Enhancing Scene Sketch Understanding Through a Dual-Network: Visio-Temporal Segmentation and Context-Aware Sketch Recognition

by

Aleyna Kütük

A Dissertation Submitted to the
Graduate School of Sciences and Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in

Computer Science and Engineering

September 9, 2024

Enhancing Scene Sketch Understanding Through a Dual-Network: Visio-Temporal Segmentation and Context-Aware Sketch Recognition

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Aleyna Kütük

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

_____

Prof. Dr. Metin Tevfik Sezgin (Advisor)

_____

Prof. Dr. Yusuf Sahillioğlu

_____

Prof. Dr. Yücel Yemez

Date: _____

I dedicated this to my family and friends who were always there for me, even on the tough days.

ABSTRACT

Enhancing Scene Sketch Understanding Through a Dual-Network: Visio-Temporal Segmentation and Context-Aware Sketch Recognition
Aleyna Kütük
Master of Science in Computer Science and Engineering
September 9, 2024

Understanding scene sketches involves segmenting and categorizing individual objects within the sketch. Semantic segmentation in scene sketches is crucial for distinguishing distinct sketches, but current methods often treat sketches as bitmap images, which can result in a loss of stroke order information. However, people tend to draw objects sequentially, so leveraging this temporal order could enhance segmentation performance. Moreover, traditional methods typically focus on class-level segmentation, failing to differentiate between instances within the same category.

Another important aspect of scene sketch understanding is classifying individual sketch objects within a scene. These objects often lack the detail needed for standalone recognition, making their identification challenging without contextual information. For instance, a sketch that is fluffy and circular could be interpreted as a "bush" or a "cloud," depending on its position and size within the scene. Bushes are typically drawn on the ground, while clouds are usually sketched in the sky. Despite their similar appearances, their interpretation can vary based on their context. To enhance recognition accuracy, information about relative position and size is often used in computer vision tasks like object recognition and image classification. However, many current sketch recognition methods treat sketches in isolation, overlooking the contextual information present in the scene.

To address these issues, I propose a dual-network approach comprising two novel networks for separate tasks: scene sketch segmentation and scene sketch recognition. The first network, the Class-Agnostic Visio-Temporal Network (CAVT), detects individual objects in a scene sketch using a class-agnostic object detector and groups strokes with its post-processing module. This network can distinguish object instances at the stroke level, independent of their categories. The second network, Context-Aware Graph Attention Transformer Network (CGAT-Net), processes in-

dividual sketch objects within the scene and leverages inter-object relationships to find their appropriate categories. This work is the first to apply a context-based sketch recognition approach by leveraging a novel Transformer-based Graph Attention Network within scene sketches.

Additionally, the literature lacks free-hand scene sketch datasets with both instance and stroke-level class annotations. To fill this gap, I collected the largest Free-hand Instance- and Stroke-level Scene Sketch dataset (FrISS) that contains 1,000 scene sketches and covers 403 different object classes with dense annotations. Extensive experiments on FrISS and other scene sketch datasets demonstrate that the dual-network approach, combining CAVT and CGAT-Net, as well as each network individually, outperforms existing methods in their respective domains.

ÖZETÇE

Çift Ağ ile Sahne Çizimi Anlamayı Geliştirme: Görsel-Zamansal Bölütleme ve Bağlam Farkındalıklı Çizim Tanıma

Aleyna Kütük

Bilgisayar Bilimi ve Mühendisliği, Yüksek Lisans

9 Eylül 2024

Sahne çizimlerini anlamak, çizimdeki bireysel nesnelerin ayırt edilmesini ve kategorize edilmesini içerir. Sahne çizimlerindeki anlamsal bölütleme, farklı çizimleri ayırt etmek için kritik öneme sahiptir, ancak mevcut yöntemler genellikle çizimleri bitmap görüntüleri olarak ele alır ve bu da vuruş darbesi sırası bilgilerini kaybetmelerine neden olabilir. Ancak insanlar genellikle nesneleri sıralı olarak çizerler, bu nedenle bu zamansal sıralamanın kullanılması bölütleme performansını artırabilir. Ayrıca, geleneksel yöntemler genellikle sınıf seviyesindeki bölütlemeye odaklanır ve aynı kategorideki farklı örnekleri ayırt edemezler.

Sahne çizimlerini anlamanın bir diğer önemli yönü, sahnedeki bireysel çizim nesnelerinin sınıflandırılmasıdır. Bu nesneler genellikle bağımsız tanıma için gereken detaya sahip olmadığından, bağlamsal bilgi olmadan tanımlanmaları zorlaşır. Örneğin, kabarık ve dairesel bir çizim, sahnedeki konumuna ve boyutuna bağlı olarak "çalı" veya "bulut" olarak yorumlanabilir. Çalılar genellikle yere çizilirken, bulutlar genellikle gökyüzüne çizilir. Benzer görünümlerine rağmen, yorumları bağlamlarına göre değişebilir. Tanıma doğruluğunu artırmak için, nesne tanıma ve görüntü sınıflandırma gibi bilgisayarlı görüş görevlerinde genellikle göreceli konum ve boyut hakkında bilgi kullanılır. Ancak, birçok güncel çizim tanıma yöntemi çizimleri izole bir şekilde ele alır ve sahnede bulunan bağlamsal bilgileri göz ardı eder.

Bu sorunları ele almak için, iki ayrı görev için iki yenilikçi ağdan oluşan bir çift ağ yaklaşımı öneriyorum: sahne çizim bölütleme ve sahne çizimi tanıma. İlk ağ olan Kategoriden Bağımsız Görsel-Zamansal Ağı (CAVT), sınıf agnostik bir nesne dedektörü ile objeleri tespit ederken, vuruş darbelerini gruplamak için kendi artçı işlem modülünü kullanır. Bu ağ, nesne örneklerini kategorilerinden bağımsız olarak, vuruş düzeyinde ayırt edebilir. İkinci ağ olan Bağlam Bilincine Sahip Çizge Dikkat Dönüştürücü Ağı (CGAT-Net), sahne içindeki bireysel çizim nesnelerini işler ve

uygun kategorilerini bulmak için nesneler arası ilişkileri kullanır. Bu çalışma, sahne çizimlerinde yeni bir dönüştürücü tabanlı çizge dikkat ağı kullanarak bağlam tabanlı bir çizim tanıma yaklaşımını uygulayan ilk çalışmadır.

Ayrıca, literatürde hem örnek hem de vuruş darbesi seviyesinde sınıf anotasyonlarına sahip serbest el sahne çizim veri setleri eksiktir. Bu boşluğu doldurmak için, 1,000 sahne çizimi içeren ve yoğun etiketlemelere sahip 403 farklı nesne sınıfını kapsayan en büyük Serbest El Örnek ve Vuruş Darbesi Seviyesinde Sahne Çizimi veri setini (FrISS) topladım. FrISS ve diğer sahne çizim veri setleri üzerinde yapılan kapsamlı deneyler, CAVT ve CGAT-Net'in birleştirilmiş haliyle ve her ağın tek başına kullanımıyla, kendi alanlarındaki mevcut yöntemlerden daha iyi performans gösterdiğini göstermektedir.

ACKNOWLEDGMENTS

The journey through my master's degree has been challenging and transformative, and it has changed and improved me in many ways. I would like to start by expressing my gratitude to the following organizations and individuals for their invaluable support in completing this research project.

First and foremost, I am deeply thankful to my advisor, Prof. Tevfik Metin Sezgin, for his unwavering support, guidance, and encouragement. His insights and feedback were instrumental in shaping the scope and direction of this study, and I am grateful for his mentorship.

I also extend my gratitude to all my professors who taught me the essential knowledge of Deep Learning, which enabled me to develop my skills and navigate this challenging research journey. I am particularly grateful to my thesis jury members, Prof. Yusuf Sahillioğlu and Prof. Yücel Yemez, for their constructive criticism and valuable feedback, which significantly contributed to the development of my study.

I want to express my thanks to TUBITAK (project no 120E489) and KUIS AI Center of Koc University for their financial support. Their funding allowed me to focus on my thesis research without financial concerns.

I am deeply thankful to my family—my father, Mehmet Kütük; my mother, Nurcan Kütük; and my brother, Arda Kütük—for their unwavering love and support. I owe special thanks to my roommate at Koç University, Ayça Saymaz, for her patience with my early morning alarms. I will always remember our discussions in A4 Z08 and your unique way of eating mantı with ketchup (yes, I'll keep sharing that story). I also want to express my gratitude to my earlier roommate at Sabancı University, Şeyma Sel, who shared a similar journey and recently completed her M.Sc. thesis. I know she will achieve many more accomplishments in her life, and I will always be there to support her.

My deepest gratitude goes to my best friend, Barış Batuhan Topal. He stood by me throughout this journey, believing in me even more than I believed in myself. Without his encouragement and support, I would not be writing these words today.

Last but not least, I extend my thanks to all my friends who stood by me, even during the toughest times. Sena Yapsu, Hazal Demir, Gamze Böncü Atadil, Yasemin Üstem, Selin Canakgün, Şima Özdemir, Damla Aytar, Sinem Özel, Arda Tiftikçi, Okan Karabulut, Talha Akgül, Ali Karataş, Ali Boran Gazel, Mahmut Alpaycı, Miray İnan, Nesim Şişik, all my colleagues from the Intelligent User Interfaces Laboratory, and many others—you all hold a special place in my heart.

While this acknowledgment may be a bit lengthy, I must express how incredibly fortunate I am to have such amazing people in my life. The completion of my research study would not have been possible without their support and guidance along the way.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

SOTA	state-of-the-art
UI	User Interface
MLP	Multi-Layer Perceptron
GNN	Graph Neural Network
CNN	Convolutional Neural Network
OV	Open Vocabulary Network
LDP	Local Detail Perception Network
CAVT	Class-Agnostic Visio-Temporal Network
CGAT-Net	Context-Aware Graph Attention Transformer Network
CBSC	Context-based Sketch Classification Dataset
FrISS	Free-hand Instance- and Stroke-level Scene Sketch Dataset
IoU	Intersection over Union
AoN	All or Nothing Metric
S-IoU	Stroke-level Intersection over Union Metric
OVAcc	Overall Pixel Accuracy
MeanAcc	Mean Pixel Accuracy
MIoU	Mean Intersection over Union
FWIoU	Frequency Weighted Intersection over Union

Chapter 1

INTRODUCTION

Sketching is a rapid and widely adopted way for humans to visually express ideas. Especially with the increasing prevalence of touchscreen technology, understanding hand-drawn sketches has become a crucial aspect of human-computer interaction. Sketch understanding encompasses a range of tasks, including sketch recognition, sketch segmentation, sketch-based image retrieval, sketch generation, etc. In this thesis, I focus on the challenges of sketch segmentation and recognition, proposing a novel framework for handling sketches with multiple object instances. This framework includes two distinct networks: the first is dedicated to scene sketch segmentation, while the second network classifies the sketch objects that have been segmented by the first network. These networks are utilized in sequence, but both are also suitable for separate usage. Finally, I introduce the largest Free-hand Instance- and Stroke-level Scene Sketch dataset (FrISS) as another main contribution.

Sketch semantic segmentation problem stands out as a pivotal task, offering broad applicability in the analysis of sketches and facilitating tasks like sketch-based image retrieval. Despite the considerable attention given to semantic segmentation in natural images [Chen et al. (2017); Badrinarayanan et al. (2017); Noh et al. (2015)], sketch semantic segmentation remains relatively underexplored. Earlier studies on sketch segmentation have predominantly concentrated on segmenting single-object sketches into semantically meaningful parts [Kaiyrbekov and Sezgin (2019); Zheng et al. (2023); Yang et al. (2021); Wu et al. (2018); Zhu et al. (2018); Li et al. (2018)]. On the other hand, recent attention has shifted towards scene-level sketch semantic segmentation [Sun et al. (2012); Zou et al. (2018); Ge et al. (2022); Zhang et al. (2023); Bourouis et al. (2023); Yang et al. (2023)].

Sketches are processed either as stroke sequences or as bitmap images. Many methodologies treat sketches as images and address sketch segmentation similarly to image segmentation tasks [Sun et al. (2012); Zou et al. (2018); Ge et al. (2022); Bourouis et al. (2023); Yang et al. (2023)]. However, this direct approach often leads to the loss of temporal stroke information. As sketches are composed of stroke sequences, capturing the stroke order can significantly enhance segmentation performance. Current research on scene sketch segmentation mainly focuses on assigning a class to each pixel or stroke within a scene, thus segmenting scene sketches at the class level. However, these methods cannot distinguish between individual objects that belong to the same class, such as two zebra instances in the same scene. To overcome these limitations, I introduce the Class-Agnostic Visio-Temporal Network (CAVT) that takes a scene sketch as input and generates stroke-level groupings of instances without relying on predefined class labels. This approach combines visual information and temporal stroke order within sketches. The object detector handles the visual aspects, while stroke order information is incorporated into the network through both the post-processing module and the RGB-coloring technique used during data preprocessing.

Scene sketch recognition is a follow-up problem of distinguishing individual objects within a scene. However, sketches are ambiguous standalone due to their highly abstract nature, and thus recognition of sketches is challenging without contextual information. The meaning of a sketch object can vary significantly depending on its context. For example, a circular object drawn in a kitchen environment might represent a "pizza" or a "plate", while the same sketch drawn in a street environment can depict a "car wheel" or a "soccer ball". Therefore, accurately recognizing sketched objects necessitates considering the surrounding objects within the same scene. Figure 1.1 shows scene sketches from the FrISS dataset, illustrating examples of sketch objects with similar appearances but different semantic meanings depending on the context of a scene (e.g., fish vs. airplane, rail vs. fence, streetlight vs. shower head, plate vs. pond). This underscores the need for a novel sketch recognition pipeline that incorporates contextual information for more accurate classification, rather than relying on standalone recognition.

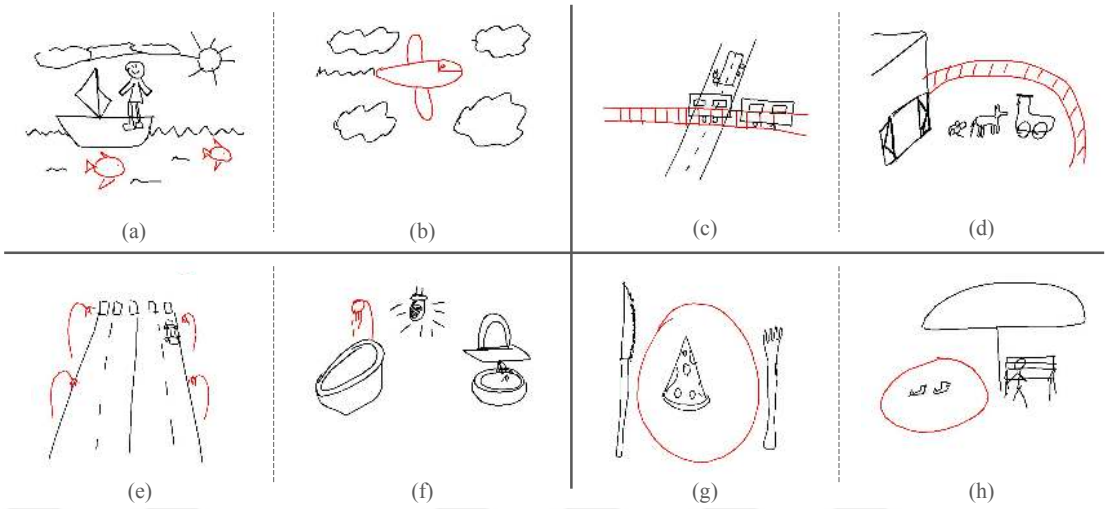


Figure 1.1: Sample scene sketches sourced from the FrISS dataset are depicted. Paired scene sketches feature objects with similar shapes and appearances, but their semantic meanings are different based on their surrounding objects. The similar-looking sketch objects are colored in red and annotated with their respective categories: (a) fish, (b) airplane, (c) rail, (d) fence, (e) streetlight, (f) shower head, (g) plate, and (h) pond.

Context plays a crucial role in scene understanding by providing key insights into the relationships between objects, such as their relative positions, sizes, and co-occurrences [Wang and Zhu (2023)]. For example, the general understanding that a car is larger than a person helps us distinguish between objects by taking their surrounding context into account. Additionally, the co-occurrence of objects acts as a critical contextual cue for recognition. For instance, knowing that a monitor is usually found on a desk can improve the accuracy of identifying a monitor when a desk is present in the scene. In other words, certain objects are commonly associated with specific locations and sizes, and understanding these relationships helps in accurately identifying individual objects by considering their surroundings.

Extensive research has been conducted on context understanding for object recognition in natural images [Rabinovich et al. (2007); Divvala et al. (2009); Parikh et al. (2011); Mottaghi et al. (2014); Bell et al. (2016); Choi et al. (2010); Galleguil-

los et al. (2008); Chen and Gupta (2017); Hu et al. (2018)]. However, context-understanding studies are limited in the sketch domain. The only work on context-based sketch recognition is conducted by Zhang et al. (2018), but their approach is not open source and relies on data distribution priors rather than deep learning techniques. Other previous studies on sketch recognition concentrate on classifying sketch objects individually, without considering the surrounding object information [Yang et al. (2020); Ribeiro et al. (2020); Xu et al. (2021b); Li et al. (2020)]. On the other hand, people usually draw sketch objects in close relation to one another within a specific context. Therefore, I propose a novel network that is designed to classify individual sketches within a scene by leveraging inter-object relationships, enhancing the recognition performance of each sketch.

The second novel network, the Context-Aware Graph Attention Transformer Network (CGAT-Net), processes a scene sketch consisting of segmented objects and generates class predictions for each sketch object. To identify individual object instances within a scene, I leverage my proposed scene sketch segmentation network, CAVT, but other segmentation methods can also be used as a preliminary step. In CGAT-Net, visual features of the individual sketches are extracted and passed into the Transformer-based Graph Attention module. This attention module exploits spatial inter-object relations to implicitly learn the context within a scene. Finally, a Multi-Layer Perceptron (MLP) Classification module produces class predictions based on the learned context embeddings for each object in the scene. In summary, CGAT-Net represents a pioneering approach to scene sketch recognition, utilizing object relationships in a novel transformer-based graph attention network to incorporate context understanding into the sketch domain.

The primary challenge in scene-level sketch studies is the lack of large-scale scene sketch datasets. Existing datasets are usually created by inserting pre-defined clip-art or free-hand single-object sketches into the layouts of reference images [Zou et al. (2018); Gao et al. (2020); Ge et al. (2022)]. These datasets only preserve scene sketches in image format, limiting their use in stroke-based sketch studies. More recently, scene datasets have been collected by instructing participants to draw scenes based on reference natural images [Chowdhury et al. (2022); Zhang et al.

(2023)]. However, this approach often diminishes the natural drawing behavior of participants, as they tend to replicate the object positions and postures from the reference images.

In this work, I introduce the largest Free-hand Instance- and Stroke-level Scene Sketch dataset (FrISS), which includes free-hand scene sketches paired with textual descriptions, stroke- and instance-level class annotations, and verbal audio recordings of drawing descriptions. To capture participants' natural drawing behavior, only textual scene descriptions are provided during the drawing process, without showing any reference images. This approach ensures that the FrISS dataset contains a variety of scene sketches that are not mere copies of reference images, allowing participants to sketch their envisioned scenes based solely on the given textual descriptions. Additionally, I avoided prolonged drawing sessions or multiple drawing attempts during data collection, preventing artificially polished scene sketches.

In summary, the main contributions of my thesis study are highlighted as follows:

1. I introduce CAVT which is the first network that utilizes both visual and temporal information of sketches for scene sketch segmentation. This is the pioneering study in performing scene sketch segmentation at both the instance and stroke levels.

2. I propose CGAT-Net that leverages inter-object relationships for scene sketch recognition. This is the first work that introduces context understanding to the sketch domain through a transformer-based graph attention network.

3. I collect the FrISS dataset that includes 1K free-hand scene sketches covering 403 object categories. It features dense annotations, including instance-level and stroke-level category labels, as well as verbal audio descriptions. Thus, FrISS can facilitate future research in stroke-based scene-level and multi-modal sketch studies.

4. I conduct extensive experiments showing that both CAVT and CGAT-Net achieve state-of-the-art performance in their respective domains on the FrISS

and other freehand scene sketch datasets. Moreover, the integrated dual network, which combines CAVT and CGAT-Net in an end-to-end manner, outperforms existing scene sketch semantic segmentation networks on these datasets. As a result, both the individual networks (CAVT and CGAT-Net) and the Dual Network (CAVT + CGAT-Net) establish new benchmarks and pave the way for advanced applications in the field.

Chapter 2

RELATED WORK

2.1 Sketch Semantic Segmentation

Existing works on sketch semantic segmentation primarily target single-object sketch datasets, focusing on dividing a single sketch object into its semantically meaningful parts [Kaiyrbekov and Sezgin (2019); Zheng et al. (2023); Yang et al. (2021); Wu et al. (2018); Zhu et al. (2018); Li et al. (2018)]. On the other hand, scene-level sketch semantic segmentation aims to distinguish and classify individual objects within a scene sketch. These studies can be categorized into two main approaches based on how sketches are processed: image-based and sequence-based methods. Image-based methods treat sketches as raster images and produce pixel-level segmentation predictions, while sequence-based methods use stroke-level information to assign class labels to each stroke. Although most single-object sketch semantic segmentation studies employ sequence-based methods [Kaiyrbekov and Sezgin (2019); Zheng et al. (2023); Yang et al. (2021); Wu et al. (2018)], there is a scarcity of sequence-based approaches for scene-level sketch semantic segmentation in the literature. This is primarily due to the lack of large-scale scene sketch datasets with stroke-level class annotations.

