UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Cross Domain Visual Search with Feature Learning using Multi-stream Transformer-based Architectures

Leo Sampaio Ferraz Ribeiro

Tese de Doutorado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

ICMC USP SÃO CARLOS

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Leo Sampaio Ferraz Ribeiro

Cross Domain Visual Search with Feature Learning using Multi-stream Transformer-based Architectures

Doctoral dissertation submitted to the Instituto de Ciências Matemáticas e de Computação – ICMC-USP, in partial fulfillment of the requirements for the degree of the Doctorate Program in Computer Science and Computational Mathematics. *EXAMINATION BOARD PRESENTATION COPY*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. Moacir Antonelli Ponti

USP – São Carlos
January 2023

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

S484c

Sampaio Ferraz Ribeiro, Leo
Cross Domain Visual Search with Feature Learning using Multi-stream Transformer-based Architectures / Leo Sampaio Ferraz Ribeiro; orientador Moacir Antonelli Ponti. -- São Carlos, 2023.
198 p.

Tese (Doutorado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2023.

1. Sketch-based Image Retrieval. 2. Cross-domain Representation Learning. 3. Feature Learning. 4. Transformer. 5. Graph Neural Network. I. Antonelli Ponti, Moacir, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Leo Sampaio Ferraz Ribeiro

Busca entre Domínios Visuais com Aprendizado de Características através de Arquiteturas Baseadas em Transformers

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Doutora em Ciências – Ciências de Computação e Matemática Computacional. *EXEMPLAR DE DEFESA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Supervisor: Prof. Dr. Moacir Antonelli Ponti

USP – São Carlos
Janeiro de 2023

ACKNOWLEDGEMENTS

I would like to thank my supervisor for this Ph.D., Professor Moacir Ponti. Moacir was a teacher in more ways than one: in the literal sense for many lectures, as my academic supervisor for all my undergraduate and graduate research projects and as a guide and friend while I figured out how to grow into a better researcher and person. He first taught me what "Machine Learning" meant 8 years ago, came up with the idea to explore sketch-based tasks a year later and was the person to take a random piece of paper and sketch me a Convolutional Neural Network when I did not know what was this "CNN" that everyone in the lab was talking about. For being with me at every step of the way and doing it all with unrelentless kindness, I thank you.

I would also like to thank long time research collaborators and friends, Professor John Collomosse and Tu Bui, with whom me and Moacir made many research breaktroughts and who also made me feel as welcome in the University of Surrey as I do at the Instituto de Ciências de Computação e Matemática (ICMC/USP). At ICMC, I am thankful for all of those whose zealous work made my research and the research of countless others possible, specially Leonardo, Fernanda, Lycaena and Gustavo.

Because the work is never separate from who we are, I would also like to thank my family; my parents and siblings, for their undying support and love, and my girlfriend Lais Miranda, who is my safe haven and my partner-for-life and who wished this section mentioned her as my fiancé but in her words, her "idea for a proposal is too grand and it must be done this way, so we'll be girlfriends a little longer'. I extend these thanks also to my friends, who lovingly listened to me going on about random specific interests and complaining about academia and the patriarchy: Alex, Aline, Ana, Anayã, André, Damares, Eric, Fabio, Fernando, Gabriel, Gabriela, Gil, Helena, João, Lucy, Luisa, Luna, Matheus, Sherlon, Tobias, Tommy, Tu, Violeta, Yusuf.

This work was financially supported in part by Fundação de Amparo à Pesquisa do Estado de São Paulo (grants 2017/22366-8, 2019/02808-1 and 2018/22482-0) and charitable donations from Adobe Inc.

“We step out of our solar system into the universe
seeking only peace and friendship
– to teach, if we are called upon;
to be taught, if we are fortunate.”
(Becky Chambers)

RESUMO

RIBEIRO, L. S. F. **Busca entre Domínios Visuais com Aprendizado de Características através de Arquiteturas Baseadas em Transformers**. 2023. 197 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2023.

Dentro do campo de Visão Computacional, a tarefa de Busca Visual entre Domínios é uma das mais úteis e estudadas e ainda assim raramente experimentadas em nosso cotidiano. Nesta tese exploramos a Busca Visual entre Domínios, usando a tarefa de Busca de Imagens baseada em Desenhos (*Sketch-based Image Retrieval*, SBIR) como sua representante. Descrevemos quatro hipóteses distintas sobre como avançar esse campo de pesquisa e comprovamos a validade de cada uma através das contribuições explicitadas. Primeiro apresentamos uma nova arquitetura, chamada Sketchformer, para o aprendizado de representações de desenhos, que troca as redes convolucionais tradicionais pelo recente modelo Transformer. Então exploramos duas definições alternativas para a tarefa de SBIR, cada uma aproximando melhor a escala e generalização necessárias para implementação no mundo real. Para ambas tarefas introduzimos modelos do estado-da-arte: nosso Scene Designer combina tradicionais redes de múltiplos ramos com uma rede neural em grafo para aprender representações de cenas desenhadas com múltiplos objetos; nosso Sketch-an-Anchor mostra que é possível usar o conhecimento geral que existe em modelos pré-treinados para a tarefa de Zero-shot SBIR. Estas contribuições tem um impacto direto na literatura de tarefas baseadas em desenhos e um impacto em cascata para as áreas de Compreensão de Imagens e Representações Entre Domínios em geral.

Palavras-chave: Busca de Imagens Baseada em Desenhos, Aprendizado de Representações entre Domínios, Aprendizado de Representações, Transformer, Redes Neurais em Grafos.

ABSTRACT

RIBEIRO, L. S. F. **Cross Domain Visual Search with Feature Learning using Multi-stream Transformer-based Architectures**. 2023. 197 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2023.

Within the general field of Computer Vision, the task of Cross-domain Visual Search is one of the most useful and studied and yet it is rarely seen throughout our daily lives. In this thesis we explore Cross-domain Visual Search using the specific and mature Sketch-based Image Retrieval (SBIR) task as a canvas. We draw four distinct hypothesis as to how to further the field and demonstrate their validity with each contribution. First we present a new architecture for sketch representation learning that forgoes traditional Convolutional networks in favour of the recent Transformer design, called Sketchformer. Then we explore two alternative definitions for the SBIR task that each approach the scale and generalisation necessary for implementation in the real world. For both tasks we introduce state-of-the-art models: our Scene Designer combines traditional multi-stream networks with a Graph Neural Network to learn representations for sketched scenes with multiple object; our Sketch-an-Anchor shows that it is possible to harvest general knowledge from pre-trained models for the Zero-shot SBIR task. These contributions have a direct impact on the literature of sketch-based tasks and a cascaded impact on Image Undestanding and Cross-domain representations at large.

Keywords: Sketch-based Image Retrieval, Cross-domain Representation Learning, Feature Learning, Transformer, Graph Neural Network.

LIST OF FIGURES

Figure 1 – Sketch-based Image Retrieval 22
Figure 2 – Category-level SBIR versus fine-grain SBIR 30
Figure 3 – Autoencoder diagram 31
Figure 4 – Generative Models Categorisation 33
Figure 5 – GAN Training Diagram 35
Figure 6 – Feature learning with triplet loss 38
Figure 7 – Attention Mechanism 41
Figure 8 – Transformer Architecture 42
Figure 9 – ViT Architecture 43
Figure 10 – Comparing different sketch encodings 52
Figure 11 – Visualisation of the impact of quantisation on sketch reconstruction 53
Figure 12 – Diagram of the Sketchformer architecture 54
Figure 13 – Visualisation of Sketchformer's learned embedding 56
Figure 14 – Schematic of Sketchformer's multi-task representation 58
Figure 15 – Comparison of sketch reconstruction results 60
Figure 16 – Comparison of representative sketch interpolations 61
Figure 17 – Visualisation of sketches reconstructed from mean embedding 62
Figure 18 – Visual search results over the Stock10M dataset 63
Figure 19 – Quantifying search accuracy 64
Figure 20 – Scene Designer applied to iterative composition design 69
Figure 21 – Diagram of Scene Designer's architecture 73
Figure 22 – Grid-based Positional Encoding 77
Figure 23 – Synthesised negative samples for contrastive loss 78
Figure 24 – Sample paired data from the proposed QuickDrawCOCO-92c dataset 83
Figure 25 – Visual retrieval results using the SketchyCOCO and QuickDrawCOCO sets 85
Figure 26 – Qualitative comparison against SceneSketcher 86
Figure 27 – Comparison between generated semantic layouts 88
Figure 28 – Sensitivity to object pose and combination of objects from both domains . . 88
Figure 29 – Comparing pre-trained models before and after Sketch-an-Anchor training . 97
Figure 30 – Compare embedding before and after training with Sketch-an-Anchor . . . 98
Figure 31 – Diagram for the Sketch-an-Anchor architecture 100
Figure 32 – Diagram depicting how the Adapted Semantic Anchors are learned 105
Figure 33 – Evaluation on the Sketchy validation set across 10 thousand iterations . . . 108

Figure 34 – Visual ZSSBIR results with the Sketchy dataset 109
Figure 35 – Visual ZSSBIR results with the Tu-Berlin dataset 109
Figure 36 – Experiment with Image Selection 111

LIST OF TABLES

Table 1 – Sketch classification results over QuickDraw 59
Table 2 – User study quantifying accuracy of sketch reconstruction 60
Table 3 – User study quantifying interpolation quality for a pair of sketches 62
Table 4 – Quantifying the performance of Sketch2Sketch retrieval 64
Table 5 – Quantifying accuracy of Sketchformer for Sketch2Image search 65
Table 6 – Scene Designer layers' hyperparameters 81
Table 7 – Comparing existing datasets against our QuickdrawCOCO 84
Table 8 – Recall@K metrics on the SketchyCOCO dataset 86
Table 9 – Recall@K metrics on the QuickDrawCOCO-92c set 87
Table 10 – MTurk sketch search results on the SketchyCOCO dataset 87
Table 11 – Generation metrics when using samples from SketchyCOCO and QuickDrawCOCO-92c 89
Table 12 – Ablation Studies, showing Recall@k and FID on QuickDrawCOCO and SketchyCOCO 89
Table 13 – Comparing backbones for the encoders 91
Table 14 – Investigating Scene Designer's use of GNN and Transformer architectures with the QuickDrawCOCO set 91
Table 15 – Investigating Scene Designer's use of GNN and Transformer architectures with the SketchyCOCO set 92
Table 16 – Comparing Word and Visual Semantic Anchors 103
Table 17 – Comparison against baselines at the ZSSBIR and Binarized ZSSBIR tasks . . 115
Table 18 – Comparison of Sketch-an-Anchor and baselines on the Generalized ZSSBIR task 116
Table 19 – ZSSBIR Performance across architectures and pre-training protocols 116
Table 20 – Evaluation of ablated models on the Tu-Berlin and Sketchy N.O. 116
Table 21 – List of published and submitted articles co-authored by the candidate within the Ph.D. program. Listed in chronological order. 140

CONTENTS

1 INTRODUCTION 21
1.1 Related Work and Context 22
1.2 Contributions and Hypothesis 25
1.3 Outline 26

2 TECHNICAL BACKGROUND 29
2.1 Sketch-based Image Retrieval 29
2.2 Autoencoders 31
2.3 Generative Models 32
2.4 Generative Adversarial Networks 34
2.4.1 Generator/Discriminator *34*
2.4.2 Losses and Training *35*
2.5 Multistream Networks 36
2.5.1 Siamese Networks with Contrastive Loss *36*
2.5.2 Triplet Networks with Triplet Loss *37*
2.5.3 Self-Supervised Learning *38*
2.6 The Transformer Architecture 40
2.7 Graph Neural Networks 44
2.8 Few-shot and Zero-shot Learning 45

3 SKETCHFORMER: TRANSFORMER-BASED REPRESENTATION FOR SKETCHED STRUCTURE 47
3.1 Introduction 48
3.2 Related Work 49
3.3 Sketch Representation 50
3.3.1 Pre-processing and tokenisation *50*
3.3.2 Transformer Architecture for Sketch *52*
3.3.2.1 Training Losses *55*
3.3.3 Cross-modal Search Embedding *56*
3.4 Experiments and Discussion 58
3.4.1 Evaluating Sketch Classification *59*
3.4.2 Reconstruction and Interpolation *60*
3.4.3 Cross-modal Matching *62*
3.5 Conclusion 65

4 SCENE DESIGNER: COMPOSITIONAL SKETCH-BASED IMAGE RETRIEVAL . . . 67
4.1 Introduction . . . 68
4.2 Related Work . . . 70
4.3 Methodology . . . 72
4.3.1 Object-level Representation (OLR) . . . *74*
4.3.2 Scene-aware Object-level Representation (SA-OLR) . . . *74*
4.3.3 Scene Representation (SR) . . . *75*
4.3.4 Compositional Sketch-based Image Retrieval . . . *77*
4.3.5 Semantic Scene Layout Generation . . . *78*
4.3.6 Training Stages . . . *80*
4.3.7 Implementation Details . . . *80*
4.4 Experiments and Discussion . . . 81
4.4.1 Datasets . . . *82*
4.4.2 Evaluating Retrieval . . . *84*
4.4.3 Evaluating Layout Synthesis . . . *86*
4.4.4 Ablation studies . . . *89*
4.5 Design Choices . . . 90
4.6 Conclusion . . . 93
4.7 Data Availability Statement . . . 93

5 SKETCH-AN-ANCHOR: ZERO-SHOT SKETCH-BASED IMAGE RETRIEVAL . . . 95
5.1 Introduction . . . 96
5.2 Related Work . . . 99
5.3 The Sketch-an-Anchor Method . . . 100
5.3.1 Problem Setting . . . *101*
5.3.2 A Strong Baseline . . . *101*
5.3.3 Semantic Anchors . . . *102*
5.3.4 Anchored Contrastive Loss . . . *104*
5.3.5 Adapted Semantic Anchors . . . *104*
5.3.6 Randomised Input Semantic Anchors . . . *105*
5.4 Experiments . . . 106
5.4.1 Datasets . . . *106*
5.4.2 Implementation Details . . . *106*
5.4.3 A Thousand Iterations are All You Need . . . *107*
5.4.4 Comparison Against Other Methods . . . *107*
5.4.5 An Image is Worth an Entire Class . . . *110*
5.4.6 Correlation between Initial Image-to-Image Retrieval performance and Sketch-an-Anchor's . . . *111*

5.4.7 Ablation Study 113
5.5 Conclusion 113

6 CONCLUSION 117

BIBLIOGRAPHY 121

APPENDIX A PUBLICATIONS 139

ANNEX A ORIGINAL PUBLICATIONS 141
A.1 Sketchformer: Transformer-based representation for sketched structure. 141
A.2 Scene designer: a unified model for scene search and synthesis from sketch. 152
A.3 Scene designer: Compositional sketch-based image retrieval with contrastive learning and an auxiliary synthesis task. 163
A.4 Sketch-an-anchor: Sub-epoch fast model adaptation for zero-shot sketch-based image retrieval. 187

CHAPTER

1

INTRODUCTION

The impact of "Artificial Intelligence" applications in our daily lives is undeniable. Even humanity's creative endeavours are not safe from the A.I Revolution (REESE, 2022), with systems such as the commercial ChatGPT (BROWN *et al.*, 2020) able to write convincing school essays and the Open-source Stable Diffusion (ROMBACH *et al.*, 2021) able to generate pleasing imagery that looks like real artwork. While these may be headline-worthy advancements, they target mimicking human behaviour before being assistive; automation before augmentation of our capabilities.

With collaboration and augmentation in mind, this thesis instead turns its attention to *Cross-domain Search*, a Computer Vision application that receives generous attention in research yet is seldom applied in the real world. Search is an essential application of computing, with billions of queries processed daily. Search engines have however been mostly limited to text queries, which are not enough given the abundant amount of media created and uploaded daily. Investing.com reported (SHVARTSMAN, 2022) that 2.8 and 17 billion new images are shared monthly on Meta's Instagram and Messenger respectively. Text queries alone are not able to fully capture user intent when the searched media is a photo, a video or a song.

As humans, one of our most useful cognitive capacities is our ability express and understand semantic meaning throughout distinct domains. We can for example identify a "the idea of a dog" within a sketch, painting, sounds and even smells. This allows humans to distinguish different objects and ideas regardless of domain and to attribute similarity between them. We are not however able to employ this ability at scale. Our goal is then to mimic this capability, augmented to thousands or millions of samples; models capable of contrasting ideas semantically across domains, applied to large-scale image retrieval. Cross-domain retrieval is a Computer Vision task that lies both within the fields of Image Understanding and Feature Learning and comes with its own challenges.

In this thesis, we refer to "a domain" as the means of generation of a given image; some

common domains throughout this document include “natural image”, “sketch” and “painting”. Of those, we have a particular interest on the relationship between natural images and sketches, within the sub-field of sketch-based image retrieval (SBIR), chosen for three reasons: (1) its rich literature background, (2) readily availability of benchmarks and datasets when compared to other counterparts and (3) its real-world applicability front the ubiquity of touch-sensitive devices.

The setup for an SBIR system is simple: the end user provides a sketch as an input query and the system should be able to present related natural images (e.g. if one draws a plane the system must return photographs of planes). The literature can then be broken down by how sketches are represented (rasterised pixel matrices vs. vector sequences), limitations on the task (single object vs. multi-object sketches) and large scale applicability (sample or category-wise scale). An schematic for a simple SBIR system can be seen in Figure 1.

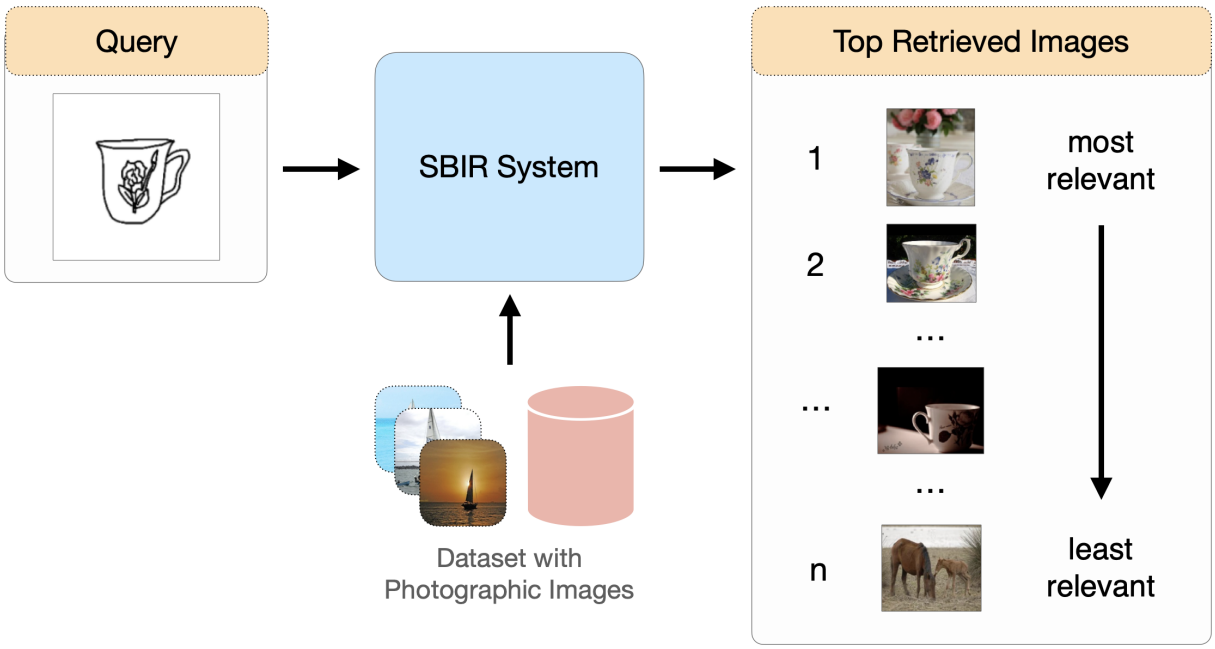


Figure 1 – Basic depiction of a general SBIR system. In this example the query is a drawing of a teacup and as such, the retrieval result ranks semantically and visually similar images higher than unrelated images within the natural images dataset.

1.1 Related Work and Context

In this section we contextualise the studies presented in this thesis within the greater corpus of Sketch-based Image Retrieval literature. Each of the chapters describing our contributions (Chapters 3, 4 and 5) will better explore specific sections of the literature that regard the presented tasks.

The literature for SBIR has for the most part reflected advancements in computer vision. From the seminal work of Ashley *et al.* (1995) up until mid-2010s (EITZ *et al.*, 2009; DALAL;

TRIGGS, 2005; HU; BARNARD; COLLOMOSSE, 2010; HU; COLLOMOSSE, 2013; EITZ *et al.*, 2011; QI *et al.*, 2015; JAMES; FONSECA; COLLOMOSSE, 2014; BUI; COLLOMOSSE, 2015), hand-crafted feature extraction methods were the norm in SBIR research. Ever more complex frameworks of extraction were built with domain knowledge and common-at-the-time methods for classification and retrieval, such as Bag of Features and Spatial Pyramid Matching (LAZEBNIK; SCHMID; PONCE, 2006; NOWAK; JURIE; TRIGGS, 2006). The shortcomings of those ideas became apparent as the literature evolved: the feature set used to represent images and sketches was being chosen by hand using our understanding and prejudiced views regarding representation. Worse still, each task (such as classification or retrieval) and even each dataset required a different set of features. The studies of Saavedra and Bustos (2014), Saavedra and Barrios (2015) started a new trend in the field. Their feature extractors were first designed to identify known "keyshapes" in images, the presence of ellipses, arcs and lines. The follow up study then made use of a sketch dataset to *learn the "keyshapes" from data*, moving away from assumptions and letting the model learn some of the representation by itself.

Again following computer vision, the sketch literature benefited from the "deep learning boom". The retrieval task is very dependent on semantic representations. This aspect was particularly benefited by the deep learning paradigm shift as an entirely new field was created: Feature Learning (term defined by Goodfellow, Bengio and Courville (2016)). To feature learn is to project data into feature spaces, automatically choosing those as to better represent a specific data distribution in a way that solves a given task. In summary: to perform data-driven feature extraction.

While deep convolutional neural networks (CNNs) have led to progress leaps in SBIR, we show on this thesis that definitions within the task should be re-examined before these systems are deployed in real scenarios. Each of the publications we describe in Chapters 3, 4 and 5 explores a different way to interpret SBIR and other sketch-based tasks. In the next paragraphs we will place each of these studies within the recent history of the literature.

The revival of CNNs together with deep learning impacted Computer Vision as a whole. Within SBIR, success was found in the use of one specific variation of CNNs: multi-stream networks. This family of methods implements composition of two or more networks into branching paths/streams, where each stream performs its computation independently and the final result can be combined in a loss function or used separate for applications. In SBIR, multi-stream networks are usually combined with metric learning. Metric learning losses are sample-distance-based functions designed to guide training of domain-specific network branches towards the projection of data into common representations. Data projected into these spaces is then used with ease for visual search with a nearest-neighbours approach.

While the first use of deep learning for sketch was for classification (YU *et al.*, 2015), work on SBIR soon followed with Wang, Kang and Li (2015) showing the potential for sketch-based 3D shape retrieval and Qi *et al.* (2016). Both solutions implemented a multi-stream

siamese network (one stream per domain) trained with the contrastive loss (covered in Chapter 2). The literature then followed the general trends in metric learning designs. The FaceNet model (SCHROFF; KALENICHENKO; PHILBIN, 2015) introduced the triplet network and loss to much celebration; their model was designed to learn representations for faces to allow semantic comparisons and matching. This new metric learning design was quickly adopted for SBIR, first by Yu *et al.* (2016) and Sangkloy *et al.* (2016a). Both studies are also the first to change the constraints of SBIR and introduce "fine-grain" retrieval.

In the classical formulation, the task of SBIR is to match the category of the query sketch in the retrieved images; we now call this the "coarse-grain" or category-level formulation. Yu *et al.* (2016) and Sangkloy *et al.* (2016a) designed their models to retrieve exact matches for their queries. The Sketchy dataset used by them had users draw precise sketches after looking at a given photograph for a few seconds.

Exploration on how to best use metric learning was the focus of the literature for a few years. Bui *et al.* (2017) introduced the triplet network to category-level SBIR. A follow up work (BUI *et al.*, 2018a; BUI *et al.*, 2016) discussed using photographic images without common pre-processing (edgemap extraction) and the use of contrastive and triplet losses in different training stages. The work of (SEDDATI; DUPONT; MAHOUDI, 2017) introduced a quadruplet loss, combining category-level and fine-grain annotations to train a robust model. The work of Bui *et al.* (2018b) then defined a "mid-grain" annotation, defining relevance as both matched categories and object pose.

The studies presented on this thesis start in this context. SBIR is a mature field in the literature with most studies evaluating the constraints of the task, how to better apply these systems in the real world and how to design new architectures and losses to suit those setups. As was shown, multi-stream networks combined with metric learning are the norm to this day, and we follow suit with our contributions.

All of the studies up to this point have used rasterised sketches as input. This is expected as most of the improvements brought by deep models are computer vision improvements. Transforming sketches to pixel matrices is a natural way of approximating them to the photographic domain and allows for shared architectures between domains. It is not the only approach however. Sketches are originally captured in vector form: a sequence of points in a 2D plane. Recurrent neural networks (SCHUSTER; PALIWAL, 1997) and Long-short Term Memory networks (HOCHREITER; SCHMIDHUBER, 1997) were also revived together with other neural network designs thanks to cheaper hardware and renewed interest from the deep learning community. While most RNN-based models were being applied to Natural Language Processing (HUANG; XU; YU, 2015), the work of Ha and Eck (2018) introduced SketchRNN and used an LSTM architecture to encode sketches in vector form, learning a semantic representation that could be used to mix and generate novel sketches.

More studies in the literature followed SketchRNN, using it as basis to encode vector

sketches for sketch-based sketch retrieval (XU *et al.*, 2018), sketch classification (LI *et al.*, 2018) and of course SBIR (COLLOMOSSE; BUI; JIN, 2019). Our first contribution follows this trend. The Transformer architecture (better explained in Chapter 2) was introduced by Vaswani *et al.* (2017) and quickly supplanted LSTMs as the norm in natural language processing. With Sketchformer (Chapter 3) we have designed the first Transformer model for vector sketches and show that the same learned representation can be used for sketch classification, generation and SBIR.

This thesis then follows the literature in re-evaluating how the SBIR task can be better defined to be applicable on real world scenarios: e-commerce, art reference search and large scale image retrieval. We have found and followed two paths in this regard, (i) single-object search is a constraint that does not match user expectations for retrieval and (ii) these systems must be able to retrieve concepts beyond what is seen in training as training sets will never cover all image content produced.

Starting with (i), the work of Gao *et al.* (2020) was the first to introduce the concept of Sketch Scenes or Sketch Compositions to computer vision. Their model takes in sketched scenes with multiple objects and synthesises a photo-realistic image from the reference. Then Liu *et al.* (2020) used the sketched scenes dataset they introduced to design the first Compositional SBIR model, where the sketch queries and retrieved images are composed of multiple objects as well. In this context we introduced Scene Designer (Chapter 4), a model that combines the image retrieval and synthesis task to achieve the state-of-the-art in Compositional SBIR.

Regarding the problem of scale (ii), (SHEN *et al.*, 2018) introduced the concept of Zero-Shot SBIR (ZSSBIR). This new task follows the Zero-shot training paradigm (XIAN *et al.*, 2018), where the categories used for training are disjointed with the categories used for testing, better simulating the real world scenario and its "unlimited categories". ZSSBIR quickly became a hot-topic in the sketch literature, with models matching complex word semantics (DUTTA; AKATA, 2020; ZHANG *et al.*, 2020; DEY *et al.*, 2019a; CHAUDHURI *et al.*, 2020; DENG *et al.*, 2020), trying to generate objects unseen in training (DUTTA; SINGH; BISWAS, 2021; SHEN *et al.*, 2018; YELAMARTHI *et al.*, 2018a; PANDEY *et al.*, 2020) and preserving knowledge from pre-trained models (WANG *et al.*, 2022; TURSUN *et al.*, 2022; WANG *et al.*, 2021; TIAN *et al.*, 2022; LIU *et al.*, 2019). We introduce Sketch-an-Anchor in Chapter 5, a model that can be trained in under an epoch and yet perform at the state-of-the-art for ZSSBIR.

1.2 Contributions and Hypothesis

The main goal of this thesis is to advance the field of cross-domain representation learning as a whole. We do this by improving the task of Sketch-based Image Retrieval and its variants. Our work comprised three main contributions, each connected to a published or submitted study described in Chapters 3, 4 and 5:

C1. Sketchformer, the first Transformer-based architecture designed for vector sketches. We show the model's learned representation can be used for classification, synthesis and retrieval.

C2. Scene Designer, the state-of-the-art in Compositional SBIR. This model uses a graph interpretation of composite scenes and combines the retrieval task with image synthesis to yield better scene representations.

C3. Sketch-an-Anchor, the fast converging, state-of-the-art model for Zero-shot SBIR. We show that it is possible to preserve the general visual semantic sub-space learned by pre-trained models and map sketches to a similar sub-space.

These contributions have a direct relationship to our set of hypothesis; those four hypothesis are the guiding framework of this thesis and are listed below:

H1. The success of the attention-based Transformer **architecture** in natural language processing can be transferred to vector sketch representations;

H2. It is possible to learn **semantic representations for Sketch Scenes** by combining classic single-object encoding with a graph representation for scenes;

H3. The **generaliseable knowledge** of models trained on large scale photographic image sets can be transferred to the sketch domain;

H4. Representation learning models benefit from **combining metric learning with generative tasks**;

Each of Hypothesis **H1**, **H2** and **H3** directly drive each of our main contributions. The last Hypothesis, **H4**, is one of the key findings presented in both Chapters 3 and 4.

1.3 Outline

This document includes, besides this Introduction, chapters regarding the necessary Technical Background, one chapter for each main contribution and one with concluding thoughts. In Chapter 2 (Technical Background), we will scrutinise the main theoretical framework that will be used through the document and will detail less-known jargon and technical terms. Our contributions are presented in the three chapters that follow. We have included revised and expanded versions of the studies previously published or submitted for publication. Chapter 3 presents Sketchformer, an attention-based architecture for vector sketches designed to learn representations useful for multiple sketch tasks. Chapter 4 describes Scene Designer, a model that merges the strength of classic multi-stream learning with scene graphs and shows the potential

of combining retrieval and synthesis tasks to improve the learned representations. Chapter 5 then shows our Sketch-an-Anchor, a ZSSBIR model that converges in under an epoch by relying on the existing knowledge captured by pre-trained image models. Finally, on Chapter 6 we lay our final set of considerations.

CHAPTER

2

TECHNICAL BACKGROUND

This chapter introduces supporting ideas for the thesis, from methodologies, definitions and model architectures. Aside from Sketch-based Image Retrieval, we do not detail other cross-domain visual search systems because, for the best of our knowledge, SBIR is the most well defined and studied field of visual search within the literature. We also do not give details on Deep Learning and CNNs given their ubiquity and the existence of better sources such as the Deep Learning book by Goodfellow, Bengio and Courville (2016) the surveys of Ponti *et al.* (2017) and Ponti *et al.* (2021).

2.1 Sketch-based Image Retrieval

We have chosen SBIR as our base problem for its extensive literature coverage. We highlight again Figure 1 found on Chapter 1 as the expected outcome of an SBIR system. A good SBIR model should rank most relevant images first while gradually ranking remaining images according to their semantic relevance. The creation and evaluation of a ranking system depends first on the definition of what is relevant.

As aforementioned, classic SBIR follows two main definitions for relevancy. Category-level (or "coarse") SBIR considers relevance to be category-wide, which means that any image from the same category is considered relevant without regards to its positioning, shape and context. Fine-grain SBIR considers that there is a one-to-one mapping between domains and the retrieved results should include the "correct" match match among the first ranked images. In Figure 2 we exemplify the difference when considering the same set of retrieved images. Some examples that follow the category-level definition are the works of Bui *et al.* (2017), Bui *et al.* (2018a), Seddati, Dupont and Mahoudi (2017). And some examples that follow the fine-grain definition are Sangkloy *et al.* (2016a), Lin *et al.* (2019), Bhunia *et al.* (2021a). The work presented in this thesis covers both modalities. In Chapter 3 our representations are used for category-level retrieval; In Chapter 4 our model does Compositional SBIR, which in turn

follows the fine-grain definition; Finally in Chapter 5 we show a solution for Zero-shot SBIR, always defined as a category-level problem in the current literature.

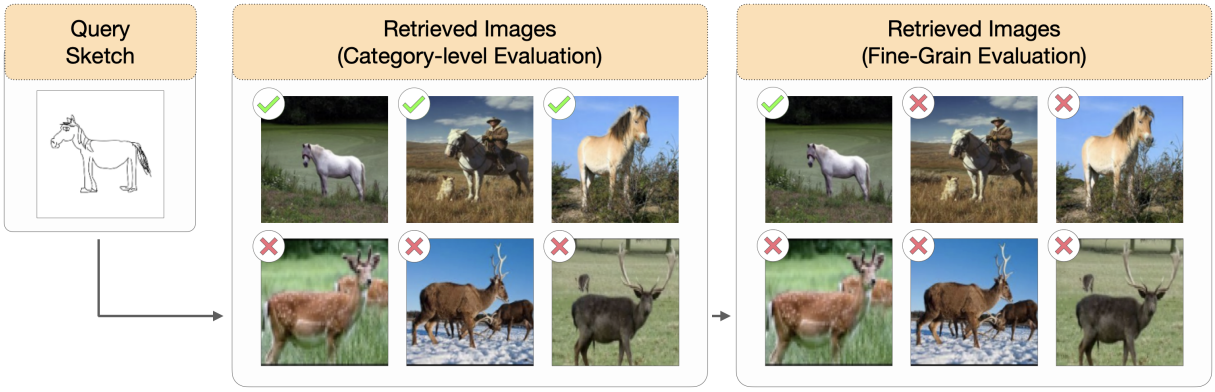


Figure 2 – Difference between evaluation in category-level SBIR versus fine-grain SBIR. The box on the right represents the query sketch whilst both boxes to the right contain the same retrieval result but are highlighted differently according to the chosen evaluation method. A green marker determines a relevant images while a red one marks irrelevant ones.

Apart from how we define relevancy for SBIR, evaluation metrics are commonly lifted from the general information retrieval field. We highlight Precision, Recall (CLEVERDON, 1967), Average Precision (CHOWDHURY, 2010) and Mean Average Precision (mAP). Recall is defined as the fraction of all relevant images retrieved within the first k images in the ranking:

$$R = \frac{\{\text{relevant images}\} \bigcap \{\text{retrieved images}\}}{\{\text{relevant images}\}} \tag{2.1}$$

Recall R is often presented as "Recall-at-k" or $R@k$; considering all of the images in the ranking ($k = N$ with N being the size of the dataset) would always be $R = 1.0$. Precision is the counterpart to Recall and is defined as the fraction of the first k retrieved images that are relevant:

$$P = \frac{\{\text{relevant images}\} \bigcap \{\text{retrieved images}\}}{\{\text{retrieved images}\}} \tag{2.2}$$

Precision is also presented as "Precision-at-k" or $P@k$. Larger values of k are less telling of the user experience as users often value top-ranked results more. With this constraint in mind, Average Precision is a metric designed to "reward" relevant images at the top of the ranking:

$$AP = \frac{\sum_{k=1}^{n} (P@k \times rel(k))}{\text{number of relevant images}} \tag{2.3}$$

Where $rel(k)$ is a function defined as $rel(k) = 1$ if the document at rank k is relevant and $rel(k) = 0$ otherwise; N is the size of the dataset. Because precision tends to get lower as k increases, this formulation makes it so that AP grows as the *concentration of relevant images* at the top increases. Average Precision can also be computed as the area under the Precision-Recall curve. As with the other metrics, $AP@K$ refers to computing the AP up until $k = K$.

Within the context of SBIR, benchmarks for categorical SBIR usually have a test set of queries for which it is possible to compute Average Precision. The metric that evaluates the

model for all queries is then the mean Average Precision (mAP), defined simply as the average AP across the test queries. For fine-grain SBIR, the mean Recall-at-k metric is often used to compare systems; this is because there is only one relevant image and so $R@k$ answers the question: "Was the relevant image among the first k results?".

2.2 Autoencoders

One of the first neural networks architecture ever designed (GOODFELLOW; BENGIO; COURVILLE, 2016), autoencoders are known for their minimalistic design and for learning representations without supervision. The goal of an autoencoder function $f_{ae}(.)$ is to learn a mapping from an input x to itself, behaving like an identity function. Good data representations come from restricting the model's internal capacity for representation as to avoid simple copy (the identity function). The optimisation loss for such network is then very simple:

$$\mathscr{L}(f_{ae}, \theta_{ae}, x) = ||x - f_{ae}(x)|| \tag{2.4}$$

Where $f_{ae}(.)$ is the autoencoder, θ_{ae} is the set of learnable parameters and x is a sample. Internally, the architecture is a composition of two sub-modules known as the encoder ($f_{encoder}(.)$) and the decoder ($f_{decoder}(.)$). The encoder should project the image into a restricted space (named the "code" or the "latent space") whereas the decoder gets as input a representation from this latent space and should be able to generate (or rather reconstruct) the same image that yielded such representation. A simple schematic of an autoencoder can be found on Figure 3.

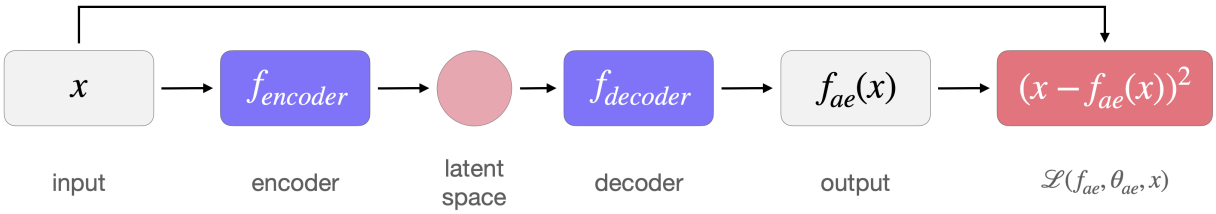


Figure 3 – Generic autoencoder architecture showcasing the individual encoder and decoder modules as well as the loss function.

The most common capacity restriction used is to choose the latent space's dimensionality to be lower than that of the original data space, enforcing that the encoder and decoder should retain information about the training data distribution should they be able to properly reconstruct the images. That method is known as an Undercomplete Autoencoders (AEs).

Another options for limiting capacity besides using low dimensionality are the inclusion of noise on input data or enforcing sparsity on the latent space, both of which could be used together with Undercomplete AEs or even with Overcomplete AEs (models with latent spaces larger than the data space, restricted only through these alternatives).

Besides its use for feature learning, work on autoencoders have also had an impact on the field of generative models: the encoder can be interpreted as a mapper between the latent

space and the data space, behaving as an estimator for the joint probability distribution across both spaces. Further exploring this idea, Kingma and Welling (2013) introduced the Variational Autoencoder (VAE). The VAE is set up such that the learned latent space is actually a set of parameters for Gaussian distributions, essentially mapping each image into its own Gaussian distribution.

To encourage the latent space to mimic a single Gaussian distribution, a Kullback-Leibler divergence between the space's samples and a $\mathcal{N}(0,1)$ distribution is added to the loss function. This forces the latent space to retain semantic relationships as a way to save "space" and still retain the necessary information. This latent space can then be used with random variables to produce new or real-data-neighbouring samples retaining the prime features of the original data.

Autoencoder designs are commonly used across Computer Vision. In the SBIR literature, VAEs were used by Sain *et al.* (2021), Dutta, Singh and Biswas (2021) to disentangle stylistic information in sketches from the semantic content. They were also used by Yelamarthi *et al.* (2018a) to map sketch features to image features. We use an Autoencoder design for our Sketchformer, presented in Chapter 3.

2.3 Generative Models

Without lack of generality, a generative model is one that, given a probability distribution P_{data}, learns to produce samples from a distribution P_{model} that is an estimation of the original P_{data}.

There are different paradigms to determine how close one distribution is from another and how such estimation is made, but most currently studied models rely on the principle of Maximum Likelihood. Within this frame, it is assumed that P_{model} is a parameterised distribution with parameters θ_{model} and that learning the model is changing this set of parameters so that P_{model} becomes a better estimation of P_{data}. The meaning of "better" in this context is determined by the likelihood $\mathscr{L}(\theta_{model}|X)$, computed as the sum of probabilities of known samples X (from training data, defined by P_{data}) within the learned P_{model}:

$$\mathscr{L}(\theta_{model}|X) = \prod_{i=1}^{N} P_{model}(x_i) \tag{2.5}$$

Maximising this equation is a clear path towards approximating P_{model} and P_{data}. With the Maximum Likelihood principle in focus, it is possible to break down generative models that rely on it into two main groups: *(i)* models the explicitly learn the density function and *(ii)* ones that implicitly do so. In figure 4 we reproduce a more complete categorisation proposed by Goodfellow (2016) and highlight where Generative Adversarial Networks (GANs, described in Section 2.4) and Variational Autoencoders (already discussed in Section 2.2) fall into. The figure however contains other well-known generative models. Within explicit models, Fully Visible Belief Networks were first designed by Frey, Hinton and Dayan (1996) and have been

used in state-of-the-art audio generation models such as the more recent WaveNet (OORD *et al.*, 2016). Boltzmann Machines (FAHLMAN; HINTON; SEJNOWSKI, 1983) (BMs) are a classic generative model that uses Markov Chains to achieve maximum likelihood through explicit estimation, differentiating themselves from Generative Stochastic Nets (BENGIO *et al.*, 2014), which also use Markov Chains but supports sample generation dependent on previous samples.

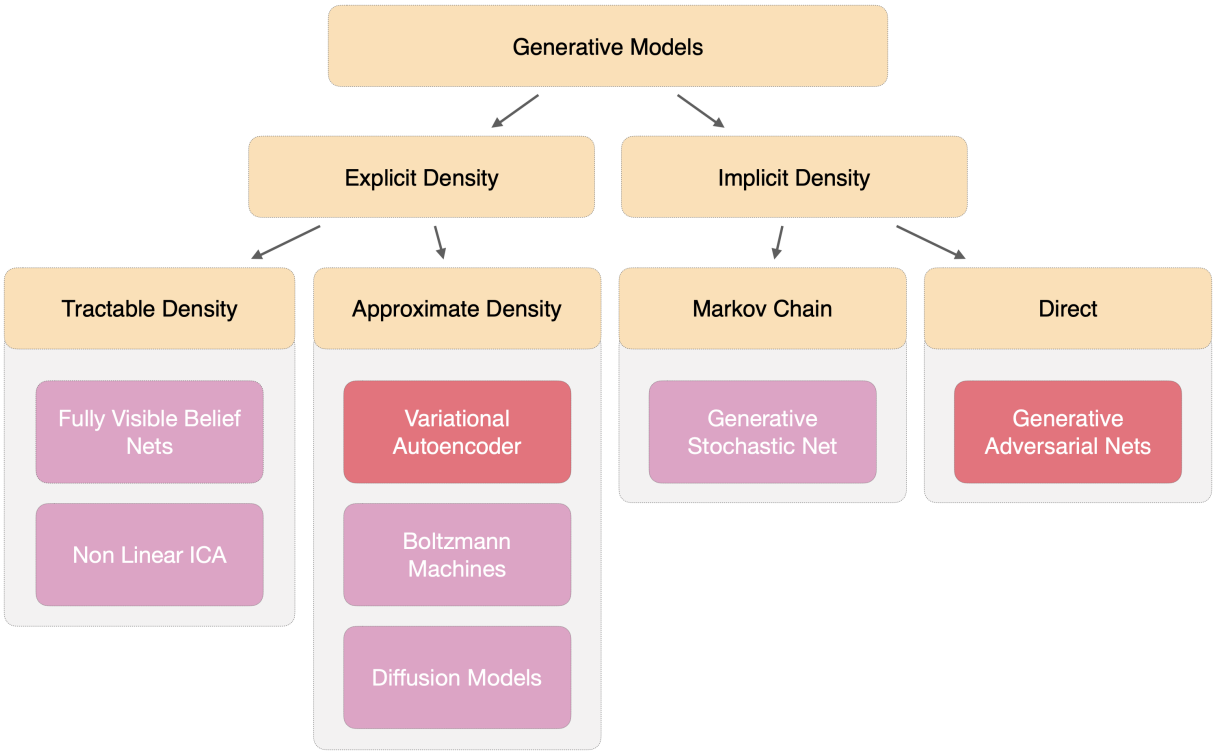


Figure 4 – Categorisation of Generative models proposed by Goodfellow (2016). Models specifically described in this thesis are highlighted (Variational Autoencoders and GANs). We have amended the categorisation with the inclusion of the recent Diffusion Models.

Generative Adversarial Networks (GANs) are well known implicit density estimation models. Although they were not designed with the Maximum Likelihood principle as a guide, the formulation can be easily amended to surface the relationship and allow comparison (GOODFELLOW, 2016). GANs were successful for they do not need to learn the density function explicitly and when compared to GSMs they do not make use of Markov Chains, avoiding the computational cost often associated with those as each sample depends on previous ones to exist.

Given their versatility, GANs emerged as both a one of the best generative frameworks available and also as a method for visual style transfer and cross-domain mapping. In the following section we are going to further explore how GANs work and how they were used in SBIR and in this thesis' studies.

Additionally, we have amended the categorisation of Goodfellow (2016) to include Diffusion Models; a category of models recently introduced to the generative model pantheon. While not used in this thesis, Diffusion Models are worth mentioning as they have found mainstream success in Image Synthesis and are currently supplanting GANs on many applications. Intro-

duced by Sohl-Dickstein *et al.* (2015), the idea of Diffusion is to learn the inverse conditional distribution of a Markov chain where Gaussian noise is added to an image until the entire image is the result of a Gaussian distribution. These models, combined with large language models (BROWN *et al.*, 2020; RADFORD *et al.*, 2021), are the current state-of-the-art in image synthesis (RAMESH *et al.*, 2021; ROMBACH *et al.*, 2021; DHARIWAL; NICHOL, 2021).

2.4 Generative Adversarial Networks

The Generative Adversarial Networks (GAN) formulation was introduced by Goodfellow *et al.* (2014), it was presented as a novel framework for building generative models that could learn to sample from a probability distribution just from its real-world samples. They should be capable of implicitly capturing the density functions that guide the sampled distribution.

2.4.1 Generator/Discriminator

Within the GAN framework, two models are trained simultaneously; one is the f_{gen} generator and the other is the f_{disc} discriminator. For f_{gen}, given an input z from a known probability distribution (e.g. Gaussian), it should output a sample x from the implicitly learned distribution P_{gen}; Our f_{disc}, however, is designed as a classifier that judges if an input sample x was sampled from P_{data} (the original distribution) or P_{gen} (generated by f_{gen}).

While discriminator f_{disc} is trained to perform better classifications using samples from both P_{data} and P_{gen}, the generator f_{gen} is trained to "fool" f_{disc} by worsening the results of its classification and, as a consequence, producing samples that are better resemble ones from the original P_{data}.

An easy-to-grasp metaphor was introduced by the original author: The generator f_{gen} is like a money counterfeiter trying to fool the police (f_{disc}). The police has access to real money (from P_{data}) and also counterfeits (from P_{gen}) and its P&D department is often learning new ways of detecting counterfeits. However, each time the counterfeiters are caught, they study how the police identified their fakes and improve their techniques.

Finally, GANs are a structured probabilistic model with latent variables z and observed variables x. The two components f_{disc} and f_{gen} are generally represented as functions and the only strong requirement is that they are both differentiable. That means that neural networks (or deep neural networks) fit the description and are therefore widely used as both f_{gen} and f_{disc}, trained traditionally with any variation of the Stochastic Gradient Descent (SGD) optimisation. In figure 5 we represented the general architectures of GANs, with all its components and relationships.

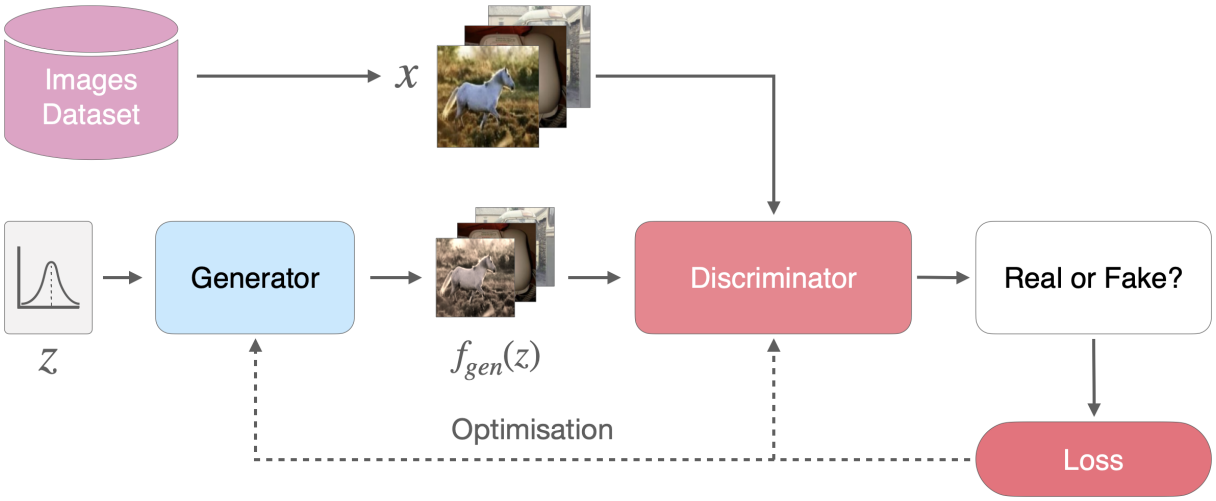


Figure 5 – Diagram representing the training procedure of a GAN model. The Generator gets a known distribution as input (z) and maps this vector into the data space (x). The discriminator classifies samples from the real dataset and set of generated samples into "real" or "fake" classes. Finally, the gradient of the classification loss of the discriminator is backpropagated into its weights while its inverse is backpropagated into the generator's weight.

2.4.2 Losses and Training

Optimisation of GAN models is done through SGD-based methods. The setup is similar to any model trained with gradient descent: a minibatch of samples is collected from P_{data} (the training dataset) and from the known distribution P_z. The z set is used as input to f_{gen} to synthesise a set from P_{gen}. Then, each network is trained with their respective loss function, which combined constitute the zero-sum game proposed by the author. The original losses are defined as a single value function:

$$\min_{f_{gen}} \max_{f_{disc}} V(f_{gen}, f_{disc}) = \mathbb{E}_{x \sim p_{data}(x)}[\log f_{disc}(x)] + \mathbb{E}_{z \sim p_{gen}(z)}[\log(1 - f_{disc}(f_{gen}(z)))] \tag{2.6}$$

which can easily be broken down into the two losses:

$$\begin{aligned} \mathscr{L}_{f_{disc}}(\theta_{f_{disc}}, \theta_{f_{gen}}) &= -\frac{1}{2}\mathbb{E}_{x \sim p_{data}(x)}[\log f_{disc}(x)] - \frac{1}{2}\mathbb{E}_z[\log(1 - f_{disc}(f_{gen}(z)))] \\ \mathscr{L}_{f_{gen}}(\theta_{f_{disc}}, \theta_{f_{gen}}) &= \mathbb{E}_z[\log(1 - f_{disc}(f_{gen}(z)))] \end{aligned} \tag{2.7}$$

These losses and the value function setup constitute a zero-sum game (from the field of game theory) and also were mathematically proved to achieve its purpose of equating distributions P_{data} and P_{gen} at the end of convergence. While the proof presented by Goodfellow *et al.* (2014) is sound, it makes strong assumptions about the functions that represent f_{gen} and f_{disc}: they should be convex and updated in function space (not parameter space); two assumptions that neural networks as currently designed do not fulfil.

Variations of this functions of course exist, with the most commonly implemented version of the algorithm not actually being the original. The first version of the game suffers from convergence instability in one specific instance: when f_{disc} is performing well and consistently

discriminating real from false samples correctly, its loss $\mathscr{L}_{f_{disc}}$ gradient approaches zero and in turn, by being its negative the loss $\mathscr{L}_{f_{gen}}$ also approaches zero, stagnating the gradient descent optimisation much earlier than convergence is achieved. The common solution is simply a rewrite of $\mathscr{L}^{(G)}$ that, while breaking the zero-sum game, works better in practice:

$$\mathscr{L}_{f_{gen}}(\theta_{f_{disc}}, \theta_{f_{gen}}) = -\frac{1}{2}\mathbb{E}_z \log f_{disc}(f_{gen}(z)) \tag{2.8}$$

and constitutes what is called the GAN non-saturating game.

Before the recent Diffusion Models, GANs were widely used in the literature for all tasks related to image synthesis. The work of Mirza and Osindero (2014) introduced class-conditional generation and Isola *et al.* (2017) showed how to design a GAN model that can map between domains (including sketch to image). Many other models followed their success (ZHU *et al.*, 2017; WANG *et al.*, 2018; PARK *et al.*, 2019b; RADFORD; METZ; CHINTALA, 2015; LU *et al.*, 2018). In this thesis we make use of the GAN framework to design an auxiliary synthesis task in Chapter 4; we show the combination of the retrieval and synthesis tasks improves the learned representations significantly. GANs are also present in Zero-shot SBIR methods as solutions for ZSSBIR often use generative models to simulate unseen data during training (DUTTA; SINGH; BISWAS, 2021; SHEN *et al.*, 2018; YELAMARTHI *et al.*, 2018a).

2.5 Multistream Networks

Multi-stream networks are a subset of neural networks where the computation is performed on multiple inputs at the same time, producing multiple outputs as well, usually under common or related loss functions. These models are interpreted as being the combination of multiple independent networks with interactions introduced as shared weights or losses.

Initial proposals designed these networks with two streams (SIMONYAN; ZISSERMAN, 2014); new architectures have explored three or more streams depending on the application. Most uses of multistream networks use siamese or triplet setups designed to project inputs into a semantically sensible embedding. These spaces usually have a lower dimension than the original inputs and are commonly trained through metric learning. Each network contained within a multistream model can share weights with one another, entirely, partially or not at all.

2.5.1 Siamese Networks with Contrastive Loss

A typical siamese network is made of two identical networks that share all weights but do not share inputs and outputs. The training is simultaneous and optimised by gradient descent over the contrastive loss function:

$$\mathscr{L}(\theta_{siam}, (y, x_1, x_2)) = (1-y)(f_{siam}(x_1) - f_{siam}(x_2))^2 + y\max\{0, m - (f_{siam}(x_1) - f_{siam}(x_2))^2\} \tag{2.9}$$

Where f_{siam} is the network with θ_{siam} as its set of weights, $y \in [0,1]$ is a label that determines if a given pair (x_1, x_2) is semantically similar or dissimilar and m is a margin term that avoids the problem of ever growing distances between dissimilar pairs. This original function, called *contrastive loss* was proposed by Hadsell, Chopra and LeCun (2006) and then used by Chopra, Hadsell and LeCun (2005) for facial recognition and by Qi *et al.* (2016) for SBIR. We will explore in Section 2.5.3 a variant of the contrastive loss that is commonly used in self-supervised models.

2.5.2 Triplet Networks with Triplet Loss

A triplet network follows the same principle as the siamese, but further branches the model into three networks. Two branches get as input semantically similar examples (the anchor and positive branches) while the third branch (the negative branch) gets a dissimilar example as input.

While siamese networks optimise the positive-pair distances to zero and the negative-pair distances to infinity, the triplet loss function is designed to optimise the relative difference between the inner distance of each kind of pair. To avoid the undesirable convergence of positive/negative to zero/infinity, a margin is introduced to and bounds the relative difference:

$$\begin{aligned}\mathscr{L}(\theta_{branch^a}, \theta_{branch^{pn}}, (x_a, x_p, x_n)) = \frac{1}{2}\max\{0, m + (f_{branch^a}(x_a) - f_{branch^{pn}}(x_p))^2 \\ - (f_{branch^a}(x_a) - f_{branch^{pn}}(x_n))^2\}\end{aligned} \tag{2.10}$$

where (x_a, x_p, x_n) is the triplet input, m is the aforementioned margin, θ_{branch^a} and $\theta_{branch^{pn}}$ are the weights of the anchor and positive networks, which are represented by functions f_{branch^a} and f_{branch^pn} respectively. Negative examples do not go through a separate network as the same weights and architecture are re-used from the positive branch. When the triplets are from the same domain, even the anchor branch shares all the weights, as was done in FaceNet (SCHROFF; KALENICHENKO; PHILBIN, 2015), the first study to implement a triplet network. The authors of FaceNet train their triplet network so that faces from the same person are close in the embedding and other people are projected further away; the final model is then used for facial recognition.

Triplet designs were quickly adopted for SBIR (YU *et al.*, 2016; SONG *et al.*, 2016; SONG *et al.*, 2017; BUI *et al.*, 2017) and are used to this day (LING *et al.*, 2022; JING *et al.*, 2022; SAIN *et al.*, 2022). Figure 6 shows the expected outcome of cross-domain training with the Triplet Loss. Weight sharing in cross-domain setup is yet a discussed topic. While in Bui *et al.* (2017) weights are partially shared between the sketch and image branches, the works of Sangkloy *et al.* (2016a), Bui *et al.* (2018a) do not share any weights. In this thesis we have implemented the Triplet loss both for Sketchformer (Chapter 3) and Scene Designer (Chapter 4). In both instances weights of the encoders are not shared except for the last projection layer.

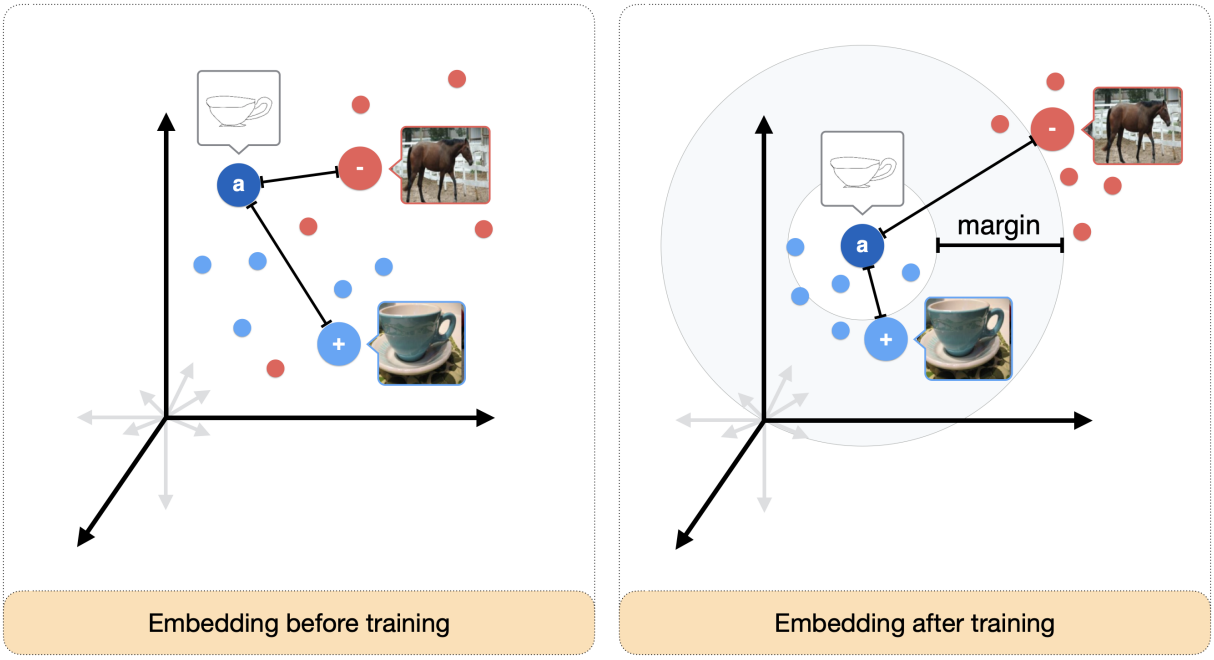


Figure 6 – Expected outcome of training a feature extraction network with triplet loss. In this example, anchors are sketches while positive and negative examples are images from same and distinct categories, respectively. Before training the random feature space usually mixes all categories together; We expect that after training similar samples will be at least a margin closer than dissimilar ones.

2.5.3 Self-Supervised Learning

Another field that makes considerable use of multi-stream networks is Self-Supervision. Most of the initial work on deep learning for Computer Vision has relied on the availability of large annotated datasets. Some even mention the creation of the ImageNet Large Scale Visual Recognition Challenge (RUSSAKOVSKY *et al.*, 2015) and the accompanying ImageNet dataset as the starting point for the "Deep Learning Boom" (KRIZHEVSKY; SUTSKEVER; HINTON, 2012). This reliance on ever growing labelled datasets is counter-productive to research, experimentation and real world applications. The idea for self-supervised pre-training started within the field of Natural Language Processing with the introduction of GPT (RADFORD *et al.*, 2018) and BERT (DEVLIN *et al.*, 2019). Each of those models is trained with a pretext task. The core idea is to learn a model that can then be quickly fine-tuned for a diverse set of downstream tasks. The GPT model is trained to predict the next word in a sentence while BERT is trained to predict random masked words in sentences. Both models are trained with large unlabelled datasets for millions of iterations. The learned representations are later finetuned for tasks such as question-and-answering and machine translation.

The field of Computer Vision followed with a similar concept of self-supervision. The works of Oord, Li and Vinyals (2018), Wu *et al.* (2018) introduced the idea of self-supervision through contrastive learning and Chen *et al.* (2020) were the first to design self-supervised models as they are used today. Such approaches showed good generalisation capacity and robustness

when compared to supervised pre-trained models (CAVALLARI; PONTI, 2021). The SimCLR model (CHEN *et al.*, 2020) had success in training a multi-stream, two branch model where positive pairs are two copies of the same image augmented through different random functions (random crops, rotations, changed colours, luminance properties and others) and negative images are random augmentations of other images in the dataset. This pretext task teaches image representations that are invariant to the augmentations applied and are therefore more general. Besides introducing the idea of contrastive self-supervision, the work of Oord, Li and Vinyals (2018) also introduced a new contrastive loss tailored for this application: Information Noise-Contrastive Estimation, or InfoNCE. InfoNCE is designed around Noise Contrastive Estimation, the idea being to learn a new distribution (here, the latent space conditioned on the input images) by comparing it with a pre-defined noise distribution. This is similar in concept to Word2Vec training (MIKOLOV *et al.*, 2013), where the predicted word is compared to other random words (noise distribution) that should not be predicted. In self-supervised Computer Vision, InfoNCE is defined as:

$$\mathscr{L}_{InfoNCE}(q,k_+,\{k_0,\cdots,k_i\}) = -log\frac{exp(q.k_+/\tau)}{\sum_{i=0}^{K} exp(q.k_i/\tau)} \tag{2.11}$$

where q is the "query" image and k_+ is the "key" image, each being an augmentation of the same image. All other K images $\{k_0,\cdots,k_i\}$ are random negative samples and τ is a temperature parameter that controls how discriminative the loss is (temperatures are $\tau < 0$ and lower values increase the discrimination). This jargon was introduced by He *et al.* (2020), where they formulated the self-supervised pretext task as a dictionary learning task: query images must match their single key in a dictionary; the "dictionary" are the images in a batch and the key is the other augmented version of the query image. The authors also explicit that InfoNCE is equivalent to learning a $(K+1)$ classification problem, where q must be classified as class k_+. This comparison is easy to see as the loss is defined as a cross-entropy between the positive and negative sample distances.

Many studies on self-supervised learning followed on the steps of SimCLR (CHEN *et al.*, 2020) and MoCO (HE *et al.*, 2020). Most follow up studies focus on (i) solving the need for large batch sizes and number of iterations (CARON *et al.*, 2021; YEH *et al.*, 2022) and (ii) creating new ways of avoiding the degenerate solution where the model maps all images to a single vector (CARON *et al.*, 2020; CHEN *et al.*, 2020). The work of Grill *et al.* (2020), BYOL, avoids the need for negative samples (noise distribution) and InfoNCE by adding a projector network from q to k_+ and using a momentum encoder. In sketch literature, a self-supervised design was proposed by Bhunia *et al.* (2021b) with the pretext task of rasterising or vectorising a sketch input and the architecture of BYOL was used by Saavedra, Morales and Murrugarra-Llerena (2022) to train an SBIR model. Most of the impact to SBIR comes however not from applying self supervision directly, but instead from applying the InfoNCE loss for cross-modal contrastive learning and finetuning these pre-trained models for SBIR (TIAN *et al.*, 2022; WANG *et al.*, 2022; WANG *et al.*, 2021). In the Sketch-an-Anchor model (Chapter 5) we follow both ideas,

using the InfoNCE loss and finetuning a self-supervised pre-trained model.

The self-supervised model used in Sketch-an-Anchor and other SBIR models (TIAN *et al.*, 2022; WANG *et al.*, 2022) is called DINO and was introduced by Caron *et al.* (2021). DINO is the first self-supervised model to be implemented with Vision Transformers (which we cover in Section 2.6). They design their pretext task as a self-distillation task. Distillation is commonly used to transfer knowledge between architectures (GOU *et al.*, 2021), but here it is used to approximate the outputs of each branch as if they were probability distributions: outputs are normalised with softmax and compared using cross-entropy ($-alogb$). The knowledge distillation approach was chosen because each branch gets specific augmented versions of images as input; the "teacher" branch with large image crops and the "student" with much smaller ones. DINO's representations were found to perform well when used with k-nearest-neighbours classifiers; this finding implies that their learned embedding is also good for retrieval models, hence the common use for SBIR.

2.6 The Transformer Architecture

Transformers are a recent model architecture for deep learning models, introduced first for natural language processing and then expanded to Computer Vision. They are attention-based models, and therefore rely on the core concept of attention layers. The first proposal for the attention mechanism came from Bahdanau, Cho and Bengio (2014); their model is a machine translation bi-directional LSTM (SCHUSTER; PALIWAL, 1997; Li; Wu, 2015) and their idea was to introduce a way of dynamically averaging the context c of each word in the source sentence and amend this information to the layer producing the probabilities for the next word in the target language. In this original design attention is computed between the hidden states of all encoder steps and the current hidden state of the decoder:

$$a_{ij} = \frac{exp(f(s_{i-1}, h_j))}{\sum_{k=0}^{K} exp(f(s_{i-1}, h_k))} \tag{2.12}$$

where s_{i-1} is the current decoder state and $h_0, \cdots, h_K$ are the encoder hidden states and f is a learnable function, implemented as a multi-layer perceptron (MLP). The resulting attention vector is used to make a weighted average of the encoder's contexts that is used as the input context to that step in the decoder: $c_i = \sum_{k=0}^{K} a_{ik} h_k$.

While other attention designs followed (SUKHBAATAR *et al.*, 2015a; LUONG; PHAM; MANNING, 2015), traction on its use was propelled by the work of Vaswani *et al.* (2017). They introduced the Transformer architecture, a machine translation model designed exclusively with attention layers (plus normalisation and projection layers). Their attention mechanism is called *scaled-dot-product attention* and is defined as:

$$attention(Q, K, V) = \sigma(\frac{QK^T}{\sqrt{d_k}}V) \tag{2.13}$$

where Q, K, V are called the *queries*, *keys* and *values*; σ is the softmax function and $\sqrt{d_k}$ is the square root of the *keys* dimensions. Each of Q, K, V are matrices and the proposed concept is that the attention mechanism is a continuous dictionary retrieval mechanism. *Queries* are compared to the *keys* through dot-product; the result of this comparison is a similarity matrix that is normalised through softmax and the resulting matrix is used to aggregate one different weighted average of the *values* per *query*. Figure 7 shows a simple diagram of this process.

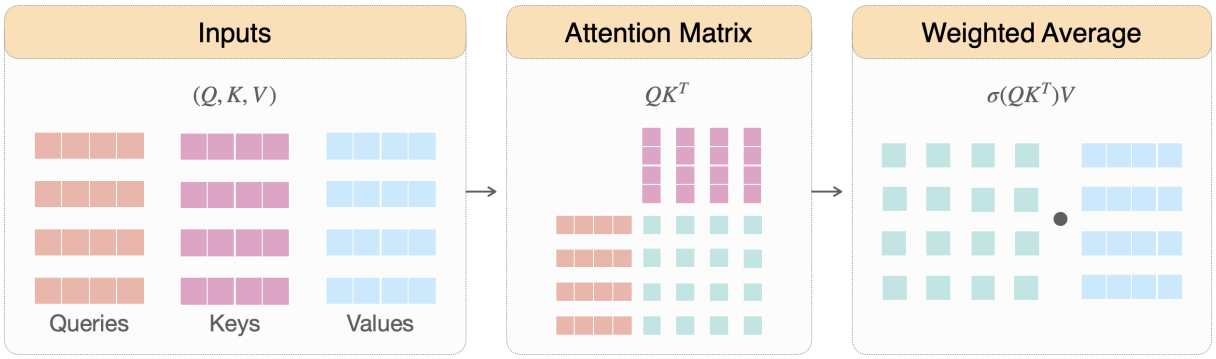


Figure 7 – Diagram of the attention mechanism. Three matrices, Q, K, V are taken as input. The dot product of Q, K creates the attention matrix, which is then used to get weighted averages of the V vectors.

This attention mechanism is implemented in "multi-head" fashion. Multi-head attention is another now-common idea from Vaswani *et al.* (2017); they found that it was beneficial for the model to linearly project slices of the Q, K, V sequences to lower dimensions and compute the attention on each slice separately, concatenating the result of all heads and projecting the result back to the original dimensionality at the end.

The architecture was initially designed for machine translation and as such is composed of an encoder and decoder. The Encoder structure is a sequence of blocks composed of multi-head attention, normalisation and a feed forward projection (a fully connected layer). The input to each multi-head attention in the encoder is the input sequence, projected by three separate learnable projectors, resulting in Q, K, V; the idea is that the encoder is learning to attend to itself so that it can learn better representations to input to the decoder. This particular type of attention is called multi-head self-attention. The Decoder architecture is similar, but each block is composed of one multi-head self-attention, followed by normalisation, followed by a multi-head attention, a feed forward and normalisation. Note that the second attention layer is not self-attending, it takes the current decoder sequence as the *values* V and takes *queries* Q and *keys* K from the encoder output, connecting the two.

Because the attention mechanism is invariant to sequence order (inherited from the dot product operation), sequence dependency is introduced through Positional Encoding. This are position-identifying vectors, based on position-dependent sin and cosine waves in the original paper, that are added to the input word embeddings of the encoder and decoder. Completing the setup, the decoder is designed to predict one word of the target sequence at a time, and depends

both on current output sentence and the encoder representations to compute the next words. To avoid having to compute the output sequentially at training time the authors mask the attention weights of the decoder's input sequence so that each word in a sequence can only attend to previous words on that sequence. The same masking technique is used to accommodate batch training with multiple sequences of variable length (masking their padding vectors). We have reproduced the complete architecture proposed by Vaswani *et al.* (2017) in Figure 8.

