Sketch Quality Prediction using Transformers

Sarah B. Maxseiner

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Engineering

A. Lynn Abbott, Chair
Creed F. Jones
Yue Wang

December 12, 2022
Blacksburg, Virginia

Keywords: computer vision, sketches, transformers, machine learning

Sketch Quality Prediction using Transformers

Sarah B. Maxseiner

(ABSTRACT)

The quality of an input sketch can affect performance of the computational algorithms. However, the quality of a sketch is not often considered when working with sketch tasks and automated sketch quality prediction has not been previously studied. This thesis presents quality prediction on the "Sketchy" dataset. The method presented here predicts a quality label rather than a zero to one quality metric. This thesis predicts an understandable label rather than a computer-generated quality metric with no human input. Previous tasks like sketch classification have used a transformer architecture to leverage the vector format of sketches. The architecture used in sketch classification was called Sketchformer. The Sketchformer was adopted and trained to predict quality labels of hand-drawn sketches. This Sketchformer architecture achieves 66% accuracy when predicting the 5-labels. The same transformer achieves up to 97% accuracy in a different experiment when combining the different labels into good versus bad (2-label) experiments. The sketchformer significantly outperforms the SVM baseline. The results of the experiments show that the transformer embedding space facilitates separation of 'good' sketch quality from 'bad' sketch quality with high accuracy.

Sketch Quality Prediction using Transformers

Sarah B. Maxseiner

(GENERAL AUDIENCE ABSTRACT)

If pictures are worth 1000 words, then sketches are worth a few hundred words. Sketches are easy to create using a pen and tablet. Objects in the sketches can be drawn many ways, depending on the talent of the creator and pose of the object. The quality of the sketches vary pretty drastically. When using sketches in computer vision tasks, the quality of a sketch can affect the performance of the computational algorithm. However, the quality of a sketch is not often considered when working with other sketch tasks. One common sketch task is called Sketch-Based Image Retrieval (SBIR). The input of this task is the sketch of an object/subject, and the model returns a matching image of the same object/subject. If the quality of the input sketch is bad, the output of this model will be poor. This thesis predicts the quality of sketches. The dataset used is called the "Sketchy" dataset, this dataset was originally used to study SBIR. However, the creators of the dataset provided quality labels for the sketches. This allows for quality prediction on this dataset, which has not previously been completed. There are 5 different labels assigned to sketches. One of the experiments completed for this thesis was predicting 1 of the 5 labels for each sketch. The other experiments for this thesis create good and bad labels by combining the 5 labels. The Sketchformer architecture created by Ribeiro et al. is used to run the experiments. The Sketchformer achieves 66% on the 5-label experiment and up to 97% on the good and bad (2-label) experiment. This transformer outperforms a Support Vector Machine baseline on this quality labels. The results of the experiments show that the transformer applied to this dataset is a valuable contribution by surpassing the baseline on multiple tasks. Additionally, accuracy values from these experiments are similar to values found in the corresponding image quality prediction task.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Motivation Images are used as a form of communication and expression. Often, an image can be a more efficient form of communication than speech. Although photographic images are useful when available, sometimes an image representing the idea you want to convey can be difficult to find or does not exist at all. This is where hand-drawn sketches come in. If a sketch is drawn well, it can carry a similar amount of information as a picture.

Advancements in the computing industry have given rise to sketches that are created and stored digitally by using devices such as smart pens. The quality of these sketches can vary dramatically, depending on the creator's artistic abilities and the time devoted to creating the drawing. For example, Figure 1.1 shows differences in sketch quality.

Previous work done in the computer vision field has taken digital sketches and attempted to classify them, but most of these models have ignored any consideration of sketch quality. For example, sketch classification assigns a label, like 'cat', to describe the subject of the sketch. If the input sketch is of bad quality, the results are likely to be poor. The ability to identify the quality label of a sketch can provide a model with the information necessary

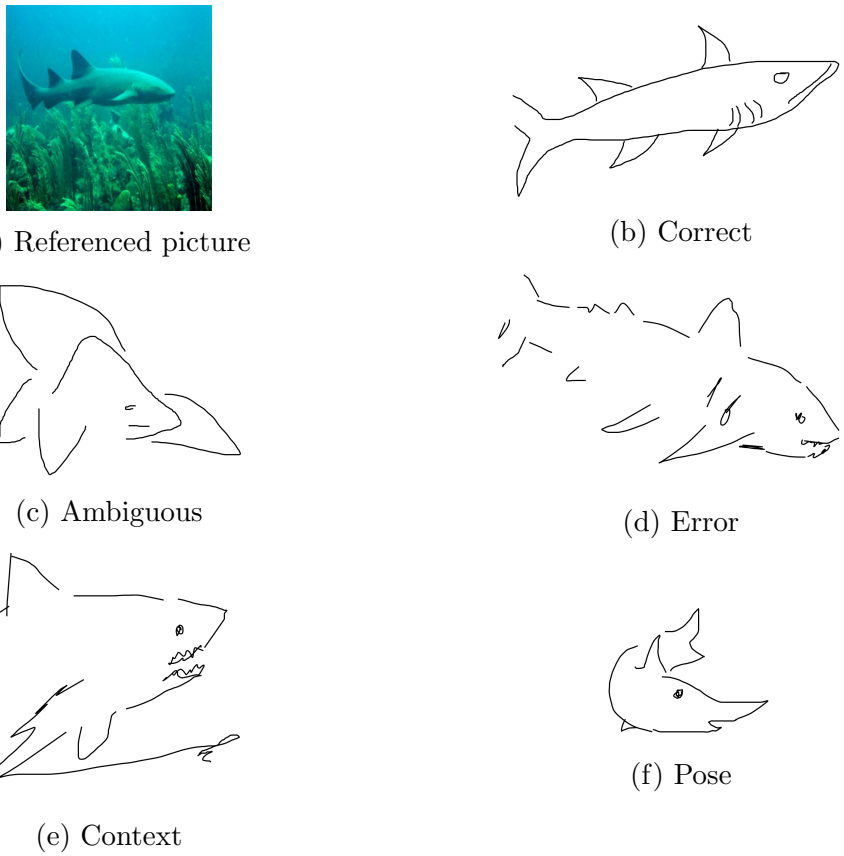
(a) Bad Sketch (b) Good Sketch

Figure 1.1: Example of the differences between quality in owl sketches taken from the "Sketchy" dataset [61].

to distinguish "good" and "bad" input. The assignment of the appropriate quality label improves the interpretability of the results and better characterizes the input sketch, thus allowing sketch classification to be improved. For example, a misclassified, poor quality sketch can be explained if the model returns a bad quality label. However, the first step to achieving this improved performance is to accurately assign these quality labels.

In total, there are five quality labels from the "Sketchy" dataset that will be associated with an input sketch: correct, pose, ambiguous, context, and error. Sangkloy et al. details the "Sketchy" dataset and provides a brief description of each label[61]. The "correct" label is assigned to sketches that accurately reflect the reference image used when drawing. The "ambiguous" label means that the subject of the drawing cannot be discerned. The "pose" label is given to sketches that were drawn in a different pose or perspective than the reference image. The "context" label designates sketches with additional environmental context. Finally, the "error" label is reserved for erroneous sketches that are incorrect representations of the reference image [61]. The "Sketchy" dataset was developed for use in sketch based image retrieval (SBIR) models. SBIR is defined to input a sketch and output a corresponding image of the same subject. The dataset was created by showing the drawer the reference image for a few seconds, then removing the image from view, and lastly capturing

the drawing on a digital tablet. Figure 1.2 shows an example of what quality labels would be assigned to various sketches of a shark.



(a) Referenced picture (b) Correct

(c) Ambiguous (d) Error

(e) Context (f) Pose

Figure 1.2: Example of quality labels associated with various shark sketches[61].

Previous Approaches Previously, significant research has been done in the field of sketch based image retrieval (SBIR), where given a sketch, a model will return an image of the same object or subject [44, 61]. With new tasks emerging, computational models have started to perform sketch classification, predicting the class label of a sketch. An example is the Sketchformer architecture proposed by Ribeiro et al.[58]. which achieved state-of-the-art accuracy for sketch classification in 2020. These previous approaches implemented SBIR, sketch classification, and sketch reconstruction, but none attempted to utilize the quality

labels to provide important information about the sketches.

Although quality labels have not been assigned by a model in the sketch domain, image quality assessment is an active research area in which degradation in images and videos is quantified and corrected. Recent approaches have used neural networks to improve quality of compressed videos [80], underwater images [29], unmanned aerial video [35], and many more. One practical application is in MRIs, where neural networks are used to filter out motion artifacts, leading to better overall diagnoses [45].

Our Approach While models have been developed with improved performance on sketches in the SBIR and sketch classification tasks, there is a lack of models that have been developed for quality prediction. We use transformers trained on the "Sketchy" dataset [61, 66] to predict quality labels. Transformers need sequential input, thus requiring 'Stroke 3' format as input to the transformer. The Scalable Vector Graphics (SVG) version of the "Sketchy" dataset was used, since it can be easily converted to the 'Stroke 3' format defined by the "QuickDraw!" dataset [1]. The "Sketchy" dataset was chosen for the quality labels and amount of sketches it provided. Transformers were chosen since they show good results on sequence input data.

Contributions We made the following technical contributions:

- Trained the Sketchformer architecture on the "Sketchy" dataset to achieve 66% accuracy when predicting one of the five quality labels for a sketch. This dataset contains 75,000 sketches and is the only dataset that contains quality labels for the sketches provided. Previously to this thesis, no attempts by automated systems have been made to predict the quality labels.

- Achieved up to 97% accuracy when quality labels of the "Sketchy" dataset were merged

into “good” vs. “bad” categories. These results compare favorably with recent research on image quality assessment, which, depending on the dataset, achieves correlation values between 75-95% [2].

- Achieved dramatically better performance than a baseline support vector machine (SVM) that was trained on a raster version of the “Sketchy” dataset. Additionally, an altered version of the Sketchformer architecture outperformed the same SVM on the raster version of the same dataset.

- Prepared the “Sketchy” dataset by translating from SVG to ‘Stroke 3’ format and balancing quality labels through random sampling. This new version of the dataset will be available on GitHub.

Chapter 2

Review of Literature

Cross modal learning became an active research area starting in the 2010s when datasets such as PACS (Photos, Art painting, Cartoons, and Sketches) [79] came out, encouraging training to identify objects across multiple different modes of inputs. Around the same time, Sketch Based Image Retrieval (SBIR) became more popular. Inspired by SBIR, additional research was completed in the sketch classification area, providing the foundation for this paper.

2.1 Sketch Based Image Retrieval

Sketch Based Image Retrieval (SBIR) has been around for many years, with some of the earliest examples of SBIR being from the 1990s [8]. However, most of the improvement occurred later with larger datasets, such as the "Sketchy" dataset. Before the "Sketchy" dataset, the datasets were very small, containing only 40 to 300 sketches [21]. The 2010s saw the release of slightly larger datasets, but none were larger than 15,000 sketches [32]. After the "Sketchy" dataset was released, containing 75,000 sketches with 125 categories,

there was an increase in research and application of deep learning to the SBIR task. Larger datasets have since emerged for other sketch tasks, such as "SketchyCOCO" [26], which is used for image generation based on a sketch.