Previous works on scene sketch semantic segmentation have generally treated the task as a semantic image segmentation problem, neglecting the significance of stroke order information [Zou et al. (2018); Ge et al. (2022); Bourouis et al. (2023); Yang et al. (2023)]. SketchyScene [Zou et al. (2018)] was the pioneering study on scene sketch segmentation, employing an image-based architecture that assigns class labels to each pixel within a scene sketch. Ge et al. (2022) proposed a deep-shallow feature fusion network based on the DeepLab-v2 architecture [Chen et al. (2017)], examining the influence of local details on scene sketch segmentation. Bourouis

et al. (2023) introduced the first language-supervised scene sketch segmentation method, leveraging sketch captions. In contrast, Zhang et al. (2023) developed an RNN-GCN-based architecture, marking the first stroke-level approach to scene sketch semantic segmentation. Their work is particularly relevant to my study as it also incorporates visual, sequential, and spatial information from stroke sequences. However, their code is not publicly available for comparison.

Scene sketch segmentation methods have traditionally focused on assigning each pixel or stroke to a specific class within a scene [Zou et al. (2018); Ge et al. (2022); Bourouis et al. (2023); Yang et al. (2023); Zhang et al. (2023)]. However, these approaches cannot distinguish pixels or strokes of different objects within the same category, rendering them inadequate for instance-level differentiation. In contrast, I propose a novel class-agnostic scene sketch segmentation network that can identify individual object instances within a scene, irrespective of their categories. This model is pioneering in its use of both visual and temporal information to achieve instance- and stroke-level scene sketch segmentation.

2.2 Sketch Recognition

Sketch recognition methods aim to identify the appropriate category for a given sketch object. Initially, Convolutional Neural Network (CNN) based models were commonly used for this purpose. Sketch-a-Net [Yu et al. (2015)] was the first CNN-based model specifically designed for sketch recognition. Consequently, the field has shifted towards architectures that can retain temporal stroke order information, such as Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), and Transformers. SketchRNN [Ha and Eck (2017)] is a pioneering study that utilizes an LSTM structure in sketch recognition for the first time. Nonetheless, RNNs and LSTMs are limited in their ability to represent the relationships between long-range sketch stroke values compared to Transformers or Graph Neural Networks (GNN). Therefore, later studies have been proposed that combine different architectures with RNN architecture. For example, Sketch-R2CNN [Li et al. (2020)] integrates RNN and CNN, while S3Net [Yang et al. (2020)] combines RNN and GNN

to leverage spatial relationships between sketch strokes using Graph Convolutional Networks (GCN). Sketchformer [Ribeiro et al. (2020)] introduces a Transformer-based sketch recognition pipeline that encodes long-term temporal dependencies of strokes within a sketch for the first time. Multi Graph Transformer (MGT) [Xu et al. (2021b)] integrates Transformer and GNN architectures. It represents sketches as graph nodes, allowing the model to capture both geometric structure and temporal information from the sketch graphs. Unfortunately, all these works classify single-object sketches independently, failing to capture scene-level contextual information during the recognition of individual objects. Zhang et al. (2018) introduced the first study on context-based sketch recognition; however, their approach depends on data distribution priors rather than leveraging a deep learning-based architecture to harness contextual information. In contrast, I propose the first scene sketch recognition pipeline that integrates both visual features of sketches and inter-object relationships to enhance the recognition performance of individual objects within a scene.

2.3 Context Understanding

Context understanding in object detection has been studied significantly over the years [Rabinovich et al. (2007); Divvala et al. (2009); Parikh et al. (2011); Mottaghi et al. (2014); Bell et al. (2016)]. Among these studies, various sources of context have been explored in the literature, including spatial and semantic contexts [Wang and Zhu (2023)]. While spatial context captures the relative positions and sizes between objects within a scene, semantic context refers to the co-occurrence of an object to be found in certain scenes but not others. Earlier efforts utilized spatial and co-occurrence priors to detect objects [Choi et al. (2010); Galleguillos et al. (2008)]. However, these works do not utilize any deep learning model to exploit object relations for object detection. With the advancement of deep learning techniques, models have increasingly incorporated contextual cues by leveraging deep learning architectures. Bell et al. (2016) introduced the inside–outside net (ION), which encodes contextual information using RNNs. Chen and Gupta (2017) pro-

posed a Spatial Memory Network (SMN) to model the instance-level context. Later, Hu et al. (2018) introduced a Transformer-like attention mechanism to model relationships between objects for the first time. These studies highlight the importance of contextual information in improving computer vision tasks, especially object detection. However, context understanding has been rarely explored in the sketch domain. To address this gap, I propose a context-based scene sketch recognition network that utilizes object relationships within scene sketches, including relative position and size information.

2.4 Sketch Datasets

Sketch datasets can be divided into two main categories: single-object and scene sketch datasets. Single-object datasets contain sketches of individual object instances, while scene sketch datasets include drawings with multiple objects. Table 2.1 summarizes various sketch datasets, including my collected scene sketch dataset, FrISS. The upper half of Table 2.1 lists single-object sketch datasets, while the lower half details the scene sketch datasets.

Table 2.1: Summary of the sketch datasets. C, I and D denote class-level annotations, instance-level annotations, and scene sketch textual descriptions, respectively.

Dataset	# of Sketches	# of Cat.	Vector	Free-hand	Scene-level	Publicly Available	Annot. Type
QMUL Shoe [Yu et al. (2016)]	419	1		✓		✓	C, I
QMUL Chair [Yu et al. (2016)]	297	1		✓		✓	C, I
Sketchy [Sangkloy et al. (2016)]	75K	125	✓	✓		✓	C, I
TU-Berlin [Eitz et al. (2012)]	20K	250	✓	✓		✓	C, I
QuickDraw [Ha and Eck (2018)]	50M+	345	✓	✓		✓	C, I
SketchNet [Dede (2023)]	120	995	✓	✓		✓	C, I
SketchyScene [Zou et al. (2018)]	7K+	45			✓	✓	C, I
SketchyCOCO [Gao et al. (2020)]	14K+	17			✓	✓	C, I
SKY-Scene [Ge et al. (2022)]	7K+	30			✓	✓	C
TUB-Scene [Ge et al. (2022)]	7K+	35			✓	✓	C
CBSC [Zhang et al. (2018)]	331	74	✓	✓	✓	✓	C, I
FS-COCO [Chowdhury et al. (2022)]	10K	92-150	✓	✓	✓	✓	D
SFSD [Zhang et al. (2023)]	12K+	40	✓	✓	✓		C
FrISS (Ours)	1K	403	✓	✓	✓	✓	C, D, I

QMUL Shoe [Yu et al. (2016)], QMUL Chair [Yu et al. (2016)], and Sketchy [Sangkloy et al. (2016)] are multi-modal single-object sketch datasets that contain corresponding natural images paired with each sketch, thus supporting sketch-based image retrieval tasks. TU Berlin [Eitz et al. (2012)] is the first large-scale free-hand single-object sketch dataset, that is collected via crowdsourcing. QuickDraw [Ha and Eck (2018)] is another widely used free-hand sketch dataset, gathered through the online game 'Quick, Draw!'. Different from other single-object sketch datasets, SketchNet [Dede (2023)], a recently proposed dataset, includes multiple interpretations for each sketch object. It comprises 995 class categories for only 120 different single-object sketches. The sketches are randomly selected from various QuickDraw classes and post-processed to capture their intermediate drawings. These sketches are then presented to volunteers to obtain different interpretations in four orientations (0, 90, 180, and 270 degrees) and varying contexts.

A growing number of large-scale scene-level sketch datasets have been proposed to address the need for higher-level sketch understanding. SketchyScene [Zou et al. (2018)] was a pioneer in this area, created by placing clip-art-like single-object sketches onto reference images used as layout templates. SketchyCOCO [Gao et al. (2020)] is another synthetically generated scene sketch dataset that integrates free-hand single-object sketches into the corresponding mask areas of COCO-Stuff real images [Caesar et al. (2018)]. More recently, Ge et al. (2022) introduced two additional semi-synthetic scene datasets, SKY-Scene and TUB-Scene. Although synthetic scene sketch data generation offers a quick solution to the scarcity of large-scale scene datasets, it lacks the authenticity of human drawing behavior. Additionally, none of these synthetic datasets are available in vector storage formats, rendering them unsuitable for stroke-based approaches. FS-COCO [Chowdhury et al. (2022)] stands out as the first free-hand scene sketch dataset collected in vector format, accompanied by scene captions. However, it lacks stroke- or object-level annotations, limiting its applicability in semantic segmentation studies. The Scene-level Free-hand Sketch Dataset (SFSD) [Zhang et al. (2023)] is another free-hand scene sketch dataset that offers both vector storage format and stroke-level class annotations, but it is not publicly available. Lastly, Zhang et al. (2018) introduced

CBSC, the only publicly accessible free-hand scene sketch dataset with instance-level class annotations in vector format. As a result, CBSC was the only dataset available for evaluating my proposed networks.

To address the lack of a large-scale free-hand scene sketch dataset with comprehensive class annotations, I introduce the FrISS dataset to the literature. FrISS features free-hand scene sketches annotated at both the instance and stroke levels, accompanied by textual scene descriptions and verbal audio recordings of participants describing the sketches.

Chapter 3

CAVT: CLASS-AGNOSTIC VISIO-TEMPORAL NETWORK FOR SCENE SKETCH SEGMENTATION

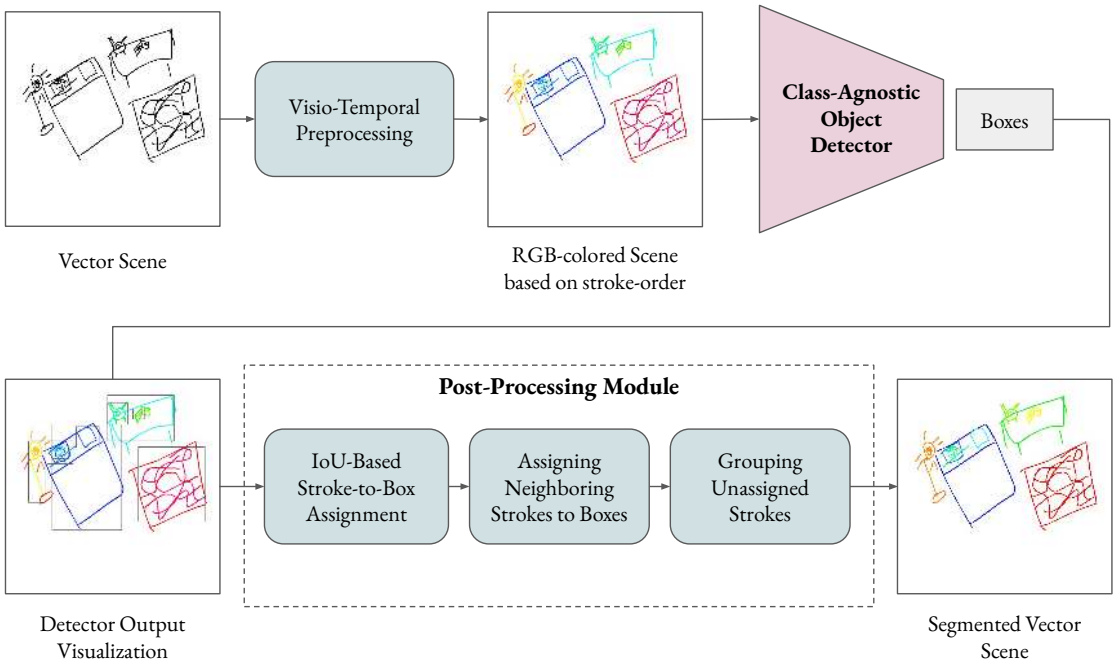


Figure 3.1: The complete architecture of CAVT

In this section, I explain the architecture of CAVT, a novel network for scene sketch segmentation. As illustrated in Figure 3.1, CAVT consists of two sub-modules: (i) a Class-Agnostic Visio-Temporal object detector and (ii) a Post-Processing module. Initially, each scene sketch is pre-processed to preserve the temporal stroke order using an RGB coloring technique. These color-coded sketches are then passed through the Class-Agnostic Object Detector to generate prediction boxes. Following this, each object instance is further segmented at the stroke level through the

post-processing module. This module refines the detector's outputs using a set of rules for stroke-level instance grouping, leveraging temporal stroke order and spatial features (i.e., pairwise stroke distances and stroke intersections with bounding boxes). The final output of CAVT consists of stroke groups corresponding to each object instance within a given scene. The details of each module are explained in the subsequent sections.

3.1 Class-Agnostic Visio-Temporal Object Detector

Object detection models are widely used in the literature and demonstrate significant success in natural images [Zhang et al. (2022); Li et al. (2023); Xu et al. (2021a)]. However, their direct application to the sketch domain faces challenges due to the domain shift from real-life images to scene sketches. Achieving fully supervised detector training on sketches necessitates a large-scale instance-level scene sketch dataset. Furthermore, converting sketches from vector storage to image format leads to temporal information loss. To effectively utilize temporal cues, the training dataset should maintain strokes in vector storage format. Unfortunately, none of the existing large-scale datasets offer both instance-level annotation and vector storage format [Zou et al. (2018); Gao et al. (2020); Ge et al. (2022); Chowdhury et al. (2022)]. Consequently, there is a critical need for a large-scale scene sketch dataset encompassing stroke vector storage and instance-level annotations.

To train an object detector in the sketch domain, I constructed a synthetically generated large-scale scene sketch dataset. I employed the widely known Quick-Draw single-object sketch dataset [Ha and Eck (2018)] by keeping the stroke order of individual sketches. Each scene is composed of a minimum of 2 and a maximum of 8 randomly chosen objects from a pool of 345 categories, with 70K drawing instances per category. Objects are randomly scaled to have a large side length ranging from 50 to 700 pixels and positioned randomly within the scene. To prevent extreme overlapping between objects, I ensure that the intersection-over-union (IOU) value between them remains below 0.35. Scenes are created in two potential sizes: 720x1280 or 1280x720 pixels. Each hyperparameter related to synthetic scene

generation is chosen based on an analysis of similar existing sketch datasets and by checking synthetic visual outputs.

To capture stroke order, I adopted an RGB coloring technique to maintain a 3-channel input and values ranging from 0 to 255 for the detector. In the RGB coloring technique, the neighboring strokes are represented with colors closer in the spectrum that spans from blue to red. Therefore, the strokes of the same object are expected to contain similar colors. Although a single object may not be entirely drawn in one stroke sequence, individual sequences are expected to exhibit consistent patterns. Besides the shape and distance of strokes, I expect that the object detector will learn to recognize groups of consecutively sketched strokes. An illustrative example of a scene colored according to stroke order is depicted in Figure 3.2. In total, I generated 11.5K synthetic drawing scenes under these settings, allocating 10K for training and 1.5K for validation purposes.

Figure 3.2: A sample scene sketch from the CBSC dataset [Zhang et al. (2018)], colored using an RGB coloring technique based on stroke order. Each stroke in the scene is color-coded according to its drawing order, with colors ranging from blue to red, as illustrated by the spectrum at the bottom.

Fully-supervised detectors are typically trained to recognize specific predefined classes, restricting their ability to detect objects beyond these predetermined categories. To address this constraint, a class-agnostic detector is decided to be trained,

in which the detector solely predicts potential object areas without the need for classification. Additionally, objects of the prepared synthetic dataset are selected from an extensive set of categories and diverse sketch styles to enhance the robustness of the class-agnostic object detector to different categories and drawing styles. I conducted an ablation study to evaluate the impact of class-agnostic training and discuss it in Section 8.1.1.

To select an appropriate object detector, I reviewed cross-domain object detection studies in the literature [Deng et al. (2021); Jiang et al. (2021); Topal et al. (2023)]. DASS-Detector [Topal et al. (2023)] stands out for its high performance within its domain, leveraging the YOLOX architecture [Ge et al. (2021)]. Inspired by their work, I also utilize the YOLOX framework in this study. The trained YOLOX detector provides predictions for potential object regions within sketch scenes, offering object-bounding boxes on the coordinate plane. To refine these predictions, I introduce a post-processing module designed to group object strokes based on the bounding box predictions. Details of the post-processing module are discussed in the following section.

3.2 Post-Processing Module

This module generates stroke-level segmentation for individual sketches by utilizing the outputs of the object detector. The complete algorithm for the post-processing module is outlined in Algorithm 1.

First, the predicted bounding boxes are sorted in ascending order based on their area, from smallest to largest. Starting with the smallest bounding box, the stroke sequence with the highest Intersection over Union (IoU) value relative to the selected bounding box is identified. If this IoU value exceeds a threshold called $IoU\_threshold$, the corresponding stroke set is assigned to the selected box. The remaining strokes that have not been assigned to a bounding box undergo an overlap ratio assessment. For each remaining longest stroke sequence, if the overlap ratio between this sequence and the nearest bounding box exceeds a threshold called $OR\_threshold$, the strokes in the set are assigned to that box. The overlap ratio is determined by dividing the

Algorithm 1: Post-Processing Module

Input: boxes, IoU_threshold, OR_threshold

Output: segmented stroke groups

while *there is alternation in stroke grouping* **do**
- Mark all strokes as unassigned.
- Sort the boxes by area in ascending order.
- **for** *each box* b_i *in boxes* **do**
  - Find the longest stroke sequence S that has the highest IoU with the box b_i.
  - **if** *the overlap ratio between* S *and* b_i *is more than IoU_threshold* **then**
    - Assign stroke sequence S to bounding box b_i.
- **for** *each unassigned longest stroke sequence* S_u **do**
  - Find the nearest bounding box b_i.
  - **if** *the overlap ratio between* S_u *and* b_i *is more than OR_threshold* **then**
    - Assign strokes in S_u to bounding box b_i.
- **for** *each longest stroke sequence* S_u *that are unassigned* **do**
  - Find the boundaries S_u: x_min, y_min, x_max, y_max.
  - Define a new box b_{new} from values x_min, y_min, x_max, y_max.
  - Append b_{new} to the boxes.
  - Assign each stroke in S_u to b_{new}.
- **for** *each box* b_i *in boxes* **do**
  - Update the coordinates of each b_i according to the most recent assignment of strokes.

intersection area between the box and the stroke set by the total area of the stroke set. Both the *IoU_threshold* and *OR_threshold* thresholds are determined using a grid-search algorithm, as detailed in Section 8.1.2. Strokes that remain unassociated with any bounding box after these initial steps are treated as distinct objects and their coordinates are added to the list of predicted boxes. The coordinates of each bounding box are then updated according to the most recent assignment of strokes.

These steps are repeated until there is no further change in the grouping of strokes, ensuring that every stroke is assigned to a potential object bounding box.

As the object detector produces bounding boxes without class predictions, strokes are grouped without class information. This approach enables the utilization of an external sketch object classifier, offering several advantages:

- Both stroke- and image-based single sketch classifiers can be employed, each capable of identifying broader or narrower object categories, or sketches with varying complexities.
- Inference time and required memory can be adjusted based on the chosen classifiers.

Chapter 4

CGAT-NET: CONTEXT-AWARE GRAPH ATTENTION TRANSFORMER NETWORK FOR SCENE SKETCH RECOGNITION

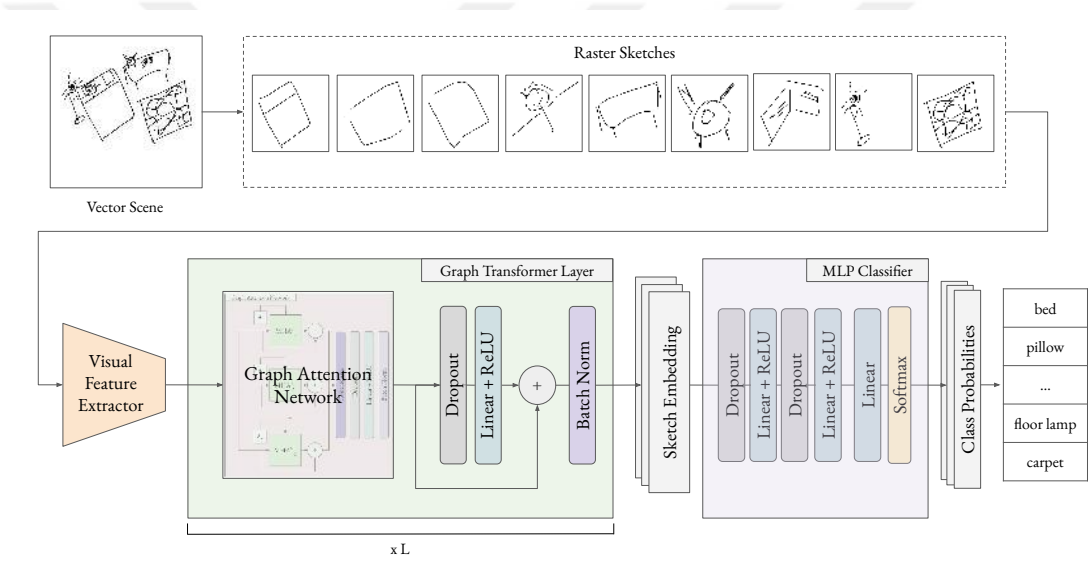


Figure 4.1: The complete pipeline of CGAT-Net

In this section, I explain the details of CGAT-Net. The overall architecture is illustrated in Figure 4.1. CGAT-Net takes previously segmented sketch objects as its input. To segment sketch objects, I utilize my proposed segmentation network CAVT and its details are discussed in Section 3. However, any external scene sketch segmentation network could also be utilized to obtain the segmented sketches within a scene.