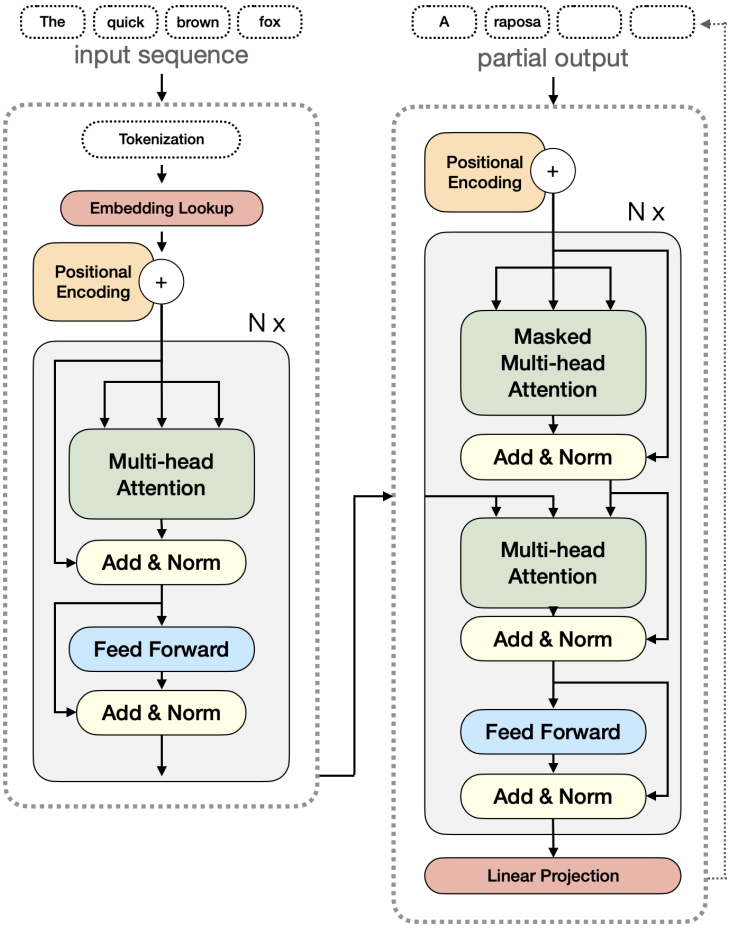


Figure 8 – Diagram of the Transformer Architecture, reproduced from Vaswani *et al.* (2017)

The success of Transformers for language when compared with LSTMs comes from how inputs are processed. Recurrent networks like the LSTM repeatedly take each time step of the input and update context and memory values, sequentially. Because of the attention mechanism, Transformers take the whole sequence of vectors as input and are able to associate words in a sentence to any other word in that sentence, unrestricted by the order. This success was quickly applied to other tasks in natural language processing; GPT (RADFORD *et al.*, 2018) and BERT (DEVLIN *et al.*, 2019) introduced Transformer variants trained under self-supervised protocols, applying the learned representations to a variety of downstream tasks such as question-answering, semantic sentence comparison and sentiment classification. GPT 3 was a much larger model from Brown *et al.* (2020), and they showed that with this 175-billion-parameters model it was possible to use the learned representations for downstream tasks without fine-tuning the model

(zero-shot).

Soon after supplanting LSTMs and other recurrent networks in language, proposals for Computer Vision models with Transformer gained traction. The Image GPT (CHEN *et al.*, 2020) model is trained to perform the same next-in-the-sequence pretext task of GPT but with pixels, effectively learning an image generation model; it was used for inpainting and image representation. The Vision Transformer (ViT) from Dosovitskiy *et al.* (2021) took another approach in converting the model for images; in their design each image is broken down into patches and each patch is considered to be one vector in the input sequence. Their model is trained for image classification and is able to approach the state-of-the-art when trained with a large enough dataset. They follow BERT in introducing a learnable vector (the "CLS" token) to the sequence which is projected and normalised at the output to give class scores. The work of Touvron *et al.* (2021) introduced the DeiT (data-efficient image transformer) to solve the need for large datasets with the use of knowledge distillation specifically designed for Transformers. We have included a diagram of the ViT/DeiT design in Figure 9.

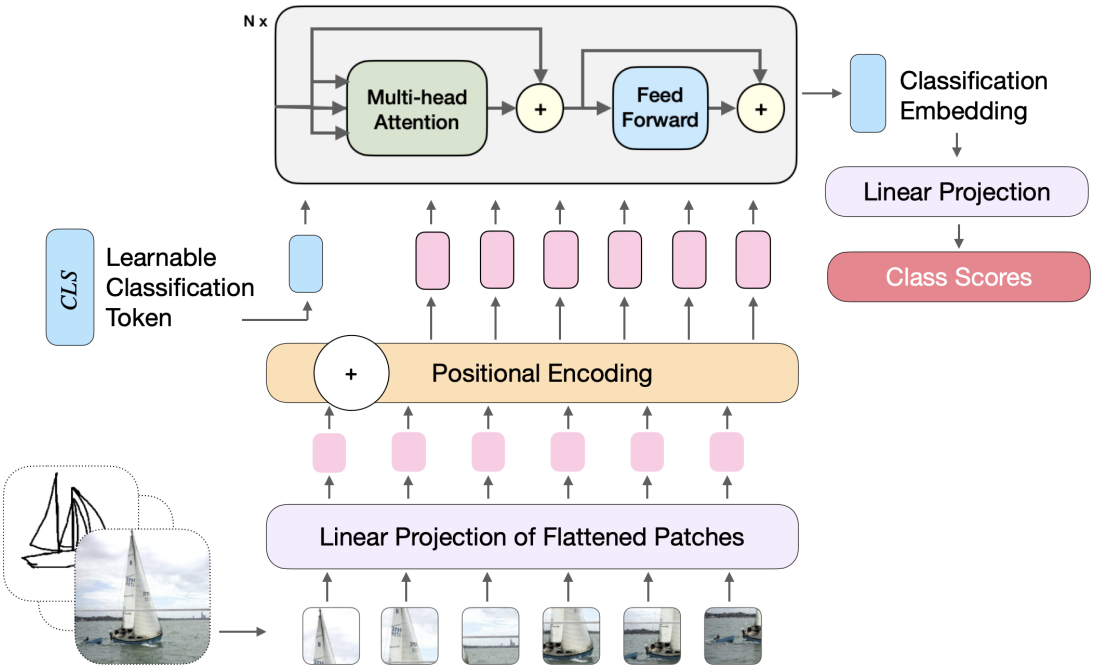


Figure 9 – Diagram of the Vision Transformer (ViT) Architecture, reproduced from Dosovitskiy *et al.* (2021)

With increasing impact to all of Computer Vision, Transformers had a similar impact on SBIR, with the works of Wang *et al.* (2022), Tian *et al.* (2022) using a ViT-based DINO model as encoders for images and sketches in their zero-shot SBIR models and Seddati *et al.* (2022) using a CLIP-based (RADFORD *et al.*, 2021) encoder for fine-grain SBIR. We introduced the first Transformer design specifically for sketches in Sketchformer (Chapter 3); it is based on the original translation architecture but expanded to learn a representation for multiple downstream tasks. Transformers were then also used to encode SVG-based drawings (CARLIER *et al.*, 2020) and generate sketches of fictional animals (BHUNIA *et al.*, 2022).

Besides introducing Sketchformer, the studies presented in this thesis also use Transformer-based modules to aggregate object representations in Scene Designer (Chapter 4) and to to encode images and sketches – with a ViT-based DINO model – in Sketch-an-Anchor (Chapter 5).

2.7 Graph Neural Networks

Given the recent successes of neural networks in computer vision and neural language processing, expansion to other research fields was inevitable. Because many kinds of data can be described as graphs – from social networks to knowledge graphs – research was and is being done to generalise neural networks to work with graph structures. The first ideas for graph neural networks (GNNs) came before these successes however, with the works of (GORI; MONFARDINI; SCARSELLI, 2005) and (SCARSELLI *et al.*, 2008) implementing *message passing* as recurrent networks.

Graph Neural Networks are often defined by how each architecture performs *message passing*. Given a graph $\mathscr{G} = \{\mathscr{E}, \mathscr{V}\}$ with edges $\mathscr{E}$ and vertices/nodes $\mathscr{V}$, processing in a GNN regards how each node "interacts" with other nodes through the edges, and "message passing" is the name given to that process. Recent works have tried to replicate the design principles of Convolutional Neural Networks, but for graph structures. These principles being translation invariance (convolutional kernels are evenly applied across matrices) and locality (kernels aggregate local information). At the same time these designs incorporate a desirable principle for graph processing: permutation equivariance. GNNs of this kind are called Graph Convolutional Networks (GCNs) and are defined by the following message passing function:

$$h_i^{l+1} = f(\phi(h_i^l) + \sum_{j \in \mathscr{N}_i} c_{ij} \rho(h_j^l)) \tag{2.14}$$

where h_i^l is the node representation for node i at the lth step/layer; ϕ and ρ are parametric functions, commonly defined as MLPs; $\mathscr{N}_i$ is the neighbourhood of node i, as in the set of nodes that are connected to node i. Finally, c_{ij} is a the normalised weight of the connection. This simple definition has been shown to preserve common mathematical properties of convolutions. A traditional convolution in the space domain is equivalent to multiplication in the frequency domain, and the same is true for a GCN when considering spectral graph theory and the graph Fourier transform (SHUMAN *et al.*, 2013; SANDRYHAILA; MOURA, 2013; WU *et al.*, 2020). Some well known examples of GCNs are the works of Wu *et al.* (2019), Defferrard, Bresson and Vandergheynst (2016). The work of Kipf and Welling (2016) is a highlight as it keeps the mathematical properties while simplifying the implementation:

$$\mathscr{V}^{l+1} = \hat{A}\mathscr{V}^l W^l \tag{2.15}$$

where $\mathscr{V}^l$ is a matrix containing all node representations at layer l, $\hat{A}$ is the normalised adjacency matrix A and W^l is a learnable weight matrix. We have used this GCN architecture in Chapter 5 to learn better representations for the semantic guides of Sketch-an-Anchor.

Beyond the constraints of GCNs, other message passing designs have also been proposed. The work of Velickovic *et al.* (2018) and Palowitch and Perozzi (2019) have introduced the concept of attention to GNNs. Instead of relying on pre-defined scalar values for edges as GCNs do, these models apply an attention function between nodes to get the scalar edge "strength". Other architectures take more general approaches, as is the example of GraphNet (BATTAGLIA *et al.*, 2018). The authors define three types of update functions (again commonly defined as MLPs) and also learn vector representations for the graphs' edges:

$$e'_k = \phi^e(e_k, v_{o_k}, v_{s_k}, u) \qquad \hat{e}'_k = \psi^{e\to v}(E'_i) \tag{2.16}$$

$$v'_i = \phi^e(\hat{e}'_i, v_i, u) \qquad \hat{e}' = \psi^{e\to u}(E') \tag{2.17}$$

$$u' = \phi^u(\hat{e}', \hat{v}', u) \qquad \hat{e}'_k = \psi^{e\to u}(V') \tag{2.18}$$

where $e_0, \cdots, e_i$ and $v_0, \cdots, v_i$ are vector representations for edges and nodes; v_{o_k}, v_{s_k} represent the subject and object nodes of edge k; u is a "global attribute", taken as a representation for the whole graph. The update functions then update respectively the edges, the vectors and the global attribute. The ψ functions aggregate information, with $\psi^{e\to v}$ aggregating the learned relationships (e) per node (v) and both $\psi^{e\to u}$ and $\psi^{e\to u}$ aggregating edge (e) and node (v) representations globally (u). This general architectural framework is used by Johnson, Gupta and Fei-Fei (2018) to process Scene Graphs in their Scene-graph to Image model and we use a similar architecture to learn our scene-aware object representations in Scene Designer (Chapter 4).

Besides our use of GNNs (Chapters 4 and 5), these models were similarly applied in other studies in the SBIR literature. The work of Liu *et al.* (2020) processed Scene Graphs with the GCN of Kipf and Welling (2016) for the task of Compositional SBIR and Zhang *et al.* (2020) mix visual and word-semantic information using the GCN design from Satorras and Estrach (2018) for Zero-Shot SBIR.

2.8 Few-shot and Zero-shot Learning

While the field of computer vision has made great strides, progressing from models able to classify images (HE *et al.*, 2016) – or sketches (SANGKLOY *et al.*, 2016a) – better than humans or generate high quality images from textual prompts (RAMESH *et al.*, 2021; ROMBACH *et al.*, 2021), these impressive results come at a cost. They require millions or billions of samples to be trained to these performances, and the training process itself is computationaly very expensive (SHARIR; PELEG; SHOHAM, 2020). This is reflected in the field of SBIR as well (SHEN *et al.*, 2018). Beyond the cost of annotation and training, it is reasonable to assume that no matter how large a sketch dataset might get, it will not contain all the categories present and being created in the real world.

With this problem of sample and class-scale in mind, a new machine learning paradigm was created, *Few-shot Learning* (THRUN, 1995; BART; ULLMAN, 2005) (FSL). Arguably,

learning in FSL is more similar to how humans learn than traditional machine learning models; humans can, for example, recognise abstract drawings of things never seen before, from unicorns to hardware design specs. We are able to build meaning from smaller concepts up (FELDMAN, 1997; LAKE *et al.*, 2017), a capability desired but yet far from Computer Vision's grasp (HINTON, 2021; GARAU *et al.*, 2022).

In the survey by Wang *et al.* (2020), FSL is defined as:

Definition 1 (Few Shot Learning)**.** FSL is a type of machine learning problem (specified by Experience E, task T, and Performance measure P), where E contains only a limited number of examples with supervised information for the target T.

When regarding the task T of classification, for example, the model learns a classifier h from the samples in the training set D_{train} and h is then tested on a test set D_{test} to evaluate the final performance; within the FSL paradigm D_{train} is very small. Calling such a problem a N-way-K-shot classification problem is the usual notation for "how small" D_{train} really is, with N referring to the number of classes and K the number of samples available each class. FSL is also applied to classification (RESENDE; PONTI, 2022) and other tasks such as regression (FINN; ABBEEL; LEVINE, 2017; YOON *et al.*, 2018), reinforcement learning (AL-SHEDIVAT *et al.*, 2017), and object detection (BANSAL *et al.*, 2018). In SBIR this paradigm is applied in its hardest form: Zero-Shot SBIR. The name implies that we have an FSL task of the zero-way-zero-shot kind, that is, no samples are available for any class in D_{test}, the class set for evaluation is completely disjointed with the class set for training. This means the model must learn generaliseable concepts from D_{train} such that they can be applied to unseen structures and semantics in D_{test}. This setup is desirable in SBIR for the task does not regard classification but retrieval; it is reasonable to assume that there is a general way of comparing images and sketches, learning how humans abstract shapes into drawings in general.

The body of work in Zero-shot SBIR is extensive, a few highlights being the seminal work of Shen *et al.* (2018), that introduced the task and has also taken on the scale of dataset sizes with their hashing solution. Liu *et al.* (2019) changed the ZSSBIR field by showing the benefit of using knowledge distillation techniques for knowledge *preservation* instead, improving the use of pre-trained models for this task. In Chapter 5 we present Sketch-an-Anchor – our own ZSSBIR solution – and explore the literature more thoroughly.

CHAPTER

3

SKETCHFORMER: TRANSFORMER-BASED REPRESENTATION FOR SKETCHED STRUCTURE

In this chapter we present Sketchformer, a novel transformer-based representation for encoding free-hand sketches input in a vector form, *i.e.*, as a sequence of strokes. Sketchformer was first presented at the CVF Conference on Computer Vision and Pattern Recognition (CVPR 2020):

RIBEIRO, L. S. F.; BUI, T.; COLLOMOSSE, J.; PONTI, M. Sketchformer: Transformer-based representation for sketched structure. In: **Proc. CVPR**. [S.l.: s.n.], 2020.

With this model, we have contributed the first application of the Transformer architecture – now often employed in CV – for the sketch domain. We show Sketchformer can learn semantic representations from abstract sketches that are useful for multiple tasks: sketch classification, sketch-based image retrieval (SBIR), creative generation and semantic interpolation of sketches. Being the first to implement Transformer for sketches by exploring alternative designs for encoding the vector sketch sequence, from continuous "as-created" inputs to tokenised representations akin to neural language processing. Finally, Sketchformer is shown to outperform other sequence-to-sequence architectures on classification and retrieval tasks.

This chapter is comprised of an extended version of the published manuscript, revised for clarity and to highlight the contributions to the thesis:

- New sketch-based Image Retrieval model, our Transformer-based model is the state-of-the-art for vector-based representations for sketches (Hypothesis **H1**)

- Our single representation is used to perform multiple tasks successfully: sketch synthesis, classification and retrieval (Hypothesis **H4**)

3.1 Introduction

Sketch representation and interpretation remains an open challenge, particularly for complex and casually constructed drawings. Yet, the ability to classify, search, and manipulate sketched content remains attractive as gesture and touch interfaces reach ubiquity. Advances in recurrent network architectures within language processing have recently inspired sequence modelling approaches to sketch (*e.g.* SketchRNN from Ha and Eck (2018)) that encode sketch as a variable length sequence of strokes, rather than in a rasterised or 'pixel' form. In particular, long-short term memory (LSTM) networks have shown significant promise in learning search embeddings (XU *et al.*, 2018; COLLOMOSSE; BUI; JIN, 2019) due to their ability to model higher-level structure and temporal order versus convolutional neural networks (CNNs) on rasterised sketches (BUI *et al.*, 2017; QI *et al.*, 2016; COLLOMOSSE *et al.*, 2017; SANGKLOY *et al.*, 2016b). LSTMs have however – like all recurrent network designs – a limited temporal "coverage"; because recurrent networks process each sequence one time step at a time and accumulate the result "over time", computation is heavily dependent on the temporal extent of the input and longer sequences end up creating smaller gradient signals to the network. This limitation restricts the structural complexity of sketches that may be accommodated in sequence embeddings. In language modelling domain, this shortcoming has been addressed through the emergence of Transformer networks (introduced by Vaswani *et al.* (2017) and expanded by Devlin *et al.* (2019), Dai *et al.* (2019)) in which slot masking and parallel processing (versus sequential in Recurrent networks) enhances the ability to learn complex structures that are represented by longer sequences.

This paper proposes Sketchformer, the first Transformer based network for learning a deep representation for free-hand sketches. We build on the language modelling Transformer architecture of Vaswani *et al.* (2017) to develop several variants of Sketchformer that process sketch sequences in continuous and tokenised forms. We evaluate the efficacy of each learned sketch embedding for common sketch interpretation tasks. We make three core technical contributions:

1. **Sketch Classification.** We show that Sketchformer driven by a dictionary learning tokenisation scheme outperforms state of the art sequence embeddings for sketched object recognition over QuickDraw! (THE..., 2018); the largest and most diverse public corpus of sketched objects.

2. **Generative Sketch Model.** We show that for more complex, detailed sketches comprising lengthy stroke sequences, Sketchformer improves generative modelling of sketch – demonstrated by higher fidelity reconstruction of sketches from the learned embedding. We also show that for sketches of all complexities, interpolation in the Sketchformer embedding is stable, generating more plausible intermediate sketches for both inter- and intra-class blends.

3. **Sketch based Image Retrieval (SBIR)**. We show that Sketchformer can be unified with raster embedding to produce a search embedding for SBIR (similar to the work Collomosse, Bui and Jin (2019) with LSTMs) to deliver improved prevision over a large photo corpus (Stock10M).

These enhancements to sketched object understanding, generative modelling and matching demonstrated for a diverse and complex sketch dataset suggest Transformer architectures as a promising direction for stroke sequence modelling (Hypothesis **H1**). Additionally, Sketchformer's design demonstrates that a single semantic representation can be used for synthesis and retrieval (Hypothesis **H4**).

3.2 Related Work

Representation learning for sketch has received extensive attention within the domain of visual search. Classical sketch based image retrieval (SBIR) techniques explored region (HU; WANG; COLLOMOSSE, 2011), graph (JAMES *et al.*, 2013), edge-let (CAO *et al.*, 2010), and sparse gradient features (EITZ; HAYS; ALEXA, 2012; HU; COLLOMOSSE, 2013) building upon the success of dictionary learning based models (e.g. bag of words) (HU; BARNARD; COLLOMOSSE, 2010; BUI; COLLOMOSSE, 2015; SCHNEIDER; TUYTELAARS, 2014). With the advent of deep learning, convolutional neural networks (CNNs) were rapidly adopted to learn search embedding (ZHANG *et al.*, 2016). Triplet loss models are commonly used for visual search in the photographic domain (WANG *et al.*, 2014; RADENOVIĆ; TOLIAS; CHUM, 2016; GORDO *et al.*, 2016), and have been extended to SBIR. Sangkloy *et al.* (2016b) used a three-branch CNN with triplet loss to learn a general cross-domain embedding for SBIR. Fine-grained (within-class) SBIR was similarly explored by Yu *et al.* (2016). Qi *et al.* (2016) instead use contrastive loss to learn correspondence between sketches and pre-extracted edge maps. Bui *et al.* (2018a), Bui *et al.* (2017) perform cross-category retrieval using a triplet model and combined their technique with a learned model of visual aesthetics (WILBER *et al.*, 2017) to constrain SBIR using aesthetic cues in Collomosse *et al.* (2017). A quadruplet loss was proposed by Seddati, Dupont and Mahoudi (2017) for fine-grained SBIR. The generalisation of sketch embeddings beyond training classes have also been studied (BUI *et al.*, 2018a; PANG *et al.*, 2019), and parameterised for zero-shot learning (SHEN *et al.*, 2018; DEY *et al.*, 2019b), a branch of SBIR further discussed in Chapter 5. Such concepts were later applied in sketch-based shape retrieval tasks by Xu *et al.* (2018). Variants of CycleGAN (ZHU *et al.*, 2017) have also shown to be useful as generative models for sketch by Song *et al.* (2018), showing the benefit of using similar representations for multiple tasks. Sketch-A-Net was a seminal work for sketch classification that employed a CNN with large convolutional kernels to accommodate the sparsity of stroke pixels (YU *et al.*, 2017). Recognition of partial sketches has also been explored by Seddati, Dupont and Mahmoudi (2016). Wang, Chen and Zha (2018) proposed

sketch classification by converting sketches to point clouds and sampling unordered points to learn a canonical order.

All the above works operate over rasterised sketches *e.g.* converting the captured vector representation of sketch (as a sequence of strokes) to pixel form, discarding temporal order of strokes, and requiring the network to recover higher level spatial structure. Recent SBIR work has begun to directly input vector (stroke sequence) representations for sketches (MUHAMMAD *et al.*, 2018), notably SketchRNN; an LSTM based sequence to sequence (seq2seq) variational autoencoder proposed by Ha and Eck (2018), trained on the largest public sketch corpus, 'Quick-Draw!' (THE..., 2018). SketchRNN's embedding was incorporated in a triplet network by Xu *et al.* (2018) to search for sketches using sketches. A variation using cascaded attention networks was proposed by Li *et al.* (2018) to improve vector sketch classification over Sketch-A-Net. Later, LiveSketch (COLLOMOSSE; BUI; JIN, 2019) extended SketchRNN to a triplet network to perform SBIR over tens of millions of images, harnessing the sketch embedding to suggest query improvements and guide the user via relevance feedback. The limited temporal scope of LSTM based seq2seq models can prevent such representations from modelling long, complex sketches, a problem mitigated by our Transformer based model which builds upon the success shown by such architectures for language modelling (VASWANI *et al.*, 2017; DAI *et al.*, 2019; DEVLIN *et al.*, 2019). Transformers encode long term temporal dependencies by modelling direct connections between data units, unrestricted by temporal proximity. The temporal range of such dependencies was increased via the Transformer-XL (DAI *et al.*, 2019) and BERT (DEVLIN *et al.*, 2019), which set, upon its publication, new state-of-the-art performance on sentence classification and sentence-pair regression tasks using a cross-encoder. Recent work explores transformer beyond sequence modelling to 2D images (PARMAR *et al.*, 2018).

Our work was the first to apply these insights to the problem of sketch modelling, incorporating the Transformer architecture of Vaswani *et al.* (2017) to deliver a multi-purpose embedding that exceeds the state of the art for several common sketch representation tasks.

3.3 Sketch Representation

We propose Sketchformer; a multi-purpose sketch representation from stroke sequence input. In this section we discuss the pre-processing steps, the adaptions made to the core architecture proposed by Vaswani *et al.* (2017) and the three application tasks.

3.3.1 Pre-processing and tokenisation

Following Ha and Eck (2018) we simplify all sketches using the RDP algorithm (DOUGLAS; PEUCKER, 1973) and normalise stroke length. Sketches for all our experiments are drawn from QuickDraw50M (THE..., 2018) (see Sec. 3.4; for dataset partitions).

1. **Continuous**. Quickdraw50M sketches are released in the 'stroke-3' format where each point $(\delta x, \delta y, p)$ stores its relative position to the previous point together with its binary pen state. To also include the 'end of sketch' state, the stroke-5 format is often employed: $(\delta x, \delta y, p_1, p_2, p_3)$, where the the pen states p_1 - *draw*, p_2 - *lift* and p_3 - *end* are mutually exclusive (HA; ECK, 2018). Our experiments with continuous sketch modelling use the 'stroke-5' format. Because each vector in the sequence would only have five dimensions, the vectors are first projected linearly to the common model dimension by a feed forward layer.

2. **Dictionary learning**. We build a dictionary of K code words ($K = 1000$) to model the relative pen motion *i.e.*, $(\delta x, \delta y)$. We randomly sample 100k sketched pen movements in the training set for clustering via K-means. Each transition point $(\delta x, \delta y)$ is then assigned to the nearest code word, resulting in a sequence of discrete cluster identifiers; each identifier is associated with a learnable token. We also include 4 special tokens; a Start of Sketch (SOS) token at the beginning of every sketch, an End of Sketch (EOS) token at the end, a Stroke End Point (SEP) token to be inserted between strokes (indicate pen lifting) and a padding (PAD) token to pad the sketch to a fixed length. We found that equally sampling inter-stroke (*i.e.*, relative transition when the pen is lifted) and within-stroke transitions leads to the former not being well encoded by the dictionary for the relative rarity of inter-stroke transitions; to amend this imbalance we allocate 20% of the sampled sketch points for inter-stroke transitions.

3. **Spatial Grid**. The sketch canvas is first quantized into $n \times n$ ($n = 100$) square cells, each cell is represented by a token in our dictionary. Given the absolute (x, y) sketch points, we determine which cell contains this point and assign the cell's token to the point. The same four special tokens above are used to complete the sketch sequence.

In Fig. 10 we have diagrammed representations for each of the encoding methods and how each sequence would look like with a simple example "sketch". Then Fig. 11 visualises sketch reconstruction under the tokenised methods to explore sensitivity to the quantisation parameters. Compared with the stroke-5 format (continuous) the tokenisation methods are more compact. Dictionary learned tokenisation (Tok-Dict) can have a small dictionary size and is invariant to translation since it is derived relative pen translations (present in the stroke-3 format). On the other hand quantisation error could accumulate over longer sketches if dictionary size is too low, shifting the position of strokes closer to the sequence's end. Conversely, the spatial grid-based tokenisation method (Tok-Grid) does not accumulate error but is sensitive to translation and yields a larger vocabulary (n^2) that may be sparsely used by the model (*e.g.* strokes seldom reach the boundaries of the canvas).

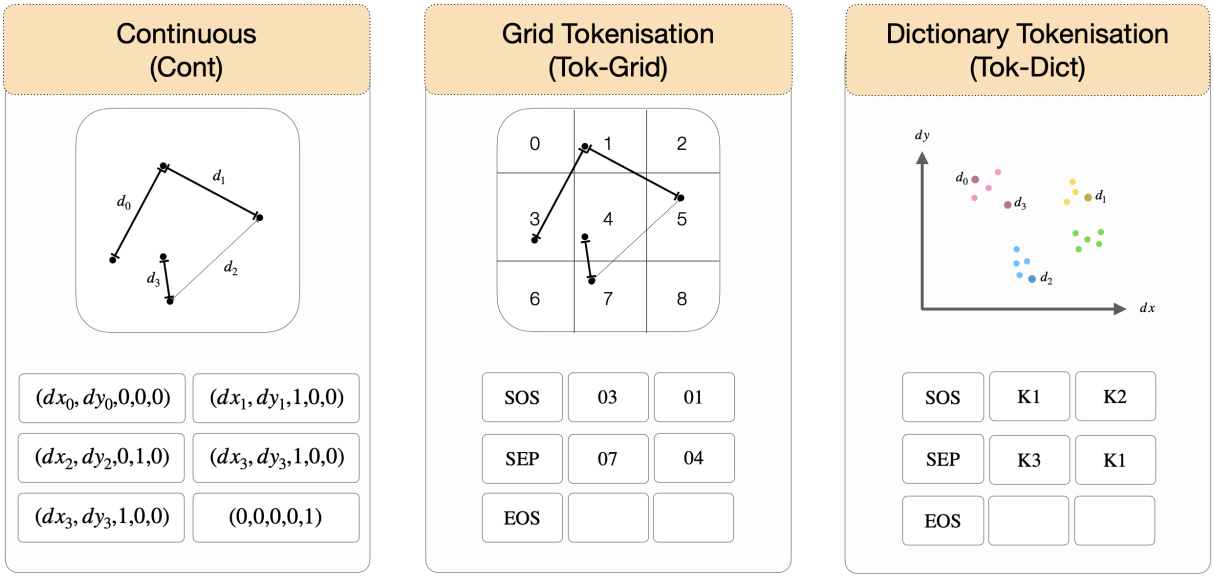


Figure 10 – Diagram with visual example of how each encoding alternative generates its sketch sequence. Continuous sequences use the "stroke-5" format; Grid-based tokenisation quantises the absolute position of each stroke point; Dictionary-based tokenisation clusters stroke delta's and represents each point by the closest centroid. Sequences were represented ordered left to right, top-down; empty squares and zero'ed out values represent padding.

3.3.2 Transformer Architecture for Sketch

Sketchformer uses the Transformer network of Vaswani *et al.* (2017). We add stages (*e.g.* self-attention and modified bottleneck) and adapt parameters in their design to learn a multi-purpose representation for stroke sequences, rather than language. A transformer network consists of encoder and decoder blocks, each comprising several layers of multihead attention (MHA) followed by a feed forward network. Fig. 12 illustrates the architecture with dotted lines indicating re-use of architecture stages from Vaswani *et al.* (2017). In Fig. 14 we show how our learned embedding is used across multiple applications. Compared to Vaswani *et al.* (2017) we use 4 MHA blocks versus 6 and a feed-forward dimension of 512 instead of 2048.

Traditional sequence modelling method such as RNN and LSTM (Li; Wu, 2015) are computed over each vector in a sequence, one after the other, accumulating the results in "memory" and using this "memory" to compute the final output. For these models then the result depends on each time step sequentially (and on the inverted sequence for bidirectional models); the attention mechanism in transformers contrast this, as attention is computed across the entire sequence at once, with time steps able to "attend" to any other time step in the sequence.

The attention mechanism follows the scaled-dot-product design shown in Section 2.6, Equation 2.13, reproduced for clarity:

$$attention(K, Q, V) = \sigma\left(\frac{(QK^T)V}{\sqrt{d_k}}\right) \tag{3.1}$$

where K, Q, V are the respective *keys*, *queries* and *values*, σ is the *softmax* function and d_k the dimensionality of the vectors in K. The mechanism computes similarities between pairs of

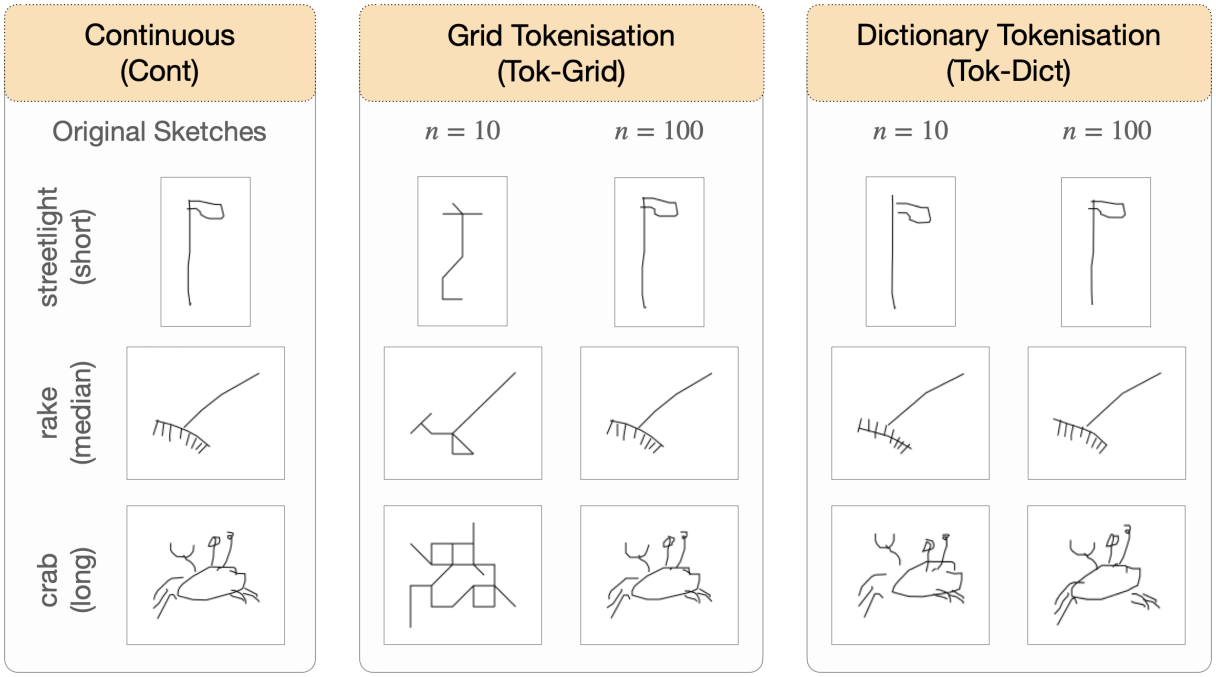


Figure 11 – Visualising the impact of quantisation on the reconstruction of short, median and long sequence length sketches. We compare the continuous (original) sketch representation with reconstructions from tokenised sketches. Grid sizes of $n = [10, 100]$ (Tok-Grid)) and dictionary sizes of $K = [500, 1000]$ (Tok-Dict)) are considered. Sketches are decoded from the tokenised sketch representations, independent of the transformer model.

queries and *keys*, normalises the scores and uses them to accumulate (weighted average) the *values*. In Vaswani *et al.* (2017)'s design, K, Q and V are each projections of the input sequence in the encoder and $attention(.)$ is implemented as part of multi-head attention:

$$mha(K,Q,V) = [attention_0(K_0,Q_0,V_0), ...attention_m(K_m,Q_m,V_m)]W \tag{3.2}$$

where the subscript on K, Q, V refers to taking a subset of the dimensions of each vector, $[.]$ is the concatenating operation and W is a learnable projection matrix.

The result of $mha(.)$ is fed to a positional feed forward network $f_{pff}(.)$, comprised of two fully connected layers with Rectified linear-unit (ReLU) activation. The $t^e(.) = f_{pff}(mha(.))$ blocks are the basis of the encoder side of our network (f_{T_e}):

$$t^e(x) = |f_{pff}(|mha(x,x,x)|)| \tag{3.3}$$

$$f_{T_e}(x) = t^e_N(..(t^e_1(x))) \tag{3.4}$$

where $|.|$ indicates layer normalisation (applied before and after $f_{pff}(.)$) and N is number of $t^e(.) = f_{pff}(mha(.))$ blocks.

The decoder takes as inputs the encoder output and target sequence in an auto-regressive fashion. In our case we are learning a transformer autoencoder so the target sequence is the input sequence shifted forward by 1. Considering $h = f_{T_e}(x)$ as the encoder output, the fundamental decoder blocks follow the design:

$$t^d(h,x) = |f_{pff}(|mha(h,h,|mha(x,x,x)|)|)| \tag{3.5}$$

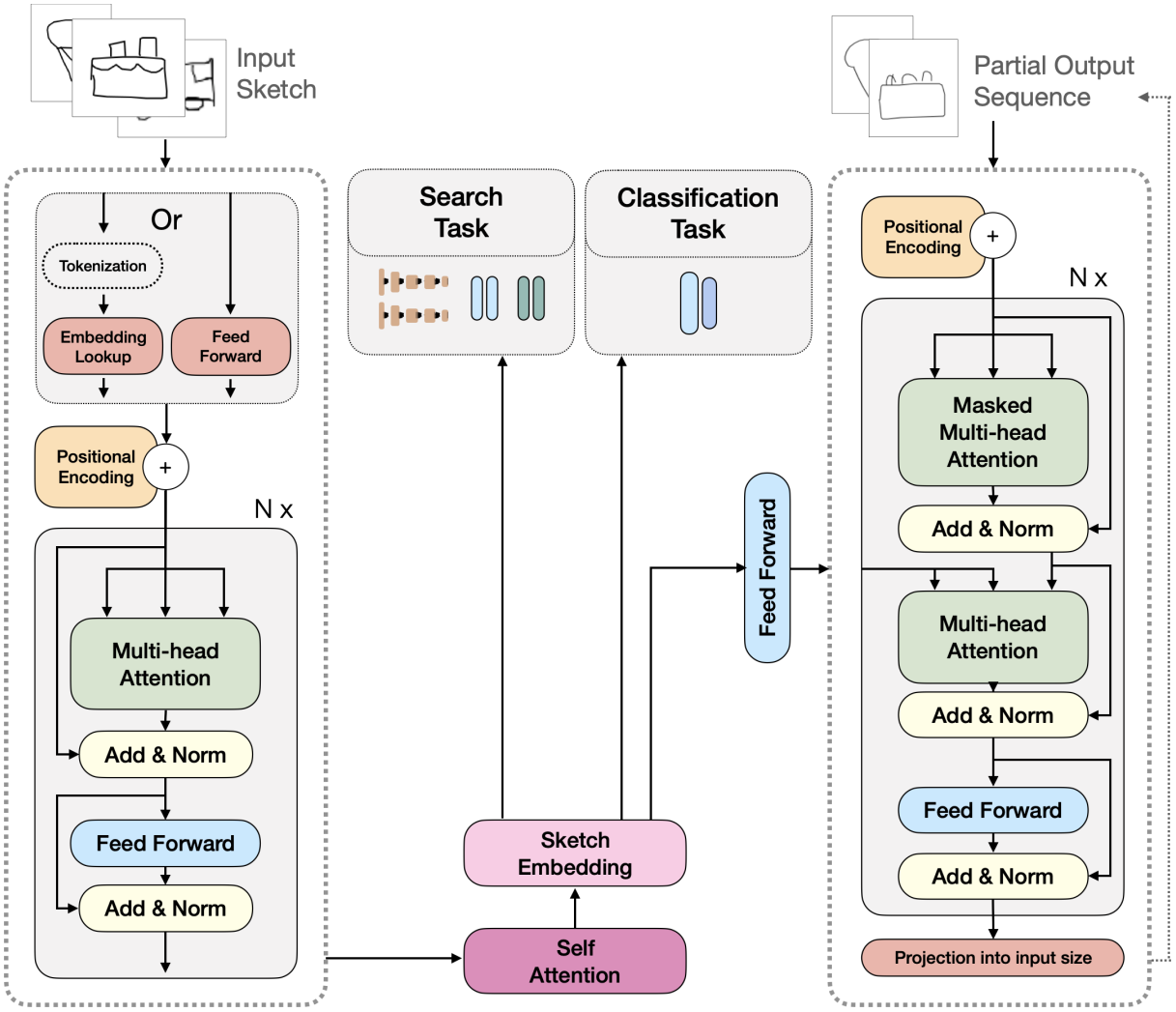


Figure 12 – Schematic of the Transformer architecture used for Sketchformer, which utilises the original architecture of Vaswani *et al.* (2017) but modifies it with an alternative mechanism for the bottleneck (sketch embedding) layer using a self-attention block, as well as configuration changes e.g. MHA head count (see Sec. 3.3.2). We use Sketchformer's learned embedding for three tasks: sketch synthesis (through the decoder), sketch-based retrieval and sketch classification

and N of those blocks compose the decoder:

$$f_{T_d}(h, x^*) = t_N^d(h, t_{N-1}^d(h, ...t_1^d(h, x^*))) \tag{3.6}$$

where x^* is the shifted auto-regressive version of input sequence x.

In practice, the implementation of the transformer is more complicated, incorporating positional encoding (input x is fused with sine waves to encode relative positions within the sequence, since the transformer itself holds no order information) and masked multi-head attention (sequence padding is ignored during $mha(.)$ computation) – see Section 2.6 and Vaswani *et al.* (2017)'s manuscript for further details.

The conventional transformer is designed for language translation and thus does not provide a feature embedding as required in Sketchformer (the unchanged output of $f_{T_e}(.)$ is also

a sequence of vectors of the same length as x). To learn a compact representation for sketch we propose to apply self-attention on the encoder output, inspired by Sukhbaatar *et al.* (2015a):

$$s = \sigma(tanh(hK^T + b)v) \tag{3.7}$$

$$z = \sum_i s_i h_i \tag{3.8}$$

which is similar to the attention mechanism however the *keys* K, a single *value* vector v and bias b are trainable parameters. This self-attention layer learns a weight vector s describing the importance of each time step in sequence h, which is then accumulated to derive the compact embedding z. On the decoder side, z is linearly projected sequence-length times to resume the original sequence shape of h. These are the key novel modifications to the original Transformer architecture of Vaswani *et al.* (2017) (beyond above-mentioned parameter changes).

We also had to change how masking worked on the decoder. The Transformer, being designed for text translation, uses a padding mask to stop attention blocks from giving weight to out-of-sequence points. Since we want a meaningful embedding for reconstruction and interpolation, we removed this mask from the decoder, forcing our transformer to learn reconstruction without previously knowing the sequence length, using only the given embedding representation.

3.3.2.1 Training Losses

We employ two losses in training Sketchformer. A classification (cross-entropy) loss is connected to the sketch embedding z to preserve semantic information while a reconstruction loss ensures the decoder can reconstruct the input sequence from its embedding. If the input sequence is continuous (*i.e.*, stroke-5) the reconstruction loss consists of an L^2 loss term modelling relative transitions $(\delta x, \delta y)$ and a 3-way classification term modelling the pen states. Otherwise the reconstruction loss uses cross-entropy and classifies each of the output vectors as one of the tokens in the dictionary, as is done on language models with word tokens. This yields:

$$\mathscr{L}(x,y) = ylog(\sigma(f_{cls}(f_{T_e}(x))) + (x_r - f_T(x)_r)^2 + x_p log(\sigma(f_T(x)_p)) \tag{3.9}$$

for continuous inputs, where x, y are the input sketch and one-hot encoded class label, f_{cls} is a fully-connected (fc) layer which linearly projects the embedding to the class score space, f_T is the full encoder-decoder Transformer and the subscripts r and p refer to the relative stroke transitions and the pen state. The analogous loss for the tokenised inputs is:

$$\mathscr{L}(x^{token},y) = ylog(\sigma(f_{cls}(f_{T_e}(x^{token}))) + x^{token} log(\sigma(f_T(x^{token}))) \tag{3.10}$$

where x^{token} is the one-hot encoded input token identifiers. We found these losses simple yet effective in learning a robust sketch embedding. Fig. 13 visualises the learned embedding for each of the three pre-processing variants, alongside that of a state of the art sketch encoding model using stroke sequences (COLLOMOSSE; BUI; JIN, 2019).

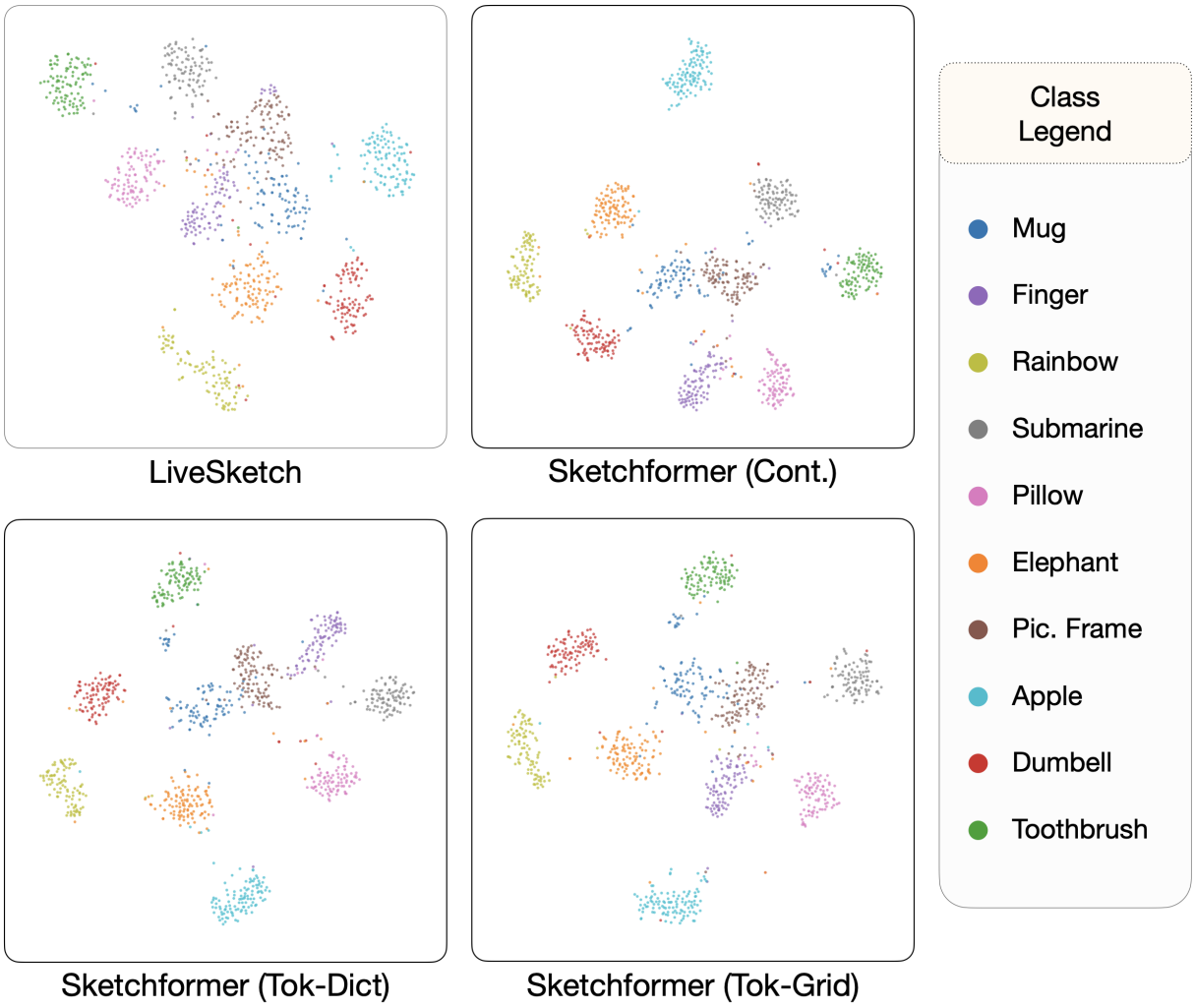


Figure 13 – Visualisation (using t-SNE from Maaten and Hinton (2008)) of the learned embedding space from Sketchformer's three variants, compared to LiveSketch (top left) and computed over the QD-862k test set; 10 categories and 1000 samples were randomly selected.

3.3.3 Cross-modal Search Embedding

To use our learned embedding for SBIR, we follow the joint embedding approach first presented by Collomosse, Bui and Jin (2019) and train an auxiliary network that unifies the vector (sketch) and raster (image corpus) representations into a common subspace.

This auxiliary network is composed of four fully connected layers (see Fig. 14) with ReLU activations. These are trained within a triplet framework and have input from three pre-trained branches: an anchor branch that models vector representations (our Sketchformer), plus positive and negative branches extracting representations from raster space.

The first two fully connected layers are domain-specific and we call each set $f_{vector}(.), f_{raster}(.)$, referring to vector-specific and raster-specific. The final two layers are shared between domains; we refer to this set as $f_{shared}(.)$. Thus the end-to-end mapping from vector sketch and raster

sketch/image to the joint embedding is:

$$u_v = f_{shared}(f_{vector}(f_{T_e}(x_v))) \tag{3.11}$$
$$u_r = f_{shared}(f_{raster}(f_{R_e}(x_r))) \tag{3.12}$$

where x_v and x_r are the input vector sketches and raster images respectively, and u_v and u_r their corresponding representations in the common embedding. $f_{T_e}(.)$ is our sketchformer encoder – that models vector representations – and $f_{R_e}(.)$ is the one for raster images. In the original LiveSketch (COLLOMOSSE; BUI; JIN, 2019), the vector encoder is implemented as a SketchRNN (HA; ECK, 2018)-based model. For $f_{R^{encoder}}(.)$ we use the same off-the-shelf GoogLeNet-based network, pre-trained on a joint embedding task (from (BUI *et al.*, 2018a)).

The training is performed using triplet loss regularised with the help of a classification task. Training requires an aligned sketch and image dataset *i.e.*, a sketch set and image set that share the same category list. This is not the case for Quickdraw, which is a sketch-only dataset without a corresponding image set. Again following Collomosse, Bui and Jin (2019), we use the raster sketch as a medium to bridge vector sketch with raster image. The off-the-shelf $f_{R_e}(.)$ (from Bui *et al.* (2018a)) was trained to produce a joint embedding model unifying raster sketch and raster image; This allowed the authors train the f_{raster}, f_{vector} and f_{shared} networks using vector and raster versions of sketch only. By following the same procedure, we eliminate the need of having an aligned image set for Quickdraw; our network never sees an image feature during training.

The training is implemented in two phases. At phase one, the anchor and positive samples are vector and raster forms of random sketches in the same category while raster input of the negative branch is sampled from a different category; this phase is akin to the usual training protocol for categorical (coarse-grained) retrieval models. At phase two, we take the same sketch in vector and raster forms to build the anchor and positive samples. For the negative sample we mine hard negatives (the ones closest to the anchor); this is similar to the training protocol for fine-grained (one to one matching) retrieval models. The triplet loss maintains a margin between the anchor-positive and anchor-negative distances:

$$\begin{aligned}\mathscr{L}_{tri}(x_a,x_p,x_n) = max\{0, &(f_{shared}(f_{vector}(f_{T_e}(x_a))) - f_{shared}(f_{raster}(f_{R_e}(x_p))))^2 \\ &- (f_{shared}(f_{vector}(f_{T_e}(x_a))) - f_{shared}(f_{raster}(f_{R_e}(x_n))))^2 + m\}\end{aligned} \tag{3.13}$$

where x_a, x_p, x_n are the anchor, positive and negative samples; m is the margin, set to $m = 0.2$ in phase one and $m = 0.05$ in phase two, both values found empirically through validation. This is the same as equation 2.10 with $f_{shared}(f_{vector}(f_{T_e}(.)))$ and $f_{shared}(f_{raster}(f_{R_e}(.)))$ used as f_{branch^a} and $f_{branch^{pn}}$.

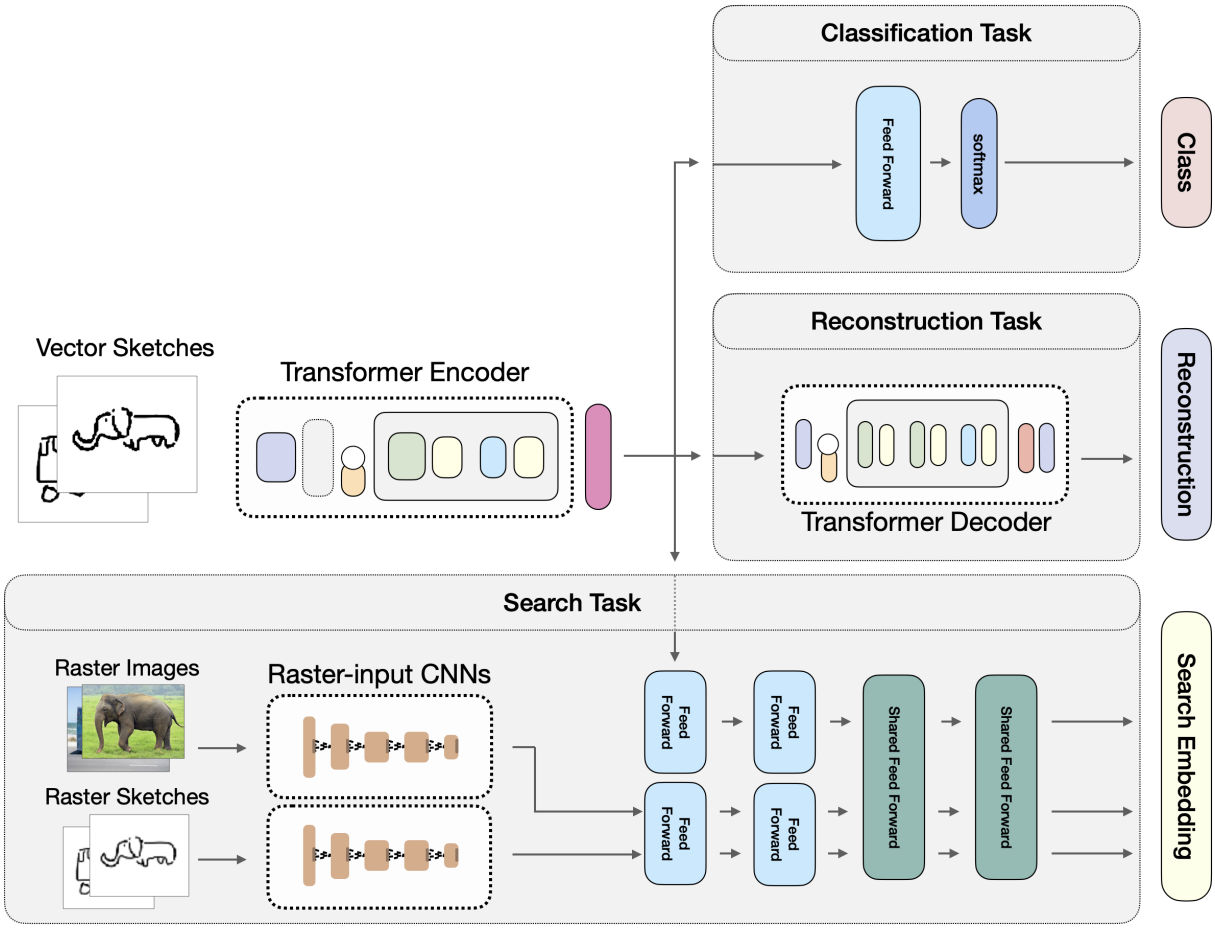


Figure 14 – Schematic showing how the learned sketch embedding is leveraged for sketch synthesis (reconstruction/interpolation), classification and cross-modal retrieval experiments (see encoder/embedding inset, refer to Fig. 12 for detail). Classification appends a fully-connected (fc) layer to the embedding space. Retrieval tasks require unification with a raster (CNN) embedding for images (BUI *et al.*, 2018a) via several fc layers trained via triplet loss.

3.4 Experiments and Discussion

We evaluate the performance of the proposed transformer embeddings for three common tasks; sketch classification, sketch reconstruction and interpolation, and sketch based image retrieval (SBIR). We compare against two baseline sketch embeddings for encoding stroke sequences; SketchRNN (HA; ECK, 2018) (also used for search in (XU *et al.*, 2018)) and LiveSketch (COLLOMOSSE; BUI; JIN, 2019). We evaluate using sketches from QuickDraw50M (THE..., 2018), and a large corpus of photos (Stock10M).

QuickDraw50M (THE..., 2018) comprises over 50M sketches of 345 object categories, crowd-sourced within a gamified context that encouraged casual sketches drawn at speed. Sketches are often messy and complex in their structure, consistent with tasks such as SBIR. We sample 2.5M sketches randomly with even class distribution from the public Quickdraw50M training partition to create training set (QD-2.5M) and use the public test partition of QuickDraw50M (QD-862k) comprising 2.5k $\times 345 = 862k$ sketches to evaluate our trained models. For SBIR and interpolation experiments we sort QD-862k by sequence length, and sample

three datasets (QD345-S, QD345-M, QD345-L) at centiles 10, 50 and 90 respectively to create a set of short, medium and long stroke sequences. Each of these three datasets samples one sketch per class at random from the centile yielding three evaluation sets of 345 sketches. We sampled an additional query set QD345-Q for use in sketch search experiments, using the same 345 sketches as LiveSketch (COLLOMOSSE; BUI; JIN, 2019). The median stroke lengths of QD345-S, QD345-M, QD345-L are 30, 47 and 75 strokes respectively (after simplification via RDP (DOUGLAS; PEUCKER, 1973)).

Stock67M is a diverse, unannotated corpus of photos used in prior SBIR work (COLLOMOSSE; BUI; JIN, 2019) to evaluate large-scale SBIR retrieval performance. We sample 10M of these images at random for our search corpus (Stock10M). This is a private dataset comprised of images from Adobe Stock and access to the images was provided by Adobe for this study.

3.4.1 Evaluating Sketch Classification

We evaluate the class discrimination of the proposed sketch embedding by attaching a fully connected layer to the transformer encoder stage, and training a 345-way classifier on QD2.5M. Table 1 reports the classification performance (using mean average precision, mAP) over QD-862k for each of the three proposed transformer embeddings, alongside two LSTM baselines – the SketchRNN (HA; ECK, 2018) and LiveSketch (COLLOMOSSE; BUI; JIN, 2019) variational autoencoder networks. Whilst all transformers outperform the baseline, the tokenised variant of the transformer based on dictionary learning (Tok-Dict) yields highest accuracy. We explore this further by shuffling the order of the sketch strokes retraining the transformer models from scratch. We were surprised to see comparable performance, perhaps due to the transformer's larger information scope rather than relying solely temporal information.

Method		mAP%
Baseline	LiveSketch (COLLOMOSSE; BUI; JIN, 2019)	72.93
	SketchRNN (HA; ECK, 2018)	67.69
Shuffled	Sketchformer (Cont.)	76.95
	Sketchformer (Tok-Grid)	76.22
	Sketchformer (Tok-Dict)	76.66
Proposed	Sketchformer (Cont.)	77.68
	Sketchformer (Tok-Grid)	77.36
	Sketchformer (Tok-Dict)	**78.34**

Table 1 – Sketch classification results over QuickDraw! (THE..., 2018) for three variants of the proposed transformer embedding, contrasting each to models learned from randomly permuted stroke order. Comparing to two recent LSTM based approaches for sketch sequence encoding (COLLOMOSSE; BUI; JIN, 2019; HA; ECK, 2018).

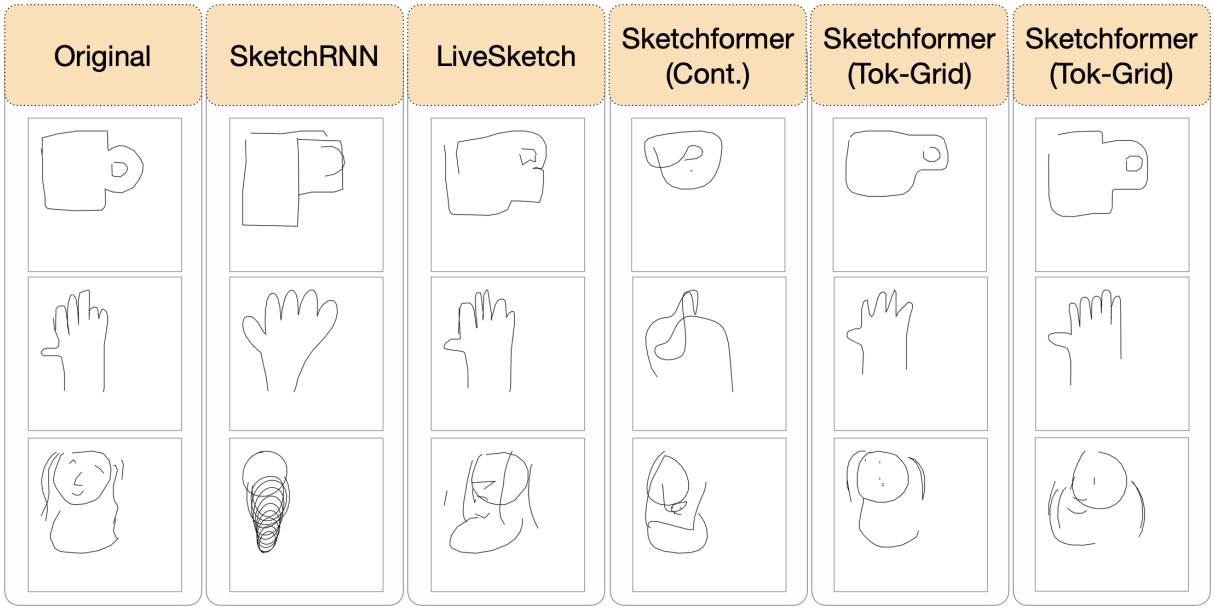


Figure 15 – Representative sketch reconstructions from each of the five embeddings evaluated in Table 2. (a) Original, (b) SketchRNN, (c) LiveSketch, (d) Sketchformer (Cont.), (e) Sketchformer (Tok-Grid) and (f) Sketchformer (Tok-Dict). The last row represents a hard-to-reconstruct sketch.

Method		Short	Mid	Long
Baseline	LiveSketch	**62.1**	**59.2**	27.8
	SketchRNN	4.05	3.70	1.38
Proposed	Sketchformer (Cont.)	0.00	0.00	0.00
	Sketchformer (Tok-Grid)	6.75	6.10	5.56
	Sketchformer (Tok-Dict)	24.3	28.4	**51.4**
	Uncertain	2.70	2.47	13.9

Table 2 – User study quantifying accuracy of sketch reconstruction. Preference is expressed by 5 independent workers, and results with $> 50\%$ agreement are included. Experiment repeated for short, medium and longer stroke sequences. For longer sketches, the proposed transformer method Sketchformer (Tok-Dict) is preferred.

3.4.2 Reconstruction and Interpolation

We explore the generative power of the proposed embedding by measuring the degree of fidelity with which: 1) encoded sketches can be reconstructed via the decoder to resemble the input; 2) a pair of sketches may be interpolated within, and synthesised from, the embedding. The experiments are repeated for short (QD345-S), medium (QD345-M) and long (QD345-L) sketch complexities. We assess the fidelity of sketch reconstruction and the visual plausibility of interpolations via Amazon Mechanical Turk (MTurk). MTurk workers are presented with a set of reconstructions or interpolations and asked to make a 6-way preference choice; 5 methods and a 'cannot determine' option. Each task is presented to five unique workers, and we only include results for which there is $> 50\%$ (*i.e.*, > 2 worker) consensus on the choice.

Reconstruction results are shown in Table 2 and favour the LiveSketch (COLLOMOSSE;

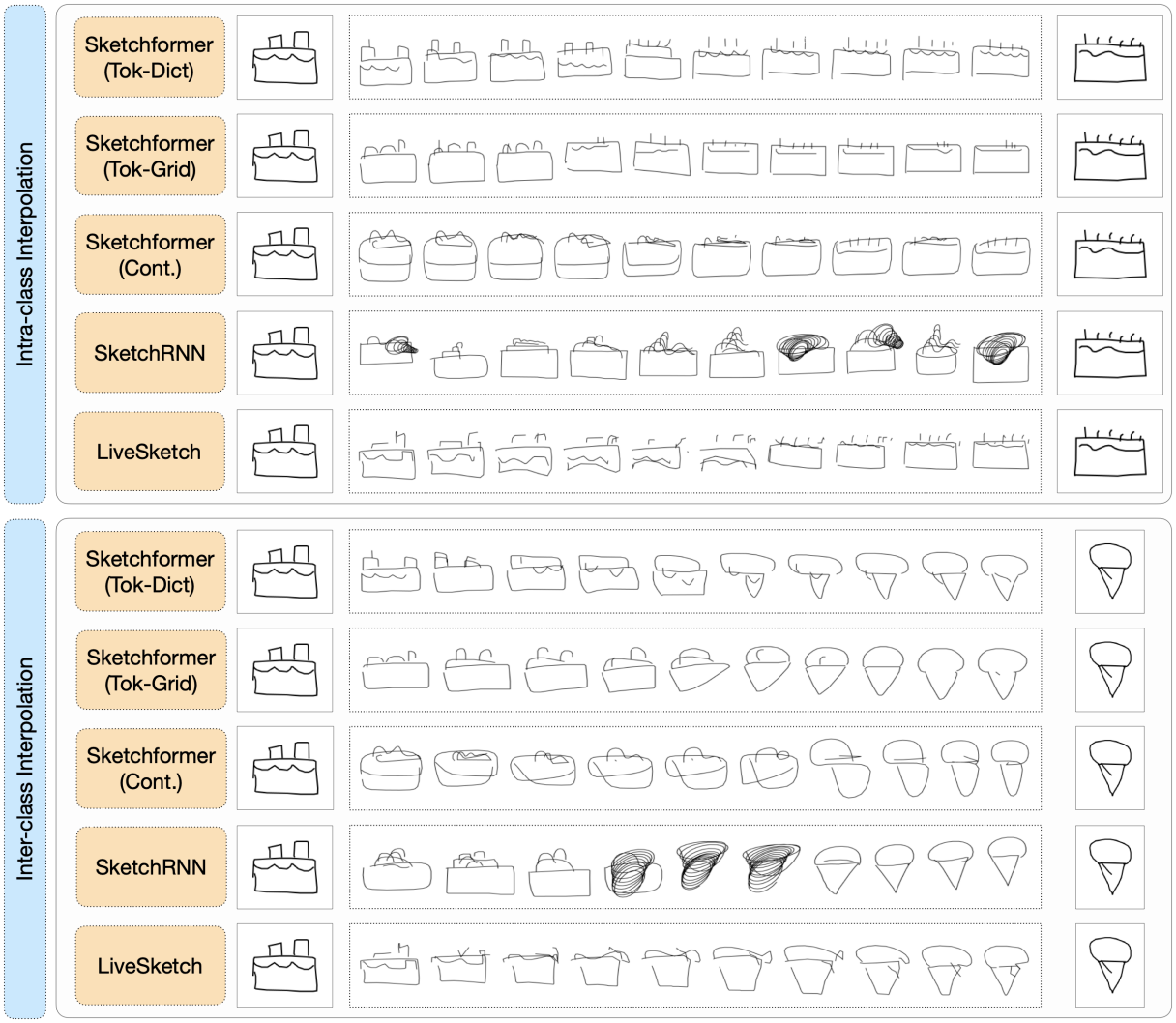


Figure 16 – Representative sketch interpolations from each of the five embeddings evaluated in Table 3. The first block shows intra-class interpolation from 'birthday cake' to 'ice-cream' and the second block intra-class interpolation between two 'birthday cakes'.

BUI; JIN, 2019) embedding for short or medium length strokes, with the proposed tokenised transformer (Tok-Dict) producing better results for more complex sketches aided by the improved representational power of transformer for longer stroke sequences. Fig 15 provides representative visual examples for each sketch complexity.

We explore interpolation in Table 3 blending between pairs of sketches within (intra-) classes and between (inter-) classes. In all cases we encode sketches separately to the embedding, interpolate via slerp (after Ha and Eck (2018), Collomosse, Bui and Jin (2019) in which slerp was shown to offer best performance), and decode the interpolated point to generate the output sketch. Fig. 16 provides visual examples of inter- and intra-class interpolation for each method evaluated. In all cases the proposed tokenised transformer (Tok-Dict) outperforms other transformer variants and baselines, although the performance separation is narrower for shorter strokes echoing results of the reconstruction experiment. The stability of our representation is further demonstrated via local sampling within the embedding in Fig. 17.

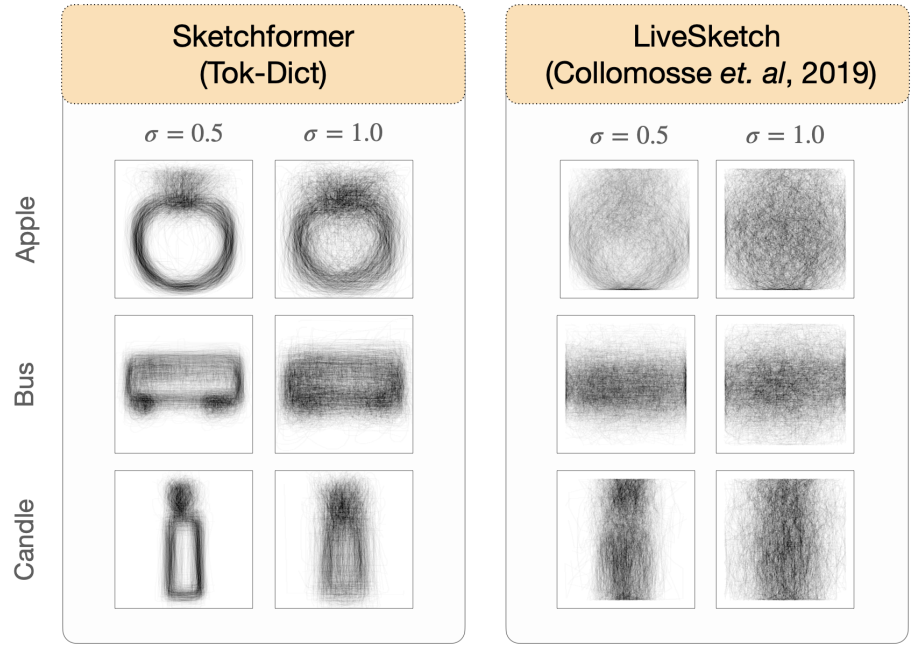


Figure 17 – Visualisation of sketches reconstructed from mean embedding for 3 object categories. We add Gaussian noise with standard deviation $\sigma = 0.5$ and $\sigma = 1.0$ to the mean embedding of three example categories on the Quickdraw test set. The reconstructed sketches of Sketchformer (Tok-Dict) retain salient features even with high noise perturbation.

Partition	Embedding	Intra-	Inter-
Short (10th cent.)	SketchRNN (HA; ECK, 2018)	0.00	2.06
	LiveSketch (COLLOMOSSE; BUI; JIN, 2019)	25.8	30.9
	Sketchformer (Cont.)	14.0	6.18
	Sketchformer (Tok-Grid)	19.4	17.5
	Sketchformer (Tok-Dict)	**31.2**	**33.0**
	Uncertain	8.60	10.3
Mean (50th cent.)	SketchRNN (HA; ECK, 2018)	0.00	0.00
	LiveSketch (COLLOMOSSE; BUI; JIN, 2019)	25.2	20.2
	Sketchformer (Cont.)	15.6	16.0
	Sketchformer (Tok-Grid)	15.6	19.1
	Sketchformer (Tok-Dict)	**35.8**	**35.1**
	Uncertain	7.36	9.57
Long (90th cent.)	SketchRNN (HA; ECK, 2018)	0.00	0.00
	LiveSketch (COLLOMOSSE; BUI; JIN, 2019)	25.0	21.1
	Sketchformer (Cont.)	12.5	8.42
	Sketchformer (Tok-Grid)	16.7	10.5
	Sketchformer (Tok-Dict)	**40.6**	**50.5**
	Uncertain	5.21	9.47

Table 3 – User study quantifying interpolation quality for a pair of sketches of same (intra-) or between (inter-) classes. Preference is expressed by 5 independent workers, and results with $> 50\%$ agreement are included. Experiment repeated for short, medium and longer stroke sequences.

3.4.3 Cross-modal Matching

We evaluate the performance of Sketchformer for sketch based retrieval of sketches (S-S) and images (S-I).