The architectures used for SBIR have changed drastically as time has gone on. Before the invention of deep learning, all the models were dictionary based. For example, Tu Bui et al. extended Gradient Field Histogram of Oriented Gradients (GF-HoG) to allow for color sketches to influence the images retrieved [6]. GF-HoG is a version of HoG [15], which was an influential type of descriptor in the field of computer vision in the early 2000s. The next significant development was the "Bag Of Features" descriptors [22], which computes the similarity of a sketch-image pair by finding edges in the image using an edge detector. As the deep learning field evolved, so did the SBIR field.

The invention of Convolutional Neural Networks (CNNs) was another milestone in the field of SBIR research. These CNNs were adopted into the sketch domain and were applied in many different ways. One way CNNs were applied to SBIR was three branch CNNs with triplet loss. This architecture processes three inputs: a sketch, a positive image, and a negative image [51]. A positive image is a picture of the object that the sketch represents. Conversely, a negative sample image is a picture of an object the sketch does not represent. Siamese networks are a type of a three branch CNN. Siamese networks were experimented with because they used identical subnetworks to predict similarity between images [55]. This concept was extended to sketches using a triplet input of a reference sketch, a positive edge map, and a negative edge map instead of the triplet reference image, positive image, and negative image. The Siamese network use edge maps of images because edge maps are more similar to sketches than the images themselves. "Sketch Me That Shoe" [78] implemented a similar Siamese network. However, the "Sketch Me That Shoe" network was trained to distinguish different shoe types, a more detail oriented task. Siamese networks, triplet loss,

and three branch CNNs allowed neural networks to place related objects close together in the high dimensional learning space and unrelated things farther apart. All three concepts were developed around the same time, because triplet loss is needed in a three branch CNN and a Siamese network is a type of three branch CNN. These creations facilitated greater separation of the classes in the training space, leading to better generalization and performance. These concepts translated well to the SBIR research area.

Because CNNs produced better results than previous algorithms, SBIR task definitions were altered. Previously, a network only needed to retrieve one image from a lot of potentially many images that contain the same subject as the sketch. For example, if the input sketch was of a dog, the network only needed to return one of the many different images of the dog. Due to this lack of specificity, a new task called fine-grained SBIR (FG-SBIR), was defined so only one image matches a given sketch.

Another way that CNN research influenced the SBIR field was zero-shot and one-shot learning [18, 19]. These concepts were applied to sketches to decrease the dependence on labeled data. Another example of this influence is that the four-branch quadruplet network was created to be able to handle color sketches, rather than just back and white sketches [25]. The four branches corresponded to a sketch, a positive image, a negative image, and a positive-negative image. This new positive-negative image resembles the correct object with the wrong coloring. In addition, there were attempts made to speed up SBIR using hashing in "Deep Sketch Hashing" [44]. As CNN research progressed and more variants appeared, sketch research was expanded to solve more complex tasks such as sketch classification, sketch reconstruction, and image generation from freehand sketches. For example, Generative Adversarial Networks (GANs) were trained with image-sketch pairs to generate an image of a freehand sketch [11, 26, 68, 76].

2.2 Sketch Classification

Sketch Classification assigns a class label like 'dog' or 'cat' to a sketch. Research on sketch classification began in 2012 when Eitz et al. [23] created a classification dataset with 20,000 sketches. This dataset was nicknamed "TU-Berlin" after the school that created the dataset. This dataset also provided human performance data to compare the model against. In 2017, the "QuickDraw!" dataset, containing 50 million drawings, was released for sketch classification when deep learning gained popularity and large datasets were desirable.

One of the first algorithms used for sketch classification was pairwise SVMs [62]. Many different methods were applied to the sketch classification task, including 'bag of features' [24] and 'handcrafted features' [20]. However, performance increased dramatically with the invention of the CNN. One example of CNNs created at this time was Sketchnet [81], a CNN using triplet loss, which improved performance on the "TU-Berlin" dataset. Later, Sketch-A-Net [63] was able to outperform humans on the "TU-Berlin" dataset. Recent work trained on "QuickDraw!" started with Sketch-RNN [30], which uses long-term memory units to work with vector sketches. These vector sketches are stroke sequences and can allow for sequence generation, such as predicting endings for an incomplete sketch. LiveSketch [13] further improved the Sketch-RNN structure and implemented SBIR. Both of these works led to Sketchformer [58] and sketch-BERT [41] which utilize transformers to perform sketch classification, sketch retrieval, sketch reconstruction, and a new task known as sketch gestalt. The sketch gestalt task takes in a masked input sketch and fills in the missing parts of the sketch.

There are numerous research papers involving sketches not mentioned above: sketch based video retrieval [59, 60, 72], semi-supervised learning [4, 10], self-supervised learning [5, 42, 50], graph neural networks [12], reinforcement learning [37, 57], and others [52, 70]. Examples

of additional sketch tasks are Sketch Based 3D Shape Retrieval [67, 73], Sketch Based 3D Shape Modeling [27, 48], Sketch Based Incremental Learning [3], Sketch Enhancement [39], Sketch Completion [43, 56, 77], and Sketch Based Image Synthesis [26, 40]. A paper by Yang et al.[75] trained a model to predict class labels using quality metrics and will be discussed in more detail in Section 3.3.

2.3 Signature Verification

Sketch workshops have started to include papers [74] on signature verification because a signature can be considered a type of sketch. Signature verification is not a new research area [31]. However, sketch research and signature verification have started to overlap because of similarities in input formats. Instead of sketches of objects, datasets hold various participants' signatures, both forged and valid. Signature verification can be represented similarly to the 'Stroke 3' and 'Stroke 5' formats. Research in signature verification was completed using CNNs with raster versions of the signature [9, 17, 69, 71]. Li et al. [38] implemented a Transformer on some of the main signature verification datasets. This is an example of a situation where sketch quality would be beneficial. When signing on a screen, the quality of the signature depends on how accurately the screen can capture the electronic pen's movement. A low quality screen would produce an error ridden signature. Additionally, if someone has just scribbled on the screen, it would be important to recognize the ambiguous signature to prevent fraud. Both of these situations could be detected with the quality classification proposed in this thesis.

2.4 Image Quality Assessment

While automated sketch quality prediction has not been studied, image quality assessment (IQA) is a very large research area that spans multiple modalities. The main goal is to restore a distorted image back to the original image. The algorithms are evaluated by calculating a correlation coefficient between the original image and the restored image. There are many different datasets to work with, all having various distortions. Within a dataset, there are multiple types of distortion. For example, the LIVE dataset [64] is an image quality dataset with 5 different distortions placed on the images. These 5 different distortions are JPEG2000 compression, JPEG compression, white noise, Gaussian blur, and simulated fast fading Rayleigh (wireless) channel (transmission loss). The compression degradation and transmission loss can occur when saving/transporting the image. White noise and Gaussian blur imitate camera noise and lens blurring. There are many datasets similar to LIVE, including IVC [7], CSIQ [36], TID2008 [53], TID2013 [54], LIVEMD [33], MDID2013 [65], and CCID2014 [28]. Image quality assessment has expanded to many different applications including video quality assessment [46, 47, 80], underwater image quality prediction [29], MRIs [45], and tone-mapped images [49]. Although image quality assessment does not predict a quality label, the concept of correcting multiple distortions is similar to predicting what type of distortion is in the image.

Image Quality Assessment has many variants of tasks. As a result, the equations/methodologies used to compare performance are different. SSIM (Structural Similarity Index Measure) is used when a model takes in a distorted image and returns a restored image. The SSIM equation compares three main parts of the image: contrast, luminescence, and structure.

Recently, Image Quality Prediction has been introduced [2]. However, Image Quality prediction is very different than sketch quality prediction. Features are extracted from both

the distorted image and the undistorted image. The features extracted are then compared using a correlation method. Two example correlation methods are Spearman rank order correlation coefficient and average Pearson's correlation coefficient. The equations for those computations are below.

Pearson:

$$\rho = \frac{\text{cov}(X, Y)}{\sigma_x \sigma_y} \tag{2.1}$$

Spearman:

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)} \tag{2.2}$$

where d refers to the pairwise distances of the ranks of the variables and n is the number of samples.

These methods cannot be applied to sketches. Images and sketches are very different. Sketches are only black and white and can be represented in vector form. On the other hand, images contain grids of RGB pixel values with a specific resolution. Also, there is not one way to make a "good" sketch. So when Image Quality Assessment creates an undistorted image, there is only one right answer. Since the sketch does not have a corrected version, the evaluation technique cannot be applied. This thesis is predicting a quality label and not trying to create the similar features from both a "good" and a "bad" sketch, like Image Quality Prediction. Thus, that evaluation technique cannot be used in this thesis.

2.5 Transformers

Transformers[66] are trained on sequences of data, which often means strings of information that are time dependent. For example, the sentence "The cat went inside the house" cannot be written as "The house went inside the cat" without changing the meaning of the sentence.

Each word is a piece of information in the sequence, and the order of the words matter in most languages. Before Transformers became popular, recurrent neural networks (RNNs) and/or long short-term memory (LSTMs) were used when working with sequences. Transformers have some benefits over LTSMs and RNNs. RNNs create correlation with samples that are close together in the sequence. Unlike RNNs, transformers allow attention to be placed anywhere in the sequence. All transformer architectures take in sequential data and take advantage of the temporal aspect of the sequential data to achieve better results. LTSMs are made up of separate blocks to process each piece of information in the string. These blocks are trained sequentially to preserve the temporal data used throughout the sequence. Due to the parallel structure shown in Figure 2.1 and the positional encoding, transformers can be trained in parallel and more efficiently than an LSTM network. Transformers improved performance over the LTSM and RNN predecessors. For example, Transformer-XL [14] and BERT [16] exceeded the previous performance on sentence classification and sentence-pair regression tasks.