CGAT-Net consists of three main components: a Sketch Visual Feature Extractor, a stack of Graph Transformer Layers, and a Multi-Layer Perceptron (MLP) Classification Module. First, I utilize a CNN-based Visual Feature Extractor to ob-

tain visual features of individual sketch objects within a scene. Secondly, the Graph Transformer Layer processes extracted visual features and inter-object relation matrices through a Transformer-based architecture. These matrices are designed to capture spatial relations among sketch objects within a scene. As depicted in Figure 4.1, the Graph Transformer Layer is iterated L times, with each layer comprising a Graph Attention Network and a Feed-Forward layer. Finally, the MLP Classifier identifies the category of each sketch object.

4.1 Graph Attention Matrices

I introduce five different graph attention matrices to be used in the Graph Transformer Layers. Graph Attention Network learns inter-object relations between individual sketch objects in a scene by utilizing these matrices. I provide the construction details for each matrix in the following subsections. In these subsections, I use the following notation for a pair of sketch objects: o_i and o_j, where i and j belong to the range $[1, m]$, and m represents the number of object instances in a scene.

4.1.1 Spatial Distance Matrix

The center points of objects o_i and o_j can be represented as 2-d vectors c_i and c_j, with the diagonal length of the scene denoted as d. The normalized Euclidean distances between two object centers are calculated as follows:

$$A_{i,j} = 1 - \frac{\|c_i - c_j\|_2}{d} \tag{4.1}$$

where $A_{i,j} \in [0, 1]$.

4.1.2 Directed-Horizontal Distance Matrix

I calculate the x-axis distances of two object centers in a given scene. The x-coordinates of the center points for objects o_i and o_j are represented as x_{c_i} and x_{c_j}.

$$A_{i,j} = \frac{x_{c_i} - x_{c_j}}{w} \tag{4.2}$$

where w is the width of the scene. $A_{i,j} = -A_{j,i}$ due to the relative position of objects to each other, and $A_{i,j} \in [-1, 1]$.

4.1.3 Directed-Vertical Distance Matrix

Similar to the directed-horizontal distance matrix, the y-axis distances of two object centers are calculated in a given scene. The y-coordinates of the center points for objects o_i and o_j are represented as y_{c_i} and y_{c_j}.

$$A_{i,j} = \frac{y_{c_i} - y_{c_j}}{h} \tag{4.3}$$

where h is the height of the scene. Similarly, $A_{i,j} = -A_{j,i}$ and $A_{i,j} \in [-1, 1]$.

4.1.4 Occlusion Matrix

I measure the occlusion between paired objects by calculating the area of their intersecting bounding box. The intersecting area is represented as $u_{i,j}$, while the areas of individual objects o_i and o_j are denoted as a_i and a_j, respectively.

$$A_{i,j} = \frac{u_{i,j}}{a_i} \tag{4.4}$$

where $A_{i,j} \in [0, 1]$.

4.1.5 Size-Ratio Matrix

I calculate the relative areas of object bounding boxes and normalize them with the original area of the scene. The object areas are represented as a_i and a_j.

$$A_{i,j} = \frac{a_i}{a_j \cdot w \cdot h} \tag{4.5}$$

where w and h are the width and height of the scene, respectively, and $A_{i,j} \in (0, 1]$.

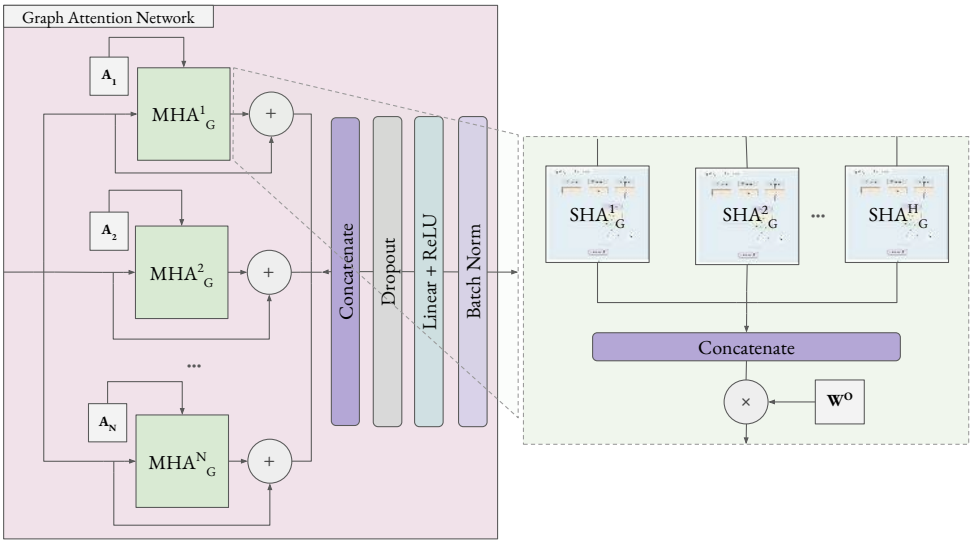


Figure 4.2: The Graph Attention Network architecture is illustrated. Each visual sketch feature is processed through N different Graph Multi-Head Attention (MHA_G) blocks. MHA_G takes an attention matrix, denoted as A_i where $i \in [1, N]$ and N is the number of attention matrices. MHA_G contains H different Graph Single-Head Attention (SHA_G) blocks. The concatenated outputs of SHA_G blocks are multiplied with W^O to transform the output dimension. The architecture of the SHA_G block is illustrated in Figure 4.3.

4.2 Sketch Visual Feature Extraction

Consider a scene sketch composed of m individual sketch objects $o_1, o_2, ..., o_m$. In this module, each object is processed as a bitmap image and input into a CNN-based deep learning model to extract visual features, denoted as $\mathcal{E}_i$ where $i \in [1, m]$. An ablation study was conducted on various backbone architectures in Table 8.3. Inception-V3 [Szegedy et al. (2016)] offers a lightweight design and strong performance in single-sketch object classification, making it an ideal choice for extracting visual features from sketch objects. The complete CGAT-Net pipeline is illustrated in Figure 4.1.

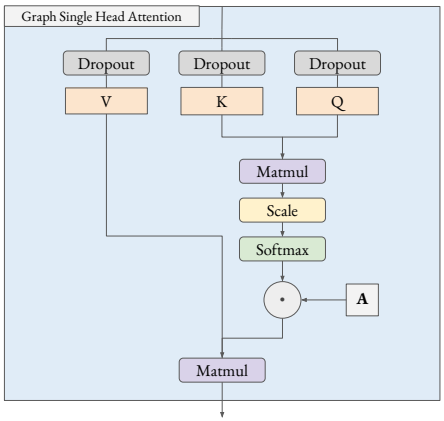


Figure 4.3: The Graph Single Head Attention (SHA_G) block is illustrated. I multiply graph attention matrices (A) with the output of the softmax operation in the original design of "Scaled Dot-Product Attention" which is proposed by [Vaswani et al. (2017)].

4.3 Graph Transformer Layer

This layer contains the Graph Attention Network and a Feed-Forward layer. It takes visual features and outputs sketch embeddings (see Figure 4.1). Individual components are explained in the following sub-sections.

4.3.1 Graph Single-Head Attention

I adopt the "Scaled Dot-Product Attention" [Vaswani et al. (2017)] for the Graph Single-Head Attention (SHA_G). Initially, I apply dropout just before the linear projections of Q, K, and V. Later, I multiply graph attention matrices (A) with the output of the softmax operation in the original "Scaled Dot-Product Attention" implementation. The illustration of the SHA_G block is given in Figure 4.3. The Graph Single-Head Attention block formulation is as follows:

$$SHA_G(Q, K, V, A) = A \odot softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{4.6}$$

where A is a graph attention matrix of size m x m where m is the number of objects in the scene, and $\odot$ is the Hadamard product operation. Let Q, K, and V

be the query, key, and value matrices of size m x d_q, m x d_k, and m x d_v in order. Each MHA_G includes H different SHA_G blocks where H is set to 6. Additionally, $d_q = d_k = d_v = \frac{d_{model}}{H}$, where d_{model} is calculated as H multiplied by the feature vector dimension, which is 512.

4.3.2 Graph Multi-Head Attention

The Graph Multi-Head Attention (MHA_G) block consists of H different Graph Single-Head Attention (SHA_G) blocks. Each MHA_G^i leverages a different attention matrix A_i where $i \in [1, N]$. The right side of Figure 4.2 zooms into the architecture of the MHA_G^i block. The formulation of this block is as follows:

$$MHA_G(Q, K, V, A) = Concat(head_1, ..., head_H)W^O \tag{4.7}$$

where $Concat(.)$ is the concatenation and $W^O \in \mathbb{R}^{Hd_v \times d_{model}}$. The concatenated outputs of SHA_G blocks are multiplied with W^O to transform the output dimension to d_{model}. SHA_G block outputs are represented as $head_i$ and computed as follows:

$$head_i = SHA_G(QW_i^Q, KW_i^K, VW_i^V, A) \tag{4.8}$$

where $W_i^Q \in \mathbb{R}^{d_{model} \times d_q}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, and $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$.

4.3.3 Graph Attention Network

Each MHA_G^i block operates in a residual manner, adding the visual features of objects, $\mathcal{E}_i$, to the output of MHA_G^i as follows:

$$h_i = MHA_G^i(Q, K, V, A_i) + \mathcal{E}_i \tag{4.9}$$

Subsequently, the outputs of each MHA_G block, denoted as h_i, are concatenated and processed through a series of operations: dropout, a linear layer, ReLU activation, and batch normalization. The left side of Figure 4.2 illustrates the architecture of the Graph Attention Network.

4.3.4 Feed Forward Network

The output of the Graph Attention Network is passed sequentially through a Feed-Forward Network, which consists of dropout, a linear layer, ReLU activation, a residual connection, and batch normalization. In the end, this network generates sketch embeddings for each object. The layers of the Feed-Forward Network are illustrated in Figure 4.1.

4.4 Multi-Layer Perceptron Classification Module

The Multi-Layer Perceptron (MLP) Classification Module takes the sketch embeddings obtained from the stack of Graph Transformer Layers. In the MLP, these sketch embeddings are processed through a series of dropout, linear, and ReLU activation operations. Finally, a Softmax operation is applied to obtain m distinct class predictions. The MLP Classifier is illustrated in Figure 4.1.

4.5 Training Dataset Preparation

CGAT-Net requires a large-scale free-hand scene sketch dataset that is publicly available and includes instance-level class annotations. However, none of the existing free-hand scene sketch datasets provides the necessary information [Chowdhury et al. (2022); Zhang et al. (2023)]. To address this gap, I constructed a synthetic large-scale scene sketch dataset, following methods similar to previous synthetic scene sketch construction approaches [Gao et al. (2020); Zou et al. (2018); Ge et al. (2022)]. These previous datasets use natural scene image datasets as reference images and replace original objects with free-hand single sketches to construct synthetic scene sketches. However, they cover a limited range of object categories and feature detailly drawn sketches like those in the TU Berlin [Eitz et al. (2012)] and Sketchy [Sangkloy et al. (2016)] datasets. To create scene sketches that resemble everyday human drawings, I prefer to use free-hand single sketches with fewer strokes per sketch. Therefore, I leverage QuickDraw [Ha and Eck (2018)], SketchNet [Dede (2023)], and individual sketch objects from the FrISS dataset for scene construction, rather than TU Berlin and Sketchy datasets. For scene layout, I refer to the MS COCO dataset [Lin et al.

(2014)], similar to SketchyCOCO [Gao et al. (2020)]. In this way, I introduce a synthetic scene sketch dataset that covers 194 different object categories. These object categories are shared in the Supplementary Material A.1.

To replace natural objects with sketches, a category mapping between source and target datasets is necessary. Accordingly, I mapped categories from MS COCO to QuickDraw, FrISS, and SketchNet using text embedding similarities as proposed by Ge et al. (2022), and manually verified each match. Figure 4.1 includes sample object category mappings for a small subset of categories. The full category mapping from all MS COCO classes to their corresponding sketch classes can be found in Supplementary Material A.2.

During the construction of synthetic scene sketches, if a source object category corresponds to multiple target categories, one is randomly selected. The chosen target objects are then placed on an empty canvas using the original bounding boxes of the source objects. Sketch instances with the closest width-to-height ratio to the source object are selected. Each scene contains between 1 and 15 objects, with no more than 3 objects from any single category. For categories with more than 3 instances, the 3 largest objects by area are chosen for sketch replacement, and the remaining instances are excluded. In total, 120K synthetic drawing scenes are generated, with 115K used for training and 5K for validation.

Table 4.1: A subset of the object category mapping between source image dataset (MS COCO), and target sketch datasets (QuickDraw, FrISS, SketchNet)

Source Category (MS COCO)	**Target Category (QuickDraw & FrISS & SketchNet)**
clock	clock, alarm clock, wristwatch
cup	coffee cup, cup, mug
desk-stuff	pencil, eraser, computer case
fruit	blackberry, blueberry, grapes, pear, pineapple, strawberry, watermelon
refrigerator	cooler
sky-other	sun, moon, star
truck	truck, ambulance, firetruck, pickup truck, van

Chapter 5

THE DUAL-NETWORK: MERGING CAVT AND CGAT-NET

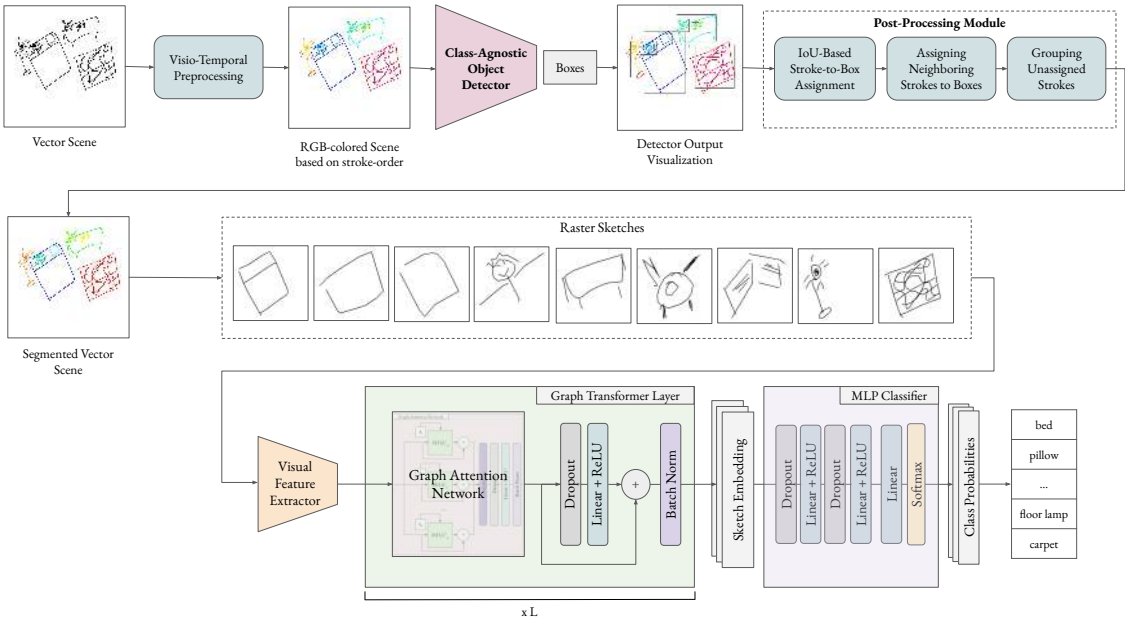


Figure 5.1: The entire pipeline of the dual-network integrates CAVT and CGAT-Net in an end-to-end approach

In this section, I present the Dual Network illustrated in Figure 5.1, which consists of two main networks: (i) the Class-Agnostic Visio-Temporal Network (CAVT) and (ii) the Context-Aware Graph Attention Transformer Network (CGAT-Net). While both CAVT and CGAT-Net can operate independently in their respective domains—scene sketch segmentation and scene sketch recognition—they also collaborate within a dual-network architecture for end-to-end processing. The workflow begins with CAVT, which segments each object instance within a vector scene sketch by grouping strokes at the instance level. I conducted an ablation study on the use

of temporal stroke order (see Section 8.2.4). However, integrating temporal stroke order did not enhance recognition performance, leading to its exclusion during the classification phase. Consequently, segmented objects are converted into black-and-white bitmap images and input into CGAT-Net, which generates class predictions for each object instance by leveraging inter-object relationships within the scene.

As discussed in Section 3.2, various single-object sketch classifiers can be used to categorize the segmented sketch objects further. To demonstrate the compatibility of CAVT with an external classifier and to highlight the importance of context-based classification in the dual network, I replace CGAT-Net with an external single-sketch classifier in the dual-network setup. I evaluate the performance of several state-of-the-art sketch classifiers, including those proposed by [Xu et al. (2021b); Ha and Eck (2017); Ribeiro et al. (2020)]. Among these, Sketchformer [Ribeiro et al. (2020)] shows superior performance in classifying segmented sketches. Therefore, Sketchformer is also utilized with CAVT in an end-to-end manner. The results of CAVT + Sketchformer and my dual network, CAVT + CGAT-Net, are presented in Section 8.3.1.

Chapter 6

THE FRISS DATASET

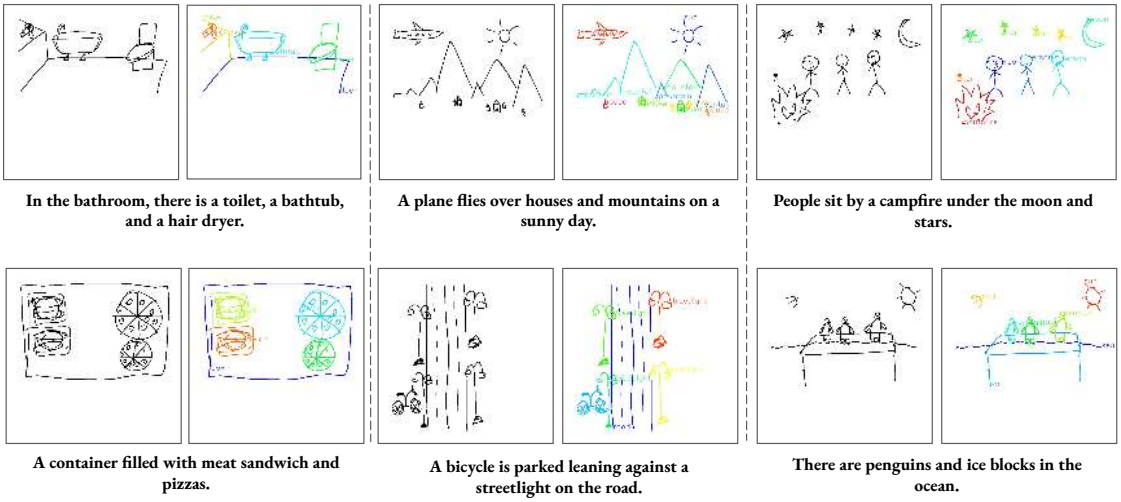
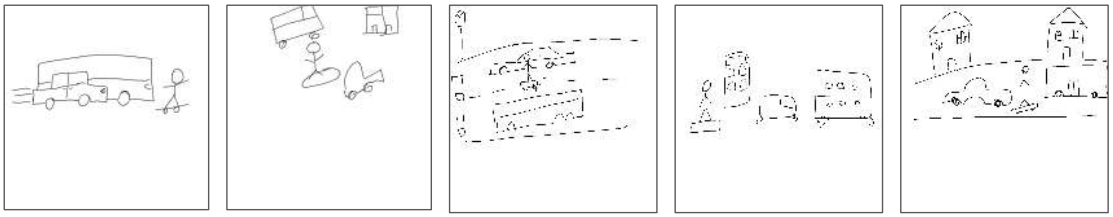


Figure 6.1: Sample scene sketches from the FrISS dataset, each paired with corresponding textual scene descriptions. For each pair, the left image shows the original black-and-white sketch, while the right image highlights the instance and stroke-level class annotations. In the right image, each object instance is annotated at the stroke level with corresponding classes and color-coded accordingly.

The largest Free-hand Instance- and Stroke-level Scene sketch dataset (FrISS) includes stroke-level class annotations, sketch-text pairs, and verbal audio clips paired with each scene, all stored in vector format. The data construction process involves two primary stages: (i) sketch collection and (ii) sketch annotation. This section elaborates on these stages and provides statistics and analysis on the FrISS dataset.

6.1 Sketch Collection

I developed a web application to collect scene sketches, using data collection methods similar to those in previous studies [Chowdhury et al. (2022); Zhang et al. (2023)]. Figures 6.3 and 6.4 show the UI of the web application during the drawing and annotation phases, respectively. For the data collection, I recruited 100 volunteer participants with varying levels of drawing skills, each tasked with creating 10 distinct scene sketches based on textual scene descriptions.



A boy speeds between cars and a bus on his skateboard on the street in a city.