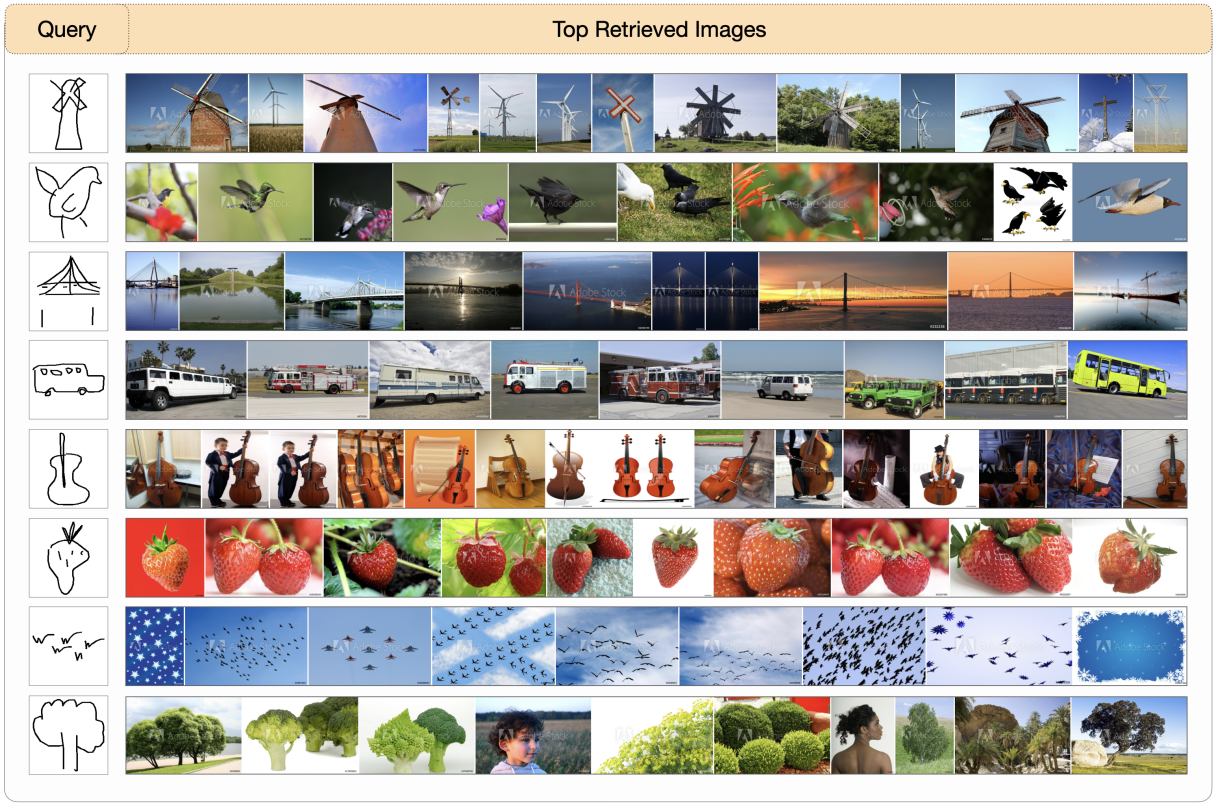


Figure 18 – Representative visual search results over Stock10M indexed by our proposed embedding (Sketchformer (Tok-Dict)) for a vector sketch query. The two bottom rows ('animal migration' and 'tree') are failure cases.

Sketch2Sketch (S-S) Matching. We quantify the accuracy of retrieving sketches in one modality (raster) given a sketched query in another (vector, *i.e.*, stroke sequence) – and vice-versa. This evaluates the performance of Sketchformer in discriminating between sketched visual structures invariant to their input modality. Sketchformer is trained on QD-2.5M and we query the test corpus QD-826k using QD-345Q as the query set. We measure overall mean average precision (mAP) for both coarse grain (*i.e.*, categorical) and fine-grain (*i.e.*, instance-level) similarity. As explained in Section 2.1, for the former we consider a retrieved record a match if it matches the sketched object class. For the latter, exactly the same single sketch must match (in its different modality). To run raster variants, a rasterised version of QD-862k (for V-R) and of QD345-Q (for R-V) is produced by rendering strokes to a 256×256 pixel canvas using the CairoSVG Python library. Table 4 shows that for both class and instance level retrieval, the R-V configuration outperforms V-R. Given the gap in sample size between query set QD345-Q (345 samples) and test set QD-862k (862k samples), we hypothesise this performance gain comes from encoding the larger set with the better representation from Sketchformer. In contrast to other experiments reported, the continuous variant of Sketchformer appears slightly preferred, matching higher for early ranked results for the S-S case – see Fig. 19 for category-level precision-recall curve. Although Sketchformer outperforms RNN baselines by 1-3% in the V-R case the gain is more limited and indeed the performance over baselines is equivocal in the S-S where the search index is formed of rasterised sketches.

Method		Instance	Category
Livesketch (COLLOMOSSE; BUI; JIN, 2019)	V-R	6.71	20.49
	R-V	7.15	20.93
Sketchformer (Cont.)	V-R	5.29	22.22
	R-V	6.21	**23.48**
Sketchformer (Tok-Grid)	V-R	6.42	21.26
	R-V	**7.38**	22.10
Sketchformer (Tok-Dict)	V-R	6.07	21.56
	R-V	7.08	22.51

Table 4 – Quantifying the performance of Sketch2Sketch retrieval under the LiveSketch RNN baseline and three proposed variants. We report category- and instance-level retrieval (mAP%).

Sketch2Image (S-I) Matching. We evaluate sketch based image retrieval (SBIR) over Stock10M dataset of diverse photos and artworks, as such data is commonly indexed for large-scale SBIR evaluation (COLLOMOSSE *et al.*, 2017; COLLOMOSSE; BUI; JIN, 2019). We compare against the state of the art SBIR algorithms: LiveSketch from Collomosse, Bui and Jin (2019) that takes in vector input as we do, and the Multi-stage Learning (MSL) model from Bui *et al.* (2018a), that was designed for rasterised sketch inputs. Since no ground-truth annotation is possible for this size of corpus, we crowd-source per-query annotation via Mechanical Turk (MTurk) for the top-15 results and compute both mAP@k and precision@k across all QD345-Q query sketches, both with varying $k = [1 - 15]$. Table 5 compares performance of our tokenised variants to these baselines, alongside associated Precision@k curves in Fig. 19. The proposed dictionary learned transformer embedding (Tok-Dict) delivers the best performance (visual results in Fig. 18).

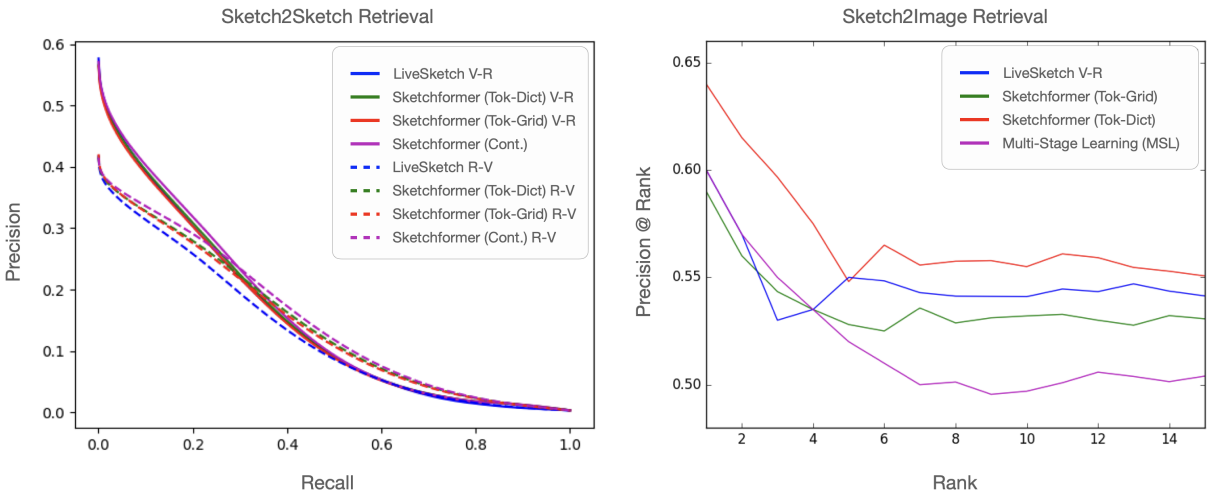


Figure 19 – Quantifying search accuracy. We show Sketch2Sketch performance via precision-recall (P-R) curves for Vector-2-Raster and Raster-2-Vector category-level retrieval. And Sketch2Image (SBIR) accuracy is quantified via precision @ k=[1,15] curve over the Stock10M dataset.

Method		mAP%
Baseline	LiveSketch (COLLOMOSSE; BUI; JIN, 2019)	54.80
	MSL (BUI *et al.*, 2017)	51.97
Proposed	Sketchformer (Tok-Grid)	53.75
	Sketchformer (Tok-Dict)	**56.96**

Table 5 – Quantifying accuracy of Sketchformer for Sketch2Image search (SBIR). We compare against an RNN-based baseline (COLLOMOSSE; BUI; JIN, 2019) and a CNN-based one (BUI *et al.*, 2018a). Mean average precision (mAP) computed to rank 15 over Stock10M for the QD345-Q query set.

3.5 Conclusion

In this chapter (and in the original publication (RIBEIRO *et al.*, 2020)) we presented Sketchformer: a learned representation for sketches based on the Transformer architecture (VASWANI *et al.*, 2017). While designing Sketchformer we were able further the field of representation learning for sketches, applying this unified representation to classification, retrieval and sketch synthesis.

Several variants were explored using continuous and tokenised input; a dictionary learning based tokenisation scheme delivers classification performance gains of 6% on previous LSTM autoencoder models (SketchRNN and derivatives). We showed interpolation within the embedding yields plausible blending of sketches within and between classes, and that reconstruction (auto-encoding) of sketches is also improved for longer, more complex sketches.

The success of Sketchformer as a generative model is connected to how well it is able to semantically represent sketches, as shown by the effectiveness of the learned representations as a basis for indexing sketch and image collections for sketch based visual search (Hypothesis **H4**).

Future work on Sketchformer should further explore our continuous representation variant, or other variants with more symmetric encoder-decoder structures similar to the BERT model from Devlin *et al.* (2019). We have demonstrated the potential for Transformer networks to learn a multi-purpose representation for sketch, but believe many further applications of Sketchformer exist beyond the three tasks studied here.

Finally, in the original manuscript we wrote another suggestion for future work:

> [A Transformer's] fusion with additional modalities might enable sketch driven photo generation (PARK *et al.*, 2019a) using complex sketches, or with a language embedding for novel sketch synthesis applications. (RIBEIRO *et al.*, 2020)

which was the seed for Scene Designer, a study presented in Chapter 4, exploring image retrieval and synthesis from composite sketches.

CHAPTER

4

SCENE DESIGNER: COMPOSITIONAL SKETCH-BASED IMAGE RETRIEVAL

In this chapter we present Scene Designer, a novel method for Compositional Sketch-based Image Retrieval (CSBIR). With this model, we show that combining a synthesis task with representation learning not only boosts retrieval performance, but also enables new creative workflows. Scene Designer was first presented at the Sketching for Human Expressivity Workshop on the CVF International Conference on Computer Vision (ICCV 2021):

RIBEIRO, L. S. F.; BUI, T.; COLLOMOSSE, J.; PONTI, M. Scene designer: a unified model for scene search and synthesis from sketch. In: **Proc. ICCV WS**. [S.l.: s.n.], 2021. p. 2424–2433.

The study was then expanded, and the extended version's manuscript published on Springer's Multimedia Tools and Applications (MTAP), Special Issue on Content-Based Image Retrieval:

RIBEIRO, L. S. F.; BUI, T.; COLLOMOSSE, J.; PONTI, M. Scene designer: Compositional sketch-based image retrieval with contrastive learning and an auxiliary synthesis task. **Multimedia Tools and Applications**, Springer, 2022.

While most studies on sketch-based retrieval follow the thread of single-object retrieval, we argue that for most applications it is desirable to search complete scenes with multiple objects, increasing the flexibility and potential specificity of each query.

To make Scene Designer possible we've designed a way of synthesising hard negative samples for contrastive learning with scenes and we show that combining our contrastive training with an image synthesis task improves the semantic scene representations ever more.

The Scene Designer study was among the first few to address Compositional SBIR, yet it remains the state-of-the-art at the time of writing, and when published achieved more than double the recall@1 metric previously presented by competing methods. Finally, together with

the model we have introduced the first large-scale sketched scene dataset and benchmark, called QuickDrawCOCO.

In this chapter we present a revised version of the manuscript published on Multimedia Tools and Applications (reproduced with permission from Springer Nature) and highlight the following contributions to the thesis:

- We take on Compositional SBIR (CSBIR), a new task within SBIR where multiple sketched objects can compose a scene query making SBIR more applicable and useful.
- Scene Designer'a architecture composes a traditional multi-stream network design with a Graph Neural Network and Transformer to learn aggregate representations for objects in a scene. (Hypothesis **H2**)
- A key finding of the study is the clear benefit of combining generative modelling and representation learning in retrieval systems. (Hypothesis **H4**)

4.1 Introduction

As we have become used to having great cameras always in our pockets, the number of new images taken or made every day has skyrocketed. With the wealth of content the problem of retrieval becomes ever more pressing; it is natural then that research on Sketch-based Image Retrieval (SBIR) has similarly boomed within the last decade (WANG; KANG; LI, 2015; QI *et al.*, 2016; BUI *et al.*, 2017; SANGKLOY *et al.*, 2016a; BUI *et al.*, 2018a). Instead of using text queries to search for images in SBIR one searches by drawing the desired object, which is more intuitive and flexible, allowing the user to express desired object pose and specific shapes without having to have a similar image at hand (as would be the case for the Content-based Image Retrieval task).

While booming, the field of SBIR has seldom explored retrieval beyond single-object sketches and images save for one recent publication (LIU *et al.*, 2020). While this limitation is acceptable for e-commerce retrieval, it impedes the use of SBIR for large scale general retrieval, creative workflows such as reference retrieval and doesn't even allow for search in personal image collections; in these scenarios, the photographs are rarely of single objects and instead represent complete scenes with multiple interacting objects and people depicted. We sought then to design a model for Compositional SBIR, enabling users to sketch scenes with multiple objects and choose their positions as well as shapes and poses.

This paper presents Scene Designer; a model for searching scenes using free-hand sketches of scene compositions. Our model provides a cross-modal search embedding where similarity of visual compositions comprising sketches and images (or mixtures of both) can be measured. The training task is comprised of a novel take on contrastive learning together with a

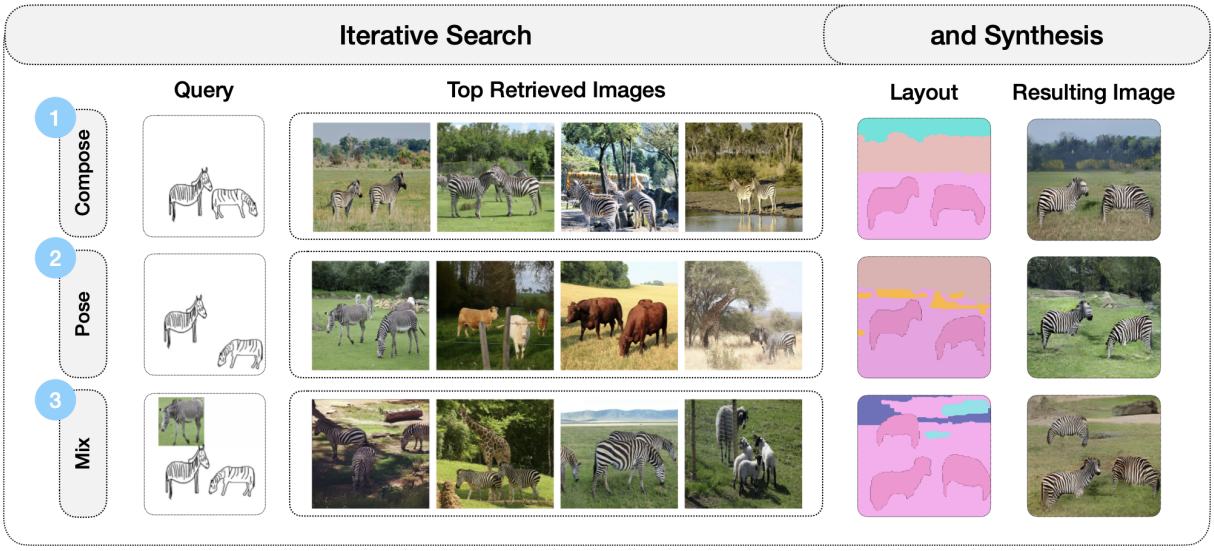


Figure 20 – Scene Designer enables iterative design of image compositions through retrieval and synthesis. In (1) the user sketches initial concept, matching scenes are retrieved; then (2) the user refines the search by reposing the objects and finally in (3) a retrieved object is used to add to the composition, mixing domains. For each retrieval step we also show the layout synthesised by Scene Designer and the resulting generated image.

generative task for synthesising scene layouts, leading to a retrieval performance that more than doubles the previous state-of-the-art (SoTA). Our technical contributions are:

1. **Compositional Sketch Search**. We propose a hybrid graph neural network (GNN) and Transformer architecture to learn a metric search embedding for comparing sketched and photographic scenes. Existing sketch based image retrieval (SBIR) methods predominantly match queries containing a single, dominant object invariant to its position within an image. Our novel contrastive training matches sketched compositions containing multiple objects.

2. **Fusion of Synthesis and Cross-Domain Search**. Training a single mixed-task model yields improvements – more than doubling recall@1 – on the state of the art for scene search whilst providing competitive scene synthesis results. We also enable a novel creative workflow in which image compositions may be created iteratively; users sketch an initial composition (one or more objects), and elements of the retrieved images may be incorporated potentially with further sketched objects to drive either further searches or full image synthesis. Fig. 20 demonstrates how our fused model enables iterative design of generative compositions. We argue that this novel workflow shows the potential of combining both tasks beyond improving performance and can be applied to other unified models as well.

3. **New Large Scale Dataset of Sketched and Photographic Scenes**. We designed a new dataset to train our model, called QuickDrawCOCO. By synthetically combining the descriptive segmentation labels of COCO-stuff with the large Quick! Draw sketch dataset,

we introduce the largest (by 8x) and most class-diverse (with 92 classes) dataset of its kind.

4.2 Related Work

Sketch based image retrieval (SBIR). We have covered in Chapters 1 and 3 how SBIR evolved from "shallow" hand-crafted features combined with dictionary-learning (EITZ *et al.*, 2011; HU; COLLOMOSSE, 2013) to deep representation learning driven by multi-stream designs (WANG; KANG; LI, 2015; QI *et al.*, 2016; BUI *et al.*, 2017; BUI *et al.*, 2018a). This Chapter addresses our take on a natural extension from single object sketches to scenes with multiple objects: Compositional SBIR; the task was first introduced by Liu *et al.* (2020), making their study the closest in task to ours. Their model is however dependent on known category labels of photos and sketches, a dependency we amend in our design while also implementing our own take on a GNN encoder.

Conditional Image Generation has been a rapidly developing field since the introduction of Conditional Generative Adversarial Networks (GANs) (MIRZA; OSINDERO, 2014). Initial work exploring class label priors were soon followed by models with image based priors such as Pix2Pix (Isola *et al.*, 2017) and more recently high definition synthesis models such as Pix2PixHD (WANG *et al.*, 2018), SPADE (PARK *et al.*, 2019b) and OASIS (SCHÖNFELD *et al.*, 2021). Those models learn from pairs of matching samples from each domain, and are capable of mapping semantic layouts to images. We make use of one such model (PARK *et al.*, 2019b) to show how combining the tasks of search and synthesis can enable the new creative workflow seen in Fig. 20, where our model's semantic layout is used to drive the conditional generator.

Text-Conditioned Image Generators have been on the spotlight lately. Led by the combined leaps in large language models (BROWN *et al.*, 2020) and image synthesis with Diffusion Probabilistic Models (DHARIWAL; NICHOL, 2021; SOHL-DICKSTEIN *et al.*, 2015, 2015), unified text and image models such as CLIP (RADFORD *et al.*, 2021), Dall-E (RAMESH *et al.*, 2021) and Stable Diffusion (ROMBACH *et al.*, 2021) are being used even beyond research for creative applications online. Our model, while not competing in image quality, overcomes a shortcoming of these models: the lack of fine pose and placement control when using only text as input, requiring users to finetune prompts and even share "prompt tutorials" online. Given Scene Designer's versatility regarding the image generator used (we use it as a black box model), we find it is trivial to improve the image quality of our model by substituting the SPADE model for a newer Diffusion based Layout-to-Image generator (*e.g.* the model by Wang *et al.* (2022)).

Scene Graphs are a representation for images where individual objects are defined as nodes with the graph edges describe their relationships. They were first used for text-based image retrieval by Johnson *et al.* (2015), where they showed that representations derived from scene

graphs were considerably more robust at representing user intent than low-level image features or simple object lists; we have taken the insights of Johnson *et al.* (2015) at heart and designed our model to interpret object-level features through Scene Graphs, using a Graph Neural Network to learn correlations object- and image-wise. We differ in the approach to retrieval, as Johnson *et al.* (2015) compares images' graphs to each other with a learned conditional random field and our approach uses aggregated vector representations and simple euclidean distance.

Given its success for retrieval, Scene Graphs are now often in the literature as learning *targets*, with researchers training models capable of extracting Scene Graphs from natural language (ZHU *et al.*, 2022; GU *et al.*, 2019) and directly from images (QI *et al.*, 2019). Our approach takes the ground truth scene graph from images as input, which means we assume a full system would either have the user specify the scene graph or have it extracted with models such as the one from Qi *et al.* (2019). The Scene Graphs we use is simplified and could also be synthesised from Object Detection (ZAIDI *et al.*, 2022) models' bounding boxes.

Finally, Scene Graphs were also used by Johnson, Gupta and Fei-Fei (2018) and Ashual and Wolf (2019) as initial representations to drive semantic layout generation and image synthesis based on those layouts. Our approach follows a similar principle for encoding scene graphs into object representations which then drive layout synthesis. The main difference of course is that we aggregate these vectors to represent the entire scene and perform the CSBIR task.

Image-synthesis from Sketch is a challenging problem due to the abstract and ambiguous nature of most sketch applications; the application of facial sketch-photo synthesis for example is well developed (WANG *et al.*, 2013; PENG *et al.*, 2015; XU; JOSHI; BRESSON, 2021) in the literature thanks to its application in forensics and the use of realistic and unambiguous sketches. Synthesis from abstract sketches on the other hand have been approached by methods such as Sketchy-GAN (CHEN; HAYS, 2018) and ContextualGAN (LU *et al.*, 2018), which achieved promising results generating images from single-object sketches. The rising Diffusion Models for image synthesis are recently impacting sketch-photo synthesis as well, with Voynov, Aberman and Cohen-Or (2022) showing a new State-of-the-Art.

However, as was shown by Gao *et al.* (2020), the aforementioned methods cannot accommodate sketched scenes with multiple objects. Gao *et al.* (2020) were the first to implement image synthesis from sketch scenes and introduced the paired SketchyCOCO dataset. We have used the task of layout synthesis from sketched compositions to drive better scene representations which in turn leads to improved CSBIR; our model does nonetheless generate semantic layouts that are aligned with the current SoTA.

Joint image search and synthesis has been tried in early work such as Sketch2Photo (CHEN *et al.*, 2009) that composes an output image from objects found in the retrieved results. As we've shown in Chapter 3, Sketchformer (RIBEIRO *et al.*, 2020) also harvests the search and synthesis combination but only for a single domain, sketch. Similarly, Pang *et al.*, (PANG *et al.*, 2017) uses a sketch decoder to assist their cross-domain learning, showing the usefulness of mixing

both to yield better representations. This combination has also been used within the constraints of Zero-Shot SBIR (DUTTA; AKATA, 2020; SHEN *et al.*, 2018; DUTTA; SINGH; BISWAS, 2021; PANDEY *et al.*, 2020; YELAMARTHI *et al.*, 2018b), further cementing that including a synthesis task helps with model generalisation; each of those studies and the one presented in this chapter supports our Hypothesis (**H4**).

Closest to our suggested creative workflow is the work of Yu & Kovashka (YU; KOVASHKA, 2020), where the generated images are used to gather user feedback and improve the retrieval result; their method however is only text-conditioned. Our method learns a cross-domain embedding for scene sketches and images that is improved upon by the addition of layout synthesis in the framework; furthermore we show that images generated from our layouts are comparable to synthesis-only models.

4.3 Methodology

We developed a cross-modal representation learning framework in order to represent images and sketches with multiple objects in a common feature space; this representation is used to drive both cross-modal retrieval and layout generation. Our model learns representations for individual objects first, introduces scene-level information to these representations, and then finally aggregates those into a vector representing the entire scene. Fig. 21 describes the architecture and stages of the representation, which are each summarised below:

1. **Object-level representation (OLR):** Given an input composition, we encode individual objects (which might be sketched or photo-real) to a common representation. We refer to this embedding, in which each object is independently represented, as the OLR (subsec. 4.3.1).

2. **Scene Graph (SG)**: is constructed using the OLR to encode objects as graph nodes while discrete positional relationships of object pairs are used to build the edges.

3. **Scene-aware OLR (SA-OLR):** A graph neural network (GNN) re-encodes object appearances (the OLR, nodes in the Scene Graph) together with their pairwise relationships (subsec. 4.3.2). This representation is learned end-to-end to be aggregated for retrieval and used as input to the semantic layout generator .

4. **Scene representation (SR):** The SA-OLR vectors encode each object in the scene, to obtain the unified Scene Representation those vectors are aggregated by a function $r(.)$. We've explored three alternative designs for $r(.)$: (a) simple averaging, (b) self-attention (SUKHBAATAR *et al.*, 2015b) and (c) a small Transformer network; we have opted for the latter option as the most robust one and use it for all experiments when not otherwise mentioned (See Section 4.3.3 for details). This single-vector representation is driven by contrastive learning and is later used for retrieval. (See Section 4.3.4)

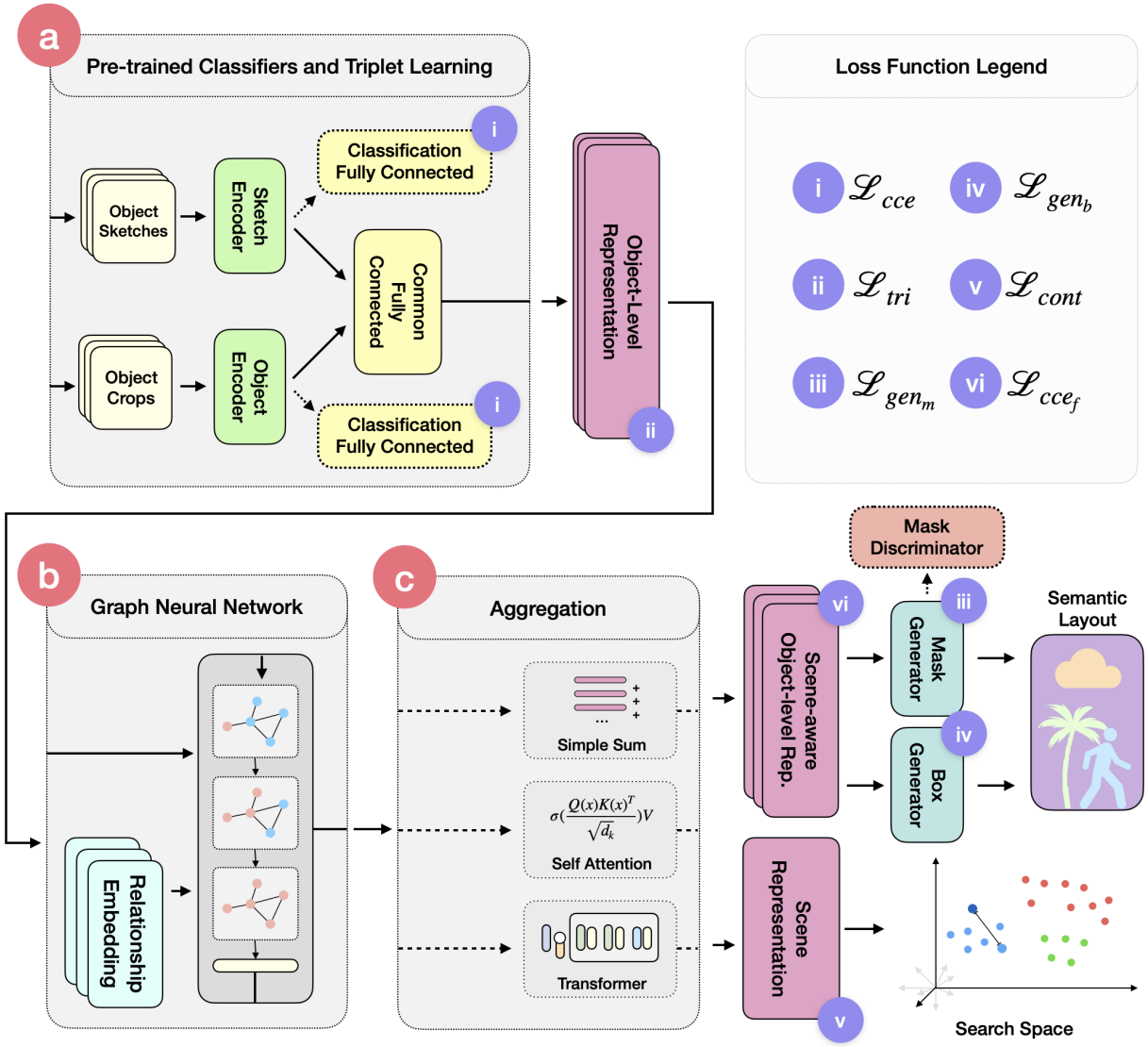


Figure 21 – Proposed architecture of Scene Designer. At stage (a) individual objects of each scene are encoded independent of composition to produce the *object-level representation*. A Scene Graph is constructed from these and encoded via GNN (b) to produce *scene-aware object-level representations*, which are subsequently aggregated (c) with the use of simple sum, self-attention or a small transformer to produce the *scene representation*. The roman numerals indicate where each of the losses (see Sec. 4.3.1, 4.3.5 and 4.3.4) are applied during training: (i) Cross-entropy classification $\mathscr{L}_{CCE}$, (ii) Triplet loss $\mathscr{L}_{tri}$, (iii) Mask generation losses $\mathscr{L}_{gen_m}$, (iv) Bounding box generation losses $\mathscr{L}_{gen_b}$, (v) Contrastive loss $\mathscr{L}_{cont}$, (vi) Final cross-entropy classification $\mathscr{L}_{CCE_f}$.

As input, our model takes for each scene the individual sketched/photographic objects and their positions (which can be coarsely defined). In a real scenario, a user would define these coarse positions or the system could employ an off-the-shelf Object Detector to do the same. These object positions are used to build the Scene Graph. We now describe in further detail how each stage of the representation is learned.

4.3.1 Object-level Representation (OLR)

We begin by independently encoding each object within the input composition to a common feature embedding, regardless of its input modality. Following contemporary single-object SBIR (BUI *et al.*, 2018a; SANGKLOY *et al.*, 2016a; YU *et al.*, 2016) we adopt a deep metric learning approach using a heterogeneous triplet architecture. This means we have separate branches for each modality – sketch and photo – and the two branches are used to compute representations for one triplet at a time composed of one sketch anchor (a) and a positive (same class, p) and negative (different class, n) image . Our network comprises a MobileNet (HOWARD *et al.*, 2017) backbone (subsec. 4.5), terminated with two shared (as opposed to the heterogeneous backbone) fully connected (fc) layers, the latter yielding the OLR embedding with 256 dimensions.

The MobileNet is pre-trained on ImageNet (RUSSAKOVSKY *et al.*, 2015), and finetuned for cross-domain learning using a combined triplet and categorical cross-entropy (CCE) losses, with the triplet loss following the classical formulation (Equation 2.10):

$$\begin{aligned}\mathscr{L}_{tri}(x_a^{skt}, x_p^{img}, x_n^{img}) = \max\{0, m + (f_{skt}(x_a^{skt}) - f_{img}(x_p^{img}))^2 \\ - (f_{skt}(x_a^{skt}) - f_{img}(x_n^{img}))^2\}\end{aligned} \tag{4.1}$$

where $x_a^{skt}, x_p^{img}, x_n^{img}$ are the anchor object sketch and the positive and negative object images; $f_{skt}(.)$ is the MobileNet that encodes sketched objects, $f_{img}(.)$ the one that encodes photo-real objects cropped from images, each followed by the shared fc layers that encodes the MobileNet's outputs to the Object Level Representation; $m = 0.5$ is the margin. Hereafter, we use shorthand $f_{olr}(.)$ to refer to the object encoder irrespective of its modality. The Object-level Representation (OLR) is trained using only the objects of each scene. The objects are represented by rasterised sketches (anchor a), and objects cropped from the COCO-stuff dataset with corresponding (positive p) and differing (negative n) object class; see subsec. 4.4.1 for dataset details.

To aid convergence (BUI *et al.*, 2017), the equal-weighted cross-entropy loss $\mathscr{L}_{cce}$ is applied to the output of a further fc layer $f_{cls}(.)$, appended to the backbone networks to project the object representations into the class scores' space:

$$\mathscr{L}_{cce}(x_a, x_p, x_n) = \sum_{c \in \{a,p,n\}} y_c log(\sigma(f_{cls}(x_c))) \tag{4.2}$$

where y_a, y_p, y_n are the one-hot vectors encoding the semantic class of each object and σ is the normalising softmax function.

4.3.2 Scene-aware Object-level Representation (SA-OLR)

A scene graph (SG) describes objects and their spatial relationships within the composition. Formally an SG is a graph $\mathscr{G}_{sgsg} = (\mathscr{V}_{sg}, \mathscr{E}_{sg})$ where $\mathscr{V}_{sg} = \{o_1, \cdots, o_i, \cdots, o_k\}$ is set of nodes representing the k objects, and $(o_i, r_{ij}, o_j) \in \mathscr{E}_{sg}$ the set of edges encoding discrete relationships,

with the relationships $r_{ij} \in \mathscr{R}_{sg}$ encoded by type. We have followed Ashual and Wolf (2019) in their Scene Graph design and used "left of", "right of", "above", "below", "contains" and "inside of" as our discrete relationships $\mathscr{R}_{sg}$. These relationships are determined analytically at run time by using the object's coarse positions; we build our graphs to be fully connected (every node is connected to every other node both as a relationship object and subject *e.g.* incoming and outgoing edge).

The Scene Graph is encoded via a graph neural network (GNN). We represent each node using the Object-level Representation (OLR), $f_{olr}(o_i)$, encoding the pose and appearance of object i. As was done by Ashual and Wolf (2019), we adopt a learnable embedding $f_{emb}(r_{ij})$ to map relationships $\mathscr{R}_{sg}$ into dense vectors, completing the starting vectors:

$$\begin{aligned} v_i^0 &= f(o_i) \\ r_{ij}^0 &= f_{emb}(r_{ij}) \\ v_j^0 &= f(o_j) \end{aligned} \tag{4.3}$$

where the i and j subscripts refer to the subject and object of the relationship r_{ij}, and v_i^0, v_j^0 are the starting node representations. The GNN $g_r(.)$ comprises a sequence of 4 layers. In each we follow the architecture of Johnson, Gupta and Fei-Fei (2018) (which in turn follows the generic design of (BATTAGLIA *et al.*, 2018)) and implement three fully-connected (fc) layers: the relationship fc layer f_{rel}, the outgoing fc layer f_{out} and the incoming fc layer f_{in}. Each relationship representation is updated through f_{rel}:

$$r_{ij}^{l+1} = f_{rel}^l(< v_i^l, r_{ij}^l, v_j^l >) \tag{4.4}$$

where $< . >$ represents concatenation, l the layer index. Because the graph is fully connected, each node will participate in $N = 2(K-1)$ relationships, as object and subject. We compute new vectors for each relationship by taking the average node vector across all relationships:

$$\hat{v}_i^{l+1} = \frac{\sum_{j=1, j\neq i}^{K} f_{in}^l(< v_i^l, r_{ij}^l, v_j^l >) + \sum_{j=1, j\neq i}^{K} f_{out}^l(< v_j^l, r_{ji}^l, v_i^l >)}{2(K-1)} \tag{4.5}$$

And finally, this aggregated $\hat{v}_i^{l+1}$ goes through a final projection with an fc layer f_{agg}:

$$v_i^{l+1} = f_{agg}^l(\hat{v}_i^{l+1}) \tag{4.6}$$

These layers are shared between images and sketches and trained end-to-end as part of our model. We call the output of this module the *Scene-aware* Object-level Representations (SA-OLR), the 256-D vectors that describe each individual object in a scene.

4.3.3 Scene Representation (SR)

To obtain a final representation for each scene, the individual Scene-aware Object-level Representations need to be aggregated into a single vector. We consider three options for aggregation: (a) Simple Averaging, (b) Self-attention or (c) a small 3-layered Transformer.

With **Simple Averaging**, the Scene-aware Object-level Representations (SA-OLR), $g_r(f(o_i))$, are averaged per scene. Given a scene graph $\mathscr{G}_{sg}$ with set of nodes $\mathscr{V}_{sg} = \{o_1, \cdots, o_i, \cdots, o_k\}$, we compute:

$$r(\mathscr{G}_{sg}) = \frac{\sum_{i=0}^{k} g_r(f(o_i))}{k} \tag{4.7}$$

to obtain the Scene Representation for $\mathscr{G}_{sg}$. This is by far the easiest aggregation solution and is already enough for a well rounded model as we explore in Section 4.5.

Self-attention can be considered a variant of the averaging solution, except here we use the obtained attention matrix to perform a weighted-average of the SA-OLR vectors from each scene. This is the same self-attention we have used in Chapter 3, Equation 3.7 for Sketchformer. Our design again follows Sukhbaatar *et al.* (2015a). The attention matrix is obtained by using two learnable projection matrices K and V in the place of *keys* and *values* in the attention mechanism:

$$s = \sigma((tanh(G_r K^T + b)V) \tag{4.8}$$

where $G_r = \{g_r(f_{olr}(o_1)), g_r(f_{olr}(o_2)), \cdots, g_r(f_{olr}(o_k))\}$. The learnable bias b allows the model to learn a translation on top of the vectors; V aggregates the resulting square matrix into the $s = \{s_1, \cdots, s_i, \cdots, s_k\}$ attention weights; The resulting attention weights behave as a probability function that weights the $g_r(f(o_i))$ values:

$$r(\mathscr{G}_{sg}) = \sum_{i=0}^{k} s_i g_r(f_{olr}(o_i)) \tag{4.9}$$

We show through experiments in Section 4.5 how **Simple averaging** or a **Self-Attention layer** are both simpler to implement yet competitive options. For other experiments we have however chosen to use our third option, the small **3-layered Transformer** (VASWANI *et al.*, 2017) encoder, for its improved robustness. This approach was inspired by the use of LSTMs for single vector Scene Graph representation in (GUO *et al.*, 2019) and by the use of recurrent encoders in general to aggregate sequences. Since Transformer's are known to scale with sequence size, our version adds little computational cost: the maximum number of objects in a scene is kept at 8. Aside from aggregating, the Transformer module naturally provides further refinement of the Scene-Aware Object-level Representation (SA-OLR).

Furthermore, Transformers employ positional encoding, providing a good information entry-point for the objects' positional information. We adapt traditional sequence-based positional encoding to scenes, following Ashual and Wolf (2019), the scene is broken into a 5x5 grid and a position $p \in 0, 1, ..., 24$ is attributed to each object based on where its centre falls on the grid. This p is used to compute its positional encoding (see Fig. 22). We show in Section 4.4 that including this positional information is essential to the method's success; so much so that in the designs without a Transformer (Section 4.5) we concatenate the position p as a one-hot encoded vector to the Scene Graph node representation.

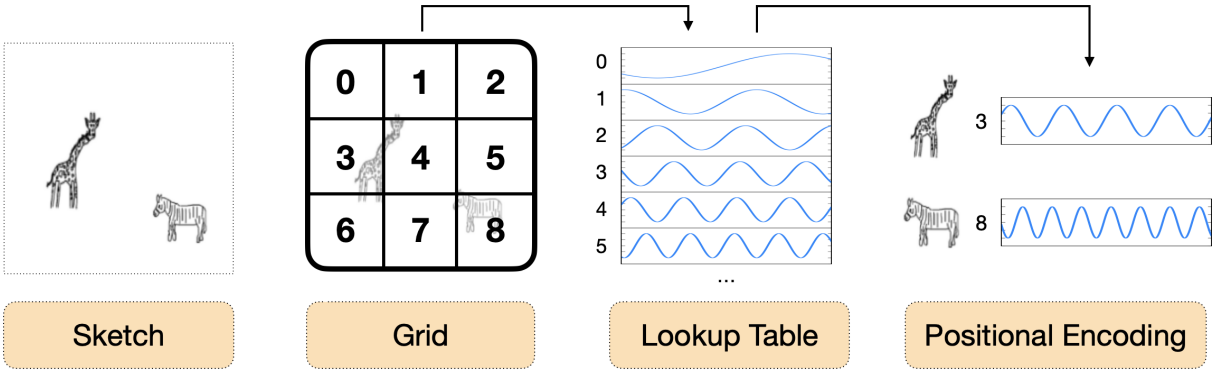


Figure 22 – Grid-based Positional Encoding. We adapted the positional encoder from traditional Transformers by attributing one encoding to each block of the grid-ed scene.

We specify our Transformer with an encoder-only design $t^e(.)$ with 3 layers and 16 attention heads on each. The input is the SA-OLR $g_r(f(o_i))$ for each object in the image and one empty vector $\vec{0}$ for aggregation, used in the same manner as a "[CLS]" token would in a BERT transformer (DEVLIN *et al.*, 2019). In each layer the model computes the attention weights that relate each vector to all the vectors in the sequence, following the *scaled-dot-product attention* mechanism specified in Equation 2.13. In the output, the position that contained $\vec{0}$ has the Scene Representation $r(\mathscr{G}_{sg})$ (SR) while other positions contain further processed SA-OLR for each object. When using the Transformer for aggregation, we use those new SA-OLR vectors as the object representations for the steps that follow.

4.3.4 Compositional Sketch-based Image Retrieval

The single *Scene Representation* (SR) is used for contrastive learning, designed as a search embedding in which the similarity of images and the sketched input may be measured.

During contrastive training, we sample half of our negatives randomly from other scene photos in the training set. The other half of the negatives are synthesised by swapping objects of different class into the positions of the objects in the positive image (Fig. 23). This is possible because our model does not take the entire image as input but instead takes object crops that can be manipulated. By including such a class-swapped version of the image as a negative, the model is encouraged to discriminate change in object class to a greater degree than changes in object positions and relationships. Our contrastive loss, adapted to incorporate these synthetic negatives, is:

$$\begin{aligned}\mathscr{L}_{cont}(x^{skt},x^{img},x^{img}_{sn}) =& \frac{1}{2}(Y)((x^{skt}-x^{img})^2) \\ &+\frac{1}{2}(1-Y)(1-(x^{skt}-x^{img})^2) \\ &+\frac{1}{2}(1-(x^{skt}-x^{img}_{sn})^2),\end{aligned} \tag{4.10}$$

where x^{skt} and x^{img} are respectively the sketch-based and image-based SR, Y is a label indicating

if a pair of vectors are from the same scene. Finally, x_{sn} is the SR of the synthesized scenes with swapped objects.

4.3.5 Semantic Scene Layout Generation

To further improve the scene representation learning (see subsec. 4.4.4) and enable novel use cases we incorporate Semantic Scene Layout generation as an auxiliary task. Given the SA-OLR as input we train generators for bounding boxes and masks, the composite parts of a Semantic Layout.

The box generator is trained using Generalised Intersection-over-Union (GIOU) (REZATOFIGHI *et al.*, 2019) and mean-squared error (MSE). The first was presented by Rezatofighi *et al.* (2019) for object detection models, they showed that changing to a loss that better reflected the intent (low Intersection-over-Union) led to improved bounding box computation. The generator itself is a 2-layer MLP. Note that the because the model takes coarse object positions as input, the box generator works as an autoencoder, ensuring the positional information is retained in the final object representations.

Taking b as ground truth bounding box, $\hat{x}$ as the SA-OLR of each object, our generation

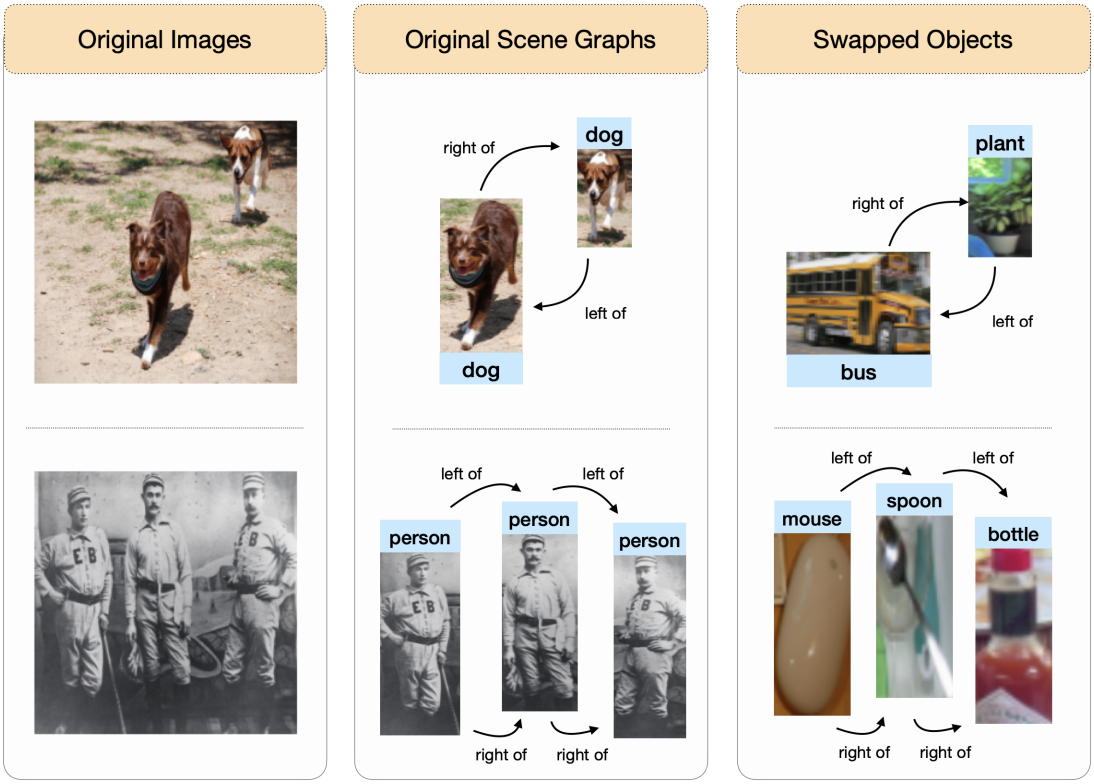


Figure 23 – During contrastive training of the SR, negatives are synthesised by swapping objects in the scene for others of differing class. This helps the model prioritise class over spatial relationships. First column shows the original image, the second the individual cropped objects, and the third the swapped objects.

losses are, for bounding boxes:

$$\mathscr{L}_{gen_b}(\hat{x}) = \lambda_1 \mathscr{L}_{GIOU}(b, f_{gen_b}(\hat{x})) + \lambda_2 (b, f_{gen_b}(\hat{x}))^2 \tag{4.11}$$

for each object on each scene, with $\lambda_1 = \lambda_2 = 10$.

The mask generator is trained within a GAN framework (GOODFELLOW *et al.*, 2014). We use the LSGAN formulation from Mao *et al.* (2016) for the losses, plus mean squared error (MSE, to regularise the adversarial training). Additionally, another cross-entropy classification loss $\mathscr{L}_{CCE_f}$ is applied to the SA-OLR to encourage it to remain semantic.

Training under the GAN framework means we need both a generator f_{gen_m} – the network that synthesises masks – and a discriminator f_{disc_m} – the network trained to classify masks as real or synthesised. Both are trained together under a min-max setup, with f_{gen_m} learning to "fool" f_{disc_m} and f_{disc_m} learning the tells of synthesised samples. We implemented f_{gen_m} as a simple block network, with 5 blocks composed of nearest-neighbours up-sampling, 2D convolution, batch normalisation and ReLU activation. The f_{disc_m} network is a 3-layered PatchGAN discriminator, designed by Isola *et al.* (2017) to classify individual image patches as real or synthesised and aggregate the scores to build the output; this is also the same discriminator used by Park *et al.* (2019b).

Taking our ground truth masks as m_g, and the object labels as $y \in C$, where C is the set of object classes. The loss for the mask generator f_{gen_m} is:

$$\mathscr{L}_{gen_m}(\hat{x}, m_g, y) = \lambda_3 (m_g - f_{gen_m}(\hat{x}))^2 + \lambda_4 (f_{disc_m}(f_{gen_m}(\hat{x}), y) - 1)^2 \tag{4.12}$$

with f_{disc_m} as the conditional discriminator network. Those losses back-propagate through all representation levels, and are applied to both sketched and photographic scenes. The weights are $\lambda_3 = 10$, $\lambda_4 = 0.5$. f_{disc_m} itself is trained with the complementary adversarial loss in classic GAN fashion:

$$\mathscr{L}_{disc_m}(\hat{x}, m_g, y) = \frac{1}{2}(f_{disc_m}(m_g, y) - 1)^2 + \frac{1}{2} f_{disc_m}(f_{gen_m}(\hat{x}) - y)^2 \tag{4.13}$$

where we highlight the simplicity of the LSGAN loss design, different from the original GAN formulation (Equation 2.6) as it was designed to simply approximate outputs to either 0 or 1. This $\mathscr{L}_{disc_m}$ loss back-propagates only through the discriminator network and relies on a separate optimiser from the whole Scene Designer.

Having a trained model, at inference time we turn the masks and boxes into a semantic layout by "pasting" masks within the confines of the object's predicted bounding box. In this object-only layout, objects are placed ordered by their size, so that bigger objects are behind smaller ones (after Ashual and Wolf (2019)). The generated layout is combined with one for the background, selected from the top search results or coarsely drawn by the user, as our model is focused on objects only. The synthesised semantic layout is not used for training the network. It is instead combined with an off-the-shelf Layout-to-Image model to generate photographic images. We specifically used SPADE from Park *et al.* (2019b) for this purpose.

4.3.6 Training Stages

Training is performed in three stages. In the **First Stage** only the networks that yield the Object-level Representation (OLR) are trained ,independently of the rest of the model. Training is done through the combination of cross-entropy and triplet losses:

$$\mathscr{L}_{OLR}(x_a^{skt}, x_p^{img}, x_n^{img}) = \mathscr{L}_{tri}(x_a^{skt}, x_p^{img}, x_n^{img}) + \mathscr{L}_{cce}(x_a^{skt}, x_p^{img}, x_n^{img}) \tag{4.14}$$

with $x_a^{skt}, x_p^{img}, x_n^{img}$ being the triplets of single-object sketches (a) and images (p, n), as explained in Section 4.3.1. The negative image objects in this stage are taken randomly from other scenes. The OLR is trained for 100 thousand iterations with a batch size of 128 samples.

In the **Second Stage** we use our novel soft-paired sketch and image scenes dataset (see Sec. 4.4.1) to train entire model end-to-end for a further 120K iterations, combining all of the model's losses:

$$\begin{aligned} \mathscr{L}_{OLR}(x_a^{skt}, x_p^{img}, x_n^{img}) &+ \frac{1}{2}\mathscr{L}_{gen_b}(\hat{x}^{skt}) + \frac{1}{2}\mathscr{L}_{gen_b}(\hat{x}^{img}) \\ &+ \frac{1}{2}\mathscr{L}_{gen_m}(\hat{x}^{skt}) + \frac{1}{2}\mathscr{L}_{gen_m}(\hat{x}^{img}) \\ &+ \mathscr{L}_{cont}(x^{skt}, x^{img}, x_{sn}^{img}) \end{aligned} \tag{4.15}$$

note how we apply synthesis-related functions to both sketch ($\hat{x}_s$) and image-based ($\hat{x}_i$) generation.

Finally, for the **Third Stage** we finetune the model with a hard-paired dataset (See Sec. 4.4.1), training for a further 10K iterations. Training for all stages is done via the Adam (KINGMA; BA, 2015) optimiser, with $\beta_1 = 0.5$ and $\varepsilon = 10^{-9}$; learning rate is $\eta = 10^{-4}$ for the first two stages and $\eta = 10^{-5}$ for the finetuning stage. The mask discriminator follows the *Two Time-Scale Update Rule* (TTUR) from Heusel *et al.* (2017) with $\eta_{disc_m} = 4\eta$. Batch size is kept at 16 samples at all stages.

4.3.7 Implementation Details

The object crops are scaled to 96x96 in pre-processing, as well as their values normalised to the $[-1, 1]$ floating point range. The QuickDraw sketches are from the set used by Ha and Eck (2018) and are rasterised to 96x96 images from the stroke3 format.

To create our **Object-Level Representation (OLR)**, the sketches and images crops go through two separate MobileNet's, the specific model used was trained on ImageNet with 96x96 image size and is available through Tensorflow/Keras[1]. The output of each network is average pooled to decrease the dimensionality and then the model splits into two steps: one fc layer with softmax activation for each domain classifies the inputs while a set of 2 shared fc layers computes the OLR used in the triplet loss. The specifics of those layers are in Table 6.

[1] <https://www.tensorflow.org/api_docs/python/tf/keras/applications/MobileNet>

Layer	In	Out	Act.
MobileNet	(96, 96, 3)	(7, 7, 1024)	-
AveragePool	(7, 7, 1024)	(1024,)	-
fc^1	(1024,)	(256,)	-
fc_{class}	(256,)	(172,)	Softmax
fc^1_{olr}	(256,)	(128,)	ReLU
fc^2_{olr}	(128,)	(128,)	ReLU

Table 6 – Detailed layers of the Multi-stream module that computes the Object-level Representation (OLR).

Our Graph Neural Network (GNN) encodes the Scene Graph (SG) into a **Scene Aware Object-level Representation (SA-OLR)**. Our GNN has 6 layers, each built with hidden dimensionality of 512 (the output of the first internal fc layer), output of 256 and all activations done via LeakyRelu with $\alpha = 0.3$. Our code was based on Johnson, Gupta and Fei-Fei (2018)'s implementation[2] of a GNN for Scene Graphs.

Finally, our Transformer architecture takes the SA-OLR and computes the single **Scene Representation (SR)**. Our grid-based positional encoding follows traditional encoders and is defined as:

$$p_{i,j} = \begin{cases} sin(\frac{i}{10000^{j/256}}) & \text{if } j \text{ is even} \\ cos(\frac{i}{10000^{(j-1)/256}}) & \text{if } j \text{ is odd} \end{cases} \tag{4.16}$$

where $i \in 0, 1, ..., 24$ is the position in the grid while $j \in 0, 1, ..., 255$ is the position in the vector (which is 256D long). Our Transformer has 3 attention layers that use multi-head attention (16 heads), a hidden dimension of 512D, dropout with rate 0.1 and Layer Normalisation with $\varepsilon = 10^{-6}$. Our implementation was based on the official TensorFlow code[3].

4.4 Experiments and Discussion

We evaluate the performance of Scene Designer for both compositional sketch search (CSBIR) and synthesis, contrasting performance against contemporary baselines for both tasks. We've chosen to evaluate the auxiliary generative task as well as the main CSBIR one to demonstrate that retrieval is not detrimental to generation, with the latter remaining competitive even if it doesn't improve upon the SoTA in layout synthesis. This section will also show that the synthesis task greatly improves CSBIR results beyond SoTA.

For CSBIR we compare against four baselines: a scene-level technique (SceneSketcher, from Liu *et al.* (2020)) and three single-object techniques. Of these, two are fine-grained SBIR models (Sketchy, from Sangkloy *et al.* (2016a), and Sketch-me-that-shoe, from Yu *et al.* (2016)), and one is coarse-grained (Multi-stage Learning or MSL, from Bui *et al.* (2018a)). For a fairer

[2] <https://github.com/google/sg2im>
[3] <https://www.tensorflow.org/tutorials/text/transformer>

comparison, we set our embedding dimension to 256-D, same as MSL, which uses the lowest dimension representation vectors among these baselines.

For the task of scene synthesis we compare against the sketch driven method of Gao *et al.* (2020) (proposed alongside their SketchyCOCO dataset) and the method of Ashual and Wolf (2019), designed to take semantic scene graphs (spatial arrangements of keywords) as input. To fully show the versatility of our model's input, we will explore sketch, image and mixed domain compositions in the experiments.

4.4.1 Datasets

We require paired data as input: sketched compositions and corresponding images of the scenes depicted by those sketches.

SketchyCOCO is a recent dataset of 14K such pairs, proposed by Gao *et al.* (2020) for sketch based scene synthesis. SketchyCOCO was created by sampling scenes and associated annotations (bounding boxes, masks) from the COCO-stuff (CAESAR; UIJLINGS; FERRARI, 2018) dataset – a photo-only collection – and augmenting these with sketched depictions of the same scenes. The sketched scenes were built by extracting features of samples from the Sketchy (SANGKLOY *et al.*, 2016a) dataset with a pre-trained retrieval network and matching those with the features from COCO-stuff's object crops.

Approximately 4 thousand of SketchyCOCO's images contain object instances (the remainder are solely background material; 'stuff' in the COCO-stuff definition). Only 14 classes of objects are annotated. To mitigate overfitting, we use the SketchyCOCO training partition (4,060 samples) only for fine-tuning our model. We include the test partition in our evaluation and filter it per experiment requirements: retrieval evaluation selects 233 samples as the work of Liu *et al.* (2020) needs scenes with 2 or more objects and most of the test set only has one object depicted; generation tests needs to use the overlap between test sets used in each of the compared models, a selection that yields only 137 samples. We also emulate Liu *et al.* (2020) in constructing an 'Extended SketchyCOCO' search corpus by holding out 5K random images from the COCO-stuff training set and adding these to the SketchyCOCO test set as distractors.

QuickDrawCOCO-92c is a novel dataset we propose, composing scenes with sketches from QuickDraw, the largest public sketch dataset (50M) (THE. . . , 2018). Inspired by SketchyCOCO, we similarly leverage scenes from COCO-stuff, taking advantage of the class overlap (92 classes) with QuickDraw. We synthesise sketch scenes by compositing sketches from QuickDraw onto a canvas, selecting a random sketch within the class corresponding to each object in a COCO-stuff image (Fig. 24). Our dataset is soft-paired, in that the sketches for each object do not necessarily match pose and appearance with the image, only the class label. We use QuickDrawCOCO-92c for all training stages except for the final finetuning stage where we combine samples from both SketchyCOCO and QuickDrawCOCO-92c. We separately

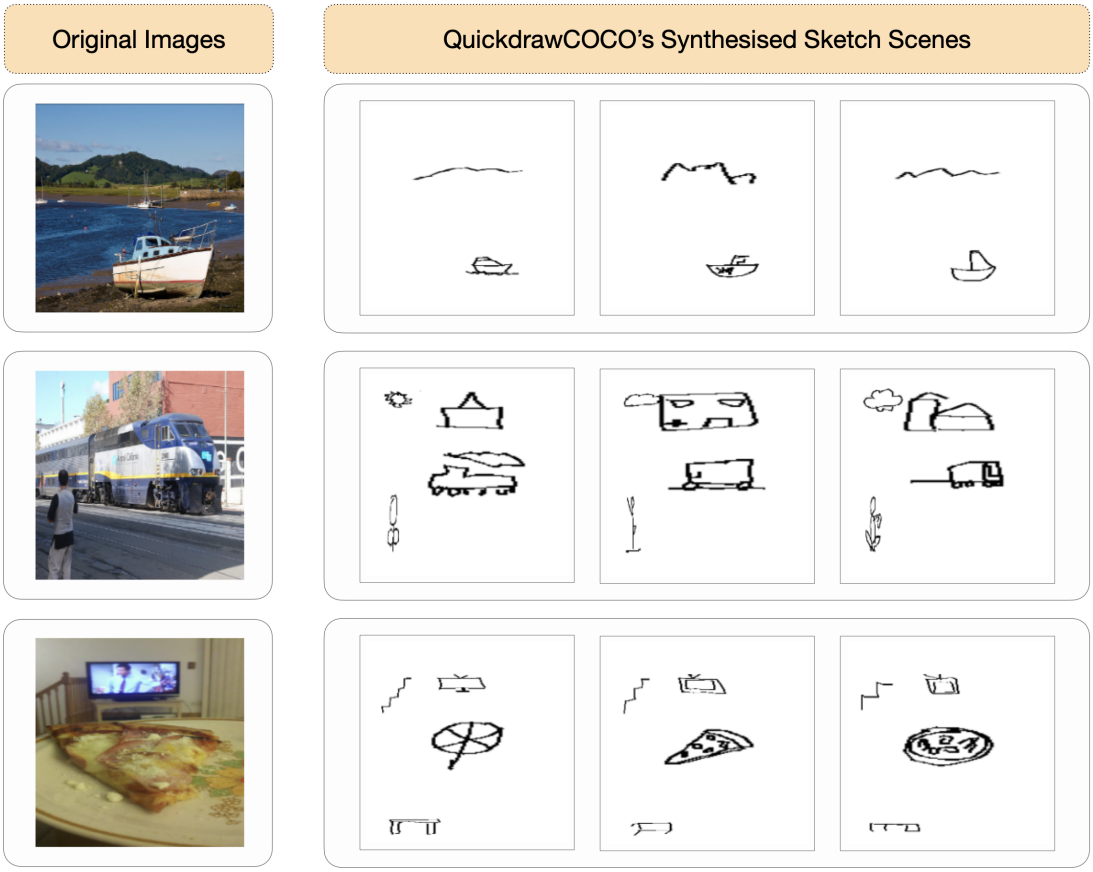


Figure 24 – Sample paired data from the proposed QuickDrawCOCO-92c dataset. Each row shows one COCO-stuff (CAESAR; UIJLINGS; FERRARI, 2018) image and three examples of synthetic sketched compositions matching that image derived from QuickDraw (THE. . . , 2018).

fuse the training and test partitions of the QuickDraw and COCO-stuff datasets to partition QuickDrawCOCO-92c into 111,112 training, 2,788 test and 1,907 validation scenes.

This dataset is 6x more category-diverse and has 8x more image samples than Sketchy-COCO; furthermore, since scene sketches are synthesised, each image scene has thousands of possible sketch compositions, making this the most diverse sketch dataset to date. To make QuickDrawCOCO possible, a map from QuickDraw classes to COCO classes was made; Out of 345 sketch classes in QuickDraw, 119 have correspondents in the COCO set, while from the 172 classes in COCO-stuff, 92 have sketch correspondents. The mapping is not 1:1 because QuickDraw is "more specific" than COCO-stuff, e.g. 'school bus' and 'bus' are both classes in QuickDraw, but both are mapped to 'bus' under COCO-stuff's set.

See Table 7 for comparisons between existing scene datasets, where we also compare against SKY-Scene and TUB-Scene, two recently introduced datasets (GE *et al.*, 2022). Unfortunately these datasets have not been made public by the time of writing and we are therefore not able to test Scene Designer with them.

Table 7 – Comparing existing datasets against our QuickdrawCOCO. We specify SketchyCOCO[*1] to refer to the subset of SketchyCOCO scenes that have any object in them, the remainder of SketchyCOCO contains only background instances. [*2]Since the sketch scenes are synthesised using COCO-stuff specs and Quick Draw! sketches, the number of distinct sketch scenes is nearly unlimited, but restricted to the scene layouts in COCO-stuff.

Dataset	#Imgs	#Skts	Paired?	Annot.	#Classes	Multi-obj.
COCO-stuff	164K	-	No	Segm.	171	Yes
Quick Draw!	-	25.8Mi	No	Labels	345	No
SketchyCOCO	14K	14K	Yes	Segm.	14	Yes
SketchyCOCO[*1]	4K	4K	Yes	Segm.	14	Yes
SKY-Scene	7K	7K	Yes	Segm.	30	Yes
TUB-Scene	7K	7K	Yes	Segm.	35	Yes
QuickDrawCOCO	116k	116k[*2]	Yes	Segmentation	92	Yes

4.4.2 Evaluating Retrieval

We adopt the evaluation proposed in SceneSketcher (LIU *et al.*, 2020), the only prior technique exploring SBIR for scenes: retrieved images are relevant only if they match the sketched scene exactly. This setup mirrors that of classic Fine-grained SBIR. Models are evaluated on SketchyCOCO, QuickDrawCOCO-92c and 'Extended SketchyCOCO'.

Compositional search. Table 8 reports recall at rank $k = \{1, 5, 10\}$ (R@k) for our proposed search embedding for sketched scenes on the SketchyCOCO benchmark. We compare against SceneSketcher (LIU *et al.*, 2020), the work most similar to ours and also against three single-object SBIR baselines: Sketch-me-that-shoe (YU *et al.*, 2016); Sketchy (SANGKLOY *et al.*, 2016a); and MSL (BUI *et al.*, 2018a). These results show our model more than doubles previous SotA recall@1 results and greatly improves other metrics. We also report performance of our method using mixed-domain queries, where we substitute half of the sketched objects for their image cropped counterparts. This evaluation modality yields an even higher result, likely due to half of the objects sharing the same domain as the test corpus. The final row reports on our model trained on QuickDrawCOCO-92c only, prior to finetuning on SketchyCOCO training set (Second Stage of training). This demonstrates that our model generalises sufficiently to beat the SotA on SketchyCOCO without explicit training on the sketched scenes of that set.

In Table 9 we report the same metrics on out newly introduced QuickdrawCOCO benchmark dataset. We were unable to include metrics from Liu *et al.* (2020) and Yu *et al.* (2016)'s models due to the lack of public models or code for these methods. QuickDrawCOCO-92c is a more challenging query set due to its increased size and the more messy / abstract sketches within QuickDraw's original set. We present our results alongside public Sketchy (SANGKLOY *et al.*, 2016a) and MSL (BUI *et al.*, 2018a) models trained on QuickdrawCOCO. The comparison shows the necessity of speciality models for scenes, with the models focused on single-object retrieval failing on the new modality.

In Fig. 25 we show visual query results for both datasets and in Fig. 26 we have tested

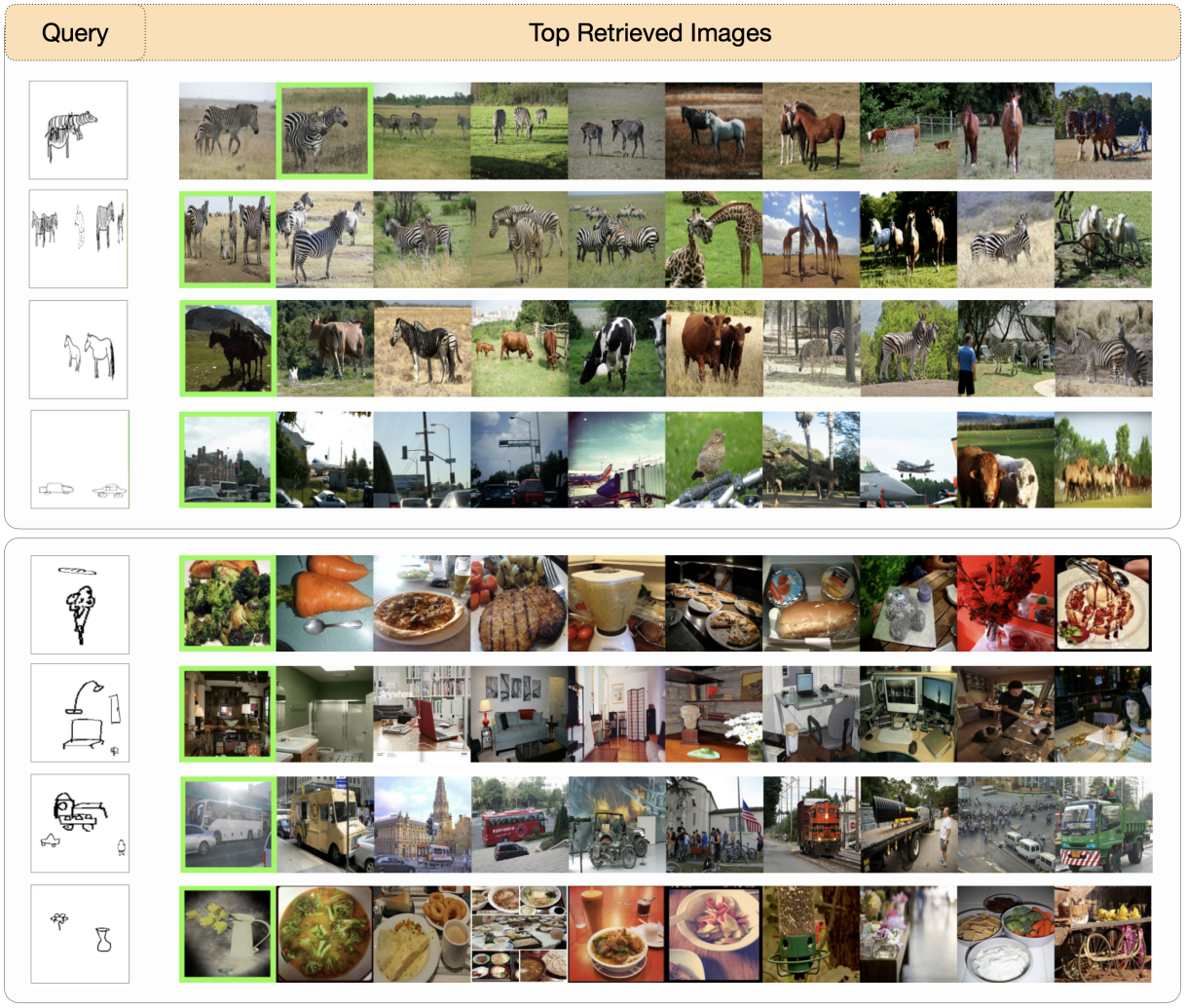


Figure 25 – Retrieval results using the SketchyCOCO (first block) and QuickDrawCOCO sets (second block). The results often contain the exact match and are contextually consistent, even when the model doesn't retrieve perfect class matches it does retrieve results that visually match the sketch (see last row on the second block, where the flower is confused with food and the vase with cups). To allow for qualitative comparison with SceneSketcher (LIU *et al.*, 2020), in (c) we've repeated the same queries of their qualitative evaluation on SketchyCOCO (Fig. 5 on their paper). Relevant results in green

the same queries presented in Fig. 5 of the SceneSketcher (LIU *et al.*, 2020) paper, allowing for a side-by-side comparison (their models and code are not public).

Single object search. We evaluate the performance of our model at single-object retrieval, baselining against both the Sketchy and MSL models on the SketchyCOCO dataset. We sample 120 queries randomly from SketchyCOCO containing only one object. In contrast to composition search, many relevant results exist in the SketchyCOCO dataset for a given sketched object. We consider a result relevant if both its category and pose match the sketched query. Since no ground-truth annotation with these criteria is available for SketchyCOCO, we crowd-source per-query annotation via Mechanical Turk (MTurk) for the top-k (k=15) results and compute both recall@k and precision@k in Table 10. For each ranked result we ask 5 MTurk participants to indicate relevance (or not), and filter on consensus level. Our approach significantly outperforms

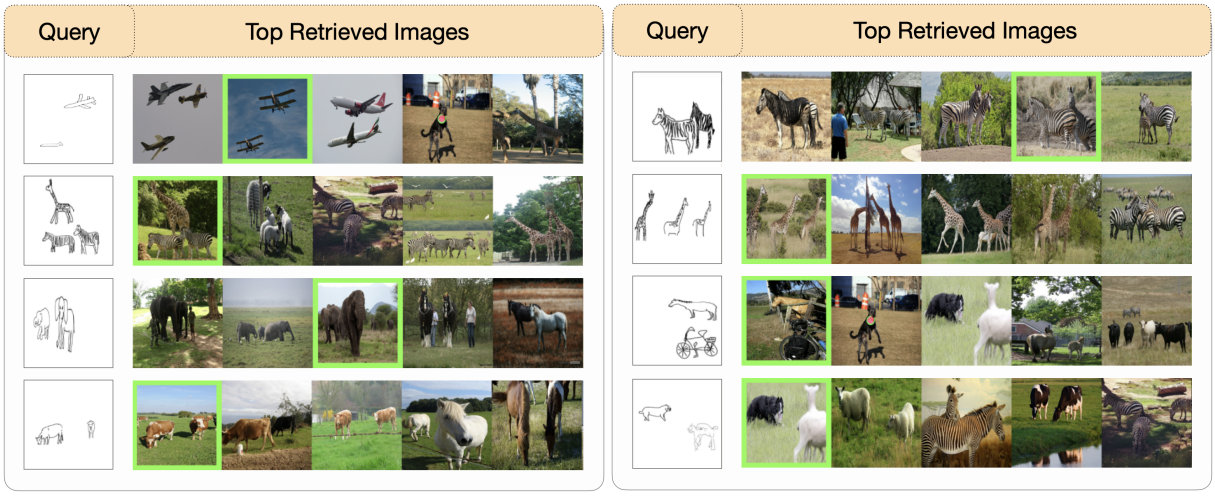


Figure 26 – Qualitative comparison with SceneSketcher (LIU *et al.*, 2020); to allow for comparison – and given there is no public model available – in this figure we've chosen the same queries of their qualitative evaluation on SketchyCOCO (Fig. 5 on their paper). Comparisons can be drawn by looking at their paper as well as this figure. Relevant results in green.

