size per sketch towards better retrieval (Fig. 4). Contrarily, *B-BFR* fails to lower FLOPs as it fixes a 224x224 input resolution for a satisfactory Acc@1 compared to *Triplet-SN*. *B-Crop* fairs better, however sketch being sparse, cropped sketches leads to confused representations, lowering accuracy. Despite coming closest in accuracy to ours on *Triplet-SN*, *B-DRS* lags behind considerably on compute, thanks to our canvas-selector working in vector space (inputs s_v) and eliminating costly convolutional operations, unlike others using full-resolution sketch-raster as input. Being untrained to handle *sparse-nature* of sketches during such *high* compression, performance of both *B-ThiNet* and *B-Prune* collapses (*vs.* *B-VGG16-SN*) unlike its *SketchyNetV2*. Notably, the significant drop in parameter-count (4 - 10 times) against SoTAs, comes from choosing lightweight MobileNetV2 [54] as a student coupled with our canvas-size selector in every *SketchyNetV2* variant. Compared to ShoeV2, FLOPs increase slightly for ChairV2, Sketchy and FS-COCO, owing to higher object (ChairV2) or scene-level (FS-COCO) details, and lower structural correspondence with photos (Sketchy), thus training a poorer retrieval model which in turn acts as a poor critic for our canvas-selector. Our canvas-selector alone takes a negligible 51.84K parameters with 0.02G FLOPs.

5.3. Ablation Study

As our teacher (F_T) can be any pre-trained FG-SBIR network, we limit our ablation to justifying design choices of our student model and canvas-size selector. We use *B-VGG16-SN* as a reference F_T – *Type-*$\emptyset$ in Table 3.

Training Canvas-Size selector ψ_C : To justify loss objectives of ψ_C we evaluate on ShoeV2, eliminating r^{-1}, $\mathcal{L}^R_{\text{Tri}}$, $\mathcal{L}_{\text{F}}$ one by one from Eq. 6 (Table 3). While eliminating accuracy rewards, (*w/o* $rank^{-1}$, *w/o* $\mathcal{L}^R_{Tri}$) increases constraint on lowering FLOPs, dropping accuracy, ignoring $\mathcal{L}_{\text{F}}$ (*w/o* $\mathcal{L}_F$) gains accuracy but costs high FLOPs. We thus optimally balance between the two using λ_{F} (Fig. 5 left). One can also choose to train for *faster* retrieval (lower FLOPs), given a decent Acc@5 (Fig. 5 right).

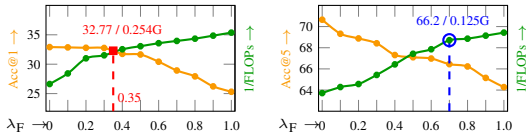

Figure 5. Impact of varying λ_{F} on Acc@1 and Acc@5

Architectural insights: To judge the efficiency of our student's backbone, we explore a few cheaper backbone-networks (*Types IV, V* in Table 3). Evidently, MobileNetV2 (Ours) balances optimally between compute and accuracy. As an option to vector input for canvas-selector, we re-design it with a simple seven-layer CNN module following [78] to use the *full-resolution* sketch-image as its input, and select a target canvas-size to downscale it for retrieval. With extra computational cost coming from spatial domain, total FLOPs incurred is much higher (*VI*). Next we compare efficacy of sketch as input to ψ_C with respect to types of vector format – offset-coordinate (*VI*) [22] *vs.* absolute ones (*Ours*). Turns out the former is better for encoding. Comparing sketch encoders among LSTM (*VIII*), GRU (Ours) and Transformer (*IX*), showed ours to be optimum empirically (Table 3). Furthermore, prepending recent pre-processing modules like *stroke-subset selector* of [9] to our canvas-selector, yields 42.95% (*vs.* 43.7% [9]) Acc@1 ShoeV2, proving its off-the-shelf compatibility.