Figure 6.2: Sample scenes from the FrISS dataset that are drawn by five individuals by referring to the same textual scene description

To prevent participants from being influenced by predefined layouts or postures in reference images, which could bias their drawings, I intentionally refrained from providing any visual references. This approach aimed to encourage participants to depict their intended scene sketches freely. As illustrated in Figure 6.2, the diversity and arrangement of objects within the scenes exhibited significant variations when participants were not guided by visual images.

The textual scene descriptions used in this study are either sourced from captions within the MS COCO dataset [Lin et al. (2014)] or constructed by me. I aimed to select or create descriptions featuring at least 3 objects per scene, ensuring that the prompts are straightforward and drawable by individuals without professional drawing skills. However, relying solely on MS COCO image captions was insufficient for covering a broader range of object categories due to the dataset's limited number of unique object categories. To achieve greater variety in the depicted scenes, the majority of descriptions were manually constructed.

Each participant was allocated 1.5 minutes for each scene. The time limit was determined based on pilot studies conducted with few volunteer participants. These studies revealed that shortening this timeframe often resulted in incomplete drawings while extending it led to excessively detailed sketches. While participants could redraw any objects within the time limit, they were not permitted to restart the scene with additional time. Allowing multiple sketching attempts potentially results in unrealistic and perfectly drawn scene sketches. Furthermore, participants were instructed to verbally describe their scenes as they drew, imagining themselves explaining their scenes to an observer nearby. To facilitate comfort and clarity, participants are encouraged to speak in their native language during the description process. Verbal explanations of the participants are recorded during the drawing process. Each scene sketch is paired with verbal audio clips so that this work can promote further research on tasks such as speech-based sketch applications, human-computer interaction, etc.

6.2 Sketch Annotation

In the second phase of data collection, participants were asked to label each stroke with its corresponding object category in scenes they had previously drawn. Instead of asking the participants to label the object classes during the drawing process, I collected sketch annotations separately to avoid disrupting their natural drawing flow. To ensure accurate annotations, this phase was conducted under my supervision. Each stroke in a scene was assigned to its corresponding object category, with incomplete or ambiguous strokes labeled as *'incomplete'* and subsequently excluded from the scene. I also manually reviewed the annotations to verify their accuracy and assess the quality of the scenes. Any mislabeled strokes were either corrected or removed from the dataset.

6.3 User Interface of Data Collection Web Application

In Figures 6.3 and 6.4, I present visuals from the UI of my data collection web application. Figure 6.3 provides an example of the sketch collection phase, where

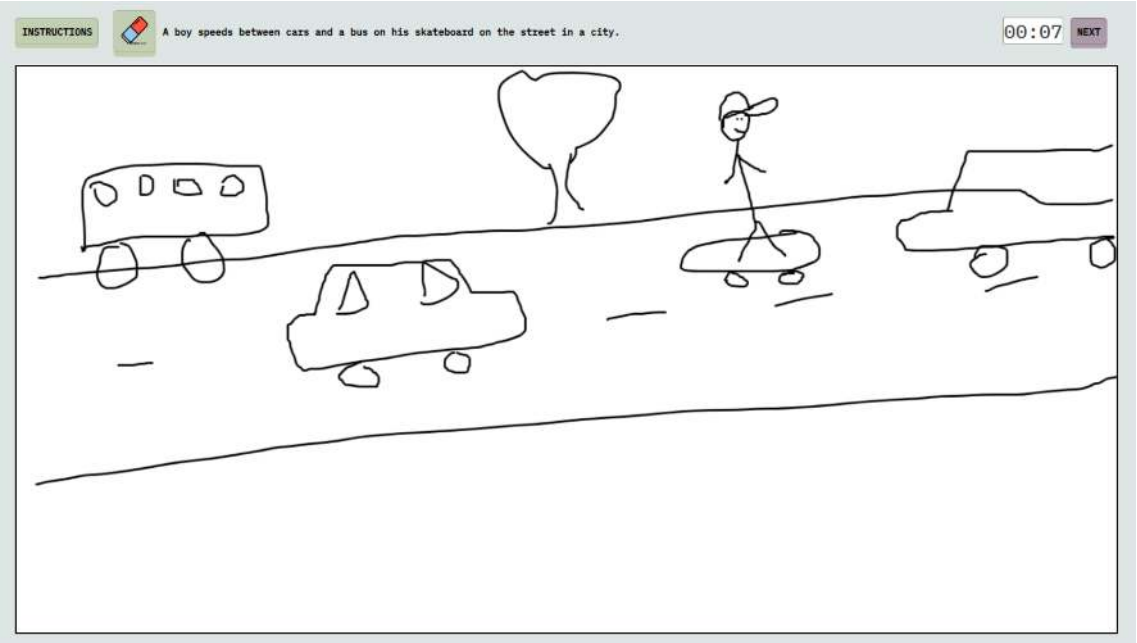


Figure 6.3: The screenshot from the UI of the data collection web application during the drawing phase

participants are tasked with illustrating a scene within a time frame of 1.5 minutes, using a provided text description as a reference. Each participant sequentially draws 10 distinct scene sketches by referring to the corresponding descriptions. Upon completing the sketch collection phase, participants proceed to the second phase, where they annotate their previously drawn sketches.

During the annotation phase, illustrated in Figure 6.4, selected strokes change from *'gray'* to *'black'*, and participants assign a category to each stroke that turns *'black'*. The annotation process continues until every object instance in the scene is labeled. Participants can choose categories from a predetermined list or introduce new categories by entering them into a designated text box (see Figure 6.4). The predetermined list includes all QuickDraw [Ha and Eck (2018)] classes and additional well-known categories not included in QuickDraw but likely to be sketched by participants (e.g., balloon, plate, carpet). This list is provided to ease the labeling process. Finally, strokes that are labeled as incompletely sketched or unrecognizable are marked as *'incomplete'* and excluded from the dataset.

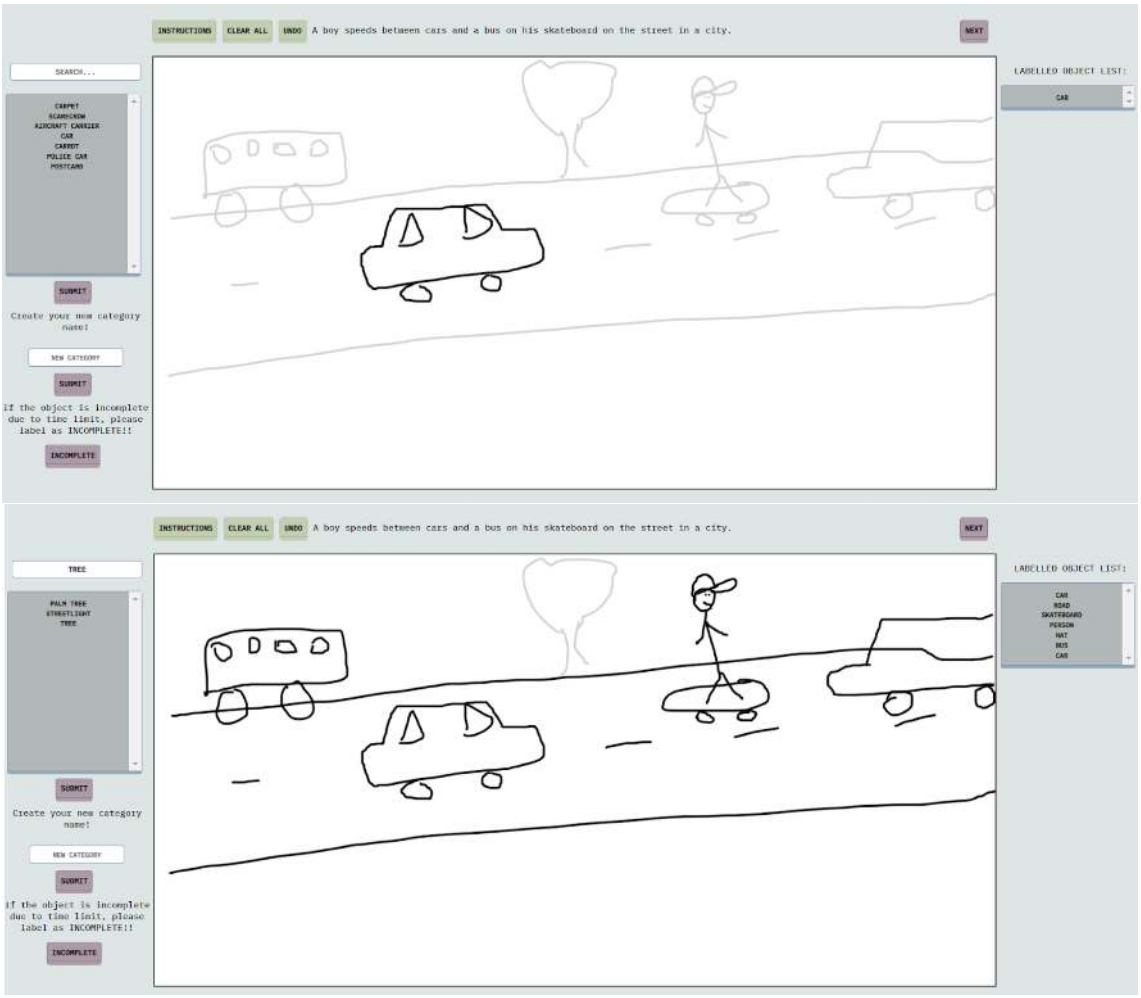


Figure 6.4: The screenshot of data collection UI during the annotation phase. The upper image is taken while labeling the strokes corresponding to the initial object, *'car'*. The lower image is taken before labeling the final drawn object, *'tree'*. Annotated object classes are listed in the upper-right corner of the UI, in the order of labeling.

6.4 Statistics and Analysis

Table 6.4 provides a statistical comparison of various scene sketch datasets, focusing on category, object, and stroke counts per sketch. Among these datasets provided in Table 6.4, CBSC [Zhang et al. (2018)], FS-COCO [Chowdhury et al. (2022)], and SFSD [Zhang et al. (2023)] contain free-hand scene sketches stored in vector format.

Table 6.1: Comparison and statistics of scene sketch datasets

Dataset	# categories	# cat. per sketch			# obj. per sketch			# stroke per sketch		
		Max	Min	Mean	Max	Min	Mean	Max	Min	Mean
SketchyScene [Zou et al. (2018)]	45	19	13	6.88	94	3	16.71	-	-	-
SketchyCOCO [Gao et al. (2020)]	17	6	1	2.33	35	2	10.93	-	-	-
CBSC [Zhang et al. (2018)]	74	10	3	4.23	16	3	4.72	185	6	33.14
FS-COCO [Chowdhury et al. (2022)]	92/150	5/25	1/1	1.37/7.17	-	-	-	561	5	75.86
SFSD [Zhang et al. (2023)]	40	11	1	4.46	43	2	7.76	699	9	146.64
FrISS (Ours)	403	10	1	4.33	30	1	6.04	186	4	35.81

In Figure 6.5, I provide a visual comparison between these and my proposed dataset, FrISS. However, I could only share the visual comparisons with CBSC and FS-COCO, since SFSD is not publicly available. I provide extra sample scene sketches from FrISS with their corresponding textual scene descriptions in the Supplementary Material.

Other free-hand scene sketch datasets [Chowdhury et al. (2022); Zhang et al. (2023)] allow more time for drawings and multiple drawing attempts, which results in extremely detailed scene sketches. CBSC [Zhang et al. (2018)] and FS-COCO [Chowdhury et al. (2022)] are collected under similar conditions: participants are permitted multiple drawing attempts, with an average completion time of 3 minutes per scene. In contrast, I imposed a drawing time limit of 1.5 minutes for each scene, allowing redraw attempts only within this constrained timeframe, without permitting complete redraws. As depicted in Table 6.5, FrISS contains an average of approximately 36 strokes per scene, significantly fewer than other datasets in complexity.

Furthermore, during the collection of FS-COCO [Chowdhury et al. (2022)], participants were presented with natural images as references during the drawing process. However, providing reference images results in scene sketches being in similar object positions and postures as those in the referenced images. On the other hand, the CBSC [Zhang et al. (2018)] dataset was collected by instructing participants

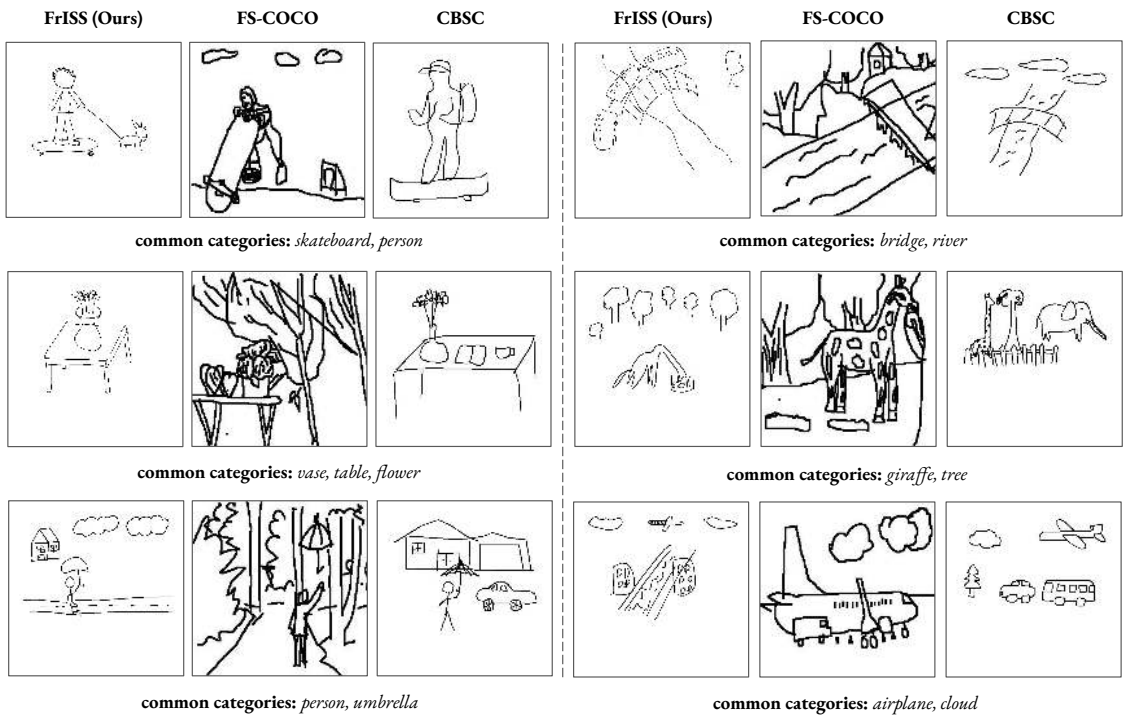


Figure 6.5: A comparison of scene sketches from FrISS with those from FS-COCO [Chowdhury et al. (2022)] and CBSC [Zhang et al. (2018)]. The visuals are selected to ensure that each set of scene sketches shares at least two object categories in common, with the common classes listed below each group of three.

to quickly draw simple scene sketches that convey semantic meaning to humans, without any visual references. As a result, CBSC contains scene sketches that are similar in object complexity to those in FrISS. However, FrISS is distinguished by its larger scale and more detailed annotations. While CBSC includes 331 scene sketches across 74 object categories, FrISS offers 1,000 freehand scene sketches, encompassing a broader range of 403 object categories. Furthermore, as shown in Figure 6.1, FrISS pairs scene sketches with their textual descriptions and annotates each object instance at the stroke level with corresponding classes, offering unique multi-modal annotations.

Chapter 7

RESULTS & DISCUSSION

In this section, I present the details of the training process and provide an in-depth discussion of the experimental results for two individual networks, CAVT and CGAT-Net, as well as the Dual Network that integrates both in an end-to-end manner.

7.1 Datasets

I rely on the temporal stroke order within scene sketches, which restricts the range of suitable datasets for evaluation. While FS-COCO [Chowdhury et al. (2022)] is the first large-scale free-hand scene sketch dataset, the authors did not provide stroke-level class annotations within scenes. Consequently, I evaluated the performance of CAVT and CGAT-Net using only two publicly available free-hand scene sketch datasets: the FrISS dataset, which I introduced, and the CBSC dataset proposed by Zhang et al. (2018). Figure 6.5 offers a visual comparison of publicly available free-hand scene sketch datasets. A summary of the FrISS and CBSC datasets is provided below:

- **FrISS:** This dataset contains 1,000 free-hand scene sketches across 403 object categories, with all scenes stored in vector format. Each sketch includes instance- and stroke-level class annotations, making it suitable for stroke-based applications at the scene-level. For CGAT-Net, 500 scene sketches were reserved for testing, while the remaining sketches were divided into validation (145 sketches) and training (355 sketches) sets.

- **CBSC:** This dataset comprises 332 free-hand scene sketches across various scene contexts (222 for test, 110 for validation partitions), such as dining

room, home, outdoor activities, nature scenes, and street views. CBSC provides instance-level class annotations and covers 74 object categories, which align with those in the QuickDraw dataset [Ha and Eck (2018)], except for the *'person'* class. However, the visual characteristics of the *'yoga'* class in QuickDraw closely resemble those of the *'person'* class in other scene sketch datasets. Therefore, I mapped the *'person'* class to the *'yoga'* class during network evaluation.

In the following parts, I compare the performance of my networks with the models publicly available in the literature. However, these models are trained on different datasets and to compare them with my study, various subsets of CBSC and FrISS need to be generated. These variations are listed below:

- **FrISS-SKY and CBSC-SKY:** The Dual Network is compared with the Local Detail Perception (LDP) method proposed by Ge et al. (2022). However, since LDP is only trained on the SKY-Scene or SketchyScene datasets, I filter the FrISS and CBSC scene sketches to ensure a fair comparison. Only the common classes supported by both LDP and CGAT-Net are included in the scenes. This variation focuses on scene sketches containing objects from classes shared by SKY-Scene and CGAT-Net.

- **FrISS-SS and CBSC-SS:** This variation includes scene sketches featuring objects that belong to the common categories between the SketchyScene and CGAT-Net classes.

- **FrISS-QD:** All single sketch classifiers that are compared in this thesis are trained on the QuickDraw dataset. To enable a fair comparison with CGAT-Net, a variation of FrISS was created that includes only the classes common to both QuickDraw and CGAT-Net. Since all CBSC classes exist in QuickDraw and supported by CGAT-Net (*"person"* class is mapped to *"yoga"*), there exists no extra variation for CBSC.

- **FrISS-CG:** Not all categories in the original FrISS dataset are supported by CGAT-Net, as CGAT-Net is trained only on classes mapped from MS COCO [Lin et al. (2014)] to QuickDraw, FrISS, and SketchNet [Dede (2023)]. Therefore, a subset of FrISS was generated that includes only the objects from categories supported by CGAT-Net. Since all categories in CBSC are already supported by CGAT-Net, no additional modifications were needed for that dataset.

7.2 Performance Analysis on CAVT

7.2.1 Evaluation Metrics

There is no available metric specifically designed for instance-level scene sketch segmentation that operates in vector format. Thus, I proposed two metrics for the evaluation at the stroke level:

- **All or Nothing (AoN):** evaluates the percentage of correctly predicted stroke groups. Any mislabeling of a single stroke within an object results in the prediction being marked as incorrect.

- **Stroke-level Intersection over Union (S-IoU):** calculates the largest overlap ratio of the actual and the predicted stroke groups, and averages the overlap ratio for all ground truth stroke groups.

These metrics can be used to evaluate instance-level segmentation outputs for vector sketches. However, existing scene sketch segmentation models perform the segmentation of sketch objects at the class level and only produce bitmap outputs. Therefore, earlier works could not be evaluated using these metrics. I share the AoN and S-IoU results only for CAVT in Tables 8.1 and 8.2.

7.2.2 Implementation Details

The sole trainable component of CAVT is the Class-Agnostic Visio-Temporal Object Detector, built upon the YOLOX framework [Ge et al. (2021)]. During model

training, I employed the MMDetection library, training YOLOX with default configurations while modifying only the total number of categories to 1. The training process utilizes a single Tesla T4 GPU with a batch size of 16, spanning 600 epochs.

7.2.3 Post-Processing Time & Memory Footprint

The post-processing module takes on average 345 milliseconds per scene on CPU and has the memory upper bound of 5 times the scene in vector format.

7.2.4 Quantitative Results

CAVT is the first instance-level scene sketch segmentation network that generates stroke groupings for each object instance. However, since CAVT lacks a classification module, I integrate CGAT-Net for the sketch recognition phase. As a result, CAVT cannot be directly compared with existing state-of-the-art scene segmentation methods; this comparison is addressed in the performance analysis of the Dual Network in Section 7.4.3.

Table 7.1: The result of CAVT to assess its stroke-level grouping performance

Model	**CBSC**		**FrISS**	
	AoN	**S-IoU**	**AoN**	**S-IoU**
CAVT	68.43	84.81	53.03	74.12

In Table 7.1, I present the AoN and S-IoU results of CAVT to evaluate its instance-level stroke grouping performance. Given that these metrics do not apply to existing works due to their class-level structure, the results are provided specifically for CAVT.

7.3 Performance Analysis on CGAT-Net

7.3.1 Evaluation Metrics

I used the well-known *'top-K accuracy'* evaluation metric for the sketch recognition task. This metric measures the ratio of correctly classified sketch objects to the total objects in the dataset. A prediction is considered correct if the correct object category is among the model's top-K predictions for a given instance. I calculated top-K accuracy for the values of K = 1, 3, and 5. These results are provided in Table 7.2.