Table 8 – Recall@K metrics on the SketchyCOCO dataset and its extended version from Liu *et al.* (2020). Our model more than doubles the previous state-of-the-art recall@1 metric. Additionally, we show our performance when using mixed-domain compositions and also with the model trained only with QuickDrawCOCO-92c, before finetuning on SketchyCOCO.

Method	SketchyCOCO			Ext. SCOCO		
	r@1	r@5	r@10	r@1	r@5	r@10
Ours	**79.39**	**96.99**	**99.57**	**64.37**	**92.27**	**94.42**
Sketch-me-that-shoe (YU *et al.*, 2016)	06.19	17.15	32.86	<0.01	<0.01	01.90
Sketchy (SANGKLOY *et al.*, 2016a)	03.43	09.87	15.87	<0.01	00.85	00.85
MSL (BUI *et al.*, 2018a)	09.44	28.75	43.34	05.15	15.87	17.59
SceneSketcher (LIU *et al.*, 2020)	31.91	66.67	86.19	12.38	26.67	38.10
Ours (Mixed-domain)	82.40	98.28	99.57	59.65	79.39	85.83
Ours (Before finetuning)	36.90	73.81	89.69	29.18	60.08	76.82

both baselines at all consensus levels. The sensitivity to object orientation and pose underlines the utility of sketch over labelled boxes (per (ASHUAL; WOLF, 2019)) to specify the appearance of the desired object.

4.4.3 Evaluating Layout Synthesis

While our model generates semantic layouts instead of images, we study the later as it has quantitative metrics available, it allows for comparison with the model closest in task to us (SketchyCOCO (GAO *et al.*, 2020)) and is a good representation of applying our proposed creative workflow in a real scenario. Our goal is to show that more than improving retrieval results (see Sec. 4.4.2), our synthesised layouts lead to generated images comparable to SoTA; we also highlight that as Layout to Image Synthesis improves, the of-the-shelf model can be hot swapped in an application scenario.

Table 9 – Recall@K metrics on the QuickDrawCOCO-92c set. As we present the first model designed with this dataset in mind, our performance is unparalleled. We do not compare against Sketch-me-that-shoe (YU *et al.*, 2016) and SceneSketcher (LIU *et al.*, 2020) for lack of public code. Additionally, we show our performance when using mixed-domain compositions and also with the model trained only with QuickDrawCOCO-92c, before finetuning on SketchyCOCO.

Method	QuickDrawCOCO		
	r@1	r@5	r@10
Ours	**65.98**	**83.38**	**88.26**
Sketchy (SANGKLOY *et al.*, 2016a)	00.07	00.35	00.60
MSL (BUI *et al.*, 2018a)	00.10	00.53	01.07
Ours (Mixed-domain)	75.22	89.45	94.43
Ours (Before finetuning)	66.12	83.60	88.33

Table 10 – MTurk sketch search results for the proposed and baseline methods on the SketchyCOCO dataset. Precision@k and Recall@k are presented at different (C)oncensus thresholds.

Method	Precision@1			Precision@15		
	C2	C3	C4	C2	C3	C4
Ours	**66.12**	**52.07**	**37.19**	**22.98**	**16.14**	**10.30**
MSL (BUI *et al.*, 2018a)	16.53	14.05	09.09	10.41	07.00	04.41
Sketchy (SANGKLOY *et al.*, 2016a)	35.54	23.97	18.18	17.36	11.79	07.77
	Recall@1			Recall@15		
	C2	C3	C4	C2	C3	C4
Ours	**66.12**	**52.07**	**37.19**	**96.69**	**90.91**	**71.90**
MSL (BUI *et al.*, 2018a)	16.53	14.05	09.09	84.30	72.73	56.20
Sketchy (SANGKLOY *et al.*, 2016a)	35.54	23.97	18.18	85.95	80.17	65.29

We compare against that SketchyCOCO's accompanying public model (GAO *et al.*, 2020), and against the public model of Ashual *et al.*, (ASHUAL; WOLF, 2019) which synthesises images from spatial word maps *i.e.*, implicit scene graphs. We synthesise from the sketch, and use the paired image as ground truth for evaluating the fidelity of the output. For Ashual *et al.*, , we create an input scene graph based on object classes and bounding boxes.

Objective Metrics. We compute Fréchet Inception Distance (FID) (HEUSEL *et al.*, 2017) and Object Classification Accuracy (ASHUAL; WOLF, 2019) to compare the synthesized and ground truth images. A lower FID value indicates that the tested image distribution is closer the original and is more realistic. The Accuracy values come from finetuning a ResNet-101 classifier on crops from COCO and applying it to foreground objects cropped from the generated images.

Table 11 shows our model to score higher on FID and lose to Ashual *et al.*, work in Accuracy; we hypothesise the loss in the second metric might be due to our model adhering more closely to the sketch pose and therefore representing more unorthodox positioning. We've tried training SketchyCOCO from scratch on QuickDrawCOCO but the results were not satisfactory due to their model's reliance on matching poses; we've opted then to omit their model from this set's evaluation.

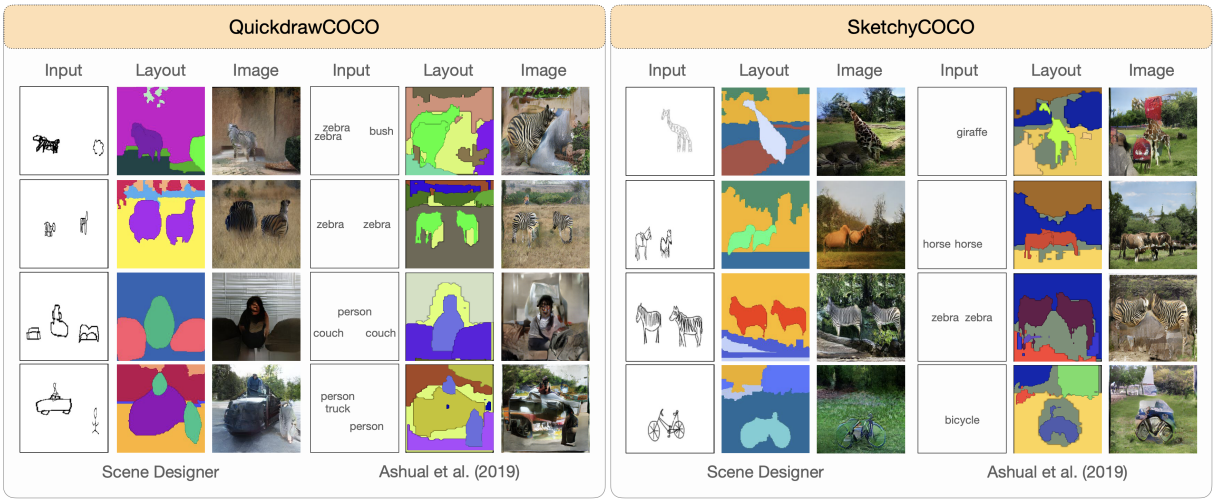


Figure 27 – Comparison between layouts generated by Scene Designer and model of Ashual and Wolf (2019). In our model, the sketched compositions drive the layout generation, which is then used with an off-the-shelf SPADE (PARK *et al.*, 2019b) model to synthesise images. In Ashual and Wolf (2019)'s model, the model takes in classes, positions and sizes and generates a layout; furthermore, they have their own layout-to-image generator. In the left block we compare ours and their layouts from QuickdrawCOCO, in the right one from SketchyCOCO. We made the representative word maps for Ashual and Wolf (2019) by hand, using original positions as guides .

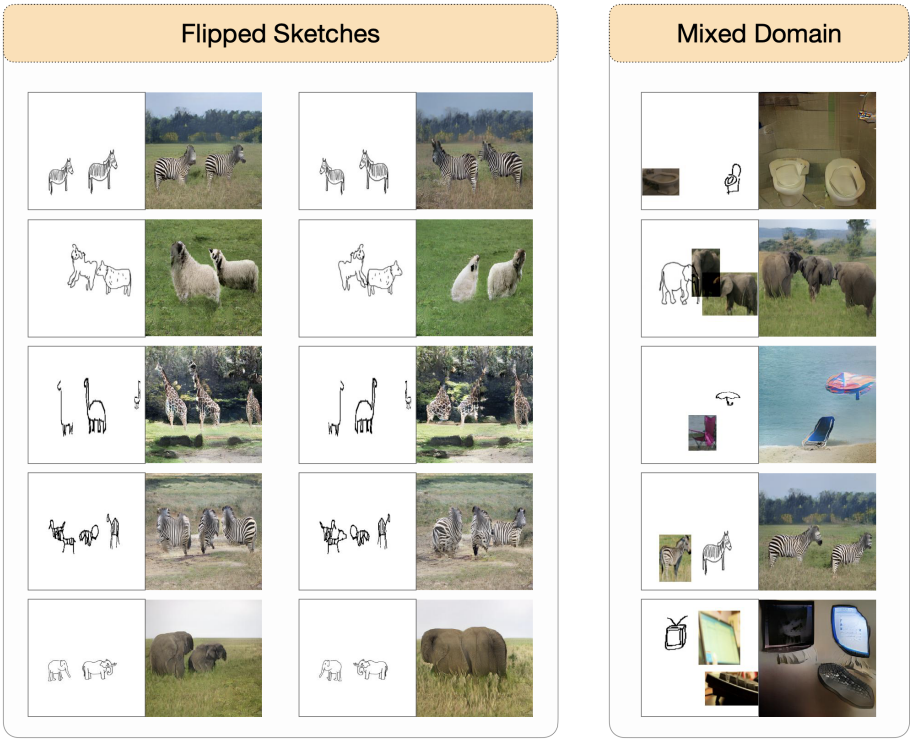


Figure 28 – We show Scene Designer is sensitive to individual object pose (left block) and that the model can seamlessly combine objects from both domains. These characteristics of the model allow for it to be used in a creative pipeline.

Table 11 – Generation metrics when using samples from SketchyCOCO and QuickDrawCOCO-92c. Both quantitative metrics were computed on the overlapping test set across the three/two models for each dataset (137 samples on the former and 1708 on the later). A lower FID and higher Object Classification Accuracy represent a better result. We've also included the metrics for the SPADE generator, representing an upper bound for our model.

Method	SketchyCOCO		QDCOCO-92c	
	FID↓	Acc.↑	FID↓	Acc.↑
Ours (sketch-based)	**145.81**	55.32	**77.77**	33.80
Ashual *et al.*, (ASHUAL; WOLF, 2019)	172.94	64.67	100.75	45.69
SketchyCOCO (GAO *et al.*, 2020)	198.17	29.14	-	-
Ours (image-based)	138.82	55.83	63.58	42.40
Ours (mixed-domain)	143.24	66.66	78.80	37.42
SPADE (PARK *et al.*, 2019b)	111.25	83.75	48.51	64.07

Table 12 – Ablation Studies, showing Recall@k and FID on QuickDrawCOCO and SketchyCOCO. With each set of models, we want to show that (A) each component is necessary, (B) the training procedure aids performance and (C) the single model can multi-task well.

Model Settings	QDCOCO			SketchyCOCO		
	r@1	r@10	FID↓	r@1	r@10	FID↓
(A1) W/o Transformer	47.59	77.21	59.78	47.63	91.84	132.12
(A2) W/o GNN	00.28	03.17	90.68	03.43	16.30	165.99
(A3) W/o positional encoding	43.87	76.67	57.69	43.34	90.12	134.01
(B1) W/o obj.-level pretraining	62.36	87.28	57.18	63.09	97.42	133.05
(B2) W/o synthesized negatives	57.67	85.22	54.31	69.52	98.28	126.00
(C1) No contrastive loss	40.17	72.09	53.05	50.64	89.69	129.46
(C2) No generation losses	57.20	84.29	121.17	57.08	96.56	249.88
Final model	65.98	88.26	55.46	79.39	99.57	133.22

We have included results applying SPADE to the ground-truth layout, which serves as an upper bound generation result for our method. Results from mixed-domain and images-only compositions, which are a unique capability of our model, are also included. These show the value in applying Scene Designer to incrementally build up a composition as part of an interactive creative process.

Qualitative Visual Results. We've also compiled visual results from ours and Ashual and Wolf (2019)'s model. In Fig. 27 we show the layouts from both models; the comparison highlights the advantages of using sketches instead of classes for the compositions as it gives the user control over the poses. Fig. 28 further highlights the flexibility of our design, showing mixed compositions and the effects of flipping individual objects in the compositions.

4.4.4 Ablation studies

We explore the significance of each stage of our proposed architecture and training methodology, comparing SBIR and generation results on QuickDrawCOCO-92c using several ablated variants as shown in Table 12, alongside the QuickDrawCOCO-92c result for the full method. We explore three categories of ablation (A-C). Category A ablates key features of the

proposed architecture. Category B ablates key training steps. Category C investigates if training a single model for both search and synthesis is beneficial to each task, as we have hypothesised in this thesis (**H4**). Each ablation experiment is detailed below:

(A1) Without Transformer. Removes the attention modules from the model, aggregating the representations from the GNN using simple addition.

(A2) Without GNN. Substitutes the GNN for an fc layer that bridges the OLR into the Transformer module.

(A3) Without Positional Encoding. Retains the proposed architecture but removes our grid-based positional encoding from the Transformer module.

(B1) Without object-level pretraining. Skips the First Stage of training, where the OLR is trained with object-level losses.

(B2) Without shuffled-objects negatives. Does not use our synthesised negative samples in the contrastive loss.

(C1) No Contrastive Loss. Does not use contrastive loss. The results on this model are driven by the masks and boxes' generation losses and object-level triplet loss only.

(C2) No Generation Losses. By removing the masks and boxes generation losses, this ablated model is only trained with search associated losses.

As expected, removing the GNN (A2) degrades both tasks' performance significantly, as the Transformer alone is not capable learning the scene representation without the relative information expressed by the graph's edges. The addition of the Transformer and its positional encoder are shown to improve the model significantly on both tasks.

Our synthesised negative examples (B2) improve search performance by 8% while pre-training the OLR encoders gives a smaller 3% boost.

Looking at the C group we can see that the combination of search and synthesis is the sensible choice for this and maybe future retrieval work. (C1) shows that even when driven only by generation learning, the scene representation is able to beat SotA in Compositional SBIR (compare against Tables 8 and 9). The experiment in (C2) then shows that combining the semantic knowledge of the synthesis task with retrieval brings an 8% boost to recall@1. Both results show that *generative models are also representation learners*, corroborating our Hypothesis **(H4)**.

4.5 Design Choices

We sought to consciously design each aspect of our model, and in this section we give support to some of those choices through more experiments.

The choice of Backbone Network. First we compare different choices of ImageNet-

Table 13 – Choosing the best backbone for the encoders. The first block of Scene Designer uses two networks (pretrained on ImageNet) to encode individual object photos and sketches. We investigate which would be the best backbone for this encoder. Results are reported for Search and Generation on QuickDrawCOCO-92c and SketchyCOCO. Parameter count (Par.) is listed in Millions.

		QuickdrawCOCO			SketchyCOCO		
Backbone	Par.	r@1	r@10	FID↓	r@1	r@10	FID↓
MobileNetV2	17.62	**65.98**	**88.26**	**55.46**	**79.39**	99.57	133.22
InceptionV3	58.86	61.21	85.55	55.81	77.25	**100.0**	137.82
ResNet50	62.47	58.25	85.33	57.36	73.81	98.28	**132.49**

Table 14 – Investigating the use of GNN and Transformer architectures by themselves against our combined model and the use of different methods for sequence aggregation. Results are reported for Search and Generation on QuickDrawCOCO. On Transformers E and F include Relative Positional Encoding (rel. pos. enc.); GNN D-H include concatenated positional information and GNN F-H include Self Attention (SA) layers for aggregation. Parameter counts (Par.) are listed in millions.

						QuickdrawCOCO		
Arch.	Layers	Relations	Pos.	Dim	Par.	r@1	r@10	FID↓
Transf. A	3	-	pos.enc.	512	27.7	03.25	20.22	65.62
Transf. B	6	-	pos.enc.	256	13.5	03.21	17.65	65.73
Transf. C	8	-	pos.enc.	256	14.7	06.28	29.86	62.43
Transf. D	16	-	pos.enc.	256	18.9	06.31	33.94	-
Transf. E	3	rel. pos. enc.	pos.enc.	512	27.8	04.62	25.46	64.00
Transf. F	6	rel. pos. enc.	pos.enc.	256	13.6	02.88	19.28	62.15
GNN A	4	gnn emb.	-	512	43.7	43.73	71.54	57.53
GNN B	6	gnn emb.	-	256	17.9	50.81	77.57	58.17
GNN C	4	gnn emb.	-	256	16.1	49.90	76.85	57.85
GNN D	4	gnn emb.	concat.	512	43.8	64.93	86.09	55.39
GNN E	4	gnn emb.	concat.	256	16.1	66.19	86.74	55.78
GNN F	4(G)+1(SA)	gnn emb.	concat.	256	16.1	65.76	85.95	56.01
GNN G	4(G)+1(SA)	gnn emb.	concat.	512	44.1	67.71	87.32	54.47
GNN H	4(G)+2(SA)	gnn emb.	concat.	256	16.1	66.77	86.81	58.32
GNN+Transf.	4(G)+3(T)	gnn emb.	pos.enc.	256	17.6	65.98	88.26	55.46
GNN+Transf. B	4(G)+3(T)	gnn emb.	pos.enc.	512	50.1	64.17	87.90	52.96
GNN+Transf. C	4(G)+3(T)	gnn emb.	pos.enc.	128	09.4	62.08	85.12	-

pretrained backbones for the OLR encoders, in Table 13 we compare MobileNetV2 (HOWARD *et al.*, 2017), InceptionV3 (SZEGEDY *et al.*, 2017) and ResNet50 (HE *et al.*, 2016). The results show that MobileNetV2 not only yields the best performance but also has a reduced computational cost profile than the alternatives. We believe the compactness of the model – which leads to increased bias in Statistical Learning Theory jargon – impacts model generalisation positively, both for quickly adapting to the retrieval task and to the new sketch domain.

Then we study our choice of GNN with a Transformer for aggregation against other options. Tables 14 and 15 compares alternative models that use either a standalone Transformer or a standalone GNN. It is easy to notice that the Transformer alone is not up for the task in

Table 15 – Investigating the use of GNN and Transformer architectures by themselves against our combined model and the use of different methods for sequence aggregation. Results are reported for Search and Generation on SketchyCOCO. On Transformers E and F include Relative Positional Encoding (rel. pos. enc.); GNN D-H include concatenated positional information and GNN F-H include Self Attention (SA) layers for aggregation. Parameter counts (Par.) are listed in millions.

						SketchyCOCO		
Arch.	Layers	Relations	Pos.	Dim	Par.	r@1	r@10	FID↓
Transf. A	3	-	pos.enc.	512	27.7	10.72	44.63	150.11
Transf. B	6	-	pos.enc.	256	13.5	11.58	49.35	142.92
Transf. C	8	-	pos.enc.	256	14.7	13.30	51.07	145.77
Transf. D	16	-	pos.enc.	256	18.9	09.44	53.21	-
Transf. E	3	rel. pos. enc.	pos.enc.	512	27.8	07.72	42.48	145.20
Transf. F	6	rel. pos. enc.	pos.enc.	256	13.6	08.15	46.32	148.37
GNN A	4	gnn emb.	-	512	43.7	47.21	88.84	129.32
GNN B	6	gnn emb.	-	256	17.9	44.63	92.70	138.33
GNN C	4	gnn emb.	-	256	16.1	47.63	91.41	130.22
GNN D	4	gnn emb.	concat.	512	43.8	78.54	98.28	130.33
GNN E	4	gnn emb.	concat.	256	16.1	77.25	99.14	132.58
GNN F	4(G)+1(SA)	gnn emb.	concat.	256	16.1	71.67	95.27	128.42
GNN G	4(G)+1(SA)	gnn emb.	concat.	512	44.1	77.25	98.71	123.65
GNN H	4(G)+2(SA)	gnn emb.	concat.	256	16.1	72.96	98.28	131.42
GNN+Transf.	4(G)+3(T)	gnn emb.	pos.enc.	256	17.6	79.39	99.57	133.22
GNN+Transf. B	4(G)+3(T)	gnn emb.	pos.enc.	512	50.1	76.82	99.57	129.70
GNN+Transf. C	4(G)+3(T)	gnn emb.	pos.enc.	128	09.4	69.52	97.85	-

any of its variations. Experiments were made with different number of layers (B, C and D) and dimensionality (A). We initially believed this loss in performance was due to the architecture not having access to the relationship information inherent to the graph design. To remediate this we have included (in E and F) the graph relationships through relative positional encoding (SHAW; USZKOREIT; VASWANI, 2018). The encoding introduced by Shaw, Uszkoreit and Vaswani (2018) is a learnable positional encoding that is added to each vector in the sequence prior to the attention computation, taking into account the relationship ("left of", "below", ...) of the vectors in the sequence. The experiments show however that this addition does not improve results, with the message-passing design of the GNN being much more efficient on its own.

The GNN presents a more interesting picture. We have similarly designed experiments with dimensionality (A) and depth (B, C) and found that the former is not worth the increase in model size and both under-perform without the Transformer. We found however that the Transformer's module task was not only to aggregate the vectors into a single representation but also to incorporate positional information through the positional encoder. To amend this information gap between models we follow Johnson, Gupta and Fei-Fei (2018) and concatenate the one-hot-encoded positional information to the graph's nodes and re-run the experiments with dimensionality (D, E).

The experiments with the concatenated positional information are much more aligned

with the ones with the Transformer module. The difference in dimensionality still yields little benefit given the increase in model size. Because in these models (D, E) the SA-OLR representations were aggregated with simple averaging, we also experiment with a more robust option, using Self Attention for aggregation in exeperiments (F, G, H). The presented results show that as long as the positional information is included in some fashion, all three choices for aggregation, be it simple addition, self attention or the Transformer, are viable to build a model withing our framework. We opt for the Transformer as it is the best performing on SketchyCOCO and it uses a form of positional encoding well known in the literature.

4.6 Conclusion

In this chapter and in the original publications (RIBEIRO *et al.*, 2021; RIBEIRO *et al.*, 2022) we introduced Scene Designer, a state-of-the-art CSBIR model that unites the tasks of retrieval and synthesis to improved performance and novel creative workflows. We developed a learning framework that learns to represent individual objects first and further into the pipeline combines them into scene-aware robust representations for each object and for the entire scene. Our combined multi-stream and sequence encoding architecture is innovative for SBIR and shows the strength of multi-stream designs in the new application of Scene Retrieval.

Scene Designer learns to embed sketched and photographic scenes into a common space, producing latent representations both for synthesising layouts from sketched scenes and for measuring compositional similarity between sketch and image scenes. The empirical evidence supports our claim that training for both tasks can yield more robust and semantic representations (Hypothesis **H4**).

The model is trained via an expanded dataset of sketch compositions and corresponding images (QuickDrawCOCO); a secondary contribution of our work. We show that the combination of feature learning and generation losses aided by our novel take on contrastive learning for scenes are responsible for obtaining SotA performance at scene search (Hypothesis **H2**). Further work could explore the use of this model in creative practice, and also study how the concept of composition retrieval (and this model) can be applied to other domains where multiple components are represented, such as 3D scenes (ABDUL-RASHID *et al.*, 2019) or vector graphics (ABE; MORITA; HAYASHI, 2018).

4.7 Data Availability Statement

Instructions and code for recreating the QuickDrawCOCO dataset described in this paper can be found at the code repository for the model: <https://github.com/leosampaio/scene-designer>.

Aknowledgements

This work was supported by Fundação de Amparo à Pesquisa do Estado de São Paulo (grants 2017/22366-8, 2019/02808-1, 2019/07316-0), Conselho Nacional de Desenvolvimento Científico e Tecnológico (304266/2020-5), and a charitable donation from Adobe Inc.

CHAPTER

5

SKETCH-AN-ANCHOR: ZERO-SHOT SKETCH-BASED IMAGE RETRIEVAL

In this Chapter we present Sketch-an-Anchor, a novel method to train Zero-Shot Sketch-based Image Retrieval (ZSSBIR) models in under an epoch. Most of the current studies in the literature break the ZSSBIR problem in two sub-problems: (a) domain alignment between both domains and (b) generalisation to unseen data. We show that a ZSSBIR solution can rely on existing pre-trained models for photographic images that excel at (b). By focusing on domain alignment and pre-trained models we are able to train Sketch-an-Anchor to state-of-the-art performance in under an epoch.

Sketch-an-Anchor was first described in a manuscript currently under review for presentation at an international computer vision conference (undisclosed to preserve double blind review integrity):

RIBEIRO, L. S. F.; PONTI, M. Sketch-an-anchor: Sub-epoch fast model adaptation for zero-shot sketch-based image retrieval. In: **Submission Under Review**. [S.l.: s.n.], 2022.

Motivation for this study came from observing that current models for photographic images already perform well even when trained without labels (under self-supervision). Our assumption then is that these representation spaces are already representative and semantically meaningful. With this in mind we introduce our Semantic Anchors, guiding embeddings learned from word-based semantic spaces (Word2Vec) and features extracted from these generaliseable off-the-shelf models. These anchors then drive our novel Anchored Contrastive Loss.

With this simple design and reliance on pre-trained models, Sketch-an-Anchor performs well while training with more than a 100 times less samples than other ZSSBIR studies. Our contributions to the thesis are then:

- Sketch-an-Anchor explores another more realistic setting for Sketch-based Image Retrieval, placed under the zero-shot paradigm. Under this new setting we achieve SoTA performance

with an efficient training regime.

- With our method's design we are able to train cross-domain models in under an epoch. We show that sketches can be directly mapped to generaliseable image spaces with little information loss. (Hypothesis **H3**)
- Under the zero-shot paradigm, Sketch-an-Anchor was designed to generalise to unseen classes and the empirical evidence shows it has succeeded. (Hypothesis **H3**)

This chapter presents an extended version of the submitted manuscript, revised for clarity and to highlight these contributions to the thesis.

5.1 Introduction

The task of Sketch-based Image Retrieval (SBIR) is a machine learning application that has been relevant for at least 20 years (ASHLEY *et al.*, 1995) and interest has only grown since.

While introducing Deep Learning methods to this task has led to great strides in performance, these systems also started depending on large datasets with both photos and sketches from the same category to perform well (DEY *et al.*, 2019a; SHEN *et al.*, 2018). That motivated Zero-shot SBIR task (ZSSBIR) (SHEN *et al.*, 2018); the protocol for ZSSBIR follows the common Zero-shot paradigm (XIAN *et al.*, 2018), the categories of images and sketches used for evaluation are not seen in training. This setup better mimics an open set scenario, which correspond to real applications for SBIR, where categories are vast and unpredictable: e.g. an e-commerce might add new products every day and an artist will search for different references for each creation.

Most studies break down the problem of ZSSBIR into two: domain alignment between images and sketches, inherited from SBIR, and generalisation to unseen data, inherent to the zero-shot protocol. We argue the latter problem can be considerably simplified. Our motivation comes from the Zero-shot retrieval performance of current state-of-the-art pre-trained models when tested on the image domain – the one they were trained for. In Figure 29, the graphs on the left show the zero-shot retrieval performance of a variety of these pre-trained models on single domain (either images or sketches) and cross-domain. While sketch and cross-domain performance are poor, image-to-image retrieval is excellent. We have designed Sketch-an-Anchor with this in mind; as the image embedding of these models is already semantically rich and generaliseable to unseen data, we focus on the domain alignment problem. We constrain the design with a caveat: the ideal learning protocol should not degrade existing image-to-image performance whilst mapping sketch features to the image embedding.

Because many pre-trained models are suitable for our Sketch-an-Anchor, we sought a model that would be more comparable against baselines yet generalised well by design. With

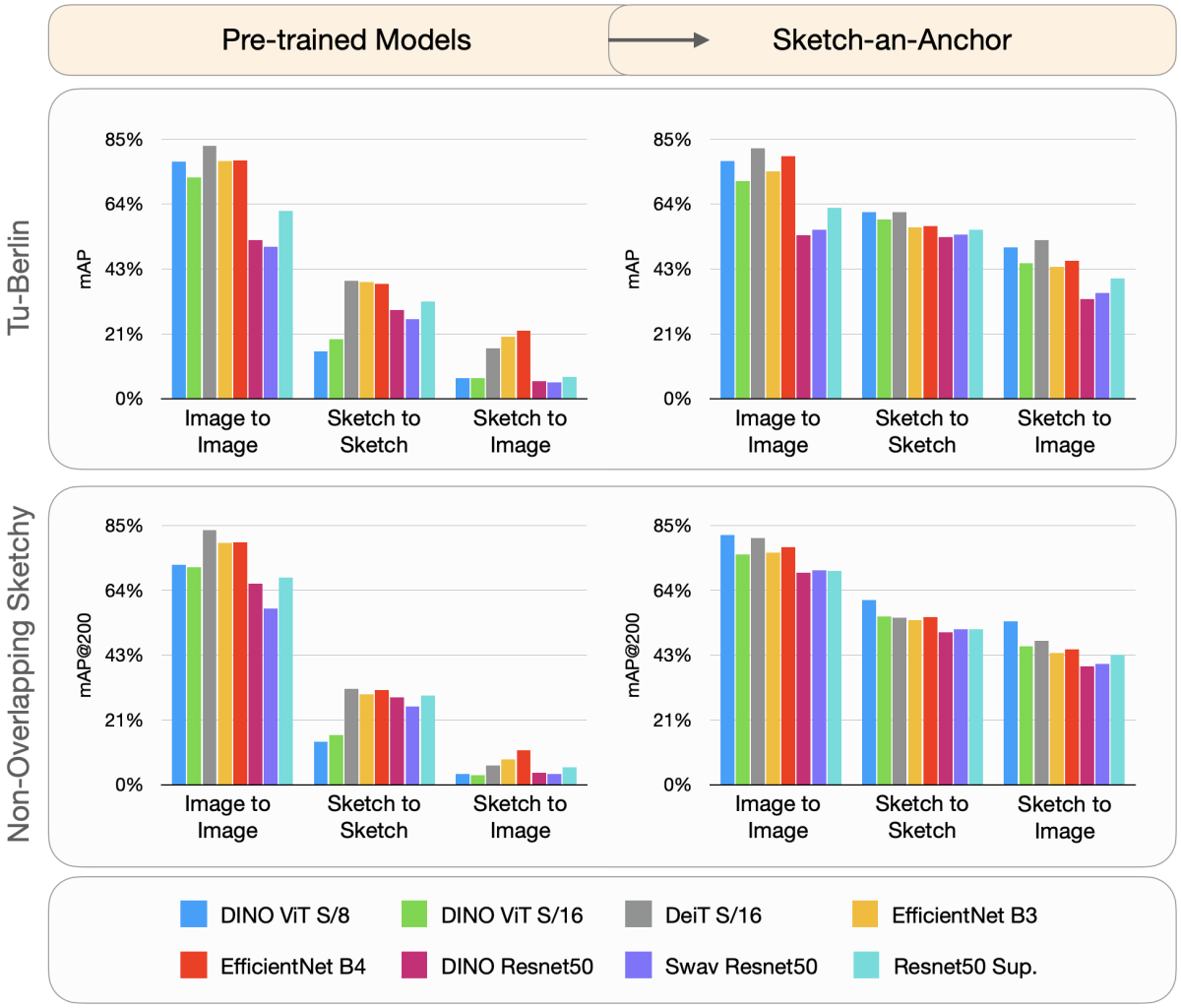


Figure 29 – Our Sketch-an-Anchor model re-frames the problem of Zero-shot Sketch-based Image Retrieval. Because pre-trained models already perform well on Zero-shot Image-based Retrieval, our efforts should go into keeping this performance while mapping sketches to the same space. In this Figure we compare several pre-trained models and their performance when trained under the Sketch-an-Anchor protocol. We show mAP (%) on the Tu-Berlin and Sketchy Zero-shot splits on three tasks, Image-based Image Retrieval, Sketch-based Sketch Retrieval and Sketch-based Image Retrieval.

these requirements in mind the self-supervision SoTA DINO S/8 was chosen as our starting model.

The field of Self-Supervised learning has seen many proposals in recent years. The task of Self-supervision is to learn – without labelled data – general image representations that can be used for diverse downstream tasks; the approach often taken (HE *et al.*, 2020; CARON *et al.*, 2021; CHEN *et al.*, 2020; GRILL *et al.*, 2020; CHEN *et al.*, 2020) is to train models effectively for image retrieval: semantically similar images should have similar representations.

We take DINO (CARON *et al.*, 2021), a recent self-supervised model and the best at k-NN classifier evaluation (where features are extracted from the frozen network to train a k-NN classifier for ImageNet). We can see in Figure 29 that both of DINO's ViT-based models

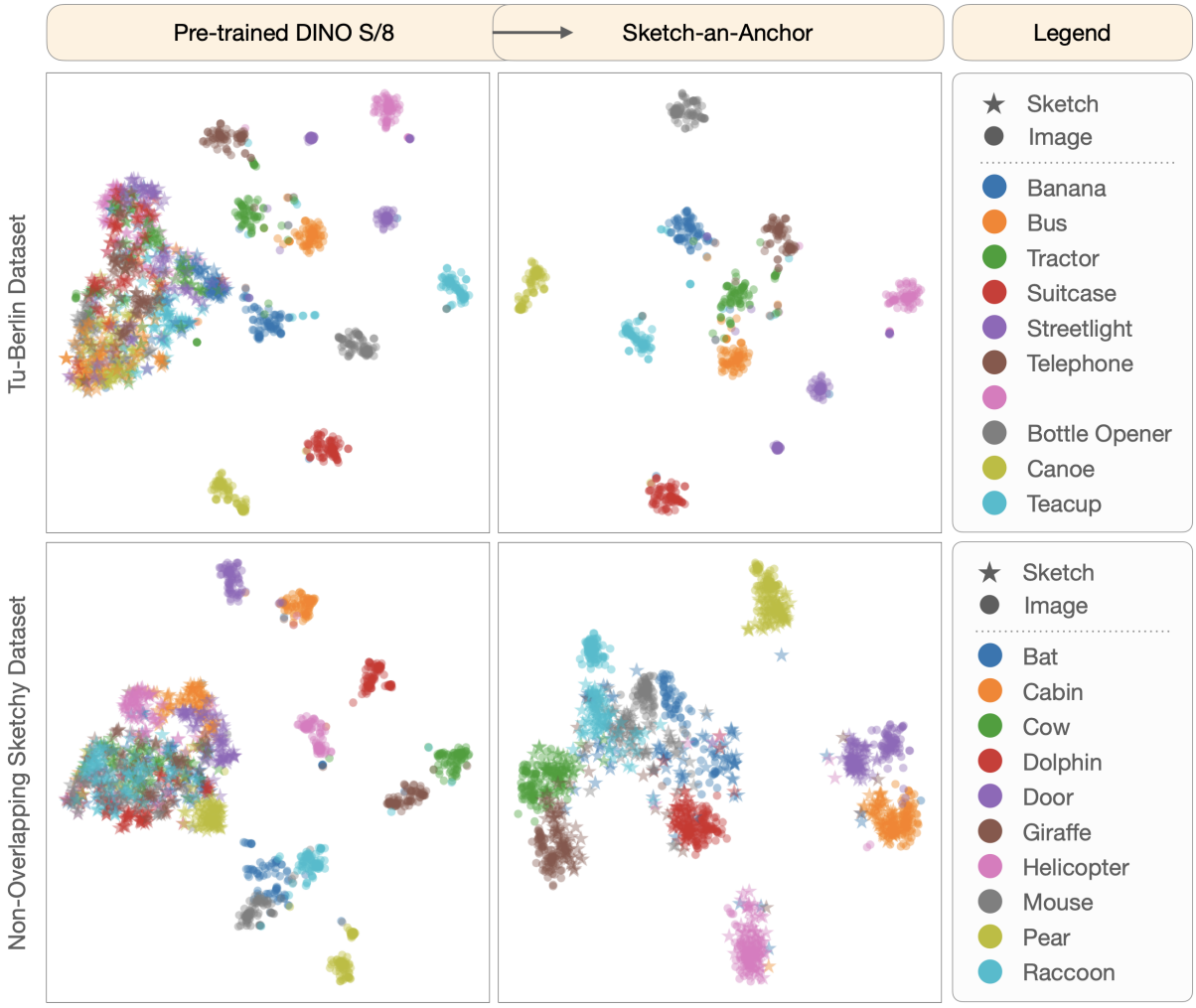


Figure 30 – Projections of embeddings using t-SNE. The figure compares the feature space of the pre-trained DINO S/8 model against the model trained under the Sketch-an-Anchor protocol. It is possible to see in the former how the space is already discriminative for unseen images but sketch representations are confused together. Sketch-an-Anchor keeps the discriminative image space while aligning sketches to the same space. Comparisons cover 10 random classes from the Tu-Berlin and Sketchy zero-shot sets.

(S/8 and S/16) perform well on image-to-image retrieval. In the same figure, on the right, we have also included a preliminary evaluation showing how each model performs when trained under the Sketch-an-Anchor protocol. The self-supervised models (DINO from Caron *et al.* (2021) and Swav from Caron *et al.* (2020)) have the best increase in performance for sketch and cross-domain retrieval. DINO S/8 is chosen for this adaptability and because it was previously applied to ZSSBIR (WANG *et al.*, 2022; TIAN *et al.*, 2022), making comparisons fairer.

To complement the motivations that drive Sketch-an-Anchor, we plot (using a t-SNE learnable projection from Maaten and Hinton (2008)) in Figure 30 the embedding of images and sketches, using the pre-trained model DINO S/8 and the model trained under Sketch-an-Anchor. The plot shows how the DINO space is discriminative for images but messy for sketches, and then how our model keeps the image relationships while learning better – and aligned – representations for the sketches.

Rethinking the sub-tasks has another remarkable benefit: since we do not need to learn the ZSIBIR sub-task, our model requires only a few iterations, 99.13% less iterations than the average ZSSBIR model to be exact, making it the fastest training ZSSBIR model in the literature by a very large margin.

To aid with our re-defined tasks we introduce the Semantic Anchors, a combination of the general Word2Vec (MIKOLOV *et al.*, 2013) semantic space and class centres based on the image representations output by the pre-trained model. We then use the Semantic Anchors and their pairwise similarities to design our novel Anchored Contrastive Loss, elevating a strong baseline model to SoTA performance.

To summarise, our contributions are the following:

- Re-think the ZSSBIR solution on matching sketch representations to an existing pre-trained space, since those are already good image retrieval models;
- The ability to train a model for ZSSBIR in under an epoch and achieve performance competitive with SoTA;
- Introduce Semantic Anchors, which learns from visual and word-based semantic spaces with a Graph Convolutional Net and guide the fast model adaptation;
- Our novel Anchored Contrastive Loss matches the sample similarity distribution to the anchors', further improving the task of matching sketch representations to the images'.

5.2 Related Work

The literature for classic Sketch-based Image Retrieval was already explored in Chapters 1 and 3 so we will therefore focus this section on Zero-shot Sketch-based Image Retrieval. The work of Shen *et al.* (2018) was the first to introduce the task whilst also proposing a new hashing method to accelerate retrieval. Including domain knowledge or side information is a common convention of solutions for Zero-shot tasks (XIAN *et al.*, 2018); therefore later works in ZSSBIR can be split by the side information they have used. The works of Dutta and Akata (2020), Zhang *et al.* (2020), Dey *et al.* (2019a), Chaudhuri *et al.* (2020), Deng *et al.* (2020) have used word-based semantics like we do to learn more general search spaces, while other models trust generative models to simulate unseen classes during training (DUTTA; SINGH; BISWAS, 2021; SHEN *et al.*, 2018; YELAMARTHI *et al.*, 2018a; PANDEY *et al.*, 2020) or combine both (DUTTA; AKATA, 2020). The study of Liu *et al.* (2019), SAKE, introduced the idea of using Knowledge Preservation to prevent pre-trained models from losing already acquired knowledge, and implemented their own custom distillation protocol to do so. Many models followed similar protocols (WANG *et al.*, 2022; TURSUN *et al.*, 2022; WANG *et al.*, 2021; TIAN *et al.*, 2022) or combinations of them with word-based semantic information. We share SAKE's authors' goal

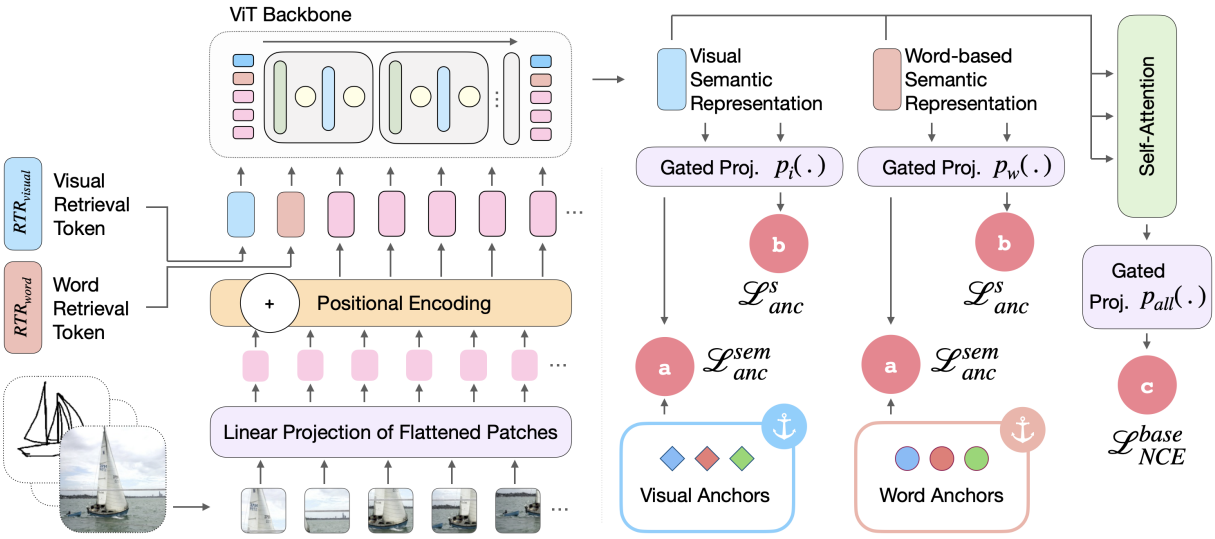


Figure 31 – Diagram for the final Sketch-an-Anchor architecture. We start with a ViT S/8 backbone pre-trained with the DINO (CARON *et al.*, 2021) protocol. To this backbone's input we include two learnable tokens, RTR_{visual} and RTR_{word}. The outputs related to these tokens will make up the Visual and Word-based Representations, which are both trained to match images and sketches with $\mathscr{L}^{s}_{anc}$ and to match the semantic anchors with $\mathscr{L}^{sem}_{anc}$. Finally, these representations are agregated through self-attention and the final representation is trained to match domains with classic InfoNCE, $\mathscr{L}^{base}_{NCE}$. The same network is used to extract features from sketches or photographic images.

of avoiding knowledge loss but designed our approach to not need to rely on a computationally expensive teacher signal. We highlight three studies from the Knowledge Preservation set that are most similar to ours. TVT (TIAN *et al.*, 2022) and PKSD (WANG *et al.*, 2022) share our DINO backbone and the latter also explores the correlation between class centres, but their training protocol uses multiple ViT instances and they take 100x longer to converge. SBTKNet (TURSUN *et al.*, 2022) is a recent study that makes use of class centres similarly to how we do our Semantic Anchors, which makes their model one of the fastest recent ones, training in under 25 epochs; however, 25 epochs is still more than 50 times longer than our Sketch-an-Anchor's training time. Our model is competitive against all of the previous studies while being simpler to implement, not requiring a teacher-student distillation setup or multiple network instances and, of course, being faster to train and to experiment with.

5.3 The Sketch-an-Anchor Method

In this section we describe the task and our solution. We start by introducing our baseline model, a strong and simple ZSSBIR solution, and then include the Semantic Anchors and Anchored Contrastive loss, additions that bring the baseline up to speed to compete with the best in the literature. A diagram for the complete architecture can be found in Fig. 31.

5.3.1 Problem Setting

Within Zero-shot Sketch-based Image Retrieval, a given model gets a sketched object as input and ranks images on a dataset based on how well they match the sketched object semantically. Under the zero-shot setting such model is evaluated with query sketches and retrieved images of objects that were not present in training.

Formally, an SBIR dataset is split into two subsets to create the zero-shot setting, $\mathscr{D}_{\text{train}} = \{S_{\text{train}}, I_{\text{train}}, C_{\text{train}}\}$ and $\mathscr{D}_{\text{test}} = \{S_{\text{test}}, I_{\text{test}}, C_{\text{test}}\}$, where S and I are the set of labelled sketches and images for each subset and C are the classes of each subset, and the zero-shot protocol is complete by ensuring that there is no intersection between classes in each subset: $C_{\text{train}} \bigcap C_{\text{test}} = \emptyset$.

5.3.2 A Strong Baseline

As presented in Sec. 5.1, we are motivated by the already great performance of self supervised models on Zero-shot Image-based Retrieval (ZSIBR) and believe this performance can be transferred to ZSSBIR with little finetuning. We chose to start with a Vision Transformer (ViT) (DOSOVITSKIY *et al.*, 2021) backbone pre-trained under the Self Supervised DINO protocol (CARON *et al.*, 2021); We explore other architectures/training protocols and the correlation of ZSIBIR performance and our ZSSBIR results in Sec. 5.4.6. A ViT encoder takes fixed-resolution ($R \times R$) patches of images as input, which are linearly projected to create a sequence of "image tokens"; a learnable token *CLS* is included in the sequence and all tokens are processed by N sequential layers composed each of a pipeline of Layer Normalisation, Multi-head Attention and feed-forward sub-layers. Each layer is connected by skip-connections as in ResNet (HE *et al.*, 2016).

These models are usually trained through supervised learning, but Caron *et al.* (2020) introduced DINO, a self-supervision protocol that makes use of multi-crop training (CARON *et al.*, 2020), a momentum encoder (HE *et al.*, 2020) and smaller ViT image patches to deliver representations that can make stellar k-NN classifiers. Because k-NN classifiers are essentially retrieval models (it is non-parametric and classification is done through search in the embedding space), we make a pre-trained DINO ViT our backbone model, specifically DINO S/8.

In addition to the pre-trained backbone $v(.)$ we include a new trainable token for retrieval, RTR, together with the patch-based image tokens. At the output, the processed RTR goes through a projection head $p(.)$, implemented as gated linear projection (MIECH; LAPTEV; SIVIC, 2017; CHEN *et al.*, 2021). The same network takes sketched and image inputs, and we take from self-supervision lingo (HE *et al.*, 2020) to call sketches queries $q = p(v(x_s))$ and the collection of images in a batch keys of a dictionary $k_i = p(v(x_i))$, where for each query there is a single key k_+ matching it. We then take the cosine similarity of each pair and use Information Noise-Contrastive Estimation (InfoNCE, introduced by (LAI, 2019) and previously presented in

equation 2.11) as our contrastive loss:

$$\mathscr{L}_{NCE}^{base} = -log\frac{exp(q.k_+/\tau)}{\sum_{i=0}^{K} exp(q.k_i/\tau)} \tag{5.1}$$

where τ is the temperature hyper-parameter and controls the balance between a uniform-distributed embedding and a discriminative one (WANG; ISOLA, 2020). Additionally, our task differs from Self-Supervised learning in that, because we are doing categorical retrieval, there is not a single key k_+ matching each query. Our solution is to enforce the single key scenario in training by making sure each batch only has one sketch and image from each class in that batch.

We show in Sec. 5.4 how this baseline is already a quickly adapting model for ZSSBIR which performs competitively.

5.3.3 Semantic Anchors

With our well-performing baseline at hand, we design additional components that lean into the strengths of the pre-trained model, starting with the *Semantic Anchors*.

Word-based Embedding. It is common practice in the ZSSBIR literature and other zero-shot tasks to include a generic semantic space as a guide to the model (DUTTA; AKATA, 2020; ZHANG *et al.*, 2020; DENG *et al.*, 2020) to prevent training from adhering too close to the training classes and make the model learn more general concepts. The most common semantic embedding used is Word2Vec, which can map words into a semantic embedding space, trained by looking at word co-occurrence on millions of texts (MIKOLOV *et al.*, 2013). We follow other methods and use it to map the words for each training class into this embedding space, creating what we call the *Word Embedding*, $E_{word} = \{c_i^w\}$, with each ith vector matching the ith seen/training class.

Visual Embedding. Because the chosen backbone has demonstrated retrieval performance for images, we argue that the space produced by its features is a good semantic embedding by itself, and it furthermore captures visual relationships that word embeddings cannot. We extract the representation for all images in the training sets and take the average vector of each class to make the *Visual Embedding* $E_{visual} = \{c_i^v\}$.

We collectively call these embeddings the *Semantic Anchors* because they are responsible for grounding the output in both general semantic information – the Word Embedding – and in the semantic information already in the model – the Visual Embedding. In Table 16 we have selected a few Anchors to show how the knowledge of each Anchor collection is complementary. The table shows the Anchors closest to the selected ones in both spaces. We can see then that the Word space brings general semantic information based on the "meaning" of each word; and the Visual space on the other hand is focused on the similarities that we can see "with our eyes". If we look at the "wineglass" word for example, "wine-bottle" is the closest Word Anchor while

Anchor	Top 5 closest Word Semantic Anchors				
arm	head	hand	ear	helmet	nose
wineglass	wine-bottle	vase	chandelier	beer-mug	spoon
tv	ipod	sponge bob	radio	satellite dish	couch
bee	frog	butterfly	squirrel	mosquito	owl
bicycle	skateboard	motorbike	person walking	van	backpack
octopus	frog	dolphin	snail	spider	sea turtle
squirrel	owl	rabbit	cat	mouse (animal)	frog
standing bird	flying bird	owl	squirrel	pigeon	frog
camera	microphone	binoculars	computer monitor	cell phone	walkie talkie
speed-boat	sailboat	helicopter	motorbike	ship	submarine

Anchor	Top 5 closest Visual Semantic Anchors				
arm	trousers	face	person sitting	head	santa claus
wineglass	beer-mug	wine-bottle	diamond	lightbulb	hourglass
tv	computer monitor	keyboard	loudspeaker	bookshelf	radio
bee	mosquito	cactus	bush	butterfly	leaf
bicycle	motorbike	wheel	skateboard	tire	person walking
octopus	fish	crab	sea turtle	frog	cactus
squirrel	mouse (animal)	rabbit	hedgehog	monkey	pigeon
standing bird	pigeon	duck	seagull	flying bird	swan
camera	microscope	binoculars	radio	walkie talkie	flashlight
speed-boat	sailboat	dolphin	submarine	helicopter	ship

Table 16 – Comparing the different semantic interpretations of the Word and Visual Anchors on the Tu-Berlin train set. We've selected 10 Anchors and show the 5 Anchors closest to them in each space. We can see how the knowledge is complementary, with the Word Anchors encapsulating general representations based on "meaning" and the Visual Anchors capturing visual similarities.

"beer-mug" is the closest Visual Anchor. We believe both of these relationships are relevant to the ZSSBIR problem and therefore use both as Anchors and to guide the contrastive loss.

Matching the output of the model to these spaces requires architectural modifications; we swap the baseline's learnable token RTR for another two: RTR_{word} and RTR_{visual}, responsible respectively for matching the Word and Visual embeddings at the output. With two outputs we have two new InfoNCE losses for matching sketches and images in each space, $\mathscr{L}^{w}_{NCE}$ and $\mathscr{L}^{v}_{NCE}$, which are averaged. Matching outputs of either sketches or images to their respective anchor is also implemented with (semantic) InfoNCE:

$$\mathscr{L}^{sem}_{NCE} = -log\frac{exp(q^{w}.c^{w}_{+}/\tau)}{\sum_{i=0}^{C} exp(q^{w}.c^{w}_{i}/\tau)} - log\frac{exp(q^{v}.c^{v}_{+}/\tau)}{\sum_{i=0}^{C} exp(q^{v}.c^{v}_{i}/\tau)} \tag{5.2}$$

with c^{w}_{+} and c^{v}_{+} being the anchors for the input's class on each embedding.

Finally, because the model needs to have a single output for retrieval, we aggregate the output vectors using Self-attention $sa(.)$, as designed by Sukhbaatar *et al.* (2015a) and previously used in Chapters 3 and 4. The weights in $sa(.)$ learn to perform a dynamic weighted average

of the vectors, and we compute queries $q = sa(q^w, q^v)$ and keys $k = sa(k^w, k^v)$ for sketches and images. These are trained as in the baseline with $\mathscr{L}_{NCE}^{base}$.

5.3.4 Anchored Contrastive Loss

Because matching the geometry of the Semantic Anchor spaces yields a performance boost we design a new take on the InfoNCE contrastive loss tailored to further drive the model towards matching those spaces: *Anchored Contrastive Loss*. We start by building the similarity matrices for each embedding:

$$\begin{aligned} A^w &= \{a_{ij}^w\}, \text{with } a_{ij}^w = c_i^w . c_j^w \\ A^v &= \{a_{ij}^v\}, \text{with } a_{ij}^v = c_i^v . c_j^v \end{aligned} \tag{5.3}$$

Then the similarities are used as probability distributions. This changes the traditional InfoNCE's goal of strongly discriminating k_+ from the other keys to smoothly matching the distribution of the similarities between the classes being compared. This new objective is then much better aligned to ours of categorical retrieval. We change $\mathscr{L}_{NCE}^s$ for $\mathscr{L}_{anc}^s$:

$$\sigma(a_{ci}^s) = \frac{exp(a_{ci}/\tau)}{\sum_{j=0}^{K} exp(a_{cj}/\tau)} \tag{5.4}$$

$$\mathscr{L}_{anc}^s = -\sum_{i=0}^{K} \sigma(a_{ci}^s) log \frac{exp(q.k_i/\tau)}{\sum_{j=0}^{K} exp(q.k_j/\tau)} \tag{5.5}$$

where s is either w or v, σ is the softmax function and c is the index of q's class; k_c in this equation is then equivalent to k_+ in $\mathscr{L}_{NCE}^s$. One can notice that our Anchored Contrastive Loss resembles a cross-entropy loss in the form $-a \log b$, computing cross-entropy between the similarities of the samples and the similarities of the anchors, both normalised into probability distributions.

We similarly apply this idea to the semantic matching Equation 5.2, swapping $\mathscr{L}_{NCE}^{sem}$ for $\mathscr{L}_{anc}^{sem}$. For these losses the new loss computes cross-entropy between the similarity between samples and anchors and the similarities between anchors, further approximating features to the anchored embedding. The final representation, post-self-attention, is still trained with classic InfoNCE loss since we don't have a similarity distribution to match the aggregated retrieval vector.

5.3.5 Adapted Semantic Anchors

With the major components in place, we complete the model with two additions to the *Semantic Anchors*; first, we make them more adaptable to the task, and second, we make the input embedding more dynamic.

To let the anchors learn with the losses, we make use of a Graph Convolutional Network. First we create for each embedding a fully connected graph $\mathscr{G}^s = \{\mathscr{V}^s, \mathscr{E}^s\}$ composed of the

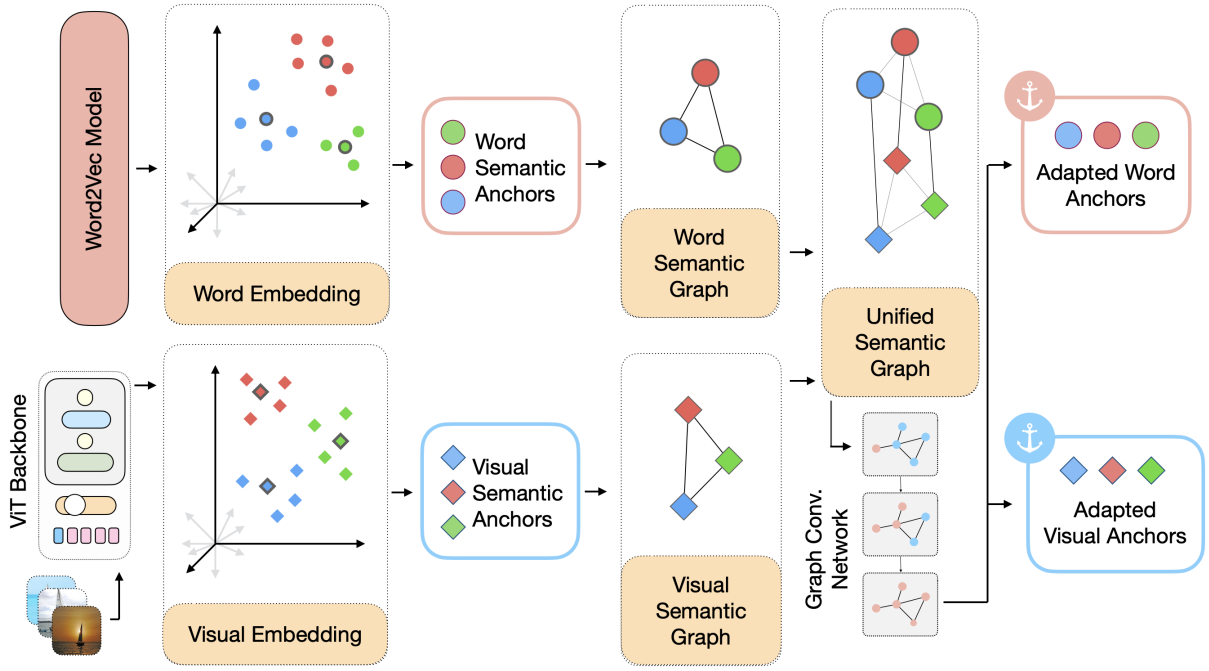


Figure 32 – Diagram depicting how the Adapted Semantic Anchors are learned. We first create fully connected graphs with the original anchors and their similarity matrices, then we connect the two graphs by the matching categories. Finally we process the unified graph through a Graph Convolutional Network and input the Adapted Anchors to our model.

embedding vectors as vertices $\mathcal{V}^s$ and the similarity matrix A^s as the adjacency matrix that makes up the edges $\mathcal{E}^s$. These graphs, $\mathcal{G}^w$ and $\mathcal{G}^v$ for Word and Visual embedding, are unified by connecting matching classes together with edges with the same-weight as the self-connections (*e.g.* the c_i^w vector for "cat" is connected to the c_i^v vector for "cat"), this allows the embeddings to share knowledge. We normalize the edge values by applying softmax line-wise to the adjacency matrix and follow Kipf and Welling (2016) to build a GCN:

$$(\mathcal{V}^{w+v})^{l+1} = \hat{A}^{w+v}(\mathcal{V}^{w+v})^l W^l \tag{5.6}$$

where the superscript $w+v$ indicates these are the unified graphs, $\hat{A}^{w+v}$ is the normalised adjacency matrix, l is a layer index, meaning these can be stacked, and W^l are learnable weights for each layer. The output nodes $(\mathcal{V}^{w+v})^L$ are used as the *Adapted Word* and *Visual Anchors*; We've included a diagram in Fig. 32 depicting this process.

5.3.6 Randomised Input Semantic Anchors

Finally, we make the input embeddings more dynamic and therefore more general. For each iteration, Word Embedding vectors are randomly swapped for one of the 10 closest word vectors in the full Word2Vec space; this means for example that the vector for "cat" can be swapped for the vectors for "cats" or "feline". For the Visual Embedding, instead of taking the class centres directly we use the centre and standard deviation of each class to make a normal distribution and generate slight variants of class Anchors to use as input. The similarity matrices A^w and A^v are updated together with the anchor variants.

The final Sketch-an-Anchor model is then trained with losses:

$$\mathscr{L}_{SAnc} = \mathscr{L}_{NCE}^{base} + \frac{1}{2}\mathscr{L}_{anc}^{w} + \frac{1}{2}\mathscr{L}_{anc}^{v} + \mathscr{L}_{anc}^{sem} \tag{5.7}$$

5.4 Experiments

In this section we detail the benchmark datasets used, implementation details, as well as empirical evidence of our model's performance. We also show how we've used the baseline to determine how few iterations are necessary and demonstrate empirically our hypothesis that the task regards mapping sketches to the already excellent image space.

5.4.1 Datasets

We follow most ZSSBIR literature and benchmark our model across three datasets, Sketchy (SANGKLOY *et al.*, 2016a), Tu-Berlin (EITZ; HAYS; ALEXA, 2012) and Quickdraw (THE..., 2018).

Sketchy is composed of 75,471 sketches of 125 categories and originally had 12,500 natural images, but has since had this number expanded by Shen *et al.* (2018) 73,002 natural images. We consider two zero-shot splits, one mirroring SAKE (LIU *et al.*, 2019) where 25 classes were chosen randomly to compose the zero-shot test set; and another created by Yelamarthi *et al.* (2018a) with 21 zero-shot classes chosen specifically to avoid overlap with ImageNet. We refer to the latter as the Non-Overlapping Sketchy set (Sketchy N.O.)

Tu-Berlin contains 20k sketches from 250 categories and was extended by Shen *et al.* (2018) with 204k images matching such categories. An unrestricted selection of 30 zero-shot classes is used; for comparability we use the same selection as SAKE (LIU *et al.*, 2019).

QuickDraw is the more recent and the largest set; Dey *et al.* (2019a) selected 110 of the 330 classes available based on how discriminative they were to humans and avoiding overlap with ImageNet; a set of 330k sketches and 204k images was built to compose QuickDraw Extended; we follow other studies in the choice of 30 classes for zero-shot.

5.4.2 Implementation Details

We use the Tensorflow 2 framework to build our models and run all of our experiments on a single NVIDIA RTX A5000; each experiment is repeated 5 times and we take the average metrics on evaluation. As mentioned, our backbone is a ViT-S/8, trained under the DINO protocol (DINO-S/8), this model comprises 12 layers with 6 attention heads in each. The gated linear projections used at the outputs all have 512 units. The GCN for the Semantic Anchors has 4 layers and each also has 512 units. We multiply the gradient to the backbone by 0.1 to accelerate learning of the projectors.

All of our models are trained with a batch size of 16 for 1500 iterations. The Adam optimizer with learning rate $\eta = 5e{-}6$ is used and η is warmed up (raised lineraly) to the base value within the first 150 iterations and is then decayed to $1e{-}6$ with a cosine schedule. In the next Section (5.4.3) we explore our choice of number of iterations and learning rate.

5.4.3 A Thousand Iterations are All You Need

One of our goals with this paper is to show that pre-trained models such as DINO can be quickly adapted to do ZSSBIR, so we designed a simple experiment to find how many iterations are needed to achieve good cross-domain performance.

To do this we created a validation set for Sketchy N.O., where we remove 10 classes from the training set, randomly picking 64 images and 64 sketches from each, making for a compact and easy to evaluate set. These classes are then not included for training. Our Baseline model is trained with Adam and fixed learning rates for 10 thousand iterations. The results of this experiment can be found plotted in Fig. 33.

It is clear that regardless of learning rate, the models reach a peak mAP value fairly quickly after around one thousand iterations. The number of iterations where performance peaks is probably enough for it to see most of the pair diversity and learn to map sketches to the image space. We select the lower learning rate $\eta = 5e{-}6$ for its stability and higher peak mAP, stopping our models at 1500 iterations instead of a thousand to allow for the inclusion of warm-up and a cosine schedule on the learning rate value.

5.4.4 Comparison Against Other Methods

We compare our Sketch-an-Anchor model against nine of the best methods in the literature. We furthermore complement the comparison with two sub-tasks: Binarized ZSSBIR and Generalised ZSSBIR. In the former retrieval is done through hamming distance and 64-dimension binary vectors are used as representations; we've followed other methods and used Iterative Quantisation (ITQ) (GONG *et al.*, 2012) to make our representations binary.

The results for classic ZSSBIR and Binarized ZSSBIR (subscript b) can be found in Table 17, and some qualitative ZSSBIR results for our model are on Figures 34 and 35. We can see that our results on ZSSBIR are very competitive against the SoTA, often being around 1% shy of the best performing model in each metric. We've included the number of training epochs (Ep.) for methods that report it and highlight that our method is trained end-to-end in 1500 iterations, being on average 114 times faster to train (0.87% the number of iterations of other methods).

We take special note of the comparison against TVT (TIAN *et al.*, 2022) and PKSD (WANG *et al.*, 2022) since both share a backbone (DINO S/8) with our model but do not enjoy the benefit of its fast adaptation properties. TVT implements three simultaneous ViT instances



Figure 33 – Evaluation on the Sketchy (Split (YELAMARTHI *et al.*, 2018a)) validation set across 10 thousand iterations with the Baseline model. The x-axis shows iteration and the y-axis shows mAP (%). We test three learning rate values and select to use $\eta = 5e{-}6$ and to stop training at 1500 iterations (dotted line on the bottom-most graph) for all our following experiments.

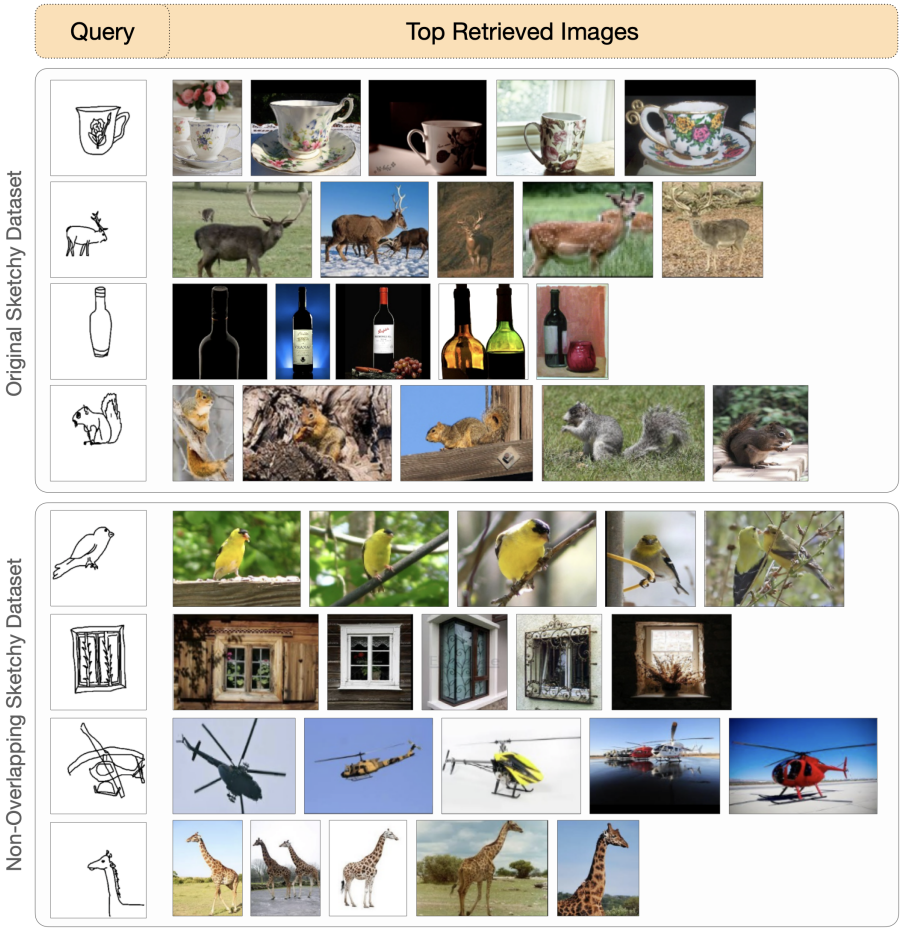


Figure 34 – Visual ZSSBIR results. For each query sketch we show the 5 top images retrieved. The first block is from the Sketchy dataset's original split, the next block from the Non-Overlapping (YELAMARTHI *et al.*, 2018a) split.

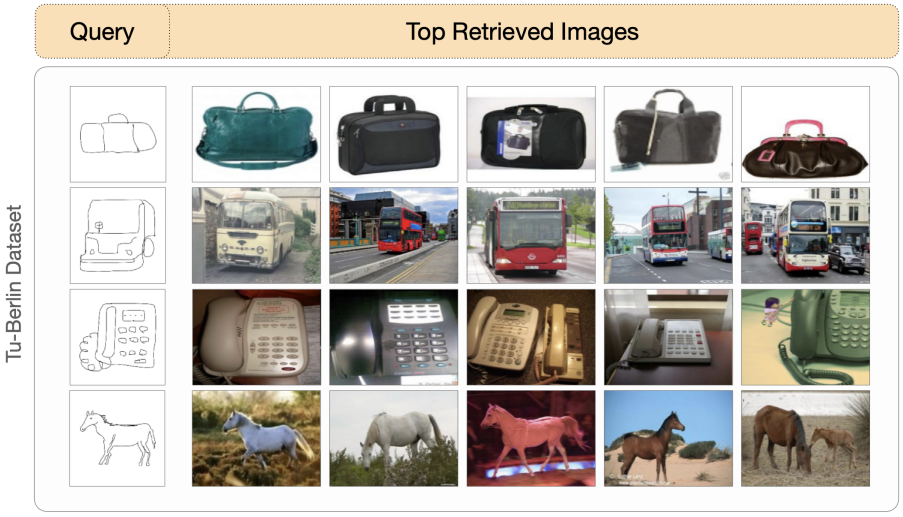


Figure 35 – Visual ZSSBIR results. For each query sketch we show the 5 top images retrieved. Results are from the Tu-Berlin dataset.

and trains for 50 epochs while PKSD uses two instances and trains for 30 epochs; they both use knowledge distillation to avoid losing the knowledge already in the pre-trained model. Our model doesn't need distillation because the training protocol of few iterations and low learning rate by itself keeps the knowledge from deprecation or "catastrophic forgetting".

The performance of Sketch-an-Anchor in Binarized ZSSBIR drops in comparison. We believe this has to do with the compactness of the search space. Because our model retains much of the original properties of the pre-trained DINO, it is also a more general model and likely "covers" more of the embedding space. Such a space is harder for the ITQ algorithm to work with as it relies on Principal Component Analysis to quantize the features.

The task of Generalized ZSSBIR, GZSSBIR, considers the search corpus to have images from both seen and unseen classes and we follow the literature to take 20% of the images from each class in the training set and adding it to the test set instead; queries remain being restricted to unseen images. Not all methods test under this task, we've included our results against three of the nine aforementioned competitors in Table 18. The results here reflect the ones in classic ZSSBIR, with our model producing competitive results while being fast to train.

5.4.5 An Image is Worth an Entire Class

We argue that when using powerful pre-trained models ZSSBIR can be taken as the task of mapping sketches to the image space while keeping representations general enough for the zero-shot protocol. Our Semantic Anchors, Visual Anchors specially, follow this thread, we introduce guiding points in the space so that images can stay within their representations and sketches can follow.