Transformers were originally created to help with natural language processing [66] where they detailed the new structure using self-attention nodes and positional encoding. Positional encoding is used in place of sequential processing of the sequence. Instead of processing sequentially, the positional encoding equation gives the model information about the relative position of the elements in the sequence. In the original paper on transformers, positional encoding is expressed by the following equations.

$$PE_{(pos,2i)} = \sin(pos/1000^{2i/d_{model}}) \tag{2.3}$$

$$PE_{(pos,2i+1)} = \cos(pos/1000^{2i/d_{model}}) \tag{2.4}$$

where pos is the position and i is the dimension. The input embedding, output embedding,

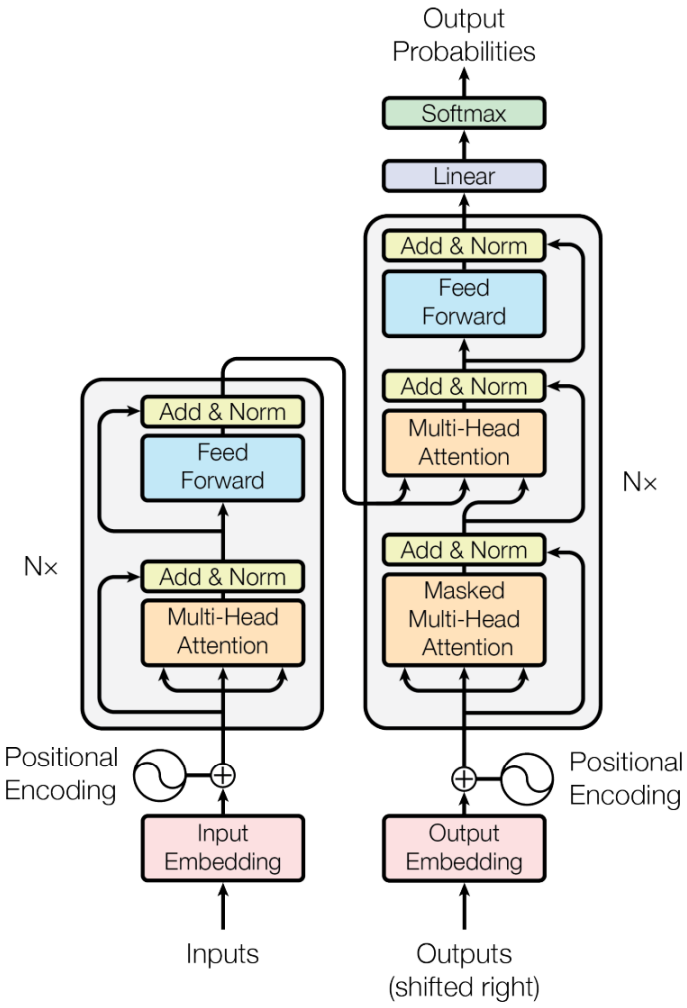


Figure 2.1: Attention is All you Need architecture[66].

and the positional encodings have the same dimension d_{model}. There are multiple positional encoding equations that can be used, but the sine-based and cosine-based equations are the original. The input embedding is created from the input sequence by using an embedding algorithm, which translates it into a vector. There are many embedding algorithms including word2vec, CBOW (continuous bag of words), and skip-gram.

Another part of the success of transformers was the use of multi-head attention nodes. A multi-head attention node concatenates the output of multiple scaled dot-product attention

nodes. Scaled dot-product attention can be defined by the following equation:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.5}$$

where Q is a query matrix, K and V are keys and values, and d_k is the key dimension. The actual values in Q, K, and V depend on location in the network. This attention node is trainable, and since positional encoding is already placed into the sequence, the attention can be learned anywhere in the sequence. Placing the attention at any point in the sequence allows for learning of the most important and most valuable aspects of the sequence.

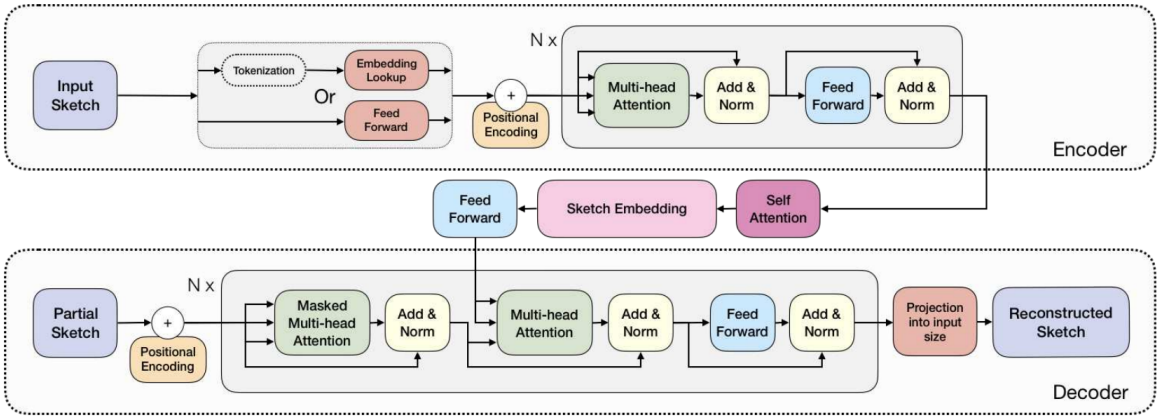


Figure 2.2: Sketchformer architecture[58].

The original transformer was altered by Ribeiro et al.[58] to be optimized with sketches shown in Figure 2.2. The main change this paper made to the structure of the transformer was the creation of a sketch embedding space between the encoder and decoder. This sketch embedding space was created by adding a self attention node after the encoder structure. This sketch embedding space was then used in sketch classification with an additional small feed-forward block. It was also used for SBIR by training two identical CNNs in tandem, as seen in Figure 2.3. One CNN was trained using the raster sketch, and the other CNN was trained using the raster picture. In this case, a raster sketch is a PNG or JPG version of

the sketch. The results of the Sketchformer paper show that this embedding space provides a valuable contribution by surpassing multiple baselines set on different tasks.

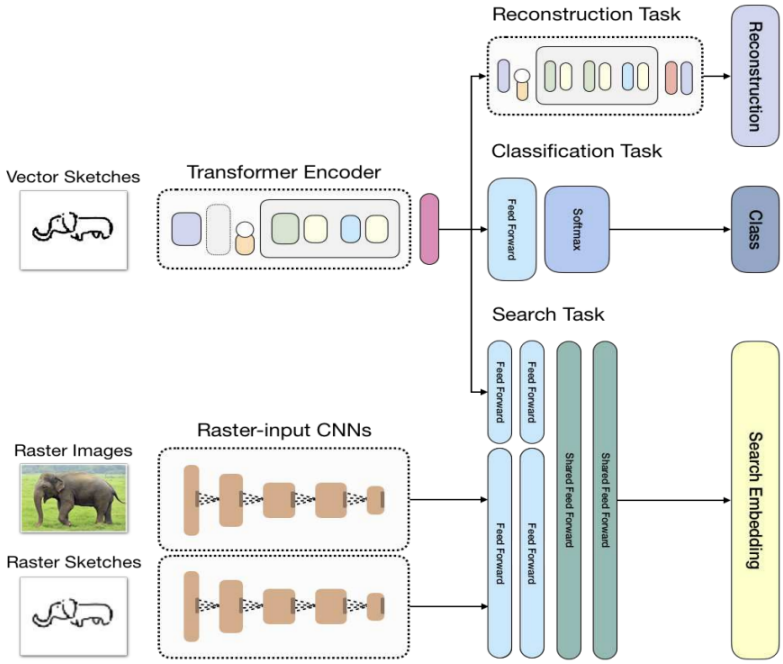


Figure 2.3: Sketchformer architecture with task flow shown [58].

Chapter 3

Method

This thesis aims to show the value in predicting quality labels on sketches. The "Sketchy" dataset provides a quality label for each sketch. To predict these quality labels, the Sketchformer architecture was used. The data preparation process is defined in Section 3.1. Previously, no models have predicted the quality labels on the "Sketchy" dataset. There are 5 quality labels provided by the "Sketchy" dataset: correct, context, ambiguous, pose, and error. Depending on the task, these different labels could be considered good quality or bad quality. Consider the pose label, which shows that an object was drawn in a pose different from the reference image. Some tasks will define pose as good quality, while others will define it as bad quality. For example, pose would not make a difference when predicting an object class of a sketch. However, pose would make a difference when trying to match a sketch to its corresponding image. A study was completed to quantify how the transformer was able to perform when combining the five labels into 2 "good" and "bad" labels.

3.1 Data Preparation

The "Sketchy" dataset has balanced the class labels well. However, the quality labels are not balanced with 64k "correct" sketches, 1k "context" sketches, 2.4k "pose" sketches, 6.1k "pose" sketches and 1k "error" sketches. To solve this problem, oversampling and undersampling were used to achieve a balanced dataset.

There are two main steps of data preparations that happen: SVG to 'Stroke 3' and 'Stroke 3' to 'Stroke 5'. The first conversion leverages code segments from the Sketchformer GitHub site; however, the code structure was written from scratch for this thesis. The second conversion from 'Stroke 3' to 'Stroke 5' was already written into the batching stage of the Sketchformer code. This last conversion from 'Stroke 3' to 'Stroke 5' is used to provide an end sequence variable that is necessary for a transformer.

'Stroke 3', 'Stroke 5', and SVG are all vectorized formats for representing sketches. These three formats can be translated between each other quickly. 'Stroke 3' is defined as a vector of the following triplet:

$$(\delta x, \delta y, p)$$

where p is a pen on/off variable. This results in a matrix of size $3 \times x$ where x is the sequence length. 'Stroke 5' is defined as a vector of the following

$$(\delta x, \delta y, p_1, p_2, p_3)$$

where p_1 is a pen on variable, p_2 is a pen off variable, and p_3 is the end of sequence variable. Variables p_1, p_2, and p_3 are mutually exclusive. This results in a matrix of size $5 \times x$ where x is the sequence length. Since SVG is a vectorized format, the translation from SVG to these different stroke formats is intuitive. Once this data preparation has been completed,

the dataset needed to be balanced.

To balance the dataset, the different quality labels were separated into train and test sections. Then, all labels in the training set were sampled to the same value. Because this under-sampling/over-sampling code was written from scratch, the extra data from each of the different labels was leveraged. For example, the smallest label class was "error", with 823 total sketches. There were 165 sketches saved for testing and the rest were saved for training. Then 165 pictures were randomly sampled from the four other labels to complete the testing set. Then, using the rest of the images not sampled for testing, 6189 sketches were randomly sampled. Repeats were allowed if the label has less than 6189 sketches, and the label was undersampled if they had more than 6189 sketches.

In addition to the preparation of the "Sketchy" dataset to the 'Stroke 3' and 'Stroke 5' format, the "Sketchy" dataset was converted from SVG to PNG format. This allows an SVM baseline to be trained on the dataset. The "split-folders" library in Python was used on the PNG/raster version dataset to determine the train/test split and to oversample or undersample the data. The method to oversample and undersample the data is slightly different from the method that was used to over sample the 'Stroke 3' version of the dataset. The differences are described in the next paragraph. Additionally, a raster version of the Sketchformer architecture was trained on the same PNG version of the "Sketchy" dataset, achieving better performance in comparison to the SVM. This shows that the improvement seen from the SVM to the vector Sketchformer is not due to the differences in data format.

Oversampling using the "split-folders" library requires a common value of the train and test sets. This value must be the same for the largest class and the smallest class. For this dataset, 700 sketches were randomly sampled for training and 100 sketches randomly sampled for testing. Then in the training section, each class is oversampled to the values seen in Table 4.2. This value to oversample to was set by the "split-folders" library.

3.2 Transformers

As described in Section 2.5, Transformers use an encoder-decoder structure. The Sketchformer architecture, which makes minor changes to the original Transformer architecture, is used in this thesis to predict quality labels. The Sketchformer architecture can perform more than one task. Only the sketch classification structure was used here to predict quality. The structure needed for sketch classification, shown in Figure 2.3, is the transformer encoder, a few feed-forward layers, and a softmax layer. This transformer takes the sketch to an encoding/feature space, then the feed-forward layer creates a classification layer and prediction from that. The decoder portion of the transformer was left out since no sequences are being generated. However, the typical structure of a transformer is shown in Figure 2.1.

The Sketchformer architecture created by Ribeiro et al.[58] was altered from the original transformer structure by changing some of the hyperparameters. The structure implemented in this thesis was the same except for the hyperparameters. This is expected because the "Sketchy" dataset was not used in [58].