Table 3. Ablative studies (accuracy on ShoeV2)

	Type	FLOPs (G)	Params (mil.)	ShoeV2 Top1 (%)	ShoeV2 Top10 (%)
$\emptyset$	B-VGG16-SN	40.18	14.71	33.03	78.51
I	w/o $rank^{-1}$-reward	0.173	2.27	25.31	64.36
II	w/o $\mathcal{L}^R_{\text{Tri}}$-reward	0.182		27.78	69.32
III	w/o $\mathcal{L}_{\text{F}}$-reward	0.833		32.91	78.39
IV	ResNet18 backbone	1.451	11.18	26.72	68.18
V	EfficientNet backbone	0.459	4.01	29.47	72.04
VI	Image-based ψ_C	5.730	7.62	32.21	77.68
VII	Offset s_v	0.255	2.27	32.61	78.40
VIII	Decoder-LSTM	0.268	2.25	31.41	77.32
IX	Decoder-Tf	0.314	2.34	31.98	78.02
	SketchyNetV2	0.254	2.27	32.77	78.21

Generalisability of canvas selector: To judge if the optimal canvas-size selected is uniform *irrespective* of the training method, we first train *four* canvas-selectors using four SoTAs of Table 2. Evaluating them on ShoeV2 [75], 89.21% of test-set sketches in ShoeV2 to obtain the *same* optimal canvas size across *four* different models, trained on four *different* methods, which further proves that our canvas-selector can work equally well as a pre-processing module for any new model. Furthermore, plugging a canvas-selector trained with SketchyNetV1 (critic) from B-VGG16-SN with a different SketchyNetV1 from StyleVAE [51] during inference achieves comparable performance (35.69% Top-1) to a canvas-selector trained on StyleVAE-SketchyNetV1 (35.88% Top-1) on ShoeV2. Importantly, 95.64% of test-set sketches yield the same optimal canvas-size across both models,

confirming the *cross-model* generalisability of our canvas selector. *(ii)* On training canvas-selector with teacher F_T (for Triplet-SN) as critic, yields *same* optimal canvas-sizes as those for student F_S, for 94.21% of test-set sketches in ShoeV2, while incurring comparable FLOPs reduction (0.268G) and accuracy (27.91% Top1) against F_S as critic.

Comparing Inference times: We evaluated on a 12 GB Nvidia RTX 2080 Ti GPU, on ShoeV2, for inference time of Triplet-SN, HOLEF-SN, Triplet-RL, and StyleVAE (SOTAs) on full-sketch to obtain 37, 38, 36 and 42 ms, against 10, 11, 11, and 12 ms for their SketchyNetV2 variants respectively. For *practical* validation, we evaluated onnx-INT8 version of Triplet-SN [75] and its SketchyNetV2 variant (ours) after post-training quantisation, on an iPhone13 via the AI-benchmark app for ShoeV2 [75] to obtain 51ms and 18ms respectively, which shows our efficiency.

Retraining SoTA on mixed resolution sketches?: To explore if retraining SoTAs on sketches with mixed resolutions, we re-train Triple-SN [75] on rendered mixed-resolution sketches of ShoeV2. Poor performance in Table 4 at lower resolutions similar to Table 2 validates our KD approach. The key lies in that the sketch-feature from teacher used for student-guidance (f_s^T) is always extracted from *full-resolution* sketch while that extracted by student is at varying resolutions, imparting scale invariance, which is ideally suited to our *cross-modal retrieval* setup.

Table 4. Evaluating SoTAs trained on mixed-resolution sketches

Canvas Size	Triplet-SN			Triplet-RL		
	Top1 (%)	Top10 (%)	FLOPs (G)	Top1 (%)	Top10 (%)	FLOPs (G)
32×32	10.91	26.65	0.083	12.97	31.08	0.142
64×64	19.04	48.12	0.338	22.96	56.23	0.577
128×128	26.07	63.87	1.397	30.04	72.95	2.299
256×256	28.74	71.68	5.280	34.21	77.24	6.041

5.4. Further Insights

Cross-dataset Generalisation: To judge the generalisation potential of our canvas-size selector across datasets we train it on ShoeV2, but plug it to a retrieval model (SketchyNetV1) trained on ChairV2, and evaluate on ChairV2. Against the standard result (Table 2) of training both modules (SketchyNetV2) on ChairV2 (0.285G, 55.32% Acc@1), this setup retains accuracy at 53.14% with slight rise in FLOPs (0.314G), thus confirming its potential.

***Faster* Sketch-recognition:** Exploring potential of our canvas-size selector, we prepend it to a standard *sketch-recognition* pipeline [74] to optimise the canvas-size of query sketch, and reduce overall compute. Unlike our ψ_C, it uses negative of cross-entropy loss as accuracy reward (R_{acc}), keeping rest same. Compared to existing Sketch-a-Net [74] incurring 5.28G FLOPs at 68.71% accuracy on Quickdraw [22], our canvas-selector fitted paradigm drops FLOPs to 0.261G while retaining accuracy at 68.14%. This shows for the first time, that sparsity of sketches can be handled, in various downstream tasks via our ψ_C.