We want to pull this thread further and ask the question: if the sketches are the ones that need to be mapped, could we use only a representative sub-set of the images to train the model? Furthermore, how representative the Visual Anchors (class centres) are of the image space?

To answer these questions we've designed a new experiment: instead of taking the entire image set to train our model, we make two "opposing" sets; in one we select only the N images from each class closer to their Visual Anchors, in the other we select the N images farther from them. We repeat this with $N = \{200, 100, 50, 10, 5, 1\}$ and show the results in Fig. 36.

As is clear from the results, Sketch-an-Anchor retains its competitive performance even when only training with a single image from each class as long as this image matches its Visual Anchor. In both Sketchy splits performance is almost unchanged across all number of images and while the Tu-Berlin set needs at least 50 images to retain performance, that is still less than 10% of the original set. Finally, using only a single representative image improves the result on Quickdraw Extended to 16.6%, which we believe is due to how noisy the full image set is on this dataset.

The fact that the images closer to the Visual Anchors are able to train the model to the

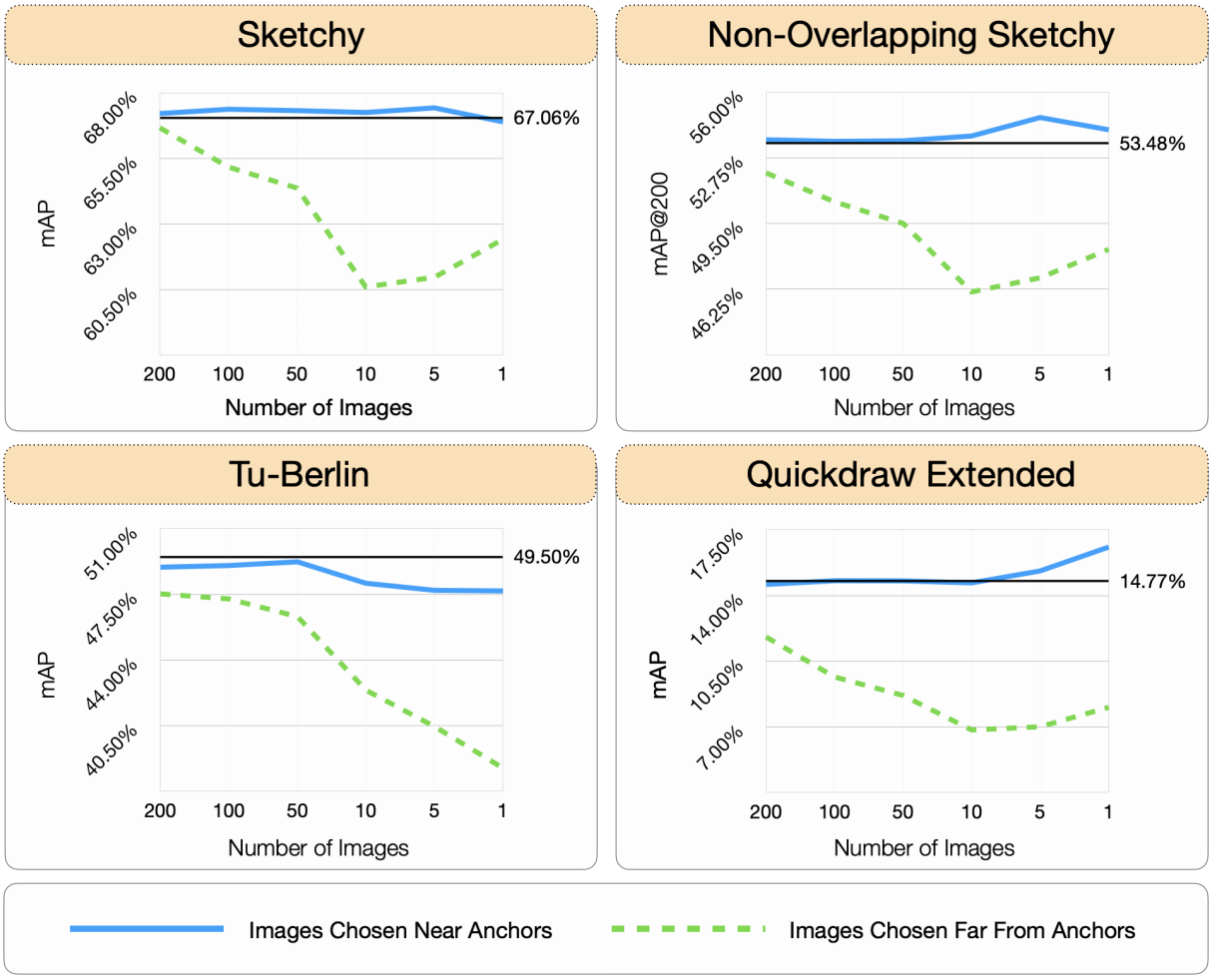


Figure 36 – Experiment with Image Selection. For each model we select the N (x-axis) images closer (blue, filled line) to their corresponding Visual Anchor or the N images further from them (green, dotted line). For each selection we train Sketch-an-Anchor using the full sketch set and the selected images. The y-axis shows mAP for each model (mAP@200 for Sketchy Non-Overlapping split). The black line marks the performance using the entire image set for training.

same performance level shows how relevant this side information is to the ZSSBIR task and our fast adapting Sketch-an-Anchor.

5.4.6 Correlation between Initial Image-to-Image Retrieval performance and Sketch-an-Anchor's

We have argued in Section 5.1 that generalisation to unseen data in the image domain already has good solutions in the literature. The design of Sketch-an-Anchor trusts that assumption, and therefore our model relies on pre-trained models that do well in Image to Image retrieval.

We show in this section:

- the strong correlation between the performance of different pre-training protocols and architectures in Image-based Image Retrieval and the final performance in ZSSBIR after

training these models with the Sketch-an-Anchor design;

- comparison of CNN-based models, which required adaptations to fit the Sketch-an-Anchor design. We have taken the output of the layer before the last and average pooled the resulting feature maps.

Because is is not possible to have multiple outputs on the CNNs as we can with ViT-based models, we've instead split the resulting vector in two, each split then being used as the Word and Visual Semantic Representation. We have found this adaptation to work better than simply applying two projections to the entire output vector. Other than this modification on CNN-based models, all experiments follow the same training procedure described in the main paper: 1500 iterations, $\eta = 5e{-}6$ and a batch size of 16 pairs.

We compare 5 network architectures across 4 training protocols. The network architectures are the Vision Transformer (ViT) from Dosovitskiy *et al.* (2021), in both S/8 and S/16 setups; EfficientNet (TAN; LE, 2019), in B3 and B4 setups; ResNet (HE *et al.*, 2016) in ResNet50 setup. We compare the common supervised protocol and the distillation-based DeiT design (TOUVRON *et al.*, 2021). Finally, we compare two self-supervised protocols: DINO (CARON *et al.*, 2021) and SwAV (CARON *et al.*, 2020).

To motivate our choice of architecture and the core ideas of Sketch-an-Anchor we have already shown the visual comparison across architectures in Figure 29. The figure compares the Image-based Image Retrieval (IBIR), Sketch-based Sketch Retrieval (SBSR) and Sketch-based Image Retrieval (SBIR) tasks between pre-trained models and the final Sketch-an-Anchor version. From this visual comparison it is easy to see how the initial IBIR is reflected in the trained Sketch-an-Anchor.

We have also computed the correlation between initial IBIR mAP and final SBIR mAP. When taking all the models into account, we get a correlation coefficient $r = 0.49$ on Sketchy and $r = 0.94$ on Tu-Berlin. Since there are supervised pre-trained models in the collection, we have found that supervision on ImageNet makes these models worse at transferring knowledge to Sketchy N.O.; this is natural as the Sketchy split used here was selected to not have overlapping classes with ImageNet. We have then computed correlation coefficients only for the auto-supervised models (DINO and Swav protocols), resulting in $r = 0.75$ on Sketchy and $r = 0.98$ on Tu-Berlin.

Finally, ZSSBIR results for all protocols/architectures are shown in Table 19. These results further support the success of Sketch-an-Anchor at the goal of mapping sketches to the image space while maintaining the discriminability of this space. Finally, the experiments also show that ViT-based models have an advantage for cross-domain retrieval in general, as corroborated by other cross-domain designs in the literature (TIAN *et al.*, 2022; JAEGLE *et al.*, 2021b; JAEGLE *et al.*, 2021a; SHVETSOVA *et al.*, 2022).

5.4.7 Ablation Study

We follow the description in Sec. 5.3 and build our ablation studies in the same fashion as we've built the model, starting from the Baseline, including the Semantic Anchors (A), Anchored Contrastive loss (B) and finally the GCN and Randomised Semantic Input (C).

The results for each of our ablated models are depicted in Table 20 for the Tu-Berlin and Sketchy N.O. datasets. Looking first at the results on Tu-Berlin, it is possible to see that adding each of the Semantic Anchors bring improvements to the baseline and our Anchored Contrastive Loss $\mathscr{L}_{anc}$ further enhances performance. Completing the model, Randomised Semantic Input (R.) and the GCN work best when used together; this is an effect of how randomisation contributes to a GCN that produces more general Adapted Semantic Anchors. All together we add 2.9% to the baseline's mAP, enough to make it the second-best Tu-Berlin mAP in the literature.

The Sketchy set benefits differently from each contribution. While performance on Tu-Berlin increases with each addition, on Sketchy performance grows as "blocks" of contributions are added. The Semantic Anchors for example are not beneficial for performance on Sketchy before they are applied together with our Anchored Contrastive Loss, and the "Final Touches" – Graph Conv. Net on the Anchors and Randomized Semantic Input – do not bring the same improvements to Sketchy as they do to Tu-Berlin. We believe this is due to particularities of the Tu-Berlin set. It has double the number of classes of Sketchy for training and the classes are more similar to each other (as shown by Dutta and Akata (2020)). These two specifics increase the importance of having more general and more adaptable Semantic Anchors, which we achieve with the Randomized Semantic Input and the GCN.

5.5 Conclusion

In this chapter and in the original submission we have presented Sketch-an-Anchor, a new competitive model for ZSSBIR. The design starts from the insight that the problem of Zero-shot Image Retrieval is almost solved by Self-Supervision models such as DINO. Sketch-an-Anchor shows we can stop training models from scratch or fine-tuning for long; focus should go on strategically mapping sketches to the already excellent image space (Hypothesis **H3**). This is a general contribution to content-based image retrieval.

With our training protocol we were able to train a baseline model that is already among the best at ZSSBIR. Introducing our Semantic Anchors and the Anchored Contrastive Loss then pushes this model to be competitive with SoTA performance under the zero-shot protocol. Also, our experiments have shown that our model only needs a few (or even only one) images per class to match sketches to the image space and that each of our additions to the baseline brings improvements in retrieval performance. Such findings may significantly change the way future approaches tackle the problem, allowing for experimentation with new models at a much higher rate. Future work should explore if our findings apply to the Domain Generalisation and

Few-shot learning fields of study, both of which share some of the challenges we've addressed on ZSSBIR.

Methods	Sketchy			Sketchy N.O.		
	Ep.	mAP	P@100	Ep.	mAP@200	P@200
$SAKE_b$ (LIU *et al.*, 2019)	20	36.4	48.7	20	35.6	47.7
$SPCYC_b$ (DUTTA; AKATA, 2020)	-	34.4	39.9	-	-	-
$SBTK_b$ (TURSUN *et al.*, 2022)	25	47.4	61.0	25	41.0	51.5
TVT_b (TIAN *et al.*, 2022)	50	55.3	72.7	50	44.7	55.4
$PKSD_b$ (WANG *et al.*, 2022)	30	**62.7**	**75.0**	30	**48.6**	**58.2**
Ours$_b$	**0.4**	48.5	58.5	**0.4**	39.7	51.5
SAKE (LIU *et al.*, 2019)	20	54.7	69.2	20	49.7	59.8
DSN (WANG *et al.*, 2021)	10	58.3	70.4	-	-	-
PCSN (DENG *et al.*, 2020)	20	52.3	61.6	-	-	-
SPCYC (DUTTA; AKATA, 2020)	-	34.9	46.3	-	-	-
SGCN (ZHANG *et al.*, 2020)	-	-	-	-	**56.8**	48.7
SBTK (TURSUN *et al.*, 2022)	25	55.3	69.8	25	50.2	59.6
TVT (TIAN *et al.*, 2022)	50	64.8	**79.6**	50	53.1	61.8
BDA (CHAUDHURI *et al.*, 2022)	50	43.7	51.4	50	55.6	45.8
PKSD (WANG *et al.*, 2022)	30	**68.8**	78.6	30	56.0	**64.5**
Ours	**0.4**	67.1	76.2	**0.4**	53.5	63.0
	TU-Berlin			QuickDraw Extended		
	Ep.	mAP	P@100	Ep.	mAP	P@100
$SAKE_b$ (LIU *et al.*, 2019)	20	35.9	48.1	-	-	-
$SPCYC_b$ (DUTTA; AKATA, 2020)	-	29.3	39.2	-	**16.4**	**24.3**
$SBTK_b$ (TURSUN *et al.*, 2022)	25	39.4	51.4	-	-	-
TVT_b (TIAN *et al.*, 2022)	50	39.6	60.6	-	-	-
$PKSD_b$ (WANG *et al.*, 2022)	30	**41.9**	**60.8**	-	-	-
Ours$_b$	**0.1**	36.9	49.7	**0.1**	10.8	15.2
SAKE (LIU *et al.*, 2019)	20	47.5	59.9	-	-	-
DSN (WANG *et al.*, 2021)	10	48.4	58.6	-	-	-
PCSN (DENG *et al.*, 2020)	20	42.4	51.7	-	-	-
SPCYC (DUTTA; AKATA, 2020)	-	29.7	42.6	-	**17.7**	25.5
SGCN (ZHANG *et al.*, 2020)	-	32.4	50.5	-	-	-
SBTK (TURSUN *et al.*, 2022)	25	48.0	60.8	25	11.9	-
TVT (TIAN *et al.*, 2022)	50	48.4	**66.2**	50	14.9	**29.9**
BDA (CHAUDHURI *et al.*, 2022)	50	37.4	50.4	50	15.4	28.6
PKSD (WANG *et al.*, 2022)	30	**50.2**	**66.2**	30	15.0	29.7
Ours	**0.1**	49.5	60.8	**0.1**	14.8	21.2

Table 17 – Comparison of Sketch-an-Anchor against nine methods from the literature at the ZSSBIR and Binarized ZSSBIR tasks. We compare on both splits of the Sketchy, on Tu-Berlin and on Quickdraw Extended. The subscript *b* refers to results with 64d hashing codes (Binarized ZSSBIR). Missing results were not reported in the original papers. We also highlight the number of Epochs (Ep.) each model takes to train when reported and bold and underline the best and second-best results for all metrics.

Methods	Sketchy		TU-Berlin	
	mAP	P@100	mAP	P@100
SEM-PCYC (DUTTA; AKATA, 2020)	30.7	29.6	19.2	21.8
SBTKNet (TURSUN *et al.*, 2022)	51.5	57.2	**33.4**	**49.4**
BDA (CHAUDHURI *et al.*, 2022)	33.8	41.3	25.1	35.7
Ours	**52.5**	**59.0**	29.0	38.1

Table 18 – Comparison of Sketch-an-Anchor against 3 methods from the literature at the Generalized ZSSBIR task. We compare on the random split of Sketchy and on Tu-Berlin to match existing reports.

Protocol	Architecture	Sketchy N.O.		TU-Berlin	
		mAP@200	P@200	mAP	P@100
DINO	ViT S/8	53.48	62.97	49.50	60.82
DINO	ViT S/16	45.46	56.14	44.43	59.11
Supervised (DeiT)	ViT S/16	47.24	57.14	52.00	60.69
Supervised	EfficientNet B3	43.25	53.87	43.14	56.51
Supervised	EfficientNet B4	44.45	54.39	45.23	56.11
DINO	Resnet50	38.79	50.80	32.61	49.31
SwAV	Resnet50	39.70	50.96	34.54	49.59
Supervised	Resnet50	42.62	53.88	39.33	53.83

Table 19 – ZSSBIR Performance across architectures and pre-training protocols. ViT-based models perform better in general and Self-Supervised training protocols improve how well knowledge transfers to the cross-domain task.

ID	E_v	E_w	$\mathscr{L}_{anc}$	Rand.	GCN	Tu-Berlin		Sketchy N.O.	
						mAP	P@100	mAP@200	P@200
Baseline						47.06	59.65	52.90	62.09
A1	✓					47.76	59.45	51.76	61.58
A2	✓	✓				47.86	59.15	51.85	61.67
B	✓	✓	✓			48.47	59.63	54.05	63.56
C1	✓	✓	✓	✓		48.66	60.03	52.94	62.70
C2	✓	✓	✓		✓	48.96	60.22	53.65	63.17
Final Model	✓	✓	✓	✓	✓	49.50	60.82	53.48	62.97

Table 20 – Evaluation of ablated models on the Tu-Berlin and Sketchy N.O., reported with mean average precision (mAP, mAP@200 on Sketchy) and precision at k ($k = 100$ for Tu-Berlin, $k = 200$ for Sketchy). We refer to the Semantic Anchors as E_v (Visual) and E_w (Word), and refer to Randomised Semantic Input as Rand.

CHAPTER

6

CONCLUSION

We have set the goal for this thesis to advance the field of cross-domain representation learning, using the task of Sketch-based Image Retrieval (SBIR) as a canvas. The study of sketch-related tasks represents well other cross-domain representation problems for its high level of abstraction. Furthermore, research in SBIR is aided by the maturity of the field, with well established benchmarks and the existence of research groups around the world specialised in sketch.

In this thesis we have taken on three challenges of SBIR and sketch representation learning: (i) how to learn better representations for abstract sketches; (ii) how to adapt our solutions for multi-object, composed scenes; (iii) how to train models that generalise beyond seen sketches and images. For each challenge we have answered with one direct contribution.

In Chapter 3 we *adapted the Transformer architecture* from the field of natural language processing to learn representations for vector sketches. The learned representation was shown to be useful for semantic sketch synthesis and interpolation, as well as for SBIR and sketch classification. This shows that the Transformer architecture's successes on sentence representation could well be applied to the vector sequences that represent sketches (Hypothesis **H1**). The multi-task nature of the representation shows how apparent disparate tasks such as synthesis and retrieval are complementary to achieve good representations (Hypothesis **H4**).

Our success with the Transformer architecture and different ways of encoding sketches show that it is possible to rethink how we represent these abstract drawings. We experimented with a sequence of vector strokes for our Sketchformer input and have encoded it with alternative tokenisers. The work of Qi *et al.* (2021) takes a grid over rasterised sketches and takes the stroke-grid overlapping points to build a graph representing the sketch; this system is then used in a model that "heals" corrupted sketches by guessing the original intent. Other solutions for this task also experiment with distinct initial sketch encodings, the studies of Qi *et al.* (2022), Zang, Tu and Xu (2022) similarly represent sketches as graphs, with nodes sampled along their

vector representations and represented by rasterised image patches from the node's location in the vector sketch. With this many disparate solutions, there is space for future work to both explore different combinations of encodings and to build upon those that are finding success now.

While the field of SBIR is well explored in the literature, it is also a useful task that has seldom been implemented at scale. Two of our contributions in this thesis are then aimed at alternative SBIR definitions for more applicable and realistic SBIR. In Chapter 4 Scene Designer is presented: *a state-of-the-art model for Compositional SBIR*. This new task approximates the desirable user experience for an SBIR system; while traditional SBIR constrains the task to single object sketches and images, in Compositional SBIR complete scenes can be drawn and retrieved. Our model was designed around what is known to work in traditional SBIR – multi-stream networks and triplet loss training – while combining it with GNN-based Scene Graph representations from the image synthesis literature (Hypothesis **H2**). The performance of Scene Designer more than doubles that of previous models on public benchmarks and we follow Hypothesis **H4**, combining an image synthesis task with retrieval, to further improve the model.

Finally, Chapter 5 presents *a fast state-of-the-art model for Zero-shot SBIR*, another alternative definition for SBIR that restricts the classes in evaluation to be disjointed with the ones used for training. This makes for models that are more adaptable and that learn sketch-image relationships that can be applied independent of object category. Our method shows it is possible to harvest the knowledge encoded in SoTA pre-trained models for images that already show good generalisation properties, learning sketch representations by mapping to these existing spaces and enjoying these properties in a cross-domain setting (Hypothesis **H3**). Sketch-an-Anchor is also shown to be a competitive model in ZSSBIR while requiring more than one hundred times less data to train than its counterparts in the literature. Together with Scene Designer, these contributions bring the task of SBIR closer to applicability and we trust future work to build upon them, making a reality the generalisable and unconstrained model that combines single-object, Compositional and Zero-shot SBIR at scale.

In this thesis then we introduced *a new architecture for learning sketch representations and state-of-the-art models for the realistic tasks of Compositional SBIR and Zero-shot SBIR*. Each of these contributions furthers the field of sketch, image and cross-domain representations, with cascading impact that goes beyond the sketch domain. While we study and present contributions with sketches – one of our most abstract forms of expression, and one of the most studied in Image Understanding literature – that is not the "be all and end all" of our research. As discussed in Chapter 1, to augment and scale the very human capacity of distinguishing objects and ideas regardless of domain is the driving principle of the presented work.

The field of Domain Generalisation (DG) (ZHOU *et al.*, 2022) for instance has been growing consistently; DG as a field covers the the broadly-defined task of training models under one domain (here defined as samples from a specific probability distribution) and having

these models work on out-of-distribution domains. With SBIR following a similar definition but being domain-specific, it is reasonable to assume that both fields should benefit from sharing techniques. The same can also be said of other cross-domain retrieval tasks like text-to-video (SHVETSOVA *et al.*, 2022) and sketch-to-shape (QI *et al.*, 2021). We close this thesis in hope that further research will build upon our contributions across Image Understanding and Cross-domain representations at large, combining new ideas with ours to build better systems that assist and augment what us, as humans, can do.

BIBLIOGRAPHY

ABDUL-RASHID, H.; YUAN, J.; LI, B.; LU, Y.; SCHRECK, T.; BUI, N.-M.; DO, T.-L.; HOLENDERSKI, M.; JARNIKOV, D.; LE, T.-K.; MENKOVSKI, V.; NGUYEN, K.-T.; NGUYEN, T.-A.; NGUYEN, V.-T.; NINH, V.-T.; REY, L. A. P.; TRAN, M.-T.; WANG, T. Extended 2d scene image-based 3d scene retrieval. In: **3DOR@Eurographics**. [S.l.: s.n.], 2019. Citation on page 93.

ABE, K.; MORITA, H.; HAYASHI, T. Similarity retrieval of trademark images by vector graphics based on shape characteristics of components. In: . New York, NY, USA: Association for Computing Machinery, 2018. (ICCAE 2018), p. 82–86. ISBN 9781450364102. Citation on page 93.

AL-SHEDIVAT, M.; BANSAL, T.; BURDA, Y.; SUTSKEVER, I.; MORDATCH, I.; ABBEEL, P. Continuous adaptation via meta-learning in nonstationary and competitive environments. **arXiv preprint arXiv:1710.03641**, 2017. Citation on page 46.

ASHLEY, J.; FLICKNER, M.; HAFNER, J.; LEE, D.; NIBLACK, W.; PETKOVIC, D. The query by image content (qbic) system. In: **Proceedings of the 1995 ACM SIGMOD International conference on Management of data**. [S.l.: s.n.], 1995. p. 475. Citations on pages 22 and 96.

ASHUAL, O.; WOLF, L. Specifying object attributes and relations in interactive scene generation. In: **Proceedings of the IEEE International Conference on Computer Vision**. [S.l.: s.n.], 2019. p. 4560–4568. ISBN 9781728148038. ISSN 15505499. Citations on pages 71, 75, 76, 79, 82, 86, 87, 88, and 89.

BAHDANAU, D.; CHO, K.; BENGIO, Y. Neural machine translation by jointly learning to align and translate. **arXiv preprint arXiv:1409.0473**, 2014. Citation on page 40.

BANSAL, A.; SIKKA, K.; SHARMA, G.; CHELLAPPA, R.; DIVAKARAN, A. Zero-shot object detection. In: **Proceedings of the European Conference on Computer Vision (ECCV)**. [S.l.: s.n.], 2018. p. 384–400. Citation on page 46.

BART, E.; ULLMAN, S. Single-example learning of novel classes using representation by similarity. In: CITESEER. **BMVC**. [S.l.], 2005. v. 1, p. 2. Citation on page 45.

BATTAGLIA, P. W.; HAMRICK, J. B.; BAPST, V.; SANCHEZ-GONZALEZ, A.; ZAMBALDI, V.; MALINOWSKI, M.; TACCHETTI, A.; RAPOSO, D.; SANTORO, A.; FAULKNER, R. *et al.* Relational inductive biases, deep learning, and graph networks. **arXiv preprint arXiv:1806.01261**, 2018. Citations on pages 45 and 75.

BENGIO, Y.; LAUFER, E.; ALAIN, G.; YOSINSKI, J. Deep generative stochastic networks trainable by backprop. In: **International Conference on Machine Learning**. [S.l.: s.n.], 2014. p. 226–234. Citation on page 33.

BHUNIA, A. K.; CHOWDHURY, P. N.; SAIN, A.; YANG, Y.; XIANG, T.; SONG, Y.-Z. More photos are all you need: Semi-supervised learning for fine-grained sketch based image retrieval. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2021. p. 4247–4256. Citation on page 29.

BHUNIA, A. K.; CHOWDHURY, P. N.; YANG, Y.; HOSPEDALES, T. M.; XIANG, T.; SONG, Y.-Z. Vectorization and rasterization: Self-supervised learning for sketch and handwriting. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2021. p. 5672–5681. Citation on page 39.

BHUNIA, A. K.; KHAN, S.; CHOLAKKAL, H.; ANWER, R. M.; KHAN, F. S.; LAAKSONEN, J.; FELSBERG, M. Doodleformer: Creative sketch drawing with transformers. In: SPRINGER. **European Conference on Computer Vision**. [S.l.], 2022. p. 338–355. Citation on page 43.

BROWN, T. B.; MANN, B.; RYDER, N.; SUBBIAH, M.; KAPLAN, J.; DHARIWAL, P.; NEELAKANTAN, A.; SHYAM, P.; SASTRY, G.; ASKELL, A.; AGARWAL, S.; HERBERT-VOSS, A.; KRUEGER, G.; HENIGHAN, T.; CHILD, R.; RAMESH, A.; ZIEGLER, D. M.; WU, J.; WINTER, C.; HESSE, C.; CHEN, M.; SIGLER, E.; LITWIN, M.; GRAY, S.; CHESS, B.; CLARK, J.; BERNER, C.; MCCANDLISH, S.; RADFORD, A.; SUTSKEVER, I.; AMODEI, D. Language models are few-shot learners. In: LAROCHELLE, H.; RANZATO, M.; HADSELL, R.; BALCAN, M.; LIN, H. (Ed.). **Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual**. [S.l.: s.n.], 2020. Citations on pages 21, 34, 42, and 70.

BUI, T.; COLLOMOSSE, J. Scalable sketch-based image retrieval using color gradient features. In: **Proc. ICCV Workshops**. [S.l.: s.n.], 2015. p. 1–8. Citations on pages 22, 23, and 49.

BUI, T.; RIBEIRO, L.; PONTI, M.; COLLOMOSSE, J. Generalisation and sharing in triplet convnets for sketch based visual search. **CoRR Abs**, arXiv:1611.05301, 2016. Citation on page 24.

BUI, T.; RIBEIRO, L.; PONTI, M.; COLLOMOSSE, J. Compact descriptors for sketch-based image retrieval using a triplet loss convolutional neural network. **CVIU**, Elsevier, 2017. Citations on pages 24, 29, 37, 48, 49, 65, 68, 70, and 74.

BUI, T.; RIBEIRO, L.; PONTI, M.; COLLOMOSSE, J. Sketching out the details: Sketch-based image retrieval using convolutional neural networks with multi-stage regression. **Computers & Graphics**, Elsevier, v. 71, p. 77–87, 2018. Citations on pages 24, 29, 37, 49, 57, 58, 64, 65, 68, 70, 74, 81, 84, 86, and 87.

BUI, T.; RIBEIRO, L.; PONTI, M. A.; COLLOMOSSE, J. Deep Manifold Alignment for Mid-grain Sketch based Image Retrieval. 2018. Available: <http://personal.ee.surrey.ac.uk/Personal/J.Collomosse/pubs/Bui-ACCV-2018.pdf>. Citations on pages 24 and 140.

CAESAR, H.; UIJLINGS, J.; FERRARI, V. Coco-stuff: Thing and stuff classes in context. In: IEEE. **Proc. CVPR**. [S.l.], 2018. Citations on pages 82 and 83.

CAO, Y.; WANG, H.; WANG, C.; LI, Z.; ZHANG, L.; ZHANG, L. Mindfinder: Interactive sketch based image search on millions of images. In: **Proc. ACM MM**. [S.l.: s.n.], 2010. Citation on page 49.

CARLIER, A.; DANELLJAN, M.; ALAHI, A.; TIMOFTE, R. Deepsvg: A hierarchical generative network for vector graphics animation. **Advances in Neural Information Processing Systems**, v. 33, p. 16351–16361, 2020. Citation on page 43.

CARON, M.; MISRA, I.; MAIRAL, J.; GOYAL, P.; BOJANOWSKI, P.; JOULIN, A. Unsupervised learning of visual features by contrasting cluster assignments. **Advances in Neural**

Information Processing Systems, v. 33, p. 9912–9924, 2020. Citations on pages 39, 98, 101, and 112.

CARON, M.; TOUVRON, H.; MISRA, I.; JÉGOU, H.; MAIRAL, J.; BOJANOWSKI, P.; JOULIN, A. Emerging properties in self-supervised vision transformers. In: **Proceedings of the IEEE/CVF International Conference on Computer Vision**. [S.l.: s.n.], 2021. p. 9650–9660. Citations on pages 39, 40, 97, 98, 100, 101, and 112.

CAVALLARI, G. B.; PONTI, M. A. Semi-supervised siamese network using self-supervision under scarce annotation improves class separability and robustness to attack. In: IEEE. **2021 34th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)**. [S.l.], 2021. p. 223–230. Citation on page 39.

CAVALLARI, G. B.; RIBEIRO, L. S.; PONTI, M. A. Unsupervised representation learning using convolutional and stacked auto-encoders: a domain and cross-domain feature space analysis. In: IEEE. **2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)**. [S.l.], 2018. p. 440–446. Citation on page 140.

CHAUDHURI, U.; BANERJEE, B.; BHATTACHARYA, A.; DATCU, M. A simplified framework for zero-shot cross-modal sketch data retrieval. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops**. [S.l.: s.n.], 2020. p. 182–183. Citations on pages 25 and 99.

CHAUDHURI, U.; CHAVAN, R.; BANERJEE, B.; DUTTA, A.; AKATA, Z. Bda-sketret: Bi-level domain adaptation for zero-shot sbir. **arXiv preprint arXiv:2201.06570**, 2022. Citations on pages 115 and 116.

CHEN, B.; ROUDITCHENKO, A.; DUARTE, K.; KUEHNE, H.; THOMAS, S.; BOGGUST, A.; PANDA, R.; KINGSBURY, B.; FERIS, R.; HARWATH, D. *et al.* Multimodal clustering networks for self-supervised learning from unlabeled videos. In: **Proceedings of the IEEE/CVF International Conference on Computer Vision**. [S.l.: s.n.], 2021. p. 8012–8021. Citation on page 101.

CHEN, M.; RADFORD, A.; CHILD, R.; WU, J.; JUN, H.; LUAN, D.; SUTSKEVER, I. Generative pretraining from pixels. In: PMLR. **International conference on machine learning**. [S.l.], 2020. p. 1691–1703. Citation on page 43.

CHEN, T.; CHENG, M.-M.; TAN, P.; SHAMIR, A.; HU, S.-M. Sketch2photo: Internet image montage. **Proc. ACM SIGGRAPH**, ACM, v. 28, n. 5, p. 124, 2009. Citation on page 71.

CHEN, T.; KORNBLITH, S.; NOROUZI, M.; HINTON, G. A simple framework for contrastive learning of visual representations. In: PMLR. **International conference on machine learning**. [S.l.], 2020. p. 1597–1607. Citations on pages 38, 39, and 97.

CHEN, W.; HAYS, J. Sketchygan: Towards diverse and realistic sketch to image synthesis. In: **Proc. CVPR**. [S.l.: s.n.], 2018. Citation on page 71.

CHEN, X.; FAN, H.; GIRSHICK, R.; HE, K. Improved baselines with momentum contrastive learning. **arXiv preprint arXiv:2003.04297**, 2020. Citations on pages 39 and 97.

CHOPRA, S.; HADSELL, R.; LECUN, Y. Learning a similarity metric discriminatively, with application to face verification. In: IEEE. **Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on**. [S.l.], 2005. v. 1, p. 539–546. Citation on page 37.

CHOWDHURY, G. G. **Introduction to modern information retrieval**. [S.l.]: Facet publishing, 2010. Citation on page 30.

CLEVERDON, C. The cranfield tests on index language devices. In: MCB UP LTD. **Aslib proceedings**. [S.l.], 1967. Citation on page 30.

COLLOMOSSE, J.; BUI, T.; JIN, H. Livesketch: Query perturbations for guided sketch-based visual search. In: **Proc. CVPR**. [S.l.: s.n.], 2019. p. 1–9. Citations on pages 25, 48, 49, 50, 55, 56, 57, 58, 59, 61, 62, 64, and 65.

COLLOMOSSE, J.; BUI, T.; WILBER, M.; FANG, C.; JIN, H. Sketching with style: Visual search with sketches and aesthetic context. In: **Proc. ICCV**. [S.l.: s.n.], 2017. Citations on pages 48, 49, and 64.

DAI, Z.; YANG, Z.; YANG, Y.; CARBONELL, J. G.; LE, Q. V.; SALAKHUTDINOV, R. Transformer-xl: Attentive language models beyond a fixed-length context. In: KORHONEN, A.; TRAUM, D. R.; MÀRQUEZ, L. (Ed.). **Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers**. Association for Computational Linguistics, 2019. p. 2978–2988. Available: <https://doi.org/10.18653/v1/p19-1285>. Citations on pages 48 and 50.

DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. In: IEEE. **Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on**. [S.l.], 2005. v. 1, p. 886–893. Citations on pages 22 and 23.

DEFFERRARD, M.; BRESSON, X.; VANDERGHEYNST, P. Convolutional neural networks on graphs with fast localized spectral filtering. **Advances in neural information processing systems**, v. 29, 2016. Citation on page 44.

DENG, C.; XU, X.; WANG, H.; YANG, M.; TAO, D. Progressive cross-modal semantic network for zero-shot sketch-based image retrieval. **IEEE Transactions on Image Processing**, IEEE, v. 29, p. 8892–8902, 2020. Citations on pages 25, 99, 102, and 115.

DEVLIN, J.; CHANG, M.-W.; LEE, K.; TOUTANOVA, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In: **Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)**. Minneapolis, Minnesota: Association for Computational Linguistics, 2019. p. 4171–4186. Available: <https://www.aclweb.org/anthology/N19-1423>. Citations on pages 38, 42, 48, 50, 65, and 77.

DEY, S.; RIBA, P.; DUTTA, A.; LLADOS, J.; SONG, Y.-Z. Doodle to search: Practical zero-shot sketch-based image retrieval. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2019. p. 2179–2188. Citations on pages 25, 96, 99, and 106.

DEY, S.; RIBA, P.; DUTTA, A.; LLADOS, J.; SONG, Y. Doodle to search: Practical zero-shot sketch-based image retrieval. In: **Proc. CVPR**. [S.l.: s.n.], 2019. Citation on page 49.

DHARIWAL, P.; NICHOL, A. Diffusion models beat gans on image synthesis. In: RANZATO, M.; BEYGELZIMER, A.; DAUPHIN, Y.; LIANG, P.; VAUGHAN, J. W. (Ed.). **Advances in Neural Information Processing Systems**. [S.l.: s.n.], 2021. v. 34, p. 8780–8794. Citations on pages 34 and 70.

DOSOVITSKIY, A.; BEYER, L.; KOLESNIKOV, A.; WEISSENBORN, D.; ZHAI, X.; UNTERTHINER, T.; DEHGHANI, M.; MINDERER, M.; HEIGOLD, G.; GELLY, S.; USZKOREIT, J.; HOULSBY, N. An image is worth 16x16 words: Transformers for image recognition at scale. In: **International Conference on Learning Representations**. [s.n.], 2021. Available: <https://openreview.net/forum?id=YicbFdNTTy>. Citations on pages 43, 101, and 112.

DOUGLAS, D. H.; PEUCKER, T. K. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. **Cartographica: the international journal for geographic information and geovisualization**, University of Toronto Press, v. 10, n. 2, p. 112–122, 1973. Citations on pages 50 and 59.

DUTTA, A.; AKATA, Z. Semantically tied paired cycle consistency for any-shot sketch-based image retrieval. **Int. J. Comput. Vis.**, v. 128, n. 10, p. 2684–2703, 2020. Available: <https://doi.org/10.1007/s11263-020-01350-x>. Citations on pages 25, 72, 99, 102, 113, 115, and 116.

DUTTA, T.; SINGH, A.; BISWAS, S. Styleguide: Zero-shot sketch-based image retrieval using style-guided image generation. **IEEE Trans. Multim.**, v. 23, p. 2833–2842, 2021. Available: <https://doi.org/10.1109/TMM.2020.3017918>. Citations on pages 25, 32, 36, 72, and 99.

EITZ, M.; HAYS, J.; ALEXA, M. How do humans sketch objects? **Proc. ACM SIGGRAPH**, p. 44:1–44:10, 2012. Citations on pages 49 and 106.

EITZ, M.; HILDEBRAND, K.; BOUBEKEUR, T.; ALEXA, M. A descriptor for large scale image retrieval based on sketched feature lines. In: **Proc. SBIM**. [S.l.: s.n.], 2009. p. 29–36. Citations on pages 22 and 23.

EITZ, M.; HILDEBRAND, K.; BOUBEKEUR, T.; ALEXA, M. Sketch-based image retrieval: Benchmark and bag-of-features descriptors. **IEEE Trans. Visualization and Computer Graphics**, v. 17, n. 11, p. 1624–1636, 2011. Citations on pages 22, 23, and 70.

FAHLMAN, S. E.; HINTON, G. E.; SEJNOWSKI, T. J. Massively parallel architectures for ai: Netl, thistle, and boltzmann machines. In: **Proceedings of the Third AAAI Conference on Artificial Intelligence**. Washington, D.C.: AAAI Press, 1983. (AAAI'83), p. 109–113. Available: <http://dl.acm.org/citation.cfm?id=2886844.2886868>. Citation on page 33.

FELDMAN, J. The structure of perceptual categories. **Journal of mathematical psychology**, Elsevier, v. 41, n. 2, p. 145–170, 1997. Citation on page 46.

FINN, C.; ABBEEL, P.; LEVINE, S. Model-agnostic meta-learning for fast adaptation of deep networks. In: PMLR. **International conference on machine learning**. [S.l.], 2017. p. 1126–1135. Citation on page 46.

FREY, B. J.; HINTON, G. E.; DAYAN, P. Does the wake-sleep algorithm produce good density estimators? In: TOURETZKY, D. S.; MOZER, M. C.; HASSELMO, M. E. (Ed.). **Advances in Neural Information Processing Systems 8**. [S.l.]: MIT Press, 1996. p. 661–667. Citation on page 32.

GAO, C.; LIU, Q.; XU, Q.; WANG, L.; LIU, J.; ZOU, C. Sketchycoco: Image generation from freehand scene sketches. In: **Proc. CVPR**. [S.l.: s.n.], 2020. Citations on pages 25, 71, 82, 86, 87, and 89.

GARAU, N.; BISAGNO, N.; SAMBUGARO, Z.; CONCI, N. Interpretable part-whole hierarchies and conceptual-semantic relationships in neural networks. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2022. p. 13689–13698. Citation on page 46.

GE, C.; SUN, H.; SONG, Y.-Z.; MA, Z.; LIAO, J. Exploring local detail perception for scene sketch semantic segmentation. **IEEE Transactions on Image Processing**, IEEE, v. 31, p. 1447–1461, 2022. Citation on page 83.

GONG, Y.; LAZEBNIK, S.; GORDO, A.; PERRONNIN, F. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, v. 35, n. 12, p. 2916–2929, 2012. Citation on page 107.

GOODFELLOW, I. NIPS 2016 Tutorial: Generative Adversarial Networks. dec 2016. Available: <http://arxiv.org/abs/1701.00160>. Citations on pages 32 and 33.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>. Citations on pages 23, 29, and 31.

GOODFELLOW, I. J.; POUGET-ABADIE, J.; MIRZA, M.; XU, B.; WARDE-FARLEY, D.; OZAIR, S.; COURVILLE, A.; BENGIO, Y. Generative adversarial nets. In: **NeurIPS**. [S.l.: s.n.], 2014. Citations on pages 34, 35, and 79.

GORDO, A.; ALMAZÁN, J.; REVAUD, J.; LARLUS, D. Deep image retrieval: Learning global representations for image search. In: **Proc. ECCV**. [S.l.: s.n.], 2016. p. 241–257. Citation on page 49.

GORI, M.; MONFARDINI, G.; SCARSELLI, F. A new model for learning in graph domains. In: **Proceedings. 2005 IEEE international joint conference on neural networks**. [S.l.: s.n.], 2005. v. 2, n. 2005, p. 729–734. Citation on page 44.

GOU, J.; YU, B.; MAYBANK, S. J.; TAO, D. Knowledge distillation: A survey. **International Journal of Computer Vision**, Springer, v. 129, n. 6, p. 1789–1819, 2021. Citation on page 40.

GRILL, J.-B.; STRUB, F.; ALTCHÉ, F.; TALLEC, C.; RICHEMOND, P.; BUCHATSKAYA, E.; DOERSCH, C.; PIRES, B. A.; GUO, Z.; AZAR, M. G. *et al.* Bootstrap your own latent-a new approach to self-supervised learning. **Advances in neural information processing systems**, v. 33, p. 21271–21284, 2020. Citations on pages 39 and 97.

GU, J.; ZHAO, H.; LIN, Z.; LI, S.; CAI, J.; LING, M. Scene graph generation with external knowledge and image reconstruction. In: **Proceedings of the IEEE/CVF conference on computer vision and pattern recognition**. [S.l.: s.n.], 2019. p. 1969–1978. Citation on page 71.

GUO, L.; LIU, J.; TANG, J.; LI, J.; LUO, W.; LU, H. Aligning linguistic words and visual semantic units for image captioning. In: **Proc. ACM Multimedia**. [S.l.: s.n.], 2019. p. 765–773. Citation on page 76.

HA, D.; ECK, D. A neural representation of sketch drawings. In: **International Conference on Learning Representations**. [s.n.], 2018. Available: <https://openreview.net/forum?id=Hy6GHpkCW>. Citations on pages 24, 48, 50, 51, 57, 58, 59, 61, 62, and 80.

HADSELL, R.; CHOPRA, S.; LECUN, Y. Dimensionality reduction by learning an invariant mapping. In: IEEE. **Computer vision and pattern recognition, 2006 IEEE computer society conference on**. [S.l.], 2006. v. 2, p. 1735–1742. Citation on page 37.

HE, K.; FAN, H.; WU, Y.; XIE, S.; GIRSHICK, R. Momentum contrast for unsupervised visual representation learning. In: **Proceedings of the IEEE/CVF conference on computer vision and pattern recognition**. [S.l.: s.n.], 2020. p. 9729–9738. Citations on pages 39, 97, and 101.

HE, K.; ZHANG, X.; REN, S.; SUN, J. Deep residual learning for image recognition. In: **Proc. CVPR**. [S.l.: s.n.], 2016. p. 770–778. Citations on pages 45, 91, 101, and 112.

HEUSEL, M.; RAMSAUER, H.; UNTERTHINER, T.; NESSLER, B.; HOCHREITER, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. **NeurIPS**, v. 30, 2017. Citations on pages 80 and 87.

HINTON, G. How to represent part-whole hierarchies in a neural network. **arXiv preprint arXiv:2102.12627**, 2021. Citation on page 46.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural computation**, MIT Press, v. 9, n. 8, p. 1735–1780, 1997. Citation on page 24.

HOWARD, A. G.; ZHU, M.; CHEN, B.; KALENICHENKO, D.; WANG, W.; WEYAND, T.; ANDREETTO, M.; ADAM, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. **arXiv preprint arXiv:1704.04861**, 2017. Citations on pages 74 and 91.

HU, R.; BARNARD, M.; COLLOMOSSE, J. P. Gradient field descriptor for sketch based retrieval and localization. In: **Proc. ICIP**. [S.l.: s.n.], 2010. v. 10, p. 1025–1028. Citations on pages 22, 23, and 49.

HU, R.; COLLOMOSSE, J. A performance evaluation of gradient field hog descriptor for sketch based image retrieval. **CVIU**, Elsevier, v. 117, n. 7, p. 790–806, 2013. Citations on pages 22, 23, 49, and 70.

HU, R.; WANG, T.; COLLOMOSSE, J. A bag-of-regions approach to sketch based image retrieval. In: **Proc. ICIP**. [S.l.: s.n.], 2011. Citation on page 49.

HUANG, Z.; XU, W.; YU, K. Bidirectional lstm-crf models for sequence tagging. **arXiv preprint arXiv:1508.01991**, 2015. Citation on page 24.

Isola, P.; Zhu, J.; Zhou, T.; Efros, A. A. Image-to-image translation with conditional adversarial networks. In: **Proc. CVPR**. [S.l.: s.n.], 2017. p. 5967–5976. Citations on pages 36 and 70.

ISOLA, P.; ZHU, J.-Y.; ZHOU, T.; EFROS, A. A. Image-to-image translation with conditional adversarial networks. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2017. p. 1125–1134. Citation on page 79.

JAEGLE, A.; BORGEAUD, S.; ALAYRAC, J.-B.; DOERSCH, C.; IONESCU, C.; DING, D.; KOPPULA, S.; ZORAN, D.; BROCK, A.; SHELHAMER, E. *et al.* Perceiver io: A general architecture for structured inputs & outputs. **arXiv preprint arXiv:2107.14795**, 2021. Citation on page 112.

JAEGLE, A.; GIMENO, F.; BROCK, A.; VINYALS, O.; ZISSERMAN, A.; CARREIRA, J. Perceiver: General perception with iterative attention. In: PMLR. **International conference on machine learning**. [S.l.], 2021. p. 4651–4664. Citation on page 112.

JAMES, S.; FONSECA, M. J.; COLLOMOSSE, J. Reenact: Sketch based choreographic design from archival dance footage. In: ACM. **Proceedings of International Conference on Multimedia Retrieval**. [S.l.], 2014. p. 313. Citations on pages 22 and 23.

JAMES, S.; HU, R.; WANG, T.; COLLOMOSSE, J. Markov random fields for sketch based video retrieval. In: **Proc. ICMR**. [S.l.: s.n.], 2013. Citation on page 49.

JING, T.; XIA, H.; HAMM, J.; DING, Z. Augmented multi-modality fusion for generalized zero-shot sketch-based visual retrieval. **IEEE Transactions on Image Processing**, IEEE, 2022. Citation on page 37.

JOHNSON, J.; GUPTA, A.; FEI-FEI, L. Image Generation from Scene Graphs. In: **Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2018. p. 1219–1228. Citations on pages 45, 71, 75, 81, and 92.

JOHNSON, J.; KRISHNA, R.; STARK, M.; LI, L. J.; SHAMMA, D. A.; BERNSTEIN, M. S.; LI, F. F. Image retrieval using scene graphs. In: **Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition**. [S.l.]: IEEE, 2015. p. 3668–3678. ISBN 9781467369640. ISSN 10636919. Citations on pages 70 and 71.

KINGMA, D.; BA, J. Adam: A method for stochastic optimization. In: **Proc. ICLR**. [S.l.: s.n.], 2015. Citation on page 80.

Kingma, D. P.; Welling, M. Auto-Encoding Variational Bayes. **ArXiv e-prints**, dec 2013. Citation on page 32.

KIPF, T. N.; WELLING, M. Semi-supervised classification with graph convolutional networks. **arXiv preprint arXiv:1609.02907**, 2016. Citations on pages 44, 45, and 105.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. Imagenet classification with deep CNNs. In: **Proc. NIPS**. [S.l.: s.n.], 2012. Citation on page 38.

LAI, C.-I. Contrastive predictive coding based feature for automatic speaker verification. **arXiv preprint arXiv:1904.01575**, 2019. Citation on page 101.

LAKE, B. M.; ULLMAN, T. D.; TENENBAUM, J. B.; GERSHMAN, S. J. Building machines that learn and think like people. **Behavioral and brain sciences**, Cambridge University Press, v. 40, 2017. Citation on page 46.

LAZEBNIK, S.; SCHMID, C.; PONCE, J. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: IEEE. **Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on**. [S.l.], 2006. v. 2, p. 2169–2178. Citation on page 23.

LI, L.; ZOU, C.; ZHENG, Y.; SU, Q.; FU, H.; TAI, C.-L. Sketch-r2cnn: An attentive network for vector sketch recognition. **arXiv preprint arXiv:1811.08170**, 2018. Citations on pages 25 and 50.

Li, X.; Wu, X. Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition. In: **2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**. [S.l.: s.n.], 2015. p. 4520–4524. Citations on pages 40 and 52.

LIN, H.; FU, Y.; LU, P.; GONG, S.; XUE, X.; JIANG, Y.-G. Tc-net for isbir: Triplet classification network for instance-level sketch based image retrieval. In: **Proceedings of the 27th ACM international conference on multimedia**. [S.l.: s.n.], 2019. p. 1676–1684. Citation on page 29.

LING, Z.; XING, Z.; LI, J.; NIU, L. Multi-level region matching for fine-grained sketch-based image retrieval. In: **Proceedings of the 30th ACM International Conference on Multimedia**. [S.l.: s.n.], 2022. p. 462–470. Citation on page 37.

LIU, F.; ZOU, C.; DENG, X.; ZUO, R.; LAI, Y.-K.; MA, C.; LIU, Y.-J.; WANG, H. Scenesketcher: Fine-grained image retrieval with scene sketches. In: SPRINGER. **ECCV**. [S.l.], 2020. p. 718–734. Citations on pages 25, 45, 68, 70, 81, 82, 84, 85, 86, and 87.

LIU, Q.; XIE, L.; WANG, H.; YUILLE, A. L. Semantic-aware knowledge preservation for zero-shot sketch-based image retrieval. In: **Proceedings of the IEEE/CVF International Conference on Computer Vision**. [S.l.: s.n.], 2019. p. 3662–3671. Citations on pages 25, 46, 99, 106, and 115.

LU, Y.; WU, S.; TAI, Y.-W.; TANG, C.-K. Image generation from sketch constraint using contextual gan. In: **Proc. ECCV**. [S.l.: s.n.], 2018. Citations on pages 36 and 71.

LUONG, M.-T.; PHAM, H.; MANNING, C. D. Effective approaches to attention-based neural machine translation. **arXiv preprint arXiv:1508.04025**, 2015. Citation on page 40.

MAATEN, L. van der; HINTON, G. Visualizing data using t-sne. **Journal of Machine Learning Research**, v. 9, n. 86, p. 2579–2605, 2008. Available: <http://jmlr.org/papers/v9/vandermaaten08a.html>. Citations on pages 56 and 98.

MAO, X.; LI, Q.; XIE, H.; LAU, R. Y. K.; WANG, Z. **Least Squares Generative Adversarial Networks**. 2016. Cite arxiv:1611.04076. Available: <http://arxiv.org/abs/1611.04076>. Citation on page 79.

MIECH, A.; LAPTEV, I.; SIVIC, J. Learnable pooling with context gating for video classification. **arXiv preprint arXiv:1706.06905**, 2017. Citation on page 101.

MIKOLOV, T.; SUTSKEVER, I.; CHEN, K.; CORRADO, G. S.; DEAN, J. Distributed representations of words and phrases and their compositionality. **Advances in neural information processing systems**, v. 26, 2013. Citations on pages 39, 99, and 102.

MIRZA, M.; OSINDERO, S. Conditional generative adversarial nets. **arXiv preprint arXiv:1411.1784**, 2014. Citations on pages 36 and 70.

MUHAMMAD, U. R.; YANG, Y.; SONG, Y.-Z.; XIANG, T.; HOSPEDALES, T. M. Learning deep sketch abstraction. In: **Proc. CVPR**. [S.l.: s.n.], 2018. p. 8014–8023. Citation on page 50.

NOWAK, E.; JURIE, F.; TRIGGS, B. Sampling strategies for bag-of-features image classification. In: **Computer Vision–ECCV 2006**. [S.l.]: Springer, 2006. p. 490–503. Citation on page 23.

OORD, A. v. d.; LI, Y.; VINYALS, O. Representation learning with contrastive predictive coding. **arXiv preprint arXiv:1807.03748**, 2018. Citations on pages 38 and 39.

OORD, A. van den; DIELEMAN, S.; ZEN, H.; SIMONYAN, K.; VINYALS, O.; GRAVES, A.; KALCHBRENNER, N.; SENIOR, A.; KAVUKCUOGLU, K. WaveNet: A Generative Model for Raw Audio. sep 2016. Available: <http://arxiv.org/abs/1609.03499>. Citation on page 33.

PALOWITCH, J.; PEROZZI, B. Monet: Debiasing graph embeddings via the metadata-orthogonal training unit. **arXiv preprint arXiv:1909.11793**, 2019. Citation on page 45.

PANDEY, A.; MISHRA, A.; VERMA, V. K.; MITTAL, A.; MURTHY, H. A. Stacked adversarial network for zero-shot sketch based image retrieval. In: **IEEE Winter Conference on Applications of Computer Vision, WACV 2020, Snowmass Village, CO, USA, March 1-5, 2020**. [S.l.]: IEEE, 2020. p. 2529–2538. Citations on pages 25, 72, and 99.

PANG, K.; LI, K.; YANG, Y.; ZHANG, H.; HOSPEDALES, T.; XIANG, T.; SONG, Y. Generalising fine-grained sketch-based image retrieval. In: **Proc. CVPR**. [S.l.: s.n.], 2019. Citation on page 49.

PANG, K.; SONG, Y.-Z.; XIANG, T.; HOSPEDALES, T. M. Cross-domain generative learning for fine-grained sketch-based image retrieval. In: **BMVC**. [S.l.: s.n.], 2017. p. 1–12. Citation on page 71.

PARK, T.; LIU, M.-Y.; WANG, T.-C.; ZHU, J.-Y. Gaugan: semantic image synthesis with spatially adaptive normalization. In: ACM. **ACM SIGGRAPH 2019 Real-Time Live!** [S.l.], 2019. p. 2. Citation on page 65.

PARK, T.; LIU, M.-Y.; WANG, T.-C.; ZHU, J.-Y. Semantic image synthesis with spatially-adaptive normalization. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2019. Citations on pages 36, 70, 79, 88, and 89.

PARMAR, N.; VASWANI, A.; USZKOREIT, J.; KAISER, L.; SHAZEER, N.; KU, A.; TRAN, D. Image transformer. In: **ICML**. [S.l.]: PMLR, 2018. (Proceedings of Machine Learning Research, v. 80), p. 4052–4061. Citation on page 50.

PENG, C.; GAO, X.; WANG, N.; TAO, D.; LI, X.; LI, J. Multiple representations-based face sketch–photo synthesis. **IEEE transactions on neural networks and learning systems**, IEEE, v. 27, n. 11, p. 2201–2215, 2015. Citation on page 71.

PONTI, M.; RIBEIRO, L. S.; NAZARE, T. S.; BUI, T.; COLLOMOSSE, J. Everything you wanted to know about deep learning for computer vision but were afraid to ask. In: **SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T 2017)**. [S.l.: s.n.], 2017. p. 1–25. Citation on page 29.

PONTI, M. A.; SANTOS, F. P. dos; RIBEIRO, L. S.; CAVALLARI, G. B. Training deep networks from zero to hero: avoiding pitfalls and going beyond. In: IEEE. **2021 34th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)**. [S.l.], 2021. p. 9–16. Citations on pages 29 and 140.

QI, A.; GRYADITSKAYA, Y.; SONG, J.; YANG, Y.; QI, Y.; HOSPEDALES, T. M.; XIANG, T.; SONG, Y.-Z. Toward fine-grained sketch-based 3d shape retrieval. **IEEE transactions on image processing**, IEEE, v. 30, p. 8595–8606, 2021. Citation on page 119.

QI, M.; LI, W.; YANG, Z.; WANG, Y.; LUO, J. Attentive relational networks for mapping images to scene graphs. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2019. p. 3957–3966. Citation on page 71.

QI, Y.; SONG, Y.-Z.; XIANG, T.; ZHANG, H.; HOSPEDALES, T.; LI, Y.; GUO, J. Making better use of edges via perceptual grouping. In: **Proc. CVPR**. [S.l.: s.n.], 2015. Citations on pages 22 and 23.

QI, Y.; SONG, Y.-Z.; ZHANG, H.; LIU, J. Sketch-based image retrieval via siamese convolutional neural network. In: IEEE. **Proc. ICIP**. [S.l.], 2016. p. 2460–2464. Citations on pages 23, 37, 48, 49, 68, and 70.

QI, Y.; SU, G.; CHOWDHURY, P. N.; LI, M.; SONG, Y.-Z. Sketchlattice: Latticed representation for sketch manipulation. In: **Proceedings of the IEEE/CVF International Conference on Computer Vision**. [S.l.: s.n.], 2021. p. 953–961. Citation on page 117.

QI, Y.; SU, G.; WANG, Q.; YANG, J.; PANG, K.; SONG, Y.-Z. Generative sketch healing. **International Journal of Computer Vision**, Springer, p. 1–16, 2022. Citation on page 117.

RADENOVIĆ, F.; TOLIAS, G.; CHUM, O. Cnn image retrieval learns from bow: Unsupervised fine-tuning with hard examples. **arXiv preprint arXiv:1604.02426**, 2016. Citation on page 49.

RADFORD, A.; KIM, J. W.; HALLACY, C.; RAMESH, A.; GOH, G.; AGARWAL, S.; SASTRY, G.; ASKELL, A.; MISHKIN, P.; CLARK, J.; KRUEGER, G.; SUTSKEVER, I. Learning transferable visual models from natural language supervision. In: MEILA, M.; ZHANG, T. (Ed.). **Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event**. [S.l.: s.n.], 2021. (Proceedings of Machine Learning Research, v. 139), p. 8748–8763. Citations on pages 34, 43, and 70.

RADFORD, A.; METZ, L.; CHINTALA, S. Unsupervised representation learning with deep convolutional generative adversarial networks. **CoRR**, abs/1511.06434, 2015. Available: <http://arxiv.org/abs/1511.06434>. Citation on page 36.

RADFORD, A.; NARASIMHAN, K.; SALIMANS, T.; SUTSKEVER, I. *et al.* Improving language understanding by generative pre-training. OpenAI, 2018. Citations on pages 38 and 42.

RAMESH, A.; PAVLOV, M.; GOH, G.; GRAY, S.; VOSS, C.; RADFORD, A.; CHEN, M.; SUTSKEVER, I. Zero-shot text-to-image generation. In: MEILA, M.; ZHANG, T. (Ed.). **Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event**. [S.l.: s.n.], 2021. (Proceedings of Machine Learning Research, v. 139), p. 8821–8831. Citations on pages 34, 45, and 70.

REESE, H. **How to survive the A.I. Revolution**. Stanford Graduate School of Business, 2022. Available: <https://www.gsb.stanford.edu/insights/how-survive-artificial-intelligence-revolution>. Citation on page 21.

RESENDE, D. C. O. de; PONTI, M. A. Robust image features for classification and zero-shot tasks by merging visual and semantic attributes. **Neural Computing and Applications**, Springer, v. 34, n. 6, p. 4459–4471, 2022. Citation on page 46.

REZATOFIGHI, H.; TSOI, N.; GWAK, J.; SADEGHIAN, A.; REID, I.; SAVARESE, S. Generalized intersection over union: A metric and a loss for bounding box regression. In: **Proc. CVPR**. [S.l.: s.n.], 2019. Citation on page 78.

RIBEIRO, L. S. F.; BUI, T.; COLLOMOSSE, J.; PONTI, M. Sketchformer: Transformer-based representation for sketched structure. In: **Proc. CVPR**. [S.l.: s.n.], 2020. Citations on pages 47, 65, 71, 140, and 141.

RIBEIRO, L. S. F.; BUI, T.; COLLOMOSSE, J.; PONTI, M. Scene designer: a unified model for scene search and synthesis from sketch. In: **Proc. ICCV WS**. [S.l.: s.n.], 2021. p. 2424–2433. Citations on pages 67, 93, 140, and 152.

RIBEIRO, L. S. F.; BUI, T.; COLLOMOSSE, J.; PONTI, M. Scene designer: Compositional sketch-based image retrieval with contrastive learning and an auxiliary synthesis task. **Multimedia Tools and Applications**, Springer, 2022. Citations on pages 67, 93, 140, and 163.

RIBEIRO, L. S. F.; PONTI, M. Sketch-an-anchor: Sub-epoch fast model adaptation for zero-shot sketch-based image retrieval. In: **Submission Under Review**. [S.l.: s.n.], 2022. Citations on pages 95, 140, and 187.

ROMBACH, R.; BLATTMANN, A.; LORENZ, D.; ESSER, P.; OMMER, B. High-resolution image synthesis with latent diffusion models. **CoRR**, abs/2112.10752, 2021. Available: <https://arxiv.org/abs/2112.10752>. Citations on pages 21, 34, 45, and 70.

RUSSAKOVSKY, O.; DENG, J.; SU, H.; KRAUSE, J.; SATHEESH, S.; MA, S.; HUANG, Z.; KARPATHY, A.; KHOSLA, A.; BERNSTEIN, M.; BERG, A. C.; FEI-FEI, L. ImageNet Large Scale Visual Recognition Challenge. **IJCV**, v. 115, n. 3, p. 211–252, 2015. Citations on pages 38 and 74.

SAAVEDRA, J. M.; BARRIOS, J. M. Sketch based image retrieval using learned keyshapes. In: **Proc. BMVC**. [S.l.: s.n.], 2015. Citation on page 23.

SAAVEDRA, J. M.; BUSTOS, B. Sketch-based image retrieval using keyshapes. **Multimedia Tools and Applications**, v. 73, n. 3, p. 2033–2062, 2014. Citation on page 23.

SAAVEDRA, J. M.; MORALES, J.; MURRUGARRA-LLERENA, N. Sbir-byol: a self-supervised sketch-based image retrieval model. **Neural Computing and Applications**, Springer, p. 1–14, 2022. Citation on page 39.

SAIN, A.; BHUNIA, A. K.; POTLAPALLI, V.; CHOWDHURY, P. N.; XIANG, T.; SONG, Y.-Z. Sketch3t: Test-time training for zero-shot sbir. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2022. p. 7462–7471. Citation on page 37.

SAIN, A.; BHUNIA, A. K.; YANG, Y.; XIANG, T.; SONG, Y.-Z. Stylemeup: Towards style-agnostic sketch-based image retrieval. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2021. p. 8504–8513. Citation on page 32.

SANDRYHAILA, A.; MOURA, J. M. Discrete signal processing on graphs. **IEEE transactions on signal processing**, IEEE, v. 61, n. 7, p. 1644–1656, 2013. Citation on page 44.

SANGKLOY, P.; BURNELL, N.; HAM, C.; HAYS, J. The sketchy database: learning to retrieve badly drawn bunnies. **ACM Trans. Graphics (TOG)**, ACM, v. 35, n. 4, p. 119, 2016. Citations on pages 24, 29, 37, 45, 68, 74, 81, 82, 84, 86, 87, and 106.

SANGKLOY, P.; BURNELL, N.; HAM, C.; HAYS, J. The sketchy database: Learning to retrieve badly drawn bunnies. In: **Proc. ACM SIGGRAPH**. [S.l.: s.n.], 2016. Citations on pages 48 and 49.

SANTOS, F. P. dos; RIBEIRO, L. S.; PONTI, M. A. Generalization of feature embeddings transferred from different video anomaly detection domains. **Journal of Visual Communication and Image Representation**, Elsevier, v. 60, p. 407–416, 2019. Citation on page 140.

SATORRAS, V. G.; ESTRACH, J. B. Few-shot learning with graph neural networks. In: **ICLR (Poster)**. [S.l.]: OpenReview.net, 2018. Citation on page 45.

SCARSELLI, F.; GORI, M.; TSOI, A. C.; HAGENBUCHNER, M.; MONFARDINI, G. The graph neural network model. **IEEE transactions on neural networks**, IEEE, v. 20, n. 1, p. 61–80, 2008. Citation on page 44.

SCHNEIDER, R. G.; TUYTELAARS, T. Sketch classification and classification-driven analysis using fisher vectors. **ACM Transactions on Graphics (TOG)**, ACM, v. 33, n. 6, p. 174, 2014. Citation on page 49.

SCHÖNFELD, E.; SUSHKO, V.; ZHANG, D.; GALL, J.; SCHIELE, B.; KHOREVA, A. You only need adversarial supervision for semantic image synthesis. In: **9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021**. [S.l.: s.n.], 2021. Citation on page 70.

SCHROFF, F.; KALENICHENKO, D.; PHILBIN, J. Facenet: A unified embedding for face recognition and clustering. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2015. p. 815–823. Citations on pages 24 and 37.

SCHUSTER, M.; PALIWAL, K. K. Bidirectional recurrent neural networks. **IEEE transactions on Signal Processing**, Ieee, v. 45, n. 11, p. 2673–2681, 1997. Citations on pages 24 and 40.

SEDDATI, O.; DUPONT, S.; MAHMOUDI, S. Deepsketch 2: Deep convolutional neural networks for partial sketch recognition. In: IEEE. **2016 14th International Workshop on Content-Based Multimedia Indexing (CBMI)**. [S.l.], 2016. p. 1–6. Citation on page 49.

SEDDATI, O.; DUPONT, S.; MAHMOUDI, S.; DUTOIT, T. Transformers and cnns both beat humans on sbir. **arXiv preprint arXiv:2209.06629**, 2022. Citation on page 43.

SEDDATI, O.; DUPONT, S.; MAHOUDI, S. Quadruplet networks for sketch-based image retrieval. In: **Proc. ICMR**. [S.l.: s.n.], 2017. Citations on pages 24, 29, and 49.

SHARIR, O.; PELEG, B.; SHOHAM, Y. The cost of training nlp models: A concise overview. **arXiv preprint arXiv:2004.08900**, 2020. Citation on page 45.

SHAW, P.; USZKOREIT, J.; VASWANI, A. Self-attention with relative position representations. **arXiv preprint arXiv:1803.02155**, 2018. Citation on page 92.

SHEN, Y.; LIU, L.; SHEN, F.; SHAO, L. Zero-shot sketch-image hashing. In: **2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018**. [S.l.]: Computer Vision Foundation / IEEE Computer Society, 2018. p. 3598–3607. Citations on pages 25, 36, 45, 46, 49, 72, 96, 99, and 106.

SHUMAN, D. I.; NARANG, S. K.; FROSSARD, P.; ORTEGA, A.; VANDERGHEYNST, P. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. **IEEE signal processing magazine**, IEEE, v. 30, n. 3, p. 83–98, 2013. Citation on page 44.

SHVARTSMAN, D. **Facebook: The leading social platform of our times**. 2022. Available: <https://www.investing.com/academy/statistics/facebook-meta-facts/>. Citation on page 21.

SHVETSOVA, N.; CHEN, B.; ROUDITCHENKO, A.; THOMAS, S.; KINGSBURY, B.; FERIS, R. S.; HARWATH, D.; GLASS, J.; KUEHNE, H. Everything at once-multi-modal fusion transformer for video retrieval. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2022. p. 20020–20029. Citations on pages 112 and 119.

SIMONYAN, K.; ZISSERMAN, A. Two-stream convolutional networks for action recognition in videos. **Advances in neural information processing systems**, v. 27, 2014. Citation on page 36.

SOHL-DICKSTEIN, J.; WEISS, E.; MAHESWARANATHAN, N.; GANGULI, S. Deep unsupervised learning using nonequilibrium thermodynamics. In: BACH, F.; BLEI, D. (Ed.). **Proceedings of the 32nd International Conference on Machine Learning**. Lille, France: PMLR, 2015. (Proceedings of Machine Learning Research, v. 37), p. 2256–2265. Citations on pages 34 and 70.

SONG, J.; PANG, K.; SONG, Y.; XIANG, T.; HOSPEDALES, T. M. Learning to sketch with shortcut cycle consistency. In: **CVPR**. [S.l.]: Computer Vision Foundation / IEEE Computer Society, 2018. p. 801–810. Citation on page 49.

SONG, J.; SONG, Y.-Z.; XIANG, T.; HOSPEDALES, T. M.; RUAN, X. Deep multi-task attribute-driven ranking for fine-grained sketch-based image retrieval. In: **BMVC**. [S.l.: s.n.], 2016. v. 1, p. 3. Citation on page 37.

SONG, J.; YU, Q.; SONG, Y.-Z.; XIANG, T.; HOSPEDALES, T. M. Deep spatial-semantic attention for fine-grained sketch-based image retrieval. In: **Proceedings of the IEEE international conference on computer vision**. [S.l.: s.n.], 2017. p. 5551–5560. Citation on page 37.

SUKHBAATAR, S.; SZLAM, A.; WESTON, J.; FERGUS, R. End-to-end memory networks. In: CORTES, C.; LAWRENCE, N. D.; LEE, D. D.; SUGIYAMA, M.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada**. [S.l.: s.n.], 2015. p. 2440–2448. Citations on pages 40, 55, 76, and 103.

SUKHBAATAR, S.; WESTON, J.; FERGUS, R. *et al.* End-to-end memory networks. **NeurIPS**, v. 28, 2015. Citation on page 72.

SZEGEDY, C.; IOFFE, S.; VANHOUCKE, V.; ALEMI, A. A. Inception-v4, inception-resnet and the impact of residual connections on learning. In: **AAAI**. [S.l.: s.n.], 2017. p. 4278–4284. Citation on page 91.

TAN, M.; LE, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In: PMLR. **International conference on machine learning**. [S.l.], 2019. p. 6105–6114. Citation on page 112.

THE Quick, Draw! Dataset. 2018. <https://github.com/googlecreativelab/quickdraw-dataset>. Accessed: 2018-10-11. Citations on pages 48, 50, 58, 59, 82, 83, and 106.

THRUN, S. Is learning the n-th thing any easier than learning the first? In: TOURETZKY, D.; MOZER, M.; HASSELMO, M. (Ed.). **Advances in Neural Information Processing Systems**. MIT Press, 1995. v. 8. Available: <https://proceedings.neurips.cc/paper/1995/file/bdb106a0560c4e46ccc488ef010af787-Paper.pdf>. Citation on page 45.

TIAN, J.; XU, X.; SHEN, F.; YANG, Y.; SHEN, H. T. Tvt: Three-way vision transformer through multi-modal hypersphere learning for zero-shot sketch-based image retrieval. **Proceedings of the AAAI Conference on Artificial Intelligence**, v. 36, n. 2, p. 2370–2378, 2022. Citations on pages 25, 39, 40, 43, 98, 99, 100, 107, 112, and 115.

TOUVRON, H.; CORD, M.; DOUZE, M.; MASSA, F.; SABLAYROLLES, A.; JÉGOU, H. Training data-efficient image transformers & distillation through attention. In: PMLR. **International Conference on Machine Learning**. [S.l.], 2021. p. 10347–10357. Citations on pages 43 and 112.