7.3.2 Implementation Details

The model and training loop are implemented using the PyTorch library, with the Graph Transformer Layer inspired by MGT [Xu et al. (2021b)]. CGAT-Net is trained for 60,000 iterations on a single Tesla V100 GPU with a batch size of 16. The best-performing checkpoints from the FrISS-QD and CBSC [Zhang et al. (2018)] validation sets are selected as the final model weights.

7.3.3 Quantitative and Qualitative Results

I provide quantitative results of CGAT-Net and SOTA single object sketch classifiers. As can be seen from Table 7.2, the highest-performing Transformer-based classifier, Sketchformer [Ribeiro et al. (2020)], is outperformed by Inception-V3 [Szegedy et al. (2016)], a CNN-based architecture trained on single object sketches from the Quick-Draw [Ha and Eck (2018)] and FrISS datasets. Inception-V3 serves as the backbone for CGAT-Net, chosen through an ablation study detailed in Section 8.2.1. The recognition performance comparison that is conducted on the CBSC and FrISS-QD datasets, demonstrates that CGAT-Net surpasses SOTA single sketch classifiers under identical conditions [Li et al. (2020); Xu et al. (2021b); Ribeiro et al. (2020); Szegedy et al. (2016)]. CGAT-Net achieves superior results, outperforming Sketchformer by approximately 8.14% on the CBSC and 7.49% on the FrISS dataset. This trend is consistent across top-3 and top-5 accuracy measures as well. Overall,

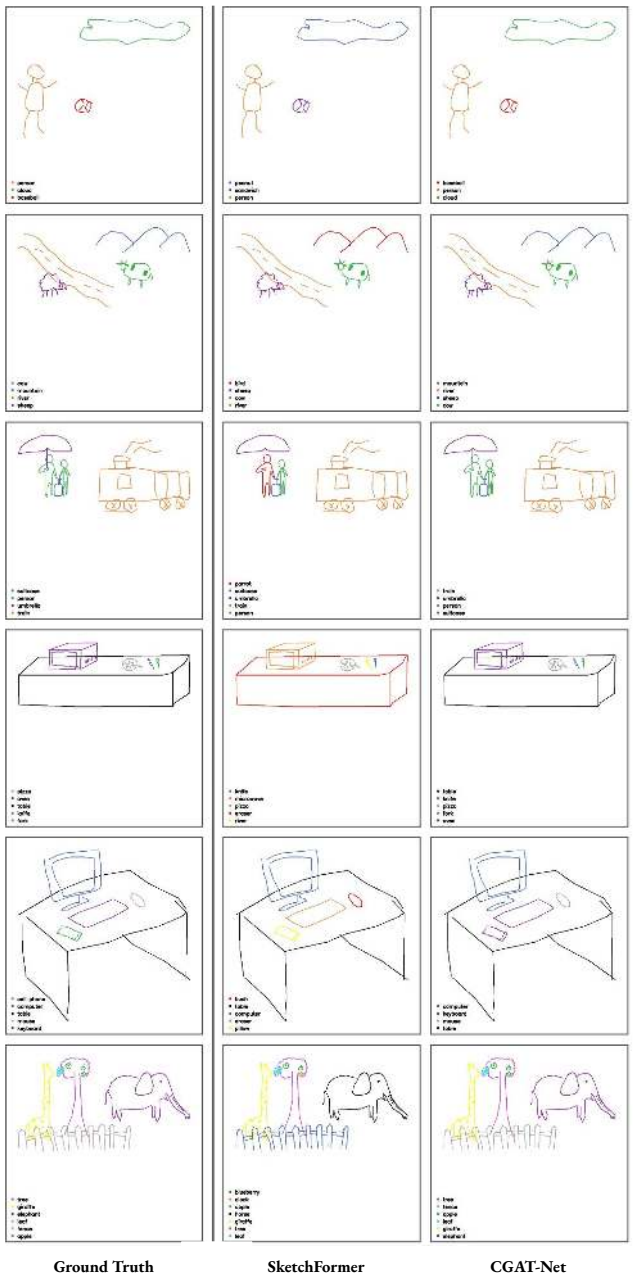


Figure 7.1: Visual comparison of CGAT-Net with Sketchformer [Ribeiro et al. (2020)], tested on CBSC dataset.

CGAT-Net establishes a new state-of-the-art benchmark for top-1, top-3, and top-5 accuracies.

I present a visual comparison between CGAT-Net and Sketchformer on the CBSC

Figure 7.2: Visual comparison of CGAT-Net with Sketchformer [Ribeiro et al. (2020)], tested on FrISS-QD dataset.

and FrISS-QD in Figures 7.1 and 7.2, respectively. Both models generate object class predictions for each object within a scene, evaluated on pre-segmented scene objects. In Figure 7.1, the fourth row highlights a case where Sketchformer incorrectly

Table 7.2: Comparison of CGAT-Net against SOTA single sketch classifiers: SketchR2CNN [Li et al. (2020)], MGT [Xu et al. (2021b)], Sketchformer [Ribeiro et al. (2020)], Inception-V3 [Szegedy et al. (2016)].

Model	Top-1 Accuracy			Top-3 Accuracy			Top-5 Accuracy		
	CBSC	FrISS-QD	Avg.	CBSC	FrISS-QD	Avg.	CBSC	FrISS-QD	Avg.
SketchR2CNN	63.04	48.65	55.85	71.57	59.13	65.35	74.12	63.06	68.59
MGT	65.29	51.78	58.54	79.22	67.85	73.54	83.63	73.40	78.52
Sketchformer	65.88	52.82	59.35	80.81	66.57	73.69	85.69	71.36	78.53
Inception-V3	67.45	55.48	61.47	82.84	70.27	76.56	86.04	74.62	80.33
CGAT-Net	**74.02**	**60.31**	**67.17**	**84.41**	**72.36**	**78.39**	**87.75**	**76.24**	**82.00**

predicts a *'table'* as an *'eraser'*, whereas CGAT-Net produces more contextually appropriate class predictions. This demonstrates CGAT-Net's ability to utilize inter-object spatial relations to enhance recognition accuracy, as well as its understanding of the relative size of a *'table'* compared to surrounding objects. Another example, shown in Figure 7.2 in the first row, depicts a birthday party scene. Here, Sketchformer misclassifies three *'cookie'* objects as *'pillow'*, *'door'*, and *'stairs'*. In contrast, CGAT-Net misclassifies two *'cookie'* objects, with incorrect predictions such as *'cup'* and *'cake'*, which are more contextually relevant to the scene. Similar cases are seen in other examples in both figures as well. This illustrates that even when CGAT-Net makes mistakes, its predictions still align more closely with the scene's context and relative size information. Thus, CGAT-Net effectively prioritizes contextually relevant classes by utilizing object relationships and spatial information within the scene. Additional visual results are shared in the Supplementary Material A.3.

7.4 Performance Analysis on Dual Network

7.4.1 Evaluation Metrics

Scene sketch segmentation studies mostly treat sketches as images and utilize metrics commonly used to evaluate image segmentation models. There are no specific

evaluation metrics that are designed for sketch segmentation studies and operating at the stroke level. Hence, I followed the standard four metrics that are used in the existing segmentation models for fair comparison [Bourouis et al. (2023); Ge et al. (2022); Zou et al. (2018)]. Each metric is explained as follows:

- **Overall Pixel Accuracy (OVAcc):** evaluates the ratio of correctly predicted pixels over the pixels in the scene sketch.
- **Mean Pixel Accuracy (MeanAcc):** calculates the average pixel accuracy for each category.
- **Mean Intersection over Union (MIoU):** calculates the average per-class ratio of the Intersection over Union (IoU) of ground truth and predicted labels.
- **Frequency Weighted Intersection over Union (FWIoU):** the weighted MIoU metric, which adjusts the per-class pixel IoU by the frequency of class appearance.

7.4.2 Implementation Details

The proposed Dual Network is a two-stage scene sketch semantic segmentation pipeline with separate segmentation (i.e., CAVT) and classification (i.e., CGAT-Net) modules. Therefore, I compare my overall framework with earlier publicly available state-of-the-art scene sketch semantic segmentation networks: Local Detail Perception (LDP) [Ge et al. (2022)] and Open Vocabulary (OV) [Bourouis et al. (2023)]. However, to enable a performance comparison between my framework and these models, several adjustments to the datasets and the evaluation process are necessary:

- LDP can only classify object categories provided in the SKY-Scene [Ge et al. (2022)] and SketchyScene [Zou et al. (2018)]. In contrast, CGAT-Net can categorize 194 object classes, which do not fully cover the categories of SKY-Scene and SketchyScene. To ensure a fair comparison, I created two subsets of

CBSC and FrISS datasets: CBSC-SKY and FrISS-SKY, CBSC-SS and FrISS-SS.

- The OV model does not rely on pixel- or stroke-level annotations; instead, it uses sketch-caption pairs. During inference, captions are generated by concatenating ground truth object categories, and OV predicts the correct class label from the given set of object classes. Therefore, I also devised another variant of my recognition pipeline (CAVT + CGAT-Net*) that limits the possible object classes to the ground truth categories in the scene for a fair comparison with the OV model. In addition to the subsets above, I evaluate both OV and CGAT-Net with FrISS-CG and complete CBSC, since categories in these datasets are supported in OV.

7.4.3 Quantitative and Qualitative Results

The comparison results of dual-network (CAVT + CGAT-Net) with prior works on the different subsets of CBSC [Zhang et al. (2018)] and FrISS datasets are shown in Tables 7.3, 7.4, and 7.5. In each dataset variation and metric, I outperform both LDP [Ge et al. (2022)] and OV [Bourouis et al. (2023)] under the same conditions. While my gap with LDP is on average 25.36% for OVAcc, 28.64% for MeanAcc, 25,83% for MIoU, and 27.92% for FWIoU (LDP vs. CAVT + CGAT-Net); it drops to 11.45% for OVAcc, 9.93% for MeanAcc, 15.84% for MIoU, and 14.78% for FWIoU with OV (OV vs. CAVT + CGAT-Net*). Still, my Dual Network achieves the best performance with a significant difference.

Additionally, Figures 7.3 and 7.4 provide a qualitative comparison between the Dual Network, LDP, and OV models. In Figure 7.3, the fourth row illustrates that both LDP and OV models produce pixel-level predictions and fail to segment *'cloud'* objects effectively. In contrast, CAVT utilizes stroke order information, ensuring that different class labels are not assigned to any point within a single stroke. Another example is shown in the first row of Figure 7.4, where CGAT-Net successfully segments and classifies each object at the stroke level, while other models fail to produce coherent results. This indicates that our approach results in

Table 7.3: Comparison of the dual network against SOTA scene sketch semantic segmentation methods (i.e., LDP [Ge et al. (2022)] and OV [Bourouis et al. (2023)]) on FrISS-SS and FrISS-SKY.

Model	FrISS-SS				FrISS-SKY			
	OVAcc	**MeanAcc**	**MIoU**	**FWIoU**	**OVAcc**	**MeanAcc**	**MIoU**	**FWIoU**
LDP	39.50	33.54	16.78	26.93	45.21	37.21	20.36	32.41
CAVT + CGAT-Net	**63.85**	**56.75**	**35.39**	**50.97**	**66.94**	**55.80**	**34.52**	**55.00**
Open Vocabulary	82.23	79.51	65.11	70.77	90.81	**89.75**	77.73	84.31
CAVT + CGAT-Net*	**88.10**	**81.83**	**72.45**	**79.33**	**95.13**	89.53	**84.91**	**91.40**

Table 7.4: Comparison of the dual network against SOTA scene sketch semantic segmentation methods (i.e., LDP [Ge et al. (2022)] and OV [Bourouis et al. (2023)]) on CBSC-SS and CBSC-SKY.

Model	CBSC-SS				CBSC-SKY			
	OVAcc	**MeanAcc**	**MIoU**	**FWIoU**	**OVAcc**	**MeanAcc**	**MIoU**	**FWIoU**
LDP	48.81	39.05	24.78	33.80	54.10	46.91	28.23	38.43
CAVT + CGAT-Net	**78.33**	**78.13**	**59.80**	**67.25**	**79.92**	**80.57**	**63.75**	**70.03**
Open Vocabulary	78.32	73.21	59.31	66.66	79.39	77.65	64.44	69.01
CAVT + CGAT-Net*	**91.13**	**89.52**	**81.24**	**84.36**	**96.35**	**95.56**	**91.10**	**93.18**

more coherent segmentation outputs. On the other hand, LDP and OV generally struggle with segmenting individual object strokes within a scene, emphasizing the significance of stroke-based segmentation methods.

In some instances, although CAVT effectively segments object instances, performance can be affected by CGAT-Net's misclassifications. For example, in the second row of Figure 7.3, CAVT accurately segments a *'chair'* object, but CGAT-Net misclassifies it as a *'bird'*. During the comparison with the OV model, as detailed

Table 7.5: Comparison of the dual network with Open Vocabulary [Bourouis et al. (2023)] on both the CBSC [Zhang et al. (2018)] and FrISS-CG datasets.

Model	**CBSC**				**FrISS-CG**			
	OVAcc	**MeanAcc**	**MIoU**	**FWIoU**	**OVAcc**	**MeanAcc**	**MIoU**	**FWIoU**
OV	60.44	57.64	40.69	47.29	66.77	58.37	41.16	53.04
CAVT + CGAT-Net*	**81.38**	**77.82**	**65.41**	**70.16**	**74.56**	**61.47**	**48.38**	**61.34**
CAVT + CGAT-Net	65.37	62.14	43.65	52.66	47.28	33.94	18.19	35.24

in Section 7.4.2, I filtered the possible classes to match the scene-specific classes, leading CGAT-Net to misclassify the *'chair'* as a *'person'*. These misclassifications influence the overall class-level segmentation results. Therefore, improvements in CGAT-Net's performance could yield more accurate class-level segmentation outputs. Additional visual comparisons are available in Supplementary Material B.

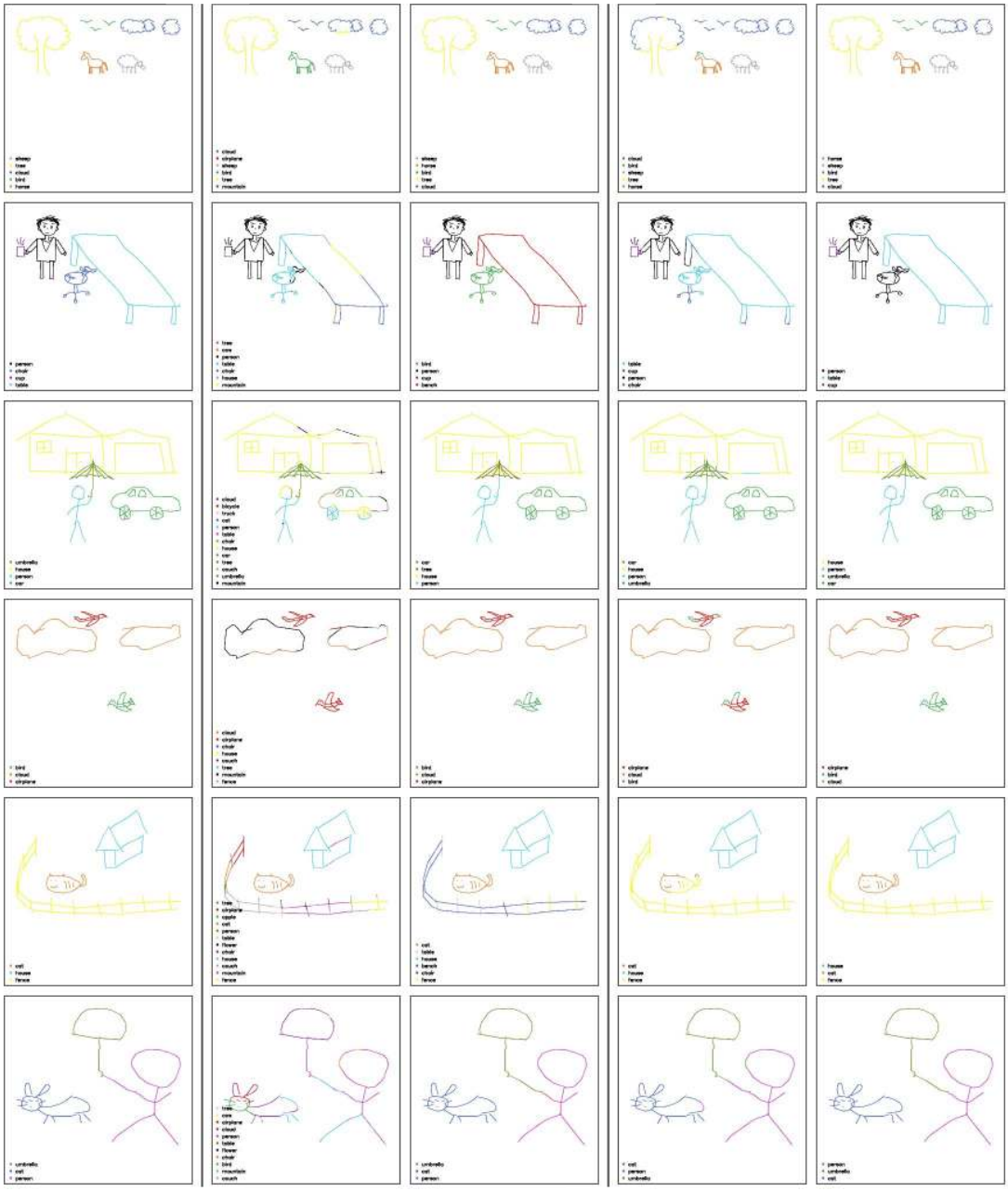


Figure 7.3: Visual comparison of Dual Network with LDP [Ge et al. (2022)] and OV [Bourouis et al. (2023)] models, tested on CBSC dataset. CAVT+CGAT-Net*: refers to predictions retrieved from the Dual Network only from the object classes present in the given scene, to ensure a fair comparison with OV.

Figure 7.4: Visual comparison of Dual Network with LDP [Ge et al. (2022)] and OV [Bourouis et al. (2023)] models, tested on FrISS dataset. CAVT+CGAT-Net*: refers to predictions retrieved from the Dual Network only from the object classes present in the given scene, to ensure a fair comparison with OV.

Chapter 8

ABLATION STUDY

8.1 Experiments on CAVT

In this section, I conduct two different experiments on my segmentation network, CAVT. While the first one measures the effects of temporal stroke order utilization, class-agnostic training, and post-processing; the second experiment deals with the hyperparameter selection on the post-processing module.

8.1.1 Effect of Individual Components

In this experiment, I examine the individual effects of each component within CAVT. The key components include the use of temporal stroke order, class-agnostic training, and the post-processing module. To evaluate the impact of the post-processing module, I implement a simple stroke grouping technique as a baseline for comparison. In this method, each stroke is assigned to the nearest predicted bounding box, and strokes assigned to the same box are grouped as a single object.

Table 8.1 illustrates the impact of each component on segmentation performance, with each one contributing a notable improvement. The contributions range from a minimum of approximately $\sim 1.5\%$ in AoN and $\sim 1.9\%$ in S-IoU to a maximum increase of around $\sim 17\%$ in AoN and $\sim 9.3\%$ in S-IoU. Among the components, the post-processing module (PP) has the most significant effect on the performance, while class-agnostic training (CA) has the least. Additionally, including temporal stroke order (T) increases the performance by at least 9.95% in AoN and 3.91% in S-IoU.

As detailed in Section 3.1, the object detector is trained using a synthetic dataset derived from QuickDraw classes [Ha and Eck (2018)]. I excluded instances belonging to QuickDraw classes to evaluate CAVT's generalizability to instances from unseen

Table 8.1: Ablation study for the impact of including or excluding three components: temporal stroke order (T), class-agnostic training (CA), and post-processing steps (PP). FrISS^{sub}: calculates metrics for a subset of categories in FrISS that are not part of QuickDraw [Ha and Eck (2018)].

Components			**CBSC**		**FrISS**		**FrISS^{sub}**	
T	**CA**	**PP**	**AoN**	**S-IoU**	**AoN**	**S-IoU**	**AoN**	**S-IoU**
			39.8	68.76	23.03	59.2	24.73	53.04
		✓	56.86	78.06	37.57	62.91	30.94	55.78
	✓	✓	58.43	80.9	39.26	64.81	33.01	57.5
✓	✓	✓	**68.43**	**84.81**	**53.03**	**74.12**	**42.96**	**74.27**

classes during the evaluation. I calculated AoN and S-IoU only for instances with class labels different from the QuickDraw categories (denoted as FrISS^{sub}). The performance drops by approximately $\sim 10\%$, from 53.03% to 42.96% on the FrISS dataset. However, there is a slight improvement between FrISS and FrISS^{sub} for $S-IoU$ results, indicating that CAVT can generalize to sketch objects from unseen classes with minimal performance loss.

8.1.2 Hyperparameter Optimization for the Post-Processing Module

To determine the best hyperparameter combination for the post-processing module, a grid-search algorithm was employed. I evaluated the AoN and S-IoU scores on the validation sets of both CBSC and FrISS and selected the top-performing parameter combination based on the average of all scores. Table 8.2 presents the results for the top-performing parameter combination. The parameters and their corresponding ranges evaluated in the ablation study are as follows:

- ***IoU_threshold:*** The threshold value determines the Intersection over Union

(IoU) of stroke sequences to boxes. For each box, if the IoU between the box and the longest intersecting stroke sequence exceeds *IoU_threshold*, the sequence is assigned to that box. For the ablation study, I adjusted the threshold within a range of 25% to 85%, increasing by 10% increments.