Extension to Category-level SBIR: For categorical SBIR we use Sketchy (ext) [36] dataset (90:10 train:test split), which holds $75k$ sketches across 125 categories having $\approx 73k$ images [36] in total. It is evaluated using mean average precision (mAP@all) and precision at top 200 retrieval (P@200) [36]. Here we compete against state-of-the-arts (SoTA) like *B-D2S* that follows [19] without its zero-shot setting), and a simple baseline using a Siamese-style VGG-16 network (*B-SBIR-SN*) following [75]. Similar to Section 5.1 adapting SoTAs as teachers in our *SketchyNetV2* paradigm (T-SoTA) shows them retaining their accuracy (Table 5) while significantly lowering compute.

Table 5. Quantitative Evaluation on Categorical SBIR

	Methods	FLOPs (G)	Params (mil.)	Sketchy(ext) mAP@all	Sketchy(ext) P@200
SoTA	B-SBIR-SN	40.18	14.71	0.715	0.861
	DSH [36]	22.14	16.32	0.711	0.858
	B-D2S [19]	17.75	136.36	0.810	0.894
	StyleVAE [51]	8.116	25.37	0.905	0.927
T-SoTA	T-SBIR-SN	0.254	2.27	0.702	0.810
	T-DSH	0.289		0.704	0.823
	T-D2S	0.268		0.798	0.845
	T-StyleVAE	0.251		0.886	0.897

Teacher-Student Pairs: To explore other teacher-student pairs apart from ours, we test SketchyNetV2 models of a few (F_T, F_S) pairs, where teachers are trained similar to B-VGG16-SN. From Table 6 we see our pair to outperform others.

Table 6. Evaluation on Teacher-Student Pairs

Teacher \ Student	MobileNetV2		EfficientNet		ResNet-18	
	Top-1	FLOPs	Top-1	FLOPs	Top-1	FLOPs
VGG-16 [30]	32.77%	0.254G	29.47%	0.459G	26.72%	1.451G
Inception-V3 [74]	27.89%	0.264G	27.16%	0.461G	25.31%	1.452G

Combining compression methods: To explore if a combination of compression methods is capable of increasing efficiency, we subject our Knowledge Distillation (KD) trained model (SketchyNetV1) of Triplet-SN [75] (Table 2) to *(i)* quantisation following [21], and *(ii)* network-pruning following [38], separately, and then train their respective canvas-selectors. While the former secures 27.34% (*vs*. 27.89% *ours*) Top-1 score with a lower inference time of 9.1 ms (*vs*. 10 ms *ours*), the latter scores 26.91% (*vs*. 27.89% ms *ours*) Top-1 with lower FLOPs of 0.249G (*vs*. 0.264G *ours*). Such competitive results further endorse exploring combining compression methods as future works.

Future Works: Sketch being an abstraction of an object's photo, when resized to a lower scale for retrieval, infers that its object's semantic is interpretable at that scale. This should ideally provide a lower bound for resizing a photo while retaining interpretability (*e.g.* high retrieval score). Consequently, designing a learnable photo-resizer aided by our trained sketch-canvas selector is a potential future work. Secondly, ours being a meta-framework, using more recent frameworks like DINOv2 [41] as large-scale teacher models for further gains is another targeted future work.

6. Conclusion

We for the first time investigate the problem of efficient inference for FG-SBIR, where we show that using existing efficient lightweight networks directly, are unfit for sketches. We thus propose two generic components that can be adapted to photo-networks for sketch-specific data. Firstly, a cross-modal knowledge distillation paradigm distils the knowledge of larger models to a smaller one, directly reducing FLOPs (params) by 97.92 (84.9)%. Secondly, an abstraction-aware canvas-size selector dynamically selects the best canvas-size for a sketch, reducing FLOPs by extra two-thirds. Extensive experiments show our method to perform at par with state-of-the-arts using much lower compute thus further conveying our method's worth in enhancing them towards deployment.