TURSUN, O.; DENMAN, S.; SRIDHARAN, S.; GOAN, E.; FOOKES, C. An efficient framework for zero-shot sketch-based image retrieval. **Pattern Recognition**, Elsevier, v. 126, p. 108528, 2022. Citations on pages 25, 99, 100, 115, and 116.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. **NeurIPS**, v. 30, 2017. Citations on pages 25, 40, 41, 42, 48, 50, 52, 53, 54, 55, 65, and 76.

VELICKOVIC, P.; CUCURULL, G.; CASANOVA, A.; ROMERO, A.; LIÒ, P.; BENGIO, Y. Graph attention networks. In: **ICLR (Poster)**. [S.l.]: OpenReview.net, 2018. Citation on page 45.

VOYNOV, A.; ABERMAN, K.; COHEN-OR, D. Sketch-guided text-to-image diffusion models. **CoRR**, abs/2211.13752, 2022. Citation on page 71.

WANG, F.; KANG, L.; LI, Y. Sketch-based 3d shape retrieval using convolutional neural networks. In: IEEE. **Proc. CVPR**. [S.l.], 2015. p. 1875–1883. Citations on pages 23, 68, and 70.

WANG, J.; SONG, Y.; LEUNG, T.; ROSENBERG, C.; WANG, J.; PHILBIN, J.; CHEN, B.; WU, Y. Learning fine-grained image similarity with deep ranking. In: **Proc. CVPR**. [S.l.: s.n.], 2014. p. 1386–1393. Citation on page 49.

WANG, K.; WANG, Y.; XU, X.; LIU, X.; OU, W.; LU, H. Prototype-based selective knowledge distillation for zero-shot sketch based image retrieval. In: **Proceedings of the 30th ACM International Conference on Multimedia**. [S.l.: s.n.], 2022. p. 601–609. Citations on pages 25, 39, 40, 43, 98, 99, 100, 107, and 115.

WANG, N.; TAO, D.; GAO, X.; LI, X.; LI, J. Transductive face sketch-photo synthesis. **IEEE transactions on neural networks and learning systems**, IEEE, v. 24, n. 9, p. 1364–1376, 2013. Citation on page 71.

WANG, T.; ISOLA, P. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In: PMLR. **International Conference on Machine Learning**. [S.l.], 2020. p. 9929–9939. Citation on page 102.

WANG, T.-C.; LIU, M.-Y.; ZHU, J.-Y.; TAO, A.; KAUTZ, J.; CATANZARO, B. High-resolution image synthesis and semantic manipulation with conditional gans. In: **Proc. CVPR**. [S.l.: s.n.], 2018. Citations on pages 36 and 70.

WANG, W.; BAO, J.; ZHOU, W.; CHEN, D.; CHEN, D.; YUAN, L.; LI, H. Semantic image synthesis via diffusion models. **CoRR**, abs/2207.00050, 2022. Citation on page 70.

WANG, X.; CHEN, X.; ZHA, Z. Sketchpointnet: A compact network for robust sketch recognition. In: IEEE. **2018 25th IEEE International Conference on Image Processing (ICIP)**. [S.l.], 2018. p. 2994–2998. Citation on page 49.

WANG, Y.; YAO, Q.; KWOK, J. T.; NI, L. M. Generalizing from a few examples: A survey on few-shot learning. **ACM computing surveys (csur)**, ACM New York, NY, USA, v. 53, n. 3, p. 1–34, 2020. Citation on page 46.

WANG, Z.; WANG, H.; YAN, J.; WU, A.; DENG, C. Domain-smoothing network for zero-shot sketch-based image retrieval. In: ZHOU, Z.-H. (Ed.). **Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21**. International Joint Conferences on Artificial Intelligence Organization, 2021. p. 1143–1149. Main Track. Available: <https://doi.org/10.24963/ijcai.2021/158>. Citations on pages 25, 39, 99, and 115.

WILBER, M.; FANG, C.; JIN, H.; HERTZMANN, A.; COLLOMOSSE, J.; BELONGIE, S. Bam! the behance artistic media dataset for recognition beyond photography. In: **Proc. ICCV**. [S.l.: s.n.], 2017. Citation on page 49.

WU, F.; SOUZA, A.; ZHANG, T.; FIFTY, C.; YU, T.; WEINBERGER, K. Simplifying graph convolutional networks. In: PMLR. **International conference on machine learning**. [S.l.], 2019. p. 6861–6871. Citation on page 44.

WU, Z.; PAN, S.; CHEN, F.; LONG, G.; ZHANG, C.; PHILIP, S. Y. A comprehensive survey on graph neural networks. **IEEE transactions on neural networks and learning systems**, IEEE, v. 32, n. 1, p. 4–24, 2020. Citation on page 44.

WU, Z.; XIONG, Y.; YU, S. X.; LIN, D. Unsupervised feature learning via non-parametric instance discrimination. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2018. p. 3733–3742. Citation on page 38.

XIAN, Y.; LAMPERT, C. H.; SCHIELE, B.; AKATA, Z. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, v. 41, n. 9, p. 2251–2265, 2018. Citations on pages 25, 96, and 99.

XU, P.; HUANG, Y.; YUAN, T.; PANG, K.; SONG, Y.-Z.; XIANG, T.; HOSPEDALES, T. Sketchmate: Deep hashing for million-scale human sketch retrieval. In: **Proc. CVPR**. [S.l.: s.n.], 2018. Citations on pages 48, 50, and 58.

XU, P.; JOSHI, C. K.; BRESSON, X. Multigraph transformer for free-hand sketch recognition. **IEEE Transactions on Neural Networks and Learning Systems**, IEEE, 2021. Citation on page 71.

XU, Y.; HU, J.; ZENG, K.; GONG, Y. Sketch-based shape retrieval via multi-view attention and generalized similarity. In: IEEE. **2018 7th International Conference on Digital Home (ICDH)**. [S.l.], 2018. p. 311–317. Citations on pages 25 and 49.

YEH, C.-H.; HONG, C.-Y.; HSU, Y.-C.; LIU, T.-L.; CHEN, Y.; LECUN, Y. Decoupled contrastive learning. In: SPRINGER. **European Conference on Computer Vision**. [S.l.], 2022. p. 668–684. Citation on page 39.

YELAMARTHI, S. K.; REDDY, S. K.; MISHRA, A.; MITTAL, A. A zero-shot framework for sketch based image retrieval. In: **Proceedings of the European Conference on Computer Vision (ECCV)**. [S.l.: s.n.], 2018. p. 300–317. Citations on pages 25, 32, 36, 99, 106, 108, and 109.

YELAMARTHI, S. K.; REDDY, S. K.; MISHRA, A.; MITTAL, A. A zero-shot framework for sketch based image retrieval. In: **Proceedings of the European Conference on Computer Vision (ECCV)**. [S.l.: s.n.], 2018. Citation on page 72.

YOON, J.; KIM, T.; DIA, O.; KIM, S.; BENGIO, Y.; AHN, S. Bayesian model-agnostic meta-learning. **Advances in neural information processing systems**, v. 31, 2018. Citation on page 46.

YU, Q.; LIU, F.; SONG, Y.-Z.; XIANG, T.; HOSPEDALES, T. M.; LOY, C.-C. Sketch me that shoe. In: **Proc. CVPR**. [S.l.: s.n.], 2016. p. 799–807. Citations on pages 24, 37, 49, 74, 81, 84, 86, and 87.

YU, Q.; YANG, Y.; LIU, F.; SONG, Y.; XIANG, T.; HOSPEDALES, T. M. Sketch-a-net: A deep neural network that beats humans. **Int. J. Comput. Vis.**, v. 122, n. 3, p. 411–425, 2017. Available: <https://doi.org/10.1007/s11263-016-0932-3>. Citation on page 49.

YU, Q.; YANG, Y.; SONG, Y.-Z.; XIANG, T.; HOSPEDALES, T. M. Sketch-a-net that beats humans. In: **Proc. BMVC**. [S.l.]: IEEE, 2015. Citation on page 23.

YU, Z.; KOVASHKA, A. Syntharch: interactive image search with attribute-conditioned synthesis. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops**. [S.l.: s.n.], 2020. p. 170–171. Citation on page 72.

ZAIDI, S. S. A.; ANSARI, M. S.; ASLAM, A.; KANWAL, N.; ASGHAR, M.; LEE, B. A survey of modern deep learning based object detection models. **Digital Signal Processing**, Elsevier, p. 103514, 2022. Citation on page 71.

ZANG, S.; TU, S.; XU, L. Linking sketch patches by learning synonymous proximity for graphic sketch representation. **arXiv preprint arXiv:2211.16841**, 2022. Citation on page 117.

ZHANG, H.; LIU, S.; ZHANG, C.; REN, W.; WANG, R.; CAO, X. Sketchnet: Sketch classification with web images. In: **Proc. CVPR**. [S.l.: s.n.], 2016. p. 1105–1113. Citation on page 49.

ZHANG, Z.; ZHANG, Y.; FENG, R.; ZHANG, T.; FAN, W. Zero-shot sketch-based image retrieval via graph convolution network. In: **Proceedings of the AAAI Conference on Artificial Intelligence**. [S.l.: s.n.], 2020. v. 34, n. 07, p. 12943–12950. Citations on pages 25, 45, 99, 102, and 115.

ZHOU, K.; LIU, Z.; QIAO, Y.; XIANG, T.; LOY, C. C. Domain generalization: A survey. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, IEEE, 2022. Citation on page 118.

ZHU, G.; ZHANG, L.; JIANG, Y.; DANG, Y.; HOU, H.; SHEN, P.; FENG, M.; ZHAO, X.; MIAO, Q.; SHAH, S. A. A. *et al.* Scene graph generation: A comprehensive survey. **arXiv preprint arXiv:2201.00443**, 2022. Citation on page 71.

ZHU, J.-Y.; PARK, T.; ISOLA, P.; EFROS, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. **arXiv preprint arXiv:1703.10593**, 2017. Citations on pages 36 and 49.

APPENDIX

A

PUBLICATIONS

Within the Ph.D. program, 7 papers were co-authored by the candidate, 6 of those were published and one is currently under review. The publications that are relevant to the thesis were described in Chapters 3, 4 and 5; copies of these original publications are annexed to this chapter and the complete set of publications, including the ones not described in this document, is listed in chronological order in Table 21.

Publication	Presented in the Thesis
CAVALLARI, G. B.; RIBEIRO, L. S.; PONTI, M. A. Unsupervised representation learning using convolutional and stacked auto-encoders: a domain and cross-domain feature space analysis. In: IEEE. **2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)**. [S.l.], 2018. p. 440–446.	No
BUI, T.; RIBEIRO, L.; PONTI, M. A.; COLLOMOSSE, J. Deep Manifold Alignment for Mid-grain Sketch based Image Retrieval. 2018. Available: <http://personal.ee.surrey.ac.uk/Personal/J.Collomosse/pubs/Bui-ACCV-2018.pdf>.	No
SANTOS, F. P. dos; RIBEIRO, L. S.; PONTI, M. A. Generalization of feature embeddings transferred from different video anomaly detection domains. **Journal of Visual Communication and Image Representation**, Elsevier, v. 60, p. 407–416, 2019.	No
RIBEIRO, L. S. F.; BUI, T.; COLLOMOSSE, J.; PONTI, M. Sketchformer: Transformer-based representation for sketched structure. In: **Proc. CVPR**. [S.l.: s.n.], 2020.	Yes
RIBEIRO, L. S. F.; BUI, T.; COLLOMOSSE, J.; PONTI, M. Scene designer: a unified model for scene search and synthesis from sketch. In: **Proc. ICCV WS**. [S.l.: s.n.], 2021. p. 2424–2433.	Yes
PONTI, M. A.; SANTOS, F. P. dos; RIBEIRO, L. S.; CAVALLARI, G. B. Training deep networks from zero to hero: avoiding pitfalls and going beyond. In: IEEE. **2021 34th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)**. [S.l.], 2021. p. 9–16.	No
RIBEIRO, L. S. F.; BUI, T.; COLLOMOSSE, J.; PONTI, M. Scene designer: Compositional sketch-based image retrieval with contrastive learning and an auxiliary synthesis task. **Multimedia Tools and Applications**, Springer, 2022.	Yes
RIBEIRO, L. S. F.; PONTI, M. Sketch-an-anchor: Sub-epoch fast model adaptation for zero-shot sketch-based image retrieval. In: **Submission Under Review**. [S.l.: s.n.], 2022.	Yes

Table 21 – List of published and submitted articles co-authored by the candidate within the Ph.D. program. Listed in chronological order.

ANNEX

A

ORIGINAL PUBLICATIONS

A.1 Sketchformer: Transformer-based representation for sketched structure.

This section includes the original published copy of Sketchformer from the Computer Vision Foundation. This paper was first presented at the CVF Conference on Computer Vision and Pattern Recognition (CVPR 2020):

RIBEIRO, L. S. F.; BUI, T.; COLLOMOSSE, J.; PONTI, M. Sketchformer: Transformer-based representation for sketched structure. In: **Proc. CVPR**. [S.l.: s.n.], 2020.

This CVPR 2020 paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Sketchformer: Transformer-based Representation for Sketched Structure

Leo Sampaio Ferraz Ribeiro[*1], Tu Bui[*2], John Collomosse[2, 3], and Moacir Ponti[1]

[1]ICMC, Universidade de São Paulo – São Carlos/SP, Brazil
leo.sampaio.ferraz.ribeiro@gmail.com, ponti@usp.br
[2]CVSSP, University of Surrey – Guildford, Surrey, UK
{t.bui,j.collomosse}@surrey.ac.uk
[3]Adobe Research, Creative Intelligence Lab – San Jose, CA, USA

Abstract

Sketchformer is a novel transformer-based representation for encoding free-hand sketches input in a vector form, i.e. *as a sequence of strokes. Sketchformer effectively addresses multiple tasks: sketch classification, sketch based image retrieval (SBIR), and the reconstruction and interpolation of sketches. We report several variants exploring continuous and tokenized input representations, and contrast their performance. Our learned embedding, driven by a dictionary learning tokenization scheme, yields state of the art performance in classification and image retrieval tasks, when compared against baseline representations driven by LSTM sequence to sequence architectures: SketchRNN and derivatives. We show that sketch reconstruction and interpolation are improved significantly by the Sketchformer embedding for complex sketches with longer stroke sequences.*

1. Introduction

Sketch representation and interpretation remains an open challenge, particularly for complex and casually constructed drawings. Yet, the ability to classify, search, and manipulate sketched content remains attractive as gesture and touch interfaces reach ubiquity. Advances in recurrent network architectures within language processing have recently inspired sequence modeling approaches to sketch (*e.g.* SketchRNN [1]) that encode sketch as a variable length sequence of strokes, rather than in a rasterized or 'pixel' form. In particular, long-short term memory (LSTM) networks have shown significant promise in learning search embeddings [2, 3] due to their ability to model higher-level structure and temporal order versus convolutional neural networks (CNNs) on rasterized sketches [4, 5, 6, 7]. Yet, the limited temporal extent of LSTM restricts the structural complexity of sketches that may be accommodated in sequence embeddings. In language modeling domain, this shortcoming has been addressed through the emergence of Transformer networks [8, 9, 10] in which slot masking enhances the ability to learn complex structures that are represented by longer sequences.

This paper proposes Sketchformer, the first Transformer based network for learning a deep representation for free-hand sketches. We build on the language modeling Transformer architecture of Vaaswani *et al*. [10] to develop several variants of Sketchformer that process sketch sequences in continuous and tokenized forms. We evaluate the efficacy of each learned sketch embedding for common sketch interpretation tasks. We make three core technical contributions:
1) Sketch Classification. We show that Sketchformer driven by a dictionary learning tokenization scheme outperforms state of the art sequence embeddings for sketched object recognition over QuickDraw! [11]; the largest and most diverse public corpus of sketched objects.
2) Generative Sketch Model. We show that for more complex, detailed sketches comprising lengthy stroke sequences, Sketchformer improves generative modeling of sketch – demonstrated by higher fidelity reconstruction of sketches from the learned embedding. We also show that for sketches of all complexities, interpolation in the Sketchformer embedding is stable, generating more plausible intermediate sketches for both inter- and intra-class blends.
3) Sketch based Image Retrieval (SBIR) We show that Sketchformer can be unified with raster embedding to produce a search embedding for SBIR (after [3] for LSTM) to deliver improved prevision over a large photo corpus (Stock10M).

These enhancements to sketched object understanding, generative modeling and matching demonstrated for a diverse and complex sketch dataset suggest Transformer as a promising direction for stroke sequence modeling.

2. Related Work

Representation learning for sketch has received extensive attention within the domain of visual search. Classical

[*]These authors contributed equally to this work

sketch based image retrieval (SBIR) techniques explored region [12], graph [13], edge-let [14], and sparse gradient features [15, 16] building upon the success of dictionary learning based models (e.g. bag of words) [17, 18, 19]. With the advent of deep learning, convolutional neural networks (CNNs) were rapidly adopted to learn search embedding [20]. Triplet loss models are commonly used for visual search in the photographic domain [21, 22, 23], and have been extended to SBIR. Sangkloy *et al*. [7] used a three-branch CNN with triplet loss to learn a general cross-domain embedding for SBIR. Fine-grained (within-class) SBIR was similarly explored by Yu *et al*. [24]. Qi *et al*. [5] instead use contrastive loss to learn correspondence between sketches and pre-extracted edge maps. Bui *et al*. [25, 4] perform cross-category retrieval using a triplet model and combined their technique with a learned model of visual aesthetics [26] to constrain SBIR using aesthetic cues in [6]. A quadruplet loss was proposed by [27] for fine-grained SBIR. The generalization of sketch embeddings beyond training classes have also been studied [28, 29], and parameterized for zero-shot learning [30]. Such concepts were later applied in sketch-based shape retrieval tasks [31]. Variants of CycleGAN [32] have also shown to be useful as generative models for sketch [33]. Sketch-A-Net was a seminal work for sketch classification that employed a CNN with large convolutional kernels to accommodate the sparsity of stroke pixels [24]. Recognition of partial sketches has also been explored by [34]. Wang *et al*. [35] proposed sketch classification by sampling unordered points of a sketch image to learning a canonical order.

All the above works operate over rasterized sketches *e.g*. converting the captured vector representation of sketch (as a sequence of strokes) to pixel form, discarding temporal order of strokes, and requiring the network to recover higher level spatial structure. Recent SBIR work has begun to directly input a vector (stroke sequence) representations for sketches [36], notably SketchRNN; an LSTM based sequence to sequence (seq2seq) variational auto-proposed by Eck *et al*. [1], trained on the largest public sketch corpus 'QuickDraw!' [11]. SketchRNN embedding was incorporated in a triplet network by Xu *et al*. [2] to search for sketches using sketches. A variation using cascaded attention networks was proposed by [37] to improve vector sketch classification over Sketch-A-Net. Later, LiveSketch [3] extended SketchRNN to a triplet network to perform SBIR over tens of millions of images, harnessing the sketch embedding to suggest query improvements and guide the user via relevance feedback. The limited temporal scope of LSTM based seq2seq models can prevent such representations modeling long, complex sketches, a problem mitigated by our Transformer based model which builds upon the success shown by such architectures for language modeling [10, 9, 8]. Transformers encode long term temporal dependencies by modeling direct connections between data units. The temporal range of such dependencies was increased via the Transformer-XL [9] and BERT [8], which recently set new state-of-the-art performance on sentence classification and sentence-pair regression tasks using a cross-encoder. Recent work explores transformer beyond sequence modeling to 2D images [38]. Our work is first to apply these insights to the problem of sketch modeling, incorporating the Transformer architecture of Vaswani *et al*. [10] to deliver a multi-purpose embedding that exceeds the state of the art for several common sketch representation tasks.

3. Sketch Representation

We propose Sketchformer; a multi-purpose sketch representation from stroke sequence input. In this section we discuss the pre-processing steps, the adaptions made to the core architecture proposed by Vaswani *et al*. [10] and the three application tasks.

3.1. Pre-processing and Tokenization

Following Eck *et al*. [1] we simplify all sketches using the RDP algorithm [39] and normalize stroke length. Sketches for all our experiments are drawn from QuickDraw50M [11] (see Sec. 4; for dataset partitions).

1) Continuous. Quickdraw50M sketches are released in the 'stroke-3' format where each point $(\delta x, \delta y, p)$ stores its relative position to the previous point together with its binary pen state. To also include the 'end of sketch' state, the stroke-5 format is often employed: $(\delta x, \delta y, p_1, p_2, p_3)$, where the the pen states p_1 - *draw*, p_2 - *lift* and p_3 - *end* are mutually exclusive [1]. Our experiments with continuous sketch modeling use the 'stroke-5' format.

2) Dictionary learning. We build a dictionary of K code words ($K = 1000$) to model the relative pen motion *i.e*. $(\delta x, \delta y)$. We randomly sample 100k sketched pen movements in the training set for clustering via K-means. We allocate 20% of sketch points for sampling inter-stroke transition, *i.e*. relative transition when the pen is lifted, to balance with the more common within-stroke transitions. Each transition point $(\delta x, \delta y)$ is then assigned to the nearest code word, resulting in a sequence of discrete tokens. We also include 4 special tokens; a Start of Sketch (SOS) token at the beginning of every sketch, an End of Sketch (EOS) token at the end, a Stroke End Point (SEP) token to be inserted between strokes (indicate pen lifting) and a padding (PAD) token to pad the sketch to a fixed length.

3) Spatial Grid. The sketch canvas is first quantized into $n \times n$ ($n = 100$) square cells, each cell is represented by a token in our dictionary. Given the absolute (x, y) sketch points, we determine which cell contains this point and assign the cell's token to the point. The same four special tokens above are used to complete the sketch sequence.

Fig. 2 visualizes sketch reconstruction under the tokenized methods to explore sensitivity to the quantization parameters. Compared with the stroke-5 format (continuous) the tokenization methods are more compact. Dictionary learned tokenization (Tok-Dict) can have a small dictionary size and is invariant to translation since it is derived from stroke-3. On the other hand quantization error could accumulate over longer sketches if dictionary size is too low,

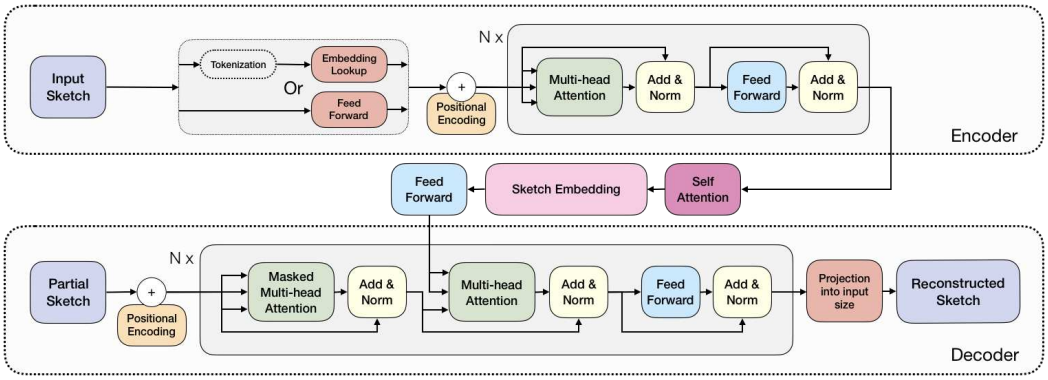


Figure 1. Schematic of the Transformer architecture used for Sketchformer, which utilizes the original architecture of Vaswani *et al*. [10] but modifies it with an alternate mechanism for formulating the bottleneck (sketch embedding) layer using a self-attention block, as well as configuration changes e.g. MHA head count (see Sec. 3.2).

shifting the position of strokes closer to the sequence's end. The spatial grid based tokenization method (Tok-Grid), on the other hand, does not accumulate error but is sensitive to translation and yields a larger vocabulary (n^2).

3.2. Transformer Architecture for Sketch

Sketchformer uses the Transformer network of Vaswani *et al*. [10]. We add stages (*e.g*. self-attention and modified bottleneck) and adapt parameters in their design to learn a multi-purpose representation for stroke sequences, rather than language. A transformer network consists of an encoder and decoder blocks, each comprising several layers of multihead attention followed by a feed forward network. Fig. 1 illustrates the architecture with dotted lines indicating re-use of architecture stages from [10]. In Fig. 4 we show how our learned embedding is used across multiple applications. Compared to [10] we use 4 MHA blocks versus 6 and a feed-forward dimension of 512 instead of 2048. Unlike traditional sequence modeling methods (RNN/LSTM) which learns the temporal order of current time steps from previous steps (or future steps in bidirectional encoding), the attention mechanism in transformers allows the network to decide which time steps to focus on to improve the task at hand. Each multihead attention (MHA) layer is formulated as such:

$$SHA(k,q,v) = softmax(\alpha q k^T)v \quad (1)$$

$$MHA(k,q,v) = [SHA_0(kW_0^k, qW_0^q, vW_0^v), ... \quad (2)$$

$$SHA_m(kW_m^k, qW_m^q, vW_m^v)]W^0 \quad (3)$$

where k, q and v are respective *Key*, *Query* and Value inputs to the single head attention (SHA) module. This module computes the similarity between pairs of *Query* and *Key* features, normalizes those scores and finally uses them as a projection matrix for the *Value* features. The multihead attention (MHA) module concatenates the output of multiple single heads and projects the result to a lower dimension. α is a scaling constant and $W_{(.)}^{(.)}$ are learnable weight matrices.

The MHA output is fed to a positional feed forward network (FFN), which consists of two fully connected layers with ReLU activation. The MHA-FFN ($F(.)$) blocks are the basis of the encoder side of our network ($\mathcal{E}(.)$):

$$FFN(x) = max(0, xW_1^f + b_1^f)W_2^f + b_2^f \quad (4)$$

$$F(x) = \overline{FFN}(\overline{MHA}(x,x,x)) \quad (5)$$

$$\mathcal{E}(x) = F_N(..(F_1(x))) \quad (6)$$

where $\overline{X}$ indicates layer normalization over X and N is number of the MHA-FFN units $F(.)$.

The decoder takes as inputs the encoder output and target sequence in an auto-regressive fashion. In our case we are learning an transformer autoencoder so the target sequence is also the input sequence shifted forward by 1:

$$G(h,x) = \overline{FFN}(\overline{MHA}_2(h,h,\overline{MHA}_1(x,x,x))) \quad (7)$$

$$\mathcal{D}(h,x*) = G_N(h, G_{N-1}(h, ...G_1(h,x*))) \quad (8)$$

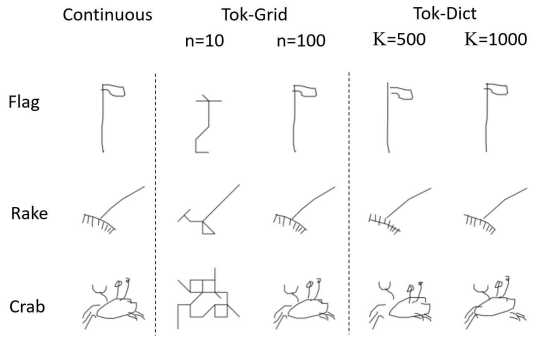


Figure 2. Visualizing the impact of quantization on the reconstruction of short, median and long sequence length sketches. Grid sizes of $n = [10, 100]$ (Tok-Grid) and dictionary sizes of $K = [500, 1000]$ (Tok-Dict). Sketches are generated from the tokenized sketch representations, independent of transformer.

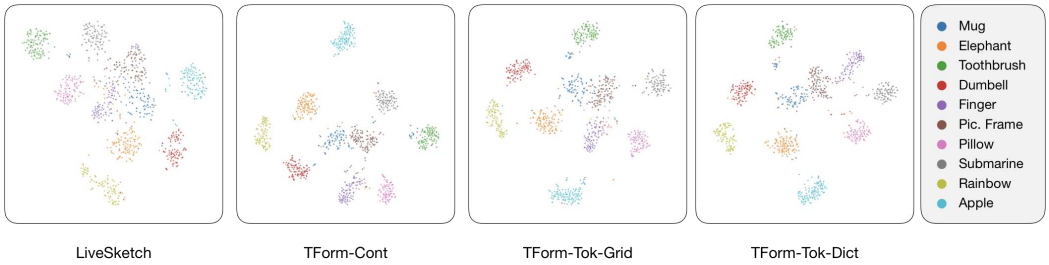


Figure 3. t-SNE visualization of the learned embedding space from Sketchformer's three variants, compared to LiveSketch (left) and computed over the QD-862k test set; 10 categories and 1000 samples were randomly selected.

where h is the encoder output, $x*$ is the shifted auto-regressive version of input sequence x.

The conventional transformer is designed for language translation and thus does not provide a feature embedding as required in Sketchformer (output of $\mathcal{E}$ is also a sequence of vectors of the same length as x). To learn a compact representation for sketch we propose to apply self-attention on the encoder output, inspired by [40]:

$$s = softmax(tanh(hK^T + b)v) \tag{9}$$

$$z = \sum_i s_i h_i \tag{10}$$

which is similar to SHA however the Key matrix K, Value vector v and bias b are now trainable parameters. This self-attention layer learns a weight vector s describing the importance of each time step in sequence h, which is then accumulated to derive the compact embedding z. On the decoder side, z is passed through a FFN to resume the original shape of h. These are the key novel modifications to the original Transformer architecture of Vaswani *et al*. (beyond above-mentioned parameter changes).

We also had to change how masking worked on the decoder. The Transformer uses a padding mask to stop attention blocks from giving weight to out-of-sequence points. Since we want a meaningful embedding for reconstruction and interpolation, we removed this mask from the decoder, forcing our transformer to learn reconstruction without previously knowing the sequence length and using only the embedding representation.

3.2.1 Training Losses

We employ two losses in training Sketchformer. A classification (softmax) loss is connected to the sketch embedding z to preserve semantic information while a reconstruction loss ensures the decoder can reconstruct the input sequence from its embedding. If the input sequence is continuous (*i.e.* stroke-5) the reconstruction loss consists of a L^2 loss term modeling relative transitions $(\delta x, \delta y)$ and a 3-way classification term modeling the pen states. Otherwise the reconstruction loss uses softmax to regularize a dictionary of sketch tokens as per a language model. We found these losses simple yet effective in learning a robust sketch embedding. Fig. 3 visualizes the learned embedding for each of the three pre-processing variants, alongside that of a state of the art sketch encoding model using stroke sequences [3].

3.3. Cross-modal Search Embedding

To use our learned embedding for SBIR, we follow the joint embedding approach first presented in [3] and train an auxiliary network that unifies the vector (sketch) and raster (image corpus) representations into a common subspace.

This auxiliary network is composed of four fully connected layers (see Fig. 4) with ReLU activations. These are trained within a triplet framework and have input from three pre-trained branches: an anchor branch that models vector representations (our Sketchformer), plus positive and negative branches extracting representations from raster space.

The first two fully connected layers are domain-specific and we call each set $F_V(.), F_R(.)$, referring to vector-specific and raster-specific. The final two layers are shared between domains; we refer to this set as $F_S(.)$. Thus the end-to-end mapping from vector sketch and raster sketch/image to the joint embedding is:

$$u_v = F_S(F_V(\mathcal{E}(x_v))) \tag{11}$$

$$u_r = F_S(F_R(\mathcal{P}(x_r))) \tag{12}$$

where x_v and x_r are the input vector sketches and raster images respectively, and u_v and u_r their corresponding representations in the common embedding. $\mathcal{E}(.)$ is the network that models vector representations and $\mathcal{P}(.)$ is the one for raster images. In the original LiveSketch [3], $\mathcal{E}(.)$ is a SketchRNN [1]-based model, while we employ our multi-task Sketchformer encoder instead. For $\mathcal{P}(.)$ we use the same off-the-shelf GoogLeNet-based network, pre-trained on a joint embedding task (from [28]).

The training is performed using triplet loss regularized with the help of a classification task. Training requires an aligned sketch and image dataset *i.e.* a sketch set and image set that share the same category list. This is not the case for Quickdraw, which is a sketch-only dataset without a corresponding image set. Again following [3], we use the raster sketch as a medium to bridge vector sketch with raster image. The off-the-shelf $\mathcal{P}(.)$ (from [28]) was trained

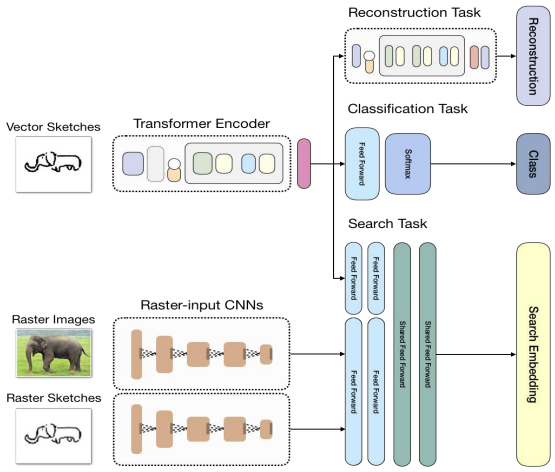


Figure 4. Schematic showing how the learned sketch embedding is leveraged for sketch synthesis (reconstruction/interpolation), classification and cross-modal retrieval experiments (see encoder/embedding inset, refer to Fig. 1 for detail). Classification appends fully-connected (fc) and softmax layers to the embedding space. Retrieval tasks require unification with a raster (CNN) embedding for images [28] via several fc layers trained via triplet loss.

to produce a joint embedding model unifying raster sketch and raster image; This allowed the authors train the F_R, F_V and F_S sets using vector and raster versions of sketch only. By following the same procedure, we eliminate the need of having an aligned image set for Quickdraw as our network never sees an image feature during training.

The training is implemented in two phases. At phase one, the anchor and positive samples are vector and raster forms of random sketches in the same category while raster input of the negative branch is sampled from a different category. At phase two, we sample hard negatives from the same category with the anchor vector sketch and choose the raster form of the exact instance of the anchor sketch for the positive branch. The triplet loss maintains a margin between the anchor-positive and anchor-negative distances:

$$\mathcal{L}_T(x, x_+, x_-) = max(0, |F_S(F_V(\mathcal{E}(x))) - F_S(F_R(\mathcal{P}(x_+)))| - ||F_S(F_V(\mathcal{E}(x))) - F_S(F_R(\mathcal{P}(x_-)))|| + m) \quad (13)$$

and margin $m = 0.2$ in phase one, $m = 0.05$ in phase two.

4. Experiments and Discussion

We evaluate the performance of the proposed transformer embeddings for three common tasks; sketch classification, sketch reconstruction and interpolation, and sketch based image retrieval (SBIR). We compare against two baseline sketch embeddings for encoding stroke sequences; SketchRNN [1] (also used for search in [2]) and LiveSketch [3]. We evaluate using sketches from QuickDraw50M [11], and a large corpus of photos (Stock10M).

QuickDraw50M [11] comprises over 50M sketches of 345 object categories, crowd-sourced within a gamified context that encouraged casual sketches drawn at speed. Sketches are often messy and complex in their structure, consistent with tasks such as SBIR. Quickdraw50M captures sketches as stroke sequences, in contrast to earlier raster-based and less category-diverse datasets such as TU-Berlin/Sketchy. We sample 2.5M sketches randomly with even class distribution from the public Quickdraw50M training partition to create training set (QD-2.5M) and use the public test partition of QuickDraw50M (QD-862k) comprising 2.5k $\times 345 = 862k$ sketches to evaluate our trained models. For SBIR and interpolation experiments we sort QD-862k by sequence length, and sample three datasets (QD345-S, QD345-M, QD345-L) at centiles 10, 50 and 90 respectively to create a set of short, medium and long stroke sequences. Each of these three datasets samples one sketch per class at random from the centile yielding three evaluation sets of 345 sketches. We sampled an additional query set QD345-Q for use in sketch search experiments, using the same 345 sketches as LiveSketch [3]. The median stroke lengths of QD345-S, QD345-M, QD345-L are 30, 47 and 75 strokes respectively (after simplification via RDP [39]).

Stock67M is a diverse, unannotated corpus of photos used in prior SBIR work [3] to evaluate large-scale SBIR retrieval performance. We sample 10M of these images at random for our search corpus (Stock10M).

4.1. Evaluating Sketch Classification

We evaluate the class discrimination of the proposed sketch embedding via attaching dense and softmax layers to the transformer encoder stage, and training a 345-way classifier on QD2.5M. Table 1 reports the classification performance over QD-862k for each of the three proposed transformer embeddings, alongside two LSTM baselines – the SketchRNN [1] and LiveSketch [3] variational autoencoder networks. Whilst all transformers outperform the baseline, the tokenized variant of the transformer based on dictionary learning (TForm-Tok-Dict) yields highest accuracy. We explore this further by shuffling the order of the sketch strokes retraining the transformer models from scratch. We were

Method		mAP%
Baseline	LiveSketch [3]	72.93
	SketchRNN [1]	67.69
Shuffled	TForm-Cont	76.95
	TForm-Tok-Grid	76.22
	TForm-Tok-Dict	76.66
Proposed	TForm-Cont	77.68
	TForm-Tok-Grid	77.36
	TForm-Tok-Dict	**78.34**

Table 1. Sketch classification results over QuickDraw! [11] for three variants of the proposed transformer embedding, contrasting each to models learned from randomly permuted stroke order. Comparing to two recent LSTM based approaches for sketch sequence encoding [3, 1].

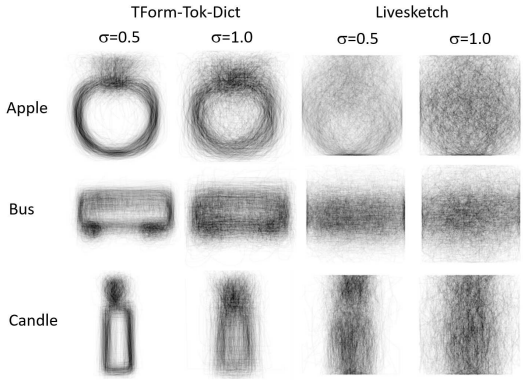


Figure 5. Visualization of sketches reconstructed from mean embedding for 3 object categories. We add Gaussian noise with standard deviation $\sigma = 0.5$ and $\sigma = 1.0$ to the mean embedding of three example categories on the Quickdraw test set. The reconstructed sketches of Tform-Tok-Dict retain salient features even with high noise perturbation.

Method		Short	Mid	Long
Baseline	LiveSketch [3]	**62.1**	**59.2**	27.8
	SketchRNN [1]	4.05	3.70	1.38
Proposed	TForm-Cont	0.00	0.00	0.00
	TForm-Tok-Grid	6.75	6.10	5.56
	TForm-Tok-Dict	24.3	28.4	**51.4**
	Uncertain	2.70	2.47	13.9

Table 2. User study quantifying accuracy of sketch reconstruction. Preference is expressed by 5 independent workers, and results with $> 50\%$ agreement are included. Experiment repeated for short, medium and longer stroke sequences. For longer sketches, the proposed transformer method TForm-Tok-Dict is preferred.

surprised to see comparable performance, perhaps due to the transformer's larger information scope rather than solely temporal information.

4.2. Reconstruction and Interpolation

We explore the generative power of the proposed embedding by measuring the degree of fidelity with which: 1) encoded sketches can be reconstructed via the decoder to resemble the input; 2) a pair of sketches may be interpolated within, and synthesized from, the embedding. The experiments are repeated for short (QD345-S), medium (QD345-M) and long (QD345-L) sketch complexities. We assess the fidelity of sketch reconstruction and the visual plausibility of interpolations via Amazon Mechanical Turk (MTurk). MTurk workers are presented with a set of reconstructions or interpolations and asked to make a 6-way preference choice; 5 methods and a 'cannot determine' option. Each task is presented to five unique workers, and we only include results for which there is $> 50\%$ (*i.e.* > 2 worker) consensus on the choice.

Reconstruction results are shown in Table 2 and favor the LiveSketch [3] embedding for short or medium length strokes, with the proposed tokenized transformer (TForm-Tok-Dict) producing better results for more complex sketches aided by the improved representational power of transformer for longer stroke sequences. Fig 6 provides representative visual examples for each sketch complexity.

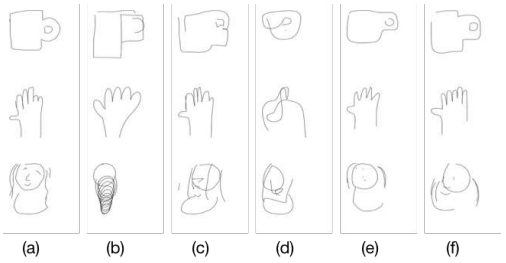


Figure 6. Representative sketch reconstructions from each of the five embeddings evaluated in Table 2. (a) Original, (b) SketchRNN, (c) LiveSketch, (d) TForm-Cont, (e) TForm-Tok-Grid and (f) TForm-Tok-Dict. The last row represents a hard-to-reconstruct sketch.

We explore interpolation in Table 3 blending between pairs of sketches within (intra-) class and between (inter-) class. In all cases we encode sketches separately to the embedding, interpolate via slerp (after [1, 3] in which slerp was shown to offer best performance), and decode the interpolated point to generate the output sketch. Fig. 7 provides visual examples of inter- and intra- class interpolation for each method evaluated. In all cases the proposed tokenized transformer (TForm-Tok-Dict) outperforms other transformer variants and baselines, although the performance separation is narrower for shorter strokes echoing results of the reconstruction experiment. The stability of our representation is further demonstrated via local sampling within the embedding in Fig. 5.

4.3. Cross-modal Matching

We evaluate the performance of Sketchformer for sketch based retrieval of sketches (S-S) and images (S-I).

Sketch2Sketch (S-S) Matching. We quantify the accuracy of retrieving sketches in one modality (raster) given a sketched query in another (vector, *i.e.* stroke sequence) – and vice-versa. This evaluates the performance of Sketchformer in discriminating between sketched visual structures invariant to their input modality. Sketchformer is trained on QD-2.5M and we query the test corpus QD-826k using QD-345Q as the query set. We measure overall mean average precision (mAP) for both coarse grain (*i.e.* class-specific) and fine-grain (*i.e.* instance-level) similarity. As per [3] for the former we consider a retrieved record a match if it matches the sketched object class. For the latter, exactly the same single sketch must match (in its different modality). To run raster variants, a rasterized version of QD-862k (for V-R) and of QD345-Q (for R-V) is produced by rendering strokes to a 256×256 pixel canvas using the CairoSVG Python library. Table 4 show that for both class and instance level retrieval, the R-V con-

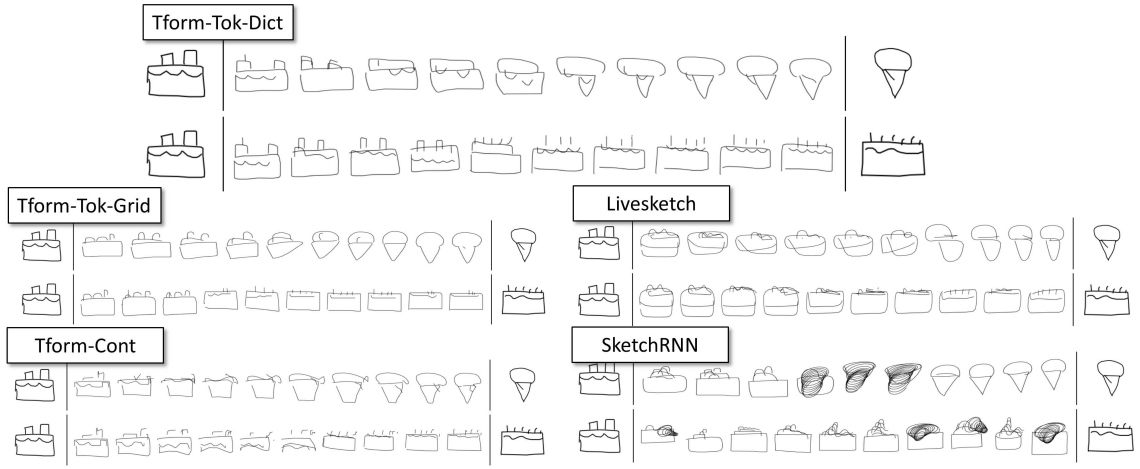


Figure 7. Representative sketch interpolations from each of the five embeddings evaluated in Table 3. For each embedding: (first row) inter-class interpolation from 'birthday cake' to 'ice-cream' and (second row) intra-class interpolation between two 'birthday cakes'.



Figure 8. Representative visual search results over Stock10M indexed by our proposed embedding (TForm-Tok-Dict) for a vector sketch query. The two bottom rows ('animal migration' and 'tree') are failure cases.

figuration outperforms V-R indicating a performance gain due to encoding this large search index using the vector representation. In contrast to other experiments reported, the continuous variant of Sketchformer appears slightly preferred, matching higher for early ranked results for the S-S case – see Fig. 9a for category-level precision-recall curve. Although Transformer outperforms RNN baselines by 1-3% in the V-R case the gain is more limited and indeed the performance over baselines is equivocal in the S-S where the search index is formed of rasterized sketches.

Sketch2Image (S-I) Matching. We evaluate sketch based image retrieval (SBIR) over Stock10M dataset of diverse photos and artworks, as such data is commonly indexed for large-scale SBIR evaluation [6, 3]. We compare against the state of the art SBIR algorithms accepting vector (LiveSketch [3]) and raster (Bui *et al*. [28]) sketched queries. Since no ground-truth annotation is possible for this size of corpus, we crowd-source per-query annotation via Mechanical Turk (MTurk) for the top-k (k=15) results and compute both mAP% and precision@k curve averaged across all QD345-Q query sketches, both with $k = [1-15]$. Table 5 compares performance of our tokenized variants to these baselines, alongside associated Precision@k curves in Fig. 9b. The proposed dictionary learned transformer embedding (TForm-Tok-Dict) delivers the best performance (visual results in Fig. 8).

Partition	Embedding	Intra-	Inter-
Short (10^{th} cent.)	SketchRNN [1]	0.00	2.06
	LiveSketch [3]	25.8	30.9
	TForm-Cont	14.0	6.18
	TForm-Tok-Grid	19.4	17.5
	TForm-Tok-Dict	**31.2**	**33.0**
	Uncertain	8.60	10.3
Mean (50^{th} cent.)	SketchRNN [1]	0.00	0.00
	LiveSketch [3]	25.2	20.2
	TForm-Cont	15.6	16.0
	TForm-Tok-Grid	15.6	19.1
	TForm-Tok-Dict	**35.8**	**35.1**
	Uncertain	7.36	9.57
Long (90^{th} cent.)	SketchRNN [1]	0.00	0.00
	LiveSketch [3]	25.0	21.1
	TForm-Cont	12.5	8.42
	TForm-Tok-Grid	16.7	10.5
	TForm-Tok-Dict	**40.6**	**50.5**
	Uncertain	5.21	9.47

Table 3. User study quantifying interpolation quality for a pair of sketches of same (intra-) or between (inter-) classes. Preference is expressed by 5 independent workers, and results with $> 50\%$ agreement are included. Experiment repeated for short, medium and longer stroke sequences.

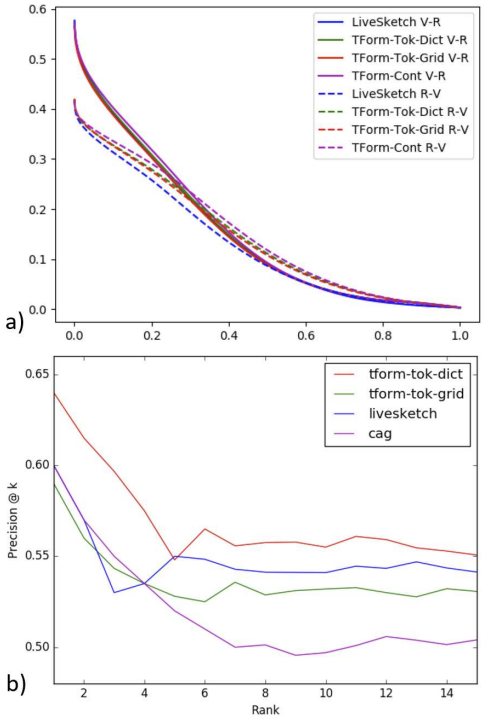


Figure 9. Quantifying search accuracy. a) Sketch2Sketch via precision-recall (P-R) curves for Vector-2-Raster and Raster-2-Vector category-level retrieval. b) Sketch2Image (SBIR) accuracy via precision @ k=[1,15] curve over Stock10M.

Method		Instance	Category
Livesketch [3]	V-R	6.71	20.49
	R-V	7.15	20.93
TForm-Cont	V-R	5.29	22.22
	R-V	6.21	**23.48**
TForm-Tok-Grid	V-R	6.42	21.26
	R-V	**7.38**	22.10
TForm-Tok-Dict	V-R	6.07	21.56
	R-V	7.08	22.51

Table 4. Quantifying the performance of Sketch2Sketch retrieval under two RNN baselines and three proposed variants. We report category- and instance-level retrieval (mAP%).

Method		mAP%
Baseline	LiveSketch [3]	54.80
	CAG [4]	51.97
Proposed	TForm-Tok-Grid	53.75
	TForm-Tok-Dict	**56.96**

Table 5. Quantifying accuracy of Sketchformer for Sketch2Image search (SBIR). Mean average precision (mAP) computed to rank 15 over Stock10M for the QD345-Q query set.

5. Conclusion

We presented Sketchformer: a learned representation for sketches based on the Transformer architecture [10]. Several variants were explored using continuous and tokenized input; a dictionary learning based tokenization scheme delivers classification performance gains of 6% on previous LSTM autoencoder models (SketchRNN and derivatives). We showed interpolation within the embedding yields plausible blending of sketches within and between classes, and that reconstruction (auto-encoding) of sketches is also improved for longer, more complex sketches. Sketchformer was also shown effective as a basis for indexing sketch and image collections for sketch based visual search. Future work could further explore our continuous representation variant, or other variants with more symmetric encoder-decoder structure. We have demonstrated the potential for Transformer networks to learn a multi-purpose representation for sketch, but believe many further applications of Sketchformer exist beyond the three tasks studied here. For example, fusion with additional modalities might enable sketch driven photo generation [41] using complex sketches, or with a language embedding for novel sketch synthesis applications.

Acknowledgments

This work was supported in part by FAPESP (grants 2017/22366-8, 2019/02808-1 and 2018/22482-0), CNPq Fellowship (307973/2017-4), and a charitable donation from Adobe Inc.

References

[1] D. Ha and D. Eck. A neural representation of sketch drawings. In *Proc. ICLR*. IEEE, 2018. 1, 2, 4, 5, 6, 8

[2] P. Xu, Y. Huang, T. Yuan, K. Pang, Y-Z. Song, T. Xiang, and T. Hospedales. Sketchmate: Deep hashing for million-scale human sketch retrieval. In *Proc. CVPR*, 2018. 1, 2, 5

[3] J. Collomosse, T. Bui, and H. Jin. Livesketch: Query perturbations for guided sketch-based visual search. In *Proc. CVPR*, pages 1–9, 2019. 1, 2, 4, 5, 6, 7, 8

[4] T. Bui, L. Ribeiro, M. Ponti, and J. Collomosse. Compact descriptors for sketch-based image retrieval using a triplet loss convolutional neural network. *Computer Vision and Image Understanding (CVIU)*, 2017. 1, 2, 8

[5] Yonggang Qi, Yi-Zhe Song, Honggang Zhang, and Jun Liu. Sketch-based image retrieval via siamese convolutional neural network. In *Proc. ICIP*, pages 2460–2464. IEEE, 2016. 1, 2

[6] J. Collomosse, T. Bui, M. Wilber, C. Fang, and H. Jin. Sketching with style: Visual search with sketches and aesthetic context. In *Proc. ICCV*, 2017. 1, 2, 7

[7] Patsorn Sangkloy, Nathan Burnell, Cusuh Ham, and James Hays. The sketchy database: Learning to retrieve badly drawn bunnies. In *Proc. ACM SIGGRAPH*, 2016. 1, 2

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, v.1*, pages 4171–4186. Association for Computational Linguistics, 2019. 1, 2

[9] Zihang Dai, Zhilin Yang, Yiming Yang, William W Cohen, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019. 1, 2

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Proc. NeurIPS*. IEEE, 2017. 1, 2, 3, 8

[11] The Quick, Draw! Dataset. https://github.com/googlecreativelab/quickdraw-dataset. Accessed: 2018-10-11. 1, 2, 5

[12] R. Hu, T. Wang, and J. Collomosse. A bag-of-regions approach to sketch based image retrieval. In *Proc. ICIP*, 2011. 2

[13] S. James, R. Hu, T. Wang, and J. Collomosse. Markov random fields for sketch based video retrieval. In *Proc. ICMR*, 2013. 2

[14] Y. Cao, H. Wang, C. Wang, Z. Li, L. Zhang, and L. Zhang. Mindfinder: Interactive sketch based image search on millions of images. In *Proc. ACM MM*, 2010. 2

[15] Mathias Eitz, James Hays, and Marc Alexa. How do humans sketch objects? In *Proc. ACM SIGGRAPH*, volume 31, pages 44:1–44:10, 2012. 2

[16] Rui Hu and John Collomosse. A performance evaluation of gradient field HOG descriptor for sketch based image retrieval. *Computer Vision and Image Understanding (CVIU)*, 117(7):790–806, 2013. 2

[17] R. Hu, S. James, T. Wang, and J. Collomosse. Motion-sketch based video retrieval using a trellis levenshtein distance. In *Proc. ICPR*, pages 121–124, 2010. 2

[18] Tu Bui and John Collomosse. Scalable sketch-based image retrieval using color gradient features. In *Proc. ICCV Workshops*, pages 1–8, 2015. 2

[19] Rosália G Schneider and Tinne Tuytelaars. Sketch classification and classification-driven analysis using fisher vectors. *ACM Transactions on Graphics (TOG)*, 33(6):174, 2014. 2

[20] Hua Zhang, Si Liu, Changqing Zhang, Wenqi Ren, Rui Wang, and Xiaochun Cao. Sketchnet: Sketch classification with web images. In *Proc. CVPR*, pages 1105–1113, 2016. 2

[21] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In *Proc. CVPR*, pages 1386–1393, 2014. 2

[22] Filip Radenović, Giorgos Tolias, and Ondřej Chum. CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples. In *Proc. ECCV*, pages 3–20, 2016. 2

[23] Albert Gordo, Jon Almazán, Jerome Revaud, and Diane Larlus. Deep image retrieval: Learning global representations for image search. In *Proc. ECCV*, pages 241–257, 2016. 2

[24] Qian Yu, Feng Liu, Yi-Zhe Song, Tao Xiang, Timothy M Hospedales, and Chen-Change Loy. Sketch me that shoe. In *Proc. CVPR*, pages 799–807, 2016. 2

[25] T. Bui, L. Ribeiro, M. Ponti, and J. Collomosse. Generalisation and sharing in triplet convnets for sketch based visual search. *CoRR Abs*, arXiv:1611.05301, 2016. 2

[26] M. Wilber, C. Fang, H. Jin, A. Hertzmann, J. Collomosse, and S. Belongie. Bam! the behance artistic media dataset for recognition beyond photography. In *Proc. ICCV*, 2017. 2

[27] O. Seddati, S. Dupont, and S. Mahoudi. Quadruplet networks for sketch-based image retrieval. In *Proc. ICMR*, 2017. 2

[28] T. Bui, L. Ribeiro, M. Ponti, and J. Collomosse. Sketching out the details: Sketch-based image retrieval using convolutional neural networks with multi-stage regression. *Elsevier Computers & Graphics*, 2018. 2, 4, 5, 7

[29] K. Pang, K. Li, Y. Yang, H. Zhang, T. Hospedales, T. Xiang, and Y. Song. Generalising fine-grained sketch-based image retrieval. In *Proc. CVPR*, 2019. 2

[30] S. Dey, P. Riba, A. Dutta, J. Llados, and Y. Song. Doodle to search: Practical zero-shot sketch-based image retrieval. In *Proc. CVPR*, 2019. 2

[31] Yongzhe Xu, Jiangchuan Hu, Kun Zeng, and Yongyi Gong. Sketch-based shape retrieval via multi-view attention and generalized similarity. In *2018 7th International Conference on Digital Home (ICDH)*, pages 311–317. IEEE, 2018. 2

[32] J. Zhu, T. Park, P. Isola, and A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proc. ICCV*, 2017. 2

[33] Y. Song T. Xiang T. Hospedales J. Song, K. Pang. Learning to sketch with shortcut cycle consistency. In *Proc. CVPR*, 2018. 2

[34] Omar Seddati, Stephane Dupont, and Saïd Mahmoudi. Deepsketch 2: Deep convolutional neural networks for partial sketch recognition. In *2016 14th International Workshop on Content-Based Multimedia Indexing (CBMI)*, pages 1–6. IEEE, 2016. 2

[35] Xiangxiang Wang, Xuejin Chen, and Zhengjun Zha. Sketchpointnet: A compact network for robust sketch recognition. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 2994–2998. IEEE, 2018. 2

[36] Umar Riaz Muhammad, Yongxin Yang, Yi-Zhe Song, Tao Xiang, and Timothy M Hospedales. Learning deep sketch abstraction. In *Proc. CVPR*, pages 8014–8023, 2018. 2

[37] Lei Li, Changqing Zou, Youyi Zheng, Qingkun Su, Hongbo Fu, and Chiew-Lan Tai. Sketch-r2cnn: An attentive network for vector sketch recognition. *arXiv preprint arXiv:1811.08170*, 2018. 2

[38] N. Parmar, A. Vaswani, J. Uszkoreit, L. Kaiser, N. Shazeer, A. Ku, and D. Tran. Image transformer. In *Proc. NeurIPS*, 2019. 2

[39] David H Douglas and Thomas K Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: the international journal for geographic information and geovisualization*, 10(2):112–122, 1973. 2, 5

[40] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448, 2015. 4

[41] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Gaugan: semantic image synthesis with spatially adaptive normalization. In *ACM SIGGRAPH 2019 Real-Time Live!*, page 2. ACM, 2019. 8

A.2 Scene designer: a unified model for scene search and synthesis from sketch.

This section includes the original published copy of Scene Designer from the Computer Vision Foundation. This paper was first presented at the Sketching for Human Expressivity Workshop on the CVF International Conference on Computer Vision (ICCV 2021):

RIBEIRO, L. S. F.; BUI, T.; COLLOMOSSE, J.; PONTI, M. Scene designer: a unified model for scene search and synthesis from sketch. In: **Proc. ICCV WS**. [S.l.: s.n.], 2021. p. 2424–2433.

This ICCV workshop paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Scene Designer: a Unified Model for Scene Search and Synthesis from Sketch

Leo Sampaio Ferraz Ribeiro[1], Tu Bui[2], John Collomosse[2, 3], and Moacir Ponti[1]

[1]ICMC, Universidade de São Paulo – São Carlos/SP, Brazil
{leo.sampaio.ferraz.ribeiro,ponti}@usp.br
[2]CVSSP, University of Surrey – Guildford, Surrey, UK
{t.bui,j.collomosse}@surrey.ac.uk
[3]Adobe Research, Creative Intelligence Lab – San Jose, CA, USA
collomos@adobe.com

Abstract

Scene Designer is a novel method for searching and generating images using free-hand sketches of scene compositions; i.e. drawings that describe both the appearance and relative positions of objects. Our core contribution is a single unified model to learn both a cross-modal search embedding for matching sketched compositions to images, and an object embedding for layout synthesis. We show that a graph neural network (GNN) followed by Transformer under our novel contrastive learning setting is required to allow learning correlations between object type, appearance and arrangement, driving a mask generation module that synthesizes coherent scene layouts, whilst also delivering state of the art sketch based visual search of scenes.

1. Introduction

Creativity is increasingly inspired by the wealth of visual content online. Visual search eases content discovery and re-use, whilst generative artwork is emerging as a novel genre, driven by models trained on thousands of images. Yet the *fusion of search and synthesis* is under-explored. Generative content oveoffers users control but rarely the quality and diversity of real images. By contrast, search offers quality but not customization. Recent works have explored both image search and generation guided by free-hand sketches; an intuitive way to communicate visual intent. Sketch therefore offers an opportunity to unify search and synthesis technologies within a creative workflow.

This paper presents Scene Designer; a unified model for searching and generating scenes using free-hand sketches of scene compositions; i.e. drawings that describe both the appearance and relative positions of multiple objects. Our model provides a cross-modal search embedding where similarity of visual compositions comprising sketches and images (or mixtures of both) can be measured, as well as an object-level embedding for the synthesis of scene layouts to generate images. Our technical contributions are:

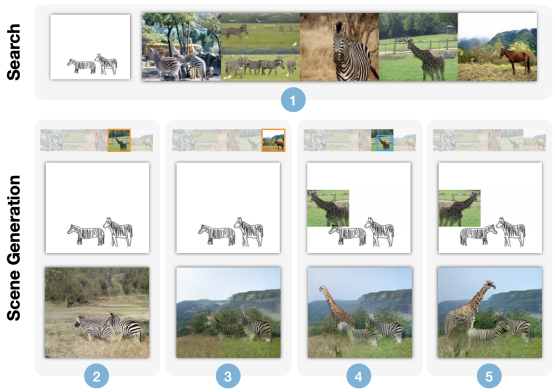


Figure 1. Scene Designer enables iterative design of image compositions through retrieval and synthesis. Row 1: User sketches initial concept, and matching scenes are returned (1). Row 2: In each column, the user uses the search results to help compose the final image; an orange square means the background was selected (2, 3) while the blue one means an object crop was chosen and added to the composition (4); on the final stage the user poses the objects as desired, and the final scene is synthesized (5).

1. Compositional Sketch Search. We propose a hybrid graph neural network (GNN) and Transformer architecture to learn a metric search embedding for comparing sketched and photographic scenes. Existing sketch based image retrieval (SBIR) methods predominantly match queries containing a single, dominant object invariant to its position within an image. Our novel contrastive training matches sketched compositions containing multiple objects.

2. Sketch-to-Scene Synthesis. We hallucinate entire scene layouts from either fully or partially sketched compositions, via a decoder that generates and combines object masks

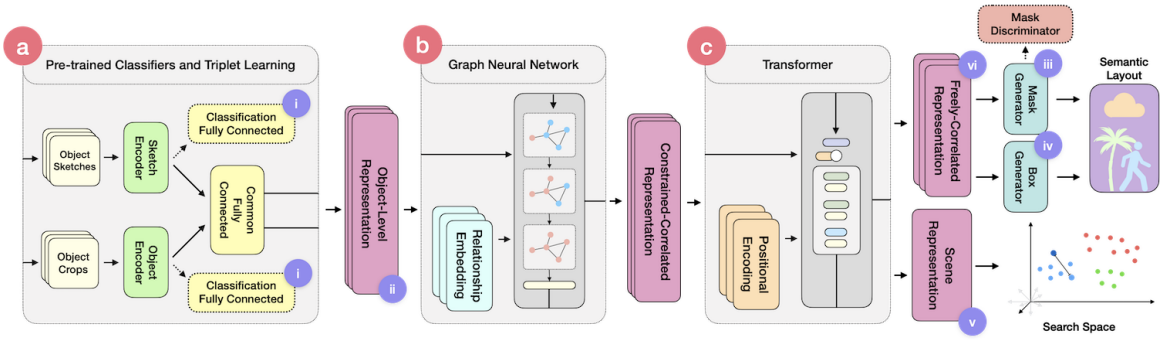


Figure 2. Proposed architecture of Scene Designer. Compositions (spatial arrangements of sketched or photoreal objects) are encoded via (a) to produce an *object-level representation*; *i.e*. a feature encoding for each object in the scene. A scene graph is constructed from these and encoded via GNN (b) to yield the *constrained-correlated representation*, and subsequently via Transformer to produce (c) the *freely-correlated representation* and *scene representation*. The roman numerals indicate where each of the losses (see Sec. 3.1, 3.4 and 3.5) are applied during training: (i) Cross-entropy classification $\mathcal{L}_{CCE}$, (ii) Triplet loss $\mathcal{L}_{tri}$, (iii) Mask generation losses $\mathcal{L}_{G_m}$, (iv) Bounding box generation losses $\mathcal{L}_{G_b}$, (v) Contrastive loss $\mathcal{L}_{cont}$, (vi) Final cross-entropy classification $\mathcal{L}_{CCE_f}$.

synthesized from learned correlations between object type, appearance and arrangement. The scene layouts drive texture synthesis to generate photographic content.

3. Fusion of Synthesis and Cross-Domain Search. Training a single mixed-task model yields improvements on the state of the art for both scene search and scene synthesis tasks. We also enable a novel creative workflow in which image compositions may be created iteratively; users sketch an initial composition (one or more objects), elements of the resulting images may be incorporated potentially with further sketched objects to drive either further searches or full image synthesis. Fig. 1 demonstrates how our fused model enables iterative design of generative compositions.

2. Related Work

Sketch based image retrieval (SBIR) has received extensive attention in the past decade. Dictionary learning methods (e.g. bag of visual words) have leveraged wavelet, edge-let, shape-let and sparse gradient features [1, 2] to match sketches to edge structures in images. With the advent of deep learning, CNNs (convnets) were rapidly adopted for cross-modal representation learning; exploring joint search embeddings for matching structure between sketches and images. Early approaches learned mappings between edge maps and sketches using contrastive [3], siamese [4] and triplet networks [5]. Fine-grained SBIR was explored by Yu *et al*. [6] and Sangkloy *et al*. [7] who used a three-branch CNN with triplet loss. Bui *et al*. learned a cross-domain embedding through triplet loss and partial weight sharing between sketch-image encoders [8]. Later studies followed a natural extension from single object sketches to scenes with multiple objects. Liu *et al*. [9] recently used a graph encoder with triplet loss to do compositional SBIR, but their results were dependent on known bounding boxes and category labels of photos and sketches. Our method also explores sketched compositions for search, learning a GNN and Transformer hybrid model for both search and synthesis.

Conditional Image Generation has been a rapidly developing field since the introduction of Conditional GANs [10]. Initial work exploring class label priors were soon followed by models with image based priors such as Pix2Pix [11] and more recently high definition synthesis models such as Pix2PixHD [12] and SPADE [13]. Those models learn from pairs of matching samples from each domain, and are capable of mapping semantic layouts to images.

Scene Graphs are a representation for images where individual objects are defined as nodes with the graph edges describe their relationships. They were first used for text-based image retrieval [14] but were more recently used by Johnson *et al*. [15] and Ashual *et al*. [16] as a initial representations to drive semantic layout generation and image synthesis based on those layouts. Our synthesis approach follows a similar principle, where our model learns to generate semantic layouts and a separate GAN (SPADE [13]) synthesizes the final image.

Image-synthesis from Sketch is a challenging problem due to the abstract and ambiguous nature of sketches. Recent work such as Sketchy-GAN [17] and Contextual-GAN [18] achieved promising results generating images from single-object sketches, but as was shown in [19], cannot accommodate sketched scenes with multiple objects. Gao *et al*. [19] were the first to implement image synthesis from sketch scenes and introduced the paired Sketchy-COCO dataset. Our approach differs, in that we learn a single mixed-task model for both search and synthesis — improving quantitatively and in user study synthesis results.

Joint image search and synthesis has been largely unexplored. Early work includes Sketch2Photo [20] that supports image search using text and sketch queries then compose an output image from objects of interest found in the

returned results. This approach uses statistical methods for image composition thus does not yield high quality output. In a more recent work, Sketchformer [21] also supports search and synthesis but only for sketch (single-domain). Similarly, Pang *et al*. [22] uses a sketch decoder mainly to assist their cross-domain learning thus have poor synthesized results. Our method learns both a cross-domain embedding for scene sketches and images and also enables image synthesis from composed objects in either domains.

3. Methodology

We developed a cross-modal representation learning framework in order to represent images and sketches with multiple objects in a common feature space; this representation is useful to both cross-modal retrieval and generation. Making use of scene graph representations, our framework looks at objects in a scene using a hierarchy based on object correlation. Fig. 2 describes the architecture and stages of the representation, which are each summarized below:

1. **Object-level representation (OLR):** Given an input composition, we encode individual objects (which might be sketched or photo-real) to a common representation. We refer to this embedding, in which each object is independently represented, as the OLR (subsec. 3.1).
2. **Scene Graph (SG)**: is formed using the OLR to encode objects, along with the discrete positional relationships of all object pairs.
3. **Constrained-correlated representation (CCR):** A graph neural network (GNN) encodes nodes in the Scene Graph (SG) into a continuous representation, encoding object appearances and their pairwise correlations (subsec. 3.2).
4. **Freely-correlated representation (FCR):** The sets of correlated vectors are fed into a Transformer module, where the attention layers allow for free correlation between all objects. The FCR encodes relationships between each object and all other objects in a weighted manner (subsec. 3.3). This representation is used for synthesis (subsec. 3.4).
5. **Scene representation (SR):** Together with the FCR, the Transformer module computes a separate single-vector latent space to which metric learning is applied to train a search embedding (subsec. 3.5).

We now describe in further detail how each stage of the representation is learned.

3.1. Object-level Representation (OLR)

We begin by independently encoding each object within the input composition to a common feature embedding, regardless of its input modality. Following contemporary single-object SBIR [8, 7, 6] we adopt a deep metric learning approach using a heterogeneous triplet architecture (*i.e*. no shared weights between the anchor (a) and positive/negative (p/n) branches). Our network comprises a MobileNet [23] backbone, terminated with two shared fully connected (fc) layers, the latter yielding the 128-D OLR embedding.

The MobileNet is pre-trained on ImageNet [24], and finetuned for cross-domain learning using a combined categorical cross-entropy (CCE) and triplet loss, with the latter defined as:

$$\begin{aligned}\mathcal{L}_{tri}\,(a,p,n) = \max\{0, m + |f_s(a), f_i(p)|_2 \\ - |f_s(a), f_i(n)|_2\}\end{aligned} \tag{1}$$

where $f_s(.)$ is the MobileNet that encodes sketches, $f_i(.)$ the one that encodes photo-real objects cropped from images, each followed by shared-domain MLP (2 fc layers) that encodes the MobileNet's outputs to the OLR, $m = 0.5$ is the margin and $|.|_2$ is the L_2 norm. Hereafter, we use shorthand $f(.)$ to refer to the encoder irrespective of its modality. The OLR is trained using rasterized sketches (a), and objects cropped from the COCO-stuff dataset with corresponding (p) and differing (n) object class; see subsec. 4.1 for dataset details. To aid convergence, the equal-weighted CCE loss $CCE(.)$ is applied to a further fc layer $f_e(.)$ appended to the network: vspace-0.4em

$$\mathcal{L}_{cce}(a,p,n) \;\; = \sum_{c\in\{a,p,n\}} CCE(f_e(c), \hat{c})) \tag{2}$$

where $\hat{a}$,$\hat{p}$, $\hat{n}$ are the one-hot vectors encoding the semantic class of each input.

3.2. Constrained-Correlated Representation (CCR)

A scene graph (SG) describes objects and their spatial relationships within the composition. Formally an SG is a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{o_1, \cdots, o_i, \cdots, o_\kappa\}$ is set of nodes representing the κ objects, and $(o_i, r_{ij}, o_j) \in \mathcal{E}$ the set of edges encoding discrete relationships, $r_{ij} \in \mathcal{R}$ (we used "left of", "right of", "above", "below", "contains", "inside of").

We encode the scene graph via a graph neural network (GNN). We represent each node via the OLR as $f(o_i)$, encoding the pose and appearance of the object. As in [16], we adopt a learnable embedding $f_r(r_{ij})$ to map relationships $\mathcal{R}$ into dense vectors. The GNN $g_r(.)$ comprises a sequence of 6 layers $g_r^k(.)$. In each we follow the architecture of [15], and edges are processed by an fc layer f_{fc1}^k:

$$\begin{aligned}< v_i^0, r_{ij}^0, v_j^0 > =< f(o_i), f_r(r_{ij}), f(o_j) > \\ < \hat{v}_i^k, r_{ij}^{k+1}, \hat{v_j}^k > = f_{fc1}^k(< v_i^k, r_{ij}^k, v_j^k >)\end{aligned} \tag{3}$$

that updates the vector for each relationship to r_{ij}^{k+1} and builds the set $\hat{v}_i^k \in V^k$ of object vectors. A function (h) gathers vectors in $\mathcal{V}$ that describe the same object and averages them; finally they are processed by another fc layer to compute the updated $v_i^{k+1} = f_{fc2}^k(h(V^k))$ for each object. These layers are shared between images and sketches and trained end-to-end as part of our model to learn the *constrained-correlated representation* (CCR).