- ***OR_threshold:*** This is the threshold value that determines the assignment of remaining stroke sequences to boxes. If the overlap ratio of the longest unassigned stroke sequence with its nearest box exceeds *OR_threshold*, the sequence is assigned to that box. For the ablation study, I set the threshold ranges from 30% to 80% in 5% increments.

- ***num_repeats:*** This refers to the total number of iterations the post-processing module undergoes to complete the stroke assignment process. As described in Section 3.2, the post-processing module continues until stroke group assignments reach a stable state. However, this approach can increase runtime, so I limited the number of iterations to evaluate the impact of different repetition counts. I tested the effect of the *num_repeats* parameter with values of 1, 3, 5, 7, and 9.

- ***stroke_thickness:*** I assessed the effect of stroke line thickness by evaluating the *stroke_thickness* parameter with values of 1, 2, 3, and 4, where higher values correspond to thicker stroke lines in the scene.

Table 8.2 illustrates the impact of each parameter, revealing that the best-performing hyperparameter combination includes these values: *IoU_threshold* set to 65%, *OR_threshold* to 60%, *num_repeats* to 3, and *stroke_thickness* to 2. As demonstrated, using a value for *stroke_thickness* different than 2 degrades performance by distorting the features of the sketches. The *num_repeats* parameter does not significantly affect performance when increased, indicating that the stroke assignment operation completes effectively within a few iterations, minimizing the need for extended runtime. Setting *OR_threshold* to a low percentage can lead to incorrect stroke assignments, as some strokes that should be labeled as separate objects are

merged with other stroke sequences. Therefore, setting *OR_threshold* higher than 50% generally results in better performance. A range of 55%-65% for *IoU_threshold* yields the best results. Lower *IoU_threshold* values can lead to incorrect stroke-to-box assignments, while higher values may hinder the accurate assignment of strokes to their corresponding boxes.

Table 8.2: The top-performing hyperparameter combinations for the post-processing module are presented in descending order.

Index	***IoU_threshold***	***OR_threshold***	***num_repeats***	***stroke_thickness***	**CBSC**		**FrISS**		**Avg**
					AoN	**S-IoU**	**AoN**	**S-IoU**	
1	65%	60%	3	2	74,17	88,19	57,96	79,20	**74,88**
2	75%	45%	1	2	73,53	87,56	59,37	78,98	74,86
3	55%	60%	3	2	74,17	88,19	57,71	79,32	74,85
4	65%	75%	3	2	73,56	87,94	58,08	79,25	74,71
5	65%	70%	3	2	73,56	87,94	58,10	79,16	74,69
6	55%	55%	5	2	74,07	88,11	57,38	78,99	74,64
7	55%	70%	3	2	73,56	87,94	57,85	79,18	74,63
8	25%	60%	3	2	74,40	88,47	56,67	78,99	74,63
9	45%	60%	3	2	74,17	88,27	56,79	79,11	74,59
10	65%	70%	1	2	73,43	88,02	57,75	79,05	74,56
11	25%	75%	3	2	73,79	88,20	57,03	78,96	74,49
12	45%	75%	3	2	73,56	88,00	57,14	79,09	74,45
13	75%	70%	1	2	72,69	87,64	58,45	78,97	74,44
14	25%	70%	3	2	73,79	88,20	56,81	78,85	74,41
15	35%	75%	3	2	73,34	87,97	57,14	79,09	74,38
16	75%	50%	1	2	72,89	87,61	58,32	78,58	74,35
17	55%	50%	5	2	73,76	87,84	57,02	78,75	74,34
18	35%	75%	1	2	73,21	88,05	56,68	79,17	74,28
19	55%	50%	1	2	73,63	87,89	56,67	78,81	74,25
20	85%	75%	3	1	73,23	87,41	58,40	77,78	74,21
Lowest	85%	50%	7	3	66,97	83,00	53,17	75,74	69,72

8.2 Experiments on CGAT-Net

In this section, I conduct four experiments on my sketch recognition network, CGAT-Net. While the first experiment investigates different backbone architectures for the model, the second one analyzes the model structure of CGAT-Net and its training hyperparameters, the third one measures the effect of different graph attention matrices, and the last experiment discusses the preprocessing settings. A subset of the FrISS dataset, referred to as FrISS-QD, is selected for these experiments. Since state-of-the-art single-sketch classifiers are typically trained on QuickDraw [Ha and Eck (2018)], FrISS-QD is used to facilitate direct comparisons between our experiments and these SOTA classifiers.

Table 8.3: The ablation study is performed to measure the effect of different backbone architectures and the effect of including the FrISS dataset in the training set. The highest average score is highlighted in green, the second highest in blue, and the third highest in red for each aspect (i.e., backbone type and FrISS contribution).

Model	**Train Dataset**		**Accuracy**		
	QD	**FrISS**	**CBSC**	**FrISS-QD**	**Avg.**
Inception-V3 [Szegedy et al. (2016)]	✓		65.69	50.07	57.88
VGG19 [Simonyan (2014)]	✓		64.02	50.69	57.36
ResNet18 [He et al. (2016)]	✓		63.04	48.79	55.92
ResNet50 [He et al. (2016)]	✓		62.84	48.03	55.44
MobileNetV3-Small [Howard et al. (2019)]	✓		61.37	46.85	54.11
MobileNetV3-Large [Howard et al. (2019)]	✓		60.88	48.89	54.89
MobileNet-V2 [Sandler et al. (2018)]	✓		62.55	47.75	55.15
Inception-V3	✓	✓	67.45	55.48	**61.47**
VGG19	✓	✓	65.98	55.24	60.61
ResNet18	✓	✓	67.65	53.11	60.38

8.2.1 Effect of the Backbone Network and Contribution of FrISS

In this experiment, various CNN-based classifiers are trained to investigate different backbone architectures. Each CNN classifier is trained with QuickDraw [Ha and Eck (2018)] and evaluated on CBSC [Zhang et al. (2018)] and FrISS-QD. Moreover, the effect of adding sketch objects from FrISS to the training set is also investigated. Table 8.3 shows the results of the ablation studies.

Among the classifier architectures, the best recognition performance is achieved with Inception-V3 [Szegedy et al. (2016)]. Therefore, Inception-V3 is selected as the Sketch Visual Feature Extractor backbone architecture, with further details provided in Section 4.2. Additionally, the inclusion of FrISS sketches enhances performance on both FrISS-QD and CBSC. While QuickDraw features single objects without contextual interaction, FrISS sketches depict objects within scenes, interacting with other objects. Thus, QuickDraw objects tend to be standalone recognizable due to their nature but FrISS can include more abstract representations of the same object classes. Including FrISS in the training set allows the model to learn these representations as well.

8.2.2 Analysis of the Model Structure and Training Settings

In this part, I tested the effect of using different backbones, utilizing a pre-trained CNN backbone, experimenting with different learning rates, and changing the number of transformer layers in CGAT-Net. Table 8.4 presents the results of these tests. My findings on this part are listed below:

- Indices 1, 2, and 3 in Table 8.4 measures the effect of different CNN-based backbone models. Parallel to my observation in Subsection 8.2.1, utilizing Inception V3 as the backbone results in the best performance.

- In indices 4, 5, 6, and 7, I investigate the effect of an intermediate classification loss that is calculated from the backbone outputs. While adopting this loss to the training improves the overall performance (see index 5), a loss weight of 0.5 suits better than 1.0 (index 6 and 7).

Table 8.4: The ablation study on the design choices of model architecture. *B_type* is the backbone model type for CGAT-Net Sketch Visual Feature Extractor (discussed in Section 4.2), *B_pre-trained* indicates if the backbone is initialized from pre-trained weights or scratch during the training phase, *B_freeze* shows if the backbone is frozen or updated during the training phase, *B_class-probs* represents whether to use un-normalized class probabilities of the pre-trained backbone or the feature embeddings taken from an intermediate layer, *B_loss* is the weight of intermediate classification loss that is calculated from the backbone outputs, *LR* refers to the learning rate, and *Num_layers* is the number of transformer layers in CGAT-Net (*L* refers to the *Num_layers* in Figure 4.1). Results in 4, 8, 15, and 18 belong to the same model. This model is selected as our control model and added to each design choice part for easiness. For each design choice, the highest score is highlighted in green, the second highest in blue, and the third highest in red.

Index	*B_type*	*B_pre-trained*	*B_freeze*	*B_class-probs*	*B_loss*	*LR*	*Num_layers*	CBSC	FrISS-QD	Avg.
1	VGG				0.0	1,00E-04	2	40.29	31.67	35.98
2	Resnet18				0.0	1,00E-04	2	66.08	51.64	58.86
3	Inception-V3				0.0	1,00E-04	2	68.53	55.24	61.89
4	Inception-V3				0.0	1,00E-04	1	68.63	57.14	62.89
5	Inception-V3				0.5	1,00E-04	1	71.47	56.90	64.19
6	Inception-V3				0.5	1,00E-04	2	67.75	54.67	61.21
7	Inception-V3				1.0	1,00E-04	2	66.67	54.81	60.74
8	Inception-V3				0.0	1,00E-04	1	68.63	57.14	62.89
9	Inception-V3				0.0	1,00E-04	2	68.53	55.24	61.89
10	Inception-V3				0.0	1,00E-04	3	68.04	53.25	60.65
11	Inception-V3				0.0	1,00E-04	4	67.65	50.07	58.86
12	Inception-V3				0.0	1,00E-04	5	65.20	50.26	57.73
13	Inception-V3				0.0	1,00E-03	1	11.67	17.50	14.59
14	Inception-V3				0.0	5,00E-04	1	10.98	16.55	13.77
15	Inception-V3				0.0	1,00E-04	1	68.63	57.14	62.89
16	Inception-V3				0.0	5,00E-05	1	69.12	55.62	62.37
17	Inception-V3				0.0	1,00E-05	1	61.37	48.79	55.08
18	Inception-V3				0.0	1,00E-04	1	68.63	57.14	62.89
19	Inception-V3	✓			0.0	1,00E-04	1	67.94	55.48	61.71
20	Inception-V3	✓	✓		0.0	1,00E-04	1	67.45	52.30	59.88
21	Inception-V3	✓	✓	✓	0.0	1,00E-04	1	61.67	54.39	58.03

- Between the 8th and 12th experiments, CGAT-Net is initialized with a different number of transformer layers. As seen from the Table, increasing the number of layers decreases the overall performance. The best score is taken when the number of layers is set to 1.

- The influence of learning rate is analyzed between the indices 13-17. While the range between 1e-4 and 5e-5 is suitable for CGAT-Net, I select 1e-4 for the next steps, since it provides the greatest average performance.

- In the last experiment between indices 18 and 21, the effect of pre-training is investigated. The best performance is achieved when no pre-training and no weight freezing is used. According to the Table, it can be evaluated that the backbone plays a great role in the overall architecture, and freezing its weights significantly harms the performance. Additionally, starting the model from pre-trained weights also results in the model stopping at a premature local minimum.

8.2.3 Effect of Graph Attention Matrices

There are five different graph attention matrices: occlusion, spatial distance, directed-vertical distance, directed-horizontal distance, and size ratio. All matrix calculations are detailed in Section 4.1. The quantitative effect of each attention matrix and their combinations are analyzed in Table 8.5. The best combination for graph matrices is found when spatial distance, directed-vertical distance, and size-ratio matrices are utilized together (index 18).

Among the individual attention matrices, the most influential one is found as the spatial distance matrix. While integrating other matrices separately does not outperform using a vanilla-transformer design with no graph attention (index 1), using only a spatial distance matrix (index 3) is enough to surpass the score of a vanilla-transformer. Since the objects that are close to each other are expected to interact more, spatial distance is a good indicator for pairwise object relations.

Although directed-vertical distance and size-ratio matrices do not play significant

Table 8.5: The ablation study on the effectiveness of each attention matrix. Different combinations of these matrices are tested to demonstrate their impact, with the matrices used during model training marked by check marks. Other configurations that are common to each model is given as follows: Inception-V3 as the CNN backbone, no pre-trained weights, backbone loss weight is set to 0.5, the learning rate is 1e-4, and has 1 transformer layer. The highest score is highlighted in green, the second highest in blue, and the third highest in red.

Index	Occlusion	Spatial D.	Directed-Vertical D.	Directed-Horizontal D.	Size-Ratio	CBSC	FrISS-QD	Avg.
1						71.47	56.90	64.19
2	✓					71.86	56.00	63.93
3		✓				73.14	57.85	65.50
4			✓			71.57	56.09	63.83
5				✓		67.94	55.10	61.52
6					✓	68.53	55.19	61.86
7	✓	✓				71.27	57.66	64.47
8	✓		✓			71.18	57.33	64.26
9	✓			✓		68.82	57.75	63.29
10	✓				✓	68.82	57.75	63.29
11		✓	✓			71.57	56.90	64.24
12		✓			✓	73.14	58.08	65.61
13			✓	✓		69.51	56.14	62.83
14	✓	✓	✓			69.41	57.66	63.54
15	✓	✓		✓		70.88	57.85	64.37
16	✓	✓			✓	70.88	58.18	64.53
17		✓	✓	✓		71.96	56.42	64.19
18		✓	✓		✓	72.94	58.37	65.66
19	✓	✓	✓	✓		71.37	56.24	63.81
20	✓	✓	✓	✓	✓	70.78	58.23	64.51

roles individually (indices 4 and 6), merging these matrices with the spatial distance matrix further increases the performance (index 18). While the size-ratio matrix is designed to provide the model an insight into the relative sizes of object pairs (a "car" is usually bigger than a "person"), the directed-vertical distance matrix provides valuable information on the object's identity (e.g., "sun" is expected to be above of most objects, while "grass" is below). Combining the three attention

matrices enables the model to exploit the pairwise object proximity, pairwise relative sizes, and the relative positions of the objects.

8.2.4 Effect of Preprocessing Steps

This experiment contains the analyses regarding the preprocessing design: coloring sketch objects, the maximum number of objects in a scene, and the maximum number of categories in a scene. The results are shared in Table 8.6.

Table 8.6: The ablation study on pre-processing related design choices about the generation of the synthetic scene sketch dataset that is used during training. *Color Strokes* indicates whether coloring each object strokes based on the RGB-coloring technique or leaving as black-and-white, *Max Objs* refers to the maximum number of objects in a scene, and *Max Cats* is the maximum number of categories in a scene. The changes are applied to the control model, which has these configurations: Inception-V3 as the CNN backbone, no pre-trained weights, no backbone loss weight, the learning rate is 1e-4, has 1 Transformer layer, does not utilize any Graph Attention matrix. The highest score is highlighted in green, the second highest in blue, and the third highest in red.

Index	***Color Strokes***	***Max Objs***	***Max Cats***	**CBSC**	**FrISS-QD**	**Avg.**
1	✓	15	3	68.14	53.82	60.98
2		15	3	68.63	57.14	62.89
3		5	2	68.53	54.77	61.65
4		10	3	71.08	54.34	62.71
5		20	5	69.80	56.05	62.93

First of all, I adopted the temporal stroke order to the training of CGAT-Net by applying RGB coloring to each semantically segmented object based on the spectrum spanning blue to red, similarly as applied in the CAVT network (see Section 3.1). Unlike the segmentation task, integrating temporal stroke order did not produce a positive effect on the model performance compared to black-white sketch

object coloring (index 1 vs 2). I assume that this occurs because the RGB coloring technique helps in distinguishing object instances with similar stroke colors during segmentation. In contrast, during the recognition phase, all object strokes are considered together as a whole for assigning the correct object category.

Secondly, the impact of thresholds on the maximum number of objects and per category is examined. CBSC, which has fewer objects per sketch on average compared to FrISS-QD, shows improved performance with a threshold of 10 for the total number of objects and 3 per category (index 4). However, these settings do not perform as well on FrISS-QD, which generally contains more objects per sketch. As a result, setting the thresholds to 20 for the total number of objects and 5 per category yields the best average performance (index 5). These settings are therefore selected for the final model.

8.2.5 Random Object Selection in the Training

In this part, I executed two experiments on sketch object dataset selection during training. CGAT-Net is trained with a synthetic dataset, that is constructed by referencing MS COCO images [Lin et al. (2014)] and placing sketch objects from the same category to the position of the image object (see Section 4.5 for details). To measure the effect of the sketch datasets on the training performance, I performed two different training settings in Table 8.7. These settings are explained below:

- **CGAT-Net with QD prior**: If the target object category is available in QuickDraw [Ha and Eck (2018)], select a sketch object sample from QuickDraw. If it is not present there, choose a sample from either the FrISS or SketchNet [Dede (2023)] datasets if available.

- **CGAT-Net with random select**: If the target object category appears in multiple sketch datasets, choose a dataset randomly from QuickDraw, FrISS, and SketchNet with probabilities of 0.5, 0.3, and 0.2, respectively. Given that FrISS and SketchNet have fewer sketch samples compared to QuickDraw, the higher probability for QuickDraw results in a greater data variance.

Table 8.7: This ablation study examines the impact of random object class selection during the synthetic scene construction of the CGAT-Net training dataset. The models have the following configurations: Inception-V3 as the CNN backbone, no pre-trained weights, backbone loss weight is set to 0.5, learning rate is 1e-4, has 1 transformer layer, utilize spatial distance, directed-vertical distance, and size-ratio attention matrices.

Model	**CBSC**	**FrISS-QD**	**Avg.**
CGAT-Net with QD prior.	71.96	56.42	64.19
CGAT-Net with random select.	**74.02**	**60.31**	**67.17**

In Section 8.2.1, I explained that FrISS may include more abstract representations from the same category compared to QuickDraw. Since SketchNet is also ambiguous due to its nature (see Section 2.4), including these datasets in the training enables the model to learn a larger set of representations from the same category. Thus, the performance of the model increases in both CBSC and FrISS-QD. The final model also adopts random selection during training.

8.3 Experiments on Dual Network

Table 8.8: Comparison of the CGAT-Net against Sketchformer [Ribeiro et al. (2020)] when both are integrated into a dual network setting: CAVT + CGAT-Net and CAVT + Sketchformer.

Model	**CBSC**				**FrISS-QD**			
	OVAcc	**MeanAcc**	**MIoU**	**FWIoU**	**OVAcc**	**MeanAcc**	**MIoU**	**FWIoU**
CAVT + Sketchformer	60.00	60.62	40.82	48.53	45.39	42.71	22.58	34.75
CAVT + CGAT-Net	**65.37**	**62.14**	**43.65**	**52.66**	**53.70**	**42.78**	**24.42**	**41.72**

8.3.1 Utilization of External Classifier

As mentioned in Section 5, while CAVT and CGAT-Net can be integrated into a Dual Network for end-to-end use, they can also function independently or in combination with other classification or segmentation models. To evaluate the performance of CGAT-Net within the Dual Network, I substituted it with the state-of-the-art single sketch classifier, Sketchformer [Ribeiro et al. (2020)]. The results, shown in Table 8.8, demonstrate that in both CBSC and FrISS-QD, the combination of CAVT + CGAT-Net consistently outperforms CAVT + Sketchformer across all four metrics: OVAcc, MeanAcc, MIoU, and FWIoU.

Chapter 9

LIMITATIONS & FUTURE WORK

In this section, I provide a discussion around the challenges of this study and suggest additional directions to promote future developments for researchers.

- **Support for Alternating Object Strokes:** People often draw sketch objects in sequential order within scene sketches, making temporal stroke order important for distinguishing between object strokes. However, objects may be sketched in an alternating manner during the drawing phase of a scene sketch. Moreover, segmentation of scene sketches remains challenging due to shared strokes among different objects. Therefore, stroke-based sketch segmentation networks, such as my proposed CAVT, could benefit from developing more advanced algorithms that can differentiate between strokes of each instance in a scene, even with shared strokes or variations in drawing behavior.

- **Enlarging Freehand Scene Sketch Datasets:** Scene sketch datasets vary widely in their collection methods. However, many of these datasets lack crucial annotations, such as instance-level category information. Although FrISS provides densely annotated scenes, its size is insufficient for large-scale training. To address this, I generated synthetic scene sketches for training purposes. While synthetic sketch generation is a quick and convenient solution, it lacks the interaction between sketch objects found in free-hand sketches. Future research could focus on collecting free-hand scene sketches to enable complete training with non-synthetic data.

- **Integrating Extra Contextual Features into the Context-Aware Network:** In this study, I introduced five different graph attention matrices to encode inter-object spatial relations and enhance context-aware sketch recog-

nition. Expanding the variety of attention matrices could further improve context understanding in CGAT-Net. Additionally, incorporating absolute coordinate information for each object could provide more detailed insights into individual objects. While context understanding is rarely explored in scene sketches, I hope that CGAT-Net will inspire further research and advancements in this field.

- **Including Stroke Information into the Context-Aware Network:** In CGAT-Net, I employed an image-based CNN architecture to extract visual features from sketch objects. However, other network types, such as stroke-based models, could also be used to obtain feature embeddings. I also performed an ablation study to assess the impact of incorporating temporal information about sketch objects. Despite using an RGB-coloring technique to integrate stroke information, this approach did not improve overall performance. Consequently, a promising future direction could involve investigating alternative methods for incorporating temporal information into our context-aware network structure.