References

[1] Manoj Alwani, Yang Wang, and Vashisht Madhavan. Decore: Deep compression with reinforcement learning. In *CVPR*, 2022. 2

[2] Aneeshan Sain and Ayan Kumar Bhunia and Pinaki Nath Chowdhury and Aneeshan Sain and Subhadeep Koley and Tao Xiang and Yi-Zhe Song. CLIP for All Things Zero-Shot Sketch-Based Image Retrieval, Fine-Grained or Not. In *CVPR*, 2023. 1

[3] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? *NeurIPS*, 2014. 2

[4] Hessam Bagherinezhad, Maxwell Horton, Mohammad Rastegari, and Ali Farhadi. Label refinery: Improving imagenet classification through label progression. *arXiv preprint arXiv:1805.02641*, 2018. 2

[5] Ayan Kumar Bhunia, Ayan Das, Umar Riaz Muhammad, Yongxin Yang, Timothy M Hospedales, Tao Xiang, Yulia Gryaditskaya, and Yi-Zhe Song. Pixelor: A competitive sketching ai agent. so you think you can sketch? *ACM TOG*, 2020. 3

[6] Ayan Kumar Bhunia, Yongxin Yang, Timothy M Hospedales, Tao Xiang, and Yi-Zhe Song. Sketch less for more: On-the-fly fine-grained sketch based image retrieval. In *CVPR*, 2020. 1, 2, 3, 6

[7] Ayan Kumar Bhunia, Pinaki Nath Chowdhury, Aneeshan Sain, Yongxin Yang, Tao Xiang, and Yi-Zhe Song. More photos are all you need: Semi-supervised learning for fine-grained sketch based image retrieval. In *CVPR*, 2021. 1, 2, 3, 4

[8] Ayan Kumar Bhunia, Pinaki Nath Chowdhury, Yongxin Yang, Timothy M Hospedales, Tao Xiang, and Yi-Zhe Song. Vectorization and rasterization: Self-supervised learning for sketch and handwriting. In *CVPR*, 2021. 3

[9] Ayan Kumar Bhunia, Subhadeep Koley, Abdullah Faiz Ur Rahman Khilji, Aneeshan Sain, Pinaki Nath Chowdhury, Tao Xiang, and Yi-Zhe Song. Sketching without worrying: Noise-tolerant sketch-based image retrieval. In *CVPR*, 2022. 1, 2, 3, 7

[10] Ayan Kumar Bhunia, Subhadeep Koley, Amandeep Kumar, Aneeshan Sain, Pinaki Nath Chowdhury, Tao Xiang, and Yi-Zhe Song. Sketch2Saliency: Learning to Detect Salient Objects from Human Drawings. In *CVPR*, 2023. 1

[11] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020. 3

[12] Ying Chen, Liang Qiao, Zhanzhan Cheng, Shiliang Pu, Yi Niu, and Xi Li. Dynamic low-resolution distillation for cost-efficient end-to-end text spotting. In *ECCV*, 2022. 5, 6, 7

[13] Pinaki Nath Chowdhury, Ayan Kumar Bhunia, Viswanatha Reddy Gajjala, Aneeshan Sain, Tao Xiang, and Yi-Zhe Song. Partially does it: Towards scene-level fg-sbir with partial input. In *CVPR*, 2022. 1, 2

[14] Pinaki Nath Chowdhury, Ayan Kumar Bhunia, Viswanatha Reddy Gajjala, Aneeshan Sain, Tao Xiang, and Yi-Zhe Song. Partially does it: Towards scene-level fg-sbir with partial input. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 6

[15] Pinaki Nath Chowdhury, Aneeshan Sain, Ayan Kumar Bhunia, Tao Xiang, Yulia Gryaditskaya, and Yi-Zhe Song. Fs-coco: Towards understanding of freehand sketches of common objects in context. In *ECCV*, 2022. 5, 6

[16] Chowdhury, Pinaki Nath and Bhunia, Ayan Kumar and Sain, Aneeshan and Koley, Subhadeep and Xiang, Tao and Song, Yi-Zhe. Democratising 2D Sketch to 3D Shape Retrieval Through Pivoting. In *ICCV*, 2023. 2

[17] John Collomosse, Tu Bui, and Hailin Jin. Livesketch: Query perturbations for guided sketch-based visual search. In *CVPR*, 2019. 2, 5

[18] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. *NeurIPS*, 28, 2015. 2, 6

[19] Sounak Dey, Pau Riba, Anjan Dutta, Josep Llados, and Yi-Zhe Song. Doodle to search: Practical zero-shot sketch-based image retrieval. In *CVPR*, 2019. 2, 8

[20] Anjan Dutta and Zeynep Akata. Semantically tied paired cycle consistency for zero-shot sketch-based image retrieval. In *CVPR*, 2019. 1

[21] Suyog Gupta, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan. Deep learning with limited numerical precision. In *ICML*, 2015. 2, 6, 8

[22] David Ha and Douglas Eck. A neural representation of sketch drawings. In *ICLR*, 2018. 1, 5, 7, 8