The Vision Transformer (ViT) implemented a 'Patches' structure [34] that was used to convert the raster sketch into a sequence on which to train. This structure was used in the raster Sketchformer experiment. This method takes in an image and splits it up into a certain amount of smaller 2D chunks. These 2D chunks are flattened to create sequences, positional encoding is added, and then the sequences are passed into the transformer.

3.3 Finding Badly Drawn Bunnies

One of the most important papers, referenced in Section 2.2, was the paper proposed by Yang et al.[75]. This paper created an "unsupervised" quality metric to achieve better performance

on the "QuickDraw!" sketch classification task. This would be a good paper to compare with and to provide a second dataset to run on. However, there were a few reasons this was not able to be completed as a part of this thesis. It should be noted that no code was given in the paper from Yang et al. When asked if the code could be provided, the corresponding author provided the Geometric Aware Classification layer (GACL). This GACL replaces a dense/softmax combination layer and, thus, needs to be trained to assign accurate quality metrics. Additionally, the paper didn't provide much detail about the LSTM architecture implemented. The quality metric, which is a value between 0 and 1, depends on the accuracy of the sketch classification. As is illustrated in Figure 3.1, the quality metric is defined to be the distance the sketch is placed away from the class center. The highest quality is closest to the class center; as the quality decreases, the sketch gets placed farther away from the class center and closer to the class decision boundary. This means that the LSTM model would need to be built from scratch and trained on the dataset to achieve good classification accuracy to give a potentially accurate quality metric.

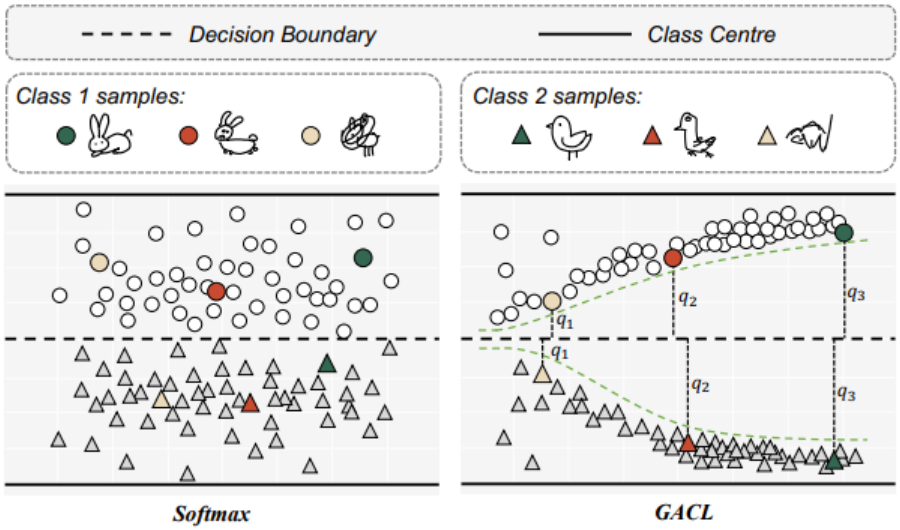


Figure 3.1: Softmax vs. GACL illustration [75].

Figure 3.2 shows sketches of bunnies labeled with the GACL quality metric. Figure 1.2

shows the quality metrics provided from the "Sketchy" dataset. These two figures show the difference between the quality metrics side by side. The main difference is that the "Sketchy" dataset provides five different labels, whereas the GACL shows a quality metric with a value between zero and one.

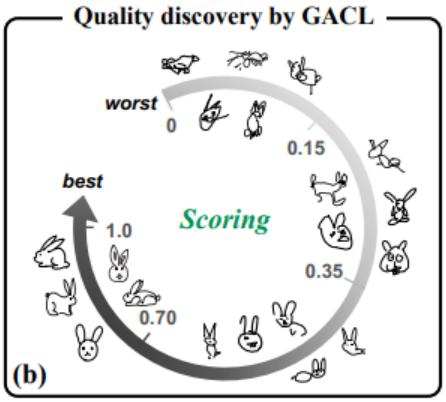


Figure 3.2: GACL quality metric value for bunnies [75].

Additionally, when considering the implications of potentially using the trained quality metric provided by Yang et al., there were some inconsistencies. Firstly, the LSTM and GACL structure would label the "QuickDraw!" dataset with a value between zero and one. A threshold could be set to provide a cut off between "good" and "bad" quality. This could then be used as a second dataset for the groupings experiment. Although this information could be interesting, it does not provide a direct comparison between the two methods. The proposed method would be to try to predict the quality label assigned by Yang et al. instead of a label assigned by a third party. The labels provided by Yang et al. could be biased towards the values that make the most sense to their model, thus providing biased labels to learn from. Second, if the "QuickDraw!" dataset was labeled with the method proposed by Yang et al. to create a second dataset, the same bias issue would be a concern. Since the labels being used to create the accuracy here are not from a third party, they cannot be assured that they are unbiased. The quality metric proposed by Yang et al. is completely

unsupervised, therefore there is no human oversight to make sure that the values coming out make sense for all 50 million sketches in the "QuickDraw!" dataset. The dataset labels should be created to try to ensure unbiased labels where the assigned values are understandable. This situation does not meet these standards, thus the time it would take to achieve this extra part of the thesis was not completed, since the result would be inconclusive. Additionally, the experiment with the second dataset could be more extensive, since threshold for labeling "good" and "bad" quality can be variable.

It should be noted that Yang et al. published the "Finding Badly Drawn Bunnies" paper in CVPR in June 2022. This paper was discovered in September when completing final Literature Review research. The topic of theis thesis and experiment had been decided before this paper was published. This shows that the quality metric topic was considered cutting edge and worthy of researching. While the quality metric predicted was different between this thesis and GACL based quality metric, the concept of predicting a quality metric is the same.

Chapter 4

Experimental Results

These experimental results were implemented using TensorFlow on an NVIDIA p100 GPU on the Virginia Tech Advanced Research Computing (ARC@VT) facilities. A wide range of experiments were completed, and they are described in this chapter.

4.1 Setup

The "Sketchy" dataset was used for training. This data set provides 5 quality labels. Only SBIR components of the dataset have previous baselines, and the quality labels did not. An SVM has been trained on a raster version of this dataset to provide a baseline to compare performance.

Generally quality is thought of as "good quality" or "bad quality" and not one of the following five labels: correct, context, pose, ambiguous, and error. To account for this, experiments were performed with "good" and "bad" labels as well. There are 5 main experiments shown here, predicting the 5 labels described throughout this paper and the 4 combinations that are

possible when combining the 5 labels. To create the "good" and "bad" labels, combinations of between 2 and 4 of the 5 labels were created. These combinations were picked because the ordering from correct to error shown in Table 4.1 listed the labels as best quality to the worst quality. Picking combinations that disrupted this ordering would be illogical, and thus those combinations were disregarded. Table 4.1 provides a description of what labels make up the different combinations of the five labels and what label they were given (good or bad). These different groups emulate different tasks that require different definitions of quality.

Table 4.1: Description of Good vs. Bad Experiment Grouping.

Labels	Group 1	Group 2	Group 3	Group 4
Correct	Good	Good	Good	Good
Context	Good	Good	Good	Bad
Pose	Good	Good	Bad	Bad
Ambiguous	Good	Bad	Bad	Bad
Error	Bad	Bad	Bad	Bad

There are two different prepared versions of the dataset, the vector/SVG version and the raster/PNG version. Table 4.2 below shows the original values and the oversampled values. The method used to balance the dataset is described in Section 3.1.

Table 4.2: Number of sketches per class in each dataset.

Dataset Version	Correct	Context	Pose	Ambiguous	Error
Original	64,393	1,049	2,433	6,191	892
Raster Over sampled	63,793	63,793	63,793	63,793	63,793
Vector Over sampled	6,189	6,189	6,189	6,189	6,189

It should be noted that while oversampling does not change the performance of an SVM, the dataset was oversampled for the raster Sketchformer. However, the SVM should be compared to the raster Sketchformer directly, so the oversampled version of the dataset was

used. Since SVMs make decisions based off the support vectors, the accuracy should be the same regardless of if the dataset is oversampled.

4.2 Results

Table 4.3 summarizes the main results of this experiment. This shows two baselines compared to the vector Sketchformer architecture. The PNG version of the "Sketchy" dataset was created to allow the SVM baseline method to be applied. To show performance increase using a transformer, a raster version of the Sketchformer architecture was applied to the same PNG dataset. Then the vector version of the Sketchformer architecture was applied to the SVG version of the dataset.

For the SVM baseline, a simple version of the SVM was used. The PNG input image was of size $(64, 64, 3)$. Two convolution layers and two max pool layers were used to extract features from the images. The convolutional layers has 32 filters with a kernel size of 3. The max pool layers used a pool size of (2,2). The SVM algorithm was then applied to the extracted features. A structure proposed by Dosovitskiy et al.[34] allowed PNG images to be converted to sequences so that transformers could be implemented on the PNG version of the dataset. This structure takes out smaller "patches" of the larger image, then flattens them to get the sequence to train on. Once the patches were implemented, the Sketchformer architecture was added to create the raster Sketchformer. The SVM and raster Sketchformer are both trained on the PNG version of the dataset, whereas the raster Sketchformer and the vector Sketchformer have the same encoder structure. This allows us to compare performance of the baseline and the Sketchformer architecture while also utilizing the vector version of the dataset.

The SVM was the most basic algorithm tested, which is reflected in the results. We can

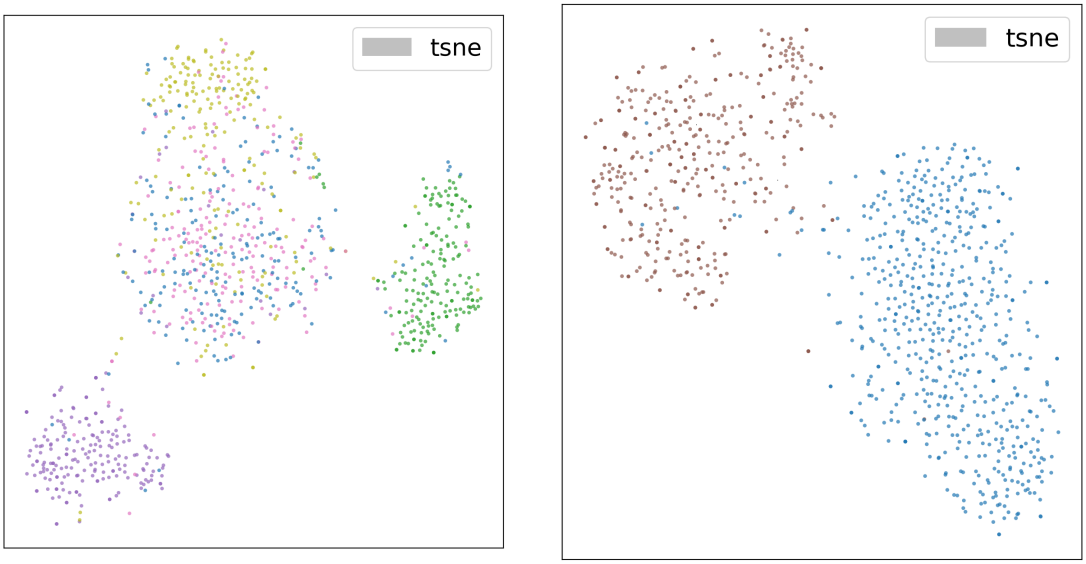
Table 4.3: Comparison of proposed method to a baseline method.