- **Utilization of Oriented Object Detection:** In CAVT, the performance of the detector plays a huge role in the overall performance. Consequently, improving the detector architecture could be further improved in future work. Numerous studies have explored Oriented Object Detection [Sandler et al. (2018); Han et al. (2021); Xie et al. (2021)], which employs oriented bounding boxes to provide more precise object boundaries. Incorporating this technique could further improve the accuracy of the stroke-to-box assignment process in CAVT.

Chapter 10

CONCLUSION

In this thesis, I focused on scene sketch understanding and proposed two novel networks to address well-known research problems: scene sketch segmentation and scene sketch recognition. Additionally, I collected a free-hand scene sketch dataset with detailed annotations. Finally, I conducted extensive experiments and ablation studies to highlight the superior performance of my proposed networks.

First, I introduced a novel Class-Agnostic Visio-Temporal (CAVT) Network that leverages visual and temporal information in sketches for scene sketch segmentation. CAVT is the first segmentation network designed to identify individual object instances within scene sketches at both the stroke and instance levels. In CAVT, objects are segmented without any class assignment, enabling the differentiation of object instances in a given scene without being limited by a predefined category list. Unlike previous approaches, CAVT is capable of distinguishing between multiple object instances even when they belong to the same category.

Secondly, I proposed a novel Context-Aware Graph Attention Transformer Network (CGAT-Net) for scene sketch recognition. This novel network processes the objects within a scene sketch and generates class predictions for each object instance. As a pioneering approach, CGAT-Net leverages inter-object relationships within the scene to improve classification performance on individual objects using a Transformer-based Attention Network. This work highlights the importance of context in accurately identifying individual objects within a scene in the sketch domain. CGAT-Net sets the stage for future studies that will focus on context-aware sketch recognition.

Moreover, I collected the largest Free-hand Instance- and Stroke-level Scene sketch dataset (FrISS), including 1K free-hand scene sketches covering 403 object categories. This dataset is richly annotated, featuring instance and stroke-level cat-

egory labels, sketch-text pairs, and verbal audio descriptions associated with each scene. I hope that FrISS will serve as a valuable resource for a variety of studies, including stroke-level scene sketch segmentation, speech-based sketch applications, and cross-modal research utilizing sketch-text pairs.

Finally, extensive experiments demonstrate that both CAVT and CGAT-Net achieve state-of-the-art performance in their respective domains on the FrISS and other freehand scene sketch datasets. Due to their designs, CAVT and CGAT-Net can be utilized independently in their specific areas or sequentially in combination. By integrating both networks into a unified end-to-end framework, I assessed the performance of the Dual Network, which surpassed that of existing scene sketch segmentation networks on the same datasets. In summary, both the individual networks and the integrated framework establish new benchmarks and support a range of advanced applications in the field.

BIBLIOGRAPHY

Badrinarayanan, V., Kendall, A., and Cipolla, R. (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495.

Bell, S., Zitnick, C. L., Bala, K., and Girshick, R. (2016). Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2874–2883.

Bourouis, A., Fan, J. E., and Gryaditskaya, Y. (2023). Open vocabulary semantic scene sketch understanding. *arXiv preprint arXiv:2312.12463*.

Caesar, H., Uijlings, J., and Ferrari, V. (2018). Coco-stuff: Thing and stuff classes in context. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1209–1218.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2017). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848.

Chen, X. and Gupta, A. (2017). Spatial memory for context reasoning in object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 4086–4096.

Choi, M. J., Lim, J. J., Torralba, A., and Willsky, A. S. (2010). Exploiting hierarchical context on a large database of object categories. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 129–136. IEEE.

Chowdhury, P. N., Sain, A., Bhunia, A. K., Xiang, T., Gryaditskaya, Y., and Song, Y.-Z. (2022). Fs-coco: Towards understanding of freehand sketches of common objects in context. In *European Conference on Computer Vision*, pages 253–270. Springer.

Dede, E. (2023). Sketch-based creativity assistant. Master's thesis, Koc University.

Deng, J., Li, W., Chen, Y., and Duan, L. (2021). Unbiased mean teacher for cross-domain object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4091–4101.

Divvala, S. K., Hoiem, D., Hays, J. H., Efros, A. A., and Hebert, M. (2009). An empirical study of context in object detection. In *2009 IEEE Conference on computer vision and Pattern Recognition*, pages 1271–1278. IEEE.

Eitz, M., Hays, J., and Alexa, M. (2012). How do humans sketch objects? *ACM Transactions on graphics (TOG)*, 31(4):1–10.

Galleguillos, C., Rabinovich, A., and Belongie, S. (2008). Object categorization using co-occurrence, location and appearance. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

Gao, C., Liu, Q., Xu, Q., Wang, L., Liu, J., and Zou, C. (2020). Sketchycoco: Image generation from freehand scene sketches. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5174–5183.

Ge, C., Sun, H., Song, Y.-Z., Ma, Z., and Liao, J. (2022). Exploring local detail perception for scene sketch semantic segmentation. *IEEE Transactions on Image Processing*, 31:1447–1461.

Ge, Z., Liu, S., Wang, F., Li, Z., and Sun, J. (2021). Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*.

Ha, D. and Eck, D. (2017). A neural representation of sketch drawings. *arXiv preprint arXiv:1704.03477*.

Ha, D. and Eck, D. (2018). A neural representation of sketch drawings. In *International Conference on Learning Representations*.

Han, J., Ding, J., Li, J., and Xia, G.-S. (2021). Align deep features for oriented object detection. *IEEE transactions on geoscience and remote sensing*, 60:1–11.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Howard, A., Sandler, M., Chu, G., Chen, L.-C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., et al. (2019). Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324.

Hu, H., Gu, J., Zhang, Z., Dai, J., and Wei, Y. (2018). Relation networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3588–3597.

Jiang, J., Chen, B., Wang, J., and Long, M. (2021). Decoupled adaptation for cross-domain object detection. *arXiv preprint arXiv:2110.02578*.

Kaiyrbekov, K. and Sezgin, M. (2019). Deep stroke-based sketched symbol reconstruction and segmentation. *IEEE computer graphics and applications*, 40(1):112–126.

Li, C., Li, L., Geng, Y., Jiang, H., Cheng, M., Zhang, B., Ke, Z., Xu, X., and Chu, X. (2023). Yolov6 v3. 0: A full-scale reloading. *arXiv preprint arXiv:2301.05586*.

Li, L., Fu, H., and Tai, C.-L. (2018). Fast sketch segmentation and labeling with deep learning. *IEEE computer graphics and applications*, 39(2):38–51.

Li, L., Zou, C., Zheng, Y., Su, Q., Fu, H., and Tai, C.-L. (2020). Sketch-r2cnn: an rnn-rasterization-cnn architecture for vector sketch recognition. *IEEE transactions on visualization and computer graphics*, 27(9):3745–3754.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer.

Mottaghi, R., Chen, X., Liu, X., Cho, N.-G., Lee, S.-W., Fidler, S., Urtasun, R., and Yuille, A. (2014). The role of context for object detection and semantic segmentation in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 891–898.

Noh, H., Hong, S., and Han, B. (2015). Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1520–1528.

Parikh, D., Zitnick, C. L., and Chen, T. (2011). Exploring tiny images: The roles of appearance and contextual information for machine and human object recognition. *IEEE transactions on pattern analysis and machine intelligence*, 34(10):1978–1991.

Rabinovich, A., Vedaldi, A., Galleguillos, C., Wiewiora, E., and Belongie, S. (2007). Objects in context. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE.

Ribeiro, L. S. F., Bui, T., Collomosse, J., and Ponti, M. (2020). Sketchformer: Transformer-based representation for sketched structure. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14153–14162.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520.

Sangkloy, P., Burnell, N., Ham, C., and Hays, J. (2016). The sketchy database: learning to retrieve badly drawn bunnies. *ACM Transactions on Graphics (TOG)*, 35(4):1–12.

Simonyan, K. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sun, Z., Wang, C., Zhang, L., and Zhang, L. (2012). Free hand-drawn sketch segmentation. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part I 12*, pages 626–639. Springer.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Topal, B. B., Yuret, D., and Sezgin, T. M. (2023). Domain-adaptive self-supervised pre-training for face & body detection in drawings.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, X. and Zhu, Z. (2023). Context understanding in computer vision: A survey. *Computer Vision and Image Understanding*, 229:103646.

Wu, X., Qi, Y., Liu, J., and Yang, J. (2018). Sketchsegnet: A rnn model for labeling sketch strokes. In *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE.

Xie, X., Cheng, G., Wang, J., Yao, X., and Han, J. (2021). Oriented r-cnn for object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3520–3529.

Xu, M., Zhang, Z., Hu, H., Wang, J., Wang, L., Wei, F., Bai, X., and Liu, Z. (2021a). End-to-end semi-supervised object detection with soft teacher. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3060–3069.

Xu, P., Joshi, C. K., and Bresson, X. (2021b). Multigraph transformer for free-hand sketch recognition. *IEEE Transactions on Neural Networks and Learning Systems*, 33(10):5150–5161.

Yang, J., Ke, A., Yu, Y., and Cai, B. (2023). Scene sketch semantic segmentation with hierarchical transformer. *Knowledge-Based Systems*, 280:110962.

Yang, L., Sain, A., Li, L., Qi, Y., Zhang, H., and Song, Y.-Z. (2020). S 3 net: Graph representational network for sketch recognition. In *2020 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE.

Yang, L., Zhuang, J., Fu, H., Wei, X., Zhou, K., and Zheng, Y. (2021). Sketchgnn: Semantic sketch segmentation with graph neural networks. *ACM Transactions on Graphics (TOG)*, 40(3):1–13.

Yu, Q., Liu, F., Song, Y.-Z., Xiang, T., Hospedales, T. M., and Loy, C.-C. (2016). Sketch me that shoe. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 799–807.

Yu, Q., Yang, Y., Song, Y.-Z., Xiang, T., and Hospedales, T. (2015). Sketch-a-net that beats humans. *arXiv preprint arXiv:1501.07873*.

Zhang, H., Li, F., Liu, S., Zhang, L., Su, H., Zhu, J., Ni, L. M., and Shum, H.-Y. (2022). Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*.

Zhang, J., Chen, Y., Li, L., Fu, H., and Tai, C.-L. (2018). Context-based sketch classification. In *Proceedings of the Joint Symposium on Computational Aesthetics and Sketch-Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering*, pages 1–10.

Zhang, Z., Deng, X., Li, J., Lai, Y., Ma, C., Liu, Y., and Wang, H. (2023). Stroke-based semantic segmentation for scene-level free-hand sketches. *The Visual Computer*, 39(12):6309–6321.

Zheng, Y., Xie, J., Sain, A., Song, Y.-Z., and Ma, Z. (2023). Sketch-segformer: Transformer-based segmentation for figurative and creative sketches. *IEEE transactions on image processing*.

Zhu, X., Xiao, Y., and Zheng, Y. (2018). Part-level sketch segmentation and labeling using dual-cnn. In *Neural Information Processing: 25th International Conference, ICONIP 2018, Siem Reap, Cambodia, December 13-16, 2018, Proceedings, Part I 25*, pages 374–384. Springer.

Zou, C., Yu, Q., Du, R., Mo, H., Song, Y.-Z., Xiang, T., Gao, C., Chen, B., and Zhang, H. (2018). Sketchyscene: Richly-annotated scene sketches. In *ECCV*, pages 438–454. Springer International Publishing.

Appendix A

CGAT-NET: CATEGORY LIST AND ADDITIONAL VISUAL RESULTS

A.1 Categories Supported by CGAT-Net

I provide the list of categories that are supported by CGAT-Net:

- *airplane, alarm clock, ambulance, apple, asparagus, backpack, banana, baseball, baseball bat, baseball field, basket, basketball, bear, bed, bench, bicycle, billboard, bird, birthday cake, blackberry, blanket, blueberry, book, boomerang, bowl, bread, bridge, broccoli, bus, bush, cage, cake, canoe, car, carpet, carrot, castle, cat, ceiling fan, cell phone, chair, chandelier, cheese, chicken, church, clock, cloud, coffee cup, computer, computer case, cookie, cooler, couch, counter, cow, cruise ship, cup, curtain, dishwasher, dog, donut, door, dresser, elephant, eraser, eyeglasses, fence, field, fire hydrant, firetruck, fishing net, flashlight, flip flops, floor lamp, flower, fork, frisbee, giraffe, glove, grapes, grass, hair dryer, hamburger, hat, helicopter, helmet, horse, hospital, hot dog, house, house plant, jail, keyboard, kite, knife, lantern, laptop, leaf, light bulb, microwave, mirror, moon, motorbike, mountain, mouse, mug, mushroom, necktie, net, ocean, onion, orange, oven, palm tree, panda, pants, paper, parrot, peanut, pear, peas, pencil, pickup truck, pillow, pineapple, pizza, plate, police car, potato, purse, rail, rain, remote control, river, road, sailboat, sand, sandwich, school bus, scissors, seagull, sheep, shelf, shoe, sidewalk, sink, skateboard, skyscraper, snowflake, soccer ball, soccer field, speedboat, spoon, stage, stage lights, stairs, star, steak, stone, stop sign, stove, strawberry, streetlight, string bean, suitcase, sun, sunglasses, surfboard, t-shirt, table, teddy-bear, television, tennis racquet, tent, toaster, toilet, toothbrush, tower, traffic light, train, tree, truck,*

umbrella, van, vase, washing machine, watermelon, window, wine bottle, wine glass, wood, wristwatch, person, zebra.

A.2 Category Mappings between MS COCO and Sketch Datasets

The complete category mappings from MS COCO [Lin et al. (2014)] to QuickDraw [Ha and Eck (2018)], FrISS, and SketchNet [Dede (2023)] are detailed below. The keys represent categories from the source dataset (MS COCO), while the values are combinations of categories from the three target datasets (QuickDraw, FrISS, SketchNet). If a value is marked as *null*, it indicates that no corresponding category exists in the target datasets. Please note that these mappings are not direct synonyms; rather, each target category is chosen for its potential to appear in a similar location, size, and context within the scene.

- {*airplane: [airplane, helicopter], clock: [alarm clock, clock, wristwatch], truck: [ambulance, firetruck, pickup truck, truck, van], apple: [apple], vegetable: [asparagus, onion, peas, potato, string bean, mushroom], backpack: [backpack], banana: [banana], house: [house], sports ball: [baseball, basketball, soccer ball], baseball bat: [baseball bat], bear: [bear, panda], bed: [bed], bench: [bench], bicycle: [bicycle], bird: [bird, seagull, parrot], cake: [birthday cake, cake], fruit: [blackberry, blueberry, grapes, pear, pineapple, strawberry, watermelon], book: [book], tie: [necktie], food-other: [bread, peanut, steak, cheese, chicken], bridge: [bridge], broccoli: [broccoli], bus: [bus, school bus], bush: [bush], boat: [canoe, cruise ship, sailboat, speedboat], car: [car, police car], carrot: [carrot], cat: [cat], cell phone: [cell phone], chair: [chair], building-other: [church, hospital, castle], clouds: [cloud], cup: [coffee cup, cup, mug], laptop: [computer, laptop], refrigerator: [cooler], couch: [couch], cow: [cow], dog: [dog], donut: [donut, cookie], door-stuff: null, door: [door], cabinet: [dresser], elephant: [elephant], fence: [fence], fire hydrant: [fire hydrant], light: [floor lamp, lantern, light bulb, stage lights, flashlight], flower: [flower], fork: [fork], giraffe: [giraffe], sandwich: [hamburger, sandwich], horse: [horse], hot dog: [hot dog], potted plant: [house plant], cage: [jail, cage], keyboard: [keyboard], knife: [knife],*

paper: [paper], microwave: [microwave], motorcycle: [motorbike], mountain: [mountain], mouse: [mouse], sea: [ocean], oven: [oven, stove, dishwasher, washing machine], wall-panel: null, pillow: [pillow], pizza: [pizza], handbag: [purse], waterdrops: [rain], remote: [remote control], scissors: [scissors], sheep: [sheep], sink: [sink], skateboard: [skateboard], skyscraper: [skyscraper, tower], spoon: [spoon], stairs: [stairs], stop sign: [stop sign], suitcase: [suitcase, backpack], table: [table], teddy bear: [teddy-bear], tv: [television], tennis racket: [tennis racquet], tent: [tent], toaster: [toaster], toilet: [toilet], toothbrush: [toothbrush], traffic light: [traffic light], train: [train], branch: null, umbrella: [umbrella], vase: [vase], parking meter: null, frisbee: [frisbee, boomerang], skis: null, snowboard: null, kite: [kite], baseball glove: [glove], surfboard: [surfboard], bowl: [bowl, basket], orange: [orange], dining table: [table], hair drier: [hair dryer], bottle: [wine bottle], wine glass: [wine glass], person: [yoga], zebra: [zebra], street sign: [stop sign, streetlight], hat: [hat, helmet], shoe: [shoe, flip flops], eye glasses: [eyeglasses, sunglasses], plate: [plate], mirror: [mirror], window: [window], desk: [table], blender: null, hair brush: null, banner: [billboard], blanket: [blanket], cardboard: [paper], carpet: [carpet], ceiling-other: [chandelier, ceiling fan], ceiling-tile: null, cloth: [t-shirt, pants], clothes: null, counter: [counter], cupboard: [dresser], curtain: [curtain], desk-stuff: [pencil, eraser, computer case], dirt: null, floor-marble: null, floor-other: null, floor-stone: null, floor-tile: null, floor-wood: null, fog: null, furniture-other: null, grass: [grass], gravel: null, ground-other: null, hill: [mountain], leaves: [leaf], mat: [carpet], metal: null, mirror-stuff: null, moss: null, mud: null, napkin: null, net: [net, fishing net], pavement: [sidewalk], road: [road], plant-other: null, plastic: null, platform: [stage], playingfield: [soccer field, baseball field, field], railing: [fence], railroad: [rail], river: [river], rock: [stone], roof: null, rug: [carpet], salad: null, sand: [sand], shelf: [shelf], sky-other: [sun, moon, star], snow: [snowflake], solid-other: null, stone: [stone], straw: null, structural-other: null, textile-other: null, towel: null, tree: [tree, palm tree], wall-brick: null, wall-concrete: null, wall-other: null, wall-stone: null, wall-tile: null, wall-wood: null, water-other: null,

window-blind: null, window-other: null, wood: [wood] }

A.3 Additional Visual Results on CGAT-Net with SOTA Single Sketch Classifier

Here, I present additional visual comparisons between CGAT-Net and Sketchformer [Ribeiro et al. (2020)], the best-performing single sketch classifier (detailed comparisons are made in Section 7.3.3). Figures A.1 and A.2 showcase class prediction results for the CBSC [Zhang et al. (2018)] and FrISS scene sketch datasets. These examples demonstrate that CGAT-Net successfully benefits from the proposed graph attention matrices (i.e., spatial distance, directed-vertical distance, size ratio) and effectively utilizes contextual information within a scene for object classification. Details of these graph attention matrices can be found in Section 4.1. Even when CGAT-Net misclassifies an object, the incorrect label remains contextually appropriate based on the object's relative size and position. In contrast, Sketchformer makes predictions independently of the scene context, leading to less accurate results. Overall, these results support the notion that context-based sketch recognition for scene sketches is a promising area for future research.

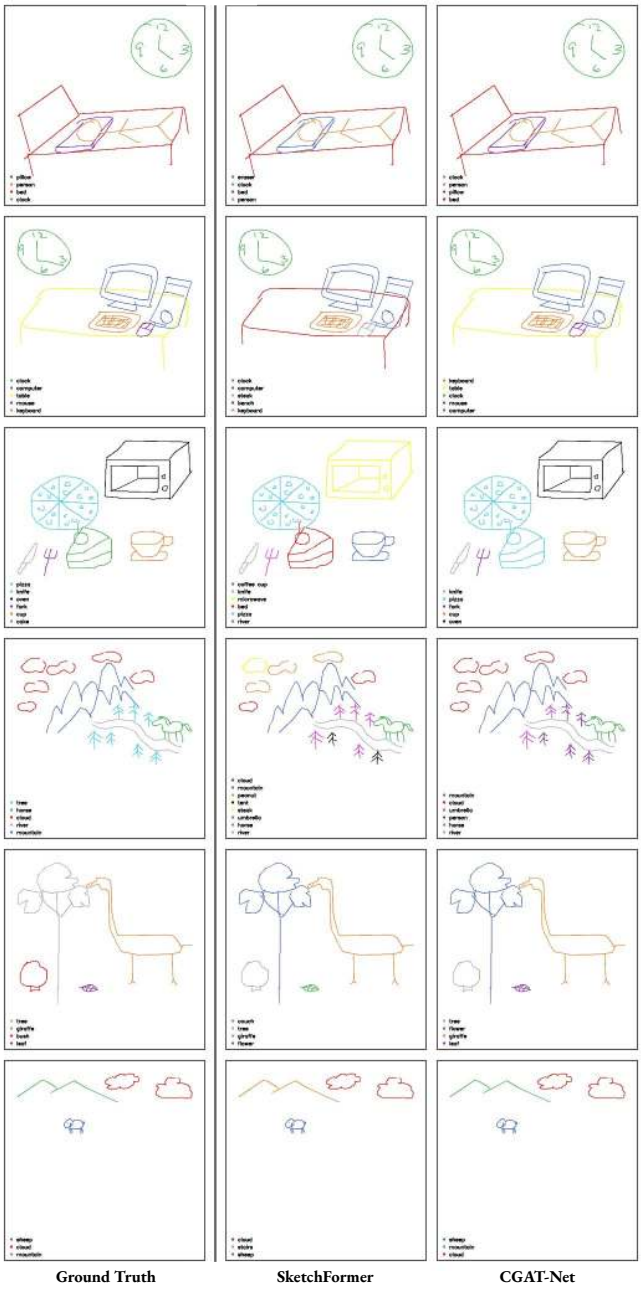


Figure A.1: Visual comparison of CGAT-Net with Sketchformer [Ribeiro et al. (2020)], tested on CBSC dataset.