[23] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *ICLR*, 2016. 2

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3

[25] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NeurIPS Deep Learning Workshop*, 2014. 2

[26] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 3

[27] Shouyou Huang and Qiang Wu. Robust pairwise learning with huber loss. *Journal of Complexity*, 2021. 4

[28] Zehao Huang and Naiyan Wang. Like what you like: Knowledge distill via neuron selectivity transfer. *arXiv preprint arXiv:1707.01219*, 2017. 2

[29] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *JAIR*, 1996. 3, 5

[30] Andrew Zisserman Karen Simonyan. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 3, 4, 8

[31] Debang Li, Huikai Wu, Junge Zhang, and Kaiqi Huang. A2-rl: Aesthetics aware reinforcement learning for image cropping. In *CVPR*, 2018. 3

[32] Debang Li, Junge Zhang, and Kaiqi Huang. Learning to learn cropping models for different aspect ratio requirements. In *CVPR*, 2020. 6

[33] Xiaodan Liang, Lisa Lee, and Eric P Xing. Deep variation-structured reinforcement learning for visual relationship and attribute detection. In *CVPR*, 2017. 3

[34] Darryl Lin, Sachin Talathi, and Sreekanth Annapureddy. Fixed point quantization of deep convolutional networks. In *ICML*, 2016. 2

[35] Hangyu Lin, Yanwei Fu, Yu-Gang Jiang, and Xiangyang Xue. Sketch-bert: Learning sketch bidirectional encoder representation from transformers by self-supervised learning of sketch gestalt. In *CVPR*, 2020. 5

[36] Li Liu, Fumin Shen, Yuming Shen, Xianglong Liu, and Ling Shao. Deep sketch hashing: Fast free-hand sketch-based image retrieval. In *CVPR*, 2017. 2, 8

[37] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *ICCV*, 2017. 2, 6

[38] Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*, 2016. 6, 8

[39] Umar Riaz Muhammad, Yongxin Yang, Yi-Zhe Song, Tao Xiang, and Timothy M Hospedales. Learning deep sketch abstraction. In *CVPR*, 2018. 3

[40] Umar Riaz Muhammad, Yongxin Yang, Timothy Hospedales, Tao Xiang, and Yi-Zhe Song. Goal-driven sequential data abstraction. In *ICCV*, 2019. 1, 3

[41] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning Robust Visual Features without Supervision. *arXiv preprint arXiv:2304.07193*, 2023. 8

[42] Kaiyue Pang, Yi-Zhe Song, Tony Xiang, and Timothy M Hospedales. Cross-domain generative learning for fine-grained sketch-based image retrieval. In *BMVC*, 2017. 2

[43] Kaiyue Pang, Ke Li, Yongxin Yang, Honggang Zhang, Timothy M Hospedales, Tao Xiang, and Yi-Zhe Song. Generalising fine-grained sketch-based image retrieval. In *CVPR*, 2019. 1

[44] Kaiyue Pang, Yongxin Yang, Timothy M Hospedales, Tao Xiang, and Yi-Zhe Song. Solving mixed-modal jigsaw puzzle for fine-grained sketch-based image retrieval. In *CVPR*, 2020. 2, 3

[45] Nikolaos Passalis and Anastasios Tefas. Learning deep representations with probabilistic knowledge transfer. In *ECCV*, 2018. 6

[46] Pinaki Nath Chowdhury and Ayan Kumar Bhunia and Aneeshan Sain and Subhadeep Koley and Tao Xiang and Yi-Zhe Song. SceneTrilogy: On Human Scene-Sketch and its Complementarity with Photo and Text. In *CVPR*, 2023. 2

[47] Pinaki Nath Chowdhury and Ayan Kumar Bhunia and Aneeshan Sain and Subhadeep Koley and Tao Xiang and Yi-Zhe Song. What Can Human Sketches Do for Object Detection? In *CVPR*, 2023. 1

[48] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antonie Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *ICLR*, 2015. 2, 4, 6

[49] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 4

[50] Aneeshan Sain, Ayan Kumar Bhunia, Yongxin Yang, Tao Xiang, and Yi-Zhe Song. Cross-modal hierarchical modelling forfine-grained sketch based image retrieval. In *BMVC*, 2020. 2, 3, 4

[51] Aneeshan Sain, Ayan Kumar Bhunia, Yongxin Yang, Tao Xiang, and Yi-Zhe Song. Stylemeup: Towards style-agnostic sketch-based image retrieval. In *CVPR*, 2021. 1, 2, 3, 4, 6, 7, 8