Experimental Accuracy					
Architecture	5 Labels	Group 1	Group 2	Group 3	Group 4
SVM	3.25%	54.5%	54%	54.5%	50%
Raster Sketchformer	42.6%	90.1%	65.1%	63.6%	79.1%
Vector Sketchformer (ours)	66.6%	97.1%	97.8%	78.5%	79.7%

see that the Sketchformer architecture achieves better results using raster presentation, increasing the performance by about 40% for the 5-label experiment. As expected, both of the transformer-based architecture significantly outperformed the SVM baseline. On the grouping experiments, the SVM baseline barely outperformed random choice. The feature space created by the convolutional layers did not allow for separation. The encoder aspect of the transformers learned a more descriptive feature space that allowed for separation of the sketches based on the quality label. The raster Sketchformer had a lower accuracy than the vector Sketchformer because the raster Sketchformer lost important information when creating the sequences from the PNG images.

The results in the vector Sketchformer row of Table 4.3 show some differences in accuracies based on groupings. If Table 4.1 is referenced, it can be seen that the first two groupings were the most intuitive "good" versus "bad" split. In groups 1 and 2, "correct", "context", and "pose" are obviously good quality labels and "error" is obviously a bad quality label. Groups 3 and 4 start labeling "context", and "pose" as bad labels which is not as intuitive as groups 1 and 2. This decrease in performance was expected for group 3 and group 4 and validated the good results in group 1 and group 2. The 5 label version of the dataset was left out of this analysis because the change in class size influences performance, and thus no conclusions could be drawn.

The most generic "good" and "bad" split is Group 2. It will be used as the main point of



(a) 5 label

(b) Group 2

Figure 4.1: T-SNE embeddings.

some qualitative analysis. Figure 4.1b shows a T-SNE plot for group 2. T-SNE stands for t-stochastic neighbor embedding. This algorithm takes a high dimensional embedding space and represents it in two dimensions for visualization. Figure 4.1b shows better separation between the groups in comparison with Figure 4.1a. This is because the 5 label experiment only has an accuracy of 66% whereas group 2 has an accuracy of 97%. Thus, the two labels in group 2 will be separated more than the 5 label experiment. Since group 2 is the most common grouping of the "bad" versus "good" experiments, it would make sense that the accuracy would increase.

Table 4.1 shows the "good" versus "bad" labels. We can see that 'context' and 'pose' are labeled as good for group 2. Since group 2 has been labeled the most common or generic "good" versus "bad" grouping, it should be verified that 'pose' and 'context' are visually good quality. Figure 4.2 gives 5 examples for different subjects that were given a 'context' label. Figure 4.3 gives 5 examples of different subjects that were given a 'pose' label. The 'context' label can be seen to have extra details that are not related to the subject of the

(a) Owl

(b) Frog

(c) Snake

(d) Bird

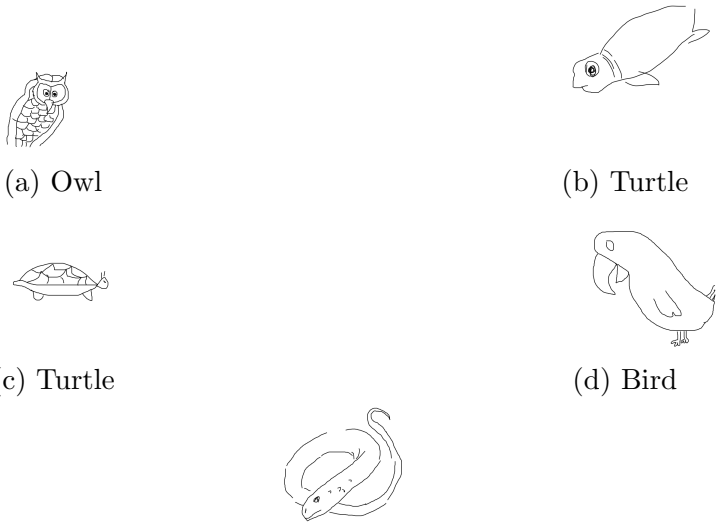
(e) Bird

Figure 4.2: Context label examples.

sketch. For example, Subfigure 4.2d shows a bird in water. The extra details of the water cause this image to be labeled with 'context' and not 'correct'. Similarly, for the 'pose' label, the orientation of the sketch does not quite match the reference image. However, these sketches can be considered "good" quality even though they are not quite what was asked for.

Additionally, Figure 4.4 and Figure 4.5 show some examples of correctly classified sketches. Figure 4.6 shows a portion of the misclassified sketches. There is a slight trend for sketches with lots of dark, filled in areas to be misclassified. This would make sense because the sequence to show that would be a lot of dense back and forth movement. This back and forth movement could have confused the model, causing some of the misclassification. Not all the misclassified sketches have this attribute, just enough to be notable.

For more analysis, the ROC curve and confusion matrix for the group 2 experiment are provided. The Confusion matrix in Figure 4.7 and the ROC curve is Figure 4.8. Both are consistent with the other results from the group 2 experiments.



(a) Owl

(b) Turtle

(c) Turtle

(d) Bird

(e) Snake

Figure 4.3: Pose label examples.

4.3 Ablation Study

An ablation study was completed to increase performance on the vector Sketchformer architecture. The Sketchformer architecture that was trained on the SVG version of the "Sketchy" dataset had 3 main hyperparameters: d_{model}, *dff*, and $numLayers$. The hyperparameter d_{model} was the first studied, and defines the size of the embedding space between the encoder and decoder space. Table 4.4 shows the results for various d_{model} sizes. Group 4 showed an increase in performance of approximately 4.5% as d_{model} increased, while group 2 showed a 4.5% decrease in performance as d_{model} increased. Group 3 also showed about a 2% increase in accuracy as d_{model} increased. All the other groups deviated by less than a percent. Groups 2 and 3 followed an expected trend, with performance increasing as the feature/embedding space increased. The embedding space became more descriptive, allowing for better separation of the classes. However, group 2 did not follow this pattern. Group 2 was the most logical separation of the classes, so it makes sense that a more complex embedding space could result in overfitting when the separation is easily described using fewer dimensions.

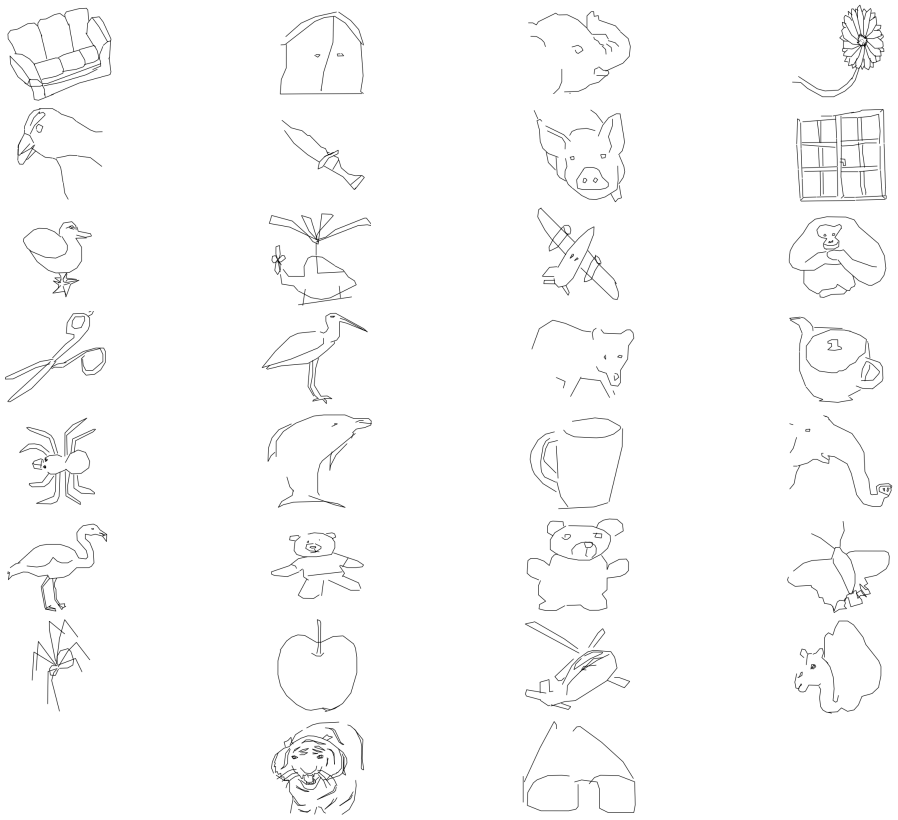


Figure 4.4: Correctly classified good sketches.

The next hyperparameter studied was *dff*, which is an intermediate value between the embedding space and the final classification layer. In some ways, this hyperparameter had the least impact on the results, both theoretically and practically. We can see that most of the group accuracies differed less than a percentage point as *dff* changes. The only groups that showed a significant change using the *dff* variable were groups 2 and 4.

The last hyperparameter studied was $numLayers$. This variable had significant impact on the performance of the architecture. This is shown in Table 4.6 when a value of 2 for $numLayers$ showed a drastic drop in accuracy over all the experiments. While comparing performance between $numLayers$ of 8 vs 4, there is less change than seen in a $numLayers$ of 2. Most experiments, except group 3, stayed within a percent change of one another.