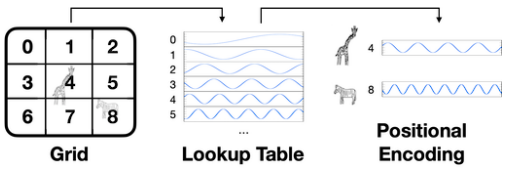


Figure 3. Grid-based Positional Encoding. We adapted the positional encoder from traditional Transformers by attributing one encoding to each block of the gridded scene.

3.3. Freely-Correlated Representation (FCR) and Scene Representation (SR)

We require a final representation that allows for objects to learn from their connection to all others in the scene, without being constrained to the pairwise encoding of the SG. We model this via attention layers, stacked similarly to a Transformer architecture [25]. This approach was inspired by the use of LSTMs for SG representation in [26] in order to accommodate a variable number of objects within a single representation. Transformers offer an attention mechanism to analyze each vector against all others, making them a good candidate for our model; They employ positional encoding so that the sequence order information is kept during processing. Our objects do not have a defined order, but we want their spacial positions to be taken into account. Following [16], we break the scene into a 5x5 grid and attribute a position $p \in 0, 1, ..., 24$ to each object based on where its center falls on the grid. This p is used to compute its positional encoding (see Fig. 3). Note that Ashual *et al*. concatenated this p to the object embeddings processed by the GNN; we took our approach because the vectors processed by our GNN are dynamic and because it aligns better with the Transformer architecture.

We specify our Transformer $t(.) = Z$ with 3 layers, and 16 attention heads on each. The input is the CCR $g_r(f(o_i))$ for each object in the image and one empty vector $\vec{0}$. In each layer $t^i(.)$ the model computes the attention weights that relate each vector to all the vectors in the sequence:

$$\begin{aligned} Z^0 &= s\left(F_q^0(Q+E)F_k^0(Q+E)^T\right)F_v^0(Q+E) \\ Z^{i+1} &= s\left(F_q^i(Z^i)F_k^i(Z^i)^T\right)F_v^i(Z^i), \end{aligned} \tag{4}$$

where multiplications are matrix-based, s is the softmax function, F_q^i, F_k^i, F_v^i are fc layers, Q is a matrix made by stacking $\vec{0}$ and $g_r(f(o_i))$ CCR vectors and E is our positional encodings $f_p(p)$, stacked in the same fashion as Q. In the output, the position that contained $\vec{0}$ has the SR while the other ones contain the FCR for each object.

3.4. Semantic Layout Generation

The learned FCR is used to synthesize a semantic scene layout (c.f. Fig. 7) that is used by our image generator. The layout is made from bounding boxes and masks for each object. We train two generators, one for masks and one for bounding boxes. The box generator is trained using Generalized Intersection-over-Union (GIOU) [27] and mean-squared error (MSE). The generator itself is a 2-layer MLP. The mask generator is trained within a GAN [28] framework, using a conditional setup [10] with LSGAN [29] losses plus MSE and Feature Matching to regularize the adversarial training. Additionally, another CCE classification loss $\mathcal{L}_{CCE_f}$ is applied to the FCR to encourage it to remain semantic.

Taking b as ground truth bounding box, $\hat{x}_i$ as the FCR based on image input and $\hat{x}_s$ as the FCR based on sketch input, our generation losses are, for bounding boxes:

$$\mathcal{L}_{G_b}(x) = \lambda_1 \mathcal{L}_{GIOU}(b, G_b(\hat{x})) + \lambda_2 |b, G_b(\hat{x})|_2 \tag{5}$$

for each object on each image, applied to both $\hat{x}_i$ and $\hat{x}_s$, with $\lambda_1 = \lambda_2 = 10$. Using a similar notation, take our ground truth masks as m_g, and the object labels as $y \in C$, where C is the set of object classes. The loss for the mask generator is:

$$\begin{aligned} \mathcal{L}_{G_m}(\hat{x}) = \lambda_3 |m_g, G_m(x)|_2 + \lambda_4 |D(G_m(\hat{x}), y), 1|_2 \\ + \lambda_5 \mathcal{L}_{FM}(\hat{D}(m_g, y), \hat{D}(G_m(x), y)) \end{aligned} \tag{6}$$

with D as the discriminator and also applied to both $\hat{x}_i$ and $\hat{x}_s$. The weights are $\lambda_3 = 10$, $\lambda_4 = 0.25$, $\lambda_5 = 10$. The $\mathcal{L}_{FM}(.)$ is the feature matching loss, the L1 difference in the activations of D ($\hat{D}(.)$). D itself is trained with the adversarial loss in classic GAN fashion. Those losses backpropagate through all representation levels.

Finally, we turn the masks and boxes into a semantic layout. In this object-only layout, objects are placed ordered by their size, so that bigger objects are behind smaller ones (after [16]); then it may be combined with a layout for the background (*e.g.* selected from the top search results or coarsely drawn). Ultimately the final layout is passed through a SotA SPADE model [13] to synthesize an image.

3.5. Compositional Sketch-based Image Retrieval

For scene retrieval, we learn a single *Scene Representation* (SR) as a search embedding in which the similarity of images and the sketched input may be measured. This is an additional latent vector computed by the Transformer, that has metric properties encouraged through an adaptation of the contrastive loss.

During contrastive training, we sample half of our negatives randomly from other images of scenes in the training set. The other half of the negatives are synthesized by swapping objects of different class into the positions of the objects in the positive image (Fig. 4). By including such a class-swapped version of the image as a negative, the model is encouraged to discriminate change in object class to a greater degree than changes in object positions. Our contrastive loss, adapted to incorporate these synthetic negatives, is:

$$\mathcal{L}_{cont}(x_s, x_i, x_{sn}) = \frac{1}{2}(Y)(\mathcal{D}(x_s, x_i)) + \frac{1}{4}(1-Y)(1-\mathcal{D}(x_s, x_i)) - \frac{1}{4}\mathcal{D}(x_s, x_{sn}), \quad (7)$$

where $\mathcal{D}$ computes $|.|_2$ between all vectors on one set and all vectors on another set, x_s and x_i are respectively the sketch-based and image-based SR, Y is a label indicating if a pair of vectors are from the same scene. Finally, x_{sn} is the SR of the synthesized scenes with swapped objects.

3.6. Training Stages

Training is performed in three stages. First the OLR is trained independent of the rest of the model, using the dual triplet and categorical cross-entropy loss, for 100K iterations, on single-object sketches and object crops. In the second stage we use our novel soft-paired sketch and image scenes dataset (see Sec. 4.1) to train entire model end-to-end for a further 120K iterations. Finally, we finetune the model with a hard-paired dataset, training for a further 5K iterations. Training is via the ADAM [30] optimizer, with $\beta_1 = 0.5$ and $\epsilon = 1e-9$; learning rate (lr) is $lr = 10^{-4}$ for the first two stages and $lr = 10^{-5}$ for the finetuning stage. The mask discriminator follows the *Two Time-Scale Update Rule* (TTUR) [31] with $lr_D = 4lr$.

4. Experiments and Discussion

We evaluate the performance of Scene Designer for both compositional sketch search (SBIR) and synthesis, contrasting performance against contemporary baselines for both tasks. We explore the efficacy of our model for both search and synthesis from sketch, image and mixed domain compositions. For SBIR we compare against four baselines: a scene-level technique (SceneSketcher [9]) and three single-object techniques. Of these, two are fine-grained SBIR models (Sketchy [7], and Sketch-me-that-shoe [6]), and one is coarse-grained (Multi-stage Learning or MSL [8]). For scene synthesis we compare against the sketch driven method of Gao *et al*. [19] (proposed alongside their SketchyCOCO dataset) and the method of Ashual *et al*. [16] that accepts semantic scene graphs (spatial arrangements of keywords) as input.

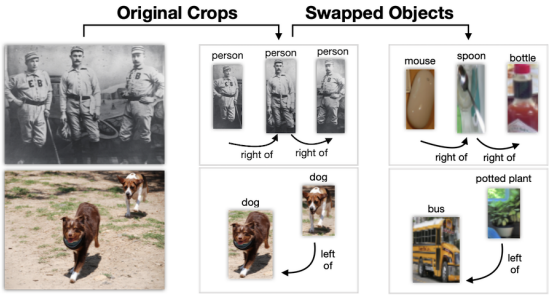


Figure 4. During contrastive training of the SR, negatives are synthesized by swapping objects in the scene for others of differing class. This helps the model prioritize class over structure. First column shows the original image, the second the individual cropped objects, and the third the swapped objects.

4.1. Datasets

We require paired data; sketched compositions and corresponding images of the scenes depicted by those sketches. **SketchyCOCO** is a recent dataset of 14K such pairs, proposed for sketch based scene synthesis [19]. SketchyCOCO samples scenes and associated annotations (bounding boxes, masks) from the COCO-stuff [32] images dataset, augmenting these with sketched depictions of those scenes. Approximately 4K of SketchyCOCO images contain object instances (the remainder are solely background material; 'stuff'). Only 14 classes (14c) of objects are annotated. To mitigate overfitting, we use the SketchyCOCO training partition (4,060 samples) only for fine-tuning our model. We include the test partition in our evaluation and split it per experiment requirements: retrieval eval. uses 233 samples as [9] needs scenes with 2 or more objects; generation tests needs to use the overlap between test sets used in each of the compared models: 137 samples. We also emulate [9] in constructing an 'Extended SketchyCOCO' search corpus by holding out 5K random images from the COCO-stuff training set and adding these to the SketchyCOCO test set as distractors.

QuickDrawCOCO-92c is a novel dataset we propose, composing scenes with sketches from QuickDraw, the largest public sketch dataset (50M) [33]. Inspired by SketchyCOCO, we similarly leverage scenes from COCO-stuff, taking advantage of the class overlap (92c) with QuickDraw. We synthesize sketch scenes by compositing sketches from QuickDraw onto a canvas, selecting a random sketch within the class corresponding to each object in a COCO-stuff image (Fig. 5). Our dataset is soft-paired, in that the sketches for each object do not necessarily match pose and appearance with the image, only the class label. We use QuickDrawCOCO-92c for all training stages except for the final finetuning stage where we combine samples from both SketchyCOCO and QuickDrawCOCO-92c. We separately fuse the training and test partitions of the QuickDraw and COCO-stuff datasets to partition QuickDrawCOCO-92c into 111,112 training, 2,788 test and 1,907 validation scenes. This dataset is 6x more category-diverse and has 8x more samples than SketchyCOCO.

4.2. Evaluating Retrieval

We adopt the evaluation proposed in SceneSketcher [9], the only prior technique exploring SBIR for scenes; retrieved images are relevant only if they match the sketched scene exactly. Models are evaluated on SketchyCOCO, QuickDrawCOCO-92c and 'Extended SketchyCOCO'.

Compositional search. Tab. 1 reports recall at rank $k =$

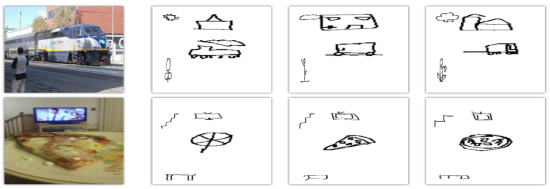


Figure 5. Sample paired data from the proposed QuickDrawCOCO-92c dataset. Each row shows one COCO-stuff [32] image and three examples of synthetic sketched compositions matching that image derived from QuickDraw [33].

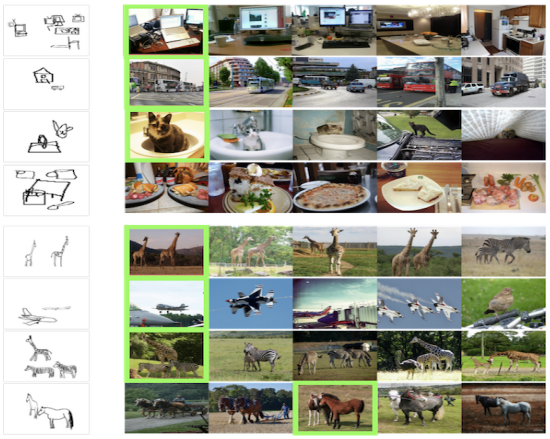


Figure 6. Retrieval results using the QuickDrawCOCO (first block) and SketchyCOCO sets (second). The results often contain the exact match and are contextually consistent, such as in the last row of QuickDrawCOCO, where the context of "table-spoon-mug-etc" retrieves sets of "meals". Relevant results in green; final row of each block represents a failure case.

$\{1, 5, 10\}$ (R@k) for our proposed search embedding for sketched scenes, that of SceneSketcher [9], and three single-object SBIR baselines: Sketch-me-that-shoe [6]; Sketchy [7]; and MSL [8]. We report R@k over SketchyCOCO and Extended SketchyCOCO for all methods; figures for SceneSketcher and Sketch-me-that-shoe are taken from Liu *et al*. [9]. We are unable to report values for these two models over our QuickDrawCOCO-92c test data, due to the lack of public models/code for these methods. QuickDrawCOCO-92c is a more challenging query set due to its increased size and the more messy / abstract sketches within QuickDraw. We present our results alongside public Sketchy [7] and MSL [8] models (final column, Tab. 1).

Our model more than doubles previous SotA recall results and greatly improves other metrics. The penultimate row of Tab. 1 reports performance our method using mixed-domain queries, where we substitute half of the sketched objects for their image cropped counterparts, which yields an even higher result likely due to half of the objects sharing the same domain as the test corpus. The final row reports on our model trained on QuickDrawCOCO-92c only; prior

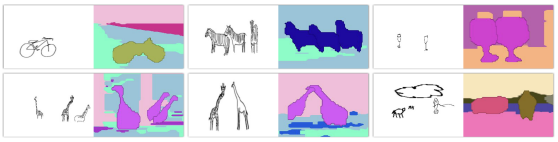


Figure 7. Example layouts synthesized by Scene Designer before they are transformed to images by the SPADE [13] generator. First 2 columns show SketchyCOCO scenes and third column shows QuickDrawCOCO-92c scenes.

to finetuning on SketchyCOCO training set. This demonstrates that our model generalizes sufficiently to beat the SotA on SketchyCOCO without explicit training on it. Visual results for both datasets are in Fig. 6.

Single object search. We evaluate the performance of our model at single-object retrieval, baselining against both the Sketchy and MSL models on the SketchyCOCO dataset. We sample 120 queries randomly from SketchyCOCO containing only one object. In contrast to composition search, many relevant results exist in the SketchyCOCO dataset for a given sketched object. We consider a result relevant if both its category and pose match the sketched query. Since no ground-truth annotation with these criteria is available for SketchyCOCO, we crowd-source per-query annotation via Mechanical Turk (MTurk) for the top-k (k=15) results and compute both recall@k and precision@k in Tab. 2. For each ranked result we ask 5 MTurk participants to indicate relevance (or not), and filter on consensus level. Our approach significantly outperforms both baselines at all consensus levels. The sensitivity to object orientation and pose underlines the utility of sketch over labeled boxes (per [16]) to specify the appearance of the desired object.

4.3. Evaluating Scene Synthesis

We evaluate the performance of our proposed model at image synthesis (see Fig.8) using SketchyCOCO, comparing against that dataset's accompanying public model [19], and against the public model of Ashual *et al*. [16] which synthesizes images from spatial word maps *i.e*. implicit scene graphs. We synthesize from the sketch, and use the paired image as ground truth for evaluating the fidelity of the output. For Ashual *et al*., we create an input scene graph based on object classes and bounding boxes.

Objective Metrics. We compute Fréchet Inception Distance (FID) [31] and Object Classification Accuracy [16] to compare the synthesized and ground truth images. A lower FID value indicates that the tested image distribution is closer the original and is more realistic. The Accuracy values come from finetuning a ResNet-101 classifier on crops from COCO and applying it to foreground objects cropped from the generated images.

Tab. 4 shows our model to score higher than both baselines on both FID and Accuracy. We have included results applying SPADE to the ground-truth layout, which serves as an upper bound generation result for our method. Results from mixed-domain and images-only compositions,

Method	SketchyCOCO			Ext. SCOCO			QuickDrawCOCO		
	r@1	r@5	r@10	r@1	r@5	r@10	r@1	r@5	r@10
Ours	**75.53**	**96.56**	**99.14**	**66.09**	**90.55**	**96.13**	**62.15**	**78.90**	**85.07**
Sketch-me-that-shoe [6]	06.19	17.15	32.86	<0.01	<0.01	01.90	-	-	-
Sketchy [7]	03.43	09.87	15.87	<0.01	00.85	00.85	00.07	00.35	00.60
MSL [8]	09.44	28.75	43.34	05.15	15.87	17.59	00.10	00.53	01.07
SceneSketcher [9]	31.91	66.67	86.19	12.38	26.67	38.10	-	-	-
Ours (Mixed-domain)	89.69	99.57	100.0	87.12	96.56	98.71	93.22	97.37	98.23
Ours (Before finetuning)	45.49	80.25	93.56	15.45	41.20	50.21	64.27	81.95	87.23

Table 1. Recall@K metrics on the SketchyCOCO dataset, its extended version and the QuickDrawCOCO-92c set. Our model more than doubles the previous state-of-the-art recall@1 metric. Additionally, we show our performance when using mixed-domain compositions and also with the model trained only with QuickDrawCOCO-92c, before finetuning on SketchyCOCO.

Method	Precision@1			Precision@15		
	C2	C3	C4	C2	C3	C4
Ours	**66.12**	**52.07**	**37.19**	**22.98**	**16.14**	**10.30**
MSL [8]	16.53	14.05	09.09	10.41	07.00	04.41
Sketchy [7]	35.54	23.97	18.18	17.36	11.79	07.77
	Recall@1			Recall@15		
	C2	C3	C4	C2	C3	C4
Ours	**66.12**	**52.07**	**37.19**	**96.69**	**90.91**	**71.90**
MSL [8]	16.53	14.05	09.09	84.30	72.73	56.20
Sketchy [7]	35.54	23.97	18.18	85.95	80.17	65.29

Table 2. MTurk sketch search results for the proposed and baseline methods on the SketchyCOCO dataset. Precision@k and Recall@k are presented at different (C)oncensus thresholds.

which are a unique capability of our model, are also included. These show the value in applying Scene Designer to incrementally build up a composition as part of an interactive creative process.

User Perceptual Study. We assess the quality of our synthesized images following the subjective evaluation methodology proposed by Gao *et al*. [19]. Specifically we score how 'realistic' each image is, and how 'faithful' each synthesized image is in representing the spatial object arrangement of the input scene (in the case of [16], spatial arrangement of labeled boxes). User opinion on 'faithfulness' is scored on a scale 1 ('very dissatisfied') to 4 ('very satisfied') using crowd-sourced annotations collected via MTurk. For realism, images synthesized by each method are presented to participants, inviting selection of the most realistic. In both cases each task is annotated by 4 unique participants, and results tabulated for differing levels of participant consensus. We consider only results where 2 or more MTurkers agree *i.e*. there was consensus. Tab. 3 indicates our method is preferred for content faithfulness and realism, although faithfulness scores are low in general (circa 2.5 on the scale) this is in line with reported figures for other methods e.g. 1.57 for [16] as shown in [19].

4.4. Ablation studies

We explore the significance of each stage of our proposed architecture and training methodology, comparing SBIR and generation results on QuickDrawCOCO-92c using several ablated variants as shown in Tab. 5, alongside the QuickDrawCOCO-92c result for the full method. We explore three categories of ablation (A-C). Category A ablates key features of the proposed architecture. Category B ablates key training steps. Category C investigates if training a single model for both search and synthesis does not degrade the model's performance at either.

(A1) Without Transformer. Removes the attention modules from the model, aggregating the representations from the GNN using simple addition.

(A2) Without GNN. Substitutes the GNN for an fc layer that bridges the OLR into the Transformer module.

(A3) Without Positional Encoding. Retains the proposed architecture but removes our grid-based positional encoding from the Transformer module.

(B1) Without object-level pretraining. Skips the first stage of training the OLR.

(B2) Without shuffled-objects negatives. Does not use our synthesized negative samples in the contrastive loss.

(C1) No Contrastive Loss. Does not use contrastive loss. The results on this model are driven by the masks and boxes generation losses and object-level triplet loss only.

(C2) No Generation Losses. By removing the masks and boxes generation losses, this ablated model is only trained with search associated losses.

Removing either the GNN (A2) or the grid-based positional encoding (A3) degrades both tasks' performance significantly while the Transformer (A1) is shown to be helpful for generation (improves FID by 25%), and crucial for search.

Our synthesized negative examples (B2) improve search by 8% and are essential for generation to work. This follows the result of (C1) where removing the Contrastive Loss also diminishes generation performance; given that both of those are associated with search but also impact generation show that the strength of our multi-task model lies on the synergy between those tasks.

When we then look at (C2) that hypothesis is further proven, as removing the synthesis-related losses impact the search results as well with a drop of 6% in recall@1. Training for both tasks does not decrease performance, but boosts it meaningfully. We conclude that learning general representations requires the model to retain both visual and semantic information, which we achieve by mixing generation and feature learning losses.



(a) Sketch composition (b) Mixed domain composition (c) Posing of individual objects

Figure 8. Generating images with Scene Designer. The first 3 rows show compositions and sketches from SketchyCOCO whilst the last one shows those from QuickDrawCOCO-92c. There are three features shown: (a) sketch composition. (b) compositions that mix image and sketched objects. (c) individual object posing, where we flipped each object individually and compare the generations.

Method	SketchyCOCO						QuickDrawCOCO					
	Realism			Faithfulness			Realism			Faithfulness		
	C2	C3	C4	C2	C3	C4	C2	C3	C4	C2	C3	C4
Ours	**64.17**	**72.73**	**82.61**	2.67	2.50	**2.16**	**62.77**	**63.64**	**75.68**	**2.66**	2.56	2.58
Ashual *et al.* [16]	13.33	13.64	04.35	2.56	2.34	1.90	37.23	36.36	24.32	2.61	**2.58**	**2.88**
SketchyCOCO [19]	22.50	13.64	13.04	**2.79**	**2.65**	2.04	-	-	-	-	-	-

Table 3. User perceptual study (via MTurk) evaluating the generated images. Realness is a comparative score between the models while faithfulness is individually scored per method on continuous scale [1,4]. Results are thresholded for different levels of participant (C)onsensus, from 2 to 4 (out of 4) participant agreements.

Method	SketchyCOCO		QDCOCO-92c	
	FID↓	Acc.↑	FID↓	Acc.↑
Ours (sketch-based)	**130.87**	**63.46**	**76.64**	**65.47**
Ashual *et al.* [16]	170.40	56.20	103.14	50.69
SketchyCOCO [19]	198.17	29.14	-	-
Ours (image-based)	138.82	57.36	69.08	59.52
Ours (mixed-domain)	143.24	65.74	83.13	58.53
SPADE [13]	111.25	81.21	58.39	80.95

Table 4. Generation metrics when using samples from SketchyCOCO and QuickDrawCOCO-92c. Both quantitative metrics were computed on the overlapping test set across the three/two models for each dataset (137 samples on the former and 1300 on the later). A lower FID and higher Object Classification Accuracy represent a better result. We've also included the metrics for the SPADE generator, representing an upper bound for our model.

Model Settings	r@1	r@10	FID↓
(A1) W/o Transformer	07.20	20.62	102.79
(A2) W/o GNN	02.69	17.61	118.04
(A3) W/o positional encoding	14.45	45.87	122.01
(B1) W/o obj.-level pretraining	55.84	85.43	102.14
(B2) W/o synthesized negatives	54.12	81.16	128.58
(C1) No contrastive loss	45.48	74.67	129.88
(C2) No generation losses	55.84	81.34	129.98
Final model	**62.15**	**85.07**	**76.64**

Table 5. Ablation Studies, showing Recall@k and FID on QuickDrawCOCO. With each set of models, we want to show that (A) each component is necessary, (B) the training procedure aids performance and (C) the single model can multi-task well.

5. Conclusion

We introduced Scene Designer; a single unified model for searching and generating images using free-hand sketches of scenes. We developed a mixed-task learning framework with three levels of representation (OLR, CCR, FCR) using a hybrid GNN-Transformer architecture. Scene Designer learns to embed sketched and photographic scenes into a common space, producing latent representations for both synthesizing layouts from sketched scenes and for measuring compositional similarity between sketch and image scenes. The model is trained via an expanded dataset of sketch compositions and corresponding images (QuickDrawCOCO-92c); a secondary contribution of our work. We show that the combination of feature learning and generation losses aided by our novel take on contrastive learning for scenes are responsible for obtaining SotA performance at scene search and synthesis tasks. The ability to sketch an initial composition, and incorporate components of results into hybrid queries for synthesis (or further searches) creates a novel mechanism for interactively constructing scenes from digital image collections. Further work could explore this interactive model in creative practice, and perhaps explore how other facets of generative artwork (*e.g.* neural style transfer) might be incorporated into the framework for example to enable fine-grained control over the appearance of objects within the composition.

Acknowledgments

This work was supported by FAPESP (grants 2017/22366-8, 2019/02808-1, 2019/07316-0), CNPq Fellowship (304266/2020-5), and a charitable donation from Adobe Inc.

References

[1] Mathias Eitz, Kristian Hildebrand, Tamy Boubekeur, and Marc Alexa. Sketch-based image retrieval: Benchmark and bag-of-features descriptors. *IEEE Trans. Visualization and Computer Graphics*, 17(11):1624–1636, 2011. 2

[2] Rui Hu and John Collomosse. A performance evaluation of gradient field hog descriptor for sketch based image retrieval. *Computer Vision and Image Understanding*, 117(7):790–806, 2013. 2

[3] Fang Wang, Le Kang, and Yi Li. Sketch-based 3d shape retrieval using convolutional neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 1875–1883. IEEE, 2015. 2

[4] Yonggang Qi, Yi-Zhe Song, Honggang Zhang, and Jun Liu. Sketch-based image retrieval via siamese convolutional neural network. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 2460–2464. IEEE, 2016. 2

[5] Tu Bui, Leo Ribeiro, Moacir Ponti, and John Collomosse. Compact descriptors for sketch-based image retrieval using a triplet loss convolutional neural network. *Computer Vision and Image Understanding*, 2017. 2

[6] Qian Yu, Feng Liu, Yi-Zhe Song, Tao Xiang, Timothy M Hospedales, and Chen-Change Loy. Sketch me that shoe. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 799–807, 2016. 2, 3, 5, 6, 7

[7] Patsorn Sangkloy, Nathan Burnell, Cusuh Ham, and James Hays. The sketchy database: learning to retrieve badly drawn bunnies. *ACM Transactions on Graphics (TOG)*, 35(4):119, 2016. 2, 3, 5, 6, 7

[8] Tu Bui, Leo Ribeiro, Moacir Ponti, and John Collomosse. Sketching out the details: Sketch-based image retrieval using convolutional neural networks with multi-stage regression. *Computers & Graphics*, 71:77–87, 2018. 2, 3, 5, 6, 7

[9] Fang Liu, Changqing Zou, Xiaoming Deng, Ran Zuo, Yu-Kun Lai, Cuixia Ma, Yong-Jin Liu, and Hongan Wang. Scenesketcher: Fine-grained image retrieval with scene sketches. 2020. 2, 5, 6, 7

[10] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 2, 4

[11] P. Isola, J. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5967–5976, 2017. 2

[12] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2

[13] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2, 4, 6, 8

[14] Justin Johnson, Ranjay Krishna, Michael Stark, Li Jia Li, David A. Shamma, Michael S. Bernstein, and Fei Fei Li. Image retrieval using scene graphs. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 07-12-June, pages 3668–3678. IEEE, jun 2015. 2

[15] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image Generation from Scene Graphs. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1219–1228. IEEE, jun 2018. 2, 3

[16] Oron Ashual and Lior Wolf. Specifying object attributes and relations in interactive scene generation. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2019-Octob, pages 4560–4568, sep 2019. 2, 3, 4, 5, 6, 7, 8

[17] Wengling Chen and James Hays. Sketchygan: Towards diverse and realistic sketch to image synthesis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2

[18] Yongyi Lu, Shangzhe Wu, Yu-Wing Tai, and Chi-Keung Tang. Image generation from sketch constraint using contextual gan. In *The European Conference on Computer Vision (ECCV)*, September 2018. 2

[19] Chengying Gao, Qi Liu, Qi Xu, Limin Wang, Jianzhuang Liu, and Changqing Zou. Sketchycoco: Image generation from freehand scene sketches. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2, 5, 6, 7, 8

[20] Tao Chen, Ming-Ming Cheng, Ping Tan, Ariel Shamir, and Shi-Min Hu. Sketch2photo: Internet image montage. *Proc. ACM SIGGRAPH*, 28(5):124, 2009. 2

[21] Leo Sampaio Ferraz Ribeiro, Tu Bui, John Collomosse, and Moacir Ponti. Sketchformer: Transformer-based representation for sketched structure. In *Proc. CVPR*, 2020. 3

[22] Kaiyue Pang, Yi-Zhe Song, Tony Xiang, and Timothy M Hospedales. Cross-domain generative learning for fine-grained sketch-based image retrieval. In *BMVC*, pages 1–12, 2017. 3

[23] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 3

[24] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 3

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017. 4

[26] Longteng Guo, Jing Liu, Jinhui Tang, Jiangwei Li, Wei Luo, and Hanqing Lu. Aligning linguistic words and visual semantic units for image captioning. In Laurent Amsaleg, Benoit Huet, Martha A. Larson, Guillaume Gravier, Hayley Hung, Chong-Wah Ngo, and Wei Tsang Ooi, editors,

Proceedings of the 27th ACM International Conference on Multimedia, MM 2019, Nice, France, October 21-25, 2019, pages 765–773. ACM, 2019. 4

[27] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 4

[28] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, 2014. 4

[29] Xudong Mao, Qing Li, Haoran Xie, Raymond Y. K. Lau, and Zhen Wang. Least squares generative adversarial networks, 2016. cite arxiv:1611.04076. 4

[30] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations (ICLR)*, 2015. 5

[31] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6629–6640, Red Hook, NY, USA, 2017. Curran Associates Inc. 5, 6

[32] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Cocostuff: Thing and stuff classes in context. In *Computer vision and pattern recognition (CVPR), 2018 IEEE conference on*. IEEE, 2018. 5, 6

[33] The Quick, Draw! Dataset. https://github.com/googlecreativelab/quickdraw-dataset, 2018. Accessed: 2018-10-11. 5, 6

A.3 Scene designer: Compositional sketch-based image retrieval with contrastive learning and an auxiliary synthesis task.

This section includes the original published copy of the extended journal version of the Scene Designer paper, reproduced here with permission from Springer Nature. This extended paper was first published on Springer's Multimedia Tools and Applications (MTAP), Special Issue on Content-Based Image Retrieval:

RIBEIRO, L. S. F.; BUI, T.; COLLOMOSSE, J.; PONTI, M. Scene designer: Compositional sketch-based image retrieval with contrastive learning and an auxiliary synthesis task. **Multimedia Tools and Applications**, Springer, 2022.

Multimedia Tools and Applications
https://doi.org/10.1007/s11042-022-14282-0
1227: CONTENT-BASED IMAGE RETRIEVAL

Scene designer: compositional sketch-based image retrieval with contrastive learning and an auxiliary synthesis task

Leo Sampaio Ferraz Ribeiro[1] · Tu Bui[2] · John Collomosse[2,3] · Moacir Ponti[1,4]

Received: 11 May 2022 / Revised: 30 September 2022 / Accepted: 3 December 2022
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2022

Abstract
Scene Designer is a novel method for Compositional Sketch-based Image Retrieval (CSBIR) that combines semantic layout synthesis with its main task both to boost performance and enable new creative workflows. While most studies on sketch focus on single-object retrieval, we look to multi-object scenes instead for increased query specificity and flexibility. Our training protocol improves contrastive learning by synthesising harder negative samples and introduces a layout synthesis task that further improves the semantic scene representations. We show that our object-oriented graph neural network (GNN) more than doubles the current SoTA recall@1 on the SketchyCOCO CSBIR benchmark under our novel contrastive learning setting and combined search and synthesis tasks. Furthermore, we introduce the first large-scale sketched scene dataset and benchmark in QuickDrawCOCO.

Keywords Sketch-based image retrieval · Cross-domain representation learning · Image synthesis · Scene understanding

✉ Leo Sampaio Ferraz Ribeiro
leo.sampaio.ferraz.ribeiro@usp.br

Tu Bui
t.bui@surrey.ac.uk

John Collomosse
j.collomosse@surrey.ac.uk

Moacir Ponti
ponti@usp.br

1 ICMC, Universidade de São Paulo, São Carlos, SP, Brazil

2 CVSSP, University of Surrey, Guildford, Surrey, UK

3 Creative Intelligence Lab, Adobe Research, San Jose, CA, USA

4 Mercado Livre, Osasco, SP, Brazil

Published online: 20 December 2022

1 Introduction

As we've become used to having great cameras always in our pockets, the number of new images taken or made every day has skyrocketed. With the wealth of content the problem of retrieval becomes ever more pressing; it is natural then that research on Sketch-based Image Retrieval (SBIR) has similarly boomed within the last decade [5, 6, 33, 41, 51]. Instead of using text queries to search for images in SBIR one searches by drawing the desired object, which is more intuitive and flexible, allowing the user to express desired object pose and specific shapes without having to have a similar image at hand (as would be the case for the Content-based Image Retrieval task).

While booming, the field of SBIR has seldom explored retrieval beyond single-object sketches and images save for one recent publication [25]. While this limitation is acceptable for e-commerce retrieval, it impedes the use of SBIR for large scale general retrieval, creative workflows such as reference retrieval and doesn't even allow for search in personal image collections; in these scenarios, the photographs are rarely of single objects and instead represent complete scenes with multiple interacting objects and people depicted. We sought then to design a model for Compositional SBIR, enabling users to sketch scenes with multiple objects and choose their positions as well as shapes and poses.

This paper presents Scene Designer; a model for searching scenes using free-hand sketches of scene compositions. Our model provides a cross-modal search embedding where similarity of visual compositions comprising sketches and images (or mixtures of both) can be measured. The training task is comprised of a novel take on contrastive learning together with a generative task for synthesising scene layouts, leading to a retrieval performance that more than doubles the previous state-of-the-art (SoTA). Our technical contributions are:

1. **Compositional Sketch Search.** We propose a hybrid graph neural network (GNN) and Transformer architecture to learn a metric search embedding for comparing sketched and photographic scenes. Existing sketch based image retrieval (SBIR) methods predominantly match queries containing a single, dominant object invariant to its position within an image. Our novel contrastive training matches sketched compositions containing multiple objects.
2. **Fusion of Synthesis and Cross-Domain Search.** Training a single mixed-task model yields improvements – more than doubling recall@1 – on the state of the art for scene search whilst providing competitive scene synthesis results. We also enable a novel creative workflow in which image compositions may be created iteratively; users sketch an initial composition (one or more objects), and elements of the retrieved images may be incorporated potentially with further sketched objects to drive either further searches or full image synthesis. Figure 1 demonstrates how our fused model enables iterative design of generative compositions. We argue that this novel workflow shows the potential of combining both tasks beyond improving performance and can be applied to other unified models as well.
3. **New Large Scale Dataset of Sketched and Photographic Scenes.** We designed a new dataset to train our model, called QuickDrawCOCO. By synthetically combining the descriptive segmentation labels of COCO-stuff with the large Quick! Draw sketch dataset, we introduce the largest (by 8x) and most class-diverse (with 92 classes) dataset of its kind.

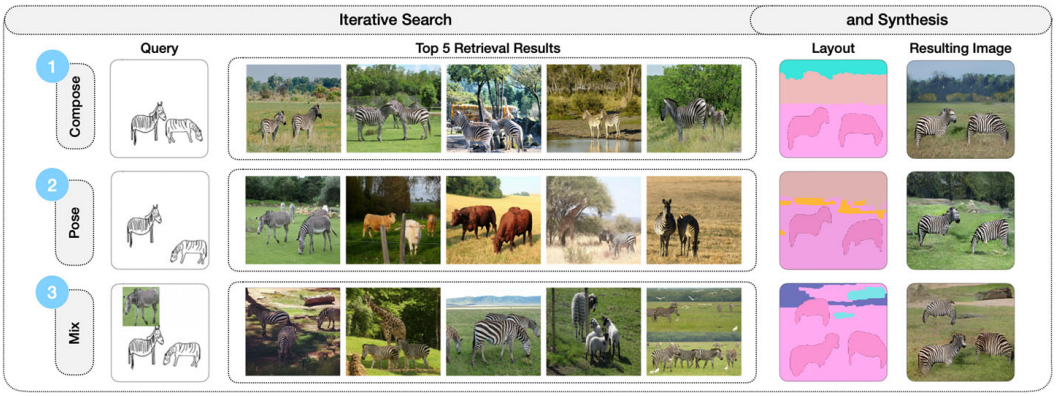


Fig. 1 Scene Designer enables iterative design of image compositions through retrieval and synthesis. In (1) the user sketches initial concept, matching scenes are retrieved; then (2) the user refines the search by reposing the objects and finally in (3) a retrieved object is used to add to the composition, mixing domains. For each retrieval step we also show the layout synthesised by Scene Designer and the resulting generated image

Furthermore, this work is a journal extension of the workshop paper on Scene Designer [38], where we first presented the concept of combining search and synthesis. This extension improves on the first publication with these novel contributions:

- **(I)** **New architecture, discussion and conclusions.** we propose a different architecture when compared to the previous paper including Simple Averaging and Self-attention as well as Transformer, improving results and significantly changing the discussion as well as reaching a novel conclusion.
- **(II)** **Backbone study** Different backbones for the object encoder are explored.
- **(III)** **Better performance.** By tuning the loss weights we've improved on the original model by 4% in recall@1 for both benchmarks.
- **(IV)** **Exhaustive Ablations.** Aside from new architecture alternatives, we have also expanded the existing ablation studies to cover both benchmarks.
- **(V)** **Improved Qualitative Comparisons.** We have provided new qualitative CSBIR samples that match those presented by [25], allowing for visual comparisons. We've also included matching layout synthesis samples from [3] to better place the performance of our model at the secondary generative task.

2 Related work

Sketch based image retrieval (SBIR) has received extensive attention in the past decade. Dictionary learning methods (e.g. bag of visual words) have leveraged wavelet, edge-let, shape-let and sparse gradient features [13, 20] to match sketches to edge structures in images. With the advent of deep learning, CNNs (convnets) were rapidly adopted for cross-modal representation learning; exploring joint search embeddings for matching structure between sketches and images. Early approaches learned mappings between edge maps and sketches using contrastive [51], siamese [33] and triplet networks [5]. Fine-grained SBIR was explored by Yu et al. [57] and Sangkloy et al. [41] who used a three-branch CNN with triplet loss. Bui et al. learned a cross-domain embedding through triplet loss and partial

weight sharing between sketch-image encoders [6]. Later studies followed a natural extension from single object sketches to scenes with multiple objects. Liu et al. [25] recently used a graph encoder with triplet loss to do Compositional SBIR, making this study the closest in task to ours, their model is however dependent on known category labels of photos and sketches, a dependency we amend in our design while also implementing our own take on a GNN encoder.

Conditional image generation has been a rapidly developing field since the introduction of Conditional Generative Adversarial Networks (GANs) [28]. Initial work exploring class label priors were soon followed by models with image based priors such as Pix2Pix [21] and more recently high definition synthesis models such as Pix2PixHD [52], SPADE [31] and OASIS [42]. Those models learn from pairs of matching samples from each domain, and are capable of mapping semantic layouts to images. We make use of one such model [31] to show how combining the tasks of search and synthesis can enable the new creative workflow seen in Fig. 1, where our model's semantic layout is used to drive the conditional generator.

Text-Conditioned Image Generators have been on the spotlight lately. Led by the combined leaps in large language models [4] and image synthesis with Diffusion Probabilistic Models [10, 45], unified text and image models such as CLIP [34], Dall-E [35] and Stable Diffusion [39] are being used even beyond research for creative applications online. Our model, while not competing in image quality, overcomes a shortcoming of these models: the lack of fine pose and placement control when using only text as input, requiring users to finetune prompts and even share "prompt tutorials" online.

Scene graphs are a representation for images where individual objects are defined as nodes with the graph edges describe their relationships. They were first used for text-based image retrieval [23] but were more recently used by Johnson et al. [22] and Ashual et al. [3] as initial representations to drive semantic layout generation and image synthesis based on those layouts. Our approach follows a similar principle for encoding scene graphs into object representations which then drive layout synthesis. The main difference of course is that we aggregate these vectors to represent the entire scene and perform the CSBIR task.

Image-synthesis from sketch is a challenging problem due to the abstract and ambiguous nature of most sketch applications; the application of facial sketch-photo synthesis for example is well developed [32, 53, 54] in the literature thanks to its application in forensics and the use of realistic and unambiguous sketches. Synthesis from abstract sketches on the other hand have been approached by recent methods such as Sketchy-GAN [9] and ContextualGAN [26], which achieved promising results generating images from single-object sketches, but as was shown in [14], cannot accommodate sketched scenes with multiple objects. Gao et al. [14] were the first to implement image synthesis from sketch scenes and introduced the paired SketchyCOCO dataset. We have used the task of layout synthesis from sketched compositions to drive better scene representations which in turn leads to improved CSBIR; our model does nonetheless generate semantic layouts that are aligned with the current SoTA.

Joint image search and synthesis has been tried in early work such as Sketch2Photo [8] that composes an output image from objects found in the retrieved results. In a more recent work, Sketchformer [37] also harvests the search and synthesis combination but only for

sketch (single-domain). Similarly, Pang et al. [30] uses a sketch decoder to assist their cross-domain learning, showing the usefulness of mixing both to yield better representations. This combination has also been used within the constraints of Zero-Shot SBIR [11, 12, 29, 44, 55], showing that including a synthesis task helps with model generalisation.

Closest to our suggested creative workflow is the work of Yu & Kovashka [56], where the generated images are used to gather user feedback and improve the retrieval result; their method however is only text-conditioned. Our method learns a cross-domain embedding for scene sketches and images that is improved upon by the addition of layout synthesis in the framework; furthermore we show that images generated from our layouts are comparable to synthesis-only models.

3 Methodology

We developed a cross-modal representation learning framework in order to represent images and sketches with multiple objects in a common feature space; this representation is used to drive both cross-modal retrieval and layout generation. Our model learns representations for individual objects first, introduces scene-level information to these representations, and then finally aggregates those into a vector representing the entire scene. Figure 2 describes the architecture and stages of the representation, which are each summarised below:

1. **Object-Level Representation (OLR):** Given an input composition, we encode individual objects (which might be sketched or photo-real) to a common representation. We refer to this embedding, in which each object is independently represented, as the OLR (Section 3.1).
2. **Scene Graph (SG)**: is constructed using the OLR to encode objects, along with the discrete positional relationships of all object pairs.
3. **Scene-Aware OLR (SA-OLR):** A graph neural network (GNN) re-encodes object appearances (the OLR, nodes in the Scene Graph) together with their pairwise relationships (Section 3.2).

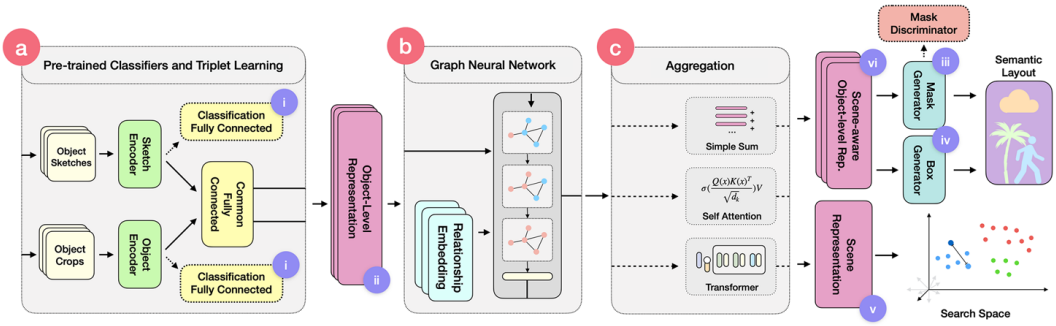


Fig. 2 Proposed architecture of Scene Designer. At stage (a) individual objects of each scene are encoded independent of composition to produce the *object-level representation*. A Scene Graph is constructed from these and encoded via GNN (b) to produce *scene-aware object-level representations*, which are subsequently aggregated (c) with the use of simple sum, self-attention or a small transformer to produce the *scene representation*. The roman numerals indicate where each of the losses (see Sections 3.1, 3.4 and 3.5) are applied during training: (i) Cross-entropy classification $\mathcal{L}_{CCE}$, (ii) Triplet loss $\mathcal{L}_{tri}$, (iii) Mask generation losses $\mathcal{L}_{G_m}$, (iv) Bounding box generation losses $\mathcal{L}_{G_b}$, (v) Contrastive loss $\mathcal{L}_{cont}$, (vi) Final cross-entropy classification $\mathcal{L}_{CCE_f}$

4. **Scene Representation (SR):** Each object in a scene is now represented by one SA-OLR, to obtain the unified Scene Representation those vectors are aggregated by a function $r(.)$. We've explored simple averaging, self-attention [47] and a small Transformer network as options for $r(.)$, opting for the latter option as the most robust one (Section 3.3). This single-vector representation is driven by metric learning and is later used for retrieval. (Section 3.4)

As input, our model takes for each scene the individual sketched/photographic objects and their positions (which can be coarsely defined). We now describe in further detail how each stage of the representation is learned.

3.1 Object-Level Representation (OLR)

We begin by independently encoding each object within the input composition to a common feature embedding, regardless of its input modality. Following contemporary single-object SBIR [6, 41, 57] we adopt a deep metric learning approach using a heterogeneous triplet architecture (i.e. no shared weights between the anchor (a) and positive/negative (p/n) branches). Our network comprises a MobileNet [19] backbone (Section 5), terminated with two shared (as opposed to the heterogeneus backbone) fully connected (fc) layers, the latter yielding the 256-D OLR embedding.

The MobileNet is pre-trained on ImageNet [40], and finetuned for cross-domain learning using a combined triplet and categorical cross-entropy (CCE) losses, with the former defined as:

$$\mathcal{L}_{tri}\,(a,\,p,\,n) = \max\{0,\, m + \|f_s(a),\, f_i(p)\|_2 - \|f_s(a),\, f_i(n)\|_2\} \tag{1}$$

where $f_s(.)$ is the MobileNet that encodes sketched objects, $f_i(.)$ the one that encodes photo-real objects cropped from images, each followed by the shared fc layers that encodes the MobileNet's outputs to the OLR, $m = 0.5$ is the margin and $\|.\|_2$ is the L_2 norm. Hereafter, we use shorthand $f(.)$ to refer to the encoder irrespective of its modality. The OLR is trained using rasterized sketches (a), and objects cropped from the COCO-stuff dataset with corresponding (p) and differing (n) object class; see Section 4.1 for dataset details. To aid convergence, the equal-weighted CCE loss $CCE(.)$ is applied to a further fc layer $f_e(.)$ appended to the backbone networks:

$$\mathcal{L}_{cce}(a,\,p,\,n) \;=\; \sum_{c\in\{a,p,n\}} CCE\left(f_e(c), \hat{c}\right) \tag{2}$$

where $\hat{a}$, $\hat{p}$, $\hat{n}$ are the one-hot vectors encoding the semantic class of each object.

3.2 Scene-Aware Object-Level Representation (SA-OLR)

A scene graph (SG) describes objects and their spatial relationships within the composition. Formally an SG is a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{o_1, \cdots, o_i, \cdots, o_\kappa\}$ is set of nodes representing the κ objects, and $(o_i, r_{ij}, o_j) \in \mathcal{E}$ the set of edges encoding discrete relationships, $r_{ij} \in \mathcal{R}$ (we used "left of", "right of", "above", "below", "contains", "inside of").

We encode the scene graph via a graph neural network (GNN). We represent each node via the OLR as $f(o_i)$, encoding the pose and appearance of the object. As in [3], we adopt a learnable embedding $f_r(r_{ij})$ to map relationships $\mathcal{R}$ into dense vectors. The GNN $g_r(.)$

comprises a sequence of 4 layers $g_r^k(.)$. In each we follow the architecture of [22], and edges are processed by an fc layer f_{fc1}^k:

$$\begin{aligned} \left\langle v_i^0, r_{ij}^0, v_j^0 \right\rangle &= < f(o_i), f_r(r_{ij}), f(o_j) > \\ \left\langle \hat{v}_i^k, r_{ij}^{k+1}, \hat{v_j}^k \right\rangle &= f_{fc1}^k \left(\left\langle v_i^k, r_{ij}^k, v_j^k \right\rangle \right) \end{aligned} \tag{3}$$

that updates the vector for each relationship to r_{ij}^{k+1} and builds the set $\hat{v}_i^k \in V^k$ of object vectors. A function (h) gathers vectors in $\mathcal{V}$ that describe the same object and averages them; finally they are processed by another fc layer to compute the updated $v_i^{k+1} = f_{fc2}^k(h(V^k))$ for each object. These layers are shared between images and sketches and trained end-to-end as part of our model. We call the output of this module the *Scene-aware* Object-level Representations (SA-OLR), the 256-D vectors that describe each individual object in a scene.

3.3 Scene Representation (SR)

To obtain a final representation for each scene, the individual Scene-aware Object-level Representations need to be aggregated into a single vector. Here is where our new architecture differs the most from the previous version presented in [38] by including 2 new aggregation alternatives. We consider three options for aggregation: (a) Simple Averaging, (b) Self-attention or (c) a small 3-layered Transformer.

3.3.1 Simple averaging

With Simple Averaging, the Scene-aware Object-level Representations (SA-OLR), $g_r(f(o_i))$, are averaged per scene. Given a scene graph $\mathcal{G}$ with set of nodes $\mathcal{V} = \{o_1, \cdots, o_i, \cdots, o_\kappa\}$, we compute:

$$r(\mathcal{G}) = \frac{\sum_{i=0}^{\kappa} g_r(f(o_i))}{\kappa} \tag{4}$$

to obtain the Scene Representation for $\mathcal{G}$.

3.3.2 Self-attention

Self-attention can be considered a variant of the averaging solution, except here we use the obtained attention matrix to perform a weighted-average of the SA-OLR vectors from each scene.

We follow Sukhbaatar et al. [46] in our design. The attention matrix is obtained by using two learnable projection matrices W and V:

$$A = \sigma((\mathcal{G}_r W)V) \tag{5}$$

where W performs simple projection whilst V aggregates the resulting square matrix into the $A = \{a_1, \cdots, a_i, \cdots, a_\kappa\}$ attention weights; σ is the softmax function that makes the resulting attention weights behave as a probability function that weights the $g_r(f(o_i))$ values:

$$r(\mathcal{G}) = \sum_{i=0}^{\kappa} a_i g_r(f(o_i)) \tag{6}$$

3.3.3 Transformer

We show in Section 5 how Simple averaging or a Self-Attention layer are both simpler to implement yet competitive options. For other experiments we have however chosen to use our third option, the small 3-layered Transformer [50] encoder, for its improved robustness. This approach was inspired by the use of LSTMs for single vector SG representation in [16]. Since they are known to scale with sequence size, our version adds little computational cost (as the maximum number of objects in a scene is kept at 8), and aside from aggregating, they naturally provide further refinement of the SA-OLR.

Furthermore, Transformers employ positional encoding, providing a good information entry-point for the objects' positional information. We adapt traditional sequence-based positional encoding to scenes, following [3], we break the scene into a 5x5 grid and attribute a position $p \in 0, 1, ..., 24$ to each object based on where its centre falls on the grid. This p is used to compute its positional encoding (see Fig. 3).

We specify our Transformer $t(.) = Z$ with 3 layers and 16 attention heads on each. The input is the SA-OLR $g_r(f(o_i))$ for each object in the image and one empty vector $\vec{0}$ for aggregation. In each layer $t^i(.)$ the model computes the attention weights that relate each vector to all the vectors in the sequence:

$$\begin{aligned} Z^0 &= \sigma\left(F_q^0(Q+E)F_k^0(Q+E)^T\right)F_v^0(Q+E) \\ Z^{i+1} &= \sigma\left(F_q^i(Z^i)F_k^i(Z^i)^T\right)F_v^i(Z^i), \end{aligned} \tag{7}$$

where multiplications are matrix-based, σ is the softmax function, F_q^i, F_k^i, F_v^i are fc layers, Q is a matrix made by stacking $\vec{0}$ and $g_r(f(o_i))$ SA-OLR vectors and E is our positional encodings $f_p(p)$, stacked in the same fashion as Q. In the output, the position that contained $\vec{0}$ has the Scene Representation $r(\mathcal{G})$ (SR) while the other ones contain further processed SA-OLR for each object. When using the Transformer for aggregation, we use those new SA-OLR as the SA-OLR for the steps that follow.

3.4 Compositional sketch-based image retrieval

The single *Scene Representation* (SR) is used for contrastive learning, designed as a search embedding in which the similarity of images and the sketched input may be measured.

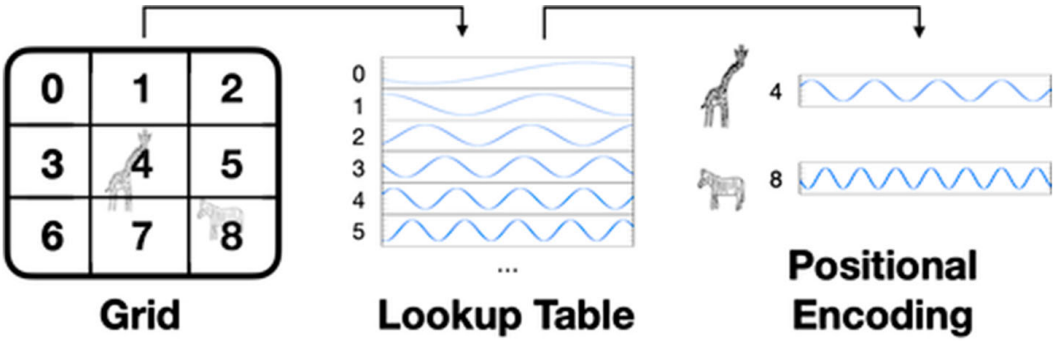


Fig. 3 Grid-based Positional Encoding. We adapted the positional encoder from traditional Transformers by attributing one encoding to each block of the gridded scene

During contrastive training, we sample half of our negatives randomly from other images of scenes in the training set. The other half of the negatives are synthesised by swapping objects of different class into the positions of the objects in the positive image (Fig. 4). By including such a class-swapped version of the image as a negative, the model is encouraged to discriminate change in object class to a greater degree than changes in object positions. Our contrastive loss, adapted to incorporate these synthetic negatives, is:

$$\mathcal{L}_{cont}(x_s, x_i, x_{sn}) = \frac{1}{2}(Y)(\mathcal{D}(x_s, x_i)) + \frac{1}{4}(1-Y)(1-\mathcal{D}(x_s, x_i)) - \frac{1}{4}\mathcal{D}(x_s, x_{sn}), \tag{8}$$

where $\mathcal{D}$ computes $\|.\|_2$ between all vectors on one set and all vectors on another set, x_s and x_i are respectively the sketch-based and image-based SR, Y is a label indicating if a pair of vectors are from the same scene. Finally, x_{sn} is the SR of the synthesized scenes with swapped objects.

3.5 Semantic scene layout generation

To further improve the scene representation learning (see Section 4.4) and enable novel use cases we incorporate Semantic Scene Layout generation as an auxiliary task. Given the SA-OLR as input we train generators for bounding boxes and masks.

The box generator is trained using Generalized Intersection-over-Union (GIOU) [36] and mean-squared error (MSE). The generator itself is a 2-layer MLP. The mask generator is trained within a GAN [15] framework, using a conditional setup [28] with LSGAN [27] losses plus MSE (to regularize the adversarial training). Additionally, another CCE classification loss $\mathcal{L}_{CCE_f}$ is applied to the SA-OLR to encourage it to remain semantic.

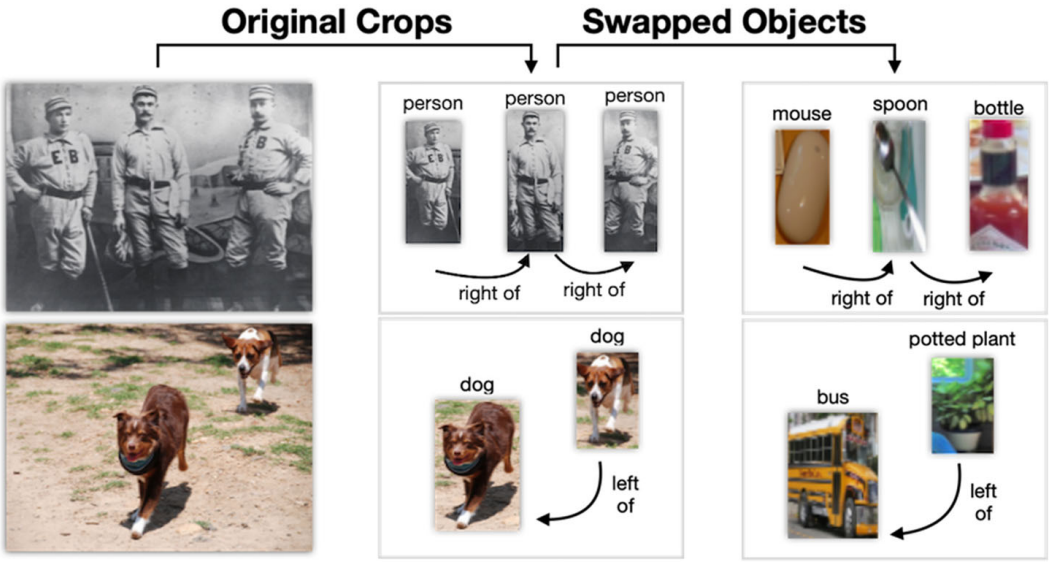


Fig. 4 During contrastive training of the SR, negatives are synthesized by swapping objects in the scene for others of differing class. This helps the model prioritize class over spatial relationships. First column shows the original image, the second the individual cropped objects, and the third the swapped objects

Taking b as ground truth bounding box, $\hat{x}$ as the SA-OLR of each object, our generation losses are, for bounding boxes:

$$\mathcal{L}_{G_b}(x) = \lambda_1 \mathcal{L}_{GIOU}(b, G_b(\hat{x})) + \lambda_2 \|b, G_b(\hat{x})\|_2 \tag{9}$$

for each object on each scene, with $\lambda_1 = \lambda_2 = 10$. Using a similar notation, take our ground truth masks as m_g, and the object labels as $y \in C$, where C is the set of object classes. The loss for the mask generator is:

$$\mathcal{L}_{G_m}(\hat{x}) = \lambda_3 \|m_g, G_m(x)\|_2 + \lambda_4 \|D(G_m(\hat{x}), y), 1\|_2 \tag{10}$$

with D as the discriminator. The weights are $\lambda_3 = 10$, $\lambda_4 = 0.5$. D itself is trained with the adversarial loss in classic GAN fashion. Those losses back-propagate through all representation levels, and are applied to both sketched and photographic scenes.

Having a trained model, at inference time we turn the masks and boxes into a semantic layout. In this object-only layout, objects are placed ordered by their size, so that bigger objects are behind smaller ones (after [3]); then it may be combined with a layout for the background (e.g. selected from the top search results or coarsely drawn). The layout is not used for training and for image generation inference it is passed through a SotA SPADE model [31].

3.6 Training stages

Training is performed in three stages. First the OLR is trained independent of the rest of the model, using the dual triplet and categorical cross-entropy loss, for 100K iterations, on single-object sketches and object crops. In the second stage we use our novel soft-paired sketch and image scenes dataset (see Section 4.1) to train entire model end-to-end for a further 120K iterations. Finally, we finetune the model with a hard-paired dataset (See Section 4.1), training for a further 10K iterations. Training is via the ADAM [24] optimizer, with $\beta_1 = 0.5$ and $\epsilon = 1e-9$; learning rate (lr) is $lr = 10^{-4}$ for the first two stages and $lr = 10^{-5}$ for the finetuning stage. The mask discriminator follows the *Two Time-Scale Update Rule* (TTUR) [18] with $lr_D = 4lr$. Batch size is kept at 16.

4 Experiments and discussion

We evaluate the performance of Scene Designer for both compositional sketch search (SBIR) and synthesis, contrasting performance against contemporary baselines for both tasks. We've chosen to evaluate the auxiliary generative task as well as the main CSBIR one to demonstrate that retrieval is not detrimental to generation, with the latter remaining competitive even if it doesn't greatly improve upon the SoTA in layout synthesis. This section will also show that, on the other way around, the synthesis task greatly improves CSBIR results beyond SoTA.

For CSBIR we compare against four baselines: a scene-level technique (SceneSketcher [25]) and three single-object techniques. Of these, two are fine-grained SBIR models (Sketchy [41], and Sketch-me-that-shoe [57]), and one is coarse-grained (Multi-stage Learning or MSL [6]). For a fairer comparison, we set our embedding dimension to 256-D, same as MSL, which has the lowest dimension among these baselines. For scene synthesis we compare against the sketch driven method of Gao et al. [14] (proposed alongside their SketchyCOCO dataset) and the method of Ashual et al. [3] that accepts semantic scene graphs (spatial arrangements of keywords) as input. To fully show the versatility of

our model's input, we will explore sketch, image and mixed domain compositions in the experiments.

4.1 Datasets

We require paired data; sketched compositions and corresponding images of the scenes depicted by those sketches.

SketchyCOCO is a recent dataset of 14K such pairs, proposed for sketch based scene synthesis [14]. SketchyCOCO samples scenes and associated annotations (bounding boxes, masks) from the COCO-stuff [7] images dataset, augmenting these with sketched depictions of those scenes. Approximately 4K of SketchyCOCO images contain object instances (the remainder are solely background material; 'stuff'). Only 14 classes (14c) of objects are annotated. To mitigate overfitting, we use the SketchyCOCO training partition (4,060 samples) only for fine-tuning our model. We include the test partition in our evaluation and split it per experiment requirements: retrieval eval. uses 233 samples as [25] needs scenes with 2 or more objects; generation tests needs to use the overlap between test sets used in each of the compared models: 137 samples. We also emulate [25] in constructing an 'Extended SketchyCOCO' search corpus by holding out 5K random images from the COCO-stuff training set and adding these to the SketchyCOCO test set as distractors.

QuickDrawCOCO-92c is a novel dataset we propose, composing scenes with sketches from QuickDraw, the largest public sketch dataset (50M) [49]. Inspired by SketchyCOCO, we similarly leverage scenes from COCO-stuff, taking advantage of the class overlap (92c) with QuickDraw. We synthesise sketch scenes by compositing sketches from QuickDraw onto a canvas, selecting a random sketch within the class corresponding to each object in a COCO-stuff image (Fig. 5). Our dataset is soft-paired, in that the sketches for each object do not necessarily match pose and appearance with the image, only the class label. We use QuickDrawCOCO-92c for all training stages except for the final finetuning stage where

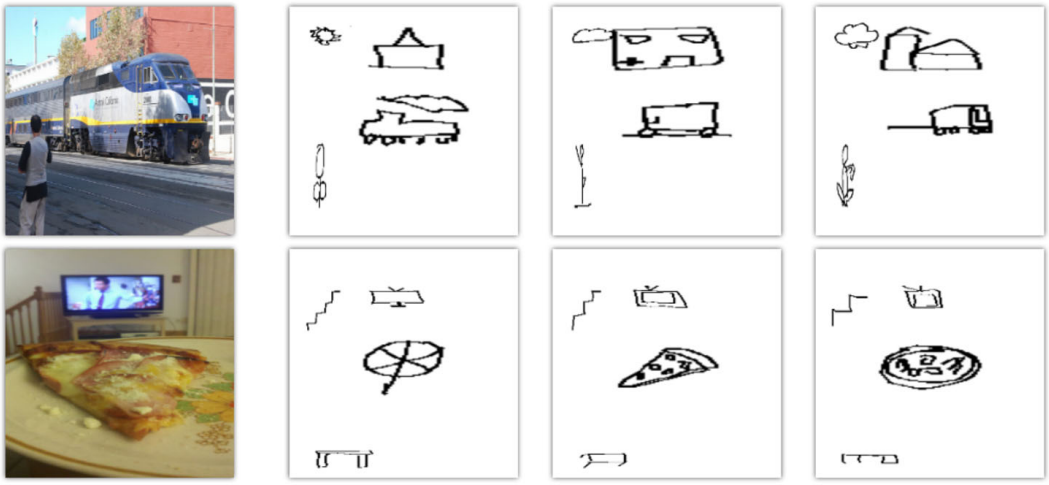


Fig. 5 Sample paired data from the proposed QuickDrawCOCO-92c dataset. Each row shows one COCO-stuff [7] image and three examples of synthetic sketched compositions matching that image derived from QuickDraw [49]

we combine samples from both SketchyCOCO and QuickDrawCOCO-92c. We separately fuse the training and test partitions of the QuickDraw and COCO-stuff datasets to partition QuickDrawCOCO-92c into 111,112 training, 2,788 test and 1,907 validation scenes. This dataset is 6x more category-diverse and has 8x more image samples than SketchyCOCO; furthermore, since scene sketches are synthesised, each image scene has thousands of possible sketch compositions, making this the most diverse sketch dataset to date. See Table 1 for comparisons between existing scene datasets.

4.2 Evaluating retrieval

We adopt the evaluation proposed in SceneSketcher [25], the only prior technique exploring SBIR for scenes; retrieved images are relevant only if they match the sketched scene exactly. Models are evaluated on SketchyCOCO, QuickDrawCOCO-92c and 'Extended SketchyCOCO'.

Compositional search Table 2 reports recall at rank $k = \{1, 5, 10\}$ (R@k) for our proposed search embedding for sketched scenes, that of SceneSketcher [25], and three single-object SBIR baselines: Sketch-me-that-shoe [57]; Sketchy [41]; and MSL [6]. We report R@k over SketchyCOCO and Extended SketchyCOCO for all methods; figures for SceneSketcher and Sketch-me-that-shoe are taken from Liu et al. [25]. We are unable to report values for these two models over our QuickDrawCOCO-92c test data, due to the lack of public models/code for these methods. QuickDrawCOCO-92c is a more challenging query set due to its increased size and the more messy / abstract sketches within QuickDraw. We present our results alongside public Sketchy [41] and MSL [6] models (final column, Table 2).

Our model more than doubles previous SotA recall results and greatly improves other metrics. The penultimate row of Table 2 reports performance our method using mixed-domain queries, where we substitute half of the sketched objects for their image cropped counterparts, which yields an even higher result likely due to half of the objects sharing the same domain as the test corpus. The final row reports on our model trained on QuickDrawCOCO-92c only; prior to finetuning on SketchyCOCO training set. This demonstrates that our model generalises sufficiently to beat the SotA on SketchyCOCO without explicit training on it. In Fig. 6 we show visual query results for both datasets (a, b) and in (c) we've tested the same queries presented in Fig. 5 of the SceneSketcher [25] paper, allowing for a side-by-side comparison (their models and code are not public).

Single object search We evaluate the performance of our model at single-object retrieval, baselining against both the Sketchy and MSL models on the SketchyCOCO dataset. We sample 120 queries randomly from SketchyCOCO containing only one object. In contrast to composition search, many relevant results exist in the SketchyCOCO dataset for a given sketched object. We consider a result relevant if both its category and pose match the sketched query. Since no ground-truth annotation with these criteria is available for SketchyCOCO, we crowd-source per-query annotation via Mechanical Turk (MTurk) for the top-k (k=15) results and compute both recall@k and precision@k in Table 3. For each ranked result we ask 5 MTurk participants to indicate relevance (or not), and filter on consensus level. Our approach significantly outperforms both baselines at all consensus levels. The sensitivity to object orientation and pose underlines the utility of sketch over labelled boxes (per [3]) to specify the appearance of the desired object.

Table 1 Comparing existing datasets against our QuickdrawCOCO

Dataset	# Images	# Sketches	Paired?	Annotation	# Classes	Multi-object
COCO-stuff	164K	–	No	Segmentation	171	Yes
Quick Draw!	–	25.8Mi	No	Labels	345	No
SketchyCOCO	14K	14K	Yes	Segmentation	14	Yes
SketchyCOCO (With Obj.)	4K	4K	Yes	Segmentation	14	Yes
SKY-Scene	7K	7K	Yes	Segmentation	30	Yes
TUB-Scene	7K	7K	Yes	Segmentation	35	Yes
QuickDrawCOCO (ours)	116k	116k*	Yes	Segmentation	92	Yes

We specify SketchyCOCO (With Obj.) to refer to the subset of SketchyCOCO scenes that have any object in them, the remainder of SketchyCOCO contains only background instances. *Since the sketch scenes are synthesised using COCO-stuff specs and Quick Draw! sketches, the number of distinct sketch scenes is nearly unlimited, but restricted to the scene layouts in COCO-stuff

Table 2 Recall@K metrics on the SketchyCOCO dataset, its extended version and the QuickDrawCOCO-92c set

Method	SketchyCOCO			Ext. SCOCO			QuickDrawCOCO		
	r@1	r@5	r@10	r@1	r@5	r@10	r@1	r@5	r@10
Ours	**79.39**	**96.99**	**99.57**	**64.37**	**92.27**	**94.42**	**65.98**	**83.38**	**88.26**
Sketch-me-that-shoe [57]	06.19	17.15	32.86	< 0.01	< 0.01	01.90	–	–	–
Sketchy [41]	03.43	09.87	15.87	< 0.01	00.85	00.85	00.07	00.35	00.60
MSL [6]	09.44	28.75	43.34	05.15	15.87	17.59	00.10	00.53	01.07
SceneSketcher [25]	31.91	66.67	86.19	12.38	26.67	38.10	–	–	–
Ours (Mixed-domain)	82.40	98.28	99.57	59.65	79.39	85.83	75.22	89.45	94.43
Ours (Before finetuning)	36.90	73.81	89.69	29.18	60.08	76.82	66.12	83.60	88.33

Our model more than doubles the previous state-of-the-art recall@1 metric. Additionally, we show our performance when using mixed-domain compositions and also with the model trained only with QuickDrawCOCO-92c, before finetuning on SketchyCOCO.
Best results are in bold



Fig. 6 Retrieval results using the SketchyCOCO (a) and QuickDrawCOCO sets (b). The results often contain the exact match and are contextually consistent, even when the model doesn't retrieve perfect class matches it does retrieve results that visually match the sketch (see last row on (b), where the flower is confused with food and the vase with cups). To allow for qualitative comparison with SceneSketcher [25], in (c) we've repeated the same queries of their qualitative evaluation on SketchyCOCO (Fig. 5 on their paper). Relevant results in green

Table 3 MTurk sketch search results for the proposed and baseline methods on the SketchyCOCO dataset

Method	Precision@1			Precision@15			Recall@1			Recall@15		
	C2	C3	C4	C2	C3	C4	C2	C3	C4	C2	C3	C4
Ours	**66.12**	**52.07**	**37.19**	**22.98**	**16.14**	**10.30**	**66.12**	**52.07**	**37.19**	**96.69**	**90.91**	**71.90**
MSL [6]	16.53	14.05	09.09	10.41	07.00	04.41	16.53	14.05	09.09	84.30	72.73	56.20
Sketchy [41]	35.54	23.97	18.18	17.36	11.79	07.77	35.54	23.97	18.18	85.95	80.17	65.29

Precision@k and Recall@k are presented at different (C)oncensus thresholds
Best results are in bold

4.3 Evaluating layout synthesis

While our model generates semantic layouts instead of images, we study the later as it has quantitative metrics available, it allows for comparison with the model closest in task to us (SketchyCOCO [14]) and is a good representation of applying our proposed creative workflow in a real scenario. Our goal is to show that more than improving retrieval results (see Section 4.2), our synthesised layouts lead to generated images comparable to SoTA; we also highlight that as Layout to Image Synthesis improves, the of-the-shelf model can be hot swapped in an application scenario.