[52] Aneeshan Sain, Ayan Kumar Bhunia, Vaishnav Potlapalli, Pinaki Nath Chowdhury, Tao Xiang, and Yi-Zhe Song. Sketch3t: Test-time training for zero-shot sbir. In *CVPR*, 2022. 2

[53] Aneeshan Sain, Ayan Kumar Bhunia, Subhadeep Koley, Pinaki Nath Chowdhury, Soumitri Chattopadhyay, Tao Xiang, and Yi-Zhe Song. Exploiting Unlabelled Photos for Stronger Fine-Grained SBIR. In *CVPR*, 2023. 1, 2

[54] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018. 1, 3, 4, 5, 7

[55] Patsorn Sangkloy, Nathan Burnell, Cusuh Ham, and James Hays. The sketchy database: learning to retrieve badly drawn bunnies. *ACM TOG*, 2016. 5, 6

[56] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 5

[57] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 1, 3

[58] Jifei Song, Yi-Zhe Song, Tony Xiang, and Timothy M Hospedales. Fine-grained image retrieval: the text/sketch input dilemma. In *BMVC*, 2017. 2

[59] Jifei Song, Qian Yu, Yi-Zhe Song, Tao Xiang, and Timothy M Hospedales. Deep spatial-semantic attention for fine-grained sketch-based image retrieval. In *ICCV*, 2017. 2, 6

[60] Subhadeep Koley and Ayan Kumar Bhunia and Aneeshan Sain and Pinaki Nath Chowdhury and Tao Xiang and Yi-Zhe Song. Picture that Sketch: Photorealistic Image Generation from Abstract Sketches. In *CVPR*, 2023. 1

[61] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *NeurIPS*, 2000. 5

[62] Hossein Talebi and Peyman Milanfar. Learning to resize images for computer vision tasks. In *ICCV*, 2021. 2, 6

[63] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019. 1, 3

[64] Burak Uzkent and Stefano Ermon. Learning when and where to zoom with deep reinforcement learning. In *CVPR*, 2020. 2

[65] Supawit Vatathanavaro, Suchat Tungjitnob, and Kitsuchart Pasupa. White blood cell classification: a comparison between vgg-16 and resnet-50 models. In *JSCI*, 2018. 3

[66] Mahes Visvalingam and J Duncan Whyatt. The douglas-peucker algorithm for line simplification: re-evaluation through visualization. In *Computer Graphics Forum*, 1990. 5

[67] Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *CVPR*, 2019. 3

[68] Yunhe Wang, Chang Xu, Shan You, Dacheng Tao, and Chao Xu. Cnnpack: Packing convolutional neural networks in the frequency domain. *NeurIPS*, 29, 2016. 2

[69] Yulin Wang, Kangchen Lv, Rui Huang, Shiji Song, Le Yang, and Gao Huang. Glance and focus: a dynamic approach to reducing spatial redundancy in image classification. In *NeurIPS*, 2020. 2

[70] Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *JMLR*, 2009. 4

[71] Peng Xu, Chaitanya K Joshi, and Xavier Bresson. Multigraph transformer for free-hand sketch recognition. *IEEE T-NNLS*, 2021. 4

[72] Peng Xu, Timothy M Hospedales, Qiyue Yin, Yi-Zhe Song, Tao Xiang, and Liang Wang. Deep learning for free-hand sketch: A survey. *IEEE TPAMI*, 2022. 1, 2

[73] Ran Xu, Fangzhou Mu, Jayoung Lee, Preeti Mukherjee, Somali Chaterji, Saurabh Bagchi, and Yin Li. SmartAdapt: Multi-branch object detection framework for videos on mobiles. In *CVPR*, 2022. 2

[74] Qian Yu, Yongxin Yang, Yi-Zhe Song, Tao Xiang, and Timothy M. Hospedales. Sketch-a-net that beats humans. In *BMVC*, 2015. 6, 8

[75] Qian Yu, Feng Liu, Yi-Zhe Song, Tao Xiang, Timothy M Hospedales, and Chen-Change Loy. Sketch me that shoe. In *CVPR*, 2016. 1, 2, 3, 4, 5, 6, 7, 8

[76] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017. 2, 6

[77] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016. 2

[78] Mingjian Zhu, Kai Han, Enhua Wu, Qiulin Zhang, Ying Nie, Zhenzhong Lan, and Yunhe Wang. Dynamic resolution network. In *NeurIPS*, 2021. 7