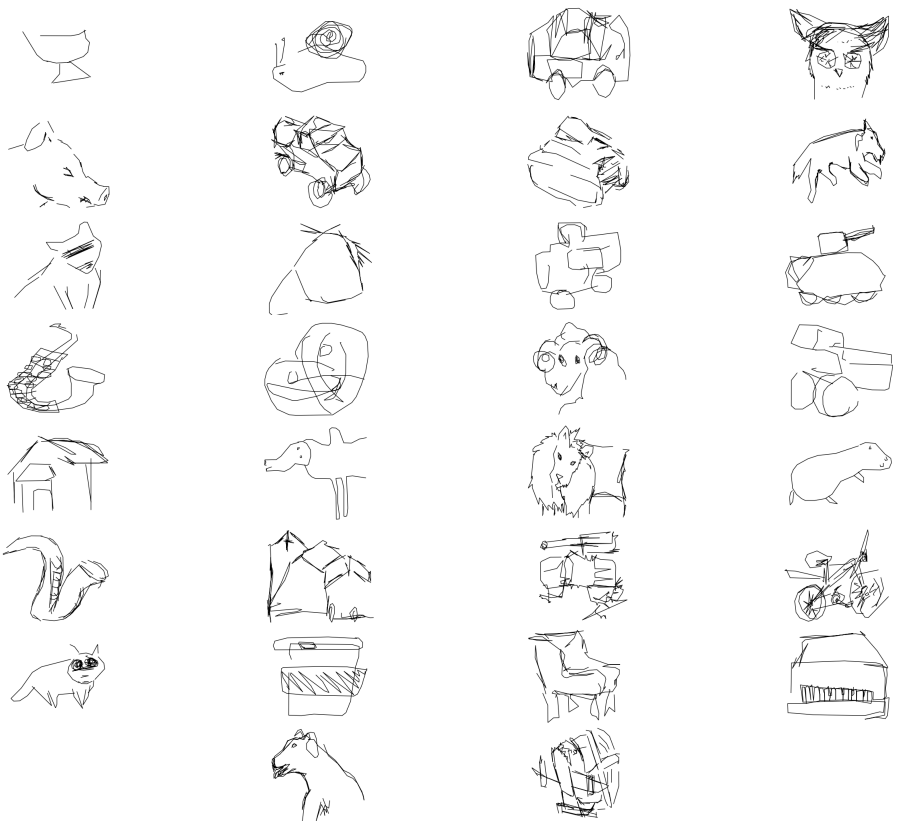


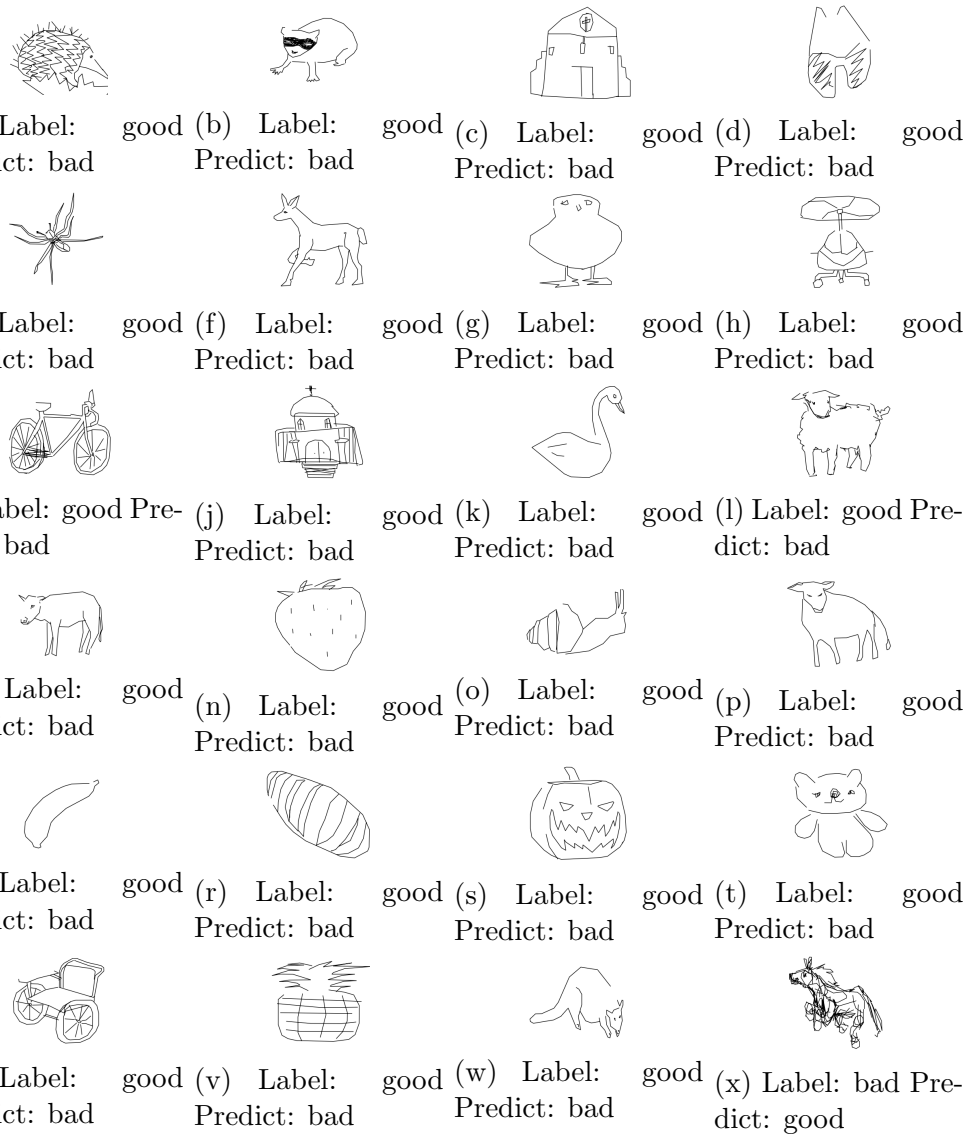
Figure 4.5: Correctly classified bad sketches.

Table 4.4: Vector Sketchformer model accuracy for different values of d_{model}

Experimental Accuracy					
d_{model}	5 Labels	Group 1	Group 2	Group 3	Group 4
512	66.9%	97.9%	94.1%	77.3%	81.2%
256	66.6%	97.1%	97.8%	78.5%	79.7%
128	67.5%	97.0%	98.3%	75.5%	78.7%
64	67.7%	97.9%	98.5%	76.7%	76.7%

Table 4.5: Vector Sketchformer model accuracy for different values of *dff*

Experimental Accuracy					
dff	5 Labels	Group 1	Group 2	Group 3	Group 4
1024	66.9%	97.0%	94.1%	77.3%	81.2%
512	66.6%	97.1%	97.8%	78.5%	79.7%
256	68.0%	97.1%	98.7%	78.2%	77%



(a) Label: good Predict: bad (b) Label: good Predict: bad (c) Label: good Predict: bad (d) Label: good Predict: bad

(e) Label: good Predict: bad (f) Label: good Predict: bad (g) Label: good Predict: bad (h) Label: good Predict: bad

(i) Label: good Predict: bad (j) Label: good Predict: bad (k) Label: good Predict: bad (l) Label: good Predict: bad

(m) Label: good Predict: bad (n) Label: good Predict: bad (o) Label: good Predict: bad (p) Label: good Predict: bad

(q) Label: good Predict: bad (r) Label: good Predict: bad (s) Label: good Predict: bad (t) Label: good Predict: bad

(u) Label: good Predict: bad (v) Label: good Predict: bad (w) Label: good Predict: bad (x) Label: bad Predict: good

Figure 4.6: Misclassified sketches.

Confusion Matrix

	Positive	Negative
Positive	341	27
Negative	1	269

Figure 4.7: Confusion Matrix for group 2 experiment.

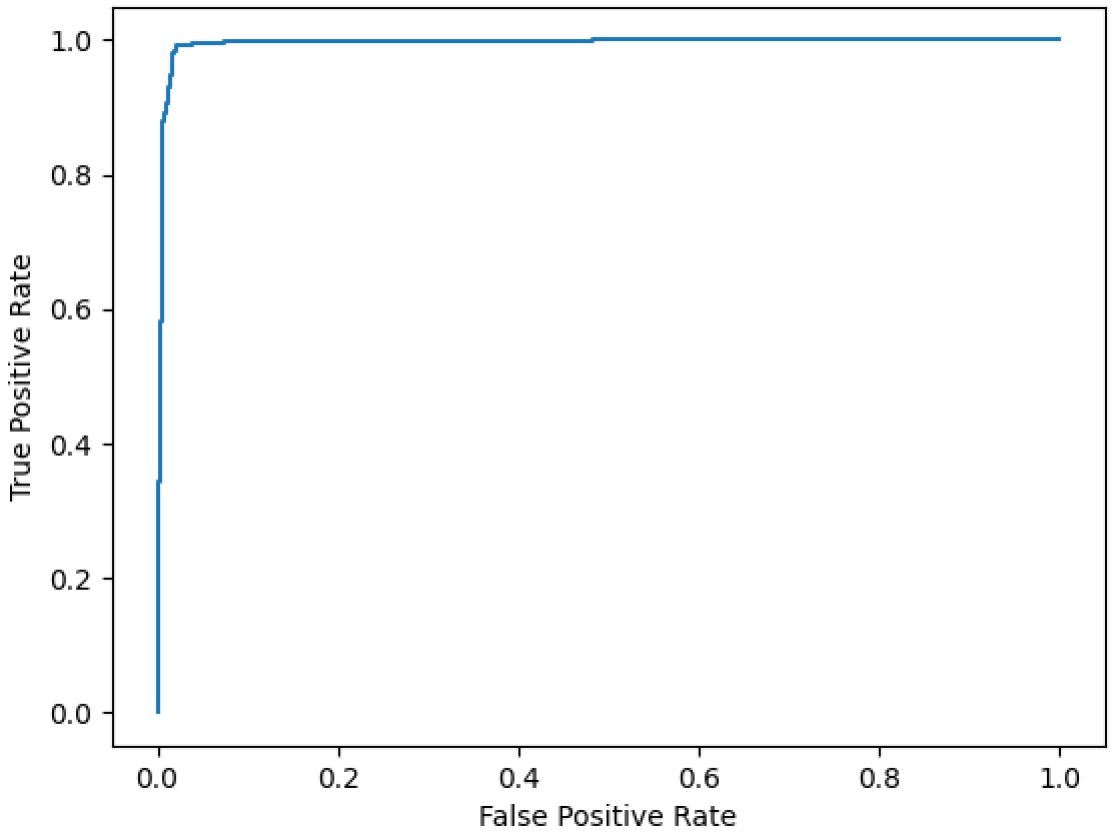


Figure 4.8: ROC curve for group 2 experiment.

Table 4.6: Vector Sketchformer model accuracy for different values of numLayers.

Experimental Accuracy					
numLayers	5 Labels	Group 1	Group 2	Group 3	Group 4
8	66.3%	97.6%	98.5%	75.6%	79.6%
4	66.6%	97.1%	97.8%	78.5%	79.7%
2	32.2%	34.8%	40.3%	53.0%	56.4%

Chapter 5

Conclusions

5.1 Discussion

This thesis presents the first system that performs quality prediction on the "Sketchy" dataset. Previous work from Yang et al. provide quality metrics that humans did not create and potentially are not as logical as the quality labels predicted by this thesis. The method implemented in this thesis improves performance on 5 different experiments in comparison to the SVM baseline. Additionally, since the SVM baseline must be run on the raster dataset, another version of the Sketchformer was created to train on the raster dataset to provide a direct comparison. The accuracy achieved shows the value of the Sketchformer's prediction of the quality labels. The improved accuracy shows that these quality labels are important moving forward to improve performance on other tasks. It also should be noted that quality prediction for sketches would be very beneficial for applications such as law enforcement sketches, where a sketch artist creates a sketch from a witness's descriptions.

5.2 Future Work

One option to extend this thesis is to train quality labels in addition to another task. For example, the Sketchformer architecture can complete 3 main tasks: SBIR, sketch classification, and sketch reconstruction. This model could be altered some way to be able to predict both a quality label and perform another task at the same time. This extension would allow for some of the concepts alluded to in this thesis to be implemented. For example, it would be expected that training SBIR and the quality prediction would result in different quality labels prioritized than the labels prioritized in sketch classification and quality prediction. Additionally, this thesis could be extended by labeling another dataset so that the experiments could be conducted on more than one dataset. Due to time limits, this was unable to be completed as part of my thesis. The sketch quality concept is just starting to be explored, some research is limited by only a single dataset providing a quality metric. Another interesting concept to explore would be to see how quality would change in different age, cultural, and experience backgrounds. Some people can draw better than others, it would be interesting to see if different cultures produce different quality of sketches. I would think that age would cause some changes in quality as well. Experimenting with how the attributes of the drawer affect the sketches would be good to do.