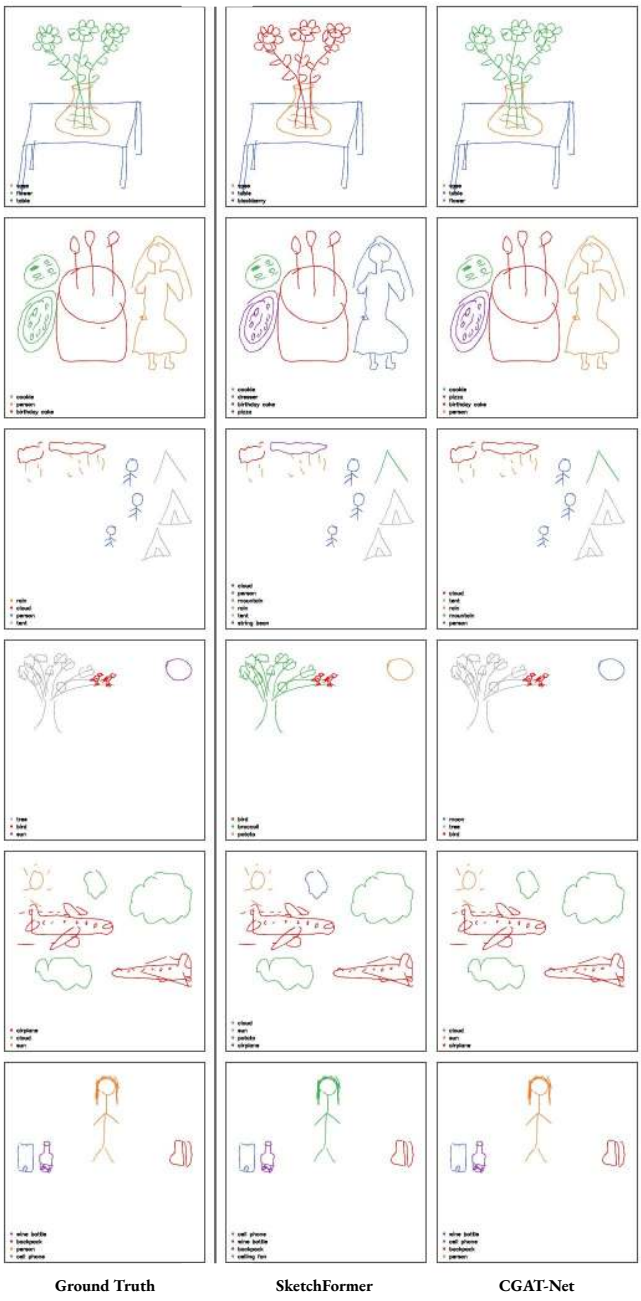


Figure A.2: Visual comparison of CGAT-Net with Sketchformer [Ribeiro et al. (2020)], tested on FrISS dataset.

Appendix B

ADDITIONAL VISUAL RESULTS ON DUAL NETWORK WITH SOTA SCENE SKETCH SEMANTIC SEGMENTATION NETWORKS

I present additional visual results for the scene sketch segmentation task. Figures B.1 and B.2 display sample results obtained from the CBSC [Zhang et al. (2018)] and FrISS scene sketches, respectively. To illustrate class-level segmentation, each pixel or stroke within the scene is colored according to its predicted object category. As detailed in Section 7.4.2, I developed a variant of my pipeline, denoted as CAVT + CGAT-Net*. This variant differs only in the sketch classification step to ensure a fair comparison with OV. The additional visual outcomes shown in Figures B.1 and B.2 demonstrate consistent segmentation results from my dual network (CAVT + CGAT-Net) and its variant (CAVT + CGAT-Net*). This indicates that leveraging stroke representations and the temporal order of stroke sequences is a promising approach for addressing the scene sketch segmentation problem.

Figure B.1: Visual comparison of Dual Network with LDP [Ge et al. (2022)] and OV [Bourouis et al. (2023)] models, tested on CBSC dataset. CAVT+CGAT-Net*: refers to predictions retrieved from the Dual Network only from the object classes present in the given scene, to ensure a fair comparison with OV.

Figure B.2: Visual comparison of Dual Network with LDP [Ge et al. (2022)] and OV [Bourouis et al. (2023)] models, tested on FrISS dataset. CAVT+CGAT-Net*: refers to predictions retrieved from the Dual Network only from the object classes present in the given scene, to ensure a fair comparison with OV.

Appendix C

ADDITIONAL DETAILS ON FRISS DATASET

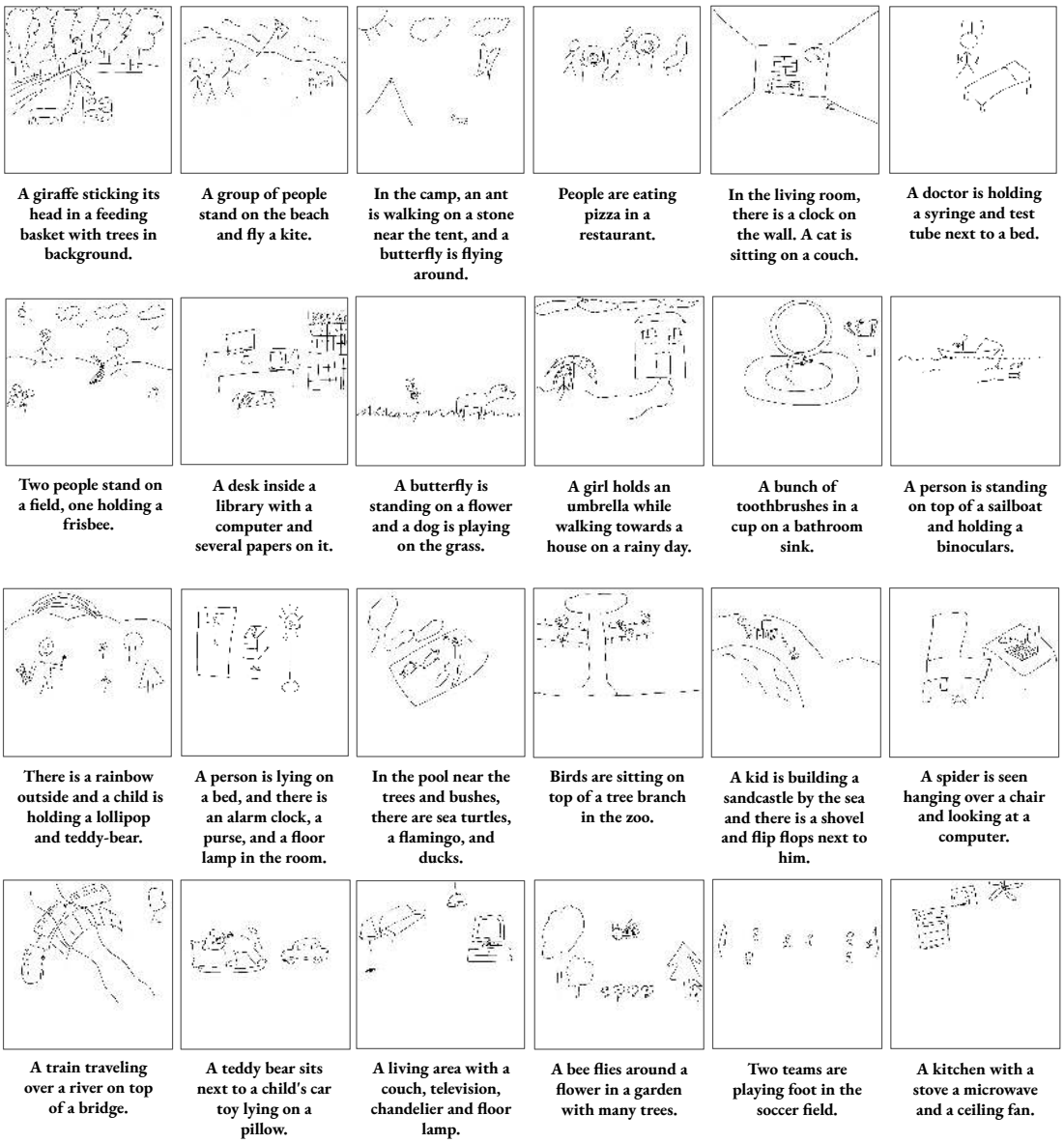


Figure C.1: Sample scene sketches from the FrISS dataset paired with their textual scene descriptions

C.1 Analysis on FrISS Dataset

Here, I provide additional analysis on the collected dataset in Figures C.2 and C.3. In Figure C.2, I observe that the count of distinct object categories within a scene varies between 1 and 10, with a dominant accumulation between 3 and 6. Figure C.3 reveals that the most frequently occurring object categories in FrISS are person, tree, table, flower, and cloud, with the remaining categories distributed more balanced throughout the dataset. Moreover, I include extra sample scene sketches from FrISS along with their corresponding textual scene descriptions, in Figure C.1.

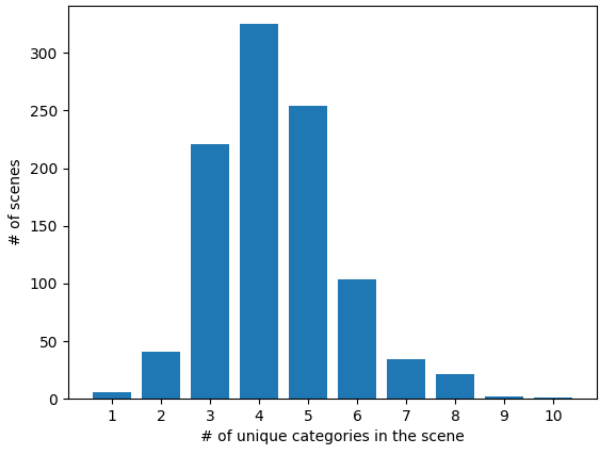


Figure C.2: The visualization of the number of unique object categories per scene in the FrISS dataset

List of Categories in FrISS: airplane, alarm clock, ambulance, ant, apple, arm, asparagus, axe, backpack, banana, bandage, barn, baseball, baseball bat, basket, basketball, bathtub, beach, bear, bed, bee, belt, bench, bicycle, binoculars, bird, birthday cake, blackberry, book, boomerang, bowtie, bracelet, bread, bridge, broom, bucket, bus, bush, butterfly, cake, calendar, camera, campfire, candle, cannon, canoe, car, carrot, castle, cat, ceiling fan, cell phone, cello, chair, chandelier, clarinet, clock, cloud, coffee cup, compass, computer, cookie, cooler, couch, cow, crab, crayon, crown, cruise ship, cup, dishwasher, dog, dolphin, donut, door, dresser, drill, drums, duck, dumbbell, elephant, eraser, eyeglasses, face, fan, fence, fire hydrant, fireplace,

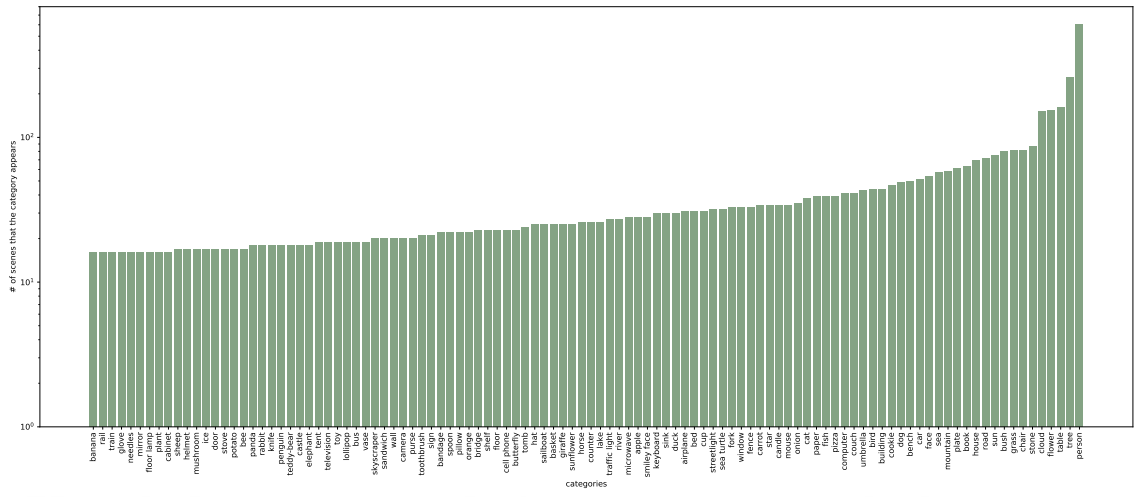


Figure C.3: The visualization of the number of scene sketches that each object category appears in. For visualization purposes, I selected the categories that have more than 15 appearances in the FrISS dataset.

fish, flamingo, flashlight, flip flops, floor lamp, flower, fork, garden, giraffe, grapes, grass, guitar, hammer, hand, harp, hat, headphones, helmet, horse, hot air balloon, hot dog, hourglass, house, ice cream, key, keyboard, knife, ladder, laptop, leaf, light bulb, lighter, lighthouse, lightning, lion, lollipop, mailbox, map, microphone, microwave, moon, motorbike, mountain, mouse, mug, mushroom, necklace, ocean, octopus, onion, oven, palm tree, panda, pants, paper clip, pear, peas, pencil, penguin, picture frame, pig, pillow, pizza, police car, pond, pool, popsicle, potato, purse, rabbit, radio, rain, rainbow, rake, remote control, rhinoceros, river, sailboat, sandwich, saw, saxophone, school bus, scissors, screwdriver, sea turtle, see saw, shark, sheep, shoe, shovel, sink, skateboard, skull, skyscraper, sleeping bag, smiley face, snake, snorkel, snowflake, snowman, soccer ball, sock, spider, spoon, squirrel, stairs, star, steak, stereo, stop sign, stove, strawberry, streetlight, string bean, submarine, suitcase, sun, swan, swing set, syringe, t-shirt, table, teddy-bear, telephone, television, tennis racquet, tent, toilet, toothbrush, toothpaste, tractor, traffic light, train, tree, truck, trumpet, umbrella, vase, washing machine, watermelon, waterslide, wheel, windmill, wine bottle, wine glass, wristwatch, yoga*, zebra, anchor, bag, ball, balloon, barrier, baseball field, basketball hoop, bee nest, bell, billboard,

board, bone, bottle, bowl, box, branch, building, button, cabinet, cable, cage, candy, carpet, cave, ceiling, cheese, chicken, cockroach, coconut, computer case, container, coral, counter, crosswalk, cupboard, curly hair, curtain, dagger, desk, dirt, dog collar, drain, drawer, earth, egg, exhibition, field, fish tank, fishing net, fishing rod, flag, floor, football field, footprint, fridge, frisbee, gas pump, gas station, glass, glass shard, glove, goal, gun, hair, hair dryer, hair tie, hammock, handcuffs, hanger, heart, hook, ice, jellyfish, kite, lake, lamp, light effect, marshmallow, meat, mirror, monitor, moon crater, mousepad, mud, museum, music note, necktie, needles, net, notebook, notes, orange, paddle, paper, path, pathway, peach, pepper, phone box, picnic rug, pipe, plant, plate, plug, present, printer, propeller, rail, restaurant, ribbon, road, rocket, roof, room, rope, ruler, safe, salt, sand, sandcastle, sausage, scarecrow, scarf, sea, sea fish, sea goggles, sea horse, sea shell, seagull, serum, shelf, shower head, sidewalk, sign, slide, smoke, soccer field, speaker, spider web, stage, stage lights, stand, staple, station, stick, stone, stool, strainer, street, suit, sunflower, sunglasses, surfboard, swim goggles, tape player, tennis court, test tube, toilet paper, tomb, tower, toy, traffic cone, trash bin, tray, tribune, turnstile, wall, walnut, water, weapon, wind, window, wing, wood.

Please note that in the FrISS dataset, *yoga** denotes the *person* class. This mapping between the two classes is due to their visual similarity.

C.2 Common Categories of FrISS and Other Scene Sketch Datasets

- **List of common categories between FrISS and SKY-Scene [Ge et al. (2022)]:** airplane, apple, banana, bench, bicycle, dresser, car, cat, chair, chicken, couch, cow, cup, dog, flower, horse, pickup truck, sheep, bird, strawberry, table, tree, umbrella, mountain, wine bottle.

- **List of common categories between FrISS and SketchyScene [Zou et al. (2018)]:** airplane, apple, banana, basket, bench, bicycle, bird, wine bottle, bus, car, cat, chair, chicken, cloud, cow, cup, dog, fence, flower, grapes, grass, horse, house, moon, mountain, person, carpet, road, sheep, couch, star, streetlight, sun, table, tree, truck, umbrella.

- **List of common categories between FrISS and QuickDraw [Ha and Eck (2018)]:** airplane, helicopter, alarm clock, clock, wristwatch, ambulance, firetruck, pickup truck, truck, leaf, van, apple, asparagus, onion, peas, potato, string bean, mushroom, backpack, banana, house, baseball, basketball, soccer ball, baseball bat, bear, panda, bed, bench, bicycle, bird, parrot, birthday cake, cake, blackberry, blueberry, grapes, pear, pineapple, strawberry, watermelon, book, bread, peanut, steak, bridge, broccoli, bus, school bus, bush, canoe, cruise ship, sailboat, speedboat, car, police car, carrot, cat, cell phone, chair, church, hospital, castle, cloud, coffee cup, cup, mug, computer, laptop, cooler, couch, cow, dog, donut, cookie, door, dresser, elephant, fence, fire hydrant, floor lamp, lantern, light bulb, flashlight, flower, fork, giraffe, hamburger, sandwich, horse, hot dog, house plant, jail, keyboard, knife, microwave, motorbike, mountain, mouse, ocean, oven, stove, dishwasher, washing machine, pillow, pizza, purse, rain, remote control, scissors, sheep, sink, skateboard, skyscraper, spoon, stairs, stop sign, suitcase, backpack, table, teddy-bear, television, tennis racquet, tent, toaster, toilet, toothbrush, traffic light, train, umbrella, vase, boomerang, basket, table, wine bottle, wine glass, person, zebra, stop sign, streetlight, hat, helmet, shoe, flip flops, eyeglasses, table, chandelier, ceiling fan, t-shirt, pants, dresser, pencil, eraser, grass, mountain, fence, river, sun, moon, star, snowflake, tree, palm tree

- **List of common categories between FrISS and CBSC [Zhang et al. (2018)]:** candle, bus, backpack, keyboard, car, camera, clock, mug, television, truck, banana, couch, elephant, flower, oven, pillow, cow, helmet, sheep, bridge, bench, table, spoon, horse, sandwich, bread, ladder, skateboard, tree, suitcase, bed, giraffe, house, fence, train, laptop, hat, bird, zebra, eyeglasses, fork, carrot, toilet, cat, person, airplane, baseball, bicycle, computer, basket, tent, stairs, chair, cell phone, river, cloud, knife, vase, umbrella, leaf, mountain, pizza, bucket, bear, cup, dog, bush, apple, key, cake, book, mouse, ocean.

- **List of common categories between FrISS and FS-COCO [Chowd-**

hury et al. (2022)]: cloud, orange, cow, net, hot dog, car, couch, laptop, frisbee, road, chair, wine glass, roof, bed, horse, fork, knife, pizza, bird, river, sandwich, fire hydrant, floor, banana, apple, counter, backpack, bear, plate, mud, toothbrush, shoe, cup, airplane, umbrella, mountain, book, scissors, window, donut, bush, spoon, stairs, keyboard, vase, grass, wood, fence, bottle, kite, plant, mirror, traffic light, cat, door, oven, dog, truck, bus, zebra, toilet, bridge, skateboard, bench, table, dirt, bicycle, cage, giraffe, tent, tree, cake, picnic rug, bowl, stop sign, branch, house, sand, elephant, clock, cell phone, paper, skyscraper, baseball bat, carrot, suitcase, field, train, stone, sheep, surfboard, flower, hat, sea, person, tennis racquet.

C.3 Ethical Considerations in Data Collection

The FrISS dataset contains free-hand scene sketches paired with their textual descriptions, audio clips of participants, and video recordings of drawing processes. During the drawing process, participants were asked to verbally explain their sketches in their native languages. At the beginning of the data collection, participants received detailed information regarding the following: the recording of their drawing screen in video format, the retention of their verbal descriptions as audio clips, and the potential release of their data in a research paper. Each participant was kindly requested to review and sign the consent form acknowledging my data collection procedures:

'I confirm that I have thoroughly read and understood the instructions. I hereby authorize the utilization of my anonymized data (i.e., drawings, video, and audio recordings) for scientific research purposes.'

Participants who consented to my data collection terms were assigned a random ID and proceeded with the data collection process. Additionally, I provided a contact address to allow participants to confidentially address any concerns regarding the release of their data.