We compare against that SketchyCOCO's accompanying public model [14], and against the public model of Ashual et al. [3] which synthesises images from spatial word maps i.e. implicit scene graphs. We synthesise from the sketch, and use the paired image as ground truth for evaluating the fidelity of the output. For Ashual et al., we create an input scene graph based on object classes and bounding boxes.

Objective metrics We compute Fréchet Inception Distance (FID) [18] and Object Classification Accuracy [3] to compare the synthesized and ground truth images. A lower FID value indicates that the tested image distribution is closer the original and is more realistic. The Accuracy values come from finetuning a ResNet-101 classifier on crops from COCO and applying it to foreground objects cropped from the generated images.

Table 4 shows our model to score higher on FID and lose to Ashual et al. work in Accuracy; we hypothesise the loss in the second metric might be due to our model adhering more closely to the sketch pose and therefore representing more unorthodox positioning. We've tried training SketchyCOCO from scratch on QuickDrawCOCO but the results were not satisfactory due to their model's reliance on matching poses; we've opted then to omit their model from this set's evaluation.

We have included results applying SPADE to the ground-truth layout, which serves as an upper bound generation result for our method. Results from mixed-domain and images-only compositions, which are a unique capability of our model, are also included. These show the value in applying Scene Designer to incrementally build up a composition as part of an interactive creative process.

Table 4 Generation metrics when using samples from SketchyCOCO and QuickDrawCOCO-92c

Method	SketchyCOCO		QDCOCO-92c	
	FID↓	Acc.↑	FID↓	Acc.↑
Ours (sketch-based)	**145.81**	55.32	**77.77**	33.80
Ashual et al. [3]	172.94	**64.67**	100.75	**45.69**
SketchyCOCO [14]	198.17	29.14	–	–
Ours (image-based)	138.82	55.83	63.58	42.40
Ours (mixed-domain)	143.24	66.66	78.80	37.42
SPADE [31]	111.25	83.75	48.51	64.07

Both quantitative metrics were computed on the overlapping test set across the three/two models for each dataset (137 samples on the former and 1708 on the later). A lower FID and higher Object Classification Accuracy represent a better result. We've also included the metrics for the SPADE generator, representing an upper bound for our model.
Best results are in bold

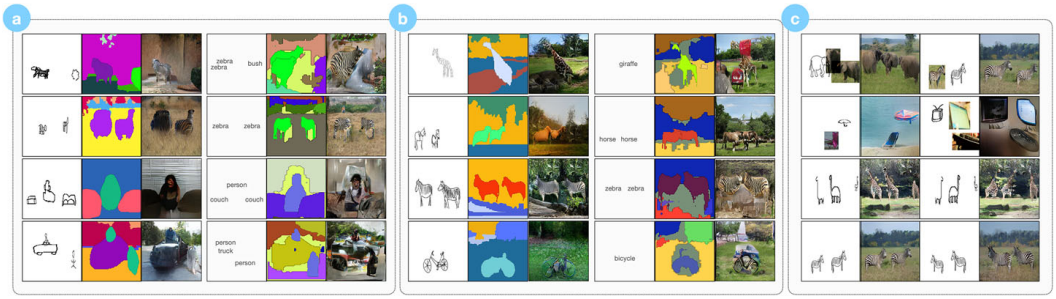


Fig. 7 Comparison between layouts generated by Scene Designer and model of Ashual et al. [3]. In our model, the sketched compositions drive the layout generation, which is then used with an off-the-shelf SPADE [31] model to synthesise images. In Ashual et al.'s model, the model takes in classes, positions and sizes and generates a layout; furthermore, they have their own layout-to-image generator. In (a) we compare ours and their layouts from QuickdrawCOCO, (b) from SketchyCOCO and in (c) we show our model's ability to combine objects from both domains and show its sensibility to sketched pose (by showing generation after flipping the sketches). We made the representative word maps for [31] by hand, using original positions as guides

Qualitative visual results We've also compiled visual results from ours and Ashual et al.'s model. In Fig. 7(a, b) we show the layouts from both models; the comparison highlights the advantages of using sketches instead of classes for the compositions as it gives the user control over the poses. Figure 7(c) further highlights the flexibility of our design, showing mixed compositions and the effects of flipping individual objects in the compositions.

4.4 Ablation studies

We explore the significance of each stage of our proposed architecture and training methodology, comparing SBIR and generation results on QuickDrawCOCO-92c using several ablated variants as shown in Table 5, alongside the QuickDrawCOCO-92c result for the full method. We explore three categories of ablation (A-C). Category A ablates key features of the proposed architecture. Category B ablates key training steps. Category C investigates if training a single model for both search and synthesis does not degrade the model's performance at either.

Table 5 Ablation studies, showing Recall@k and FID on QuickDrawCOCO and SketchyCOCO

	QDCOCO			SketchyCOCO		
Model Settings	r@1	r@10	FID↓	r@1	r@10	FID↓
(A1) W/o Transformer	47.59	77.21	59.78	47.63	91.84	132.12
(A2) W/o GNN	00.28	03.17	90.68	03.43	16.30	165.99
(A3) W/o positional encoding	43.87	76.67	57.69	43.34	90.12	134.01
(B1) W/o obj.-level pretraining	62.36	87.28	57.18	63.09	97.42	133.05
(B2) W/o synthesized negatives	57.67	85.22	54.31	69.52	98.28	126.00
(C1) No contrastive loss	40.17	72.09	53.05	50.64	89.69	129.46
(C2) No generation losses	57.20	84.29	121.17	57.08	96.56	249.88
Final model	65.98	88.26	55.46	79.39	99.57	133.22

With each set of models, we want to show that (A) each component is necessary, (B) the training procedure aids performance and (C) the single model can multi-task well

(A1) **Without Transformer.** Removes the attention modules from the model, aggregating the representations from the GNN using simple addition.
(A2) **Without GNN.** Substitutes the GNN for an fc layer that bridges the OLR into the Transformer module.
(A3) **Without Positional Encoding.** Retains the proposed architecture but removes our grid-based positional encoding from the Transformer module.
(B1) **Without object-level pretraining.** Skips the first stage of training the OLR.
(B2) **Without shuffled-objects negatives.** Does not use our synthesised negative samples in the contrastive loss.
(C1) **No Contrastive Loss.** Does not use contrastive loss. The results on this model are driven by the masks and boxes generation losses and object-level triplet loss only.
(C2) **No Generation Losses.** By removing the masks and boxes generation losses, this ablated model is only trained with search associated losses.

As expected, removing the GNN (A2) degrades both tasks' performance significantly, as the Transformer alone is not capable learning the scene representation without the graph information. The addition of the Transformer and its positional encoder are shown to improve the model significantly on both tasks.

Our synthesised negative examples (B2) improve search by 8% while pre-training the OLR encoders gives a smaller 3% boost.

Looking at the C group we can see that the combination of search and synthesis is the sensible choice for this and maybe future retrieval work; (C1) shows that even when driven only by generation learning, the scene representation is able to beat SotA in Compositional SBIR (compare against Table 2) and (C2) highlights the 8% boost to recall@1 given to retrieval performance when trained together with a synthesis task.

5 Design choices

We sought to consciously design each aspect of our model, and in this section we give support to some of those choices through more experiments.

First we compare the choice of ImageNet-pretrained backbones for the OLR encoders, in Table 6 we compare MobileNet, InceptionV3 and ResNet50 and show that the first not only yields the best performance but also has a reduced computational cost profile than the alternatives.

Then we study our choice of GNN with a Transformer for aggregation against other options. Table 7 compares alternative models that use either a standalone Transformer or a

Table 6 Choosing the best backbone for the encoders

		QDCOCO			SketchyCOCO		
Backbone	Par. (M)	r@1	r@10	FID↓	r@1	r@10	FID↓
MobileNetV2 [19]	17.62	65.98	88.26	55.46	79.39	99.57	133.22
InceptionV3 [48]	58.86	61.21	85.55	55.81	77.25	100.0	137.82
ResNet50 [17]	62.47	58.25	8533	57.36	73.81	98.28	132.49

The first block of Scene Designer uses two networks (pretrained on ImageNet) to encode individual object photos and sketches. We investigate which would be the best backbone for this encoder. Results are reported for Search and Generation on QuickDrawCOCO-92c and SketchyCOCO

Table 7 Investigating the use of GNN and Transformer architectures by themselves against our combined model

						QDCOCO			SketchyCOCO		
Architecture	Layers	Relations	Pos.	D	Param.	r@1	r@10	FID↓	r@1	r@10	FID↓
Transformer A	3	–	pos.enc.	512	27.7	03.25	20.22	65.62	10.72	44.63	150.11
Transformer B	6	–	pos.enc.	256	13.5	03.21	17.65	65.73	11.58	49.35	142.92
Transformer C	3	rel. pos. enc.	pos.enc.	512	27.8	04.62	25.46	64.00	07.72	42.48	145.20
Transformer D	6	rel. pos. enc.	pos.enc.	256	13.6	02.88	19.28	62.15	08.15	46.32	148.37
GNN A	4	gnn emb.	–	512	43.7	43.73	71.54	57.53	47.21	88.84	129.32
GNN B	4	gnn emb.	–	256	16.1	49.90	76.85	57.85	47.63	91.41	130.22
GNN C	4	gnn emb.	concat.	512	43.8	64.93	86.09	55.39	78.54	98.28	130.33
GNN D	4	gnn emb.	concat.	256	16.1	66.19	86.74	55.78	77.25	99.14	132.58
GNN+Self.Att. E	4(G)+1(SA)	gnn emb.	concat.	256	16.1	65.76	85.95	56.01	71.67	95.27	128.42
GNN+Self.Att. F	4(G)+1(SA)	gnn emb.	concat.	512	44.1	67.71	87.32	54.47	77.25	98.71	123.65
GNN+Transf.	4(G)+3(T)	gnn emb.	pos.enc.	256	17.6	65.98	88.26	55.46	79.39	99.57	133.22
GNN+Transf. B	4(G)+3(T)	gnn emb.	pos.enc.	512	50.1	64.17	87.90	52.96	76.82	99.57	129.70
GNN+Transf. C	4(G)+3(T)	gnn emb.	pos.enc.	128	09.4	62.08	85.12	–	69.52	97.85	–

We argue that their combination gives all the positional information their optimal entry-points into the representations (relations on the GNN, coarse positions on the Transformer). Results are reported for Search and Generation on QuickDrawCOCO-92c and SketchyCOCO. We did two interventions to give the ablated models the same amount of information, on Transformers C and D we included the graph relationships through relative positional encoding [43]; In GNN C and GNN+Self.Att. we've concatenated the coarse positional information into the input representations; and of course, in GNN+Self.Att. we included a self-attention layer to aggregate the object representations into the scene representation. The model used throughout other experiments in the paper is highlighted in bold

standalone GNN. It is easy to notice that the Transformer is not up for the task in any of its variations, we experimented with increased depth (B) and width (A) and tried to remedy the lack of graph relationships with relative positional encodings (C and D).

The GNN presents a more interesting picture, with A and B we can see that the positional information that is naturally introduced in the Transformer's positional encoding is missed in a traditional GNN. We fill this gap in (C-F) by concatenating a binary version of the pos. enc. to the Scene Graph nodes. Furthermore we experiment with using a Self Attention layer (E and F) to aggregate the SA-OLR into the Scene Representation instead of simple addition. The presented results show that as long as the positional information is included in some fashion, all three choices for aggregation, be it simple addition, self attention or the Transformer, are viable to build a model withing our framework. We opt for the Transformer as it is the best performing on SketchyCOCO and it uses the well-known positional encoding.

6 Conclusion

We introduced Scene Designer, a new state-of-the-art CSBIR model that unites the tasks of retrieval and synthesis to improved performance and novel creative workflows. We developed a learning framework that learns to represent individual objects first and further into the pipeline combines them into scene-aware robust representations for each object and for the entire scene. Scene Designer learns to embed sketched and photographic scenes into a common space, producing latent representations for both synthesizing layouts from sketched scenes and for measuring compositional similarity between sketch and image scenes. The model is trained via an expanded dataset of sketch compositions and corresponding images (QuickDrawCOCO); a secondary contribution of our work. We show that the combination of feature learning and generation losses aided by our novel take on contrastive learning for scenes are responsible for obtaining SotA performance at scene search. Further work could explore the use of this model in creative practice, and also study how the concept of composition retrieval (and this model) can be applied to other domains where multiple components are represented, such as 3D scenes [1] or vector graphics [2].

Data Availability Instructions and code for recreating the QuickDrawCOCO dataset described in this paper can be found at the code repository for the model: https://github.com/leosampaio/scene-designer.

Declarations

This work was supported by Fundação de Amparo à Pesquisa do Estado de São Paulo (grants 2017/22366-8, 2019/02808-1, 2019/07316-0), Conselho Nacional de Desenvolvimento Científico e Tecnológico (304266/2020-5), and a charitable donation from Adobe Inc.

References

1. Abdul-Rashid H, Yuan J, Li B, Lu Y, Schreck T, Bui N-M, Do T-L, Holenderski M, Jarnikov D, Le T-K, Menkovski V, Nguyen K-T, Nguyen T-A, Nguyen V-T, Ninh V-T, Rey LAP, Tran M-T, Wang T (2019) Extended 2D scene image-based 3D scene retrieval. In: 3DOR@Eurographics, https://doi.org/10.1007/978-3-030-01225-0_19
2. Abe K, Morita H, Hayashi T (2018) Similarity retrieval of trademark images by vector graphics based on shape characteristics of components. ICCAE 2018, pp. 82–86 Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3192975.3192988

3. Ashual O, Wolf L (2019) Specifying object attributes and relations in interactive scene generation. In: Proceedings of the IEEE international conference on computer vision, pp 4560–4568, https://doi.org/10.1109/ICCV.2019.00466
4. Brown TB, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, Agarwal S, Herbert-Voss A, Krueger G, Henighan T, Child R, Ramesh A, Ziegler DM, Wu J, Winter C, Hesse C, Chen M, Sigler E, Litwin M, Gray S, Chess B, Clark J, Berner C, McCandlish S, Radford A, Sutskever I, Amodei D (2020) Language models are few-shot learners. In: Larochelle H, Ranzato M, Hadsell R, Balcan M, Lin H (eds) Advances in neural information processing systems 33: Annual conference on neural information processing systems 2020, NeurIPS 2020, December 6-12, 2020, Virtual, https://doi.org/10.5555/3495724.3495883
5. Bui T, Ribeiro L, Ponti M, Collomosse J (2017) Compact descriptors for sketch-based image retrieval using a triplet loss convolutional neural network. CVIU. https://doi.org/10.1016/j.cviu.2017.06.007
6. Bui T, Ribeiro L, Ponti M, Collomosse J (2018) Sketching out the details: sketch-based image retrieval using convolutional neural networks with multi-stage regression. Comput Graph 71:77–87. https://doi.org/10.1016/j.cag.2017.12.006
7. Caesar H, Uijlings J, Ferrari V (2018) Coco-stuff: thing and stuff classes in context. In: Proc. CVPR. IEEE, https://doi.org/10.1109/CVPR.2018.00132
8. Chen T, Cheng M-M, Tan P, Shamir A, Hu S-M (2009) Sketch2photo: internet image montage. Proc ACM SIGGRAPH 28(5):124. https://doi.org/10.1145/1618452.1618470
9. Chen W, Hays J (2018) Sketchygan: towards diverse and realistic sketch to image synthesis. In: Proc. CVPR, https://doi.org/10.1109/CVPR.2018.00981
10. Dhariwal P, Nichol A (2021) Diffusion models beat GANs on image synthesis. In: Ranzato M, Beygelzimer A, Dauphin Y, Liang PS, Vaughan JW (eds) Advances in neural information processing systems, vol 34, pp 8780–8794
11. Dutta A, Akata Z (2020) Semantically tied paired cycle consistency for any-shot sketch-based image retrieval. Int J Comput Vis 128(10):2684–2703. https://doi.org/10.1007/s11263-020-01350-x
12. Dutta T, Singh A, Biswas S (2021) Styleguide: zero-shot sketch-based image retrieval using style-guided image generation. IEEE Trans Multim 23:2833–2842. https://doi.org/10.1109/TMM.2020.3017918
13. Eitz M, Hildebrand K, Boubekeur T, Alexa M (2011) Sketch-based image retrieval: benchmark and bag-of-features descriptors. IEEE Trans Vis Comput Graph 17(11):1624–1636. https://doi.org/10.1109/TVCG.2010.266
14. Gao C, Liu Q, Xu Q, Wang L, Liu J, Zou C (2020) Sketchycoco: image generation from freehand scene sketches. In: Proc. CVPR, https://doi.org/10.1109/CVPR42600.2020.00522
15. Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: NeurIPS, https://doi.org/10.5555/2969033.2969125
16. Guo L, Liu J, Tang J, Li J, Luo W, Lu H (2019) Aligning linguistic words and visual semantic units for image captioning. In: Proc. ACM multimedia, pp 765–773, https://doi.org/10.1145/3343031.3350943
17. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proc. CVPR, pp 770–778, https://doi.org/10.1109/CVPR.2016.90
18. Heusel M, Ramsauer H, Unterthiner T, Nessler B, Hochreiter S (2017) GANs trained by a two time-scale update rule converge to a local nash equilibrium. NeurIPS 30. https://doi.org/10.5555/3295222.3295408
19. Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H (2017) Mobilenets: efficient convolutional neural networks for mobile vision applications. arXiv:1704.04861
20. Hu R, Collomosse J (2013) A performance evaluation of gradient field hog descriptor for sketch based image retrieval. CVIU 117(7):790–806. https://doi.org/10.1016/j.cviu.2013.02.005
21. Isola P, Zhu J, Zhou T, Efros AA (2017) Image-to-image translation with conditional adversarial networks. In: Proc. CVPR, pp 5967–5976, https://doi.org/10.1109/CVPR.2017.632
22. Johnson J, Gupta A, Fei-Fei L (2018) Image generation from scene graphs. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition, pp 1219–1228, https://doi.org/10.1109/CVPR.2018.00133
23. Johnson J, Krishna R, Stark M, Li LJ, Shamma DA, Bernstein MS, Li FF (2015) Image retrieval using scene graphs. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition. IEEE, pp 3668–3678, https://doi.org/10.1109/CVPR.2015.7298990
24. Kingma D, Ba J (2015) Adam: a method for stochastic optimization. In: Proc. ICLR, https://doi.org/11245/1.505367
25. Liu F, Zou C, Deng X, Zuo R, Lai Y-K, Ma C, Liu Y-J, Wang H (2020) Scenesketcher: fine-grained image retrieval with scene sketches. In: ECCV. Springer, pp 718–734, https://doi.org/10.1007/978-3-030-58529-7_42
26. Lu Y, Wu S, Tai Y-W, Tang C-K (2018) Image generation from sketch constraint using contextual GAN. In: Proc. ECCV, https://doi.org/10.1007/978-3-030-01270-0_13

27. Mao X, Li Q, Xie H, Lau RYK, Wang Z (2016) Least squares generative adversarial networks. cite arXiv:1611.04076
28. Mirza M, Osindero S (2014) Conditional generative adversarial nets. arXiv:1411.1784
29. Pandey A, Mishra A, Verma VK, Mittal A, Murthy HA (2020) Stacked adversarial network for zero-shot sketch based image retrieval. In: IEEE winter conference on applications of computer vision, WACV 2020, Snowmass Village, CO, USA, March 1-5, 2020. IEEE, pp 2529–2538, https://doi.org/10.1109/WACV45572.2020.9093402
30. Pang K, Song Y-Z, Xiang T, Hospedales TM (2017) Cross-domain generative learning for fine-grained sketch-based image retrieval. In: BMVC, pp 1–12, https://doi.org/10.5244/C.31.46
31. Park T, Liu M-Y, Wang T-C, Zhu J-Y (2019) Semantic image synthesis with spatially-adaptive normalization. In: Proceedings of the IEEE conference on computer vision and pattern recognition, https://doi.org/10.1109/CVPR.2019.00244
32. Peng C, Gao X, Wang N, Tao D, Li X, Li J (2015) Multiple representations-based face sketch–photo synthesis. IEEE Trans Neural Netw Learn Syst 27(11):2201–2215. https://doi.org/10.1109/TNNLS.2015.2464681
33. Qi Y, Song Y-Z, Zhang H, Liu J (2016) Sketch-based image retrieval via siamese convolutional neural network. In: Proc. ICIP. IEEE, pp 2460–2464, https://doi.org/10.1109/ICIP.2016.7532801
34. Radford A, Kim JW, Hallacy C, Ramesh A, Goh G, Agarwal S, Sastry G, Askell A, Mishkin P, Clark J, Krueger G, Sutskever I (2021) Learning transferable visual models from natural language supervision. In: Meila M, Zhang T (eds) Proceedings of the 38th international conference on machine learning, ICML 2021, 18-24 July 2021, Virtual Event. Proceedings of Machine Learning Research, vol 139, pp 8748–8763
35. Ramesh A, Pavlov M, Goh G, Gray S, Voss C, Radford A, Chen M, Sutskever I (2021) Zero-shot text-to-image generation. In: Meila M, Zhang T (eds) Proceedings of the 38th international conference on machine learning, ICML 2021, 18-24 July 2021, Virtual Event. Proceedings of Machine Learning Research, vol 139, pp 8821–8831
36. Rezatofighi H, Tsoi N, Gwak J, Sadeghian A, Reid I, Savarese S (2019) Generalized intersection over union: a metric and a loss for bounding box regression. In: Proc. CVPR, https://doi.org/10.1109/CVPR.2019.00075
37. Ribeiro LSF, Bui T, Collomosse J, Ponti M (2020) Sketchformer: transformer-based representation for sketched structure. In: Proc. CVPR, https://doi.org/10.1109/CVPR42600.2020.01416
38. Ribeiro LSF, Bui T, Collomosse J, Ponti M (2021) Scene designer: a unified model for scene search and synthesis from sketch. In: Proc. ICCV WS, pp 2424–2433, https://doi.org/10.1109/ICCVW54120.2021.00275
39. Rombach R, Blattmann A, Lorenz D, Esser P, Ommer B (2021) High-resolution image synthesis with latent diffusion models. arXiv:2112.10752
40. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg AC, Fei-Fei L (2015) ImageNet large scale visual recognition challenge. IJCV 115(3):211–252. https://doi.org/10.1007/s11263-015-0816-y
41. Sangkloy P, Burnell N, Ham C, Hays J (2016) The sketchy database: learning to retrieve badly drawn bunnies. ACM Trans Graph (TOG) 35(4):119. https://doi.org/10.1145/2897824.2925954
42. Schönfeld E, Sushko V, Zhang D, Gall J, Schiele B, Khoreva A (2021) You only need adversarial supervision for semantic image synthesis. In: 9th international conference on learning representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. https://doi.org/10.1007/s11263-022-01673-x
43. Shaw P, Uszkoreit J, Vaswani A (2018) Self-attention with relative position representations. arXiv:1803.02155, https://doi.org/10.18653/v1/N18-2074
44. Shen Y, Liu L, Shen F, Shao L (2018) Zero-shot sketch-image hashing. In: 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018. Computer Vision Foundation / IEEE Computer Society, pp 3598–3607, https://doi.org/10.1109/CVPR.2018.00379
45. Sohl-Dickstein J, Weiss E, Maheswaranathan N, Ganguli S (2015) Deep unsupervised learning using nonequilibrium thermodynamics. In: Bach F, Blei D (eds) Proceedings of the 32nd International Conference on Machine Learning. Proceedings of Machine Learning Research, vol 37. PMLR, Lille, France, pp 2256–2265, https://doi.org/10.5555/3045118.3045358
46. Sukhbaatar S, Szlam A, Weston J, Fergus R (2015) End-to-end memory networks. In: Cortes C, Lawrence ND, Lee DD, Sugiyama M, Garnett R (eds) Advances in neural information processing systems 28: Annual conference on neural information processing systems 2015, December 7-12, 2015, Montreal, Quebec, Canada, pp 2440–2448, https://doi.org/10.5555/2969442.2969512
47. Sukhbaatar S, Weston J, Fergus R et al (2015) End-to-end memory networks. NeurIPS 28. https://doi.org/10.5555/2969442.2969512

48. Szegedy C, Ioffe S, Vanhoucke V, Alemi AA (2017) Inception-v4, inception-resnet and the impact of residual connections on learning. In: AAAI, pp 4278–4284, https://doi.org/10.5555/3298023.3298188
49. The Quick Draw! (2018) Dataset. https://github.com/googlecreativelab/quickdraw-dataset. Accessed 11 Oct 2018
50. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I (2017) Attention is all you need. NeurIPS 30. https://doi.org/10.5555/3295222.3295349
51. Wang F, Kang L, Li Y (2015) Sketch-based 3D shape retrieval using convolutional neural networks. In: Proc. CVPR. IEEE, pp 1875–1883, https://doi.org/10.1109/CVPR.2015.7298797
52. Wang T-C, Liu M-Y, Zhu J-Y, Tao A, Kautz J, Catanzaro B (2018) High-resolution image synthesis and semantic manipulation with conditional GANs. In: Proc. CVPR, https://doi.org/10.1109/CVPR.2018.00917
53. Wang N, Tao D, Gao X, Li X, Li J (2013) Transductive face sketch-photo synthesis. IEEE Trans Neural Netw Learn Syst 24(9):1364–1376. https://doi.org/10.1109/TNNLS.2013.2258174
54. Xu P, Joshi CK, Bresson X (2021) Multigraph transformer for free-hand sketch recognition. IEEE Trans Neural Netw Learn Syst. https://doi.org/10.1109/TNNLS.2021.3069230
55. Yelamarthi SK, Reddy SK, Mishra A, Mittal A (2018) A zero-shot framework for sketch based image retrieval. In: Proceedings of the european conference on computer vision (ECCV)
56. Yu Z, Kovashka A (2020) Syntharch: interactive image search with attribute-conditioned synthesis. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, pp 170–171, https://doi.org/10.1109/CVPRW50498.2020.00093
57. Yu Q, Liu F, Song Y-Z, Xiang T, Hospedales TM, Loy C-C (2016) Sketch me that shoe. In: Proc. CVPR, pp 799–807, https://doi.org/10.1109/CVPR.2016.93

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

A.4 Sketch-an-anchor: Sub-epoch fast model adaptation for zero-shot sketch-based image retrieval.

This section includes the original copy of the Sketch-an-Anchor paper, which is currently under review for presentation at an international computer vision conference (undisclosed to preserve double blind review integrity):

RIBEIRO, L. S. F.; PONTI, M. Sketch-an-anchor: Sub-epoch fast model adaptation for zero-shot sketch-based image retrieval. In: **Submission Under Review**. [S.l.: s.n.], 2022.

Submission #X. CONFIDENTIAL REVIEW COPY. DO NOT DISTRIBUTE.

Sketch-an-Anchor: Sub-epoch Fast Model Adaptation for Zero-shot Sketch-based Image Retrieval

Anonymous submission

Paper ID X

Abstract

Sketch-an-Anchor is a novel method to train state-of-the-art Zero-shot Sketch-based Image Retrieval (ZSSBIR) models in under an epoch. Most studies break down the problem of ZSSBIR into two parts: domain alignment between images and sketches, inherited from SBIR, and generalization to unseen data, inherent to the zero-shot protocol. We argue one of these problems can be considerably simplified and re-frame the ZSSBIR problem around the already-stellar yet underexplored Zero-shot Image-based Retrieval performance of off-the-shelf models. Our fast-converging model keeps the single-domain performance while learning to extract similar representations from sketches. To this end we introduce our Semantic Anchors – guiding embeddings learned from word-based semantic spaces and features from off-the-shelf models – and combine them with our novel Anchored Contrastive Loss. Empirical evidence shows we can achieve state-of-the-art performance on all benchmark datasets while training for 100x less iterations than other methods.

1. Introduction

As the world produces more photographic content and the internet takes the place of the real world in many fields, retrieval systems become ever more relevant. The task of Sketch-based Image Retrieval (SBIR) is a machine learning application that has been relevant for at least 20 years [1] and interest has only grown since; SBIR systems take sketched objects from users and ranks images on a database by semantic relevance.

While introducing Deep Learning methods to this task has led to great strides in performance, these systems also started depending on large datasets with both photos and sketches from the same category to perform well [12, 32]. That motivated Zero-shot SBIR task (ZSSBIR) [32]; the protocol for ZSSBIR follows the common Zero-shot paradigm [40], the categories of images and sketches used for evaluation are not seen in training. This setup better mimics an open set scenario, which correspond to real applications for SBIR, where categories are vast and unpredictable: e.g. an e-commerce might add new products every day and an artist will search for different references for each creation.

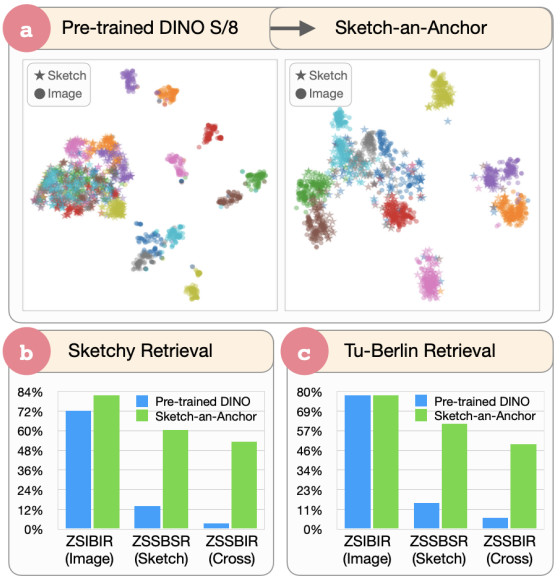


Figure 1. Our Sketch-an-Anchor model reframes the problem of Zero-shot Sketch-based Image Retrieval. Because pre-trained Self-Supervised models already perform well on Zero-shot Image-based Retrieval, our efforts should go into keeping this performance while mapping sketches to the same space. In (a) we compare our model and the pre-trained DINO S/8 that we start with by plotting a t-SNE projection of the feature vectors extracted from the zero-shot split of Sketchy. In (b) and (c) we compare both models again, but show mAP (%) on the Tu-Berlin and Sketchy Zero-shot splits on three tasks, Image-based Image Retrieval, Sketch-based Sketch Retrieval and Sketch-based Image Retrieval.

Most studies break down the problem of ZSSBIR into

two: domain alignment between images and sketches, inherited from SBIR, and generalization to unseen data, inherent to the zero-shot protocol. We argue one of these problems can be considerably simplified, as detailed later.

The field of Self-Supervised learning has also seen many proposals in recent years. The task of Self-supervision is to learn – without labeled data – general image representations that can be used for diverse downstream tasks; the approach often taken [5, 9, 10, 19, 20] is to train models effectively for image retrieval: semantically similar images should have similar representations.

We take DINO [5], a recent self-supervised model and the best at k-NN classifier evaluation (where features are extracted from the frozen network to train a k-NN classifier for ImageNet), and test its Zero-shot Image-based Image Retrieval (ZSIBIR) performance without fine-tuning on the zero-shot image split of common ZSSBIR datasets. We've found that without any further training ZSIBIR performance is already excellent (See Fig. 1), motivating us to rethink how we break down the ZSSBIR task.

Domain alignment and generalization to unseen data are still at the core of a ZSSBIR solution, but, as we've discussed, the latter is solved or close to on the image domain by contrastive self-supervised models. With this information at hand, we've designed Sketch-an-Anchor, making it our goal to learn general sketch representations that match the existing image space of the pre-trained model.

In Fig. 1 (b,c) we graph a preliminary evaluation showing that Sketch-an-Anchor indeed succeeds at this goal. We've evaluated the DINO S/8 pre-trained model and our own finetuned one at within-domain and cross-domain retrieval, showing the former is already great at ZSIBIR (image domain) while ours maintains that performance at the same time as it significantly moves the needle on within-domain Sketch-based Sketch Retrieval and cross-domain ZSSBIR. The t-SNE projection in Fig. 1 (a) corroborates our finding, showing how the DINO space is discriminative for images but messy for sketches, and then how Sketch-an-Anchor keeps the image relationships while learning better representation for the sketches.

Rethinking the sub-tasks has another remarkable benefit: since we do not need to learn the ZSIBIR sub-task, our model requires only a few iterations, 99.13% less iterations than the average ZSSBIR model to be exact, making it the fastest training ZSSBIR model in the literature by a very large margin.

To aid with our re-defined tasks we introduce the Semantic Anchors, a combination of the general Word2Vec [27] semantic space and class centers based on the image representations output by the pre-trained model. We then use the Semantic Anchors and their pairwise similarities to design our novel Anchored Contrastive Loss, elevating a strong baseline model to SoTA performance.

To summarize, our contributions are the following:

- Re-think the ZSSBIR solution on matching sketch representations to an existing pre-trained space, since those are already good image retrieval models;
- The ability to train a model for ZSSBIR in under an epoch and achieve performance competitive with SoTA;
- Introduce Semantic Anchors, which learns from visual and word-based semantic spaces with a Graph Convolutional Net and guide the fast model adaptation;
- Our novel Anchored Contrastive Loss matches the sample similarity distribution to the anchors', further improving the task of matching sketch representations to the images'.

2. Related Work

Sketch-based Image Retrieval has seen published solutions dating back to the 90s [1], pre-deep-learning methods have worked with hand-crafted features and image edge-maps to extract comparable representations between images and sketches [17, 22], which were then combined with dictionary-learning methods such as Bag-of-Visual-Words. As they dominated computer vision, CNNs became the norm in SBIR too, with studies using multi-branch networks with contrastive [36] or triplet losses [2, 29, 31, 42] and investigating the use of Edge Maps [2] and whether to share weights between branches or not [3].

Zero-shot Sketch-based Image Retrieval. The work of Shen *et al*. [32] was the first to introduce the task whilst also proposing a new hashing method to accelerate retrieval. Later works can be split by the side information they've used. The works of [6, 11, 12, 14, 43] have used word-based semantics like we do to learn more general search spaces, while other models trust generative models to simulate unseen classes during training [15, 28, 32, 41] or combine both [14]. The study of Liu *et al*. [25], SAKE, introduced the idea of using Knowledge Preservation to prevent pre-trained models from losing already acquired knowledge, and implemented their own custom distillation protocol to do so, many models follow similar protocols [34, 35, 37, 39] or combinations of them with word-based semantic information. We share SAKE's authors' goal of avoiding knowledge loss but designed our approach to not need to rely on a computationally expensive teacher signal. We highlight three studies from the Knowledge Distillation set that are most similar to ours. TVT [34] and PKSD [37] share our DINO backbone and the latter also explores the correlation between class centers, but their training protocol uses multiple ViT instances and they take 100x longer to converge. SBTKNet [35] is a recent study that makes use of class centers similarly to how we do our Semantic Anchors, which

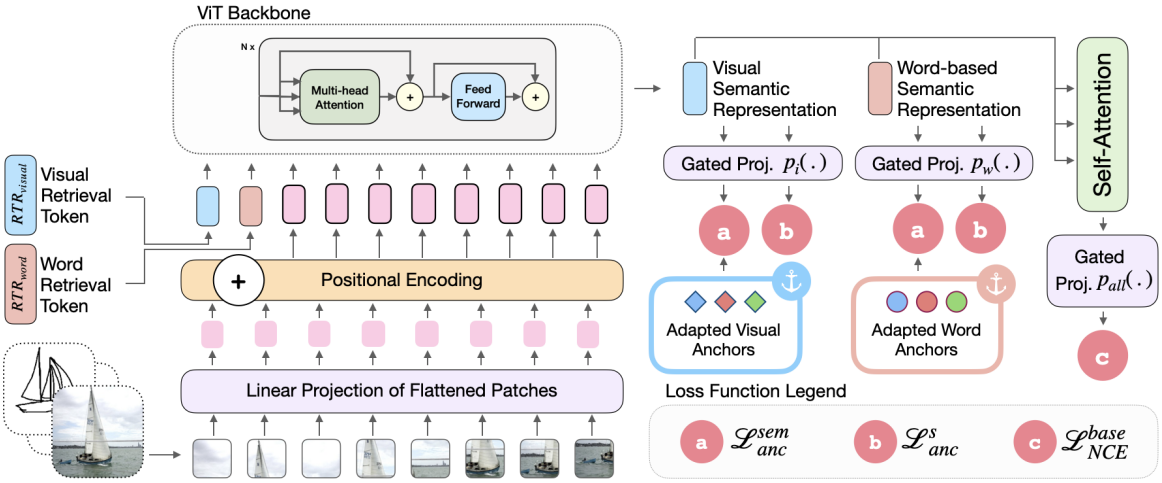


Figure 2. Diagram for the final Sketch-an-Anchor architecture. We start with a ViT S/8 backbone pre-trained with the DINO [5] protocol. To this backbone's input we include two learnable tokens, RTR_{visual} and RTR_{word}. The outputs related to these tokens will make up the Visual and Word-based Representations, which are both trained to match images and sketches with L^{s}_{anc} and to match the semantic anchors with L^{sem}_{anc}. Finally, these representations are agregated through self-attention and the final representation is trained to match domains with classic InfoNCE, L^{base}_{NCE}. The same network is used to extract features from sketches or photographic images.

makes their model one of the fastest recent ones, training in under 25 epochs; however, 25 epochs is still more than 50 times longer than our Sketch-an-Anchor's training time. Our model is competitive against all of the previous studies while being simpler to implement, not requiring a teacher-student distillation setup or multiple network instances and, of course, being faster to train and to experiment with.

3. The Sketch-an-Anchor Method

In this section we describe the task and our solution. We start by introducing our baseline model, a strong and simple ZSSBIR solution, and then include the Semantic Anchors and Anchored Contrastive loss, additions that bring the baseline up to speed to compete with the best in the literature. A diagram for the complete architecture can be found in Fig. 2.

3.1. Problem Setting

Within Zero-shot Sketch-based Image Retrieval, a given model gets a sketched object as input and ranks images on a dataset based on how well they match the sketched object semantically. Under the zero-shot setting such model is evaluated with query sketches and retrieved images of objects that were not present in training.

Formally, an SBIR dataset is split into two subsets to create the zero-shot setting, $\mathcal{D}_{\text{train}} = \{S_{\text{train}}, I_{\text{train}}, C_{\text{train}}\}$ and $\mathcal{D}_{\text{test}} = \{S_{\text{test}}, I_{\text{test}}, C_{\text{test}}\}$, where S and I are the set of labeled sketches and images for each subset and C are the classes of each subset, and the zero-shot protocol is complete by ensuring that there is no intersection between classes in each subset: $C_{\text{train}} \bigcap C_{\text{test}} = \emptyset$.

3.2. A Strong Baseline

As presented in Sec. 1, we are motivated by the already great performance of self supervised models on Zero-shot Image-based Retrieval (ZSIBR) and believe this performance can be transferred to ZSSBIR with little finetuning. We chose to start with a Vision Transformer (ViT) [13] backbone pre-trained under the Self Supervised DINO protocol [5]; We explore other architectures/training protocols and the correlation of ZSIBR performance and our ZSSBIR results in the Supp. Mat.. A ViT encoder takes fixed-resolution ($R \times R$) patches of images as input, which are linearly projected to create a sequence of "image tokens"; a learnable token CLS is included in the sequence and all tokens are processed by N sequential layers composed each of a pipeline of Layer Normalization, Multi-head Attention and feed-forward sublayers. Each layer is connected by skip-connections [21].

These models are usually trained through supervised learning, but Caron *et al.* introduced DINO, a self-supervision protocol that makes use of multi-crop training [4], a momentum encoder [20] and smaller ViT image patches to deliver representations that can make stellar k-NN classifiers. Because k-NN classifiers are essentially retrieval models (it is non-parametric and classification is done through search in the embedding space), we make a

pre-trained DINO ViT our backbone model, specifically the compact DINO S/8.

In addition to the pre-trained backbone $v(.)$ we include a new trainable token for retrieval, RTR, together with the patch-based image tokens. At the output, the processed RTR goes through a projection head $p(.)$, implemented as gated linear projection [8,26]. The same network takes sketched and image inputs, and we take from self-supervision lingo [20] to call sketches queries $q = p(v(x_s))$ and the collection of images in a batch keys of a dictionary $k_i = p(v(x_i))$, where for each query there is a single key k_+ matching it. We then take the cosine similarity of each pair and use InfoNCE [24] as our contrastive loss:

$$\mathcal{L}_{NCE}^{base} = -log\frac{exp(q.k_+/\tau)}{\sum_{i=0}^{K} exp(q.k_i/\tau)} \tag{1}$$

where τ is the temperature hyper-parameter and controls the balance between a uniform-distributed embedding and a discriminative one [38]. Additionally, our task differs from Self-Supervised learning in that, because we are doing categorical retrieval, there is not a single key k_+ matching each query, so we enforce the single key scenario in training by making sure each batch only has one sketch and image from each class in that batch.

We show in Sec. 4 how this baseline is already a quickly adapting model for ZSSBIR which performs competitively.

3.3. Semantic Anchors

With our well-performing baseline at hand, we design additional components that lean into the strengths of the pre-trained model, starting with the *Semantic Anchors*.

Word-based Embedding. It is common practice in the ZSSBIR literature and other zero-shot tasks to include a generic semantic space as a guide to the model [11, 14, 43] to prevent training from adhering too close to the training classes and make the model learn more general concepts. The most common semantic embedding used is Word2Vec, which can map words into a semantic embedding space, trained by looking at word co-occurrence on millions of texts [27]. We follow other methods and use it to map the words for each training class into this embedding space, creating what we call the *Word Embedding*, $E_{word} = \{c_i^w\}$, with each ith vector matching the ith seen/training class.

Visual Embedding. Because the chosen backbone has demonstrated retrieval performance for images, we argue that the space produced by its features is a good semantic embedding by itself, and it furthermore captures visual relationships that word embeddings cannot (See the Supp. Mat. for an investigation of the differences). We extract the representation for all images in the training sets and take the average vector of each class to make the *Visual Embedding* $E_{visual} = \{c_i^v\}$.

We collectively call these embeddings the *Semantic Anchors* because they are responsible for grounding the output in both general semantic information – the Word Embedding – and in the semantic information already in the model – the Visual Embedding.

Matching the output of the model to these spaces requires architectural modifications; we swap the baseline's learnable token RTR for another two: RTR_{word} and RTR_{visual}, responsible respectively for matching the Word and Visual embeddings at the output. With two outputs we have two new InfoNCE losses for matching sketches and images in each space, $\mathcal{L}_{NCE}^{w}$ and $\mathcal{L}_{NCE}^{v}$, which are averaged. Matching outputs of either sketches or images to their respective anchor is also implemented with (semantic) InfoNCE:

$$\begin{aligned}\mathcal{L}_{NCE}^{sem} = &- log\frac{exp(q^w.c_+^w/\tau)}{\sum_{i=0}^{C} exp(q^w.c_i^w/\tau)} \\ &- log\frac{exp(q^v.c_+^v/\tau)}{\sum_{i=0}^{C} exp(q^v.c_i^v/\tau)}\end{aligned} \tag{2}$$

with c_+^w and c_+^v being the anchors for the input's class on each embedding.

Finally, because the model needs to have a single output for retrieval, we aggregate the output vectors using Self-attention $sa(.)$, as designed by Sukhbaatar *et al.* [33], the weights in $sa(.)$ learn to perform a dynamic weighted average of the vectors, and we compute queries $q = sa(q^w, q^v)$ and keys $k = sa(k^w, k^v)$ for sketches and images. These are trained as in the baseline with $\mathcal{L}_{NCE}^{base}$.

3.4. Anchored Contrastive Loss

Because matching the geometry of the Semantic Anchor spaces has yielded a performance boost we design a new take on the InfoNCE contrastive loss tailored to further drive the model towards matching those spaces. We call it *Anchored Contrastive Loss*.

We start by building the similarity matrices for each embedding:

$$\begin{aligned}A^w = \{a_{ij}^w\}, \text{with } a_{ij}^w = c_i^w.c_j^w \\ A^v = \{a_{ij}^v\}, \text{with } a_{ij}^v = c_i^v.c_j^v\end{aligned} \tag{3}$$

Then we use the similarities as probability distributions and change the traditional InfoNCE's goal of strongly discriminating k_+ from the other keys to smoothly matching the distribution of the similarities between the classes being compared, an objective much better aligned to ours of categorical retrieval. We change $\mathcal{L}_{NCE}^{s}$ for $\mathcal{L}_{anc}^{s}$:

$$\sigma(a_{ci}^s) = \frac{exp(a_{ci}/\tau)}{\sum_{j=0}^{K} exp(a_{cj}/\tau)} \tag{4}$$

$$\mathcal{L}^s_{anc} = -\sum_{i=0}^{K} \sigma(a^s_{ci}) log \frac{exp(q.k_i/\tau)}{\sum_{j=0}^{K} exp(q.k_j/\tau)} \quad (5)$$

where s is either w or v, σ is the softmax function and c is the index of q's class; k_c in this equation is then equivalent to k_+ in $\mathcal{L}^s_{NCE}$. One can notice that our Anchored Contrastive Loss resembles a cross-entropy loss in the form $-a \log b$, computing cross-entropy between the similarities of the samples and the similarities of the anchors, both normalized into probability distributions.

We similarly apply this idea to the semantic matching Equation 2, swapping $\mathcal{L}^{sem}_{NCE}$ for $\mathcal{L}^{sem}_{anc}$. The final representation, post-self-attention, is still used with classic InfoNCE loss since we don't have a similarity distribution to match the aggregated retrieval vector.

3.5. Final Touches

With the major components in place, we complete the model with two additions to the *Semantic Anchors*; first, we make them more adaptable to the task, and second, we make the input embedding more dynamic.

To let the anchors learn with the losses, we make use of a Graph Convolutional Network. First we create for each embedding a fully connected graph $\mathcal{G}^s = \{\mathcal{V}^s, \mathcal{E}^s\}$ composed of the embedding vectors as vertices $\mathcal{V}^s$ and the similarity matrix A^s as the adjacency matrix that makes up the edges $\mathcal{E}^s$. These graphs, $\mathcal{G}^w$ and $\mathcal{G}^v$ for Word and Visual embedding, are unified by connecting matching classes together with edges with the same-weight as the self-connections (*e.g.* the c^w_i vector for "cat" is connected to the c^v_i vector for "cat"), this allows the embeddings to share knowledge. We normalize the edge values by applying softmax line-wise to the adjacency matrix and follow [23] to build a GCN:

$$(\mathcal{V}^{w+v})^{l+1} = \hat{A}^{w+v}(\mathcal{V}^{w+v})^l W^l \quad (6)$$

where the superscript $w+v$ indicates these are the unified graphs, $\hat{A}^{w+v}$ is the normalized adjacency matrix, l is a layer index, meaning these can be stacked, and W^l are learnable weights for each layer. The output nodes $(\mathcal{V}^{w+v})^L$ are used as the *Adapted Word* and *Visual Anchors*; We've included a diagram in Fig. 3 depicting this process.

Finally, we make the input embeddings more dynamic and therefore more general. For each iteration, Word Embedding vectors are randomly swapped for one of the 10 closest word vectors in the full Word2Vec space; this means for example that the vector for "cat" can be swapped for the vectors for "cats" or "feline". For the Visual Embedding, instead of taking the class centers directly we use the center and standard deviation of each class to make a normal distribution and generate slight variants of class Anchors to use as input. The similarity matrices A^w and A^v are updated together with the anchor variants.

The final Sketch-an-Anchor model is then trained with losses:

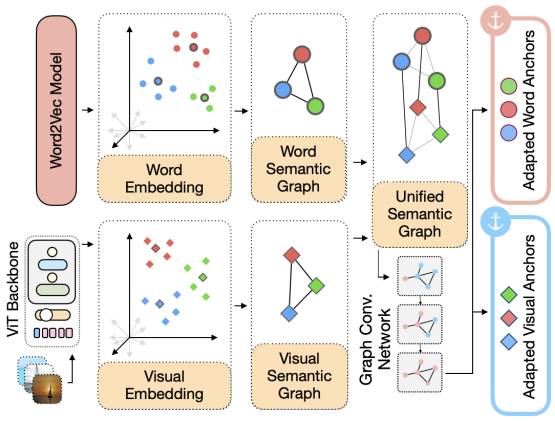


Figure 3. Diagram depicting how we learn the Adapted Semantic Anchors. We first create fully connected graphs with the original anchors and their similarity matrices, then we connect the two graphs by the matching categories. Finally we process the unified graph through a Graph Convolutional Network and input the Adapted Anchors to our model.

$$\mathcal{L}_{SAnc} = \mathcal{L}^{base}_{NCE} + \frac{1}{2}\mathcal{L}^w_{anc} + \frac{1}{2}\mathcal{L}^v_{anc} + \mathcal{L}^{sem}_{anc} \quad (7)$$

4. Experiments

In this section we detail the benchmark datasets used, implementation details, as well as empirical evidence of our model's performance. We also show how we've used the baseline to determine how few iterations are necessary and demonstrate empirically our hypothesis that the task regards mapping sketches to the already excellent image space.

4.1. Datasets

We follow most ZSSBIR literature and benchmark our model across three datasets, Sketchy [31], Tu-Berlin [16] and Quickdraw [30].

Sketchy is composed of 75,471 sketches of 125 categories and originally had 12,500 natural images, but has since had this number expanded [32] 73,002 natural images. We consider two zero-shot splits, one mirroring SAKE [25] where 25 classes were chosen randomly to compose the zero-shot test set; and another created by [41] with 21 zero-shot classes chosen specifically to avoid overlap with ImageNet.

Tu-Berlin contains 20k sketches from 250 categories and was extended [32] with 204k images matching such categories. An unrestricted selection of 30 zero-shot classes is used; for comparability we use the same selection as SAKE [25].

QuickDraw is the more recent and the largest set; In [12] a selection of 110 of the 330 classes available was chosen based on how discriminative they were to humans; a

set of 330k sketches and 204k images was built to compose QuickDraw Extended; we use the common 30 zero-shot classes.

4.2. Implementation Details

We use the Tensorflow 2 framework to build our models and run all of our experiments on a single NVIDIA RTX A5000; each experiment is repeated 5 times and we take the average metrics on evaluation. As mentioned, our backbone is a ViT-S/8, trained under the DINO protocol (DINO-S/8), this model comprises 12 layers with 6 attention heads in each. The gated linear projections used at the outputs all have 512 units. The GCN for the Semantic Anchors has 4 layers and each also has 512 units. We multiply the gradient to the backbone by 0.1 to accelerate learning of the projectors.

All of our models are trained with a batch size of 16 for 1500 iterations. The Adam optimizer with learning rate $\eta = 5\mathrm{e}{-6}$ is used and η is warmed up (raised lineraly) to the base value within the first 150 iterations and is then decayed to $1\mathrm{e}{-6}$ with a cosine schedule. In the next Section (4.3) we explore our choice of number of iterations and learning rate.

4.3. A Thousand Iterations are All You Need

One of our goals with this paper is to show that pre-trained models such as DINO can be quickly adapted to do ZSSBIR, so we designed a simple experiment to find how many iterations are needed to achieve good cross-domain performance.

To do this we created a validation set for Sketchy (split [41]), where we take 10 classes from the training set, randomly pick 64 images and 64 sketches from each, making for a compact and easy to evaluate set. We then train our Baseline model with Adam and fixed learning rates for 10 thousand iterations. The results of this experiment can be found plotted in Fig. 4.

It is clear that regardless of learning rate, the models reach a peak mAP value fairly quickly after around one thousand iterations. This number of iterations is probably enough for the model to see enough pair diversity and learn to map sketches to the image space. We select the lower learning rate $\eta = 5\mathrm{e}{-6}$ for its stability and higher peak mAP, stopping our models at 1500 iterations instead of a thousand to allow for the inclusion of warm-up and a cosine schedule on the learning rate value.

4.4. Comparison Against Other Methods

We compare our Sketch-an-Anchor model against nine of the best methods in the literature. We furthermore complement the comparison with two sub-tasks: Binarized ZSSBIR and Generalized ZSSBIR. In the former retrieval is done through hamming distance and 64-dimension binary vectors are used as representations; we've followed other

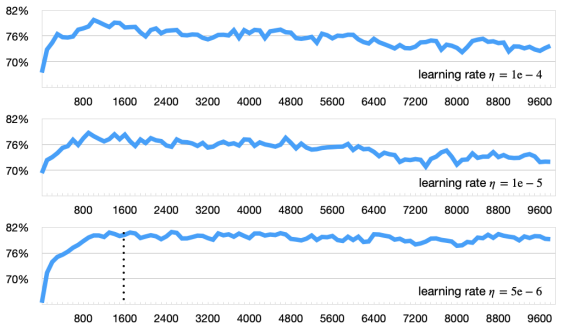


Figure 4. Evaluation on the Sketchy (Split [41]) validation set across 10 thousand iterations with the Baseline model. The x-axis shows iteration and the y-axis shows mAP (%). We test three learning rate values and select to use $\eta = 5\mathrm{e}{-6}$ and to stop training at 1500 iterations (dotted line on the bottom-most graph) for all our following experiments.

methods and used Iterative Quantization (ITQ) [18] to make our representations binary.

The results for classic ZSSBIR and Binarized ZSSBIR (subscript b) can be found in Table 1, and some qualitative ZSSBIR results for our model are on Fig. 5. We can see that our results on ZSSBIR are very competitive against the SoTA, often being around 1% shy of the best performing model in each metric. We've included the number of training epochs (Ep.) for methods that report it and highlight that our method is trained end-to-end in 1500 iterations, being on average 114 times faster to train (0.87% the number of iterations of other methods).

We take special note of the comparison against TVT [34] and PKSD [37] since both share a backbone (DINO S/8) with our model but do not enjoy the benefit of its fast adaptation properties. TVT implements three simultaneous ViT instances and trains for 50 epochs while PKSD uses two instances and trains for 30 epochs; they both use knowledge distillation to avoid losing the knowledge already in the pre-trained model. Our model doesn't need distillation because the training protocol of few iterations and low learning rate by itself keeps the knowledge from deprecation or "catastrophic forgetting".

The performance of Sketch-an-Anchor in Binarized ZSSBIR drops in comparison. We believe this has to do with the compactness of the search space. Because our model retains much of the original properties of the pre-trained DINO, it is also a more general model and likely "covers" more of the embedding space. Such a space is harder for the ITQ algorithm to work with as it relies on Principal Component Analysis to quantize the features.

The task of Generalized ZSSBIR, GZSSBIR, considers the search corpus to have images from both seen and un-

Methods	Sketchy			Sketchy split [41]			TU-Berlin			QuickDraw Extended		
	Ep.	mAP	P@100	Ep.	mAP@200	P@200	Ep.	mAP	P@100	Ep.	mAP	P@100
SAKE$_b$ [25]	20	36.4	48.7	200	35.6	47.7	200	35.9	48.1	-	-	-
SPCYC$_b$ [14]	-	34.4	39.9	-	-	-	-	29.3	39.2	-	**16.4**	**24.3**
SBTK$_b$ [35]	25	47.4	61.0	25	41.0	51.5	25	39.4	51.4	-	-	-
TVT$_b$ [34]	50	55.3	72.7	50	44.7	55.4	50	39.6	60.6	-	-	-
PKSD$_b$ [37]	30	**62.7**	**75.0**	30	**48.6**	**58.2**	30	**41.9**	**60.8**	-	-	-
Ours$_b$	**0.4**	48.5	58.5	**0.4**	39.7	51.5	**0.1**	36.9	49.7	**0.1**	10.8	15.2
SAKE [25]	20	54.7	69.2	20	49.7	59.8	20	47.5	59.9	-	-	-
DSN [39]	10	58.3	70.4	-	-	-	10	48.4	58.6	-	-	-
PCSN [11]	20	52.3	61.6	-	-	-	20	42.4	51.7	-	-	-
SPCYC [14]	-	34.9	46.3	-	-	-	-	29.7	42.6	-	**17.7**	25.5
SGCN [43]	-	-	-	-	**56.8**	48.7	-	32.4	50.5	-	-	-
SBTK [35]	25	55.3	69.8	25	50.2	59.6	25	48.0	60.8	25	11.9	-
TVT [34]	50	64.8	**79.6**	50	53.1	61.8	50	48.4	**66.2**	50	14.9	**29.9**
BDA [7]	50	43.7	51.4	50	55.6	45.8	50	37.4	50.4	50	15.4	28.6
PKSD [37]	30	**68.8**	78.6	30	56.0	**64.5**	30	**50.2**	**66.2**	30	15.0	29.7
Ours	**0.4**	67.1	76.2	**0.4**	53.5	63.0	**0.1**	49.5	60.8	**0.1**	14.8	21.2

Table 1. Comparison of Sketch-an-Anchor against nine methods from the literature at the ZSSBIR and Binarized ZSSBIR tasks. We compare on both splits of the Sketchy, on Tu-Berlin and on Quickdraw Extended. The subscript *b* refers to results with 64d hashing codes (Binarized ZSSBIR). Missing results were not reported in the original papers. We also highlight the number of Epochs (Ep.) each model takes to train when reported and bold and underline the best and second-best results for all metrics.

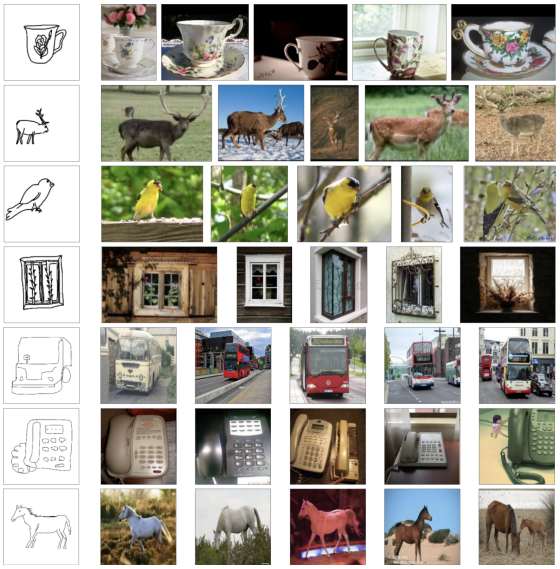


Figure 5. Visual ZSSBIR results. For each query sketch we show the 5 top images retrieved. The first two rows are from the Sketchy dataset's original split, the next two from the [41] split and the last three from Tu-Berlin.

seen classes and we follow the literature to take 20% of the images from each class in the training set and adding it to the test set instead; queries remain being restricted to unseen images. Not all methods test under this task, we've included our results against three of the nine aforementioned competitors in Table 2. The results here reflect the ones in classic ZSSBIR, with our model producing competitive results while being fast to train.

Methods	Sketchy		TU-Berlin	
	mAP	P@100	mAP	P@100
SEM-PCYC [14]	30.7	29.6	19.2	21.8
SBTKNet [35]	51.5	57.2	**33.4**	**49.4**
BDA [7]	33.8	41.3	25.1	35.7
Ours	**52.5**	**59.0**	29.0	38.1

Table 2. Comparison of Sketch-an-Anchor against 3 methods from the literature at the Generalized ZSSBIR task. We compare on the random split of Sketchy and on Tu-Berlin to match existing reports.

4.5. An Image is Worth an Entire Class

We argue that when using powerful pre-trained models ZSSBIR can be taken as the task of mapping sketches to the image space while keeping representations general enough for the zero-shot protocol. Our Semantic Anchors, Visual Anchors specially, follow this thread, we introduce guiding points in the space so that images can stay within their representations and sketches can follow.

We want to pull this thread further and ask the question:

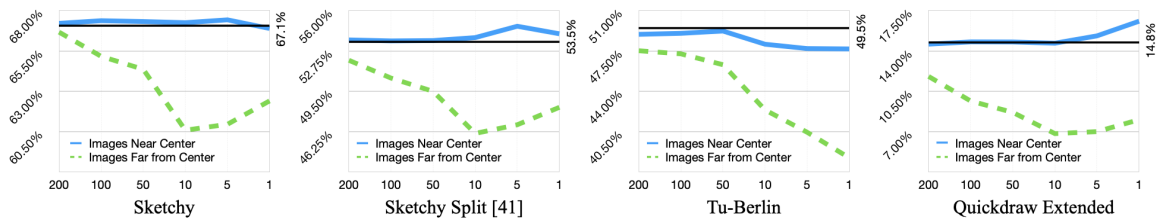


Figure 6. Experiment with Image Selection. For each model we select the N (x-axis) images closer (blue, filled line) to their corresponding Visual Anchor or the N images further from them (green, dotted line). For each selection we train Sketch-an-Anchor using the full sketch set and the selected images. The y-axis shows mAP for each model (mAP@200 for Sketchy split [41]).

if the sketches are the ones that need to be mapped, could we use only a representative sub-set of the images to train the model? Furthermore, how representative the Visual Anchors (class centers) are of the image space?

To answer these questions we've designed a new experiment: instead of taking the entire image set to train our model, we make two "opposing" sets; in one we select only the N images from each class closer to their Visual Anchors, in the other we select the N images farther from them. We repeat this with $N = \{200, 100, 50, 10, 5, 1\}$ and show the results in Fig. 6.

As is clear from the results, Sketch-an-Anchor retains its competitive performance even when only training with a single image from each class as long as this image matches its Visual Anchor. In both Sketchy splits performance is almost unchanged across all number of images and while the Tu-Berlin set needs at least 50 images to retain performance, that is still less than 10% of the original set. Finally, using only a single representative image improves the result on Quickdraw Extended to 16.6%, which we believe is due to how noisy the full image set is on this set.

The fact that the images closer to the Visual Anchors are able to train the model to the same performance level shows how relevant this side information is to the ZSSBIR task and our fast adapting Sketch-an-Anchor.

4.6. Ablation Study

We follow the description in Sec. 3 and build our ablation studies in the same fashion as we've built the model, starting from the Baseline, including the Semantic Anchors (A), Anchored Contrastive loss (B) and finally the GCN and Randomized Semantic Input (C).

The results for each of our ablated models are depicted in Table 3 for the Tu-Berlin dataset; we've also trained ablated models with Sketchy which we include in the Supp. Mat.. We can see from the studies that adding each of the Semantic Anchors bring improvements to the baseline and our Anchored Contrastive Loss $\mathcal{L}_{anc}$ further enhances performance. Completing the model, Randomized Semantic Input (R.) and the GCN work best when used together; this is an effect of how randomization contributes to a GCN that produces more general Adapted Semantic Anchors. All together we add 2.9% to the baseline's mAP, enough to make it the second-best Tu-Berlin mAP in the literature.

ID	E_v	E_w	$\mathcal{L}_{anc}$	R.	GCN	mAP	P@100
Base						47.06	59.65
A1	✓					47.76	59.45
A2	✓	✓				47.86	59.15
B1	✓	✓	✓			48.47	59.63
C1	✓	✓	✓	✓		48.66	60.03
C2	✓	✓	✓		✓	48.96	60.22
Ours	✓	✓	✓	✓	✓	**49.50**	**60.82**

Table 3. Evaluation of ablated models on the Tu-Berlin dataset, reported with Mean Average Precision (mAP) and Precision at 100. We refer to the Semantic Anchors as E_v (Visual) and E_w (Word), and refer to Randomized Semantic Input as R..

5. Conclusion

We've presented Sketch-an-Anchor, a new competitive model for ZSSBIR. We started with the insight that the problem of Zero-shot Image Retrieval is almost solved by Self-Supervision models such as DINO; with this in mind, we can stop training models from scratch or for long by focusing on mapping sketches to the already excellent image space. We've shown how to train a baseline model that is already among the best at ZSSBIR and introduced our Semantic Anchors and the Anchored Contrastive Loss to push this model to be competitive with SoTA performance. Also, our experiments have shown that our model only needs a few (or even only one) images per class to match sketches to the image space and that each of our additions to the baseline brings improvements in retrieval performance. Such findings may significantly change the way future approaches tackle the problem, allowing for experimentation with new models at a much higher rate. Future work should explore if our findings apply to the Domain Generalization and Few-shot learning fields of study, both of which share some of the challenges we've addressed on ZSSBIR.

References

[1] Jonathan Ashley, Myron Flickner, James Hafner, Denis Lee, Wayne Niblack, and Dragutin Petkovic. The query by image content (qbic) system. In *Proceedings of the 1995 ACM SIGMOD International conference on Management of data*, page 475, 1995. 1, 2

[2] Tu Bui, Leo Ribeiro, Moacir Ponti, and John Collomosse. Compact descriptors for sketch-based image retrieval using a triplet loss convolutional neural network. *CVIU*, 2017. 2

[3] Tu Bui, Leo Ribeiro, Moacir Ponti, and John Collomosse. Sketching out the details: Sketch-based image retrieval using convolutional neural networks with multi-stage regression. *Computers & Graphics*, 71:77–87, 2018. 2

[4] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33:9912–9924, 2020. 3

[5] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9650–9660, 2021. 2, 3

[6] Ushasi Chaudhuri, Biplab Banerjee, Avik Bhattacharya, and Mihai Datcu. A simplified framework for zero-shot cross-modal sketch data retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 182–183, 2020. 2

[7] Ushasi Chaudhuri, Ruchika Chavan, Biplab Banerjee, Anjan Dutta, and Zeynep Akata. Bda-sketret: Bi-level domain adaptation for zero-shot sbir. *arXiv preprint arXiv:2201.06570*, 2022. 7

[8] Brian Chen, Andrew Rouditchenko, Kevin Duarte, Hilde Kuehne, Samuel Thomas, Angie Boggust, Rameswar Panda, Brian Kingsbury, Rogerio Feris, David Harwath, et al. Multimodal clustering networks for self-supervised learning from unlabeled videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8012–8021, 2021. 4

[9] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 2

[10] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 2

[11] Cheng Deng, Xinxun Xu, Hao Wang, Muli Yang, and Dacheng Tao. Progressive cross-modal semantic network for zero-shot sketch-based image retrieval. *IEEE Transactions on Image Processing*, 29:8892–8902, 2020. 2, 4, 7

[12] Sounak Dey, Pau Riba, Anjan Dutta, Josep Llados, and Yi-Zhe Song. Doodle to search: Practical zero-shot sketch-based image retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2179–2188, 2019. 1, 2, 5

[13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 3

[14] Anjan Dutta and Zeynep Akata. Semantically tied paired cycle consistency for any-shot sketch-based image retrieval. *Int. J. Comput. Vis.*, 128(10):2684–2703, 2020. 2, 4, 7

[15] Titir Dutta, Anurag Singh, and Soma Biswas. Styleguide: Zero-shot sketch-based image retrieval using style-guided image generation. *IEEE Trans. Multim.*, 23:2833–2842, 2021. 2

[16] Mathias Eitz, James Hays, and Marc Alexa. How do humans sketch objects? *Proc. ACM SIGGRAPH*, pages 44:1–44:10, 2012. 5

[17] Mathias Eitz, Kristian Hildebrand, Tamy Boubekeur, and Marc Alexa. Sketch-based image retrieval: Benchmark and bag-of-features descriptors. *IEEE Trans. Visualization and Computer Graphics*, 17(11):1624–1636, 2011. 2

[18] Yunchao Gong, Svetlana Lazebnik, Albert Gordo, and Florent Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE transactions on pattern analysis and machine intelligence*, 35(12):2916–2929, 2012. 6

[19] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020. 2

[20] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020. 2, 3, 4

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. CVPR*, pages 770–778, 2016. 3

[22] Rui Hu and John Collomosse. A performance evaluation of gradient field hog descriptor for sketch based image retrieval. *CVIU*, 117(7):790–806, 2013. 2

[23] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. 5

[24] Cheng-I Lai. Contrastive predictive coding based feature for automatic speaker verification. *arXiv preprint arXiv:1904.01575*, 2019. 4

[25] Qing Liu, Lingxi Xie, Huiyu Wang, and Alan L Yuille. Semantic-aware knowledge preservation for zero-shot sketch-based image retrieval. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3662–3671, 2019. 2, 5, 7

[26] Antoine Miech, Ivan Laptev, and Josef Sivic. Learnable pooling with context gating for video classification. *arXiv preprint arXiv:1706.06905*, 2017. 4

[27] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and

phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013. 2, 4

[28] Anubha Pandey, Ashish Mishra, Vinay Kumar Verma, Anurag Mittal, and Hema A. Murthy. Stacked adversarial network for zero-shot sketch based image retrieval. In *IEEE Winter Conference on Applications of Computer Vision, WACV 2020, Snowmass Village, CO, USA, March 1-5, 2020*, pages 2529–2538. IEEE, 2020. 2

[29] Yonggang Qi, Yi-Zhe Song, Honggang Zhang, and Jun Liu. Sketch-based image retrieval via siamese convolutional neural network. In *Proc. ICIP*, pages 2460–2464. IEEE, 2016. 2

[30] The Quick, Draw! Dataset. https://github.com/googlecreativelab/quickdraw-dataset, 2018. Accessed: 2018-10-11. 5

[31] Patsorn Sangkloy, Nathan Burnell, Cusuh Ham, and James Hays. The sketchy database: learning to retrieve badly drawn bunnies. *ACM Trans. Graphics (TOG)*, 35(4):119, 2016. 2, 5

[32] Yuming Shen, Li Liu, Fumin Shen, and Ling Shao. Zero-shot sketch-image hashing. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 3598–3607. Computer Vision Foundation / IEEE Computer Society, 2018. 1, 2, 5

[33] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2440–2448, 2015. 4

[34] Jialin Tian, Xing Xu, Fumin Shen, Yang Yang, and Heng Tao Shen. Tvt: Three-way vision transformer through multi-modal hypersphere learning for zero-shot sketch-based image retrieval. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(2):2370–2378, 2022. 2, 6, 7

[35] Osman Tursun, Simon Denman, Sridha Sridharan, Ethan Goan, and Clinton Fookes. An efficient framework for zero-shot sketch-based image retrieval. *Pattern Recognition*, 126:108528, 2022. 2, 7

[36] Fang Wang, Le Kang, and Yi Li. Sketch-based 3d shape retrieval using convolutional neural networks. In *Proc. CVPR*, pages 1875–1883. IEEE, 2015. 2

[37] Kai Wang, Yifan Wang, Xing Xu, Xin Liu, Weihua Ou, and Huimin Lu. Prototype-based selective knowledge distillation for zero-shot sketch based image retrieval. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 601–609, 2022. 2, 6, 7

[38] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR, 2020. 4

[39] Zhipeng Wang, Hao Wang, Jiexi Yan, Aming Wu, and Cheng Deng. Domain-smoothing network for zero-shot sketch-based image retrieval. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 1143–1149. International Joint Conferences on Artificial Intelligence Organization, 8 2021. Main Track. 2, 7

[40] Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *IEEE transactions on pattern analysis and machine intelligence*, 41(9):2251–2265, 2018. 1

[41] Sasi Kiran Yelamarthi, Shiva Krishna Reddy, Ashish Mishra, and Anurag Mittal. A zero-shot framework for sketch based image retrieval. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 300–317, 2018. 2, 5, 6, 7, 8

[42] Qian Yu, Feng Liu, Yi-Zhe Song, Tao Xiang, Timothy M Hospedales, and Chen-Change Loy. Sketch me that shoe. In *Proc. CVPR*, pages 799–807, 2016. 2

[43] Zhaolong Zhang, Yuejie Zhang, Rui Feng, Tao Zhang, and Weiguo Fan. Zero-shot sketch-based image retrieval via graph convolution network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12943–12950, 2020. 2, 4, 7


ICMC
USP
SÃO CARLOS