Bibliography

[1] Quick, Draw! URL https://quickdraw.withgoogle.com/.

[2] Sören Becker, Thomas Wiegand, and Sebastian Bosse. Curiously effective features for image quality prediction. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 1399–1403, 2021. doi: 10.1109/ICIP42928.2021.9506756.

[3] A. Bhunia, V. Gajjala, S. Koley, R. Kundu, A. Sain, T. Xiang, and Y. Song. Doodle it yourself: Class incremental learning by drawing a few sketches. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2283–2292, Los Alamitos, CA, USA, jun 2022. IEEE Computer Society. doi: 10.1109/CVPR52688.2022.00233. URL https://doi.ieeecomputersociety.org/10.1109/CVPR52688.2022.00233.

[4] Ayan Kumar Bhunia, Pinaki Nath Chowdhury, Aneeshan Sain, Yongxin Yang, Tao Xiang, and Yi-Zhe Song. More photos are all you need: Semi-supervised learning for fine-grained sketch based image retrieval. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4245–4254, 2021.

[5] Ayan Kumar Bhunia, Pinaki Nath Chowdhury, Yongxin Yang, Timothy M. Hospedales, Tao Xiang, and Yi-Zhe Song. Vectorization and rasterization: Self-supervised learning

for sketch and handwriting. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5668–5677, 2021.

[6] Tu Bui and John Collomosse. Scalable sketch-based image retrieval using color gradient features. In *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, pages 1012–1019, 2015. doi: 10.1109/ICCVW.2015.133.

[7] Patrick Le Callet and Florent Autrusseau. Subjective quality assessment irccyn/ivc database. 2004.

[8] Yin Chans, Zhibin Lei, Daniel Lopresti, and S. Y. Kung. A feature-based approach for image retrieval by sketch. *Proceedings of SPIE - The International Society for Optical Engineering*, 3229:220–231, 1997. ISSN 0277-786X. doi: 10.1117/12.290343. Copyright: Copyright 2011 Elsevier B.V., All rights reserved.; Multimedia Storage and Archiving Systems II ; Conference date: 03-11-1997 Through 03-11-1997.

[9] Pooja Chaturvedi and Anamika Jain. Feature ensemble based method for verification of offline signature images. In *2022 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COM-IT-CON)*, volume 1, pages 710–714, 2022. doi: 10.1109/COM-IT-CON54601.2022.9850628.

[10] Chaofeng Chen, Wei Liu, Xiao Tan, and Kwan-Yee Kenneth Wong. Semi-supervised learning for face sketch synthesis in the wild. pages 216–231, 12 2018. ISBN 978-3-030-20886-8. doi: 10.1007/978-3-030-20887-5_14.

[11] Wengling Chen and James Hays. Sketchygan: Towards diverse and realistic sketch to image synthesis. pages 9416–9425, 06 2018. doi: 10.1109/CVPR.2018.00981.

[12] S. Cheng, Y. Ren, and Y. Yang. Ssr-gnns: Stroke-based sketch representation with graph neural networks. In *2022 IEEE/CVF Conference on Computer Vision and Pattern*

Recognition Workshops (CVPRW), pages 5127–5137, Los Alamitos, CA, USA, jun 2022. IEEE Computer Society. doi: 10.1109/CVPRW56347.2022.00561. URL https://doi.ieeecomputersociety.org/10.1109/CVPRW56347.2022.00561.

[13] John P. Collomosse, Tu Bui, and Hailin Jin. Livesketch: Query perturbations for guided sketch-based visual search. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2874–2882, 2019.

[14] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. pages 2978–2988, 01 2019. doi: 10.18653/v1/P19-1285.

[15] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, 2005. doi: 10.1109/CVPR.2005.177.

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

[17] Sounak Dey, Anjan Dutta, Juan Ignacio Toledo, Suman K. Ghosh, Josep Lladós, and Umapada Pal. Signet: Convolutional siamese network for writer independent offline signature verification. *CoRR*, abs/1707.02131, 2017. URL http://arxiv.org/abs/1707.02131.

[18] Sounak Dey, Pau Riba, Anjan Dutta, Josep Llados, and Yi-Zhe Song. Doodle to search:

Practical zero-shot sketch-based image retrieval. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[19] A. Dutta and Z. Akata. Semantically tied paired cycle consistency for zero-shot sketch-based image retrieval. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5084–5093, Los Alamitos, CA, USA, jun 2019. IEEE Computer Society. doi: 10.1109/CVPR.2019.00523. URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2019.00523.

[20] Mathias Eitz, Kristian Hildebrand, Tamy Boubekeur, and Marc Alexa. A descriptor for large scale image retrieval based on sketched feature lines. In *Proceedings of the 6th Eurographics Symposium on Sketch-Based Interfaces and Modeling*, SBIM '09, page 29–36, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605586021. doi: 10.1145/1572741.1572747. URL https://doi.org/10.1145/1572741.1572747.

[21] Mathias Eitz, Kristian Hildebrand, Tamy Boubekeur, and Marc Alexa. An evaluation of descriptors for large-scale image retrieval from sketched feature lines. *Computers and Graphics*, 34(5):482–498, 2010. ISSN 0097-8493. doi: https://doi.org/10.1016/j.cag.2010.07.002. URL https://www.sciencedirect.com/science/article/pii/S0097849310001068. CAD/GRAPHICS 2009 Extended papers from the 2009 Sketch-Based Interfaces and Modeling Conference Vision, Modeling and Visualization.

[22] Mathias Eitz, Kristian Hildebrand, Tamy Boubekeur, and Marc Alexa. Sketch-based image retrieval: Benchmark and bag-of-features descriptors. *IEEE Transactions on Visualization and Computer Graphics*, 17(11):1624–1636, 2011. doi: 10.1109/TVCG.2010.266.

[23] Mathias Eitz, James Hays, and Marc Alexa. How do humans sketch objects? *ACM Trans. Graph. (Proc. SIGGRAPH)*, 31(4):44:1–44:10, 2012.

[24] L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 524–531 vol. 2, 2005. doi: 10.1109/CVPR.2005.16.

[25] A. Fuentes and J. M. Saavedra. Sketch-qnet: A quadruplet convnet for color sketch-based image retrieval. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2134–2141, Los Alamitos, CA, USA, jun 2021. IEEE Computer Society. doi: 10.1109/CVPRW53098.2021.00242. URL https://doi.ieeecomputersociety.org/10.1109/CVPRW53098.2021.00242.

[26] Chengying Gao, Qi Liu, Qi Xu, Limin Wang, Jianzhuang Liu, and Changqing Zou. Sketchycoco: Image generation from freehand scene sketches. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5173–5182, 2020. doi: 10.1109/CVPR42600.2020.00522.

[27] Chenjian Gao, Qian Yu, Lu Sheng, Yi-Zhe Song, and Dong Xu. Sketchsampler: Sketch-based 3d reconstruction via view-dependent depth sampling. In *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part I*, page 464–479, Berlin, Heidelberg, 2022. Springer-Verlag. ISBN 978-3-031-19768-0. doi: 10.1007/978-3-031-19769-7_27. URL https://doi.org/10.1007/978-3-031-19769-7_27.

[28] Ke Gu, Guangtao Zhai, Weisi Lin, and Min Liu. The analysis of image contrast: From quality assessment to automatic enhancement. *IEEE Transactions on Cybernetics*, 46 (1):284–297, 2016. doi: 10.1109/TCYB.2015.2401732.

[29] Pengfei Guo, Hantao Liu, Delu Zeng, Tao Xiang, Leida Li, and Ke Gu. An underwater image quality assessment metric. *IEEE Transactions on Multimedia*, pages 1–14, 2022. doi: 10.1109/TMM.2022.3187212.

[30] David Ha and Douglas Eck. A neural representation of sketch drawings. In *ICLR 2018*, 2018. URL https://openreview.net/pdf?id=Hy6GHpkCW. 2018.

[31] Luiz G. Hafemann, Robert Sabourin, and Luiz S. Oliveira. Offline handwritten signature verification — literature review. In *2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–8, 2017. doi: 10.1109/IPTA.2017.8310112.

[32] Rui Hu and John Collomosse. A performance evaluation of gradient field hog descriptor for sketch based image retrieval. *Computer Vision and Image Understanding*, 117(7): 790–806, 2013. ISSN 1077-3142. doi: https://doi.org/10.1016/j.cviu.2013.02.005. URL https://www.sciencedirect.com/science/article/pii/S1077314213000349.

[33] Dinesh Jayaraman, Anish Mittal, Anush K. Moorthy, and Alan C. Bovik. Objective quality assessment of multiply distorted images. In *2012 Conference Record of the Forty Sixth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, pages 1693–1697, 2012. doi: 10.1109/ACSSC.2012.6489321.

[34] Alexander Kolesnikov, Alexey Dosovitskiy, Dirk Weissenborn, Georg Heigold, Jakob Uszkoreit, Lucas Beyer, Matthias Minderer, Mostafa Dehghani, Neil Houlsby, Sylvain Gelly, Thomas Unterthiner, and Xiaohua Zhai. An image is worth 16x16 words: Transformers for image recognition at scale. 2021.

[35] Qi Kuang, Xin Jin, Qinping Zhao, and Bin Zhou. Deep multimodality learning for uav video aesthetic quality assessment. *IEEE Transactions on Multimedia*, 22(10):2623–2634, 2020. doi: 10.1109/TMM.2019.2960656.

[36] Eric Larson and Damon Chandler. Most apparent distortion: Full-reference image quality assessment and the role of strategy. *J. Electronic Imaging*, 19:011006, 01 2010. doi: 10.1117/1.3267105.

[37] Ganghun Lee, Minji Kim, Minsu Lee, and Byoung-Tak Zhang. From scratch to sketch: Deep decoupled hierarchical reinforcement learning for robotic sketching agent. In *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, may 2022. doi: 10.1109/icra46639.2022.9811858. URL https://doi.org/10.1109%2Ficra46639.2022.9811858.

[38] Huan Li, Ping Wei, Zeyu Ma, Changkai Li, and Nanning Zheng. Offline signature verification with transformers. In *2022 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, 2022. doi: 10.1109/ICME52920.2022.9859886.

[39] Jia Li, Nan Gao, Tong Shen, Wei Zhang, Tao Mei, and Hui Ren. Sketchman: Learning to create professional sketches. In *Proceedings of the 28th ACM International Conference on Multimedia*, MM '20, page 3237–3245, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379885. doi: 10.1145/3394171.3413720. URL https://doi.org/10.1145/3394171.3413720.

[40] Yuhang Li, Xuejin Chen, Binxin Yang, Zihan Chen, Zhihua Cheng, and Zheng-Jun Zha. DeepFacePencil. In *Proceedings of the 28th ACM International Conference on Multimedia*. ACM, oct 2020. doi: 10.1145/3394171.3413684. URL https://doi.org/10.1145%2F3394171.3413684.

[41] H. Lin, Y. Fu, X. Xue, and Y. Jiang. Sketch-bert: Learning sketch bidirectional encoder representation from transformers by self-supervised learning of sketch gestalt. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6757–6766, Los Alamitos, CA, USA, jun 2020. IEEE Computer Society. doi:

10.1109/CVPR42600.2020.00679. URL https://doi.ieeecomputersociety.org/10.1109/CVPR42600.2020.00679.

[42] Bingchen Liu, Yizhe Zhu, Kunpeng Song, and A. Elgammal. Self-supervised sketch-to-image synthesis. In *AAAI Conference on Artificial Intelligence*, 2020.

[43] Fang Liu, Xiaoming Deng, Yu-Kun Lai, Yong-Jin Liu, Cuixia Ma, and Hongan Wang. Sketchgan: Joint sketch completion and recognition with generative adversarial network. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5823–5832, 2019. doi: 10.1109/CVPR.2019.00598.

[44] L. Liu, F. Shen, Y. Shen, X. Liu, and L. Shao. Deep sketch hashing: Fast free-hand sketch-based image retrieval. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2298–2307, Los Alamitos, CA, USA, jul 2017. IEEE Computer Society. doi: 10.1109/CVPR.2017.247. URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2017.247.

[45] Siyuan Liu, Kim-Han Thung, Weili Lin, Pew-Thian Yap, and Dinggang Shen. Real-time quality assessment of pediatric mri via semi-supervised deep nonlocal residual neural networks. *IEEE Transactions on Image Processing*, 29:7697–7706, 2020. doi: 10.1109/TIP.2020.2992079.

[46] Xiwen Liu, Xiaoming Tao, Mai Xu, Yafeng Zhan, and Jianhua Lu. An eeg-based study on perception of video distortion under various content motion conditions. *IEEE Transactions on Multimedia*, 22(4):949–960, 2020. doi: 10.1109/TMM.2019.2934425.

[47] Yongxu Liu, Jinjian Wu, Aobo Li, Leida Li, Weisheng Dong, Guangming Shi, and Weisi Lin. Video quality assessment with serial dependence modeling. *IEEE Transactions on Multimedia*, 24:3754–3768, 2022. doi: 10.1109/TMM.2021.3107148.

[48] Zhaoliang Lun, Matheus Gadelha, Evangelos Kalogerakis, Subhransu Maji, and Rui Wang. 3d shape reconstruction from sketches via multi-view convolutional networks. pages 67–77, 10 2017. doi: 10.1109/3DV.2017.00018.

[49] Saeed Mahmoudpour and Peter Schelkens. A multi-attribute blind quality evaluator for tone-mapped images. *IEEE Transactions on Multimedia*, 22(8):1939–1954, 2020. doi: 10.1109/TMM.2019.2950570.

[50] Javier Morales, Nils Murrugarra-Llerena, and Jose M. Saavedra. Leveraging unlabeled data for sketch-based understanding. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 5149–5158, 2022. doi: 10.1109/CVPRW56347.2022.00563.

[51] U. Muhammad, Y. Yang, Y. Song, T. Xiang, and T. M. Hospedales. Learning deep sketch abstraction. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8014–8023, Los Alamitos, CA, USA, jun 2018. IEEE Computer Society. doi: 10.1109/CVPR.2018.00836. URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2018.00836.

[52] Narges Honarvar Nazari and Adriana Kovashka. The role of shape for domain generalization on sparsely-textured images. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 5116–5126, 2022. doi: 10.1109/CVPRW56347.2022.00560.

[53] Nikolay Ponomarenko, Vladimir Lukin, Alexander Zelensky, Karen Egiazarian, Marco Carli, and Federica Battisti. Tid2008 - a database for evaluation of full-reference visual quality assessment metrics. *Advances of Modern Radioelectronics*, 10:30–45, 01 2009.

[54] Nikolay Ponomarenko, Oleg Ieremeiev, Vladimir Lukin, Karen Egiazarian, Lina Jin, Jaakko Astola, Benoit Vozel, Kacem Chehdi, Marco Carli, Federica Battisti, and C.-

C. Jay Kuo. Color image database tid2013: Peculiarities and preliminary results. In *European Workshop on Visual Information Processing (EUVIP)*, pages 106–111, 2013.

[55] Yonggang Qi, Yi-Zhe Song, Honggang Zhang, and Jun Liu. Sketch-based image retrieval via siamese convolutional neural network. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 2460–2464, 2016. doi: 10.1109/ICIP.2016.7532801.

[56] Yonggang Qi, Guoyao Su, Qiang Wang, Jie Yang, Kaiyue Pang, and Yi-Zhe Song. Generative sketch healing. *Int. J. Comput. Vision*, 130(8):2006–2021, aug 2022. ISSN 0920-5691. doi: 10.1007/s11263-022-01623-7. URL https://doi.org/10.1007/s11263-022-01623-7.

[57] Shintya Rezky Rahmayanti, Chastine Fatichah, and Nanik Suciati. Sketch generation from real object images using generative adversarial network and deep reinforcement learning. In *2021 13th International Conference on Information and Communication Technology and System (ICTS)*, pages 134–139, 2021. doi: 10.1109/ICTS52701.2021.9608634.

[58] Leo Sampaio Ferraz Ribeiro, Tu Bui, John Collomosse, and Moacir Ponti. Sketchformer: Transformer-based representation for sketched structure. In *Proc. CVPR*, 2020.

[59] Luca Rossetto, Ivan Giangreco, and Heiko Schuldt. Cineast: A multi-feature sketch-based video retrieval engine. In *2014 IEEE International Symposium on Multimedia*, pages 18–23, 2014. doi: 10.1109/ISM.2014.38.

[60] Luca Rossetto, Ivan Giangreco, and Heiko Schuldt. Cineast: A multi-feature sketch-based video retrieval engine. In *2014 IEEE International Symposium on Multimedia*, pages 18–23, 2014. doi: 10.1109/ISM.2014.38.

[61] Patsorn Sangkloy, Nathan Burnell, Cusuh Ham, and James Hays. The sketchy database:

Learning to retrieve badly drawn bunnies. 35(4), jul 2016. ISSN 0730-0301. doi: 10.1145/2897824.2925954. URL https://doi.org/10.1145/2897824.2925954.

[62] Rosália G. Schneider and Tinne Tuytelaars. Sketch classification and classification-driven analysis using fisher vectors. *ACM Trans. Graph.*, 33(6), nov 2014. ISSN 0730-0301. doi: 10.1145/2661229.2661231. URL https://doi.org/10.1145/2661229.2661231.

[63] Omar Seddati, Stéphane Dupont, and Saïd Mahmoudi. Deepsketch: Deep convolutional neural networks for sketch recognition and similarity search. In *2015 13th International Workshop on Content-Based Multimedia Indexing (CBMI)*, pages 1–6, 2015. doi: 10.1109/CBMI.2015.7153606.

[64] H.R. Sheikh, M.F. Sabir, and A.C. Bovik. A statistical evaluation of recent full reference image quality assessment algorithms. *IEEE Transactions on Image Processing*, 15(11): 3440–3451, 2006. doi: 10.1109/TIP.2006.881959.

[65] Wen Sun, Fei Zhou, and Qingmin Liao. Mdid: A multiply distorted image database for image quality assessment. *Pattern Recognition*, 61:153–168, 2017. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2016.07.033. URL https://www.sciencedirect.com/science/article/pii/S0031320316301911.

[66] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

[67] Fang Wang, Le Kang, and Yi Li. Sketch-based 3d shape retrieval using convolutional neural networks. In *2015 IEEE Conference on Computer Vision and Pattern Recogni-*

tion (CVPR), pages 1875–1883, Los Alamitos, CA, USA, jun 2015. IEEE Computer Society. doi: 10.1109/CVPR.2015.7298797. URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2015.7298797.

[68] S. Wang, D. Bau, and J. Zhu. Sketch your own gan. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14030–14040, Los Alamitos, CA, USA, oct 2021. IEEE Computer Society. doi: 10.1109/ICCV48922.2021.01379. URL https://doi.ieeecomputersociety.org/10.1109/ICCV48922.2021.01379.

[69] Ping Wei, Huan Li, and Ping Hu. Inverse discriminative networks for handwritten signature verification. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5757–5765, 2019. doi: 10.1109/CVPR.2019.00591.

[70] Karl D. D. Willis, Pradeep Kumar Jayaraman, Joseph G. Lambourne, Hang Chu, and Yewen Pu. Engineering sketch generation for computer-aided design. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2105–2114, 2021. doi: 10.1109/CVPRW53098.2021.00239.

[71] Wanghui Xiao and Di Wu. An improved siamese network model for handwritten signature verification. In *2021 IEEE International Conference on Networking, Sensing and Control (ICNSC)*, volume 1, pages 1–6, 2021. doi: 10.1109/ICNSC52481.2021.9702190.

[72] Peng Xu, Kun Liu, Tao Xiang, Timothy M. Hospedales, Zhanyu Ma, Jun Guo, and Yi-Zhe Song. Fine-grained instance-level sketch-based video retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, 31:1995–2007, 2020.

[73] Rui Xu, Zongyan Han, Le Hui, Jianjun Qian, and J. Xie. Domain disentangled generative adversarial network for zero-shot sketch-based 3d shape retrieval. In *AAAI Conference on Artificial Intelligence*, 2022.

[74] Kaihong Yan, Ying Zhang, Haoran Tang, Chengkai Ren, Jian Zhang, Gaoang Wang, and Hongwei Wang. Signature detection, restoration, and verification: A novel chinese document signature forgery detection benchmark. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 5159–5168, 2022. doi: 10.1109/CVPRW56347.2022.00564.

[75] Lan Yang, Kaiyue Pang, Honggang Zhang, and Yi-Zhe Song. Finding badly drawn bunnies. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7472–7481, 2022. doi: 10.1109/CVPR52688.2022.00733.

[76] Shuai Yang, Zhangyang Wang, and Jiaying Liu. Shape-matching gan++: Scale controllable dynamic artistic text style transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7):3807–3820, 2022. doi: 10.1109/TPAMI.2021.3055211.

[77] Jerry Yin, Chenxi Liu, Rebecca Lin, Nicholas Vining, Helge Rhodin, and Alla Sheffer. Detecting viewer-perceived intended vector sketch connectivity. *ACM Trans. Graph.*, 41(4), jul 2022. ISSN 0730-0301. doi: 10.1145/3528223.3530097. URL https://doi.org/10.1145/3528223.3530097.

[78] Qian Yu, Feng Liu, Yi-Zhe Song, Tao Xiang, Timothy M. Hospedales, and Chen Change Loy. Sketch me that shoe. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 799–807, 2016. doi: 10.1109/CVPR.2016.93.

[79] Samuel Yu, Peter Wu, Paul Liang, Ruslan Salakhutdinov, and Louis-Philippe Morency. *PACS: A Dataset for Physical Audiovisual CommonSense Reasoning*, pages 292–309. 10 2022. ISBN 978-3-031-19835-9. doi: 10.1007/978-3-031-19836-6_17.

[80] Xiangxu Yu, Neil Birkbeck, Yilin Wang, Christos G. Bampis, Balu Adsumilli, and Alan C. Bovik. Predicting the quality of compressed videos with pre-existing distortions.

IEEE Transactions on Image Processing, 30:7511–7526, 2021. doi: 10.1109/TIP.2021.3107213.

[81] Hua Zhang, Si Liu, Changqing Zhang, Wenqi Ren, Rui Wang, and Xiaochun Cao. Sketchnet: Sketch classification with web images. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1105–1113, 2016. doi: 10.1109/CVPR.2016.125.